DOI: 10.13009/EUCASS2025-675
11TH EUROPEAN CONFERENCE FOR AERONAUTICS AND SPACE SCIENCES (EUCASS)

# Rapid aerodynamic prediction for wings via physics-embedded Transformer

Yunjia Yang*, Weishao Tang*, Yufei Zhang*, and Haixin Chen*†
* *Tsinghua University, Beijing, 100084, China*
*yangyj20@mails.tsinghua.edu.cn*
† Corresponding Author

## Abstract

Fast and accurate prediction of aerodynamic flow fields is essential for efficient aircraft design but remains challenging for complex transonic wing configurations. This study advances the use of machine learning for aerodynamic modeling by developing a generalizable dataset and applying Transformer-based neural network architectures to realistic wing geometries. A comprehensive dataset of transonic swept wings is constructed using a geometric parameterization approach inspired by engineering needs, covering a wide range of wing shapes and operating conditions. Two Transformer-based models are then investigated: the Vision Transformer (ViT) and the Physics-embedded Transformer (PeT), which incorporates physics-aware tokenization and attention mechanisms. The models are trained on the proposed dataset, and results show that Transformer-based models significantly outperform a U-Net baseline in predicting surface pressure and friction distributions, as well as integrated aerodynamic coefficients. The model implementations and an interactive prediction tool are released at https://github.com/YangYunjia/flogen, supporting future applications in data-driven aerodynamic design workflows.

## 1. Introduction

Fast and accurate prediction of flow fields around transonic wings is essential for accelerating aerodynamic design in modern aircraft. Although computational fluid dynamics (CFD)-based optimization methods, particularly those utilizing adjoint techniques for gradient evaluation, have been applied for efficient wing shape optimization, they remain computationally expensive. This is especially true for multipoint and robust optimization tasks, which require repeated aerodynamic evaluations across various design points[1,2].

Surrogate modeling is a compelling alternative for reducing computational cost in aerodynamic shape optimization. With the rapid advancement of deep neural networks, these models can be trained to predict aerodynamic coefficients and flow fields from input geometries. Once trained, they can replace CFD simulations either partially or entirely within the optimization loop, significantly accelerating the design process. This paradigm, commonly referred to as data-based optimization (DBO)[3,4].

The principal advantage of DBO lies in the reusability of pretrained models across multiple optimization cases. Although building the training dataset and training the model can be time-consuming, these costs can be amortized over numerous optimization tasks. Achieving this level of reusability, however, requires models that are generalizable across diverse geometries and operating conditions. In recent years, researchers successfully trained general models for two-dimensional aerodynamic components, which has led to some notable success of DBO[3-8].

However, DBO's application to three-dimensional configurations such as wings remains limited, mainly because of the difficulty in building a general ML model for reliable wing flow field prediction. While there have been recent efforts to construct such models, many rely on simplified or constrained wing geometries, which hinders their general applicability. For example, some studies use datasets with fixed geometries and only vary flow conditions[9-17], while others perturb a baseline wing to generate training data[18-20], which is short of the generalization needs of DBO. In our previous work[21], we established a dataset for single-section wings, which, while a step forward, still lacks the complexity of realistic configurations.

Another bottleneck is the inadequacy of conventional ML architectures for handling large-scale datasets that span a broad range of geometries and flow conditions. Traditional approaches, including dynamic mode decomposition (DMD), multilayer perceptrons (MLPs), convolutional neural networks (CNNs), Fourier neural operators (FNOs), and

DOI: 10.13009/EUCASS2025-675



graph neural networks (GNNs), do not scale effectively for such demanding tasks. This motivates the exploration of advanced architectures like Transformers.

To overcome these challenges, this study introduces a simplified yet expressive geometric parameterization scheme grounded in realistic wing configurations. Using this scheme, we construct a comprehensive dataset of transonic swept wings encompassing a wide range of geometric parameters and design conditions. Building on this foundation, we develop and compare two state-of-the-art Transformer architectures for predicting wing surface flow fields from geometry: the Vision Transformer (ViT) [22] and a physics-embedded Transformer (PeT) [23], which incorporates a physics-embedded attention mechanism. The results demonstrate the advantages of the Transformer-based model architectures, which significantly enhance prediction accuracy across diverse wing configurations.

## 2. Dataset for swept wings

In this work, we construct a comprehensive dataset of wing surface flow fields based on several representative configurations commonly found in the aircraft industry. The selected planform geometry includes the characteristic "Yehudi break" [24], or "kink". The dihedral angle, twist angle, thickness, and camber are varied along spanwise according to given patterns. These variations allow the dataset to encompass a broad spectrum of realistic wing geometries, making it well-suited for training surrogate models for data-based optimization (DBO).

### 2.1 Sampling of wing geometries

#### 2.1.1 Wing geometric parameters

We begin by introducing the geometric parameters used to describe the wing, which are divided into two components: those characterizing the sectional airfoils and those defining the planform shape. A wing is constructed by stretching the sectional airfoils along the spanwise direction.

In principle, the airfoils at each spanwise station could differ entirely, leading to an extremely high-dimensional parameter space. To balance generality and simplicity, we identify common spanwise variation patterns and incorporate them into the wing geometry description.

Specifically, each wing geometry in the dataset is generated from a single baseline airfoil, parameterized using a 9th-order Class-Shape Transformation (CST) method. The upper and lower surfaces are represented independently, resulting in 20 shape parameters.

$$y_{\mathrm{u,baseline}}(x)=\sum_{i=0}^{9}u_i\Phi_i(x),\qquad y_{\mathrm{l,baseline}}(x)=\sum_{i=0}^{9}l_i\Phi_i(x) \tag{1}$$

The airfoil can also be described with its normalized thickness and camber line:

$$t(x)=\frac{y_{\mathrm{u,baseline}}(x)-y_{\mathrm{l,baseline}}(x)}{\max\limits_{x}\left(y_{\mathrm{u,baseline}}(x)-y_{\mathrm{l,baseline}}(x)\right)},\qquad \delta(x)=\frac{y_{\mathrm{u,baseline}}(x)+y_{\mathrm{l,baseline}}(x)}{\max\limits_{x}\left(y_{\mathrm{u,baseline}}(x)+y_{\mathrm{l,baseline}}(x)\right)} \tag{2}$$

Since thickness and camber are essential to determining an airfoil's aerodynamic performance, the spanwise distribution of maximum thickness and maximum camber is prescribed as functions of the spanwise station $\eta$, i.e., $t_{\max}(\eta)$ and $\delta_{\max}(\eta)$. This allows the construction of airfoil sections across the span based on the baseline airfoil and these distribution functions:

$$t(x,\eta)=t(x)\cdot t_{\max}(\eta)=y_{\mathrm{u}}(x,\eta)-y_{\mathrm{l}}(x,\eta),\qquad \delta(x,\eta)=\delta(x)\cdot\delta_{\max}(\eta)=\frac{1}{2}\left(y_{\mathrm{u}}(x,\eta)+y_{\mathrm{l}}(x,\eta)\right) \tag{3}$$

The second group of parameters defines the planform shape, as illustrated in the three-view diagram in Figure 1. From the top view, the wing can be seen as a combination of a trapezoidal part (OAGF, marked by right slash lines) and an extra surface part between the root and the kink ($\triangle$ABE). The trapezoidal part shape is determined with three parameters: the aspect ratio $AR=2b_{l/2}^2/S_{\mathrm{trap}}=4b_{l/2}/(\overline{\mathrm{FG}}+\overline{\mathrm{OA}})$, taper ratio $TR=\overline{\mathrm{FG}}/\overline{\mathrm{OA}}$, and the leading-edge sweep angle $\Lambda_{\mathrm{LE}}$. The extra surface can be determined with the kink location and root adjustment ratio $\kappa=\overline{\mathrm{AB}}/\overline{\mathrm{AC}}$.

DOI: 10.13009/EUCASS2025-675



It is worth mentioning that in the present study, the fuselage location is fixed at 10% span, and only the exposed part of the wing to the air is simulated. The simulation part corresponds to the light blue area (O'B'EGF). The reference area of the wing is the projection area of the simulation part.

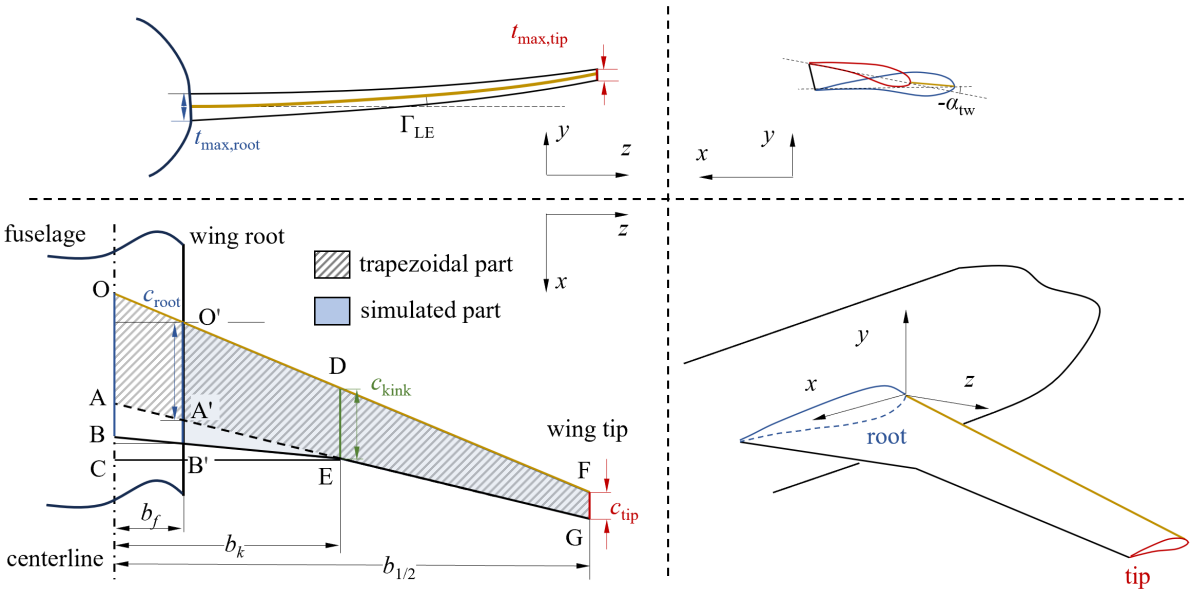



Figure 1: Three-view diagram of a typical kink wing

Besides the parameters above, there are two more parameters that control the $y$-positions and the rotation angles of every section airfoil, which are the dihedral angle $\Gamma_{LE}(\eta)$ and the twist angle $\alpha_{tw}(\eta)$, respectively. For modern wings, they always vary along spanwise to achieve the best aerodynamic performance.

### 2.1.2 Summary of the parameters of several typical wings

Based on the definitions of wing parameters in the last section, three wing benchmark models and two real-world aircraft wings are studied, and their wing parameters are listed in Table 1. The benchmark models include the DPW-W1 from the AIAA Drag Prediction Workshop (DPW), the DLR-F6 model from the German Aerospace Center, and NASA's Common Research Model (CRM). The real-world wings are from the Airbus A320 and Boeing 787. The DPW-W1 is a one-segment wing, and the others have a kink. To ensure consistency across all cases, the reference wing areas have been recalculated based on the total projected area, as the sources employed differing definitions.

Table 1: Geometric parameters of typical wings

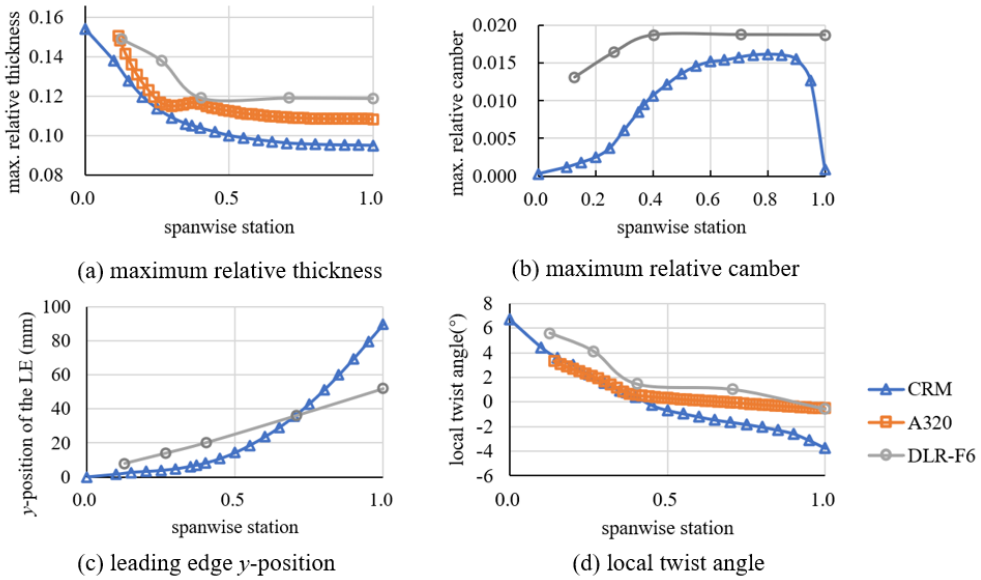
| Parameter | DPW-W1 | DLR-F6 | CRM | A320 | B787 |
|---|---|---|---|---|---|
| Cruise Mach number | 0.76 | 0.75 | 0.85 | 0.775 | 0.85 / 0.90 |
| maximum relative thickness at centerline | 0.1350 | 0.1629 | 0.1542 | 0.1394 | 0.1449 |
| ratio of maximum thicknesses at kink and centerline | — | 0.7316 | 0.6822 | — | 0.6472 |
| ratio of maximum thicknesses at tip and centerline | 1.0000 | 0.7306 | 0.6161 | 0.7166 | 0.6056 |
| taper ratio | 0.55 | 0.38 | 0.275 | 0.33 | 0.18 |
| swept angle (deg) | 15.00 | 25.15 | 35.00 | 25.00 | 32.20 |
| dihedral angle at tip (deg) | 0.5 | 5.2 | 6.5 | 4.4 | 6.0 |
| aspect ratio | 8.0 | 9.28 | 8.38 | 8.79 | 9.20 |
| kink location (%) | — | 40.1 | 37.0 | 39.2 | 37.4 |
| root adjustment | — | 1.00 | 0.67 | 1.00 | 0.88 |
| twist angle at tip | 2.90 | 6.14 | 10.47 | 3.82 | — |
| reference | [25] | [26] | [24] | [27] | Piano[a] |

[a] obtained from the free sample of the software Piano: https://www.lissys.uk/samp1/b787.html

DOI: 10.13009/EUCASS2025-675



As mentioned above, the dihedral angle, twist angle, maximum thickness, and camber along spanwise are varied. Figure 2 summarizes these spanwise distributed parameters for the DLR-F6, CRM, and A320 wing models.




(a) maximum relative thickness (b) maximum relative camber

(c) leading edge $y$-position (d) local twist angle

Figure 2: Spanwise distribution of dihedral angle, maximum thickness, and twist angle for typical kinked wings

### 2.1.3 Sampling of wing geometry parameters

By analyzing the five representative wing configurations, we determine the typical parameter ranges and use them as the basis for sampling to establish our dataset.

The first step involves generating the CST coefficients for the baseline airfoils. For this purpose, we utilize an existing database from our previous study. The CST coefficients are sampled using the Output Space Sampling (OSS) method [28], which aims to produce geometric variations that exhibit diverse and representative pressure distribution patterns. A total of 1 420 sets of CST coefficients are obtained. Figure 3 presents the corresponding normalized thickness $t(x/c)$ and camber lines $\delta(x/c)$ of the sampled airfoils.

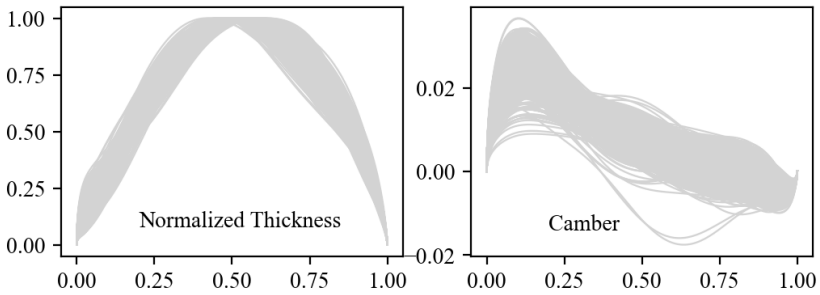



Figure 3: Normalized thickness and camber lines of the baseline airfoils

The spanwise distributions of four geometric parameters—dihedral angle, twist angle, maximum thickness, and maximum camber—are determined using cubic spline interpolation based on five spanwise control points (CPs). As illustrated in Figure 4, the CPs are numbered from 0 to 4, extending from the centerline to the tip. Specifically, CPs #0, #2, and #4 are positioned at the root, kink, and tip, while CPs #1 and #3 are located middle between adjacent points.

Among these parameters, the dihedral angle is defined by CPs #2 and #4. The leading-edge $y$-coordinates are initialized to zero at the centerline and increase linearly toward the kink, with a slope defined by the ratio $y_{LE}(\eta) \,/\, b(\eta) = \tan(\Gamma_{LE,kink})$, where $b(\eta)$ is the vertical distance from the kink to the root. Beyond the kink, the $y$-coordinates of the leading edge are determined by an additional parameter $y_{LE,tip} \,/\, b_{1/2} = \tan(\Gamma_{LE,tip})$, which defines the tip position. The intermediate segment between kink and tip is generated using a cubic spline with a slope matched to the linear segment to ensure continuity.

DOI: 10.13009/EUCASS2025-675



The distribution of maximum thickness $t_{max}(\eta)$ is defined using CPs #0, #2, #3, and #4. The thickness at CP #0, along with the ratios of thickness values at the remaining control points relative to CP #0, denoted $t_0$, $r_{t,2}$, $r_{t,3}$, and $r_{t,4}$, are used to construct the spline. Similarly, maximum camber distribution $\delta_{max}(\eta)$ is defined from CPs #0 to #4, with values fixed at zero for CP #0 and at $\delta_0$ for CP #3. The maximum cambers at CPs #1, #2, and #4 are expressed as ratios relative to $\delta_0$, denoted $r_{\delta,1}$, $r_{\delta,2}$, and $r_{\delta,4}$. Twist angles are also specified at all five CPs, with the twist at CP #0 set to zero. The angles at outer control points are defined as incremental deviations from the root.

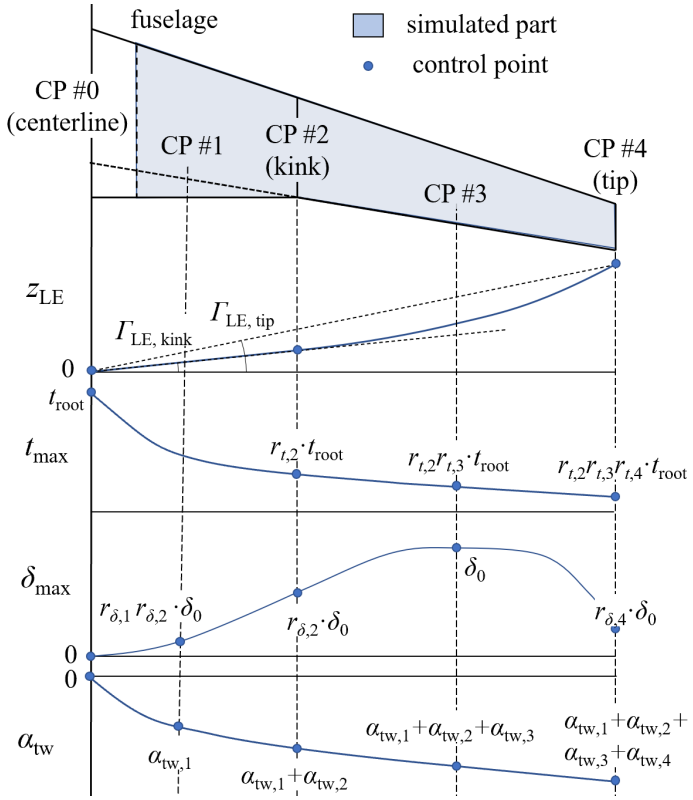



Figure 4: Spanwise control points and the variables to generate distributed wing parameters

With this parameterization, each wing geometry is fully defined by its baseline airfoil's CST coefficients and 18 additional parameters governing spanwise distribution. The CST coefficients are sampled using OSS, while the 18 spanwise parameters are randomly sampled within the ranges inferred from the five reference wing configurations, as summarized in Table 2. For each set of CST coefficients, two independent sets of spanwise parameters are sampled, resulting in a total of 2 840 distinct wing geometries. Figure 5 shows the top view of typical wings in the dataset.

Table 2: Sampling ranges of the wing geometric parameters

| Parameter | Symbol | Range | | Parameter | Symbol | Range | |
|---|---|---|---|---|---|---|---|
| sweep angle | $\Lambda_{LE}$ | 25° | 40° | thickness ratio at CP3 | $r_{t,3}$ | 0.90 | 0.98 |
| dihedral angle (tip) | $\Gamma_{LE, tip}$ | 4° | 6° | thickness ratio at CP4 | $r_{t,4}$ | 0.92 | 1.00 |
| dihedral angle (kink) | $\Gamma_{LE, kink}$ | 0.5° | 6° | camber ratio at CP1 | $r_{\delta,1}$ | 0.3 | 0.8 |
| aspect ratio (trap.) | $AR$ | 8 | 11 | camber ratio at CP2 | $r_{\delta,2}$ | 0.5 | 1.0 |
| taper ratio (trap.) | $TR$ | 0.15 | 0.40 | camber ratio at CP4 | $r_{\delta,4}$ | 0.0 | 0.8 |
| kink location | $\eta_k$ | 36% | 42% | twist angle at CP1 | $\alpha_{tw,1}$ | –4° | –2° |
| root adjustment | $\kappa_{root}$ | 50% | 110% | twist angle at CP2 | $\alpha_{tw,2}$ | –4° | –2° |
| root max. relative thickness | $t_0$ | 0.14 | 0.17 | twist angle at CP3 | $\alpha_{tw,3}$ | –3° | –1° |
| thickness ratio at CP2 | $r_{t,2}$ | 0.60 | 0.70 | twist angle at CP4 | $\alpha_{tw,4}$ | –3° | –1° |

DOI: 10.13009/EUCASS2025-675



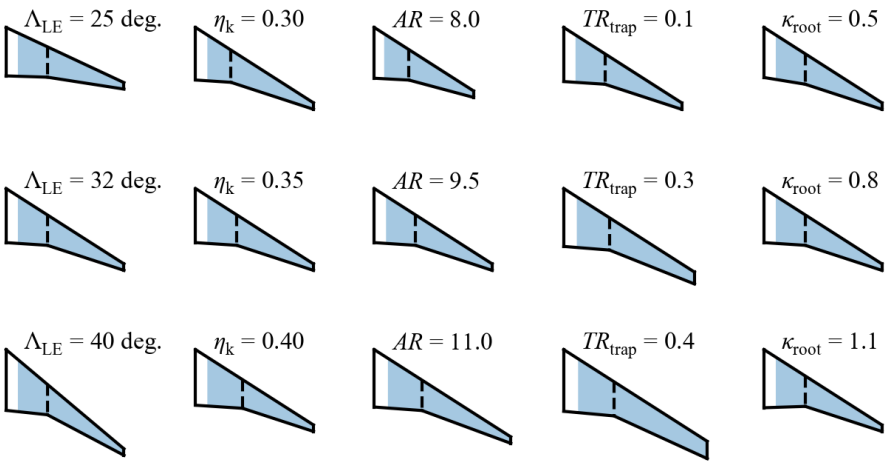



Figure 5: Top views of wings in the dataset

## 2.2 Sampling of wing operating conditions

For each wing geometry, eight operating conditions are randomly sampled. The freestream Mach number ranges from 0.75 to 0.90, and the angle of attack varies between 2° and 12°. The Reynolds number and freestream temperature are fixed at 20 million and 300 K, respectively, for all simulations.

## 2.3 Simulation of the wing flow fields

Reynolds-Averaged Navier–Stokes (RANS) simulations are performed using the open-source CFD solver suite developed by the MDOLab[1].

### 2.3.1 Mesh generation

The surface mesh for each wing is generated using an in-house meshing tool. As illustrated in Figure 6, the wing surface is divided into eight structured blocks: four on the inner segment (from root to kink) and four on the outer segment (from kink to tip). For each segment, the blocks represent the lower surface, leading-edge region, upper surface, and trailing edge, respectively. Along the airfoil circumferential direction, the blocks have 117, 9, 117, 9 mesh cells. In the spanwise direction, the inner and outer segments contain 41 and 125 cells, respectively. The mesh is refined near the leading and trailing edges in the circumferential direction and near the wing tip in the spanwise direction.

For the wing tip, the geometry and surface mesh from the standard L1 mesh of the NASA CRM model [29] are adapted. The tip mesh is scaled and rotated to align with each wing's tip section. Each surface mesh consists of 45 120 cells in total.

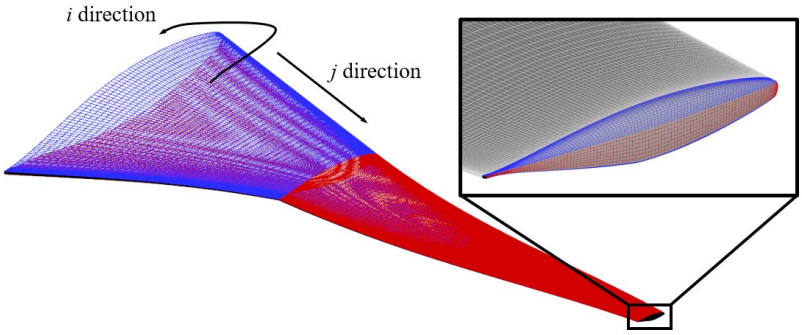



Figure 6: Surface computation mesh of the wings

[1] https://mdolab.engin.umich.edu/software

DOI: 10.13009/EUCASS2025-675



The volume mesh is then extruded from the surface mesh using the pyHyp. The extrusion contains 81 layers in the normal direction, extending up to 50 times the root chord. This results in a full 3D mesh with approximately 3.65 million cells.

#### 2.3.2 CFD simulation

The flow field around each wing is computed using ADflow [30]. A "3w" multigrid strategy is adopted to accelerate convergence: 500 solver cycles are applied on the coarser multigrid levels, followed by up to 3 000 cycles on the original fine mesh. Initially, the approximate Newton-Krylov (ANK) solver is used to solve the linear systems; once the total residual drops below $1 \times 10^{-8}$, the solver switches to the Newton-Krylov (NK) method. Convergence is declared when the residual falls below $1 \times 10^{-10}$. All other solver settings follow the default configuration.

During the simulation, the lift coefficient is monitored to ensure steady-state convergence. Only cases with lift coefficient fluctuations below 0.0005 over the last 10 outer iterations are included in the final dataset. As a result, a total of 19,381 valid wing flow fields are obtained.

### 2.3 Postprocess

From the converged simulations, surface pressure and friction coefficient distributions are extracted for model training and evaluation. The friction coefficient is decomposed into two components: the streamwise part $C_{f,\tau}$ in the $x$–$y$ plane, and the spanwise part $C_{f,z}$, in the $z$-direction. The values of the coefficients are nondimensionalized with their maximum and minimum values.

To ensure consistent data representation across all geometries, surface values are interpolated from the original CFD grid to a uniform reference grid. Along the airfoil circumferential direction ($i$-direction), a fixed set of normalized chordwise positions $\{(x/c)_i\}$ s is used for both the upper and lower surfaces. The grid is truncated near the trailing edge to retain only two cells for numerical stability.

In the spanwise direction ($j$-direction), the wing surface is sampled on 128 evenly spaced cross-sectional planes. Tip regions are excluded from the dataset. This interpolation procedure results in a final surface grid of 256 × 128 points per wing. Validation tests confirm that omitting the tip and interpolating the grid introduces less than 0.1% error in global aerodynamic coefficients.

## 3. Transformer architectures for wing flow field prediction

In this section, a Transformer architecture with a physics-embedded attention mechanism is proposed as the backbone of the wing surface flow field prediction model.

### 3.1 Vision Transformer (ViT)

#### 3.1.1 Overall architecture

The Transformer architecture was introduced by Google in the paper *Attention Is All You Need*[31]. As shown in Figure 7, the typical Transformer model consists of an encoder-only structure. Each transformer layer comprises two sublayers: a multi-head self-attention mechanism and a position-wise feedforward network, both followed by residual connections and layer normalization. It significantly speeds up training and inference compared to former recurrent neural networks (RNNs), while also improving the model's ability to capture global dependencies across tokens, regardless of their positional distance in the sequence.

While the Transformer architecture was originally designed for sequential data like text, its success inspired researchers to explore its application to computer vision tasks. Before Transformers, CNNs were the major trend to deal with images and field data, but they also suffer from limitations similar to RNNs: each CNN layer processes only a small neighborhood of the input. As a result, capturing long-range dependencies and global contextual information typically requires stacking many layers, thereby increasing model depth and complexity.

DOI: 10.13009/EUCASS2025-675



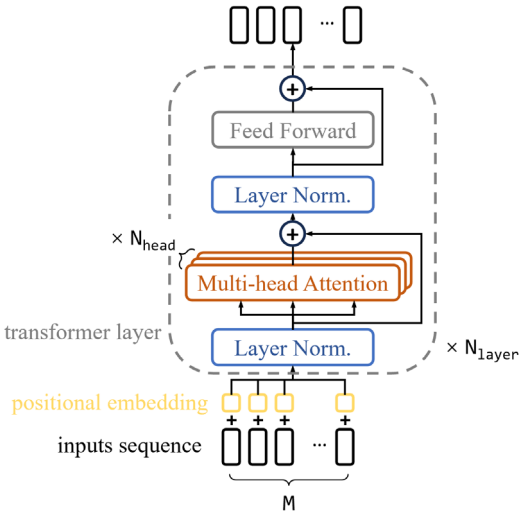



Figure 7: Architecture of the Transformer

The ViT extends the Transformer architecture to computer vision tasks. As illustrated in Figure 8, ViT divides an input image into a sequence of fixed-size patches, treating each patch as a token analogous to a word in natural language processing. These patch tokens, along with a learnable class embedding, are then fed into a standard Transformer encoder stack for representation learning and prediction.

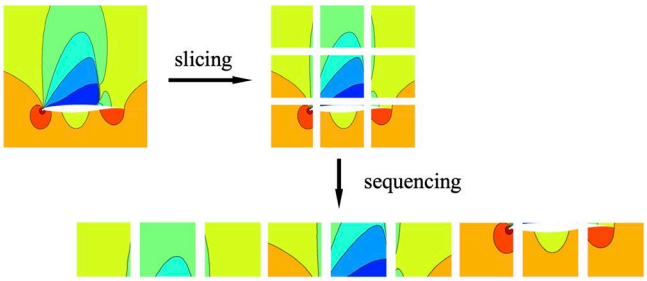



Figure 8: Spanwise distribution of dihedral angle, maximum thickness, and twist angle for typical kinked wings

### 3.1.2 Attention mechanism

The self-attention is the core of the Transformer. It determines how much each token should focus on every other token in the input sequence and helps the model learns the global correlation between every token.

The attention in the Transformer is realized with a query-answer mechanism. Suppose the input sequence has $M$ tokens with each has $N_{\text{hidden}}$ dimensions, a value vector $v_i$ with dimension of $D$ is calculated for each token with shared weights, and formulate a value matrix $V$ of size $M \times D$. Meanwhile, a query vector $q_i$ and a key vector $k_i$ are calculated for each token, also with shared weights. They contribute to the query and key matrices $Q$ and $K$. Then, for every position of token j, the query vector $q_j$ is compared with the key vectors of every token $\{k_i\}_{i=0,\dots M}$, and the output is calculated by summing up the value vectors with weights decided by the similarity between $q_j$ and the key vectors.

The scaled dot-product attention is the most commonly used approach to describe the similarity:

$$Att_j = \sum_i \text{Softmax}_i\left(\frac{1}{\sqrt{D}} q_j \cdot k_i^T\right) v_i \tag{4}$$

where Softmax is used to normalize the attention weights, and $D^{1/2}$ is the scaling factor to prevent large dot products.

The formula can be written in matrix form:

$$Att(Q,K,V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{D}}\right)V \tag{5}$$

DOI: 10.13009/EUCASS2025-675



This mechanism allows each token to attend to every other token in the sequence, dynamically adjusting the attention weights based on contextual relevance. To further enhance expressivity, the original Transformer uses multi-head attention, which projects the queries, keys, and values into multiple subspaces and performs parallel attention operations. The outputs of all heads are then concatenated and linearly transformed. This allows the model to jointly capture information from different representation subspaces.

### 3.1.3 Tokenization for field data

As shown in Figure 9, an input of size $H \times W \times C$ (height, width, channels) is divided into a grid of non-overlapping patches, each of size $P_H \times P_W$. This results in $M = (H \times W) / (P_H \times P_W)$ patches. Each patch is then flattened into a vector of size $P_H \times P_W \times C$ and linearly projected into a lower-dimensional embedding space of dimension $N_{\text{hidden}}$ using a learnable matrix. This gives a sequence of patch embeddings, serving as input tokens to the Transformer.

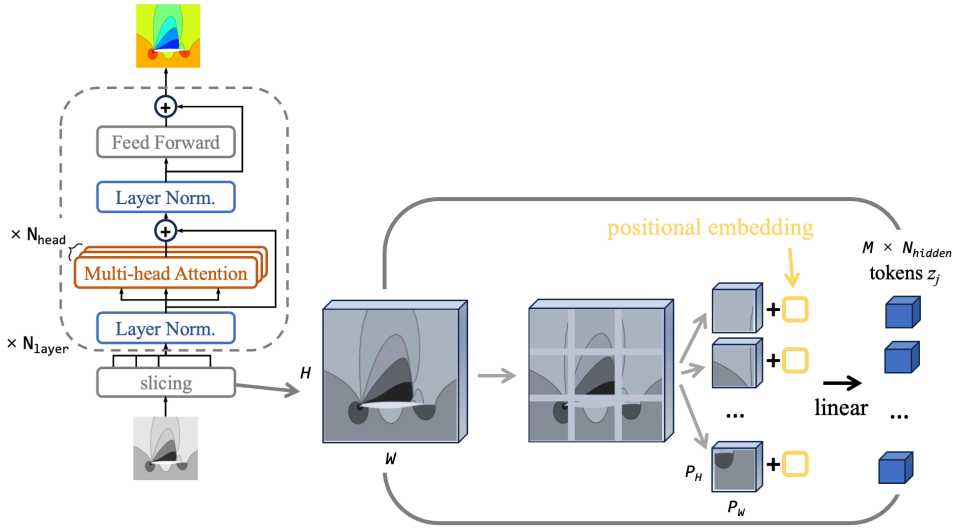



Figure 9: Architecture of the ViT

Since the Transformer treats all input tokens equally, fixed or trainable positional embeddings are added to the input token sequence before feeding it into the Transformer encoder to preserve positional information. The most common fixed positional embedding is the sinusoidal one. It uses a combination of sine and cosine functions of different frequencies to ensure each patch is associated with a unique, continuous vector that preserves relative spatial distances.

To be specific, given the patch index $(I, J)$ and the embedding dimension $w$,

$$Pos_{2D}(I,J;w) = \begin{bmatrix} Pos_{1D}(I;w) \\ Pos_{1D}(J;w) \end{bmatrix}, Pos_{1D}(I;w) = \begin{cases} \sin\left(\frac{I}{T^{2w/D}}\right), w = 2i \\ \cos\left(\frac{I}{T^{2(w-1)/D}}\right), w = 2i+1 \end{cases} \quad (6)$$

The trainable position embedding, on the other hand, simply uses a learnable matrix of shape $M \times N_{\text{hidden}}$ to be added element-wise to the respective input token embedding. They are expected to retain information about the spatial origin of each patch, thereby helping the self-attention layers distinguish between different patch locations and model the 2D structure of the image effectively.

## 3.2 Physics-embedded Transformer (PeT)

ViTs segment images or field data into discrete patches, a process that inherently disrupts the physical continuity between neighboring regions. This patch-based tokenization can obscure critical local interactions and undermine the physical priors essential for accurately predicting physical fields. As a result, there is a clear need for tokenization strategies and attention mechanisms that embed physical knowledge and better preserve the underlying spatial and physical relationships within the data.

DOI: 10.13009/EUCASS2025-675



A more physically consistent tokenization strategy involves projecting the entire input field into a low-dimensional embedding space using weighted projections, where the projection weights themselves are learnable. This approach was first introduced in the Transolver framework [23]. A schematic of the proposed PeT architecture is shown in Figure 10. PeT retains the overall Transformer structure but incorporates a physics-informed tokenization process and modifies the standard attention mechanism to reflect spatial and physical dependencies more accurately.

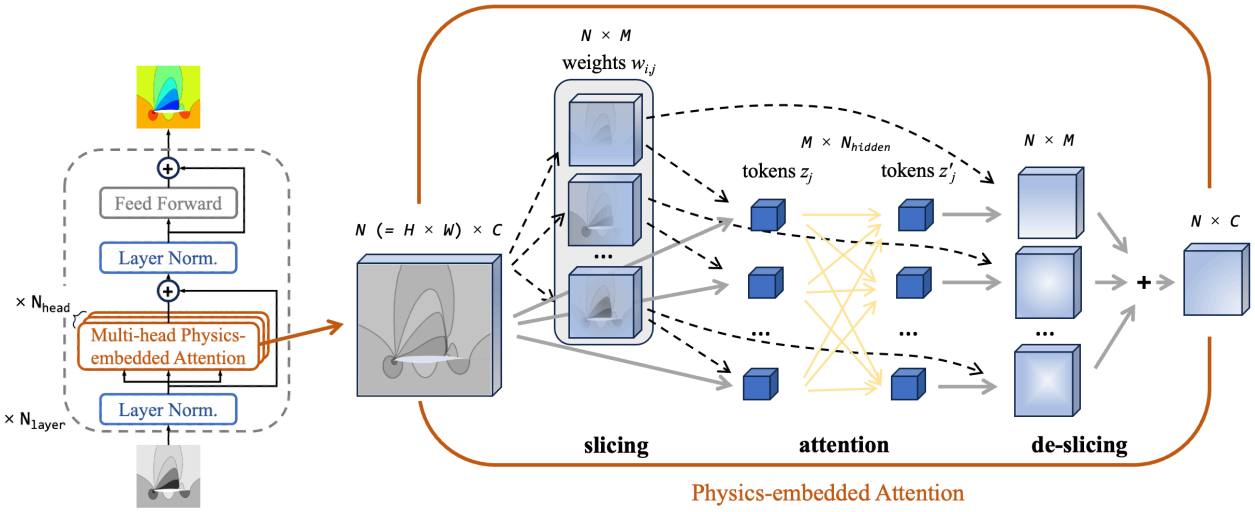



Figure 10: Architecture of the PeT

Technically, the input wing mesh $g = \{g_i\}_{i=1}^{N=H\cdot W}$ is first embedded into deep features $x = \{x_i\}_{i=1}^{N=H\cdot W}$ where the feature at each mesh point has $N_{\text{hidden}}$ channels. Then, some layers $\mathcal{F}(\cdot)$ that are shared among grid points are used to calculate a series of slice weights $w_i \in \mathbb{R}^{1\times M}$ :

$$\{w_i\}_{i=1}^{N=H\cdot W} = \left\{\text{Softmax}\left(\mathcal{F}(x_i)\right)\right\}_{i=1}^{N=H\cdot W}, w_i = \left[w_{i,1}, \cdots, w_{i,M-1}\right] \tag{7}$$

The weight $w_{i,j}$ indicates the degree that the $i$-th mesh point belongs to the $j$-th slice, and the Softmax guarantees that the weights of every mesh point have a sum of 1, i.e., $\sum_{j=1}^{M} w_{i,j} = 1$ .

Then, the global input data on the mesh points (physical space) is projected to the feature space with the weights for every slice $j$:

$$z_j = \frac{\sum_i w_{i,j} x_i}{\sum_i w_{i,j}}, j = 1, \cdots, M \tag{8}$$

where $z_j \in \mathbb{R}^{1\times N_{\text{hidden}}}$ serves the token in the attention calculation. The attention calculation among the encoded tokens $z_j$ to obtain $z'_j$ with Equation (5). Afterwards, they are de-sliced back to the mesh point with the same slice weight $w_{i,j}$.

$$x'_i = \sum_{j=1}^{M} w_{i,j} z'_j \tag{9}$$

PeT also adopts the multi-head attention strategy, where distinct slice weights are computed for each head, and attention is applied independently across the heads.

## 4. Experiments and results

In this section, we compare the two Transformer-based model architectures, ViT and PeT, with a conventional U-Net architecture for fast prediction of wing surface flow fields using the proposed dataset.

DOI: 10.13009/EUCASS2025-675



## 4.1 Experiment setup

### 4.1.1 Baseline U-Net

U-Net, built upon CNN layers, is one of the most widely adopted architectures for flow field prediction. In this study, we employ the U-Net configuration from our previous work [21] as the baseline model. Specifically, it consists of a symmetric encoder–decoder structure. Skip connections are incorporated by storing feature maps from the encoder's contracting path and concatenating them with the corresponding levels in the decoder's upsampling path. Operating conditions are incorporated into the model by concatenating them with the latent representation between the encoder and decoder stages.

Both the encoder and decoder consist of six ResNet groups. The network is designed such that feature map operations occur along the airfoil circumferential direction, while the spanwise dimension remains unchanged. Each ResNet group in the encoder includes a residual block for downsampling followed by a standard residual block. The airfoil-circumferential resolution is reduced by half at each group. In the decoder, the first residual block of each ResNet group is replaced with an upsampling block implemented using convolutional layers and linear interpolation, effectively doubling the resolution at each level. The decoder concludes with a final convolutional layer that reduces the hidden channels to three, corresponding to the predicted output variables.

Two versions of the U-Net baseline are implemented in this study, differing in the number of hidden dimensions. Detailed configurations of both models are provided in Table 3.

Table 3: Implementations of U-Nets

| Model name | Dimensions after encoder blocks | Dimension after decoder blocks | Trainable parameters |
|---|---|---|---|
| U-Net-128 | 16, 32, 32, 64, 64, 128 | 128, 64, 64, 32, 32, 16 | 9 226 778 |
| U-Net-256 | 32, 64, 64, 128, 128, 256 | 256, 128, 128, 64, 64, 32 | 24 569 882 |

### 4.1.2 ViT and PeT implement

To ensure a fair comparison, we set the Transformer-based model with the number of layers $N_{\text{layer}}$ as 5 and the channel of hidden features $N_{\text{hidden}}$ as 256, which ensures their parameter are comparable to the baseline U-Nets. Additionally, the number of attention heads and tokens is fixed at 8 and 32, respectively, in line with common practices in the literature.

Multiple architectural variants are evaluated in this study. For the ViT architecture, we examine different strategies for partitioning the structured mesh into input patches, comparing square patches ($P_H = P_W$) and stripe patches ($P_H > P_W$), where $W$ corresponds to the spanwise direction ($j$-direction in the mesh), and $H$ denotes the airfoil-circumferential direction. We also compare the use of learnable positional embeddings versus fixed sinusoidal embeddings.

For the PeT architecture, we focus on the design of the layers $\mathcal{F}(\cdot)$ used to compute slice weights. Two variants are explored: (1) flattening the structured input and applying a shared point-wise linear layer, and (2) using a convolutional layer with kernel size 3, stride 1, and padding 1 to preserve spatial context. The configurations of all tested models are summarized in Table 4.

Table 4: Implementations of Transformer-based models

| Model name | Positional embedding | Slicing approach | $\mathcal{F}(\cdot)$ | Trainable parameters |
|---|---|---|---|---|
| ViT-sq-sin | Fixed sinusoidal | Square (32 × 32) | — | 6 056 704 |
| ViT-sq-lrn | Learnable | Square (32 × 32) | — | 6 064 896 |
| ViT-st-sim | Fixed sinusoidal | Stripe (128 × 8) | — | 6 056 704 |
| PeT-pw | — | — | Point-wise | 3 776 459 |
| PeT-CNN | — | — | CNN ($k = 3$) | 9 019 339 |

DOI: 10.13009/EUCASS2025-675



### 4.1.3 Training setup and error measurement

For model training, the dataset is split such that 90% of the wing geometries are used for training and the remaining 10% for evaluation. The loss function is defined as the mean squared error (MSE) between the model predictions and the ground-truth distributions obtained from CFD simulations. All models share the same training configuration: a mini-batch size of 4, the Adam optimizer, and a learning rate schedule based on the one-cycle policy. Specifically, the learning rate increases from $1 \times 10^{-6}$ to $1 \times 10^{-4}$ in the first half of training and then decreases back to $1 \times 10^{-6}$ in the second half. Training is performed over 100 epochs, with mini-batches shuffled randomly at the beginning of each epoch.

To ensure robustness, the training process is repeated three times using different random 90% splits of the training data and distinct initializations of the model parameters. For each trained model, performance is evaluated on the held-out test set. For each sample, the relative mean absolute error (MAE) is computed for each predicted coefficient and normalized by the corresponding coefficient range. The final test error is reported as the average of these normalized MAEs across all test samples, as follows:

$$\delta x = \frac{1}{N_s}\sum_{n=0}^{N_s}\frac{\sum_{i,j=0}^{H,W}\left|x^{\text{Model}} - x^{\text{CFD}}\right|}{\max_{i,j}\left(x^{\text{CFD}}\right) - \min_{i,j}\left(x^{\text{CFD}}\right)} \times 100\%, \quad x = C_p, C_{f,\tau}, C_{f,z} \tag{10}$$

In addition to surface field errors, the mean absolute errors in key aerodynamic coefficients, including the lift, drag, and pitching moment about the leading edge, are also evaluated for further validation of predictive accuracy.

$$\Delta x = \frac{1}{N_s}\sum_{n=0}^{N_s}\left|x^{\text{Model}} - x^{\text{CFD}}\right|, \quad x = C_L, C_D, C_{M,z} \tag{11}$$

## 4.2 Model performance

Table 5 presents the mean prediction errors from three cross-validation runs across six evaluation metrics for all models on the test wing geometries.

Table 5: Model prediction errors on the test wing geometries

| Group | Model name | $\delta C_p$ | $\delta C_{f,\tau}$ | $\delta C_{f,z}$ | $\Delta C_L$ | $\Delta C_D$ | $\Delta C_{M,z}$ |
|---|---|---|---|---|---|---|---|
| Baseline U-Net | U-Net-128 | 0.762% | 0.477% | 0.540% | 0.01291 | 0.000929 | 0.01434 |
| | U-Net-256 | 0.729% | 0.461% | 0.509% | 0.01234 | 0.000879 | 0.01382 |
| ViT | ViT-sq-sin | 0.821% | 0.570% | 0.607% | 0.00822 | 0.000698 | 0.00939 |
| | ViT-sq-lrn | 0.617% | 0.465% | 0.505% | **0.00446** | **0.000401** | 0.00506 |
| | ViT-st-sim | 1.784% | 1.059% | 1.060% | 0.02547 | 0.001974 | 0.02988 |
| PeT | PeT-pw | 0.702% | 0.518% | 0.583% | 0.00581 | 0.000564 | 0.00653 |
| | PeT-CNN | **0.560%** | **0.405%** | **0.477%** | 0.00447 | 0.000426 | **0.00498** |
| PeT over U-Net | | –26.4% | –15.1% | –11.6% | –65.4% | –54.1% | –65.3% |

The Transformer-based models significantly outperform the baseline U-Net in all measured aspects. Specifically, when comparing the PeT-CNN model to the U-Net-128, which has a similar number of trainable parameters, the PeT model achieves a reduction of 11% to 26% in surface quantity prediction errors and a 54% to 65% reduction in errors for integrated aerodynamic coefficients.

Between the two Transformer architectures, the PeT architecture achieves higher accuracy in predicting surface distributions, while the ViT architecture performs better in predicting global aerodynamic coefficients. Notably, given that ViT contains significantly fewer trainable parameters than PeT, it may be considered a more efficient and scalable solution for this problem. This performance difference may be attributed to the nature of the input data: since it is

DOI: 10.13009/EUCASS2025-675



defined on a structured mesh, ViT's patch-based representation is more naturally aligned, whereas PeT is better suited for field data defined over irregular domains such as point clouds.

Among the ViT variants, the implementation using learnable positional embeddings consistently yields lower prediction errors compared to fixed sinusoidal embeddings. This suggests that the structured mesh input already encodes sufficient positional information, and the addition of external sinusoidal embeddings may interfere with this representation, thereby degrading performance. Furthermore, the use of non-square patch partitioning significantly worsens ViT's accuracy, indicating that maintaining consistent patch aspect ratios is crucial for effective spatial encoding.

For the PeT architecture, the two implementations of slice weight computation result in relatively similar performance. The CNN-based implementation slightly outperforms the point-wise linear approach. However, considering its substantially larger number of parameters, the performance gain may not justify the additional computational cost.

Figure 11 presents two examples demonstrating the performance of the PeT-CNN model on the test dataset. Even though both wing geometries were entirely unseen during training, the PeT-CNN model accurately predicts the surface pressure coefficient distributions when compared to high-fidelity CFD results. Key aerodynamic features—such as shock waves, suction peaks, and even flow separation regions—are well captured by the model, highlighting its ability to generalize to new geometries.

These results confirm that the pretrained PeT-CNN model can provide fast and reliable predictions for complex wing flow fields. To facilitate practical application, we have also developed an interactive interface capable of generating wing surface flow fields and aerodynamic coefficients based on input geometries. The interface is available through our GitHub repository: https://github.com/YangYunjia/flogen.

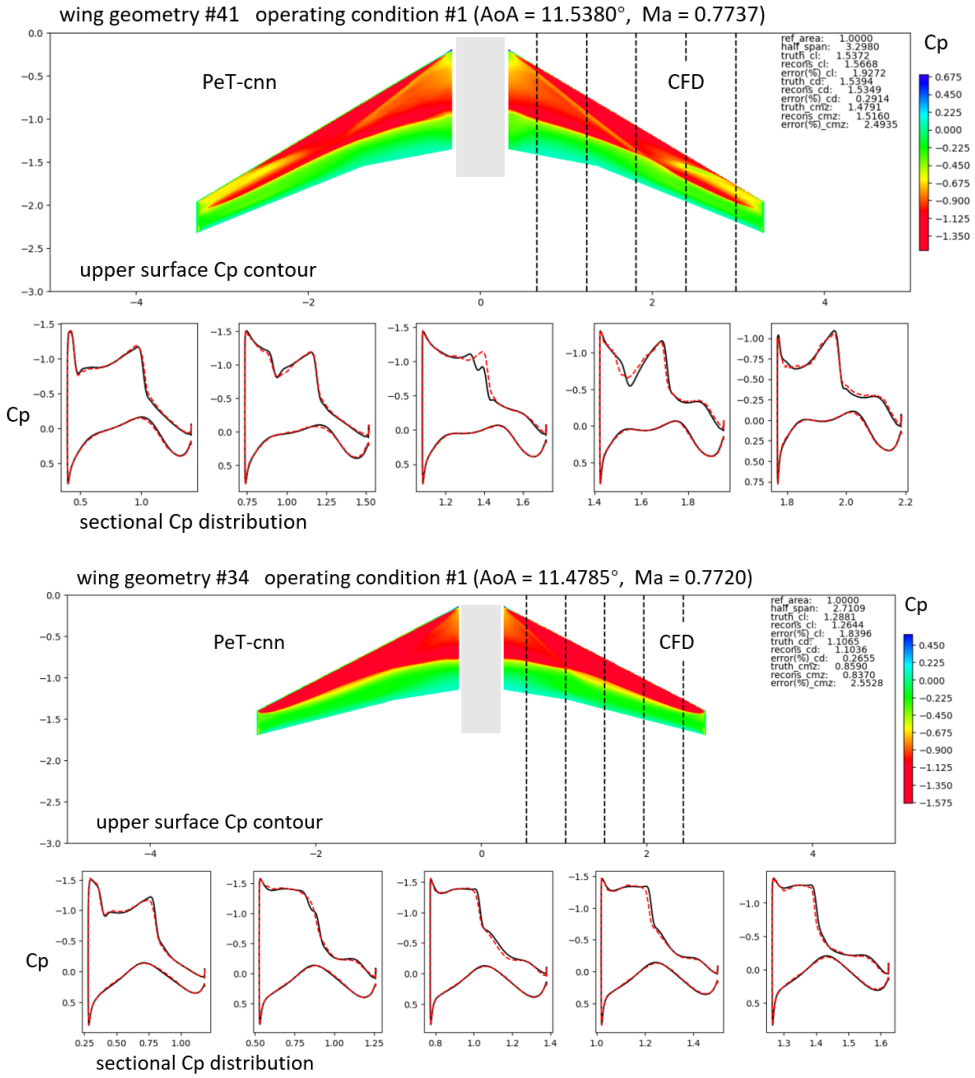



Figure 11: Surface pressure coefficients prediction results of the PeT-CNN architecture

DOI: 10.13009/EUCASS2025-675



## 5. Conclusion

Fast prediction of wing surface flow fields is critical for efficient aircraft design. This study advances flow field prediction models toward real-world wing configurations through two key contributions: (1) the construction of a general wing flow field dataset based on representative and engineering-practical transonic wing geometries, and (2) the development of Transformer-based neural network architectures, the ViT and PeT, for fast and accurate surface flow prediction.

Extensive experiments show that both Transformer-based models significantly outperform the baseline U-Net across all evaluation metrics. PeT demonstrates superior accuracy in resolving detailed surface features, while ViT performs better in estimating integrated aerodynamic coefficients with a much smaller model size. Architectural choices such as learnable positional embeddings and square patch slicing are shown to be critical in maximizing performance.

Reflecting on the development of CFD in the late 20th century, we recall that it too began with two-dimensional simulations and gradually evolved to handle three-dimensional wings and eventually complex, realistic configurations, and ultimately accepted as a vital tool in aircraft design. We believe that machine learning technologies are poised to follow a similar trajectory, and with continued advancement, will become a powerful and trusted component in the future of high-efficiency aircraft design.

## Acknowledgement

This work was supported by the National Natural Science Foundation of China Nos. 92471205. Artificial intelligence is used to polish the expressions in the paper.

## References

[1] Yang, Yunjia, Runze Li, Yufei Zhang, and Haixin Chen. 2024. “Fast Buffet-Onset Prediction and Optimization Method Based on Pretrained Flowfield Prediction Model.” AIAA Journal 62 (8): 2979–95.

[2] Zhang, Jianshe, Lin Li, Xu Dong, Ziqing Zhang, Yanfeng Zhang, and Xingen Lu. 2023. “A Discrete Adjoint Framework Coupled with Adaptive PCE for Robust Aerodynamic Optimization of Turbomachinery under Flow Uncertainty.” Aerospace Science and Technology 142 (November):108592.

[3] Li, Jichao, Xiaosong Du, and Joaquim R. R. A. Martins. 2022. “Machine Learning in Aerodynamic Shape Optimization.” Progress in Aerospace Sciences 134 (October): 100849.

[4] Li, Jichao, Mohamed Amine Bouhlel, and Joaquim R. R. A. Martins. 2019. “Data-Based Approach for Fast Airfoil Analysis and Optimization.” AIAA Journal 57 (2): 581–96.

[5] Du, Xiaosong, Ping He, and Joaquim R. R. A. Martins. 2021. “Rapid Airfoil Design Optimization via Neural Networks-Based Parameterization and Surrogate Modeling.” Aerospace Science and Technology 113 (June): 106701.

[6] Renganathan, S. Ashwin, Romit Maulik, and Jai Ahuja. 2021. “Enhanced Data Efficiency Using Deep Neural Networks and Gaussian Processes for Aerodynamic Design Optimization.” Aerospace Science and Technology 111 (April): 106522.

[7] Li, Jichao, and Mengqi Zhang. 2021. “Data-Based Approach for Wing Shape Design Optimization.” Aerospace Science and Technology 112 (May): 106639.

[8] Yang, Yunjia, Jiazhe Li, Runze Li, Yufei Zhang, and Haixin Chen. 2023. “Interactive Optimization of Fluidic Injection for Single Expansion Ramp Nozzle Based on a Modified Autoencoder.” In Aerospace Europe Conference 2023 – 10TH EUCASS – 9TH CEAS. Vol. 2023 – 343. https://doi.org/10.13009/EUCASS2023-343.

[9] Hu, Jiawei, and Weiwei Zhang. 2023. “Flow Field Modeling of Airfoil Based on Convolutional Neural Networks from Transform Domain Perspective.” Aerospace Science and Technology 136 (May):108198.

[10] Zhao, Jiachi, Lifang Zeng, and Xueming Shao. “A Novel Prediction Method for Unsteady Aerodynamic Force on Three-Dimensional Folding Wing Aircraft.” Aerospace Science and Technology 137 (June 2023): 108287.

[11] Zhang, Zilan, Yu Ao, Shaofan Li, and Grace X. Gu. “An Adaptive Machine Learning-Based Optimization Method in the Aerodynamic Analysis of a Finite Wing under Various Cruise Conditions.” Theoretical and Applied Mechanics Letters 14, no. 1 (January 2024): 100489.

[12] Castellanos, Rodrigo, Jaime Bowen Varela, Alejandro Gorgues, and Esther Andrés. “An Assessment of Reduced-Order and Machine Learning Models for Steady Transonic Flow Prediction on Wings.” In 33rd Congress of the

DOI: 10.13009/EUCASS2025-675



International Council of the Aeronautical Science. Stockholm, Sweden, 2022.

[13] Lyu, Yanfang, Xiaoyu Zhao, Zhiqiang Gong, Xiao Kang, and Wen Yao. 2023. “Multi-Fidelity Prediction of Fluid Flow Based on Transfer Learning Using Fourier Neural Operator.” Physics of Fluids 35 (7): 077118.

[14] Catalani, Giovanni, Siddhant Agarwal, Xavier Bertrand, Frédéric Tost, Michael Bauerheim, and Joseph Morlier. 2024. “Neural Fields for Rapid Aircraft Aerodynamics Simulations.” Scientific Reports 14 (1): 25496.

[15] Fonzi, Nicola, Steven L. Brunton, and Urban Fasel. 2023. “Data-Driven Modeling for Transonic Aeroelastic Analysis.” Journal of Aircraft, October, 1–13.

[16] Immordino, Gabriele, Andrea Da Ronch, and Marcello Righi. 2024. “Steady-State Transonic Flowfield Prediction via Deep-Learning Framework.” AIAA Journal 62 (5): 1915–31.

[17] Lei, Yuqi, Xiaomin An, Yihua Pan, Yue Zhou, and Qi Chen. 2024. “Prediction of Pressure Distribution and Aerodynamic Coefficients for a Variable-Sweep Wing.” Aerospace Science and Technology 155 (December):109706.

[18] Li, Jichao, and Mengqi Zhang. 2021. “Data-Based Approach for Wing Shape Design Optimization.” Aerospace Science and Technology 112 (May):106639.

[19] Hines, Derrick, and Philipp Bekemeyer. 2023. “Graph Neural Networks for the Prediction of Aircraft Surface Pressure Distributions.” Aerospace Science and Technology 137 (June):108268.

[20] Zuo, Kuijun, Zhengyin Ye, Xianxu Yuan, and Weiwei Zhang. 2025. “Flow3DNet: A Deep Learning Framework for Efficient Simulation of Three-Dimensional Wing Flow Fields.” Aerospace Science and Technology 159 (April):109991.

[21] Yang, Yunjia, Runze Li, Yufei Zhang, Lu Lu, and Haixin Chen. 2024. “Rapid Aerodynamic Prediction of Swept Wings via Physics-Embedded Transfer Learning.” AIAA Journal, December, 1–15.

[22] Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, et al. 2021. “An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale.” arXiv. https://doi.org/10.48550/arXiv.2010.11929.

[23] Wu, Haixu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. 2024. “Transolver: A Fast Transformer Solver for PDEs on General Geometries.” arXiv. http://arxiv.org/abs/2402.02366.

[24] Vassberg, John, Mark Dehaan, Melissa Rivers, and Richard Wahls. “Development of a Common Research Model for Applied CFD Validation Studies.” In 26th AIAA Applied Aerodynamics Conference. Honolulu, Hawaii: American Institute of Aeronautics and Astronautics, 2008.

[25] Sclafani, Anthony J., John C. Vassberg, Neal A. Harrison, Christopher L. Rumsey, S. Melissa Rivers, and Joseph H. Morrison. 2008. “CFL3D/OVERFLOW Results for DLR-F6 Wing/Body and Drag Prediction Workshop Wing.” Journal of Aircraft 45 (3): 762–80.

[26] Vassberg, John, Anthony Sclafani, and Mark DeHaan. 2005. “A Wing-Body Fairing Design for the DLR-F6 Model: A DPW-III Case Study.” In 23rd AIAA Applied Aerodynamics Conference. Toronto, Ontario, Canada: American Institute of Aeronautics and Astronautics.

[27] Orlita, Martin, and Roelof Vos. 2017. “Cruise Performance Optimization of the Airbus A320 through Flap Morphing.” In 17th AIAA Aviation Technology, Integration, and Operations Conference. Denver, Colorado: American Institute of Aeronautics and Astronautics.

[28] Li, Runze, Kaiwen Deng, Yufei Zhang, and Haixin Chen. “Pressure Distribution Guided Supercritical Wing Optimization.” Chinese Journal of Aeronautics 31, no. 9 (September 2018): 1842–54.

[29] Lyu, Zhoujie, Gaetan K. W. Kenway, and Joaquim R. R. A. Martins. 2015. “Aerodynamic Shape Optimization Investigations of the Common Research Model Wing Benchmark.” AIAA Journal 53 (4): 968–85.

[30] Mader, Charles A., Gaetan K. W. Kenway, Anil Yildirim, and Joaquim R. R. A. Martins. 2020. “ADflow: An Open-Source Computational Fluid Dynamics Solver for Aerodynamic and Multidisciplinary Optimization.” Journal of Aerospace Information Systems 17 (9): 508 – 27.

[31] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. n.d. “Attention Is All You Need.”