

DiTSG: Context-Aware Air Traffic Scene Generation with Multi-Agent Diffusion Transformers

Seokbin Yoon¹, Kyungmin Kim², Taejung Kwon², Keumjin Lee², Peng Wei¹

¹George Washington University, Washington, DC 20052, USA

²Korea Aerospace University, Goyang 10540, South Korea

{seokbin.yoon, pwei}@gwu.edu, {kyungmin.kim, ktj0302}@kau.kr, keumjin.lee@kau.ac.kr

Abstract—Understanding how many aircraft, and which traffic configurations, can be safely and efficiently managed by air traffic controllers within a given airspace is fundamental to air traffic management. Air traffic controllers continuously modify aircraft trajectories through interventions such as speed advisories and radar vectoring, and the patterns of these interventions are difficult to capture effectively using existing air traffic simulators. To address this limitation, we propose Diffusion Transformers for Traffic Scene Generation (DiTSG), which learns to generate air traffic scenes from recorded multi-aircraft trajectories. DiTSG employs stacked multi-agent inverted Transformer layers as the denoising backbone of a diffusion transformer, jointly generates the trajectories of multiple aircraft within a traffic scene, and conditions the generation on agent-specific required times of arrival. The resulting trajectories generated by DiTSG preserve realistic inter-aircraft interactions observed in actual air traffic operations. Experiments show that the DiTSG generates diverse traffic scenes that reproduce the statistical properties of recorded traffic while preserving realistic inter-aircraft interactions. The project page and source code are available [here](#).

Index Terms—air traffic control, trajectory generation, generative models, diffusion transformers

I. INTRODUCTION

Modeling and simulating interacting flight trajectories in airspace is fundamental to assessing the safety and efficiency of aircraft operations, including the maximum level of traffic complexity that air traffic management (ATM) systems can safely accommodate [1, 2, 3]. A key challenge is that air traffic controllers (ATCs) continuously modify aircraft trajectories through interventions such as radar vectoring and speed changes, both to maintain safe separation and to improve flight time and fuel consumption [4, 5]. Because these interventions depend on various factors, including the surrounding traffic, the real-world aircraft trajectories can unfold in many different ways even under similar levels of traffic demand. Consequently, no single traffic scenario can adequately represent the range of possible traffic situations. An air traffic generation model should therefore capture the distribution of possible realizations rather than generate only a single set of trajectories for a given scenario.

A common approach to generating traffic is to use air traffic simulators such as BlueSky [6]. Although this approach enables a large-scale simulation of air traffic operations based on predefined flight plans and airspace procedures, realistically reproducing how ATCs modify aircraft trajectories remains challenging. This limitation is particularly important in terminal airspace, where controller interventions are frequent

and substantial. As a result, conventional simulation-based approaches may have limited ability to reproduce complex traffic behaviors that emerge from real-world ATC operations.

Alternatively, data-driven generative modeling can learn air traffic behavior directly from real-world trajectories. From a probabilistic perspective, each observed traffic scene can be regarded as a sample from an underlying, generally unknown distribution of possible air traffic operations. A generative model can learn this distribution and then generate new, plausible samples beyond those in the observed dataset, broadening the range of simulated traffic scenarios.

Existing data-driven trajectory generation methods primarily focus on modeling individual aircraft behavior [7, 8, 9, 10]. However, a trajectory generated in isolation cannot preserve how ATCs perceive and manage the traffic situation as a whole. Consequently, independently sampling individual trajectories may yield a traffic scene with a collection of individually plausible flights, but does not necessarily produce a realistic traffic scene that preserves relationships among aircraft, such as separation and sequencing. Generating traffic scenes in which multiple aircraft properly interact is therefore required to capture not only individual flight behavior but also inter-aircraft dependencies.

Toward this goal, we propose Diffusion Transformers for Traffic Scene Generation (*DiTSG*), a context-aware generative model for multi-agent air traffic scenes, where all aircraft trajectories are generated jointly and each trajectory is modeled in the context of the others. DiTSG employs stacked multi-agent inverted Transformer (MAIFormer) [11] layers as the denoising backbone of a diffusion transformer. Generation is further conditioned on agent-specific required times of arrival (RTAs), enabling different temporal traffic configurations. Starting from noise, DiTSG iteratively refines the trajectories of all aircraft in a scene while modeling inter-aircraft dependencies at every denoising step. This joint denoising process enables the generated trajectories to remain mutually consistent while preserving realistic traffic interactions.

Our main contributions are threefold:

- 1) We propose DiTSG, a context-aware generative framework that formulates air traffic scene generation as a joint multi-agent generation problem, where each aircraft trajectory is generated in the context of the others rather than being modeled independently from the surrounding traffic.

- 2) We develop a diffusion transformer with a multi-agent denoising backbone that enables information aggregation across aircraft through attention at every denoising step, while supporting a variable number of aircraft and controllability through agent-specific RTAs.
- 3) We evaluate DiTSG using real-world terminal airspace data and demonstrate that it generates diverse traffic scenes under the same conditional information and maintains realistic inter-aircraft separation.

II. RELATED WORK

This work draws on two lines of research: aircraft trajectory generation and multi-agent learning for ATM systems. We review each area below and discuss their limitations for context-aware multi-agent air traffic scene generation.

A. Trajectory Generation

Before the emergence of data-driven approaches, aircraft trajectory generation largely relied on physics-based models, in which aircraft motion is governed by aerodynamic principles and performance models [12, 13]. These models provide strong physical fidelity, but have limited ability to reproduce the uncertainty and variability observed in real-world trajectories. Consequently, they are less suited to generating diverse traffic realizations under different operational conditions [14]. One of the efforts toward overcoming this limitation therefore lies in data-driven trajectory generation, in which models learn uncertainty and variability directly from the archived real-world trajectory data.

Early data-driven studies used statistical methods such as Gaussian mixture models (GMMs) [15, 16] and synthetic minority oversampling [9] for trajectory generation. Deep generative approaches later combined temporal convolutional networks (TCNs) with variational autoencoders (VAEs) for trajectory generation [8], including joint generation of trajectory pairs for collision risk assessment [17]. Generative adversarial networks (GANs) have also been applied to synthesize flight trajectories [18, 19], while transformer-based autoencoders have been paired with GMMs in the latent space [7]. More recently, diffusion and flow matching models have been explored for trajectory generation [20].

Prior work, however, primarily focuses on individual aircraft or limited pairwise interactions, which cannot fully capture the joint behavior of a traffic scene. DiTSG addresses this limitation by jointly generating multiple aircraft while explicitly modeling inter-aircraft dependencies. It also supports generating a variable number of aircraft, reflecting the fact that the number of aircraft simultaneously present within an airspace naturally varies over time.

B. Multi-Agent Learning

ATM systems are inherently multi-agent, since ATC interventions affect the trajectories of several interacting aircraft, but multi-agent learning remains less explored than its single-agent counterpart in the ATM domain. Social long short-term memory (LSTM) networks [21] and attention-based

models [22] capture inter-aircraft interactions for trajectory prediction, while transformers combined with Bayesian neural networks jointly model interactions and uncertainty in traffic scenes [23]. Agent-aware attention inspired by AgentFormer [24] has also been adapted to air traffic scenes [25]. MAIFormer was proposed more recently, using a hierarchical attention mechanism that mimics ATCs' decision-making for context-aware modeling of interacting aircraft [11].

Nevertheless, prior multi-agent learning approaches primarily focus on supervised prediction of future trajectories or flight times and are therefore not designed to generate novel traffic scenarios beyond the observed samples, limiting their ability to explore diverse plausible traffic configurations. DiTSG extends multi-agent learning from prediction to generation by jointly modeling interacting aircraft and producing diverse traffic realizations that remain consistent with patterns learned from observed data.

III. PRELIMINARIES

A. Diffusion and Flow Matching

Diffusion-based generative models transport samples from a simple prior distribution to the data distribution through a continuous-time process described by differential equations [26]. In this work, we adopt the ordinary differential equation (ODE) formulation and train our diffusion transformer with a conditional flow matching (CFM) objective on rectified linear interpolation paths [27, 28]. CFM directly learns a velocity field that transports samples along a prescribed probability path.

Let $\mathbf{x}_0 \sim p_0$ denote a sample from the prior distribution, where $p_0 = \mathcal{N}(\mathbf{0}, \mathbf{I})$, and let $\mathbf{x}_1 \sim p_1$ denote a sample from the data distribution. A continuous probability path connects p_0 and p_1 over the diffusion timestep $s \in [0, 1]$. We define this path using linear interpolation:

$$\mathbf{x}_s = (1 - s)\mathbf{x}_0 + s\mathbf{x}_1. \quad (1)$$

The path transports each paired noise and data sample along a straight line from $\mathbf{x}_0$ at $s = 0$ to $\mathbf{x}_1$ at $s = 1$, so the target velocity is constant and given by

$$\mathbf{v}_s = \frac{d\mathbf{x}_s}{ds} = \mathbf{x}_1 - \mathbf{x}_0. \quad (2)$$

CFM learns a parameterized velocity field $\mathbf{v}_\theta(\mathbf{x}_s, s)$ that approximates this target velocity by minimizing

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{\substack{s \sim \mathcal{U}(0,1), \\ \mathbf{x}_0 \sim p_0, \mathbf{x}_1 \sim p_1}} \left[\|\mathbf{v}_\theta(\mathbf{x}_s, s) - (\mathbf{x}_1 - \mathbf{x}_0)\|_2^2 \right]. \quad (3)$$

Once trained, the learned velocity field defines the ODE

$$\frac{d\mathbf{x}_s}{ds} = \mathbf{v}_\theta(\mathbf{x}_s, s). \quad (4)$$

We generate a sample by integrating this ODE from $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ at $s = 0$ toward $s = 1$. Using a first-order Euler solver, each integration step is

$$\mathbf{x}_{s+\Delta s} = \mathbf{x}_s + \Delta s \mathbf{v}_\theta(\mathbf{x}_s, s). \quad (5)$$

B. Multi-Agent Inverted Transformer

MAIFormer is a multi-agent model for air traffic scenes that captures intra-agent trajectory evolution at the low level and inter-agent influence at the high level [11]. It has proven a versatile backbone for multi-agent modeling, with subsequent extensions to arrival time prediction [29] and to interpreting air traffic situations through agent attention scores [30]. MAIFormer is an encoder-only architecture with three key components: (i) air traffic scene tokenization, (ii) masked multivariate attention (MMA), and (iii) agent attention (AA)¹.

Air Traffic Scene Tokenization. MAIFormer's tokenization extends the inverted embedding strategy of iTransformer [31], which embeds along the variate dimension to form variate tokens rather than temporal tokens and has been effective for multivariate time-series and trajectory modeling [32]. Given a multi-agent flight trajectory sequence $\mathbf{X} \in \mathbb{R}^{N \times T \times F}$ (N agents, T time steps, F variates), we reshape it into $\bar{\mathbf{X}} \in \mathbb{R}^{(N \cdot F) \times T}$ and treat the entire scene as a single multivariate time series. Scene tokenization then projects the T -dimensional temporal sequence of each variate into the model latent space of dimension D , producing

$$\mathbf{C} = \text{SceneTokenization}(\bar{\mathbf{X}}) \in \mathbb{R}^{(N \cdot F) \times D}, \quad (6)$$

where $\mathbf{C} = (\mathbf{c}_1^1, \dots, \mathbf{c}_1^F, \dots, \mathbf{c}_N^1, \dots, \mathbf{c}_N^F)$, and each token represents the temporal evolution of one variate for one aircraft.

Masked Multivariate Attention. MMA applies self-attention (SA) [33] to the variate tokens $\mathbf{C}$ to model spatio-temporal (ST) dependencies within each aircraft, using a mask $\mathbf{M}$ that blocks cross-agent interactions, such that the tokens of each aircraft attend only to other variates of the same aircraft:

$$\begin{aligned} \text{MMA}(\mathbf{Q}_{\text{ST}}, \mathbf{K}_{\text{ST}}, \mathbf{V}_{\text{ST}}) &= \text{Softmax} \left(\frac{\mathbf{Q}_{\text{ST}} \mathbf{K}_{\text{ST}}^\top}{\sqrt{d_{k, \text{ST}}}} + \mathbf{M} \right) \mathbf{V}_{\text{ST}}, \\ \mathbf{Q}_{\text{ST}} &= \mathbf{C} \mathbf{W}_{\text{ST}}^Q, \quad \mathbf{K}_{\text{ST}} = \mathbf{C} \mathbf{W}_{\text{ST}}^K, \quad \mathbf{V}_{\text{ST}} = \mathbf{C} \mathbf{W}_{\text{ST}}^V, \end{aligned} \quad (7)$$

where each $\mathbf{W}_{\text{ST}} \in \mathbb{R}^{D \times D}$ projects $\mathbf{C}$ into queries, keys, and values. By restricting information flow to the variates of each individual aircraft, MMA captures intra-agent dependencies.

Agent Attention. AA models the social context (SC) among agents by applying SA across agent-level tokens. Each agent token is formed by concatenating the variate tokens produced by the preceding MMA stage:

$$\mathbf{C}_{A,i} = (\mathbf{c}_i^1 \oplus \dots \oplus \mathbf{c}_i^F) \in \mathbb{R}^{1 \times (F \cdot D)}, \quad (8)$$

where $\oplus$ denotes concatenation along the variate dimension. SA is then applied to $\mathbf{C}_A \in \mathbb{R}^{N \times (F \cdot D)}$ with separate projections $\mathbf{W}_{\text{SC}} \in \mathbb{R}^{(F \cdot D) \times (F \cdot D)}$:

$$\begin{aligned} \text{AA}(\mathbf{Q}_{\text{SC}}, \mathbf{K}_{\text{SC}}, \mathbf{V}_{\text{SC}}) &= \text{Softmax} \left(\frac{\mathbf{Q}_{\text{SC}} \mathbf{K}_{\text{SC}}^\top}{\sqrt{d_{k, \text{SC}}}} \right) \mathbf{V}_{\text{SC}}, \\ \mathbf{Q}_{\text{SC}} &= \mathbf{C}_A \mathbf{W}_{\text{SC}}^Q, \quad \mathbf{K}_{\text{SC}} = \mathbf{C}_A \mathbf{W}_{\text{SC}}^K, \quad \mathbf{V}_{\text{SC}} = \mathbf{C}_A \mathbf{W}_{\text{SC}}^V. \end{aligned} \quad (9)$$

¹We refer readers to [11] for full architectural details.

AA therefore relates whole aircraft rather than individual variates, at a level of abstraction consistent with how human ATCs reason about air traffic scenes.

IV. TECHNICAL APPROACH

DiTSG generates multi-agent air traffic scenes by learning a joint velocity field over aircraft trajectories. We use a stack of L MAIFormer layers to parameterize this field, allowing inter-aircraft dependencies to be modeled throughout the generation process. Traffic scene generation is conditioned on the RTAs τ of all aircraft.

A. DiTSG Neural Architecture

Noisy Scene Tokenization. Let $\mathbf{X}_1 \in \mathbb{R}^{N \times T \times F}$ denote a real traffic scene. During training, we sample a diffusion timestep $s \sim \mathcal{U}(0, 1)$ and noise $\mathbf{X}_0 \in \mathbb{R}^{N \times T \times F} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and construct the interpolated scene $\mathbf{X}_s = (1 - s)\mathbf{X}_0 + s\mathbf{X}_1$. We reshape $\mathbf{X}_s$ into $\bar{\mathbf{X}}_s$ and apply scene tokenization:

$$\mathbf{C} = \text{SceneTokenization}(\bar{\mathbf{X}}_s) \in \mathbb{R}^{(N \cdot F) \times D}. \quad (10)$$

The tokens are those of a partially noised scene, so what a token encodes changes with s : near $s = 1$ it carries the variate's actual time series, and near $s = 0$ it carries little beyond noise.

Agent-wise Conditioning. DiTSG conditions the generation on two quantities. The diffusion timestep s applies to the entire scene and indicates its progression from noise toward data. The RTA $\tau_i \in \mathbb{R}$ is an agent-specific control input that specifies the time remaining until aircraft i is required to arrive at the runway. During generation, the RTAs are specified externally and may be provided by an arrival scheduler. Across agents, $\tau \in \mathbb{R}^N$ specifies the relative arrival order and temporal spacing among aircraft. Given this RTA configuration, DiTSG generates plausible traffic scenes that could occur under the specified arrival order and spacing. We embed both quantities using sinusoidal encodings followed by multi-layer perceptrons (MLPs) and combine them into one conditioning vector for each agent:

$$\mathbf{y}_i = \text{MLP}^s(\text{SinEnc}(s)) + \text{MLP}^\tau(\text{SinEnc}(\tau_i)) \in \mathbb{R}^D, \quad (11)$$

collected across agents as $\mathbf{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_N) \in \mathbb{R}^{N \times D}$. Following DiT [34], we inject $\mathbf{Y}$ through adaptive layer normalization, $\text{Mod}(\mathbf{c}, \gamma, \beta) = (1 + \gamma) \odot \text{LN}(\mathbf{c}) + \beta$, at every sublayer. An MLP maps each $\mathbf{y}_i$ to a shift, scale, and gate:

$$\{\beta_i^k, \gamma_i^k, \alpha_i^k\}_{k \in \{M, A, F\}} = \text{MLP}^{\text{mod}}(\mathbf{y}_i), \quad (12)$$

where k indexes the MMA, AA, and FFN sublayers. The shift and scale are applied through Mod , while the gate scales each sublayer output before it is added back to the residual stream. These modulation parameters are shared across all tokens of aircraft i , providing agent-specific conditioning throughout the network.

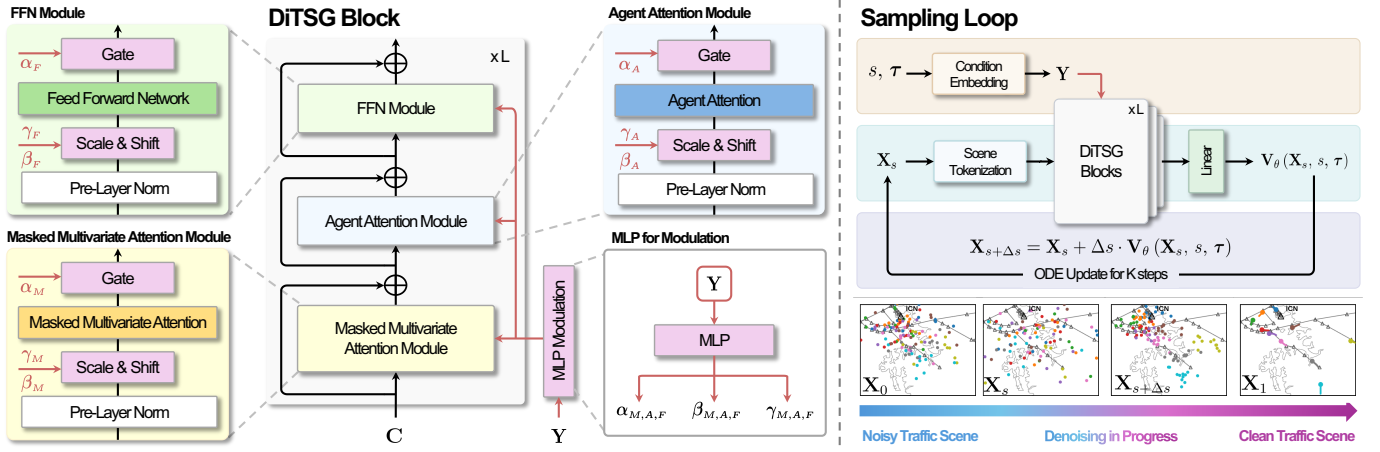


Fig. 1: Overview of DiTSG. Left: a DiTSG block, which applies masked multivariate attention, agent attention, and a feed-forward layer, each modulated by the conditioning vector $\mathbf{Y}$ through adaLN. Right: the sampling loop, which tokenizes the noisy scene $\mathbf{X}_s$, passes it through L blocks and a linear layer to obtain the velocity $\mathbf{V}_\theta$, and integrates the ODE for K Euler steps from noise to a traffic scene.

Denoising Backbone. Each DiTSG block applies the three MAIFormer sublayers with adaLN replacing the original layer normalization:

$$\begin{aligned} \mathbf{C} &= \mathbf{C} + \alpha^M \odot \text{MMA}(\text{Mod}(\mathbf{C}, \gamma^M, \beta^M)), \\ \mathbf{C}_A &= \mathbf{C}_A + \alpha^A \odot \text{AA}(\text{Mod}(\mathbf{C}_A, \gamma^A, \beta^A)), \\ \mathbf{C} &= \mathbf{C} + \alpha^F \odot \text{FFN}(\text{Mod}(\mathbf{C}, \gamma^F, \beta^F)). \end{aligned} \quad (13)$$

MMA attends within each agent’s own tokens and AA attends across agents, while the feedforward neural network (FFN) applies a nonlinear transformation to each token. Because AA is applied in every block, each aircraft is generated in the context of the surrounding traffic throughout the denoising process, rather than independently.

Velocity Prediction. A final adaLN modulates the output tokens $\mathbf{C}_{\text{out}} \in \mathbb{R}^{(N \cdot F) \times D}$, after which a linear layer projects each token back to its T -step variate sequence:

$$\bar{\mathbf{V}}_\theta = \text{Linear}(\text{Mod}(\mathbf{C}_{\text{out}}, \gamma^{\text{out}}, \beta^{\text{out}})) \in \mathbb{R}^{(N \cdot F) \times T}. \quad (14)$$

Reversing the reshape used during tokenization yields the predicted velocity field $\mathbf{V}_\theta(\mathbf{X}_s, s, \tau) \in \mathbb{R}^{N \times T \times F}$ over the multi-agent traffic scene.

B. Training Objective and Traffic Scene Generation

We train DiTSG using the CFM objective, conditioned on RTAs τ :

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{\substack{s \sim \mathcal{U}(0,1), \\ \mathbf{X}_1 \sim p_1, \mathbf{X}_0 \sim p_0}} \left[\|\mathbf{V}_\theta(\mathbf{X}_s, s, \tau) - (\mathbf{X}_1 - \mathbf{X}_0)\|_2^2 \right]. \quad (15)$$

At inference, we specify the number of aircraft N and their RTAs $\tau \in \mathbb{R}^N$, sample $\mathbf{X}_0 \in \mathbb{R}^{N \times T \times F} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and integrate the learned ODE from $s = 0$ to $s = 1$ using a K -step Euler solver. Because DiTSG does not require a fixed N , the same model can generate traffic scenes with different numbers of aircraft, reflecting the varying number of aircraft that may be present within an airspace.

V. EXPERIMENTAL SETUP

A. ADS-B Trajectory Dataset Description

We evaluate DiTSG on real-world ADS-B trajectories from the terminal airspace around Incheon International Airport (ICN), South Korea, covering arrivals to Runway 15L from January to December 2024. As one of the world’s busiest airports, ICN makes extensive use of area navigation (RNAV), allowing flexible flight-path adjustments under varying traffic conditions. This produces diverse inter-aircraft interactions, making ICN a suitable testbed for traffic scene generation.

We construct scenes using a two-minute sliding window advancing by one 6-second time step, yielding $T = 20$ steps per scene. Each scene contains the trajectories of all aircraft present during the corresponding interval. We focus on short-horizon scene generation and leave longer horizons to future work. This yields 991,665 scenes, which we split chronologically into training and test sets with a 9:1 ratio.

B. Model and Training Configurations

DiTSG stacks $L = 6$ MAIFormer layers with a hidden dimension of $D = 128$ and an FFN dimension of 512 ($4D$), using GELU activations [35]. Each aircraft trajectory consists of $F = 3$ variates: latitude, longitude, and altitude. We use four attention heads for both MMA and AA.

We train DiTSG for 300 epochs with a batch size of 1024 using AdamW [36] and a learning rate of 10^{-4} . At inference, we integrate the ODE using a 50-step Euler solver. All experiments are implemented in Python 3.11.16 with PyTorch 2.5.1 [37] and CUDA 12.1. Experiments are run on Ubuntu 24.04 LTS using an Intel Xeon W-3245 CPU, 62 GB of RAM, and an NVIDIA RTX A6000 GPU.

C. Evaluation Metrics

We evaluate the generated traffic scenes along four dimensions: fidelity to real traffic, novelty, diversity, and safety compliance.

Spatial Distribution Fidelity. We evaluate whether DiTSG generates traffic in the same regions of airspace as real aircraft. We discretize the terminal airspace into a latitude–longitude grid and compute the Jensen–Shannon divergence (JSD) between the occupancy distributions P_{real} and P_{gen} of real and generated positions:

$$D_{\text{occ}} = \text{JSD}(P_{\text{real}} \parallel P_{\text{gen}}). \quad (16)$$

A lower D_{occ} indicates greater similarity between the spatial distributions of real and generated traffic.

Novelty. A model could achieve high fidelity by reproducing training scenes, so we also measure how far generated scenes lie from the training set. For two scenes with the same N , we match their aircraft one to one and take the assignment with the minimum average trajectory distance:

$$d(A, B) = \min_{\pi \in \Pi_N} \frac{1}{N} \sum_{i=1}^N \|\mathbf{x}_i^A - \mathbf{x}_{\pi(i)}^B\|, \quad (17)$$

where $\mathbf{x}_i^A$ is the trajectory of aircraft i in scene A and Π_N is the set of all permutations of N aircraft. Defining $d(\cdot, \text{Train}) = \min_{s \in \text{Train}} d(\cdot, s)$ as the distance to the nearest training scene, we report

$$R_{\text{nov}} = \frac{\frac{1}{|\hat{S}|} \sum_{\hat{s} \in \hat{S}} d(\hat{s}, \text{Train})}{\frac{1}{|S^{\text{test}}|} \sum_{s^{\text{test}} \in S^{\text{test}}} d(s^{\text{test}}, \text{Train})}, \quad (18)$$

where $\hat{S}$ and S^{test} denote the generated and held-out test scenes, respectively. Because the test scenes represent unseen real traffic, they provide a reference for novelty: R_{nov} near one indicates that generated scenes are approximately as distant from the training data as unseen real scenes.

Diversity. We measure variation among traffic scenes generated under the same condition. For a fixed τ , we generate K scenes and compute the average pairwise horizontal distance among them in nautical miles. We interpret this metric together with D_{occ} , since greater diversity is meaningful only when generated traffic remains consistent with the spatial distribution of real traffic.

Separation Violation Rate. We evaluate safety compliance using the separation violation rate (SVR), defined as the fraction of scenes containing at least one pair of aircraft that simultaneously falls below 3 NM of horizontal separation and 1,000 ft of vertical separation. We report SVR for both generated and real test scenes to assess whether DiTSG preserves separation behavior observed in real traffic.

VI. NUMERICAL RESULTS

A. Quantitative Results

Table I compares DiTSG with a multi-agent VAE (MA-VAE) using the same MAIFormer backbone, so the main difference lies in the generative formulation: flow matching for DiTSG and variational latent modeling for MA-VAE.

DiTSG reduces D_{occ} by more than half (0.015 vs. 0.033) and lowers SVR from 0.250% to 0.101% compared with MA-VAE, showing closer agreement with the spatial distribution

TABLE I: Evaluation of generated traffic scenes.

Method	D_{occ}	R_{nov}	Diversity	SVR (%)
Real	–	1.000	–	0.000
MA-VAE	0.033	0.947	16.690	0.250
DiTSG	0.015	0.953	14.938	0.101

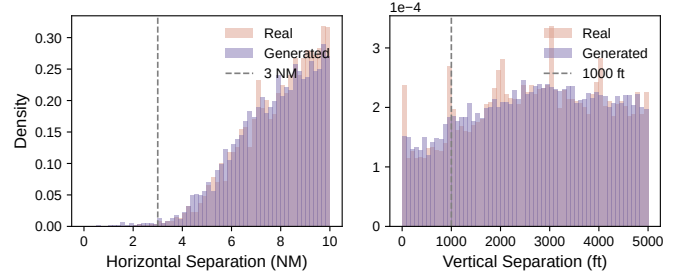


Fig. 2: Separation patterns in real and generated traffic. Distribution of the minimum horizontal (left) and vertical (right) separation between aircraft pairs. Dashed lines indicate the separation minima.

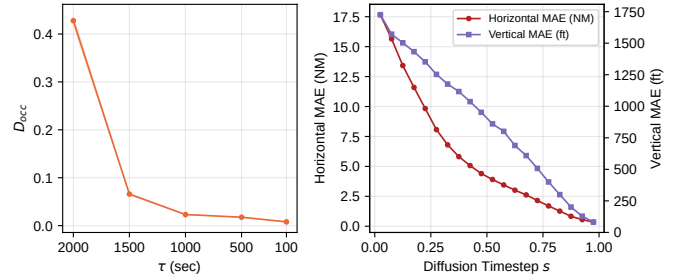


Fig. 3: Traffic fidelity across arrival stages and diffusion timesteps. JSD between real and generated traffic within each RTA (τ) range (left). Horizontal and vertical mean absolute error between DiTSG’s predicted and true positions as a function of the diffusion timestep s (right).

and inter-aircraft separation observed in real traffic. Both models achieve R_{nov} close to one (0.953 and 0.947), indicating that their generated scenes remain distinct from the training data. MA-VAE shows greater diversity under a fixed τ (16.690 NM vs. 14.938 NM), but also a substantially higher D_{occ} , suggesting that some of this variation extends beyond the spatial patterns observed in real traffic. Because diversity has no direct reference value, we interpret it mainly as evidence that DiTSG does not collapse to a single scene for a given condition rather than as a metric for ranking the two models.

The performance difference may be explained by how the two models generate air traffic scenes. During training, MA-VAE obtains agent latents from an encoder that observes the entire scene, so their joint configuration reflects the interactions among aircraft. At generation time, however, each agent latent is sampled independently from $\mathcal{N}(\mathbf{0}, \mathbf{I})$. Even if each individual latent follows the prior, independent sampling does not preserve the joint arrangement present during training, leaving the decoder to reconstruct unrealistic inter-aircraft relationships from latent combinations that have not been seen in historical data. In contrast, DiTSG starts from the same Gaussian prior it is trained to transport and follows the learned joint velocity field over multiple aircraft at inference,

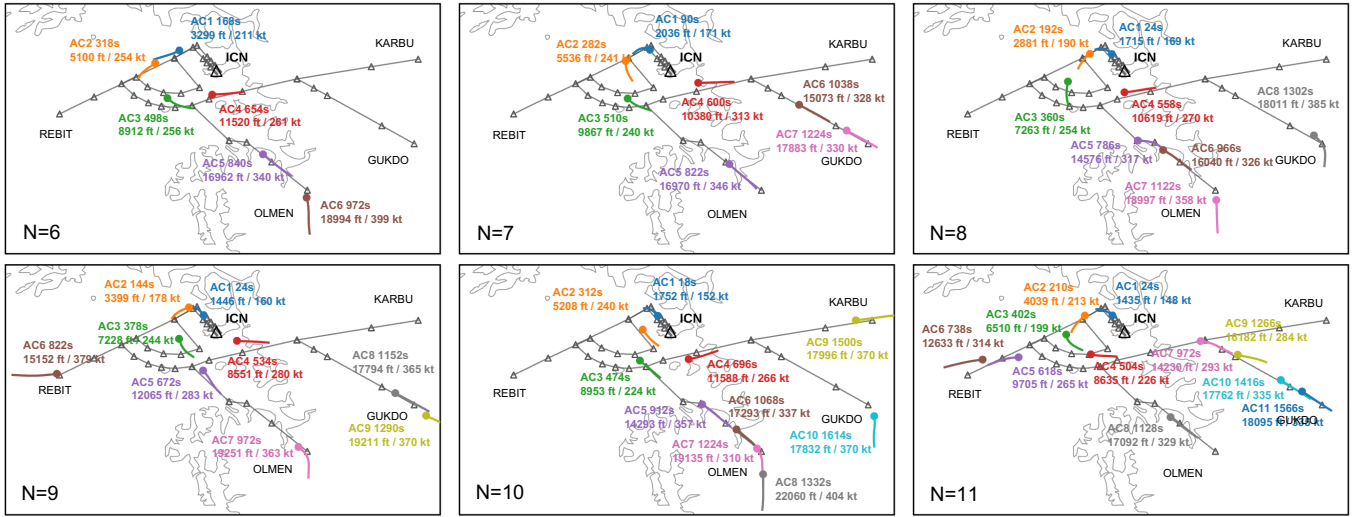


Fig. 4: Generated traffic scenes with varying numbers of aircraft. Examples of scenes generated by DiTSG for $N = 6$ to $N = 11$. Each panel shows one scene over the ICN terminal area, with STARs and runways shown in gray for reference. Aircraft are labeled AC1 to ACN in landing order, and each label reports the RTA τ , current altitude, and ground speed.

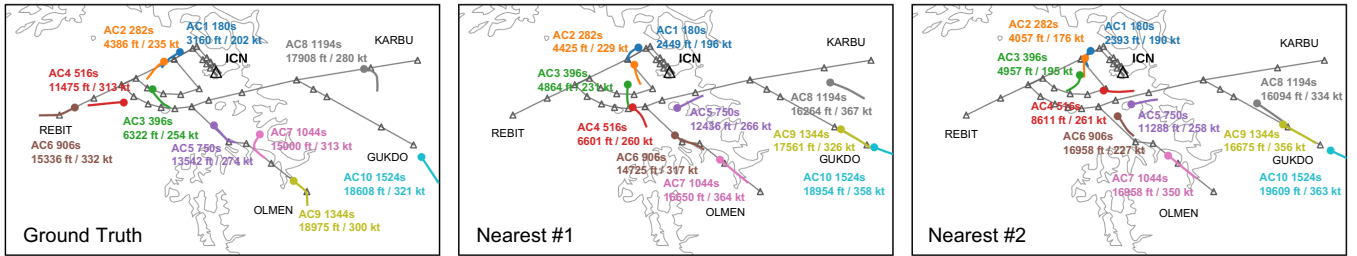


Fig. 5: Generated alternatives under the same RTA condition. Comparison between a recorded traffic scene and the closest scenes generated by DiTSG under the same τ .

reducing the mismatch between training and generation. This may contribute to its closer agreement with real traffic.

Figure 2 compares the distributions of minimum horizontal and vertical separation for every aircraft pair in real and generated traffic scenes. The distributions closely match, showing close agreement in both separation violations and overall inter-aircraft spacing. In the real data, pairs falling below the minimum are less frequent horizontally than vertically, indicating that horizontal separation is more consistently maintained under real-world ATC operations, a pattern also reproduced by DiTSG. Matching these separation patterns is important because inter-aircraft spacing affects both safety and the level of traffic an airspace can accommodate.

We report two additional analyses in Figure 3. The left panel groups aircraft by RTAs and computes D_{occ} within each group, showing how closely generated traffic matches real traffic at different stages of arrival. For RTAs below about 1000 s, D_{occ} remains near 0.02 with little variation as trajectories converge toward common arrival paths near the runway. D_{occ} increases mainly for aircraft farther from the runway, where later RTAs allow a wider range of plausible trajectories. The right panel shows the horizontal and vertical mean absolute error between predicted and true scenes as a function of the diffusion timestep s . Both decrease monotonically from $s = 0$

to $s = 1$, indicating that the model progressively refines the scene from noise throughout denoising.

B. Qualitative Results

Figure 4 shows traffic scenes generated by DiTSG for varying numbers of simultaneous aircraft ($N = 6$ to $N = 11$). For each aircraft, we report the conditioning RTA τ_i , together with the current altitude and ground speed. Across all panels, DiTSG distributes aircraft over the four arrival fixes (REBIT, OLMEN, GUKDO, and KARBU), although the conditioning does not specify which fix to use. The scenes also show coherent arrival sequencing: aircraft with earlier RTAs are closer to the runway, lower, and slower. Even when deviating from the standard terminal arrival routes (STARs), aircraft form common arrival flows resembling traffic patterns shaped by ATC vectoring.

Particularly, in the generated scene with $N = 9$, AC5, AC6, and AC7 have their RTAs separated by approximately 150 seconds. Despite this similar temporal spacing, the spatial distance between AC5 and AC6 is much smaller than that between AC6 and AC7, suggesting that AC6 takes a longer route to the runway to maintain the landing interval. Such behavior is consistent with traffic patterns observed for REBIT arrivals at ICN. Because REBIT is located closer to the airport than the other arrival fixes, ATCs often vector aircraft entering

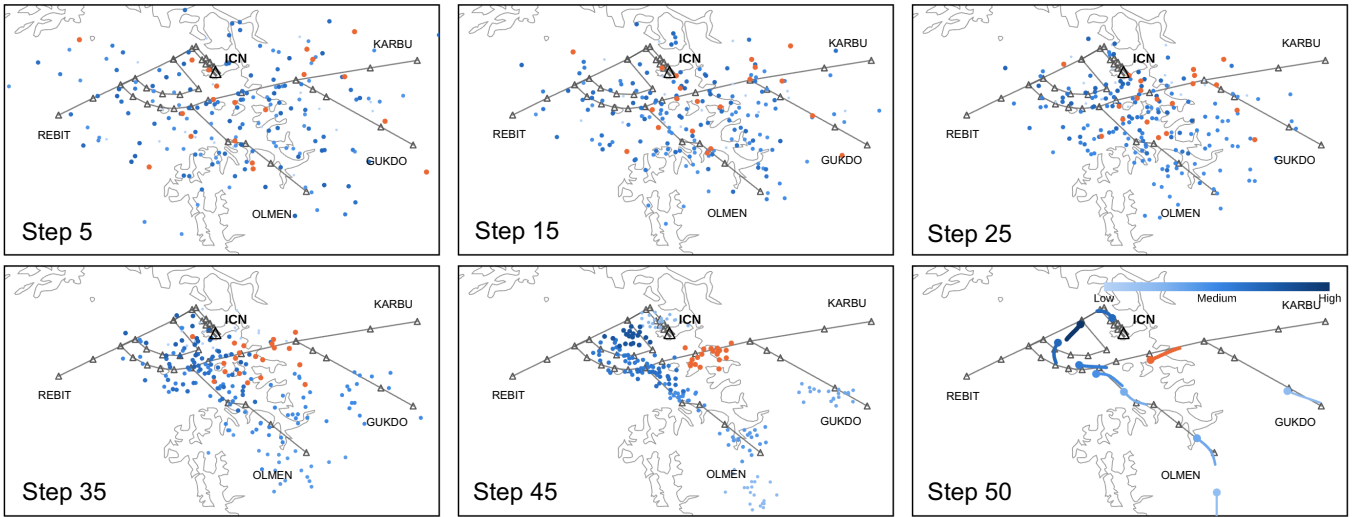


Fig. 6: Agent attention throughout the denoising process. The query aircraft is shown in orange, while the other aircraft are colored according to the attention they receive from the query. Intermediate states are shown as points rather than trajectories because they remain noisy during early denoising steps.

through REBIT to maintain separation and merge them into the arrival stream, increasing their path length and time in the terminal airspace. This example suggests that DiTSG captures not only the spatial distribution of traffic but also operational patterns induced by ATC vectoring.

To further assess whether DiTSG generates alternative traffic realizations under the same condition, we take the RTAs τ from a real scene, generate multiple scenes using that same τ . A useful generative model should represent the range of traffic scenes that could plausibly occur under a given condition, not the one scene that happened to be recorded. Figure 5 shows the ground-truth recorded scene alongside the two generated samples closest to it. Even these closest samples remain visibly different. AC7 provides a clear example: although its RTA is fixed to the same value in all scenes, its trajectory varies substantially. In the ground truth, AC7 turns right and remains outside the main arrival stream while REBIT traffic occupies the airspace near the airport, landing only after that stream has passed. In nearest #1, by contrast, AC7 joins the existing flow without the same detour. Thus, the same RTAs and arrival sequence can correspond to different plausible trajectory configurations, and DiTSG generates such alternatives rather than reproducing the recorded scene.

Lastly, Figure 6 shows the agent attention scores throughout the denoising process. The query aircraft is shown in orange, while the others are colored according to the attention they receive from the query. Up to to about step 25, the scores remain nearly uniform because the scene is still dominated by noise and little relational structure has emerged. As denoising progresses, the attention becomes more differentiated: the query assigns higher attention to aircraft ahead in the arrival sequence and lower attention to those behind. This behavior is consistent with arrival flight operations, where preceding aircraft often have a stronger influence on following traffic. DiTSG learns this interaction pattern directly during genera-

tion without explicit supervision on which aircraft should be attended to.

VII. CONCLUSION

We present DiTSG, a context-aware generative framework for multi-agent air traffic scenes. DiTSG employs MAFormer layers as the denoising backbone of a diffusion transformer trained with a flow matching objective, enabling the joint generation of interacting aircraft while supporting a variable number of aircraft and conditioning scene generation on their RTAs. Experiments on real-world terminal airspace data show that DiTSG more closely matches the spatial distribution and inter-aircraft separation of recorded traffic than a MA-VAE baseline, while generating diverse and novel traffic realizations rather than reproducing the training data.

Toward practical deployment, DiTSG can support experimental studies of how ATCs manage different traffic configurations. By conditioning on the number of aircraft and their RTAs, users can specify an arrival sequence and its temporal spacing and generate multiple traffic scenes that realize it. This enables controlled scenario generation for operational analysis and controller training. DiTSG can also be used to generate synthetic traffic data for dataset augmentation, particularly for rare or underrepresented arrival scenarios, which may help train more robust and accurate models.

In the very near future, we plan to extend DiTSG from short-horizon scene generation to full-trajectory generation, toward end-to-end air traffic simulation over longer horizons. We also plan to evaluate physical feasibility more comprehensively with respect to aircraft dynamics and operational constraints. In particular, incorporating explicit separation constraints through constrained sampling is an important direction for future work. These extensions would broaden the applicability of DiTSG to more realistic air traffic simulation and operational studies.

REFERENCES

- [1] D. K. Schmidt, "On modeling ATC Work Load and Sector Capacity," *Journal of Aircraft*, vol. 13, no. 7, pp. 531–537, 1976. [1](#)
- [2] A. Majumdar and J. Polak, "Estimating Capacity of Europe's Airspace using a Simulation Model of Air Traffic Controller Workload," *Transportation Research Record*, vol. 1744, no. 1, pp. 30–43, 2001. [1](#)
- [3] J. D. Welch, "En Route Sector Capacity Model: Final Report," Tech. Rep. ATC-426, MIT Lincoln Laboratory, 2015. [1](#)
- [4] R. Sáez, X. Prats, T. Polishchuk, and V. Polishchuk, "Traffic Synchronization in Terminal Airspace to Enable Continuous Descent Operations in Trombone Sequencing and Merging Procedures: An Implementation Study for Frankfurt Airport," *Transportation Research Part C: Emerging Technologies*, vol. 121, p. 102875, 2020. [1](#)
- [5] S. Hong and K. Lee, "Trajectory Prediction for Vectored Area Navigation Arrivals," *Journal of Aerospace Information Systems*, vol. 12, no. 7, pp. 490–502, 2015. [1](#)
- [6] J. M. Hoekstra and J. Ellerbroek, "Bluesky ATC Simulator Project: An Open Data and Open Source Approach," in *International Conference on Research in Air Transportation*, vol. 131, p. 132, 2016. [1](#)
- [7] S. Yoon and K. Lee, "Aircraft Trajectory Dataset Augmentation in Latent Space," *International Journal of Aeronautical and Space Sciences*, vol. 27, no. 2, pp. 1960–1971, 2026. [1](#), [2](#)
- [8] T. Krauth, A. Lafage, J. Morio, X. Olive, and M. Waltert, "Deep Generative Modelling of Aircraft Trajectories in Terminal Manoeuvring Areas," *Machine Learning with Applications*, vol. 11, p. 100446, 2023. [1](#), [2](#)
- [9] S. Yoon and K. Lee, "Improving Aircraft Trajectory Prediction Accuracy with Over-sampling Technique," in *2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC)*, pp. 1–6, IEEE, 2023. [1](#), [2](#)
- [10] N. Pepper, G. De Ath, B. Carvell, A. Hodgkin, T. Dodwell, M. Thomas, and R. Everson, "A Sector-Specific Probabilistic Approach for 4D Aircraft Trajectory Generation," *Transportation Research Part C: Emerging Technologies*, vol. 179, p. 105291, 2025. [1](#)
- [11] S. Yoon and K. Lee, "Multi-Agent Inverted Transformer for Flight Trajectory Prediction," *IEEE Transactions on Intelligent Transportation Systems*, 2026. [1](#), [2](#), [3](#)
- [12] A. Nuic, D. Poles, and V. Mouillet, "BADA: An Advanced Aircraft Performance Model for Present and Future ATM Systems," *International Journal of Adaptive Control and Signal Processing*, vol. 24, no. 10, pp. 850–866, 2010. [2](#)
- [13] J. Sun, J. M. Hoekstra, and J. Ellerbroek, "OpenAP: An Open-Source Aircraft Performance Model for Air Transportation Studies and Simulations," *Aerospace*, vol. 7, no. 8, p. 104, 2020. [2](#)
- [14] X. Olive, J. Sun, M. C. R. Murça, and T. Krauth, "A Framework to Evaluate Aircraft Trajectory Generation Methods," in *US-Europe Air Traffic Management Research and Development Seminar*, pp. 17–19, 2021. [2](#)
- [15] S. T. Barratt, M. J. Kochenderfer, and S. P. Boyd, "Learning Probabilistic Trajectory Models of Aircraft in Terminal Airspace from Position Data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3536–3545, 2018. [2](#)
- [16] S. Jung, A. Hardy, and M. J. Kochenderfer, "Inferring Traffic Models in Terminal Airspace from Flight Tracks and Procedures," *Journal of Aerospace Information Systems*, vol. 22, no. 8, pp. 730–740, 2025. [2](#)
- [17] T. Krauth, J. Morio, X. Olive, and B. Figuet, "Advanced Collision Risk Estimation in Terminal Manoeuvring Areas using a Disentangled Variational Autoencoder for Uncertainty Quantification," *Engineering Applications of Artificial Intelligence*, vol. 133, p. 108137, 2024. [2](#)
- [18] G. Jarry, N. Couellan, and D. Delahaye, "On the Use of Generative Adversarial Networks for Aircraft Trajectory Generation and Atypical Approach Detection," in *ENRI International Workshop on ATM/CNS*, pp. 227–243, 2019. [2](#)
- [19] S. Wijnands, A. Sharpanskykh, and K. Aly, "Generation of Synthetic Aircraft Landing Trajectories Using Generative Adversarial Networks," in *SESAR Innovation Days*, 2024. [2](#)
- [20] H. Matsuda, N. Takeishi, and T. Yairi, "High-fidelity Aircraft Trajectory Generation Using a Flow-based Generative Model," *Transactions of the Japan Society for Aeronautical and Space Sciences*, vol. 69, no. 1, pp. 9–18, 2026. [2](#)
- [21] Z. Xu, W. Zeng, X. Chu, and P. Cao, "Multi-Aircraft Trajectory Collaborative Prediction Based on Social Long Short-Term Memory Network," *Aerospace*, vol. 8, no. 4, p. 115, 2021. [2](#)
- [22] C. H. Nguyen and R. P. Liem, "Multi-Aircraft Attention-Based Model for Perceptive Arrival Transit Time Prediction," *Advanced Engineering Informatics*, vol. 64, p. 103067, 2025. [2](#)
- [23] Y. Pang, X. Zhao, J. Hu, H. Yan, and Y. Liu, "Bayesian Spatio-Temporal Graph Transformer Network (B-STAR) for Multi-Aircraft Trajectory Prediction," *Knowledge-Based Systems*, vol. 249, p. 108998, 2022. [2](#)
- [24] Y. Yuan, X. Weng, Y. Ou, and K. M. Kitani, "AgentFormer: Agent-Aware Transformers for Socio-Temporal Multi-Agent Forecasting," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9813–9823, 2021. [2](#)
- [25] C. Deng, H.-C. Choi, H. Park, and I. Hwang, "Multi-Agent Learning for Data-Driven Air Traffic Management Applications," *Journal of Air Transport Management*, vol. 128, p. 102843, 2025. [2](#)
- [26] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851, 2020. [2](#)
- [27] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow Matching for Generative Modeling," in *International Conference on Learning Representations*, 2023. [2](#)
- [28] X. Liu, C. Gong, and Q. Liu, "Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow," in *International Conference on Learning Representations*, 2023. [2](#)
- [29] K. Kim, S. Yoon, and K. Lee, "Probabilistic Multi-Agent Aircraft Landing Time Prediction," in *AIAA SCITECH 2026 Forum*, p. 1764, 2026. [3](#)
- [30] H.-a. Chai, S. Yoon, and K. Lee, "Learning to Explain Air Traffic Situation," in *US-Europe Air Transportation Research & Development Symposium*, 2025. arXiv:2502.10764. [3](#)
- [31] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, "iTransformer: Inverted Transformers are Effective for Time Series Forecasting," in *International Conference on Learning Representations*, vol. 2024, pp. 11116–11140, 2024. [3](#)
- [32] S. Yoon and K. Lee, "Aircraft Trajectory Prediction with Inverted Transformer," *IEEE Access*, vol. 13, pp. 26318–26330, 2025. [3](#)
- [33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," *Advances in Neural Information Processing Systems*, vol. 30, 2017. [3](#)
- [34] W. Peebles and S. Xie, "Scalable Diffusion Models with Transformers," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182, IEEE, 2023. [3](#)
- [35] D. Hendrycks and K. Gimpel, "Gaussian Error Linear Units (GELUs)," *arXiv preprint arXiv:1606.08415*, 2016. [4](#)
- [36] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," in *International Conference on Learning Representations*, 2019. [4](#)
- [37] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *Advances in Neural Information Processing Systems*, vol. 32, 2019. [4](#)