Journal of Open Aviation Science (2026), Vol.**4**
doi:10.59490/joas.2026.8468

JOAS

PROCEEDINGS | *The 13th OpenSky Symposium*

Generative Short-Term Aircraft Trajectory Prediction with Conditional Flow Matching

Benoit Figuet [*,1,2] Timothé Krauth [1] and Steve Barry[3]

[1]Centre for Aviation, School of Engineering, Zurich University of Applied Sciences, Winterthur, Switzerland
[2]SkAI Data Services, Zurich, Switzerland
[3]Airservices Australia, Canberra, Australia
*Corresponding author: benoit.figuet@zhaw.ch

(Received: 1 Nov 2025; Revised: 20 Jan 2026; Accepted: 10 Feb 2026; Published: 17 Feb 2026)
(Editor: Xavier Olive; Reviewers: Gabriel Jarry, Richard Alligier, and Max Li)

Abstract
Reliable short-term aircraft trajectory prediction is essential for safety and efficiency in Air Traffic Management (ATM). This work introduces a generative framework for probabilistic 4D trajectory forecasting based on Conditional Flow Matching (CFM), a recent deep generative modeling approach that combines stable likelihood-based training with efficient sampling. The model is trained on historical ADS–B data from the OpenSky Network to predict aircraft motion over a 60 s horizon, conditioned on the preceding 60 s of observations. The model generates ensembles of realistic future trajectories that capture the inherent uncertainty of aircraft motion and enable probabilistic assessment of potential conflicts. As an application, we estimate the probability of mid-air collision during a loss-of-separation event using Monte Carlo simulation over the generated trajectories, providing a quantitative risk measure. The results demonstrate that flow-based generative modeling offers a principled foundation for uncertainty-aware trajectory prediction and safety analysis in ATM.

Keywords: trajectory prediction; air traffic management; conditional flow matching; generative modeling

Abbreviations: ADS–B: Automatic Dependent Surveillance–Broadcast, ANSP: Air Navigation Service Provider, ATM: Air Traffic Management, CFM: Conditional Flow Matching, CNN–LSTM: Convolutional Neural Network–Long Short-Term Memory, CV: Constant Velocity, FIR: Flight Information Region, FL: Flight Level, GAN: Generative Adversarial Network, LV95: Swiss national coordinate reference system (CH1903+/LV95), MAE: Mean Absolute Error, MAC: Mid-Air Collision, MLP: Multilayer Perceptron, MSE: Mean Squared Error, ODE: Ordinary Differential Equation, OT: Optimal Transport, PIT: Probability Integral Transform, RK: Runge–Kutta, RMSE: Root-Mean-Square Error, SiLU: Sigmoid Linear Unit, STCA: Short-Term Conflict Alert, STTP: Short-Term Trajectory Prediction, TCAS: Traffic Collision Avoidance System, VAE: Variational Autoencoder

1. Introduction

Reliable short-term aircraft trajectory prediction is fundamental to safe and efficient Air Traffic Management (ATM). Operational safety nets such as TCAS II [1] and Short-Term Conflict Alert (STCA) [2] rely on linear extrapolations to generate collision alerts. While simple and robust, such deterministic approaches cannot capture the uncertainty and variability inherent in real-world trajectories.

Effective short-term trajectory prediction (STTP) algorithms have immediate benefit to Air Navigation Service Providers (ANSPs) and regulators. A central task for ANSPs is assessing risk for

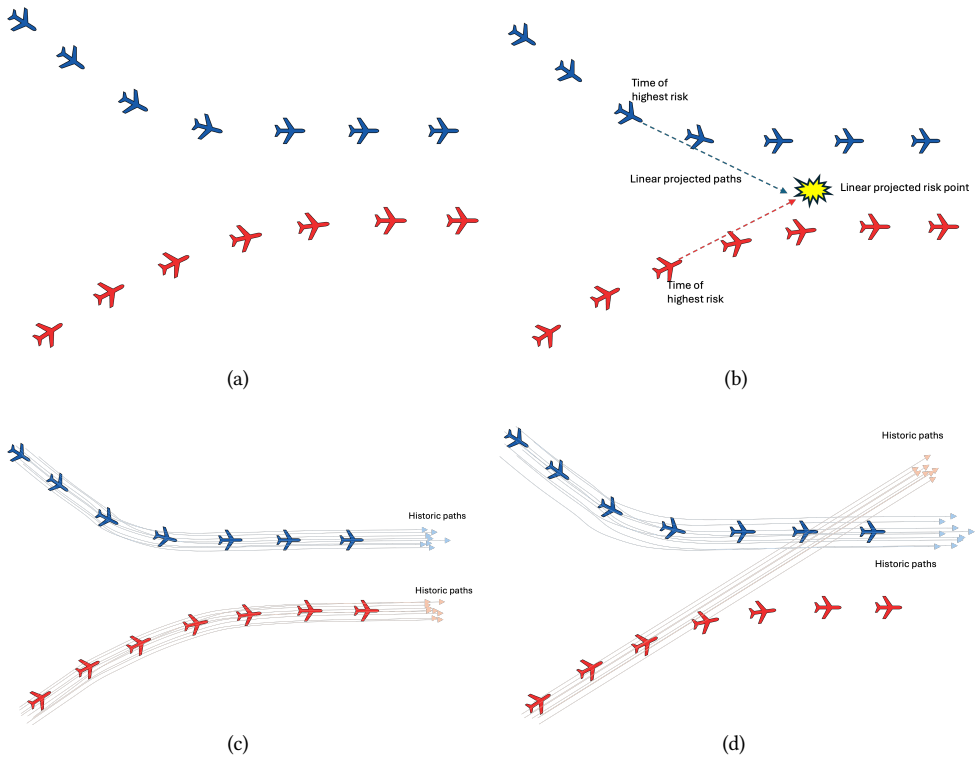

Figure 1. Schematic illustration of conflict evaluation for short-term trajectory prediction. (**a**) A potential conflict between two aircraft is detected from surveillance tracks. (**b**) Linear extrapolation of the current trajectories indicates a possible collision at the projected point of highest risk. (**c**) Historical trajectories reveal that the aircraft were expected to turn as part of a standard procedure, suggesting a false positive. (**d**) Historical trajectories reveal that the red aircraft was expected to continue straight, suggesting a true positive. The history of trajectories provides probabilistic evidence of intent, enabling data-driven classification of real versus false conflicts.

numerous airspace occurrences, such as a loss of separation (LOS) or TCAS events, as well as thousands of conflicts detected by data mining all surveillance tracks. An essential component of this assessment is determining whether each detected conflict is real or a false positive, for instance in cases where aircraft were expected to turn as part of a published procedure before any potential collision.

As illustrated schematically in Figure 1, linear extrapolation can indicate a high-risk situation if no deviation occurs, yet it is often unclear whether the aircraft had intended to turn as part of its standard path. Historical trajectories can reveal whether an aircraft was following an established procedure or deviating from it, thus determining whether the risk was genuine or merely apparent. In practice, this distinction is rarely binary: large-scale surveillance data exhibit significant variability, and data-driven models are needed to capture the range of plausible futures consistent with observed intent.

In recent years, research has increasingly explored data-driven prediction methods to address these challenges. Liu and Hansen [3] proposed a deep generative convolutional recurrent network for multimodal trajectory prediction, while Krauth et al.[4] introduced multivariate density models to synthesize realistic aircraft trajectories. Jarry et al. [5] employed a Generative Adversarial Network (GAN) to learn the probability distributions of real aircraft approach paths, enabling the genera-

tion of realistic trajectories and the detection of atypical flight behaviors. Zeng et al. [6] provide a comprehensive review of trajectory prediction techniques, emphasizing both progress and remaining challenges. Despite these advances, most models still predict a single deterministic trajectory, making uncertainty quantification difficult.

To address this limitation, Krauth et al. [7] recently proposed a multi-objective CNN–LSTM architecture that predicts not only the expected trajectory but also spatio-temporal confidence areas, enabling the construction of 95% prediction intervals for each state component. These developments highlight a growing recognition that uncertainty-aware prediction is essential for robust ATM applications.

Building on these efforts, this paper introduces a generative framework for probabilistic short-term trajectory forecasting based on Conditional Flow Matching (CFM) [8], which learns to transform random noise into plausible trajectories via ordinary differential equations (ODEs). Given one minute of observation, a Transformer-based conditional flow estimates the distribution of the trajectory for the next minute conditioned on observed inputs. The one-minute prediction horizon is chosen to align with the operational timescales of airborne safety nets such as TCAS whose alerting logic typically operates within a 30–45 s look-ahead window [1]. In practice, the proposed framework is not limited to this horizon and can be extended to longer prediction intervals as required by specific Air Traffic Management applications.

This formulation allows the generation of multiple plausible future trajectories that capture the stochastic nature of real aircraft motion, offering a data-driven means to assess both real and false conflicts within a probabilistic risk-assessment framework. We favor CFM over GANs [9], VAEs [10], or standard diffusion models [11] because it affords explicit likelihood training and hence naturally calibrated uncertainties, uses a stable regression-based objective that avoids the adversarial instabilities of GANs and the heavy noise-schedule simulation burden of diffusion models, and enables faster inference via simpler flow paths with fewer integration steps [12].

2. Background: Conditional Flow Matching

Flow Matching (FM) is a framework for training generative models via continuous flows [8, 13]. The key idea is to describe the transformation from a simple base distribution (e.g., Gaussian noise) to a complex data distribution (e.g., aircraft trajectories) as the solution of an ordinary differential equation (ODE) driven by a time-dependent vector field v_t. A flow ϕ_t, defined as the solution to this ODE, maps samples from the prior to the data space. Flow Matching provides a simulation-free method to learn this vector field by regressing it against a target vector field u_t that generates a desired probability path $\{p_t\}_{t\in[0,1]}$ connecting a prior distribution p_0 to a target data distribution p_1.

This section summarizes the Flow Matching and Conditional Flow Matching results introduced by Lipman et al. [8] and further developed in subsequent lecture notes [14].

2.1 Flow Matching

Let $p(x)$ be a simple, tractable prior distribution (e.g., a standard normal $\mathcal{N}(x|0, I)$) and let $q(x_1)$ be the target data distribution from which we can draw samples. We consider a probability path p_t such that $p_0 = p$ and p_1 approximates q. This path is generated by an unknown target vector field u_t. The goal is to train a neural network $v_t(x;\theta)$ to approximate u_t.

The Flow Matching (FM) objective is a regression loss defined as:

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{t\sim U[0,1],\, x\sim p_t(x)} \left\| v_t(x;\theta) - u_t(x) \right\|^2 . \tag{1}$$

Here $\|\cdot\|$ denotes the Euclidean (ℓ_2) norm. Minimizing this objective forces the learned vector field v_t to match the target field u_t. At inference, we can generate new samples by solving the initial value problem $\frac{d}{dt}X_t = v_t(X_t;\theta)$ from $t = 0$ to $t = 1$, with $X_0 \sim p(x)$. However, this objective is intractable because both the marginal path $p_t(x)$ and its vector field $u_t(x)$ are generally unknown.

2.2 Conditional Flow Matching

CFM reformulates the problem to be solvable in practice. The core idea is to construct the intractable marginal path $p_t(x)$ by marginalizing over a set of simpler, per-sample conditional probability paths $p_t(x|x_1)$:

$$p_t(x) = \int p_t(x|x_1)q(x_1)dx_1.$$

Each conditional path is designed to start from the prior at $t = 0$ (i.e., $p_0(x|x_1) = p(x)$) and end in a distribution concentrated around a specific data sample x_1 at $t = 1$. The corresponding marginal vector field $u_t(x)$ can also be expressed as an aggregation of the conditional vector fields $u_t(x|x_1)$.

The key insight of CFM is that the gradients of the intractable FM objective (1) are identical to the gradients of a much simpler objective that uses the conditional paths directly. The CFM objective is:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t\sim U[0,1],\, x_1\sim q(x_1),\, x\sim p_t(x|x_1)} \left\| v_t(x;\theta) - u_t(x|x_1)\right\|^2. \tag{2}$$

This loss is tractable because sampling from $p_t(x|x_1)$ and evaluating its vector field $u_t(x|x_1)$ can be done in closed form for well-chosen conditional paths.

2.3 Gaussian and Optimal Transport Paths

A general and effective choice for the conditional paths are Gaussian paths of the form:

$$p_t(x|x_1) = \mathcal{N}\big(x|\mu_t(x_1), \sigma_t(x_1)^2 I\big),$$

which define a smooth probability path from the base distribution at t=0 (typically standard normal noise) to a distribution concentrated around a particular data example x_1 at t=1. Intuitively, μ_t controls the drift toward x_1 while σ_t controls how quickly uncertainty is removed along the path. where the time-dependent mean $\mu_t(x_1)$ and standard deviation $\sigma_t(x_1)$ satisfy the boundary conditions $\mu_0(x_1) = 0, \sigma_0(x_1) = 1$ and $\mu_1(x_1) = x_1, \sigma_1(x_1) = \sigma_{\min}$, with $\sigma_{\min}$ being a small positive constant. In the remainder of this paper, we use the common simplifying choice $\sigma_{\min} = 0$ (deterministic endpoint at t=1), which yields the linear interpolation in Eq. (5). The vector field that generates this path is given by:

$$u_t(x|x_1) = \frac{\sigma_t'(x_1)}{\sigma_t(x_1)}\big(x - \mu_t(x_1)\big) + \mu_t'(x_1). \tag{3}$$

Here, primes denote derivatives with respect to the scalar flow-time t (holding x_1 fixed).

A particularly powerful instance uses linear schedules for the mean and standard deviation, which corresponds to the Optimal Transport (OT) displacement interpolant between the Gaussians at $t = 0$ and $t = 1$. Setting

$$\mu_t(x_1) = t\,x_1 \quad \text{and} \quad \sigma_t(x_1) = 1 - (1 - \sigma_{\min})\,t,$$

the target vector field in Equation (3) simplifies to:

$$u_t(x|x_1) = \frac{x_1 - (1 - \sigma_{\min})x}{1 - (1 - \sigma_{\min})t}.$$

This vector field has a direction that is constant over time, making it simpler for a neural network to learn. The resulting paths move along straight line trajectories from noise to data, leading to more efficient training and sampling.

2.4 CFM in Practice:

We train a vector-field network $v_t(x;\theta)$ (optionally $v_t(x,c;\theta)$ with context c) that predicts the instantaneous velocity of a sample x along a probability path from a simple prior p_0 to the data distribution. Conceptually, v_t replaces the unknown target field u_t and encodes "how to move" data at each time $t \in [0,1]$.

Learning is pure regression: minimize the Conditional Flow Matching loss in (2), which is an MSE between the network and a closed-form target field $u_t(x|x_1)$ defined by your chosen conditional path (e.g., the Gaussian/OT path of (3)). This objective provides unbiased gradients for the intractable FM loss and requires no likelihoods, scores, or simulation of trajectories during training.

At each iteration, draw a random time t, a data example x_1, and a synthetic point $x \sim p_t(x \mid x_1)$ from the conditional path; compute the analytic target $u_t(x|x_1)$; regress $v_t(x;\theta)$ toward it with MSE. Repeat over mini-batches with your optimizer of choice.

After training, generate by integrating the learned ODE $\frac{d}{dt}X_t = v_t(X_t;\theta)$ from $t = 0$ to $t = 1$ starting at $X_0 \sim p_0$ (e.g., standard normal). Any standard ODE solver (Euler/Heun/RK) with a modest number of steps suffices; X_1 is the synthesized sample.

3. Methodology

3.1 Data and Preprocessing

We use one month of ADS–B surveillance data from the OpenSky Network [15], restricted to flights above FL195 within the Swiss Free Route Airspace (FRA) and collected with the `traffic` library [16]. All trajectories are resampled at 1 Hz.

From ADS–B state vectors, we derive a consistent kinematic representation. Latitude and longitude are projected to the Swiss projected grid (CH1903+/LV95; EPSG:2056), yielding planar coordinates (x, y) with x Easting and y Northing; altitude is converted to meters (z). Groundspeed v and track angle θ (clockwise from North) define horizontal velocity components ($v_x = v\sin\theta,\ v_y = v\cos\theta$), while the vertical rate provides v_z (ft/min converted to m/s). A turn-rate proxy $\dot{\psi}$ is computed from the unwrapped angular change between successive horizontal velocity vectors, divided by the sampling interval (1 s), and clipped to ±0.25 rad/s to suppress outliers. Each trajectory point in the global frame is thus

$$(x,\ y,\ z,\ v_x,\ v_y,\ v_z,\ \dot{\psi}),$$

a 7-dimensional state encoding position and motion.

Examples are constructed as sliding windows across flights. Each sample comprises 60 s of history sampled at 1 Hz and a 60 s prediction horizon; futures are down-sampled every 5 s, yielding 12 targets per window. Splits are performed at flight level to eliminate leakage between train, validation, and test sets. The training set contains 1,000,000 input–output pairs, while the validation and test sets each contain 200,000 samples.

To ensure exposure to maneuvering behavior, at least 30% of the training and validation samples contain a turn, defined as ≥ 3 consecutive steps with $\dot{\psi} > 0.01$ rad/s occurring in the history and/or future portion of the window; remaining samples are drawn uniformly to preserve overall traffic statistics. The test set is sampled uniformly without turn constraints.

To reduce variance and improve generalization, we transform each window into an aircraft-centric frame that is fixed by the last observed state. The last observed position defines the origin, and the last horizontal velocity vector defines the forward axis; the frame does not rotate over the prediction

horizon. We denote aircraft-centric quantities with tildes. The per-timestep input sequence is

$$(\,\tilde{x},\ \tilde{y},\ \tilde{z},\ \tilde{v}_x,\ \tilde{v}_y,\ \tilde{v}_z,\ \dot{\psi}\,),$$

where $(\tilde{x}, \tilde{y}, \tilde{z})$ and $(\tilde{v}_x, \tilde{v}_y, \tilde{v}_z)$ are positions and velocities expressed in this fixed local frame, while the turn rate $\dot{\psi}$ is unchanged.

In addition, we provide an 8-dimensional context vector that captures the absolute reference state at the history endpoint:

$$\left(\,x_{\text{abs}},\ y_{\text{abs}},\ z_{\text{abs}},\ \cos\theta,\ \sin\theta,\ v_{\text{gs,last}},\ v_{z,\text{last}},\ \dot{\psi}_{\text{last}}\,\right),$$

where $(x_{\text{abs}}, y_{\text{abs}}, z_{\text{abs}})$ are absolute LV95 coordinates (m), $(\cos\theta, \sin\theta)$ encode the track angle, $v_{\text{gs,last}}$ is ground speed, $v_{z,\text{last}}$ is vertical speed, and $\dot{\psi}_{\text{last}}$ is the final turn rate. Thus, the model ingests (i) a 7-D aircraft-centric trajectory sequence capturing local dynamics and (ii) an 8-D global context anchoring the sequence in absolute space and orientation.

Both the sequence and the context are standardized using their own mean and variance, estimated on the training set and applied to all splits.

3.2 Model Architecture

Our predictor is a Transformer encoder–decoder that learns to map an observed flight history to a distribution of future trajectories. The model operates on three types of input:

1. **History sequence:** the last 60 s of aircraft motion (7 features per timestep).
2. **Context vector:** an 8-D descriptor of the aircraft's absolute position and orientation at the end of the history.
3. **Noisy future:** a sequence of future states during training (interpolated between Gaussian noise and target trajectory); during inference, this starts as pure Gaussian noise and gets progressively denoised to generate predictions.

History encoder: The 60 history steps (7 features each) are first linearly projected to 512 dimensions and enriched with positional encodings. The 8-D global context vector is mapped to 512 dimensions and prepended as an extra token at the front of the sequence, so that the Transformer can jointly attend to context and history (akin to a [CLS] token [17]). This combined sequence of 61 tokens (each 512-D) is processed by six Transformer encoder layers, producing a latent representation of the past trajectory that serves as memory for the decoder.

Time embedding: The flow-matching process depends on a scalar time variable $t \in [0, 1]$, which indicates how far we are between pure noise ($t = 0$) and the true future ($t = 1$). To make this information usable by the Transformer, t is first expanded into sinusoidal features using 64 frequencies (yielding 128 features: sine and cosine). These are passed through a small MLP: a fully connected layer maps 128 inputs to 256 hidden units with SiLU activation, followed by a second fully connected layer mapping 256 to 512 units. The resulting 512-D time embedding is added to both the noisy future tokens and to the encoded history, so the model always knows "when" in the flow it is operating.

Future denoiser: The noisy future sequence is projected into the latent space and processed by an eight-layer Transformer decoder with self-attention across future tokens and cross-attention to

the encoded history. Finally, a linear layer maps the decoder output back to 7 physical features per step, representing the predicted vector field

$$v_\theta(x_t, t \mid \text{history}, \text{context}) \in \mathbb{R}^{12\times 7}.$$

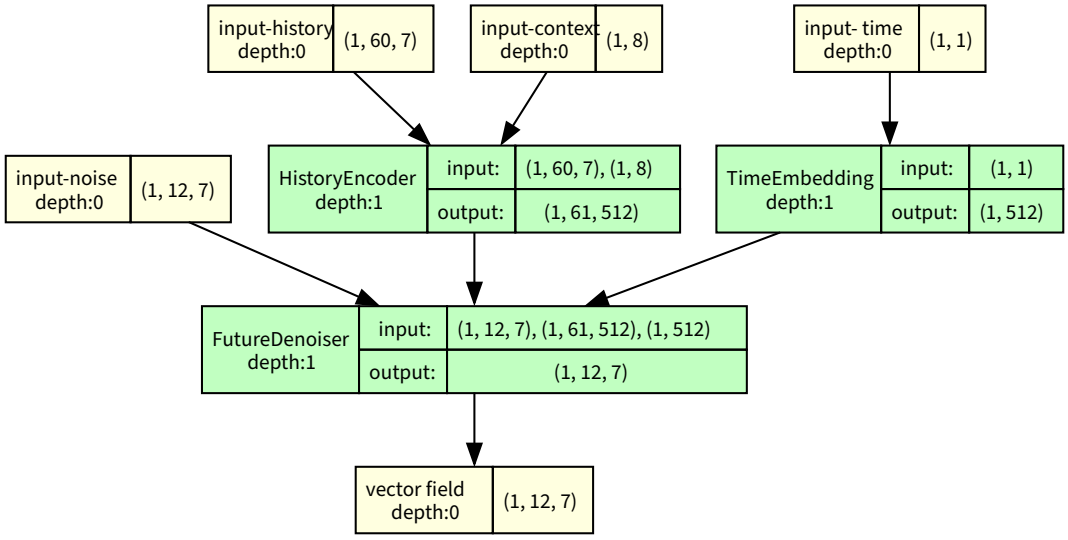

Figure 2. Architecture of the flow-matching Transformer model. The history encoder processes past trajectory data and context, the time embedding provides temporal conditioning, and the future denoiser generates denoised predictions.

In essence, the encoder compresses the past 60 s of motion into a latent memory, the time embedding guides how noise is transformed along the flow, and the decoder denoises 12 future steps conditioned on both history and context. The complete architecture is illustrated in Figure 2.

3.3 Training and Inference

Let the normalized future be $x_1 \in \mathbb{R}^{K\times 7}$ and partition the channel dimension as $x_1 = (x_1^{\text{pos}}, x_1^{\text{vel}}, x_1^{\psi})$ with shapes $K\times 3$, $K\times 3$, and $K\times 1$, respectively, where K denotes the number of time steps predicted.

We instantiate the OT-style Gaussian conditional path of Section 2.3 with $\sigma_{\min} = 0$. Sample $\varepsilon \sim \mathcal{N}(0, I)$ and $t \sim U[0, 1]$, and form

$$x_t = (1 - t)\,\varepsilon + t\,x_1, \tag{4}$$
$$x_t \mid x_1 \sim \mathcal{N}\big(t\,x_1,\ (1 - t)^2 I\big),$$

and define the conditional target vector field

$$u = u_t(x_t \mid x_1) = \frac{x_1 - x_t}{1 - t} = x_1 - \varepsilon. \tag{5}$$

This corresponds to the OT-style Gaussian path with $\sigma_{\min} = 0$ (so x_t is a convex combination of noise and data), for which $u_t(x_t \mid x_1)$ is constant in t.

The network outputs $v_\theta(x_t, t)$ and we minimize a weighted MSE (i.e., $\mathbb{E}_{t,\varepsilon}\,\|v_\theta - u\|^2$) expressed in target-space components:

$$\hat{x}_1 = \varepsilon + v_\theta(x_t, t), \tag{6}$$
$$\mathcal{L} = \lambda_{\text{pos}}\,\|\hat{x}_1^{\text{pos}} - x_1^{\text{pos}}\|^2 + \lambda_{\text{vel}}\,\|\hat{x}_1^{\text{vel}} - x_1^{\text{vel}}\|^2 + \lambda_{\psi}\,\|\hat{x}_1^{\psi} - x_1^{\psi}\|^2, \tag{7}$$

with λ_{pos}=1.0, λ_{vel}=0.5, and λ_{ψ}=0.05. These weights were chosen empirically to balance the different scales and importance of position, velocity, and turn-rate errors during training. Here $\|\cdot\|^2$ denotes the sum of squared errors over tokens and channels.

We train with AdamW, a warmup+cosine learning-rate schedule, dropout 0.1, and maintain an exponential moving average (EMA) of parameters for evaluation and checkpointing.

At test time, we sample from the learned flow by initializing $x_{t=0} \sim \mathcal{N}(0, I)$ and integrating the ODE

$$\frac{dx_t}{dt} = v_\theta(x_t, t \,|\, \text{history}, \text{context}), \tag{8}$$

from t=0 to t=1 using a predictor–corrector scheme (Heun's method; an explicit trapezoidal / second-order Runge–Kutta integrator) with a fixed number of steps. This yields a 12-token aircraft-centric future. Samples are then denormalized and mapped back to the global frame by (i) inverse-rotating $(\tilde{x}, \tilde{y})$ and $(\tilde{v}_x, \tilde{v}_y)$ using $(\cos\theta, \sin\theta)$ from the context (fixed frame), and (ii) translating by the absolute reference $(x_{abs}, y_{abs}, z_{abs})$. Repeating the sampling procedure produces ensembles of plausible 60 s futures conditioned on the observed 60 s history.

3.4 Accuracy and Probabilistic Calibration

We evaluate both point accuracy and probabilistic calibration of the proposed CFM forecaster on held-out test windows. Unless noted otherwise, we report results over N windows (default N=512) with K forecast steps (default K=12, i.e. 5 s stride over a 60 s horizon).

Given a history H, a context c, and the learned vector field $v_\theta(\cdot, t \mid H, c)$, we draw S forecast samples by integrating the ODE with S distinct initial noises in the aircraft-centric normalized frame. Each trajectory is then denormalized and mapped back to the global LV95 frame using the inverse of the per-window normalization and the fixed-frame transformation. We denote global positions by $\hat{y}_\tau^{(s)} \in \mathbb{R}^3$ (Easting, Northing, altitude) and ground-truth by $y_\tau^\star$ for τ=1, …, K.

3.4.1 Deterministic Accuracy vs. Horizon

We evaluate geometric prediction error at each forecast horizon τ (in seconds) using two metrics: mean absolute error (MAE) and root mean square error (RMSE). We compare three predictors:

1. Model (mean): Ensemble mean $\bar{y}_\tau = \frac{1}{S}\sum_{s=1}^{S} \hat{y}_\tau^{(s)}$.
2. Model (best-of-S): A diagnostic lower bound selecting the sample closest to ground truth:
$$y_\tau^{\text{best}} = \arg\min_{s \le S} \left\| \hat{y}_\tau^{(s)} - y_\tau^\star \right\|_2,$$
which probes ensemble coverage.
3. Constant-velocity baseline: Linear extrapolation in global coordinates using the last observed ground and vertical speeds.

For any predictor $\mathbf{y}_\tau^{(\cdot)}$, errors over N test windows are

$$\text{MAE}_\tau = \frac{1}{N}\sum_{n=1}^{N} \left\| y_\tau^{(\cdot,n)} - y_\tau^{\star(n)} \right\|_2, \tag{9}$$

$$\text{RMSE}_\tau = \sqrt{\frac{1}{N}\sum_{n=1}^{N} \left\| y_\tau^{(\cdot,n)} - y_\tau^{\star(n)} \right\|_2^2}. \tag{10}$$

We report MAE_τ and RMSE_τ for all three predictors as functions of the horizon $\tau = \{5, 10, \ldots, 60\}$ s (default $\Delta\tau$=5 s).

3.4.2 Probabilistic Calibration Diagnostics

To assess whether the model's forecast distribution matches empirical frequencies, we use a Probability Integral Transform (PIT) diagnostic. For each coordinate $d \in \{\tilde{x}, \tilde{y}, z\}$ and each (n, τ) (forecast step), we compute a sample-based PIT value using the empirical rank:

$$\mathrm{PIT}_{n,\tau,d} = \frac{1}{S+1}\left(\sum_{s=1}^{S} \mathbb{I}\left\{\hat{y}_{\tau,d}^{(s,n)} \le y_{\tau,d}^{\star(n)}\right\} + U_{n,\tau,d}\right), \qquad U_{n,\tau,d} \sim U[0,1],$$

which avoids degenerate values under ties in finite ensembles. For a calibrated univariate predictive distribution, PIT should be uniformly distributed on $[0, 1]$. We therefore aggregate $\mathrm{PIT}_{n,\tau,d}$ over n and τ and report axis-wise histograms; deviations from uniformity indicate under/over-dispersion or bias.

3.5 Application to a real-world conflict

To illustrate the operational use of the proposed approach, we analyze a real encounter extracted from ADS–B data (Figure 3). In this event, one aircraft was maintaining level flight at FL310 while the other was descending through FL310 on a converging path. The recorded data show that the minimum horizontal and vertical spacings fell below the prescribed separation minima (<5 nautical miles horizontal and <1,000 feet vertical), resulting in an actual LOS.

Figure 3. A pair of aircraft trajectories ending in a loss of separation.

Based on the observed histories of both aircraft, we generated S stochastic future trajectories for each one using the CFM model. Every possible combination of one sampled future from each aircraft was resampled from 0.2 Hz (5 s resolution) to 1 Hz by linear interpolation and then examined to determine whether standard separation minima were breached or a MAC observed at any point within the prediction horizon.

Throughout the forecast, we monitored the horizontal and vertical spacing between the two aircraft. A situation was classified as a LOS when, at any moment, the horizontal distance between the aircraft fell below the LOS limits. To capture more critical encounters, we also defined a *mid-air collision* (MAC) proxy, corresponding to predicted cases where the aircraft approached closer than 0.03 nautical miles horizontally and 55 feet vertically.

By counting the proportion of trajectory pairs that met either of these criteria, we obtained straightforward Monte Carlo estimates of the probabilities of a future LOS or MAC within the prediction window.

4. Results

4.1 Ensemble Forecasts on Representative Cases

Figure 4 shows ensemble forecasts for three representative flights. Each panel displays the 60 s observed history (black), the 60 s ground-truth future (red), 128 sampled futures from the CFM model (blue), and the ensemble mean trajectory (yellow).

In the left panel, most samples follow a curved path while some continue straight. In the middle panel, all samples form a narrow bundle along the observed flight direction. In the right panel, the true continuation is straight, and several samples deviate slightly toward a right-hand branch. Across the three examples, the ensemble spread increases with prediction horizon, and the ensemble mean remains near the center of the sampled futures.

Figure 4. Ensemble ("spaghetti") forecasts for three representative test flights. Black: observed history; red: ground-truth future; blue: 128 sampled futures; yellow: ensemble mean.

4.2 Flow Evolution and Vector Field Visualization

Figure 5 illustrates the temporal evolution of the learned conditional flow for one example case. The three panels correspond to integration times t=0 (noise), t=0.5 (intermediate state), and t=1 (final prediction). Each map shows predicted positions (blue), sample vectors (orange), and the grid vector field (purple), together with the observed history (black) and ground-truth future (red).

The orange *sample vectors* visualize the model's denoising dynamics in *flow time* t: they are finite-difference displacements of the predicted future tokens between two consecutive ODE integration steps (i.e., $\Delta\hat{y}/\Delta t$ in the (x, y) plane), and should not be interpreted as physical aircraft velocities in trajectory time τ. The purple *grid vector field* is obtained by evaluating the learned vector field $v_\theta(\cdot, t \mid H, c)$ on a spatial grid for a single synthetic future token (position channels set from the grid point, remaining channels set to zero), yielding a qualitative 2D slice of the full 12×7 vector field.

At t=0, sample vectors are randomly oriented. At t=0.5, the flow begins to align spatially along the future path. At t=1, the trajectories form a coherent pattern that overlaps with the true continuation. The grid vector field exhibits smooth directional changes between neighboring locations.

4.3 Forecast Error vs. Horizon

Figure 6 presents mean absolute error (MAE) and root-mean-square error (RMSE) as a function of prediction horizon. Metrics are computed over 512 test windows (prediction problems) for 3D Euclidean (x, y, z), horizontal (x, y), and vertical (z) components. Each plot compares three estimators: ensemble mean, best-of-S sample, and constant-velocity extrapolation (CV).

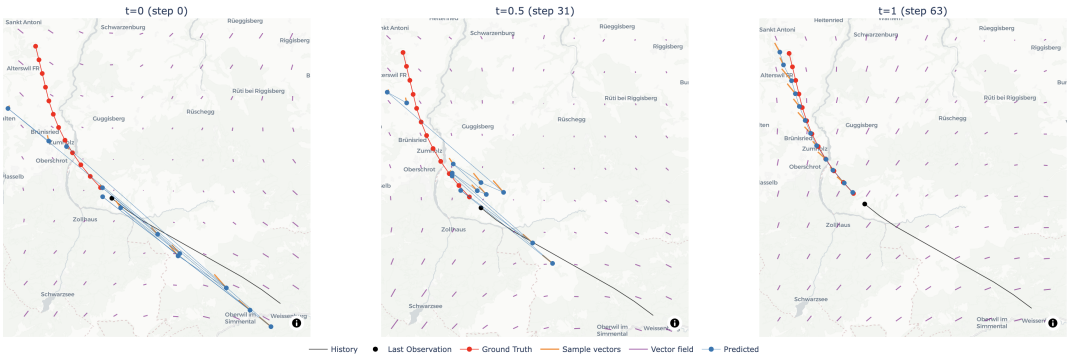

Figure 5. Evolution of the learned conditional flow for a single test case. Each map shows predicted positions (blue), sample vectors (orange), and the grid vector field (purple) at three integration times (t=0, t=0.5, t=1). Black: observed history; red: ground-truth future.

For all spatial components, errors increase monotonically with horizon. 3D and horizontal errors show similar growth patterns, while vertical errors remain smaller in magnitude.

The CV baseline yields consistently larger errors for both MAE and RMSE. Typically, over a 60 s horizon, the CFM model achieves a 3D MAE of 220 m, compared with about 320 m for the CV baseline. For RMSE, the CFM model reaches approximately 500 m, whereas the CV baseline remains just below 800 m.

The best-of-S curve remains consistently below the other curves across all horizons, with both MAE and RMSE staying below 50 m throughout the prediction window.

4.4 Probabilistic Calibration

Figure 7 shows the PIT histograms for the aircraft-centric longitudinal and lateral coordinates $(\tilde{x}, \tilde{y})$ and for altitude z. Both $\tilde{x}$ and $\tilde{y}$ exhibit a central peak around 0.5 with lighter tails, indicating over-dispersion in the horizontal plane; the effect is more pronounced for the lateral component $\tilde{y}$. This behavior is expected at short horizons because lateral motion is driven more strongly by turning intent than longitudinal motion, making it harder to infer from recent history alone. The z component is closer to uniform, suggesting better calibration in the vertical dimension.

4.5 Results on the real-world conflict

Using the ADS–B histories of the two aircraft, we generated $S = 100$ stochastic futures for each trajectory and evaluated all 100×100 combinations of predicted paths. Among all paired samples, $n_{\mathrm{LOS}} = 8,345$ combinations (83%) resulted in a predicted LOS, while $n_{\mathrm{COL}} = 36$ (0.36%) met the stricter MAC threshold. These results indicate that the model assigns a realistic, non-negligible probability to a future LOS, consistent with the outcome observed in the actual flight data. The corresponding 95% Clopper–Pearson confidence interval for the collision probability is approximately [0.25%, 0.50%].

5. Discussion

Given one minute of observed flight history, the proposed CFM model can generate multiple plausible trajectories for the following minute. Each sample represents a distinct but realistic continuation of the aircraft's motion, allowing the forecast to capture both the expected evolution and the uncertainty surrounding it. This ensemble property distinguishes the approach from deterministic predictors: instead of committing to a single extrapolated path, it provides a distribution of possible

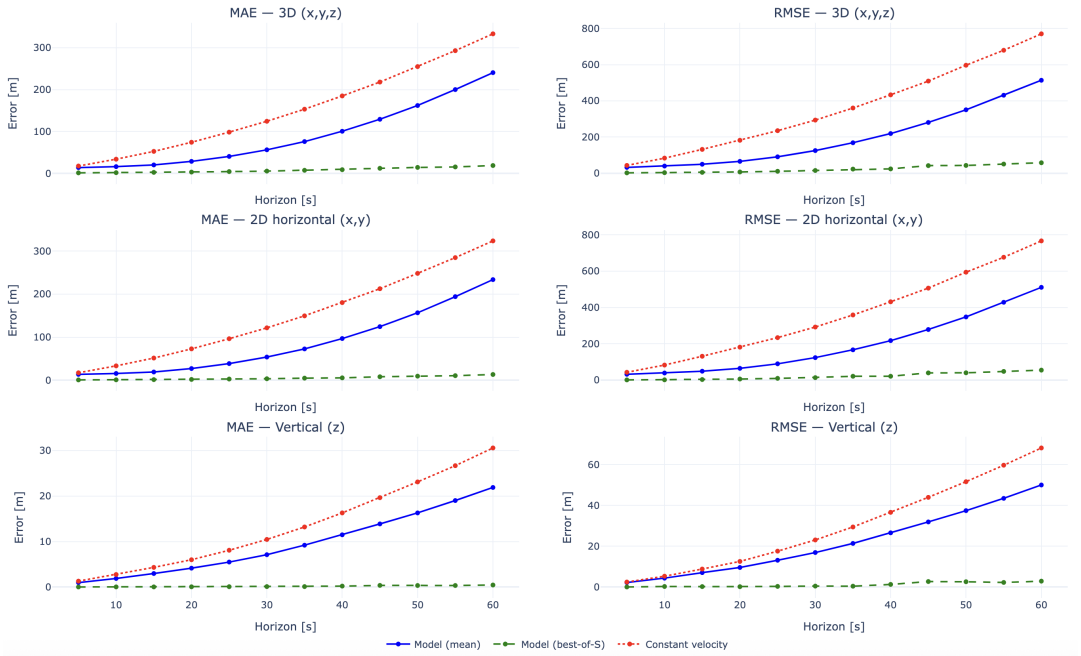

Figure 6. Mean absolute error (left) and root-mean-square error (right) versus prediction horizon, computed over 512 test samples. Rows correspond to 3D Euclidean, horizontal, and vertical components. Blue: model mean; green: best-of-S; red: constant-velocity baseline.

futures consistent with recent behavior.

The ensemble samples reflect context-dependent variability in aircraft motion: tight, low-spread ensembles emerge during stable, steady flight, while wider and occasionally multi-modal spreads appear in maneuvering phases such as turns or climbs. This adaptive spread indicates that the model has learned to represent uncertainty in a meaningful way. When motion is predictable, the ensemble converges; when intent is ambiguous, the model expresses multiple likely continuations.

The flow visualization indicates that the model learns a consistent vector field that continuously transforms random noise into structured trajectories.

Quantitatively, the CFM predictor consistently achieves lower MAE and RMSE than constant-velocity extrapolation, reducing 3D RMSE by roughly 40% at 60 s horizons. Vertical predictions are particularly accurate, reflecting the slower dynamics of en-route flight. The best-of-S results confirm that the true trajectory is typically contained within the ensemble, suggesting that the generated variability captures the range of realistic futures.

We compare against constant-velocity extrapolation because it matches the linear-motion assumptions commonly used in short-term safety nets and in practical risk modeling. More advanced physics-based or learning-based baselines would be valuable, but are left for future work.

The PIT analysis shows that the CFM forecasts are over-dispersed, especially in the horizontal (x, y) components. The central peak and light tails in their PIT histograms indicate that ensemble spreads are wider than the true variability in the test data. In contrast, the z component is better calibrated, showing a more uniform distribution. Incorporating additional contextual features, such as flight plans or intent information, could help reduce over-dispersion and improve calibration. In

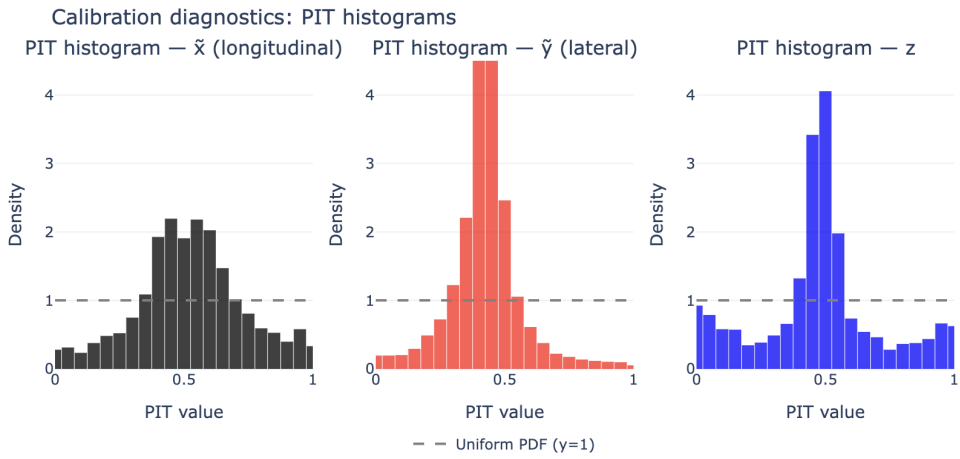

Figure 7. Probability Integral Transform (PIT) histograms for aircraft-centric $\tilde{x}$ (longitudinal), $\tilde{y}$ (lateral), and z components, aggregated across 512 test windows with 256 samples each.

operational settings, horizontal over-dispersion may be conservative for safety, but may also inflate uncertainty volumes and increase nuisance alerts; this motivates further work on calibration. In the aircraft-centric PIT analysis, the over-dispersion is more pronounced laterally ($\tilde{y}$) than longitudinally ($\tilde{x}$), consistent with the stronger influence of turning intent on short-horizon lateral motion.

The real-world encounter case study further highlights the operational relevance of the approach. When applied to a pair of aircraft that actually experienced a LOS, the model predicted an LOS in approximately 83% of all sampled trajectory pairs and a MAC in 0.36%. By representing uncertainty through ensembles rather than deterministic paths, the model enables direct estimation of the likelihood and severity of potential conflicts—offering a data-driven complement to existing risk-assessment methods. Nevertheless, accurate quantitative risk evaluation still depends on good probabilistic calibration.

Despite these promising results, several limitations remain. A small fraction of generated samples show unrealistic oscillations or curvature, indicating that the learned flow occasionally violates physical motion constraints. Incorporating kinematic regularization or lightweight flight-dynamics priors could mitigate this issue. The fixed 60-second prediction horizon, although operationally meaningful for safety-net applications, also constrains performance and should be adapted to the intended use case. Finally, the model currently relies solely on motion-derived ADS-B features; integrating contextual data such as flight plans, weather fields, or nearby traffic would likely improve both accuracy and calibration.

6. Conclusion and Outlook

This study demonstrates that Conditional Flow Matching (CFM) provides an effective generative formulation for short-term, uncertainty-aware aircraft trajectory prediction. Given one minute of observed motion, the model learns a continuous vector field that transforms stochastic perturbations into future trajectories over the following minute. The resulting ensembles capture both the expected continuation of flight and the uncertainty associated with short-term intent, producing forecasts that are more accurate and informative than conventional constant-velocity extrapolation.

Beyond predictive accuracy, the ensemble formulation of the CFM model offers a direct route from

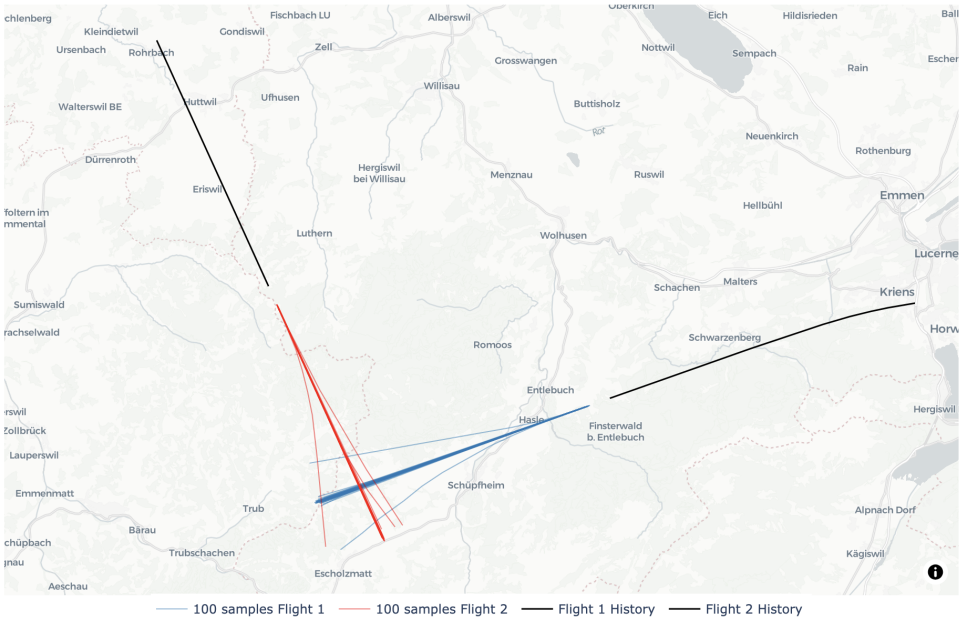

Figure 8. Ensemble forecasts for the conflicting aircraft pair. Blue and red lines correspond to sampled futures for each trajectory, while the black lines correspond to the last minute of observation for each flight.

probabilistic forecasting to operational decision support. By representing future motion as a distribution rather than a single trajectory, it becomes possible to compute interpretable risk measures such as the probability of loss of separation or mid-air collision. In the presented case study, these probabilities aligned closely with the actual outcome, demonstrating that the model can provide early and quantitative evidence of potential conflicts. Such probabilistic indicators could complement existing safety nets by replacing binary thresholding with continuous risk levels, supporting more nuanced prioritisation and review of air traffic situations. To ensure operational reliability, however, ensemble calibration must be demonstrated.

In the longer term, models of this kind could be used to complement existing safety nets such as the STCA or the Airborne Collision Avoidance System (ACAS). By producing probabilistic forecasts that explicitly quantify future risk, they could provide an additional layer of context to existing deterministic alerting systems, helping distinguish genuine conflicts from expected manoeuvres. Equally, the same generative framework can be applied retrospectively for the forensic analysis of historical encounters, allowing quantitative reconstruction of uncertainty and intent in recorded loss-of-separation or near-miss events.

From a modeling perspective, several avenues of research emerge. First, the physical fidelity of generated trajectories can be improved by incorporating lightweight kinematic regularization terms or flight-dynamics priors to suppress oscillatory samples while preserving diversity. Second, extending the conditioning inputs beyond motion-derived ADS–B features to include flight plans, meteorological conditions, or surrounding traffic is likely to enhance intent inference and reduce lateral over-dispersion. Third, the expansion of the prediction to longer horizons to increase the usability of the model.

In summary, CFM provides a principled foundation for probabilistic trajectory forecasting in Air Traffic Management. It unifies deterministic accuracy, calibrated uncertainty, and interpretability within a single generative framework, offering tangible benefits for both safety analysis and decision

support. While further work on calibration, dynamics regularization, and contextual conditioning remains, the presented results suggest that flow-based generative modeling represents a promising and operationally relevant step toward uncertainty-aware prediction and risk estimation in next-generation air traffic management systems.

Author contributions

- First Author: Conceptualization, Data Curation, Formal Analysis, Investigation, Methodology, Software, Validation, Visualization, Writing (Original Draft), Writing (Review and Editing)
- Second Author: Writing (Review and Editing)
- Third Author: Visualization, Writing (Original Draft), Writing (Review and Editing)

Funding statement

This research was funded by the Swiss Federal Office of Civil Aviation, grant number 2022-046.

Open data statement

All the data used in this study can be downloaded from the OpenSky Network.

Reproducibility statement

The source code used for model training, evaluation, and figure generation is publicly available at `github.com/figuetbe/generative-flight-predictions`.

References

[1] César Munoz, Anthony Narkawicz, and James Chamberlain. "A TCAS-II resolution advisory detection algorithm". In: *AIAA guidance, navigation, and control (GNC) conference*. 2013, p. 4622.

[2] *EUROCONTROL Guidelines for Short Term Conflict Alert Part I: Concept and Requirements*. Tech. rep. EUROCONTROL-GUID-159. Released Issue. Brussels, Belgium: European Organisation for the Safety of Air Navigation (EUROCONTROL), Jan. 2017. URL: https://www.eurocontrol.int/publication/guidelines-short-term-conflict-alert-stca.

[3] Yulin Liu and Mark Hansen. "Predicting Aircraft Trajectories: A Deep Generative Convolutional Recurrent Neural Networks Approach". In: *arXiv preprint arXiv:1812.11670* (2018). URL: https://arxiv.org/abs/1812.11670.

[4] Timothé Krauth, Jérôme Morio, Xavier Olive, Benoit Figuet, and Raphael Monstein. "Synthetic Aircraft Trajectories Generated with Multivariate Density Models". In: *Eng. Proc.* Vol. 13. 2021, p. 7. DOI: 10.3390/engproc2021013007.

[5] Gabriel Jarry, Nicolas Couellan, and Daniel Delahaye. "On the use of generative adversarial networks for aircraft trajectory generation and atypical approach detection". In: *ENRI International Workshop on ATM/CNS*. Springer. 2019, pp. 227–243.

[6] Weili Zeng, Xiao Chu, Zhengfeng Xu, Yan Liu, and Zhibin Quan. "Aircraft 4D trajectory prediction in civil aviation: A review". In: *Aerospace* 9.2 (2022), p. 91.

[7] Timothé Krauth, Jan Krummen, and Benoit Figuet. *Multi-Objective CNN-LSTM for Aircraft Trajectory Prediction with Spatio-Temporal Confidence Areas*. Available at SSRN: https://ssrn.com/abstract=5387659 or http://dx.doi.org/10.2139/ssrn.5387659. SSRN Working Paper. 2025. DOI: 10.2139/ssrn.5387659.

[8] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. "Flow Matching for Generative Modeling". In: *arXiv preprint arXiv:2210.02747* (2023). URL: https://arxiv.org/abs/2210.02747.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial networks". In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[10] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[12] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. "Improving and generalizing flow-based generative models with minibatch optimal transport". In: *arXiv preprint arXiv:2302.00482* (2023).

[13] Xingchao Liu, Chengyue Gong, and Qiang Liu. "Flow straight and fast: Learning to generate and transfer data with rectified flow". In: *arXiv preprint arXiv:2209.03003* (2022).

[14] Peter Holderrieth and Ezra Erives. *Introduction to Flow Matching and Diffusion Models*. 2025. URL: https://diffusion.csail.mit.edu/.

[15] Matthias Schäfer, Martin Strohmeier, Vincent Lenders, Ivan Martinovic, and Matthias Wilhelm. "Bringing up OpenSky: A large-scale ADS-B sensor network for research". In: *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*. IEEE. 2014, pp. 83–94.

[16] Xavier Olive. "traffic, a toolbox for processing and analysing air traffic data". In: *Journal of Open Source Software* 4 (2019), p. 1518. ISSN: 2475-9066. DOI: 10.21105/joss.01518.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186.