FlightPatchNet: Multi-Scale Patch Network with Differential Coding for Flight Trajectory Prediction

Lan Wu[1] **Xuebin Wang** [*1] **Ruijuan Chu**[1] **Guangyi Liu**[1] **Jing Zhang**[1] **Linyu Wang** [†1]

[1]Information Engineering University, Zhengzhou, Henan, China

Abstract

Accurate multi-step flight trajectory prediction plays an important role in Air Traffic Control, which can ensure the safety of air transportation. Two main issues limit the flight trajectory prediction performance of existing works. The first issue is the negative impact on prediction accuracy caused by the significant differences in data range. The second issue is that real-world flight trajectories involve underlying temporal dependencies, and most existing methods fail to reveal the hidden complex temporal variations and extract features from one single time scale. To address the above issues, we propose FlightPatchNet, a multi-scale patch network with differential coding for flight trajectory prediction. Specifically, FlightPatchNet first utilizes differential coding to encode the original values of longitude and latitude into first-order differences and generates embeddings for all variables at each time step. Then, global temporal attention is introduced to explore the dependencies between different time steps. To fully explore the diverse temporal patterns in flight trajectories, a multi-scale patch network is delicately designed to serve as the backbone. The multi-scale patch network exploits stacked patch mixer blocks to capture inter- and intra-patch dependencies under different time scales, and further integrates multi-scale temporal features across different scales and variables. Finally, FlightPatchNet ensembles multiple predictors to make direct multi-step prediction. Extensive experiments on ADS-B datasets demonstrate that our model outperforms the competitive baselines.

[*]coauthor

[†]corresponding author

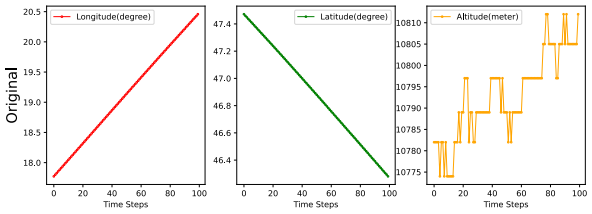
1 INTRODUCTION

Flight Trajectory Prediction (FTP) is an essential task in the Air Traffic Control (ATC) procedure, which can be applied to various scenarios such as air traffic flow prediction [Abadi et al., 2015, Lin et al., 2019], aircraft conflict detection [Chen et al., 2020], and arrival time estimation [Wang et al., 2018]. Accurate FTP can ensure the safety of air transportation and improve real-time airspace management [Lin et al., 2020, Shi et al., 2021]. Generally, FTP tasks can be divided into three categories: long-term [Jeong et al., 2017, Runle et al., 2017], medium-term [Yuan et al., 2016, Chen et al., 2016], and short-term [Huang et al., 2017, Duan et al., 2018]. Among them, short-term trajectory prediction has the greatest impact on ATC and is increasingly in demand for air transportation. In this paper, we mainly focus on the short-term FTP task, which aims to predict future flight trajectories based on historical observations.

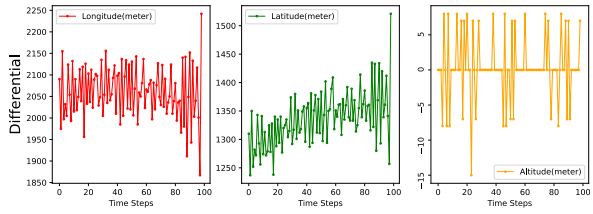
In the ATC domain, multi-step trajectory prediction can provide more practical applications than single-step prediction [Lin et al., 2020]. It can be divided into Iterated Multi-Step (IMS) prediction and Direct Multi-Step (DMS) prediction. IMS-based methods [Yan et al., 2013, Zhang et al., 2023, Guo et al., 2023] make multi-step prediction recursively, which learns a single-step model and iteratively applies the predicted values as observations to forecast the next trajectory point. Due to the error accumulation problem and the step-by-step prediction scheme, this type of methods usually fails in multi-step prediction and has poor real-time performance. By contrast, DMS-based methods [Wu et al., 2023b, Guo et al., 2024] can directly generate future trajectory points at once, which can tackle the error accumulation problem and improve prediction efficiency. Therefore, this paper performs the short-term FTP task in DMS way.

However, two main issues are not well addressed in existing works [Yan et al., 2013, Zhang et al., 2023, Wu et al., 2023b, Guo et al., 2024], limiting the trajectory prediction performance. The first issue is the negative impact on prediction accuracy caused by the significant differences in data range.

In general, longitude and latitude are denoted by degree but altitude is by meter. Since one degree is approximately 111 kilometers, the data range of longitude and latitude are extremely different from that of altitude. Some previous works [Shi et al., 2018, Ma and Tian, 2020] directly utilized normalization algorithms to scale variables into the same range, e.g., from 0 to 1. However, the actual prediction errors could be large for FTP tasks when evaluated in raw data range (as shown in Table 2). FlightBERT [Guo et al., 2023] and FlightBERT++ [Guo et al., 2024] proposed binary encoding (BE) representation to convert variables from rounded decimal numbers to binary vectors, which regards the FTP task as multiple binary classification problem. Although BE representation can avoid the vulnerability caused by normalization algorithms, one serious limitation is introduced: a high bit misclassification in binary will lead to a large absolute error in decimal.



(a) original series



(b) first-order difference series

Figure 1: The original and first-order difference series in real-world flight trajectories.

The second issue is that real-world flight trajectories involve underlying temporal dependencies, and most existing methods [Shi et al., 2021, Guo et al., 2023, 2024] fail to reveal the hidden complex temporal variations and extract features from one single time scale. As shown in Figure 1, the original series of longitude and latitude are over-smoothing and obscure abundant temporal variations, which can be observed from the first-order difference series. Besides, the temporal variation patterns of longitude and latitude are quite distinct from those of altitude which have an obvious global trend but suffer from intense local fluctuations. For example, slight turbulence can exert a significant influence on the altitude but produce a negligible effect on the longitude and latitude. A single-scale model cannot simultaneously capture both local temporal details and global trends [Wu et al., 2023a, Wang et al., 2023]. This calls for powerful multi-scale temporal modeling capacity. Furthermore, if the learned multi-scale temporal patterns are simply aggregated, the model is failed to focus on contributed patterns [Chen et al., 2023a]. Meanwhile, it is essential to explore relationships across variables [Zhang and Yan, 2023, Han et al., 2024], e.g., the velocity at current time step directly affects the location at next time step. Thus, scale-wise correlations and inter-variable relationships should be fully considered when modeling the multi-scale temporal patterns.

Based on above analysis, this paper proposes a multi-scale patch network with differential coding (FlightPatchNet) to address above issues. Specifically, we utilize differential coding to encode the original values of longitude and latitude into first-order differences and retain the original values of other variables as inputs. Due to the dependencies between nearby and distant time steps, we introduce global temporal embedding to explore the correlations between time steps. Then, a multi-scale patch network is proposed to enable the ability of powerful and complete temporal modeling. The multi-scale patch network divides the trajectory series into patches of different sizes, and exploits stacked patch mixer blocks to capture global trends across patches and local details within patches. To further promote the multi-scale temporal modeling capacity, a multi-scale aggregator is introduced to capture scale-wise correlations and inter-variable relationships. Finally, FlightPatchNet ensembles multiple predictors to make direct multi-step forecasting, which can benefit from complementary multi-scale temporal features and improve the generalization ability. The main contributions are summarized as follows:

- We utilize differential coding to effectively reduce the differences in data range and reveal the underlying temporal variations in real-world flight trajectories. Our empirical studies show that using differential values of longitude and latitude can greatly improve prediction accuracy.
- We propose FlightPatchNet to fully explore underlying multi-scale temporal patterns. A multi-scale patch network is designed to capture inter- and intra-patch dependencies under different time scales, and integrate multi-scale temporal features across scales and variables.
- We conduct extensive experiments on a real-world dataset. The experiment results demonstrate that our proposed model significantly outperforms the most competitive baselines.

2 RELATED WORK

Kinetics-and-Aerodynamics Methods The Kinetics-and-Aerodynamics methods [Thipphavong et al., 2013, Benavides et al., 2014, Soler et al., 2015, Tang et al., 2015] divide the entire flight process into several phases, and establish

motion equations for each phase to formulate the flight status. For example, Wang et al. [2009] adopted basic flight models to construct horizontal, vertical, and velocity profiles based on the characteristics of different flight phases. Zhou et al. [2016] combined the dynamics-and-kinematics models and grayscale theory to predict future trajectories. The grayscale theory can address the parameter missing problem in dynamics-and-kinematics models and improve the prediction performance. Due to numerous unknown and time-varying flight parameters of aircraft, these fixed-parameter methods cannot accurately describe the flight status, leading to poor performance and limited application scenarios.

State-Estimation Methods The Kalman Filter and its variants [Xi et al., 2008, Yan et al., 2013] are the typical single-model state-estimation algorithms for FTP tasks, which applies the predefined state equations to estimate the next flight status based on the current observation. For example, Xi et al. [2008] applied the Kalman Filter to track discrete flight trajectories by calculating a continuous state transition matrix. However, single-model algorithms cannot adapt to the complex ATC environment. To address this issue, Interactive Multi-Model algorithms [Hwang et al., 2003, Li and Jilkov, 2005] have been proposed and successfully applied for trajectory analysis. Although multi-model algorithms can achieve better prediction performance, the computational complexity is high and cannot satisfy the real-time requirement.

Deep Learning Methods With the rapid development of deep learning, there has been a surge of deep learning methods for FTP task [Xu et al., 2021, Pang et al., 2022, Sahadevan et al., 2022, Zhang et al., 2023, Guo et al., 2023, 2024]. These learning-based approaches can extract high-dimensional features from raw data, which have achieved a more magnificent performance compared to previous methods. For example, Sahadevan et al. [2022] used a Bi-directional Long-Short-Term-Memory (Bi-LSTM) network to explore both forward and backward dependencies in the sequential trajectory data. Zhang et al. [2023] proposed a wavelet transform-based framework (WTFTP) to perform time-frequency analysis of flight patterns for trajectory prediction. FlightBERT [Guo et al., 2023] employed binary encoding to represent the attributes of the trajectory points and considered the FTP task as a multi-binary classification problem. However, these works predict the future trajectory recursively and suffer from serious error accumulation. Recently, FlightBERT++ [Guo et al., 2024] has been introduced for DMS prediction, which considers the prior horizon information and directly predicts the differential values between adjacent points.

3 METHODOLOGY

The short-term FTP task can be formulated as a Multivariate Time Series (MTS) forecasting problem. Given a sequence of historical observations $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_L\} \in \mathbb{R}^{C \times L}$, where C is the state dimension, L is the look-back window size and $\mathbf{x}_t \in \mathbb{R}^{C \times 1}$ is the flight state at time step t. The task is to predict future T time steps $\hat{\mathbf{Y}} = \{\hat{\mathbf{x}}_{L+1}, ..., \hat{\mathbf{x}}_{L+T}\} \in \mathbb{R}^{C' \times T}$, where C' is the predicted state dimension. In this work, the flight state $\mathbf{x}_t$ represents longitude, latitude, altitude, and velocities along the previous three dimensions, i.e., $\mathbf{x}_t = (Lon_t, Lat_t, Alt_t, Vx_t, Vy_t, Vz_t)^\top$.

The overall architecture of **FlightPatchNet** is shown in Figure 2, which consists of Global Temporal Embedding, Multi-Scale Patch Network, and Predictors. **Global Temporal Embedding** first utilizes differential coding to transform the original values of longitude and latitude into first-order differences and embeds all variables of the same time step into temporal tokens. Global temporal attention is then introduced to capture the inherent dependencies between different tokens. **Multi-Scale Patch Network** is proposed to serve as the backbone which is composed of stacked patch mixer blocks and a multi-scale aggregator. Stacked patch mixer blocks divide trajectory series into patches of different sizes from large scale to small scale. Based on divided patches, each patch mixer block exploits a patch encoder and decoder to capture inter- and intra-patch dependencies, endowing our model with powerful temporal modeling capability. To further integrate multi-scale temporal patterns, a multi-scale aggregator is incorporated into the network to capture scale-wise correlations and inter-variable relationships. **Predictors** provide direct multi-step trajectory forecasting and each predictor is a fully connected network. All the predictor results are aggregated to reconstruct the final prediction trajectory.

3.1 GLOBAL TEMPORAL EMBEDDING

Differential Coding In the context of the WGS84 Coordinate System, the longitude and latitude are limited to the intervals $[-180°, 180°]$ and $[-90°, 90°]$ respectively, while the altitude can span from 0 up to tens of thousands of meters. The significant differences of data range caused by physical units may impair the trajectory prediction performance. Generally, normalization algorithms are applied to address this issue. However, the normalized prediction errors should be transformed into raw data range to evaluate the actual performance. For example, if the absolute prediction error of longitude is 10^{-4} after using the Min-Max normalization algorithm, the actual prediction error is $0.036°$ (approximately 4000 meters). Moreover, as shown in Figure 1, the original series of longitude and latitude are over-smoothing and reflect the overall flight trend over a period. If temporal patterns are learned from the original values of longitude and latitude, the model fails to explore the implicit semantic information and cannot focus on short-term temporal variations in flight trajectories.

To address the above issues, we utilize first-order differences

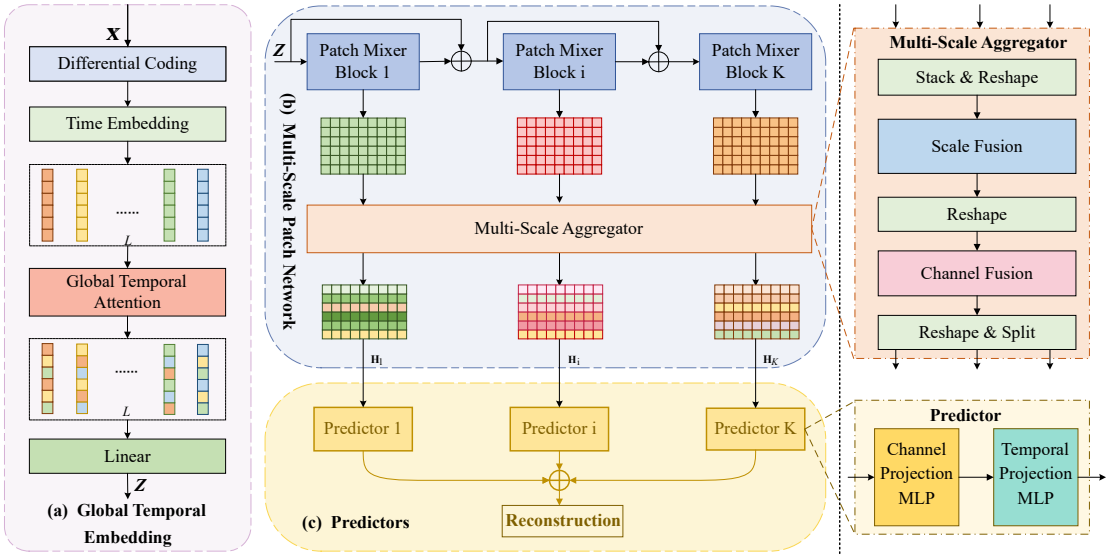


Figure 2: FlightPatchNet architecture. (a) Global Temporal Embedding to explore the correlations between different time steps. (b) Multi-Scale Patch Network to capture inter- and intra-patch dependencies under different time scales and integrate temporal features across scales and variables. (c) Predictors to exploit complementary temporal features and make direct multi-step prediction.

for longitude and latitude while original values for other variables, then the differential values are transformed into meters. This process can be formulated as:

$$\begin{cases} \Delta_{Lon} = 2R \times arcsin(\sqrt{cos^2(\varphi_{t-1}) sin^2(\frac{\phi_t - \phi_{t-1}}{2})}) \\ \Delta_{Lat} = 2R \times arcsin(\sqrt{sin^2(\frac{\varphi_t - \varphi_{t-1}}{2})}) \end{cases} \tag{1}$$

where ϕ denotes the longitude, φ denotes the latitude, and R is the radius of the earth. By using differential coding for longitude and latitude, the differences in data range are effectively reduced. For example, in our dataset, the range of latitude in original data is about $[-46°, 70°]$ and that in differential data is about $[-3860m, 3860m]$, which spans a similar data range as the altitude. Compared to the original sequences, the differential series can explicitly reflect the underlying temporal variations, which is essential for short-term temporal modeling. Besides, adopting the first-order differences instead of second- or higher-order differences enables the model to reconstruct the predicted trajectory based on the last observation. Note that we utilize the original values of altitude as inputs rather than differential values. One important reason is that altitude is more susceptible to noise, failing to reflect the actual temporal variations. To this end, the flight state at time step t becomes $\mathbf{x}_t = (\Delta_{Lon}, \Delta_{Lat}, Alt_t, Vx_t, Vy_t, Vz_t)^\top$.

Global Temporal Attention Given the trajectory series $\mathbf{X} \in \mathbb{R}^{C \times L}$, we first project flight state at each time step into d dimension to generate temporal embeddings $\mathbf{T}^0 \in \mathbb{R}^{L \times d}$. Then, we apply multi-head self-attention (MSA) [Vaswani et al., 2017] on the dimension L to capture the dependencies across all time steps. After attention, the embedding at each time step is enriched with temporal information from other time steps. This process is formulated as:

$$\begin{aligned} \mathbf{T}^0 &= TimeEmbedding(\mathbf{X}^\top) \\ \mathbf{T}^i &= LayerNorm(\mathbf{T}^{i-1} + MSA(\mathbf{T}^{i-1}), i = 1, \dots, l \\ \mathbf{T}^i &= LayerNorm(\mathbf{T}^i + FC(\mathbf{T}^i), i = 1, \dots, l \\ \mathbf{Z} &= (Linear(\mathbf{T}^l))^\top \end{aligned} \tag{2}$$

where l is the number of attention layers, $LayerNorm$ denotes the layer normalization [Ba et al., 2016] which has been widely adopted to address non-stationary issues, MSA is the multi-head self-attention layer, FC denotes a fully-connected layer and $Linear$ projects the embedding of each time step to dimension C, i.e., $\mathbb{R}^d \to \mathbb{R}^C$.

3.2 MULTI-SCALE PATCH NETWORK

Considering different temporal patterns prefer diverse time scales, the multi-scale patch network first utilizes a stack of K patch mixer blocks to capture underlying temporal patterns from large scale to small scale. A large time scale can reflect the slow-varying flight trends, while a smaller scale can retain fine-grained local details. To further promote the collaboration of diverse temporal features, a multi-scale aggregator is introduced to consider the contributed scales and dominant variables. Such a multi-scale network equips our model with powerful and complete temporal modeling capability and helps preserve all kinds of multi-scale characteristics.

3.2.1 Patch Mixer Block

Patching Only considering one single time step is insufficient for the FTP task, since it contains limited semantic information and cannot accurately reflect the flight trajectory variations. Inspired by PatchTST [Nie et al., 2023], the trajectory representation $\mathbf{Z} \in \mathbb{R}^{C \times L}$ is segmented into several non-overlapping patches along the temporal dimension, generating a sequence of patches $\mathbf{Z}_p \in \mathbb{R}^{C \times P \times N}$, where P is the length of each patch, N represents the number of patches, and $N = \lceil \frac{L}{P} \rceil$. The patching process is formulated as:

$$\mathbf{Z}_p = Reshape(ZeroPadding(\mathbf{Z})) \tag{3}$$

where $ZeroPadding(\cdot)$ refers to padding series with zeros in the beginning to ensure the length is divisible by P.

Patch Encoder-Decoder Based on the divided patches $\mathbf{Z}_p$, we utilize a patch encoder and decoder to capture temporal features in flight trajectories. Specifically, the patch encoder aims to capture the inter-patch features (i.e., the global correlations across patches) and intra-patch features (i.e., the local details within patches). After that, these features are reconstructed to the original dimension by the patch decoder. Due to the superiority of linear models for MTS [Chen et al., 2023b, Zeng et al., 2023], the patch encoders and decoders are based on pure multi-layer perceptron (MLP) for temporal modeling.

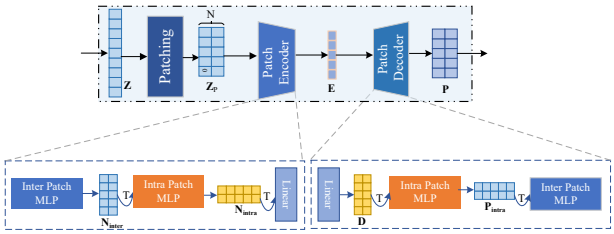


Figure 3: The structure of Patch Mixer Block

As illustrated in Figure 3, a patch encoder consists of an inter-patch MLP, an intra-patch MLP, and a linear projection. Each MLP has two fully connected layers, a GELU non-linearity layer and a dropout layer with a residual connection.

Given the patch-divided series $\mathbf{Z}_p$, an inter-patch MLP performs on the dimension N to capture the dependencies between different patches, which maps $\mathbb{R}^N \to \mathbb{R}^N$ to obtain the inter-patch mixed representation $\mathbf{N}_{inter} \in \mathbb{R}^{C \times P \times N}$:

$$\mathbf{N}_{inter} = \mathbf{Z}_p + Dropout(FC(\sigma(FC(\mathbf{Z}_p)))) \tag{4}$$

where σ denotes a GELU non-linearity layer, $Dropout$ denotes a dropout layer and $\mathbf{N}_{inter}$ reflects the global correlations across patches. After that, an intra-patch MLP performs on the dimension P to capture the dependencies across different time steps within patches, which maps $\mathbb{R}^P \to \mathbb{R}^P$ to obtain the intra-patch mixed representation $\mathbf{N}_{intra} \in \mathbb{R}^{C \times N \times P}$:

$$\mathbf{N}_{intra} = \mathbf{N}_{inter}^{\top} + Dropout(FC(\sigma(FC(\mathbf{N}_{inter}^{\top})))) \tag{5}$$

where $\mathbf{N}_{intra}$ reflects the local details between different time steps within patches. Then, we perform a linear projection on $\mathbf{N}_{intra}^{\top}$ to obtain the final inter- and intra-patch mixed representation $\mathbf{E} \in \mathbb{R}^{C \times P \times 1}$:

$$\mathbf{E} = Linear(\mathbf{N}_{intra}^{\top}) \tag{6}$$

After such a patch encoding process, the correlations between nearby time steps within patches and distant time steps across patches are finely explored. Then, we utilize a patch decoder to reconstruct the original sequence. A patch decoder comprises the same components as the encoder in a reverse order, which is formulated as follows:

$$\begin{aligned} \mathbf{D} &= Linear(\mathbf{E}) \\ \mathbf{P}_{intra} &= \mathbf{D}^{\top} + Dropout(FC(\sigma(FC(\mathbf{D}^{\top})))) \\ \mathbf{P} &= \mathbf{P}_{intra}^{\top} + Dropout(FC(\sigma(FC(\mathbf{P}_{intra}^{\top})))) \end{aligned} \tag{7}$$

where $Linear$ makes a dimensional projection to obtain $\mathbf{D} \in \mathbb{R}^{C \times P \times N}$ for reconstructing the original sequence, $\mathbf{P}_{intra} \in \mathbb{R}^{C \times N \times P}$ is the reconstructed intra-patch mixed representation, and $\mathbf{P} \in \mathbb{R}^{C \times P \times N}$ is the final reconstructed intra- and inter-patch mixed representation.

3.2.2 Multi-Scale Aggregator

To enable the ability of more complete multi-scale modeling, we introduce a multi-scale aggregator to integrate different temporal patterns. It contains two components: scale fusion and channel fusion. Scale fusion can figure out critical time scales and capture the scale-wise correlations, while channel fusion can discover dominant variables affecting temporal variations and explore the inter-variable relationships. These two components work together to help the model learn a robust multi-scale representation and improve generalization ability. Given the K scale-specific temporal representations $\{\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_K\}$, we first stack them and rearrange the data to combine the three dimensions of channel size C, patch size P and patch quantity N, resulting in $\mathbf{S}^0 \in \mathbb{R}^{K \times (C \times L)}$, where $L = P \times N$. Then we apply MSA on the scale dimension K to learn the importance of contributed time scales. This process is formulated as:

$$\begin{aligned} \mathbf{S}^0 &= Reshape(Stack(\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_K)) \\ \mathbf{S}^i &= LayerNorm(\mathbf{S}^{i-1} + MSA(\mathbf{S}^{i-1})) \\ \mathbf{S}^i &= LayerNorm(\mathbf{S}^i + FC(\mathbf{S}^i), i = 1, \ldots, l \end{aligned} \tag{8}$$

where $\mathbf{S}^l$ is the final multi-scale fusion representation within variables. Inspired by iTransformer [Liu et al., 2023], we consider each variable as a token and apply MSA to explore dependencies between different variables. We first reshape

the $\mathbf{S}^l$ to get $\mathbf{C}^0 \in \mathbb{R}^{C\times(K\times L)}$ and perform multi-head self-attention on the channel dimension C to identify dominant variables. This process is simply formulated as follows:

$$\begin{aligned}\mathbf{C}^0 &= Reshape(\mathbf{S}^l)\\ \mathbf{C}^i &= LayerNorm(\mathbf{C}^{i-1} + MSA(\mathbf{C}^{i-1}))\\ \mathbf{C}^i &= LayerNorm(\mathbf{C}^i + FC(\mathbf{C}^i), i = 1, \ldots, l\\ \mathbf{H} &= Reshape(\mathbf{C}^l)\end{aligned} \tag{9}$$

where $\mathbf{H} \in \mathbb{R}^{C\times L\times K}$ is the final multi-scale representation which involves cross-scale correlations and inter-variable relationships.

3.3 DIRECT MULTI-STEP PREDICTION

We ensemble K predictors to directly obtain the future flight trajectory series, which can exploit complementary information from different temporal patterns. The objective of our model is to predict the differential values of longitude and latitude relative to the last observation, and the raw absolute values of altitude, i.e., $\hat{\mathbf{Y}} = \{\hat{\mathbf{x}}_{L+1}, ..., \hat{\mathbf{x}}_{L+T}\}$, where $\hat{\mathbf{x}}_{L+i} = (\hat{\Delta}^{Lon}(L+i, L), \hat{\Delta}^{Lat}(L+i, L), \hat{Alt}_{L+i})^\top$ for $i = 1, \ldots, T$. We split the final multi-scale representation $\mathbf{H} \in \mathbb{R}^{C\times L\times K}$ into a sequence $\{\mathbf{H}_{*,1}, \mathbf{H}_{*,2}, \ldots, \mathbf{H}_{*,K}\}$, where $\mathbf{H}_{*,i} \in \mathbb{R}^{C\times L}$ for $i = 1, \ldots, K$, and feed each $\mathbf{H}_{*,i}$ to a predictor. Each predictor has two MLPs. The first MLP_{C_i} transforms the input channel C into the output channel C', and the second MLP_{T_i} projects the historical input sequence L to the prediction horizon T.

$$\begin{aligned}\hat{\mathbf{Y}}_i &= MLP_{T_i}(MLP_{C_i}(\mathbf{H}_{*,i}))\\ \hat{\mathbf{Y}} &= \sum_{i=1}^{K} \hat{\mathbf{Y}}_i\end{aligned} \tag{10}$$

Finally, all the predictor results are aggregated to reconstruct the final prediction trajectory according to Equation (1), which can enhance the stability and generalization of our model.

4 EXPERIMENTS

4.1 DATASET AND EXPERIMENTAL SETUP

Datasets To evaluate the performance of FlightPatchNet, we conduct extensive experiments on ADS-B data provided by OpenSky [1] from 2020 to 2022. In this paper, six key attributes are extracted from the original data, including longitude, latitude, altitude, and velocity in *x, y, z* dimensions. The dataset is chronologically divided into three parts for training, validation, and testing with a ratio of 8:1:1.

[1] https://opensky-network.org/datasets/states/

Baselines and Setup We compare our model with ten competitive models, including five IMS-based models: **LSTM** [Shi et al., 2018], **CNN-LSTM** [Ma and Tian, 2020], **Bi-LSTM** [Sahadevan et al., 2022], **FlightBERT** [Guo et al., 2023], **WTFTP** [Zhang et al., 2023]; five DMS-based model: **FlightBERT++** [Guo et al., 2024], **TimeMixer** [Wang et al., 2024], **TimesNet** [Wu et al., 2023a], **MICN** [Wang et al., 2023], **Pathformer** [Chen et al., 2024]. These models have covered mainstream deep learning architectures, including Transformer (FlightBERT, FlightBERT++, Pathformer), CNN (CNN-LSTM, TimesNet, MICN), RNN (LSTM, Bi-LSTM, CNN-LSTM, WTFTP) and MLP (TimeMixer), which helps to provide a comprehensive comparison. For fairness, all the models follow the same experimental setup with lookback window $L = 60$ and prediction horizon $T \in \{1, 3, 9, 15\}$. Our model is trained with MSE loss, using the Adam optimizer. We adopt the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) as evaluation metrics. More details about the dataset, baselines, implementation and hyper-parameters are shown in Appendix A.

4.2 MAIN RESULTS

Comprehensive flight prediction results are demonstrated in Table 1 (see Appendix B.2 for error bar). FlightPatchNet achieves the most outstanding performance across various prediction lengths for longitude and latitude in terms of both MAE and RMSE, while it does not achieve the optimal results for altitude compared with other strong baselines such as FlightBERT++. For simplification, we consider prediction horizon $T = 15$ and compare our model with the second best. FlightPatchNet achieves an overall 18.62% reduction on MAE and 41.29% reduction on RMSE for longitude, and 35.31% reduction on MAE and 44.80% reduction on RMSE for latitude. For the prediction performance of altitude, FlightBERT++ outperforms our model by 45.51 meters reduction on MAE but has a large RMSE which may caused by high-bit errors in the prediction. FlightPatchNet obtains the smallest RMSE results for all variables, indicating that our model can provide a more robust and stable prediction. Compared with the most promising multi-scale modeling MTS prediction methods, including the pure MLP-based model TimeMixer, the CNN-based methods TimesNet and MICN and the Transformer-based method Pathformer, FlightPatchNet achieves superior prediction performance. These existing multivariate time-series forecasting methods typically decompose time series into seasonal and trend components, and primarily focus on periodic modeling. However, short-term flight trajectories do not exhibit obvious periodic patterns. The trend features in altitude and the temporal variations in longitude and latitude deserve more attention. Furthermore, as the prediction horizon increases, IMS-based models suffer from serious performance degradation due to error accumulation.

Table 1: Flight trajectory prediction results. A lower MAE or RMSE represents a better prediction. The prediction horizon $T \in \{1, 3, 9, 15\}$ and look-back window size $L = 60$ for all experiments. The best results are highlighted in **bold** and the second best are underlined. Note that 0.00001° is about 1m.

	Model	Metric	Lon(0.00001°)				Lat(0.00001°)				Alt(m)			
			1	3	9	15	1	3	9	15	1	3	9	15
IMS	LSTM	MAE	56	427	1747	3132	49	493	2116	3717	92.27	159.30	549.55	882.86
		RMSE	95	691	2597	4578	89	740	2956	5143	142.05	233.39	763.84	768.45
	Bi-LSTM	MAE	155	747	2319	3890	137	824	2711	4404	432.50	761.50	1648.68	2006.21
		RMSE	202	1124	3387	5532	181	1142	3639	5982	563.74	953.37	2132.91	2420.74
	CNN-LSTM	MAE	139	700	2282	4149	131	801	2623	5139	520.03	746.67	1569.68	1136.80
		RMSE	240	1033	3263	5981	212	1130	3559	7353	1176.96	926.40	1936.63	1658.53
	WTFTP	MAE	175	1484	2002	2657	112	1169	1586	2110	145.02	230.49	588.44	957.41
		RMSE	218	1905	2606	3531	171	1739	2328	3124	415.13	497.24	1021.52	1583.38
	FlightBERT	MAE	123	241	1162	2407	88	158	963	1238	24.67	35.67	78.58	134.29
		RMSE	241	526	2189	3969	154	286	1904	3093	234.17	272.59	384.22	462.28
DMS	FlightBERT++	MAE	173	317	871	1187	85	210	612	1048	**9.39**	**21.89**	**47.84**	**78.46**
		RMSE	360	659	1846	3131	148	425	959	2127	175.29	167.16	327.93	384.18
	TimeMixer	MAE	67	765	3100	5581	42	422	1679	3028	21.18	50.02	119.80	157.47
		RMSE	115	1466	5517	9698	76	789	2976	5318	**109.57**	145.55	281.31	374.82
	TimesNet	MAE	73	1383	5459	9421	46	753	2981	5086	36.41	79.42	178.49	257.39
		RMSE	124	2281	8377	14420	83	1237	4557	7796	154.37	184.36	392.23	543.37
	MICN	MAE	69	680	2235	3912	40	384	1296	2256	48.28	544.35	1992.86	3431.19
		RMSE	116	1133	3831	6598	73	633	2167	3704	112.31	945.57	3270.77	5612.64
	Pathformer	MAE	52	232	1374	1914	45	232	863	1806	42.03	47.65	141.08	259.53
		RMSE	89	373	2227	2686	83	373	1346	2570	114.37	136.59	406.37	645.02
	FlightPatchNet (Ours)	MAE	**48**	**153**	**546**	**966**	**32**	**105**	**381**	**678**	13.34	32.65	78.57	123.97
		RMSE	**87**	**233**	**885**	**1577**	**64**	**175**	**652**	**1174**	123.78	**121.48**	**174.63**	**244.34**

Visualization of FlightPatchNet Predictions Figure 4 shows that FlightPatchNet can comprehensively capture the temporal variations of longitude and latitude, while it fails to fully reveal the temporal patterns from original altitude series.

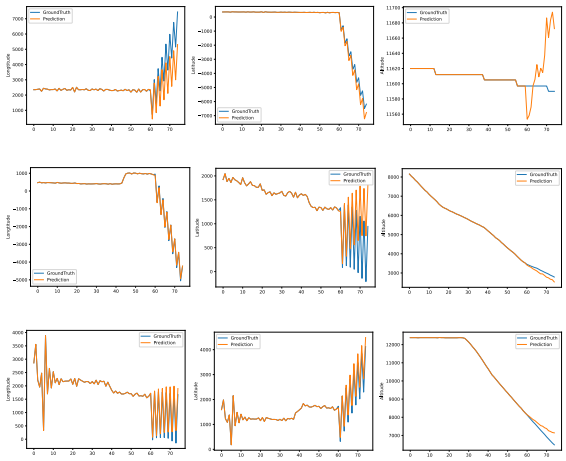


Figure 4: Visualization of the ground truth and predictions of FlightPatchNet when the prediction horizon $T = 15$ and look-back window size $L = 60$. The series of altitude are in original data while those of longitude and latitude are in differential data, all denoted by meters.

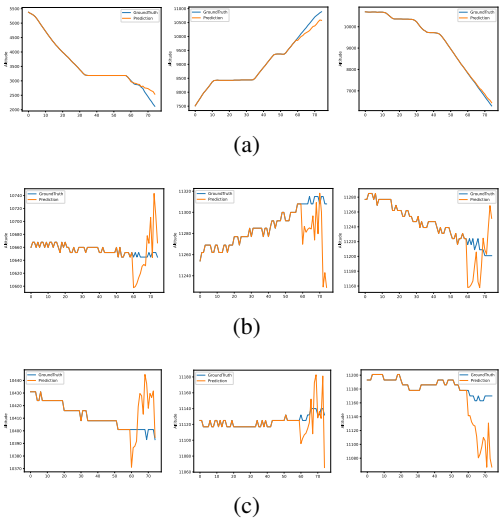


Figure 5: Visualization of ground truth for altitude and predictions of FlightPatchNet when the prediction horizon $T = 15$ and look-back window size $L = 60$, all denoted by meters.

Visualization of FlightPatchNet Predictions for Altitude We present the visualization of FlightPatchNet predictions and ground truth for the altitude in Figure 5. As shown in Figure 5(a), when the series of altitudes are relatively smooth and stationary with obvious global trends, Flight-

PatchNet can effectively capture these trends and make accurate predictions. When the series suffers from many change points caused by frequent abrupt fluctuations, as depicted in Figure 5(b) and 5(c), FlightPatchNet tends to focus more on the irregular change points during prediction, leading to a large deviation from the ground truth. As a result, FlightPatchNet struggles to capture the real temporal variations in altitude and fails to provide accurate predictions.

Effectiveness of Differential Coding The results in Table 2 show that using differential coding for longitude and latitude can significantly improve their prediction performance but slightly decrease the accuracy of altitude. The differential coding can reveal the temporal variations of longitude and latitude, which helps the temporal modeling in flight trajectories. However, the variations of altitude in the original series may come from unexpected noise. FlightPatchNet has a strong modeling capacity for temporal variations and tends to focus more on the noise points during altitude prediction, leading to a large bias towards the ground truth.

Table 2: Flight trajectory prediction results for longitude and latitude in original data and differential data when prediction horizon $T = 15$. The best results are highlighted in **bold**. Note that altitude and velocities are always in original data.

Models	Diff	Metric	Lon(°)	Lat(°)	Alt(m)
LSTM	✓	MAE	0.03132	0.03717	883
		RMSE	0.04578	0.05143	1206
	×	MAE	0.82230	0.12008	769
		RMSE	1.20424	2.44136	1053
Bi-LSTM	✓	MAE	0.03890	0.04404	2006
		RMSE	0.05532	0.05982	2421
	×	MAE	1.71433	0.19014	2091
		RMSE	2.43607	0.27621	2666
CNN-LSTM	✓	MAE	0.04149	0.05139	1137
		RMSE	0.05981	0.07353	1659
	×	MAE	8.59512	1.95957	1638
		RMSE	23.07600	8.15418	2114
FlightPatchNet (Ours)	✓	MAE	**0.00966**	**0.00678**	124
		RMSE	**0.01577**	**0.01174**	244
	×	MAE	0.19348	0.05385	**61**
		RMSE	0.26243	0.07457	**170**

Effectiveness of Multi Scales To investigate the effect of multi-scale modeling, we conduct experiments on single scale for {2,6,10,20,30}. The results in Figure 6, 7 and 8 illustrate the critical contribution of multi scales to our model. We observe that different variables prefer distinct time scales. For instance, a patch size of 10 obtains the second-best prediction performance on longitude and latitude but the worst performance on altitude when prediction horizon $T = 15$. This indicates that longitude, latitude and altitude exhibit distinct temporal patterns, and different scales can extract diverse complementary features, which can be effectively leveraged to obtain competitive and robust prediction performance.

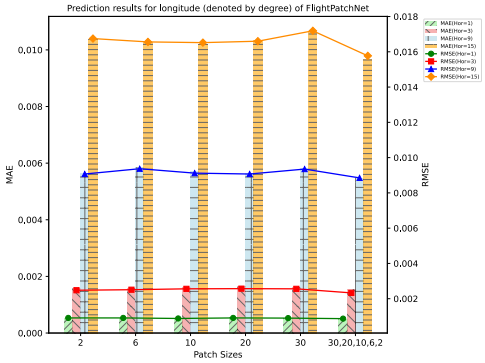


Figure 6: MAE and RMSE of FlightPatchNet for longitude with single scale and multi scales ($L = 60$).

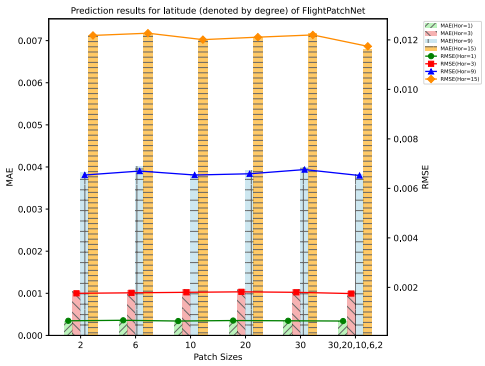


Figure 7: MAE and RMSE of FlightPatchNet for latitude with single scale and multi scales ($L = 60$).

4.3 ABLATION STUDY

We conduct ablation studies by removing corresponding modules from FlightPatchNet. Specifically, **w/o global temporal attention** does not capture the correlations between time steps. **w/o scale fusion** considers each time scale of equal importance. **w/o channel fusion** does not explore the relationships between variables. Table 3 shows the contribution of each component. Removing the global temporal attention dramatically decreases the multi-step prediction performance, demonstrating the necessary of capturing the correlations between different time steps. Scale fusion can effectively improve the prediction accuracy, indicating that different time scales of trajectory series contain rich and diverse temporal variation information. Channel fusion also improves the model performance, suggesting the importance of exploring relationships between different variables in complex temporal modeling.

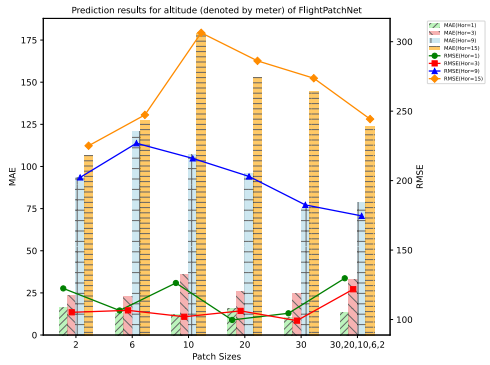


Figure 8: MAE and RMSE of FlightPatchNet for altitude with single scale and multi scales ($L = 60$).

Table 3: Performance comparisons on ablative variants. The best results are highlighted in **bold**. Hor represents the prediction horizon $T \in \{1, 3, 9, 15\}$.

Case	Hor	Lon (0.00001°)		Lat (0.00001°)		Alt (m)	
		MAE	RMSE	MAE	RMSE	MAE	RMSE
w/o global temporal attention	1	51	90	34	66	21	137
	3	190	308	132	222	**28**	112
	9	667	1085	466	791	**67**	**164**
	15	1232	2005	876	1486	**112**	**221**
w/o scale fusion	1	53	92	35	67	24	130
	3	169	268	114	188	33	**100**
	9	609	975	409	688	91	194
	15	1112	1787	759	1280	162	282
w/o channel fusion	1	50	89	34	65	20	160
	3	166	265	112	187	29	122
	9	573	924	398	667	73	174
	15	1059	1707	727	1240	132	250
FlightPatchNet	1	**48**	**87**	**32**	**64**	**13**	**124**
	3	**153**	**233**	**105**	**175**	33	122
	9	**546**	**885**	**381**	**652**	79	175
	15	**966**	**1577**	**678**	**1174**	124	244

4.4 MODEL COMPLEXITY

As shown in Table 4, our proposed FlightPatchNet achieves the greatest efficiency and has relatively fewer parameters compared to other models. For multi-step prediction, the DMS-based models demonstrate significant improvements in computational performance compared to the IMS-based models. In addition, FlightPatchNet is lightweight compared to FlightBERT++ and FlightBERT, which indicates our model can provide a promising solution for real-time air transportation management.

5 CONCLUSION, LIMITATION AND FUTURE WORK

Conclusion In this paper, we propose FlightPatchNet, a multi-scale patch network with differential coding for the short-term FTP task. The differential coding is leveraged to reduce the significant differences in the original data range and reflect the temporal variations in realistic flight trajectories. The multi-scale patch network is designed to explore global trends and local details based on divided patches of different sizes, and integrate scale-wise correlations and inter-variable relationships for complete temporal modeling. Extensive experiments on a real-world dataset demonstrate that FlightPatchNet achieves the most competitive performance and offers a significant reduction in computational complexity, presenting a promising solution for real-time air traffic control applications.

Table 4: Model Complexity Comparisons. The look-back window size $L = 60$ and the prediction horizon $T = 15$ for all models.

Type	Models	Parameters (MB)	FLOPs (M)	Running Time (s/iter)
DMS	FlightPatchNet	5.69	64.38	0.0069
	FlightBERT++	44.26	3000.00	0.0112
	Pathformer	2.64	258.55	0.02174
	TimeMixer	0.45	3115.00	0.0037
	TimesNet	37.50	196159.87	0.0534
	MICN	1.28	1235.83	0.0012
IMS	FlightBERT	25.31	1620.00	0.2406
	LSTM	0.03	1.67	0.0583
	Bi-LSTM	0.51	31.15	0.1241
	CNN-LSTM	0.04	1.22	0.0429
	WTFTP	0.23	60.00	0.0290

Limitation and Future Work The original series of altitude contains many unexpected noises. Our primary focus on modeling temporal variations enables FlightPatchNet to concentrate more on these irregular change points during altitude prediction, leading to a large bias. In the future, we will further explore temporal modeling of altitude and investigate robust noise-handling techniques such as moving average to make the series smoother and less sensitive to noise. In addition, the data-missing problem occurs commonly in realistic flight trajectory series, making downstream analysis difficult. Thus, we attempt to incorporate data imputation methods into our model to enhance the model applicability and provide a general framework for flight trajectory prediction.

References

Afshin Abadi, Tooraj Rajabioun, and Petros A. Ioannou. Traffic flow prediction for road transportation networks with limited traffic data. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):653–662, 2015. doi: 10.1109/TITS.2014.2337238.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint*, 2016. URL `https://arxiv.org/abs/1607.06450`.

Jose V Benavides, John Kaneshige, Shivanjli Sharma, Ramesh Panda, and Mieczyslaw Steglinski. Implementation of a trajectory prediction function for trajectory based operations. In *AIAA Atmospheric Flight Mechanics Conference*, page 2198. AIAA, 2014. doi: 10.2514/6.2014-2198.

Dan Chen, Minghua Hu, Ke Han, Honghai Zhang, and Jianan Yin. Short/medium-term prediction for the aviation emissions in the en route airspace considering the fluctuation in air traffic demand. *Transportation Research Part D: Transport and Environment*, 48:46–62, 2016. doi: 10.1016/j.trd.2016.08.003.

Ling Chen, Donghui Chen, Zongjiang Shang, Binqing Wu, Cen Zheng, Bo Wen, and Wei Zhang. Multi-scale adaptive graph neural network for multivariate time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 35(10):10748–10761, 2023a. doi: 10.1109/TKDE.2023.3268199.

Peng Chen, Yingying Zhang, Yunyao Cheng, Yang Shu, Yihang Wang, Qingsong Wen, Bin Yang, and Chenjuan Guo. Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=lJkOCMP2aW.

Si-An Chen, Chun-Liang Li, Sercan Ö. Arik, Nathanael C. Yoder, and Tomas Pfister. Tsmixer: An all-mlp architecture for time series forecasting. *Transactions on Machine Learning Research*, 2023, 2023b. URL https://openreview.net/forum?id=wbpxTuXgm0.

Zhengmao Chen, Dongyue Guo, and Yi Lin. A deep gaussian process-based flight trajectory prediction approach and its application on conflict detection. *Algorithms*, 13 (11):293, 2020. doi: 10.3390/a13110293.

Peibo Duan, Guoqiang Mao, Weifa Liang, and Degan Zhang. A unified spatio-temporal model for short-term traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*, 20(9):3212–3223, 2018. doi: 10.1109/TITS.2018.2873137.

Dongyue Guo, Edmond Q. Wu, Yuankai Wu, Jianwei Zhang, Rob Law, and Yi Lin. Flightbert: Binary encoding representation for flight trajectory prediction. *IEEE Transactions on Intelligent Transportation Systems*, 24(2):1828–1842, 2023. doi: 10.1109/TITS.2022.3219923.

Dongyue Guo, Zheng Zhang, Zhen Yan, Jianwei Zhang, and Yi Lin. Flightbert++: A non-autoregressive multi-horizon flight trajectory prediction framework. In *AAAI Conference on Artificial Intelligence*, pages 127–134. AAAI, 2024. doi: 10.1609/aaai.v38i1.27763.

Lu Han, Han-Jia Ye, and De-Chuan Zhan. The capacity and robustness trade-off: Revisiting the channel independent strategy for multivariate time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):7129–7142, 2024. doi: 10.1109/TKDE.2024.3400008.

Darong Huang, Zhenping Deng, Ling Zhao, and Bo Mi. A short-term traffic flow forecasting method based on markov chain and grey verhulst model. In *2017 6th Data Driven Control and Learning Systems (DDCLS)*, pages 606–610. IEEE, 2017. doi: 10.1109/DDCLS.2017.8068141.

Inseok Hwang, Jesse Hwang, and Claire Tomlin. Flight-mode-based aircraft conflict detection using a residual-mean interacting multiple model algorithm. In *AIAA guidance, navigation, and control conference and exhibit*, page 5340. AIAA, 2003. doi: 10.2514/6.2003-5340.

Donggi Jeong, Minjin Baek, and Sang-Sun Lee. Long-term prediction of vehicle trajectory based on a deep neural network. In *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 725–727. IEEE, 2017. doi: 10.1109/ICTC.2017.8190764.

X Rong Li and Vesselin P Jilkov. Survey of maneuvering target tracking. part v. multiple-model methods. *IEEE Transactions on aerospace and electronic systems*, 41(4):1255–1321, 2005. doi: 1255-1321.DOI:10.1109/TAES.2005.1561886.

Yi Lin, Jian wei Zhang, and Hong Liu. Deep learning based short-term air traffic flow prediction considering temporal–spatial correlation. *Aerospace Science and Technology*, 93:105–113, 2019. doi: 10.1016/j.ast.2019.04.021.

Yi Lin, Linjie Deng, Zhengmao Chen, Xiping Wu, Jianwei Zhang, and Bo Yang. A real-time atc safety monitoring framework using a deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 21(11):4572–4581, 2020. doi: 10.1109/TITS.2019.2940992.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=JePfAI8fah.

Lan Ma and Shan Tian. A hybrid cnn-lstm model for aircraft 4d trajectory prediction. *IEEE Access*, 8:134668–134680, 2020. doi: 10.1109/ACCESS.2020.3010963.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Yutian Pang, Xinyu Zhao, Jueming Hu, Hao Yan, and Yongming Liu. Bayesian spatio-temporal graph transformer network (b-star) for multi-aircraft trajectory prediction. *Knowledge-Based Systems*, 249:108998, 2022. doi: 10.1016/j.knosys.2022.108998.

Du Runle, Liu Jiaqi, Gao Lu, Li Zhifeng, and Zhang Li. Long term trajectory prediction based on advanced guidance law recognition. In *2017 IEEE International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pages 456–461. IEEE, 2017. doi: 10.1109/MetroAeroSpace.2017.7999617.

Deepudev Sahadevan, Palanisamy Ponnusamy, Varun P Gopi, and Manjunath K Nelli. Ground-based 4d trajectory prediction using bi-directional lstm networks. *Applied Intelligence*, 52(14):16417–16434, 2022. doi: 10.1007/s10489-022-03309-6.

Zhiyuan Shi, Min Xu, Quan Pan, Bing Yan, and Haimin Zhang. Lstm-based flight trajectory prediction. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018. doi: 10.1109/IJCNN.2018.8489734.

Zhiyuan Shi, Min Xu, and Quan Pan. 4-d flight trajectory prediction with constrained lstm network. *IEEE Transactions on Intelligent Transportation Systems*, 22(11):7242–7255, 2021. doi: 10.1109/TITS.2020.3004807.

Manuel Soler, Alberto Olivares, and Ernesto Staffetti. Multiphase optimal control framework for commercial aircraft four-dimensional flight-planning problems. *Journal of Aircraft*, 52(1):274–286, 2015. doi: 10.2514/1.C032697.

Xin-min Tang, Long Zhou, Zhi-yuan Shen, and Miao Tang. 4d trajectory prediction of aircraft taxiing based on fitting velocity profile. In *The 15th COTA International Conference of Transportation Professionals (CICTP)*, pages 1–12. ASCE, 2015. doi: 10.1061/9780784479292.001.

David P Thipphavong, Charles A Schultz, Alan G Lee, and Steven H Chan. Adaptive algorithm to improve trajectory prediction accuracy of climbing aircraft. *Journal of Guidance, Control, and Dynamics*, 36(1):15–24, 2013. doi: 10.2514/1.58508.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *International Conference on Neural Information Processing Systems*, pages 6000–6010. Curran Associates Inc., 2017. doi: 10.5555/3295222.3295349.

Chao Wang, Jiuxia Guo, and Zhipeng Shen. Prediction of 4d trajectory based on basic flight models. *Journal of southwest jiaotong university*, 44(2):295–300, 2009. doi: 10.3969/j.issn.0258-2724.2009.02.028.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. MICN: Multi-scale local and global context modeling for long-term series forecasting. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=zt53IDUR1U.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=7oLshfEIC2.

Zhengyi Wang, Man Liang, and Daniel Delahaye. A hybrid machine learning model for short-term estimated time of arrival prediction in terminal manoeuvring area. *Transportation Research Part C: Emerging Technologies*, 95:280–294, 2018. doi: 10.1016/j.trc.2018.07.019.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=ju_Uqw384Oq.

Han Wu, Yan Liang, Bin Zhou, and Hao Sun. A bi-lstm and autoencoder based framework for multi-step flight trajectory prediction. In *2023 8th International Conference on Control and Robotics Engineering (ICCRE)*, pages 44–50. IEEE, 2023b. doi: 10.1109/ICCRE57112.2023.10155614.

Lin Xi, Zhang Jun, Zhu Yanbo, and Liu Wei. Simulation study of algorithms for aircraft trajectory prediction based on ads-b technology. In *2008 Asia Simulation Conference-7th International Conference on System Simulation and Scientific Computing*, pages 322–327. IEEE, 2008. doi: 10.1109/ASC-ICSC.2008.4675378.

Zhengfeng Xu, Weili Zeng, Xiao Chu, and Puwen Cao. Multi-aircraft trajectory collaborative prediction based on social long short-term memory network. *Aerospace*, 8(4):115, 2021. doi: 10.3390/aerospace8040115.

Honglei Yan, Genghua Huang, Haiwei Wang, and Rong Shu. Application of unscented kalman filter for flying target tracking. In *2013 International Conference on Information Science and Cloud Computing*, pages 61–66. IEEE, 2013. doi: 10.1109/ISCC.2013.10.

Chengjue Yuan, Dewei Li, and Dewei Xi. Medium-term prediction of urban traffic states using probability tree. In *2016 35th Chinese Control Conference (CCC)*, pages 9246–9251. IEEE, 2016. doi: 10.1109/ChiCC.2016.7554828.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI conference on artificial intelligence*, pages 11121–11128. AAAI, 2023. doi: 10.1609/aaai.v37i9.26317.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=vSVLM2j9eie`.

Zheng Zhang, Dongyue Guo, Shizhong Zhou, Jianwei Zhang, and Yi Lin. Flight trajectory prediction enabled by time-frequency wavelet transform. *Nature Communications*, 14(1):5258, 2023. doi: 10.1038/s41467-023-40903-9.

Zhijing Zhou, Jinliang Chen, Beibei Shen, Zhigang Xiong, Hua Shen, and Fangyue Guo. A trajectory prediction method based on aircraft motion model and grey theory. In *2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, pages 1523–1527. IEEE, 2016. doi: 10.1109/IMCEC.2016.7867472.

APPENDIX

A EXPERIMENTAL DETAILS

A.1 DATASET PREPROCESSING AND DESCRIPTION

This paper exploits real-world datasets provided by OpenSky from 2020 to 2022 to validate our proposed model. The data preprocessing steps are as follows:

(1) Data Extraction: We extract seven features from the raw data, including timestamp, longitude, latitude, altitude, horizontal flight speed, horizontal flight angle, and vertical speed. The timestamp is used to identify whether the trajectory points are continuous, and the other six features are further processed as inputs to the model.

(2) Data Filtering: Due to many missing values and outliers in the raw dataset, we select 100 consecutive points without missing values as a complete flight trajectory. Then, we adopt the z-score method to find out the outliers. If one flight trajectory contains any outliers, we discard the whole trajectory. The z-score formula is as follows:

$$z = \frac{(\overline{x} - \mu)}{\sigma - \sqrt{n}} \tag{1}$$

where $\overline{x}$ is the value of each feature point, μ is the mean of each feature, σ is the variance of each feature, and n is the number of feature points.

(3) Velocity Transformation: We transform the horizontal velocity into V_x and V_y according to the angle, where V_x is the velocity in the longitude dimension and V_y is the velocity in the latitude dimension. In this way, the features become longitude, latitude, altitude, V_x, V_y and V_z.

(4) Data Segmentation: The dataset is randomly divided into three parts with a ratio of 8:1:1 for training, validation, and testing.

After the above preprocessing, 274,605 flight trajectories are selected into our dataset. The range of longitude, latitude and altitude are $[-179.86396°, 178.82147°]$, $[-46.42435°, 70.32590°]$ and $[0, 21031.00m]$, respectively. The interval between two adjacent flight trajectory points is 10 seconds.

A.2 BASELINE METHODS

We briefly describe the selected 10 competitive baselines as follows:

- LSTM [Shi et al., 2018]: Based on two layers of LSTM (with 30 and 60 nodes respectively) to encode each trajectory point, and future trajectories are predicted through a fully connected layer.
- CNN-LSTM [Ma and Tian, 2020]: Based on two layers of one-dimensional CNN (the convolution kernel size is 1×3) and two layers of LSTM (with 50 nodes) to encode each trajectory point, and future trajectories are predicted through a fully connected layer.
- Bi-LSTM [Sahadevan et al., 2022]: Based on two layers of Bi-LSTM (with 200 and 50 nodes respectively) to encode each trajectory point, and future trajectories are predicted through a fully connected layer.
- FlightBERT [Guo et al., 2023]: It utilizes a BE representation to convert the scalar attributes of the flight trajectory into binary vectors, considering the FTP task as a multi binary classification problem. It uses 18, 16, 11 and 11 bits to encode the real values (decimals) of longitude, latitude, altitude and velocities into BE representation respectively.
- FlightBERT++ [Guo et al., 2024]: It inherits the BE representation from the FlightBERT and introduces a differential prediction paradigm, which aims to predict the differential values of the trajectory attributes instead of the absolute values.
- WTFTP [Zhang et al., 2023]: It is an IMS-based method that utilizes discrete wavelet transform (DWT) to decompose the input flight trajectory into wavelet coefficients and predicts future trajectories based on the generated wavelet coefficients by inverse discrete wavelet transform (IDWT).
- TimeMixer [Wang et al., 2024]: It is a fully MLP-based architecture that mixes the decomposed seasonal and trend components in fine-to-coarse and coarse-to-fine directions separately and ensembles multiple predictors to utilize complementary forecasting capabilities in multi-scale observations.

- TimesNet [Wu et al., 2023a]: It is a task-general foundational model for time series analysis, which disentangles complex temporal variations into multiple intra-period and inter-period variations. A parameter efficient inception block is employed to capture these temporal variations in 2D space.
- MICN [Wang et al., 2023]: It adopts a multi-scale branch structure to capture the underlying information in time series. Downsampling one-dimensional convolution is used for local feature extraction and isometric convolution is employed for global correlation discovery.
- Pathformer [Chen et al., 2024]: It is a multi-scale Transformer with adaptive pathways, which integrates both temporal resolution and temporal distance for multi-scale modeling. A multi-scale router with temporal decomposition and an aggregator work together to realize adaptive multi-scale modeling for time series.

A.3 IMPLEMENTATION DETAILS

For fairness, all the models follow the same experimental setup with look-back window $L = 60$ and prediction horizon $T \in \{1, 3, 9, 15\}$, which means the observation time is 10 minutes and the forecasting time is 10 seconds, 30 seconds, 1.5 minutes, 2.5 minutes. The patch sizes in multi-scale patch mixer blocks are set to {30, 20, 10, 6, 2}. The dimension of temporal embedding d is 128. For all the MSA in this paper, the head number is 8 and the attention layer l is 3. The learning rate is set as 10^{-4} for all experiments. Our method is trained with MSE loss, using the Adam optimizer. The training process is early stopped within 30 epochs. The training would be terminated early if the validation loss does not decrease for three consecutive rounds. The model is implemented in PyTorch 2.2.1 and trained on a single NVIDIA RTX 3090 GPU with 24GB memory.

A.4 EVALUATION METRICS

Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) are exploited to evaluate the proposed model and baselines, which are defined as:

$$MAE = \frac{1}{T}\sum_{i=1}^{T}|\mathbf{Y}_i - \hat{\mathbf{Y}}_i|$$

$$RMSE = \sqrt{\frac{1}{T}\sum_{i=1}^{T}(\mathbf{Y}_i - \hat{\mathbf{Y}}_i)^2}$$

where $\mathbf{Y}_i$, $\hat{\mathbf{Y}}_i$ are the ground truth and prediction result for i-th future point, respectively.

B ADDITIONAL EXPERIMENTAL RESULTS

B.1 HYPER-PARAMETER SENSITIVITY

Number of Scales We perform experiments on different number of scales and report the MAE and RMSE results. As shown in Figure 9, we can observe that when the number of scales increases from 2 to 5, the performance of FlightPatchNet is constantly improved. This is because FlightPatchNet can capture diverse global and local temporal patterns under different scales. When the number of scales increases up to 6, the performance starts to deteriorate. This indicates that a certain number of scales is sufficient for temporal modeling, and excessive scales may lead to the overfitting problem.

Number of Attention Layers We test the number of attention layers in $\{1, 2, 3, 6\}$ for global temporal attention, scale fusion, and channel fusion. The results are shown in Figure 10(a), Figure 10(b) and Figure 10(c). We can observe that when the number of attention layers increases from 1 to 3, the values of MAE and RMSE decrease, demonstrating that our model can better capture the dependencies between different time steps, scale-wise correlations and inter-variable relationships with more layers of attention. When the number of attention layers increases up to 6, the prediction accuracy does not improve. Thus, we choose to use three layers of attention in these parts.

Look-Back Window Size L Figure 11 demonstrates the MAE and RMSE results of our model with different look-back window sizes. We set the window size L to $\{10, 20, 30, 40, 50, 60, 70, 80\}$. The overall performance of FlightPatchNet is significantly improved as the window size increases from 10 to 60, indicating that FlightPatchNet can thoroughly capture

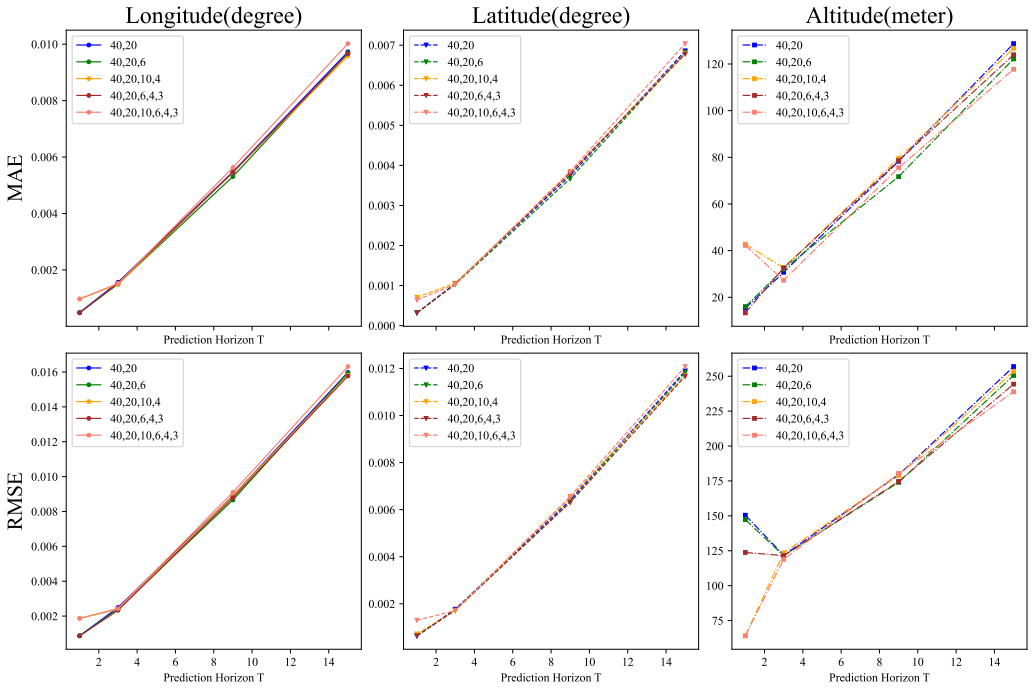
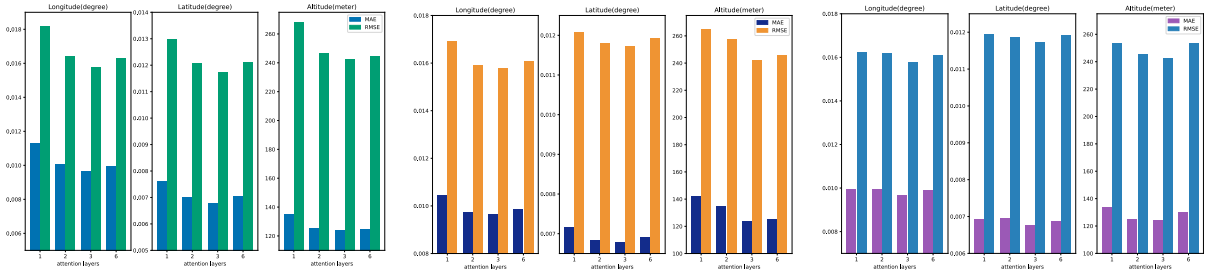


Figure 9: MAE and RMSE with different number of scales for prediction horizon $T \in \{1, 3, 9, 15\}$.



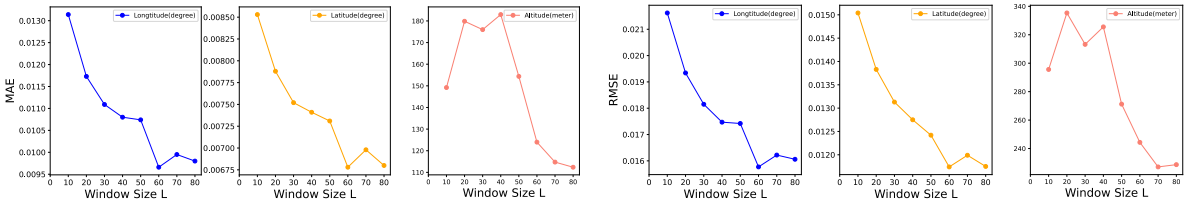
(a) MAE and RMSE of different attention layers in global temporal attention.

(b) MAE and RMSE of different attention layers in scale fusion.

(c) MAE and RMSE of different attention layers in channel fusion.

Figure 10: MAE and RMSE of different attention layers for prediction horizon $T = 15$

the temporal dependencies from long flight trajectories. Moreover, the performance of altitude fluctuates with the increase of the window size, suggesting that the series of altitude are non-stationary and easily affected by unexpected noise. Thus, we set L as 60 to achieve the overall optimal performance.



(a) (b)

Figure 11: MAE and RMSE of different look-back window sizes L for prediction horizon $T = 15$.

Order of Scales We conduct experiments on the order of patch sizes and report the MAE and RMSE results. As shown in Table 5, we can observe that patch sizes in descending order can effectively improve the prediction performance, indicating that the macro knowledge from coarser scales can guide the temporal modeling of finer scales.

Table 5: The results of flight trajectory prediction with scales in ascending and descending order. ↑ means scales in ascending order and ↓ means scales in descending order. The better results are highlighted in **bold**.

patch sizes	Lon(0.00001°)						Lat(0.00001°)				Alt(m)			
	Style	Horizon	1	3	9	15	1	3	9	15	1	3	9	15
2,6,10,20,30	↑	MAE	98	155	548	1008	99	106	385	697	54.02	33.47	79.39	127.51
		RMSE	187	241	887	1642	131	183	656	1197	**81.92**	**110.68**	184.54	248.16
	↓	MAE	**48**	**153**	**546**	**966**	**32**	**105**	**381**	**678**	**13.34**	**32.65**	**78.57**	**123.97**
		RMSE	**87**	**233**	**885**	**1577**	**64**	**175**	**652**	**1174**	129.65	121.78	**174.63**	**244.34**
3,4,6,20,40	↑	MAE	98	155	556	997	64	105	383	704	**39.77**	32.15	**76.38**	**124.18**
		RMSE	188	247	901	1631	131	175	655	1210	64.86	124.20	177.46	**243.46**
	↓	MAE	**97**	**153**	**542**	**963**	**63**	**104**	**369**	**670**	43.46	**28.96**	79.27	128.13
		RMSE	**187**	**245**	**879**	**1582**	**130**	**174**	**631**	**1167**	**64.50**	**115.76**	**176.96**	251.72
3,6,40	↑	MAE	48	156	536	994	35	105	370	691	**14.96**	31.42	79.07	**117.43**
		RMSE	87	248	876	1628	65	176	634	1193	**107.43**	118.36	177.40	238.87
	↓	MAE	**48**	**153**	**534**	**988**	**33**	**103**	**368**	**685**	16.81	**31.26**	**71.96**	118.66
		RMSE	**87**	**244**	**870**	**1620**	**64**	**173**	**633**	**1186**	145.25	**114.33**	**175.63**	**236.32**

B.2 ERROR BAR

In this paper, we repeat all the experiments five times. Here we report the standard deviation of our model and the second best model in Table 6.

C 3D TRAJECTORY VISUALIZATION

We visualize the flight trajectory prediction results of FlightPatchNet and all the baselines when the prediction horizon is 15. As shown in Figure 12, FlightPatchNet can provide stable and the most accurate predictions in longitude and latitude while it suffers from slight fluctuations in altitude.

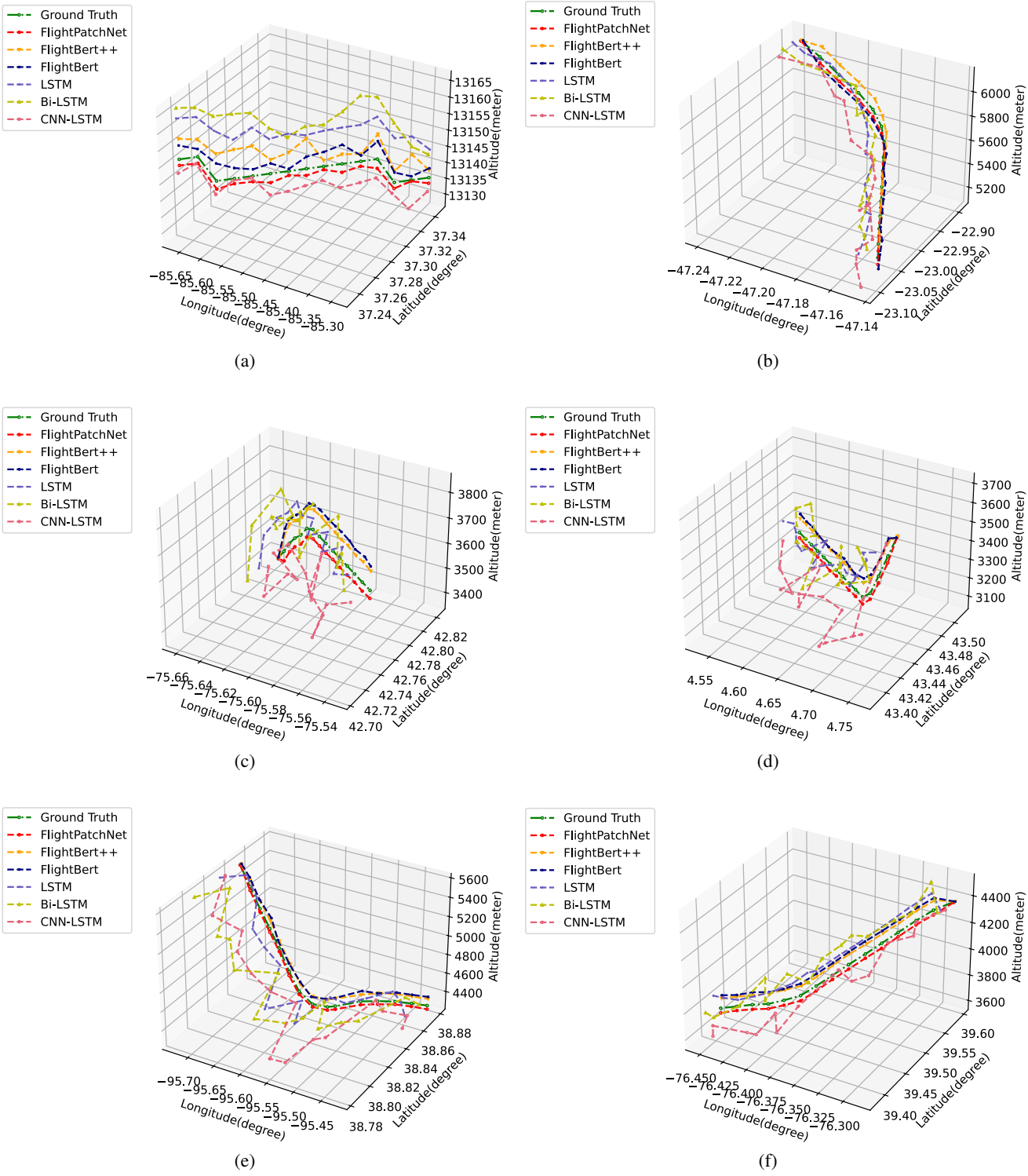


Figure 12: Visualization of flight trajectory prediction results when the prediction horizon $T = 15$ and look-back window size $L = 60$.

Table 6: Error bar of our FlightPatchNet and the second best model FlightBERT++.

Model	Horizon	Lon(0.00001°)		Lat(0.00001°)		Alt(m)	
		MAE	RMSE	MAE	RMSE	MAE	RMSE
FlightBERT++	1	173±6.45	360±8.28	85±3.33	148±13.20	9.39±1.79	175.29±29.09
	3	317±26.10	659±4.35	210±30.50	425±12.40	21.89±5.58	167.16±46.39
	9	871±17.40	1846±44.50	612±60.70	959±21.90	47.84±2.87	327.93±52.84
	15	1187±5.91	3131±53.30	1048±36.90	2127±20.10	78.46±8.13	384.18±51.82
FlightPatchNet (Ours)	1	48±1.24	87±1.02	32±1.06	64±0.84	13.34±9.43	123.78±15.13
	3	153±3.19	233±5.44e-4	105±1.19	175±2.36	32.65±1.76	121.48±2.81
	9	546±15.40	885±21.60	381±7.47	652±7.76	78.57±2.66	174.63±6.87
	15	966±36.50	1577±54.90	678±25.80	1174±35.30	123.97±5.72	244.34±6.91