AeroGTO: An Efficient Graph-Transformer Operator for Learning Large-Scale Aerodynamics of 3D Vehicle Geometries

Pengwei Liu[1*], Pengkai Wang[1*], Xingyu Ren[1*], Hangjie Yuan[2†], Zhongkai Hao[3], Chao Xu[1], Shengze Cai[1†], Dong Ni[4†]

[1]College of Control Science and Engineering, Zhejiang University, Hangzhou, China,
[2]College of Computer Science and Technology, Zhejiang University, Hangzhou, China,
[3]Department of Computer Science and Technology, Tsinghua University, Beijing, China,
[4]State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou, China,
{liupw, 0923b09, 12332063, hj.yuan, cxu, shengze_cai, dni}@zju.edu.cn, hzj21@mails.tsinghua.edu.cn

Abstract

Obtaining high-precision aerodynamics in the automotive industry relies on large-scale simulations with computational fluid dynamics, which are generally time-consuming and computationally expensive. Recent advances in operator learning for partial differential equations offer promising improvements in terms of efficiency. However, capturing intricate physical correlations from extensive and varying geometries while balancing large-scale discretization and computational costs remains a significant challenge. To address these issues, we propose **AeroGTO**, an efficient graph-transformer operator designed specifically for learning large-scale aerodynamics in engineering applications. AeroGTO combines local feature extraction through message passing and global correlation capturing via projection-inspired attention, employing a frequency-enhanced graph neural network augmented with k-nearest neighbors to handle three-dimensional (3D) irregular geometries. Moreover, the transformer architecture adeptly manages multi-level dependencies with only linear complexity concerning the number of mesh points, enabling fast inference of the model. Given a car's 3D mesh, AeroGTO accurately predicts surface pressure and estimates drag. In comparisons with five advanced models, AeroGTO is extensively tested on two industry-standard benchmarks, Ahmed-Body and DrivAerNet, achieving a 7.36% improvement in surface pressure prediction and a 10.71% boost in drag coefficient estimation, with fewer FLOPs and only 1% of the parameters used by the prior leading method.

Code — https://github.com/pengwei07/AeroGTO

Introduction

Large-scale numerical simulations of aerodynamics enhance our understanding of the physical world's natural characteristics, which are indispensable for advancing engineering design and related disciplines, particularly in the automotive industry where aerodynamic drag prediction is crucial for vehicle performance and design (Elrefaie, Dai, and Ahmed 2024). Traditionally, methods such as Direct Numerical Simulation (DNS) (Lee and Moser 2015) solve the complete Navier-Stokes equations across all scales but require significant computational resources, which are time-consuming and costly. Simplified turbulence models like Large Eddy Simulation (LES) (Piomelli 1999), Reynolds-Averaged Navier-Stokes (RANS) (Alfonsi 2009), and hybrid RANS-LES (Heinz 2020) approaches have been proposed to improve the computational efficiency. Nonetheless, these approaches require a trade-off between accuracy and computational expedience, posing a challenge for rapid design iterations.

Recent strides in machine learning present promising alternatives to traditional computational fluid dynamics (CFD) simulators, offering significantly accelerated predictions (Kochkov et al. 2021; Vinuesa and Brunton 2022; Liu et al. 2024). Among these, operator learning, which encompasses the acquisition of mappings between infinite-dimensional function spaces, demonstrates remarkable adaptability to unseen system inputs, exhibiting inherent flexibility in a manner that is independent of the discretization (Kovachki et al. 2023).

By developing a tailored operator framework for automotive dynamics, it is possible to reveal a universal mapping from complex automotive geometries to fundamental physical variables, such as surface pressure and drag. Nevertheless, traditional neural operator structures (Li et al. 2020a) encounter difficulties in effectively managing automobiles' intricate and irregular geometries, thereby restricting the model's expressive capacity. On the other hand, Graph Neural Networks (GNNs) (Pfaff et al. 2020; Cao et al. 2023) and various Transformer models (Hao et al. 2023; Lee and Oh 2024; Wu et al. 2024; Li, Shu, and Barati Farimani 2024) have showcased potential in enhancing computational efficiency for aerodynamic simulations by capturing complex data relationships.

However, these models face significant challenges when directly applied to large-scale aerodynamics in automotive design. For traditional GNNs, increasing graph size introduces two primary issues: (1) **Complexity**: as both the number of nodes and the message-passing iterations increase linearly, the computational graph's time and memory complex-

*These authors contributed equally.
†Corresponding author.

Copyright © 2025, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

ity inevitably become quadratic (Xu et al. 2018; Cao et al. 2023; Janny et al. 2023). (2) **Oversmoothing**: graph convolution acts as a low-pass filter, suppressing higher-frequency signals. Consequently, stacked MPs iteratively project information onto the graph's eigenspace, smoothing out these signals, thereby complicating training (Yu, Zhang, and Qin 2022). For traditional Transformer models, data points are projected into latent spaces and then processed by the attention block. However, using a Multilayer Perceptron (MLP) alone to learn complex relationships in large-scale point clouds of automotive geometries can lose critical topological information, limiting its ability to capture complex physical correlations.

To address the issues of representing complex physical correlations while balancing large-scale discretization and computational costs, we propose AeroGTO, an efficient graph-transformer operator specifically designed for large-scale aerodynamic learning in engineering applications. Given the 3D mesh of a car geometry, AeroGTO can accurately predict the 3D surface pressure and estimate drag.

AeroGTO mitigates the limitations of traditional GNNs in capturing fine details within complex geometries by employing a frequency-enhanced GNN augmented with k-nearest neighbors (kNN) for precise local feature extraction. Additionally, the model integrates a Transformer with global linear-complexity attention, which captures long-range dependencies between mesh points and facilitates multi-level interactions between local and global intricate physical correlations. This design effectively reduces costs for complex datasets, enabling rapid and accurate inference. We conduct extensive experiments on two industry-standard car benchmarks with various geometries, demonstrating the remarkable performance of AeroGTO.

Overall, the main contributions of this work include:

- To tackle the challenges of representing complex physical correlations in large-scale aerodynamics, we present a frequency-enhanced GNN augmented with kNN specifically designed for precise local feature extraction. This method effectively isolates and captures physical information of points and edges and projects this spatial data into a topologically structured hidden space.
- We depart from the standard Transformer architecture by introducing a global Transformer with linear projection-inspired attention. This innovation captures long-range dependencies between mesh points and enables multi-level interactions between intricate local and global physical correlations.
- We conduct numerical experiments on two industry-standard benchmarks, Ahmed-Body and DrivAerNet, showcasing that AeroGTO reduces error by 7.36% on average while requiring fewer FLOPs and using only 1% of the parameters compared to the leading method. Additionally, AeroGTO improves the prediction accuracy of the pressure drag coefficient on the unseen Ahmed-Body test set, with a 10.71% performance boost and an overall R^2 of 0.9250.

Related Work

Geometric Deep Learning Geometric deep learning (GDL) leverages neural network architectures to handle irregular geometries (Atz, Grisoni, and Schneider 2021). PointNet (Qi et al. 2017a) and PointNet++ (Qi et al. 2017b) are classical methods designed to handle point cloud data effectively. Graph neural networks (GNNs) utilize kernels on connected graphs for representation learning, making them suitable for various applications, including those in physics, biology, and more (Belbute-Peres, Economon, and Kolter 2020). Specifically, MeshGraphNet (MGN) (Pfaff et al. 2020) is a classical GNN based on multiple chained message-passing layers. In addition, EAGLE (Janny et al. 2023) clusters nodes directly for global attention, limiting its power to capture local information effectively.

Neural PDE Solvers Partial Differential Equations (PDEs) serve as foundational mathematical models that describe various phenomena across science and engineering. In recent years, machine learning-based methods for solving PDEs have become a hot topic. Physics-Informed Neural Networks (PINNs) (Raissi, Perdikaris, and Karniadakis 2019) are a promising class of models that integrate the principles of supervised learning with the underlying physics governed by nonlinear PDEs. However, these models require adaptation when the parameter space of a PDE changes, limiting their generalizability.

Neural operators are developed to address these limitations by approximating the input-output mappings in PDE-learning tasks. A notable advancement in this domain is the kernel-based neural operators by Li et al. (Li et al. 2020b), which laid the groundwork for subsequent models such as UNO and UFNO. However, these models struggle to process irregular mesh data directly. To overcome this issue, the GINO model was introduced by Li et al. (Li et al. 2024), providing an effective solution for irregular inputs.

Moreover, Transformer is gradually being adapted as neural operators in PDE learning to handle large-scale data, using the linear (Shen et al. 2021) or bilinear attentions (Kim, Jun, and Zhang 2018) as the backbone, exemplified by models like GNOT (Hao et al. 2023), IPOT (Lee and Oh 2024) and Transolver (Wu et al. 2024), showcasing its ability to handle intricate data structures.

Methodology

Our proposed AeroGTO is an efficient graph-transformer operator designed for learning large-scale aerodynamics in engineering applications. This section outlines its architecture, starting with the problem definition, followed by a model description, and concluding with a complexity analysis.

Problem Setting and Notations

Neural operators of PDEs are the mappings from the input functions, such as initial/boundary condition, geometry, coefficient, and source fields, to the solutions. Let $\mathcal{A}$ represent the space of input functions, the solution space is $\mathcal{S}$, and the neural operator is to learn an operator $\mathcal{G} : \mathcal{A} \to \mathcal{S}$. Here, we focus on learning an operator $\mathcal{G}$ with neural networks to

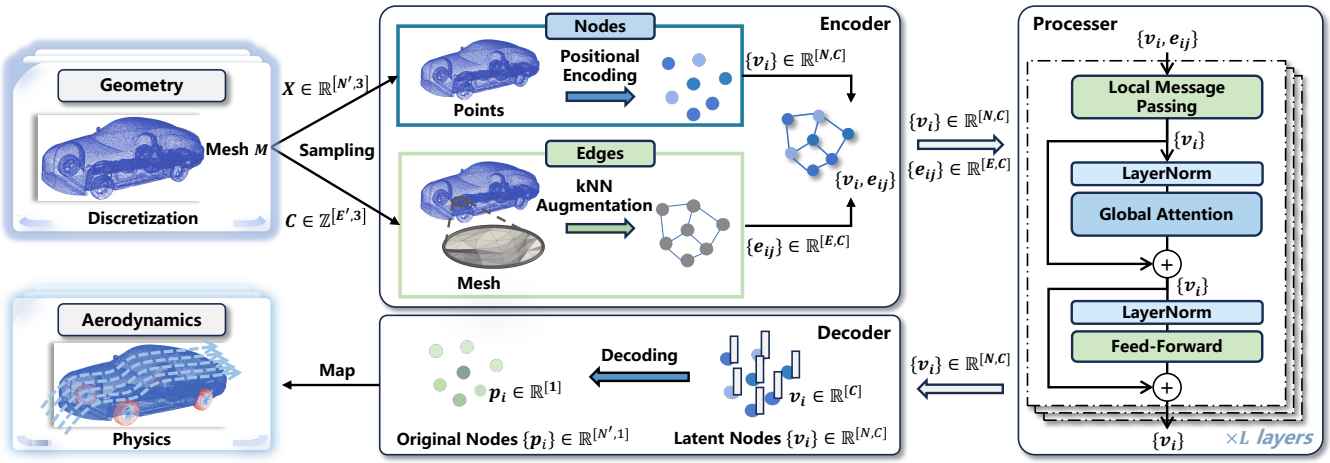

Figure 1: The overall pipeline of AeroGTO, an efficient graph-transformer operator for learning large-scale aerodynamics of 3D vehicle geometries. Taking triangular cells as an example, given the 3D mesh of a car geometry, AeroGTO can accurately predict its surface pressure.

map the arbitrary shape of a car and the given conditions to its surface pressure.

For a given k-th car shape, the input $\boldsymbol{a}^k$ consists of the shape of a car, which is discretized with a set of meshes $\boldsymbol{M}^k = (\boldsymbol{X}^k, \boldsymbol{C}^k)$, where $\boldsymbol{X}^k = \{(x_i^k, y_i^k, z_i^k)\}_{i=1}^{N'}$ is the coordinates of all nodes, and $\boldsymbol{C}^k = \{(i_l^k, \ldots, j_l^k)\}_{l=1}^{E'}$ is the cells, representing the connections of the nodes in each cell, where $1 \le i_l^k, \ldots, j_l^k \le N'$ denote the index of nodes, E' is the number of cells, N' is the number of nodes. Moreover, the conditions $\boldsymbol{A}^k \in \mathbb{R}^m$ contains some global design parameters, such as the length, width, and height of the car, Reynolds number, inlet velocity, etc. The expected output is the surface pressure discretized as $\boldsymbol{s}^k = \{\boldsymbol{p}_i^k\}_{i=1}^{N'}$, where $\boldsymbol{p}_i^k \in \mathbb{R}^1$ is the pressure corresponding to the i-th node.

In a nutshell, like most methods in neural operators, such as GINO (Li et al. 2024) and Transolver (Wu et al. 2024), for modeling this operator $\mathcal{G}(\cdot)$, we use a parameterized neural network $\hat{\mathcal{G}}_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$, inputs $\boldsymbol{a}^k = (\boldsymbol{M}^k, \boldsymbol{A}^k)$, and outputs $\hat{\mathcal{G}}_{\boldsymbol{\theta}}(\boldsymbol{a}^k) = \hat{\boldsymbol{s}}^k$, where $1 \le k \le D$. Therefore, our goal is to minimize the L_2 relative error loss between the prediction $\hat{\boldsymbol{s}}^k$ and real data $\boldsymbol{s}^k$ in the training dataset as,

$$\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \frac{1}{D}\sum_{k=1}^{D} \mathcal{L}_k(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \frac{1}{D}\sum_{k=1}^{D} \frac{\| \boldsymbol{s}^k - \hat{\boldsymbol{s}}^k \|_2}{\| \boldsymbol{s}^k \|_2}, \quad (1)$$

where $\mathcal{L}_k(\boldsymbol{\theta})$ is the mean L_2 relative error loss of index k. D is the size of the training dataset, $\boldsymbol{\theta}$ is a set of the network parameters, and $\boldsymbol{\Theta}$ is the parameter space.

Model Architecture

As illustrated in Fig. 1, AeroGTO combines local feature extraction through message passing and global correlation capturing via projection-inspired attention, which can be split into three key components: encoder, processor, and decoder.

Encoder The whole encoder is divided into two parts: the encoding of nodes and edges. The inputs contain the discretized mesh $\boldsymbol{M} = (\boldsymbol{X}, \boldsymbol{C})$ and the condition $\boldsymbol{A}$.

Sampling To reduce computational complexity when processing large-scale mesh data, we propose two distinct sampling strategies for nodes and edges, each designed for specific scenarios.

Edge-Focused Sampling Here, we begin with the cells $\boldsymbol{C}$, which form the bidirectional mesh edges $\boldsymbol{E}^M$. To approximate bidirectional edges while maintaining computational efficiency, we first randomly shuffle the connection directions by swapping the indices i and j of the edges. Then, we uniformly sample a proportion $\alpha_E \in (0, 1]$ of these shuffled edges for further processing. After this sampling, k-nearest neighbors (kNN) (Franceschi et al. 2019) augmentation is applied to establish extra connections that the original mesh structure may not have. This method is particularly suitable when cell information is available and edge relationships can be easily extracted.

Node-Focused Sampling We randomly sample the nodes, selecting a proportion $\alpha_N \in (0, 1]$ of the total nodes. After sampling, edges are constructed solely using the kNN method, focusing on the relationships among the selected nodes. This approach is more appropriate for cases where the original mesh is highly complex, making extracting edge relationships from the cell information tricky.

Nodes There are two parts of information for nodes: the coordinates of each point, $\boldsymbol{X}_i = (x_i, y_i, z_i)$, and the condition of the car, $\boldsymbol{A}$. Then we encode nodes i from the physical space to the latent space using MLP Φ_1^V as, $\mathbf{v}_i = \Phi_1^V(\boldsymbol{X}_i, \boldsymbol{A})$. Moreover, to effectively capture spatial and frequency-related characteristics of the coordinates, we employ a sinusoidal positional encoding (SPE) (Janny et al. 2023) defined as

$$F(\boldsymbol{X}) = \left[\cos\left(2^i\pi\boldsymbol{X}\right), \sin\left(2^i\pi\boldsymbol{X}\right)\right]_{i=-\delta,\ldots,\delta}, \delta \in \mathbb{Z}^+, \quad (2)$$

where combining cosine and sine functions at varying frequencies can enhance the understanding and differentiating of the positional context within the data, enhancing its abil-

ity to capture intricate physical correlations. Next, we use MLP Φ_2^V, defined as $\mathbf{v}_i = \Phi_2^V(\mathbf{v}_i, F(\boldsymbol{X}_i))$, with the same size as Φ_1^V of C.

Edges Geometric topology conveyed solely by point positions poorly represents the geometry (Pfaff et al. 2020; Cao et al. 2023), making edge encoding essential for enhancing the model's geometric representation ability. We can obtain mesh edges $\boldsymbol{E}^M$ and k-nearest neighbors edges $\boldsymbol{E}^k$ from the process of sampling. Next, we encode the relative displacement vector $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ and its norm $|\mathbf{x}_{ij}|$ into the combined set of edges $\mathbf{e}_{ij} \in \boldsymbol{E}^M \cup \boldsymbol{E}^k$. The concatenated features from both mesh edges and kNN edges are then encoded into a latent vector of size C for each edge, using the encoder MLPs Φ^E, as $\mathbf{e}_{ij} = \Phi^E(\mathbf{e}_{ij})$.

Finally, we can obtain the latent representation of a car geometry, $(\boldsymbol{V} = \{\mathbf{v}_i\}, \boldsymbol{E} = \{\mathbf{e}_{ij}\})$, where $\boldsymbol{V} \in \mathbb{R}^{[N,C]}$ and $\boldsymbol{E} \in \mathbb{R}^{[E,C]}$, and N, E are the number of nodes and edges after sampling, respectively. In the whole encoder, Φ_1^V, Φ_2^V, and Φ^E are implemented with two linear layers with SiLU activation function MLPs of the same size C.

Processor AeroGTO combines local feature extraction through message passing and global correlation capturing via projection-inspired attention.

Local Message Passing Each mesh edge $\mathbf{e}_{ij}$ and node $\mathbf{v}_i$ are updated with frequency enhancement as follows,

$$\begin{cases} \mathbf{e}_{ij} = \mathbf{e}_{ij} + \Phi_P^E(\mathbf{e}_{ij}, \mathbf{v}_i, \mathbf{v}_j), \\ \mathbf{v}_i = \mathbf{v}_i + \Phi_P^V(\mathbf{v}_i, \sum_j \mathbf{e}_{ij}, F(\boldsymbol{X}_i)), \end{cases} \tag{3}$$

where Φ_P^E, Φ_P^V are the GELU-activation-based MLPs with the residual connections of the same size C.

Remark 1. ***Local Topology Representation and Over-smoothing Challenges*** *The message-passing process enables each node to capture local topology by representing the geometric structure formed with its first-order and beyond neighbors. While stacking multiple message-passing layers extends these relationships to more distant nodes (Pfaff et al. 2020), it increases computational cost and exacerbates over-smoothing, where node representations become indistinguishable, complicating training.*

Global Attention via Projection-Inspired Attention To model the varying relationships between local regions while managing computational complexity, as shown in Fig. 2, we introduce projection-inspired attention with linear complexity. Drawing inspiration from the projection theorem in Hilbert space, we propose identifying a learned projection space, updating abstract relationships, and then projecting the results to the original latent space, as follows,

$$\begin{cases} W_1 = \text{softmax}\left(\dfrac{Q_{ls} W_0^T}{\sqrt{C}}\right) W_0, \\ W_2 = \text{softmax}\left(\dfrac{W_1 W_1^T}{\sqrt{C}}\right) W_1, \\ W_3 = \text{softmax}\left(\dfrac{W_0 W_2^T}{\sqrt{C}}\right) W_2, \end{cases} \tag{4}$$

where Q_{ls} is initialized as $Q_{ls} \sim \mathcal{N}(0, 1)$. Starting with the local representation $W_0 := \{\mathbf{v}_i\} \in \mathbb{R}^{[N,C]}$ obtained from the MP block, we use a set of learned subspace queries $Q_{ls} \in \mathbb{R}^{[M,C]}, M \ll N$, where M represents the number of tokens, and C is the dimensionality of each token. Here, each token in Q_{ls} can be interpreted as a basis vector of the learned subspace, representing specific patterns or features within the data. This process uses the dot-product attention mechanism to calculate attention scores between Q_{ls} and the original local representation W_0. These scores project the original data into the learned subspace defined by Q_{ls}, producing an intermediate state $W_1 \in \mathbb{R}^{[M,C]}$. Subsequent experimental results confirm that the tokens indeed form a basis, with the M vectors being linearly independent. Then, the representation within the subspace is refined further, updating the state to $W_2 \in \mathbb{R}^{[M,C]}$. Finally, by skipping the original input as the query in the last projection step, we effectively combine the local updates W_0 with the global context refinement W_2, yielding $W_3 \in \mathbb{R}^{[N,C]}$.

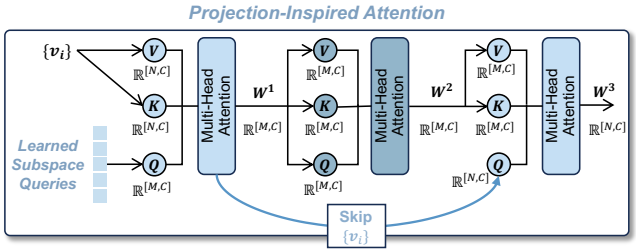

Figure 2: Detailed architecture of the proposed projection-inspired attention.

Pre-Norm Structure We adopt a pre-norm structure to facilitate more efficient computation throughout the Processor as follows:

$$\begin{cases} V_G^1 = W_0 + W_3, \\ V_G^2 = V_G^1 + \text{FFN}(\text{LN}(V_G^1)). \end{cases} \tag{5}$$

Residual connections and layer normalization are applied to further enhance training stability and support the construction of deeper networks. Here, V_G^2 represents the output of the entire Processor, FFN denotes the feed-forward network, and LN stands for layer normalization, which stabilizes the training process by normalizing the inputs to each layer.

In practice, we enhance the model's ability to capture various interactions and dependencies across different heads by introducing multi-head attention, thereby improving its capacity to represent complex relationships within the data. Moreover, by stacking multiple process blocks, the model leverages the collective effect of L layers, enabling more effective multi-level interactions between intricate local and global physical correlations.

Decoder The refined local and global features are re-projected into the original physical space. We continue employing the frequency-enhanced approach by using an MLP, denoted as Φ_D^V, to transform the latent node features $\mathbf{v}_i$ after the final processing step into physical features $\boldsymbol{p}_i$, as expressed by:

$$\boldsymbol{p}_i = \Phi_D^V(\mathbf{v}_i, F(\boldsymbol{X}_i)), \tag{6}$$

where $F(\boldsymbol{X}_i)$ is the sinusoidal positional encoding. Here $\mathbf{v}_i \in \mathbb{R}^{[C]}$ is the output of the Processor, and $\boldsymbol{p}_i \in \mathbb{R}^{[1]}$ is the output pressure at sampled node i.

Inference During inference, different sampling methods need to be considered. For the Edge-Focused sampling method, the number of nodes remains unchanged throughout the pipeline (i.e., $N = N'$), while the edges are sampled according to the setting strategy. For the Node-Focused, the full-size mesh is randomly divided into $\left\lceil \frac{1}{\alpha_N} \right\rceil$ batches based on the sampling ratio $\alpha_N \in (0, 1]$. AeroGTO then leverages the parallel computing power of the GPU to process these batches together. The physics of the sampled geometry is inferred and aggregated to produce the full-mesh output.

In summary, AeroGTO effectively integrates local feature extraction through message passing and global correlation capturing via global attention. It incorporates a frequency-enhanced GNN with k-nearest neighbors for irregular geometries and a global Transformer with projection-inspired attention, capturing long-range dependencies and enabling multi-level interactions between local and global physical correlations.

Model Complexity Analysis

In analyzing the overall computational complexity of the AeroGTO model, we focus on two main components: message-passing and global attention. The message-passing layer has a complexity of $O(N \times d \times \alpha)$, where d is the maximum graph degree, and $\alpha = \max(\alpha_N, \alpha_E/2)$ is a sampling factor that accounts for two distinct sampling strategies. The global attention has a complexity of $O(2 \times N \times M \times C + M^2 \times C)$, where N is the number of mesh points, M is the number of tokens, and C is the feature dimensionality. Since $M \leq N$, the complexity of global attention scales linearly with the number of mesh points N. Therefore, the total complexity of AeroGTO is $O(N \times \beta)$, where $\beta = d \times \alpha + 2 \times M \times C + M^2 \times C/N$. Given that $M \ll N$, the overall complexity for large-scale aerodynamic simulations remains efficient.

Experiments

To validate the effectiveness of AeroGTO for complex and large-scale aerodynamics learning tasks in automotive applications, we conduct several experiments on two challenging industry-standard benchmarks, the Ahmed-Body dataset (Li et al. 2024) and the DrivAerNet dataset (Elrefaie, Dai, and Ahmed 2024). These datasets include diverse 3D vehicle geometries with Reynolds numbers reaching up to five million.

Experimental Setup and Evaluation Metrics

Datasets. We briefly introduce the datasets used in the experiments. **Ahmed-Body** dataset (Li et al. 2024) is an industry-standard simulation of vehicle aerodynamics based on Ahmed-body shapes (Ahmed, Ramm, and Faltin 1984), containing 0.12 million surface mesh faces. It contains 551 shapes, with 500 allocated for training and 51 for testing. **DrivAerNet** dataset (Elrefaie, Dai, and Ahmed 2024) is another high-fidelity dataset with thousands of 3D car meshes, featuring 0.5 million surface mesh faces—60% larger than the previously available largest public car dataset. Due to the computational cost, we randomly selected 550 cars, using 500 for training and 50 for testing.

Baselines. We compare AeroGTO with five competitive baselines in these two car datasets. **MeshGraphNet** (MGN) (Pfaff et al. 2020) is a classical GNN-based model that relies on multiple chained message-passing layers. **GNOT** (Hao et al. 2023) is a scalable transformer framework with linear attention and mixture-of-experts (Fedus, Zoph, and Shazeer 2022). Both **Transolver** (Wu et al. 2024) and **IPOT** (Lee and Oh 2024) are transformer variants for geometric learning operators with low computational complexity. **GINO** (Li et al. 2024) is a novel neural operator for large-scale complex simulation benchmarks with FNO (Li et al. 2020a) blocks.

Evaluation metrics. According to Eq. 1, we adopt the mean L_2 relative error (L_2) as our evaluation metric. L_2 is formulated as follows:

$$L_2 := \frac{1}{D} \sum_{k=1}^{D} \frac{\| \boldsymbol{P}_k - \hat{\boldsymbol{P}}_k \|_2}{\| \boldsymbol{P}_k \|_2}, \tag{7}$$

where D is the size of the test dataset, and $\boldsymbol{P}_k, \hat{\boldsymbol{P}}_k \in \mathbb{R}^{[N,1]}$ are the k-th real and predicted physical variable distributions respectively for N input data nodes.

Evaluation Protocol. For a fair comparison, we maintain the same batch size across all methods for each task. All experiments are conducted on 1 to 3 NVIDIA A40 GPUs with 48GB of memory, except for GINO, which demands significantly more memory. Thus, GINO is executed on a single NVIDIA A100 GPU with 80GB of memory.

Main Results

Given a 3D mesh of car geometry, AeroGTO can accurately predict the 3D surface pressure and estimate the drag. Hereafter, **Bold** and Underline numbers indicate the best and second best performance, respectively.

Pressure Prediction Tab. 1 presents the main results of pressure predictions on test datasets. It contains the mean L_2 relative error (L_2), the number of trainable parameters (N_P), and computational cost (FLOPs) for each baseline method and our approach. Our observations from the results are as follows: **(i)** Our proposed model, **AeroGTO**, achieves

Model	N_P/M↓	FLOPs /GFLOPs↓	L_2 ↓ Ahmed-Body	L_2 ↓ DrivAerNet
MGN (2020)	**2.325**	506.448	0.1388	0.2254
GNOT (2023)	3.657	381.638	0.1293	0.2471
IPOT (2024)	6.731	<u>285.151</u>	0.1003	0.1934
Transolver (2024)	2.837	321.083	0.0927	0.1719
GINO (2024)	230.690	30839.283	<u>0.0831</u>	<u>0.1618</u>
AeroGTO (ours)	<u>2.442</u>	**175.298**	**0.0757**	**0.1524**

Table 1: Main results on two large-scale benchmarks. All errors are de-normalized L_2 errors.

Error	MGN	GNOT	IPOT	Transolver	GINO	**AeroGTO**
MRE ↓	0.1041	0.1002	0.0755	0.0679	0.0532	**0.0475**
R^2 ↑	0.8289	0.8370	0.8865	0.8987	0.9187	**0.9250**

Table 2: Results of C_d predictions on Ahmed-Body dataset.

Config		N_P/M ↓	FLOPs /GFLOPs ↓	L_2 ↓
GNN	no SPE	1.886	127.739	0.0860
	no kNN	2.442	175.298	0.0809
	no MP	**1.451**	86.898	0.1080
Attention	linear	1.914	174.951	0.0953
	no	1.648	147.045	0.1039
Sampling $\alpha = 25\%$	Edge-Focused	2.442	175.298	**0.0757**
	Node-Focused	2.442	**47.684**	0.0896

Table 3: Results of ablation studies on Ahmed-Body dataset.

the most balanced trade-off between performance, parameter efficiency, and computational cost across two large-scale benchmarks. **(ii)** Even with significantly fewer FLOPs and only 1% of the parameters used by the previous leading method, GINO, AeroGTO still outperforms it by a substantial margin. **(iii)** In a nutshell, our model enhances the performance by an average of 7.36% in two tasks, affirming AeroGTO as an efficient model suitable for large-scale datasets.

Drag Prediction In automotive design, another critical metric is the vehicle's drag coefficient, which is inversely related to the vehicle's efficiency in moving through a fluid. For a given car shape with unit density, the pressure drag coefficient (C_d) is defined as:

$$C_d = \frac{2}{v^2 A} \int_S p(x) \left(\mathbf{n}(x) \cdot \mathbf{i}\right) dx, \tag{8}$$

where S denotes the car's surface, $p(x)$ is the pressure, and $\mathbf{n}(x)$ is the outward unit normal vector at point x on the surface, $\mathbf{i}$ is the unit vector in the direction of the inlet flow, v is the speed of the inlet flow, and A is the area of the smallest rectangle enclosing the front of the car.

Using Eq. 8, the pressure C_d can be calculated based on the predicted surface pressure of the car. Taking the Ahmed-Body dataset as an example, Tab. 2 presents the drag prediction results compared with five advanced baselines. In line with (Elrefaie, Dai, and Ahmed 2024), we evaluate the accuracy of pressure drag coefficient predictions using the mean relative error (MRE) and the coefficient of determination (R^2) as performance metrics. Remarkably, compared with the previous leading method, GINO, AeroGTO improves the average relative error by 10.71%, achieving an overall R^2 as high as 0.9250. This demonstrates the ability of AeroGTO to estimate drag on unseen car shapes accurately, which is promising in real-world applications.

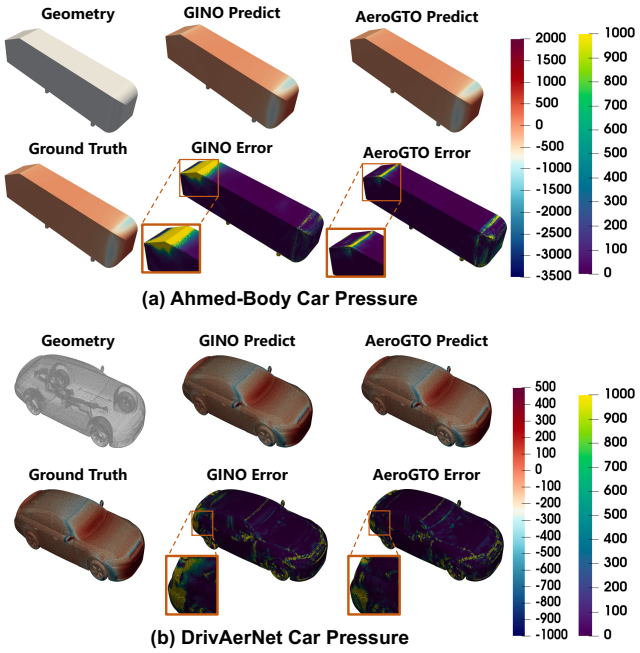

Figure 3: Visualization of a ground-truth pressure and corresponding prediction by the prior leading model, GINO, and our proposed model, AeroGTO for Ahmed-Body **(a)** and DrivAerNet **(b)** datasets, as well as the square error.

Visualization As illustrated in Fig. 3, our proposed model, AeroGTO, consistently outperforms the previous leading model, GINO, on both the Ahmed-Body and DrivAerNet datasets. Notably, AeroGTO demonstrates significantly improved performance in the curved regions of the car surfaces in both datasets, where intricate physical interactions occur. Accurately capturing geometric details and physics in these regions is crucial, and AeroGTO excels in handling complex geometries.

In addition, using the Ahmed-Body dataset as an example, Fig. 4(a) presents the correlation plot of the drag coefficient C_d predicted by AeroGTO against the ground truth for 51 unseen shapes, achieving an R^2 score of 0.9250.

Ablation Studies

Tab. 3 presents ablation studies on the Ahmed-Body dataset, evaluating the impact of various AeroGTO configurations. **Graph: no kNN** indicates that k-nearest neighbors (kNN) are not used for edge augmentation during edge-focused sampling, with the number of sampled edges kept consistent to ensure a fair comparison. **no SPE** denotes the absence of sinusoidal positional encoding (SPE), limiting the model to interpret positional context solely from coordinate data, thus reducing its ability to capture complex physics. **no MP** excludes GNN-based local message passing, eliminating edge information and the message-passing module.
Attention: linear replaces the global attention module with the linear attention mechanism used in GINO (Hao et al. 2023), illustrating the impact of this alternative approach on

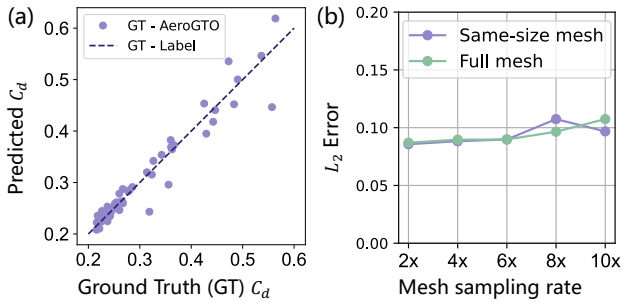

Figure 4: **(a)** Correlation plot of the drag coefficient C_d predicted by our AeroGTO model against the ground truth for the Ahmed-Body unseen test set of 51 different geometries, achieving an R^2 score of 0.9250. The dotted line denotes the line of ideal correlation. **(b)** Varying the sampling rates of the input-output mesh (same rate for training and testing), and testing on the full mesh.

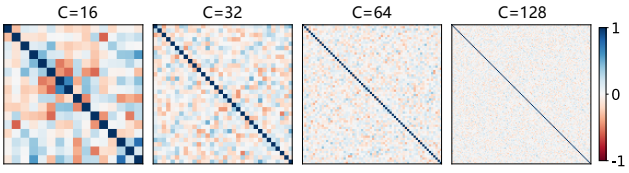

Figure 5: Correlation matrix of the learned subspace queries (Q_{ls}) of the last layer in AeroGTO, showing the correlation coefficients for $M = 16$ to $M = 128$ between M different C-dimensional vectors in Q_{ls}.

large datasets. **no** indicates removing the global attention operation, reducing the model to a structure akin to MGN (Pfaff et al. 2020).

Sampling: The same sampling ratio is applied to compare two distinct sampling strategies for nodes and edges, each suited to different scenarios. **Notably**, for the Ahmed-Body dataset, **Edge-Focused** sampling is used by default due to the well-defined mesh structure. For the DrivAerNet dataset, where the mesh is highly complex, the **Node-Focused** approach is employed.

Ablation experiments show that efficient large-scale modeling relies on capturing long-range dependencies and enabling local-global interactions, highlighting AeroGTO's success through its multi-level representation and collaborative design.

Model Analysis

Discretisation Invariance and Test on Full Mesh As shown in Fig. 4(b), similar to GINO (Li et al. 2024), we trained and tested AeroGTO using node-focused subsampled input-output meshes at various sampling rates (2x, 4x, 6x, 8x, 10x) in Ahmed-Body dataset, and observed a consistent error rate across all sampling rates. Additionally, although AeroGTO was trained on a coarse dataset, we evaluated it on full-size meshes from an unseen test set. Fig. 4(b) demonstrates the ability of AeroGTO on the full-size. This feature allows the model to be trained at a coarse resolution for a dense mesh to reduce computational requirements.

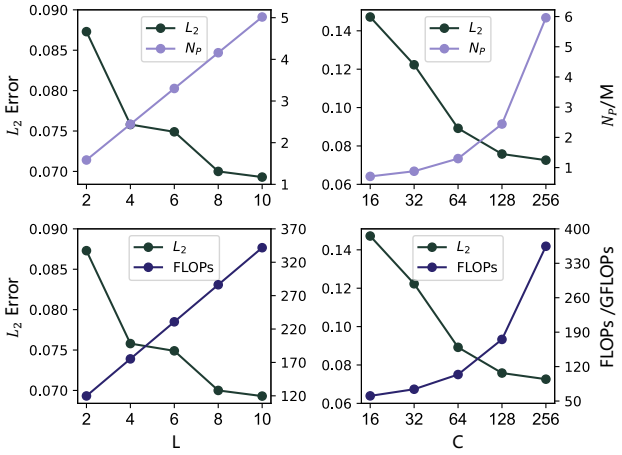

Figure 6: Model scalability on the Ahmed-Body dataset. We retrain the AeroGTO for model layers L (left) and latent dimension C (right).

Learned Subspace Queries The core of global attention in AeroGTO lies in the learned subspace queries $Q_{ls} \in \mathbb{R}^{[M,C]}$, where M represents the subspace dimension and C is the latent dimension. Results of various combinations of M and C indicate that the model's representation ability (measured by L_2) is linked to the rank of Q_{ls}, with $\text{rank}(Q_{ls}) \leq \min(M, C)$. To achieve a balanced trade-off, we set $M = C$. Additionally, Fig. 5 shows the correlation matrix of the M vectors in the final layer of Q_{ls}, displaying the correlation coefficients for $M = 16$ to $M = 128$. These results exhibit that the learned subspace queries effectively capture independent latent vectors, serving as the basis of the subspace with expressive solid power in the corresponding dimension. This further validates the benefits of our projection-inspired attention in enhancing attention learning.

Model Scalability To better understand the effect of scalability, as shown in Fig. 6, we re-trained AeroGTO with different model layers $L = [2, 4, 6, 8, 10]$ (left) and latent dimensions $C = [16, 32, 64, 128, 256]$ (right). Here, C was fixed at 128 for comparing L, and L was fixed at 4 for comparing C. The results indicate AeroGTO's improved representation ability as size increases, highlighting its potential for large-scale pre-training.

Conclusion

To address the challenges of capturing complex physics and managing large-scale computation, we propose AeroGTO, an efficient graph-transformer operator specifically designed for large-scale aerodynamic learning in engineering applications. Given the 3D mesh of a car geometry, AeroGTO can accurately predict its surface pressure and estimate drag. The accuracy and efficacy of AeroGTO are extensively validated across two industry-standard benchmarks compared with five advanced models.

Ethical Statement

Our work does not have any related ethical concerns.

Acknowledgments

This work was supported by the National Key Research and Development Program of China Grant (No. 2021YFF0500403). The authors would like to thank Guan Wang from Baidu, and Lvlin Kuang and Jiyan Qiu from NVIDIA for their meaningful discussions.

References

Ahmed, S. R.; Ramm, G.; and Faltin, G. 1984. Some salient features of the time-averaged ground vehicle wake. *SAE transactions*, 473–503.

Alfonsi, G. 2009. Reynolds averaged navier-stokes equations for turbulence modeling. *Applied Mechanics Reviews*, 63(7, Art. 040802).

Atz, K.; Grisoni, F.; and Schneider, G. 2021. Geometric deep learning on molecular representations. *Nature Machine Intelligence*, 3(12): 1023–1032.

Belbute-Peres, F. D. A.; Economon, T.; and Kolter, Z. 2020. Combining differentiable PDE solvers and graph neural networks for fluid flow prediction. In *international conference on machine learning*, 2402–2411. PMLR.

Cao, Y.; Chai, M.; Li, M.; and Jiang, C. 2023. Efficient learning of mesh-based physical simulation with bi-stride multi-scale graph neural network. In *International Conference on Machine Learning*, 3541–3558. PMLR.

Elrefaie, M.; Dai, A.; and Ahmed, F. 2024. Drivaernet: A parametric car dataset for data-driven aerodynamic design and graph-based drag prediction. *arXiv preprint arXiv:2403.08055*.

Fedus, W.; Zoph, B.; and Shazeer, N. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120): 1–39.

Franceschi, L.; Niepert, M.; Pontil, M.; and He, X. 2019. Learning discrete structures for graph neural networks. In *International conference on machine learning*, 1972–1982. PMLR.

Hao, Z.; Wang, Z.; Su, H.; Ying, C.; Dong, Y.; Liu, S.; Cheng, Z.; Song, J.; and Zhu, J. 2023. Gnot: A general neural operator transformer for operator learning. In *International Conference on Machine Learning*, 12556–12569. PMLR.

Heinz, S. 2020. A review of hybrid RANS-LES methods for turbulent flows: Concepts and applications. *Progress in Aerospace Sciences*, 114: 100597.

Janny, S.; Beneteau, A.; Nadri, M.; Digne, J.; Thome, N.; and Wolf, C. 2023. Eagle: Large-scale learning of turbulent fluid dynamics with mesh transformers. *arXiv preprint arXiv:2302.10803*.

Kim, J.-H.; Jun, J.; and Zhang, B.-T. 2018. Bilinear attention networks. *Advances in neural information processing systems*, 31.

Kochkov, D.; Smith, J. A.; Alieva, A.; Wang, Q.; Brenner, M. P.; and Hoyer, S. 2021. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21): e2101784118.

Kovachki, N.; Li, Z.; Liu, B.; Azizzadenesheli, K.; Bhattacharya, K.; Stuart, A.; and Anandkumar, A. 2023. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89): 1–97.

Lee, M.; and Moser, R. D. 2015. Direct numerical simulation of turbulent channel flow up to. *Journal of Fluid Mechanics*, 774: 395–415.

Lee, S.; and Oh, T. 2024. Inducing point operator transformer: A flexible and scalable architecture for solving PDEs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 153–161.

Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; and Anandkumar, A. 2020a. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*.

Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; and Anandkumar, A. 2020b. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*.

Li, Z.; Kovachki, N.; Choy, C.; Li, B.; Kossaifi, J.; Otta, S.; Nabian, M. A.; Stadler, M.; Hundt, C.; Azizzadenesheli, K.; et al. 2024. Geometry-informed neural operator for large-scale 3d PDEs. *Advances in Neural Information Processing Systems*, 36.

Li, Z.; Shu, D.; and Barati Farimani, A. 2024. Scalable transformer for pde surrogate modeling. *Advances in Neural Information Processing Systems*, 36.

Liu, P.; Hao, Z.; Ren, X.; Yuan, H.; Ren, J.; and Ni, D. 2024. PAPM: A Physics-aware Proxy Model for Process Systems. *arXiv preprint arXiv:2407.05232*.

Pfaff, T.; Fortunato, M.; Sanchez-Gonzalez, A.; and Battaglia, P. W. 2020. Learning mesh-based simulation with graph networks. *arXiv preprint arXiv:2010.03409*.

Piomelli, U. 1999. Large-eddy simulation: Achievements and challenges. *Progress in Aerospace Sciences*, 35(4): 335–362.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

Raissi, M.; Perdikaris, P.; and Karniadakis, G. E. 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378: 686–707.

Shen, Z.; Zhang, M.; Zhao, H.; Yi, S.; and Li, H. 2021. Efficient attention: Attention with linear complexities. In

Proceedings of the IEEE/CVF winter conference on applications of computer vision, 3531–3539.

Vinuesa, R.; and Brunton, S. L. 2022. Enhancing computational fluid dynamics with machine learning. *Nature Computational Science*, 2(6): 358–366.

Wu, H.; Luo, H.; Wang, H.; Wang, J.; and Long, M. 2024. Transolver: A fast transformer solver for PDEs on general geometries. *arXiv preprint arXiv:2402.02366*.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

Yu, W.; Zhang, Z.; and Qin, Z. 2022. Low-pass graph convolutional network for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 8954–8961.