White Rose Research Online

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk
Universities of Leeds, Sheffield and York

Deposited via The University of Leeds.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/id/eprint/165644/

Version: Accepted Version

Proceedings Paper:
Chen, D, Gao, X, Xu, C et al. (2020) FlowGAN: A Conditional Generative Adversarial Network for Flow Prediction in Various Conditions. In: 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence. 32nd International Conference on Tools with Artificial Intelligence (ICTAI 2020), 09-11 Nov 2020, Baltimore, MD, USA. IEEE. ISBN: 978-1-7281-9228-4. ISSN: 2375-0197. EISSN: 2375-0197.

https://doi.org/10.1109/ICTAI50040.2020.00057

© 2020 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Reuse
Items deposited in White Rose Research Online are protected by copyright, with all rights reserved unless indicated otherwise. They may be downloaded and/or printed for private study, or other acts as permitted by national copyright laws. The publisher or other rights holders may allow further reproduction and re-use of the full text version. This is indicated by the licence information on the White Rose Research Online record for the item.

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

UNIVERSITY OF LEEDS
University of Sheffield
UNIVERSITY of York

FlowGAN: A Conditional Generative Adversarial Network for Flow Prediction in Various Conditions

Donglin Chen*[1], Xiang Gao*[1,2], Chuanfu Xu†[1,2], Shizhao Chen[1], Jianbin Fang[1], Zhenghua Wang[1], and Zheng Wang[3]

[1]College of Computer, National University of Defense Technology, China
{chendonglin14, gaoxiang12, xuchuanfu, chenshizhao12, j.fang, zhhwang}@nudt.edu.cn
[2]State Key Laboratory of High Performance Computing,
National University of Defense Technology, China
[3]University of Leeds, United Kingdom,
z.wang5@leeds.ac.uk.

***Abstract*—Many flow-related design optimization problems like aircraft and automobile aerodynamic design are solved via computational fluid dynamics (CFD) simulations. However, CFD simulations are known to be resource-demanding and time-consuming. Deep learning (DL) is emerging as a viable means to accelerate CFD simulations by directly predicting the outcomes of multiple simulation iterations. While promising, existing DL-based models have to be re-trained whenever the flow condition changes, which incurs significant training overhead for real-life scenarios with a wide range of flow conditions. This paper presents FLOWGAN, a novel conditional generative adversarial network for accurate prediction of flow fields in various conditions. FLOWGAN is designed to directly obtain the generation of solutions to flow fields in various conditions based on observations rather than re-training. As FLOWGAN does not rely on knowledge of the underlying governing equations, it can quickly adapt to various flow conditions and avoid the need for expensive re-training. We evaluate FLOWGAN by applying it to scenarios of simulating both the whole flow field and selected regions of interest (RoI). Compared to the state-of-the-art DL based methods, FLOWGAN significantly reduces the prediction errors by 2.27% while exhibiting a better generalization ability.**

***Keywords*-Flow fields prediction; Multi-source data processing; GAN; Predictive performance.**

I. INTRODUCTION

Computational fluid dynamics (CFD) simulations are the fundamental methodology for aerodynamics related design, analysis and optimization. Traditional CFD methods like finite difference and finite volume methods [1] must iteratively solve the partial differential equations (e.g., the Navier-Stokes equations [2]) of fluid flow. These high-fidelity CFD methods can provide reliable and relatively cheap means of analysis compared with experimental methods, and can flexibly handle different boundary conditions. However, these CFD simulation methods are known to be computation-resource-demanding and time-consuming [3]. The expensive simulation overhead and resource requirement prevent iterative design space exploration and hinder quick design choice evaluation.

In recent years, data-driven approaches have been proposed to speed up CFD simulations by employing deep neural networks (DNNs) [4]–[6] to directly predict the simulation outcomes. A DNN based approach works by first learning successively higher orders of features from flow data generated by a full-order CFD solver. The learned model can then be applied to predict the flow fields of *unseen* flow problems, by taking as input a representation of the flow conditions and geometry shapes (e.g., often expressed as a 2D matrix which can be visualized as an *artificial image* to serve as input to a DNN model), and predicting the performance metrics (e.g., the velocity fields). By using a model inference to substitute the many computation iterations that a CFD solver requires, predictive modeling can significantly reduce the turnaround time of generating flow data.

While promising, emerging approaches focus on flow fields prediction under fixed flow conditions [4], [7] (e.g., by assuming the flow parameters are unchanged). The flow conditions are typically quantified by these flow parameters such as the Reynolds number[1] and the angle of attack in aerodynamic design. In practice, the flow conditions may frequently change during design time. As a result, a model trained for given flow parameters will become out-of-date when the flow condition changed. Some of the recent studies have attempted to deal with various flow conditions. These methods apply extra conversion to flow parameters. However, these conversion methods either only work for a limited set of specific flow parameters like the angle of attack [9], or remain time-consuming for predicting the whole flow fields because of using a point-by-point prediction approach [10].

*These authors contributed equally to this work.
†Corresponding author.

[1]The Reynolds number [8] is widely used in CFD simulations for predicting flow patterns in different fluid flow situations. It describes the ratio of inertial forces to viscous forces in a flowing fluid.

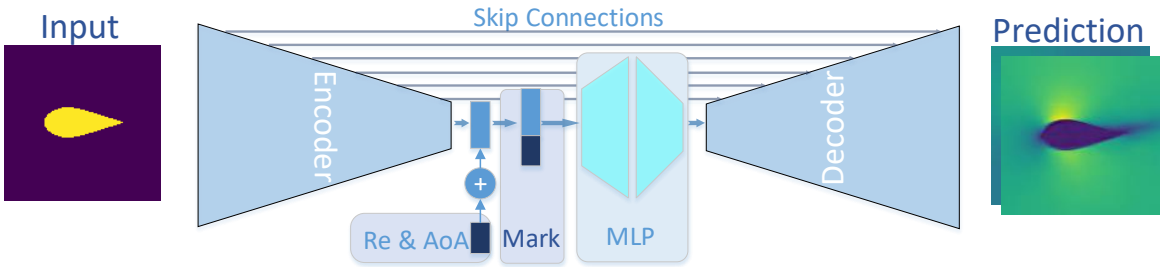


Figure 1: A high-level overview of the generator (G). Re represents the Reynolds number, AoA is the angle of attack. *Mark* is the result of concatenating the airfoil parameters and flow parameters.

This paper presents FLOWGAN, a novel conditional generative adversarial network (cGAN) for flow field prediction. FLOWGAN is designed to directly predict the outcomes of Reynolds-averaged Navier–Stokes (RANS) simulations, a widely used CFD simulation method, from various boundaries and flow conditions (including the airfoil geometry, the Reynolds number, and the angle of attack). We design a novel generator based on the U_Net network [11] to generate the predictions of flow fields and then utilize a multilayer perceptron (MLP) to merge geometry information and flow parameters at the bottom of the generator. While simple, the MLP is flexible and proven to be effective on the test dataset that contains simulations from various flow conditions. We demonstrate how cGAN and the U_Net architecture can be combined to build a generative adversarial network (GAN) to perform a one-to-one mapping from given boundary conditions and the geometry shape to its corresponding flow fields.

We evaluate FLOWGAN by applying it to a large-scale airfoil dataset to predict both the whole flow filed and specific regions of interest. Experimental results show that FLOWGAN can effectively handle various flow conditions, delivering better prediction accuracy over the state-of-the-art methods. The key contribution of our work is a general cGAN for flow fields that can adapt to various flow conditions. FLOWGAN can be used to learn causal models directly from experimental observations of flow fields where the underlying physical progress is complicated or unknown.

II. RELATED WORK

To overcome the drawbacks of CFD methods, which is time-consuming and resource-demanding. Many researchers have done numerous studies on data-driven methods based on traditional machine learning, including polynomial regression, support vector machines, and artificial neural networks [12]–[14]. These methods show some success in small-scale settings but cannot scale to the whole flow field. When dealing with elaborate designs, the data building progress requires extensive involvement of domain experts. In this work, we tackle these issues by developing a predictive model via deep learning methods. Our approach transfers shape boundaries to image-like inputs that are suitable for neural network training, which can simplify data preparation and generate a prediction for the entire flow fields.

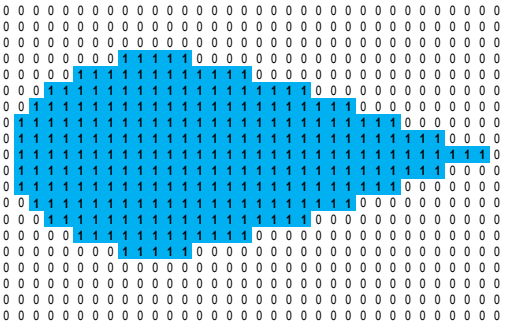


Figure 2: The flow fields data representation.

For applying deep learning to fluid dynamics, [5] construct specialized neural networks with embedded invariance for turbulence modeling to predict the Reynolds stress tensor. For unsteady flow over a circular cylinder, [15] predict the flow fields using different deep learning networks to extract both spatial and temporal features of the input flow field, which could be considered as video prediction. [4] predict steady flow fields around blunt objects under fixed flow conditions by establishing an encoder-decoder convolutional neural network. They use the signed distance function (SDF) to represent geometries, which is more demanding than our methods. [6] extend the work of [4] by introducing a fully connected (FC) layer to the networks to deal with various flow conditions. As a result, it is deficient to simply fuse the parameters into the network using an FC layer as we show later in our paper. [9] apply U_Net based networks to quantify uncertainties and improve Reynolds Averaged Navier-Stokes (RANS) models. Remarkable inference performance has been obtained but requires extra conversion from flow parameters to input feature maps of freestream. By integrating cGAN and MLP, our work contributes to simplifying the predictions under various flow conditions while providing convincing results.

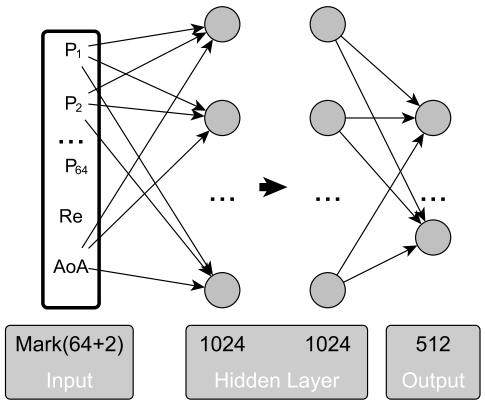


Figure 3: The structure of MLP network. $P_1 \sim P_{64}$ are the geometry parameters extracted by the encoder.

III. Background

In our paper, we mainly focus on deep learning models for the inference of the incompressible RANS solutions solved by the finite volume method (FVM). Traditionally, CFD simulates the turbulence flow fields by solving the governing equations of discrete fluid based on FVM and the turbulence equations. For 2D incompressible steady turbulence problems, the RANS governing equation can be simplified as follows:

$$\begin{aligned} &\nabla \cdot u = 0 \\ &\nabla \cdot (uu) = \nabla \cdot (\nu \nabla u) - \nabla p \end{aligned} \tag{1}$$

where u is the velocity vector of the control volume and ν is the kinematic viscosity coefficient. For incompressible flow, the density ρ is fixed and the p in Equation 1 represents the pressure divided by density.

According to the Boussinesq eddy viscosity hypothesis [16], the viscosity coefficient of RANS equations for turbulence is composed of laminar viscosity coefficient and turbulent viscosity coefficient, $\nu = \nu_L + \nu_T$ where ν_L is given by Sutherland's law [17] and ν_T is computed by solving the turbulence model equations.

The turbulence model used in our experiments is the Spalart-Allmaras (SA) model [18], a one-equation turbulence model that has been developed primarily for aerodynamic flows. The transport equation for ν_T is given by:

$$\begin{aligned} \frac{D\nu_T}{Dt} =& \frac{1}{\sigma}\left[\nabla \cdot ((\nu_L + \nu_T)\nabla \nu_T) + c_{b2}(\nabla \nu_T)^2\right] + c_{b1}\tilde{S}\nu_T - \\ & c_{w1} f_w \left[\frac{\nu_T}{d}\right]^2 \end{aligned} \tag{2}$$

where σ is a turbulent Prandtl (a free constant), and there are other two free constants c_{b1} and c_{b2}. SA has two conditions for three free constants (σ, c_{b1}, and c_{b2}), leaving a one dimensional family of solutions parametrized by the Prandtl number σ. c_{w1} is determined by c_{b1} and c_{b2}. d indicates the nearest distance to the wall. $\tilde{S}$ is computed by d and velocity u. f_w is a non-dimensional function about $\tilde{S}$ and ν_T to overcome the problem that the *destruction* term decays too slowly in the outer region of the boundary layer.

Based on the governing equations (Equation 1,2), CFD solver can solve the equations iteratively and simulate complex turbulent flows. This processing requires a very fine discretization of space-time and often relies on high-performance computing (HPC). When the flow conditions change, CFD has to start the time-consuming simulations over again. Therefore, we propose FlowGAN that can be used to directly generate simulations from observations with high fidelity and is friendly to deal with the various flow conditions.

IV. Methodology

A. Roadmap

Recall that our goal is to achieve effective predictions of flow fields in various flow conditions. We first need to reduce the high-dimension of geometry and parameterize the geometry to integrate with the flow parameters. By utilizing the U_Net architecture, the encoding part can extract the geometry parameters and the decoding part can restructure the 2D velocity flow fields. We then exploit the feasibility of developing an MLP network to integrate the flow parameters and extracted geometry information. Finally, like the image-to-image regression, the mapping from the given conditions to predictions is built by our novel cGAN network. Figure 1 shows our way to build the generator.

B. Data Representation

To predict flow fields over various objects with deep networks, we first need to have a suitable way to represent the object geometry and domain boundaries. In this paper, we focus on the fluid domains around airfoil shapes. Airfoil is the aircraft wing's cross-section, which has a significant influence on the aerodynamic performance of aircraft. The fluid domains are divided into Cartesian grids that can be regarded as images for deep networks as input feature maps. We use the binary representation to define input images and distinguish object boundaries in fluid domains. As shown in Figure 2, the blue cells are those solid parts that represent the geometry of the example 2D airfoil and assigned a value of 1 to indicate the object boundaries. Other pixels with value 0 stand for the fluid domain, and the corresponding pixels of the output feature maps represent the approximation of flow quantities after the end-to-end learning.

C. Multi-source Boundary Conditions Processing

1) Airfoil Shape Parameterization: To integrate two different forms of boundary condition data: one is the image-like airfoil shape, the other is the global flow parameters such as Re and AoA, we choose to parameterize the airfoil

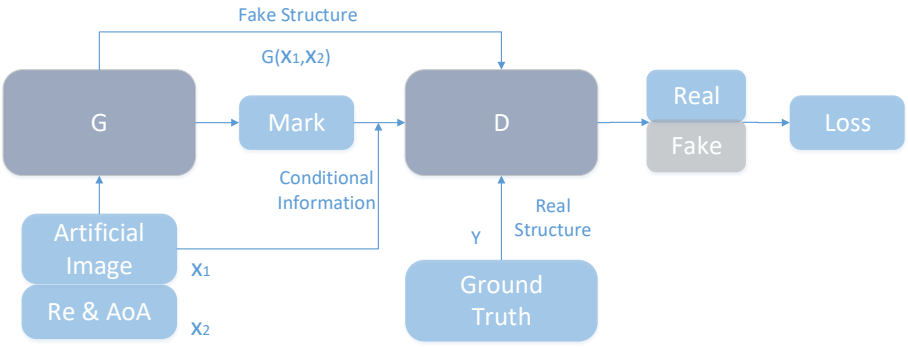


Figure 4: The detailed flowchart of cGAN approach to flow fields prediction.

Table I: Detailed description of the generator and discriminator architectures. For input and output, 128 x 128 x 1 indicates the size of the feature map is 128x128 with 1 channel. In conventional layers, 4 x 4 indicates the convolutional filter size, 64 indicates the number of filters, and 2 indicates the stride.

	G_encoder	G_decoder	D
Input	128 x 128 x 1	1 x 1 x 512	128 x 128 x 4
(DE)CONV1	4 x 4, 64, 2	2 x 2, 512, 2	4 x 4, 16, 2
(DE)CONV2	4 x 4, 128, 2	4 x 4, 256, 2	4 x 4, 16, 2
(DE)CONV3	4 x 4, 128, 2	4 x 4, 256, 2	4 x 4, 32, 2
(DE)CONV4	4 x 4, 256, 2	4 x 4, 128, 2	4 x 4, 32, 2
(DE)CONV5	4 x 4, 256, 2	4 x 4, 128, 2	4 x 4, 64, 2
(DE)CONV6	4 x 4, 512, 2	4 x 4, 64, 2	4 x 4, 64, 2
(DE)CONV7	2 x 2, 512, 2	4 x 4, 64, 2	4 x 4, 128, 2
FCONV1	...	3 x 3, 64, 1	3 x 3, 16, 1
FCONV2	...	3 x 3, 64, 1	3 x 3, 16, 1
FCONV3	...	3 x 3, 2, 1	3 x 3, 1, 1
FC layer	512⇒64	...	64+2⇒128 x 128
Output	1 x 1 x 512	128 x 128 x 2	1 x 1

shape and extract the airfoil shape parameters that can blend with flow parameters. Considering the importance of airfoil shape information: the flow quantities change rapidly in the area around the geometry. We use the U_Net based network to extract the geometry parameters. Different from the conventional shape parameterization technologies or traditional autoencoder network [19], U_Net can not only obtain low-dimensional airfoil shape parameters but reserve detailed geometry information through skip connections. Section IV-D introduces the details about the U_Net networks.

2) MLP-based Integration: MLP is generally known as a neural network consisting of several neurons (also known as nodes) connected together to form a complex network. The neuron receives different kind of inputs signals and produces the outputs.

With the encoder of generator offering airfoil shape parameters ($P_1 \sim P_{64}$), we use a 3-layer MLP, each followed by a LeakyRelu activation layer. Figure 3 shows the structure of MLP network. The airfoil parameters along with flow parameters are concatenated as the *Mark* and fed as inputs to the MLP network. Then two hidden layers with 1024 neurons perform a nonlinear input-output mapping. Finally,

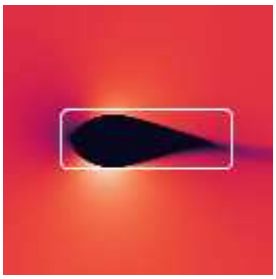


Figure 5: An example of RoI defined by the white box.

the outputs with a size of 512 reshape as the input feature maps for the decoder of the generator network.

D. Flow Prediction Network

Generative Adversarial Network (GAN) is a generative model formulated as a minimax two-player game between two models: (1) a generator G which creates samples that are intended to come from the same distribution as that of the real data and, (2) a discriminator D that determines whether the samples are from the generator or not. The cGAN extended the original GAN from an unsupervised

Table II: Comparing MRE, $\mathrm{MRE_{RoI}}$ and flexibility for different models. X and Y represent the velocity field for x- and y-directions perspectively.

Method	MRE			$\mathrm{MRE_{RoI}}$			**Flexibility**
Endec	6.08%	X Y	4.97% 14.38%	27.32%	X Y	27.14% 30.17%	☺
Unet	4.67%	X Y	3.12% 14.49%	10.91%	X Y	9.70% 20.93%	☹
FlowGAN	2.27%	X Y	1.81% 5.76%	9.56%	X Y	8.13% 20.83%	☺

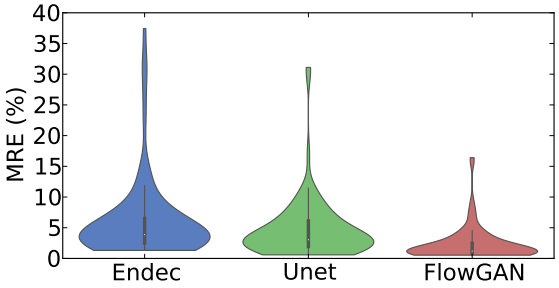


Figure 6: The distribution of MRE for different models on the test dataset.

method to a supervised one, which is more suitable for building the one-to-one mapping from the given conditions to predictions. Our model borrows idea from the conditional *Pixel2Pixel* GAN architecture [20]. Differently, we deal with the task of multi-source flow fields prediction, while *Pixel2Pixel* focuses on the solution to the image style transfer problem. Figure 4 shows how our cGAN-based model works. The inputs, including the artificial image of airfoil shape x_1 and flow parameters x_2 are fed to the generator. Then G outputs predictions of velocity fields and intermediate vector $mark$. The vector $mark$ and the artificial image are jointed as conditional information. The conditional information together with predictions $G(x_1, x_2)$ on the fake structure or the ground truth y computed by CFD methods are provided as input for the discriminator. Therefore, the objective of our cGAN can be expressed as:

$$\begin{aligned}\mathcal{L}_{cGAN}(G,D) =& \mathbb{E}_{x_1,x_2,y}[\log D(x_1, mark, y)] + \\ & \mathbb{E}_{x_1,x_2}[\log(1 - D(x_1, mark, G(x_1,x_2)))]\end{aligned} \tag{3}$$

$$\mathcal{L}_{cGAN}(G) = \mathbb{E}_{x_1,x_2}[\log(1 - D(x_1, mark, G(x_1,x_2)))] \tag{4}$$

Considering the generator is tasked to not only fool the discriminator but also generate outputs close to the ground truth, L_1 loss is applied for G when training the networks:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x_1,x_2,y}\left[\|y - G(x_1,x_2)\|_1\right] \tag{5}$$

Since cGAN is a minimax game, the discriminator and the generator work iteratively to carry out minimization and maximization on cross-entropy respectively, leading to our final objective (λ is the weight of L_1 loss):

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \lambda \mathcal{L}_{L1}(G) \tag{6}$$

1) Generator Network: G mainly comprises two parts: the left encoder with 7 contracting blocks and the right decoder with 7 expansive blocks, with an MLP network at the bottom of U_Net for knowledge integration. Each contracting block in the encoder is followed by a convolutional layer (CONV) with a stride of 2 for downsampling, an activation unit, and a batch normalization layer. The convolutional kernels have a size of 4×4, except for the one in the last contracting block because the size of its input feature map is only 1×1. To simplify the airfoil parameters, we use a fully connected (FC) layer to refine the encoded information at the bottleneck. For each expansive block in the decoder, we add an upsampling layer followed by an activation unit. We use transposed convolution (DECONV) to realize the upsampling layers since it allows the network to learn how to upsample optimally. Between the encoder and the decoder are the skip connections, concatenating all down-sampled feature maps from the contracting blocks to the corresponding ones in expansive blocks and doubling the number of channels.

The inputs of the network are the *artificial images* transformed from airfoil shapes, while the CFD solver provides the ground truth data. The outputs of the decoder have the same size of the inputs but with two channels, representing the velocity field for x- and y-directions respectively.

2) Discriminator Network: D is provided with three inputs: the vector *Mark* and the artificial image and the predicted results or ground truth data. We use an FC layer to extend *Mark* and reshape it as a feature map with the same size of the other two inputs, and then concatenate them along the channel axis to form feature maps with four channels. The feature maps are fed to D with seven convolution layers to obtain the judgment of *real/fake*. Because the task of D is more straightforward than that of G, the network parameters of D are much fewer.

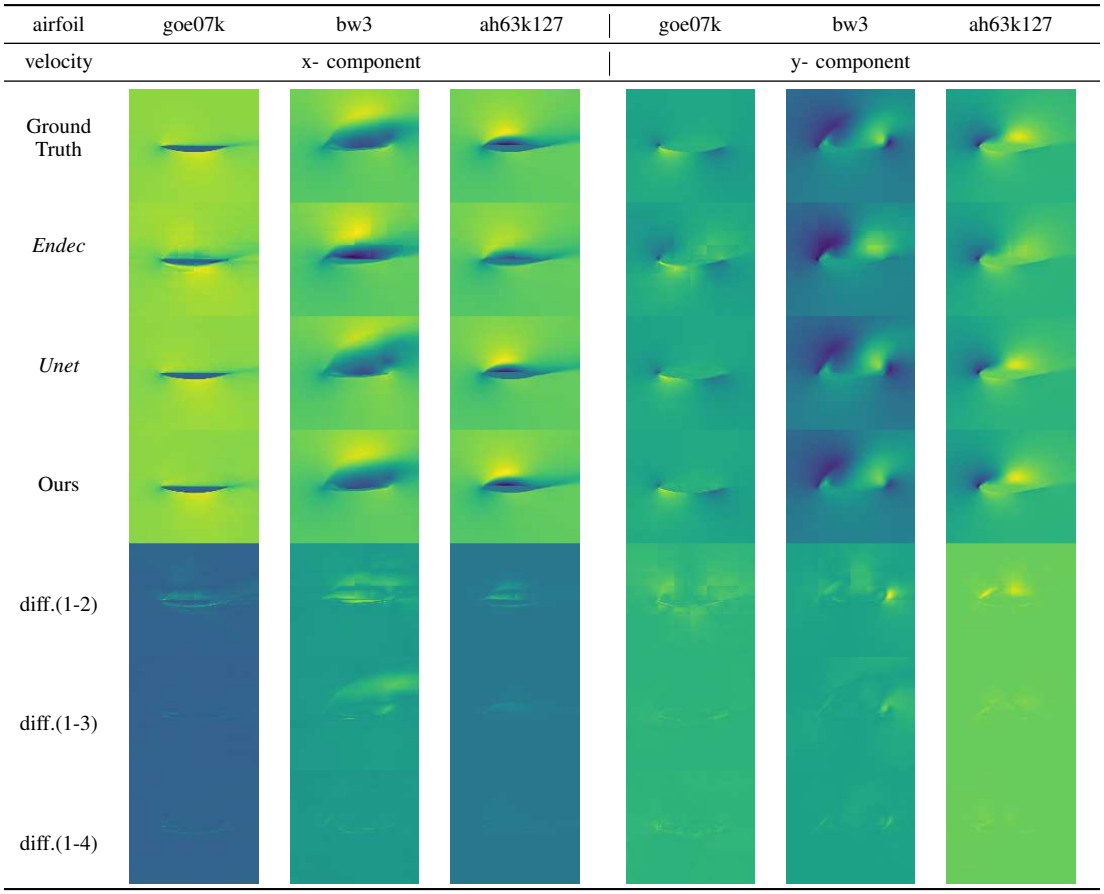
Both D and the decoder of G have three conventional layers (FCONV) with a stride of 1 for fine-tuning the output feature maps. Table I shows the details of the structure of the flow prediction networks.

V. Experimental Setup

A. Implementation Details

Our models are built on an NVIDIA Tesla V100 GPU with PyTorch 1.1.0. We train the velocity field predictive model with the adaptive moment estimation (Adam) optimizer. To obtain stable results and avoid overfitting, the training proceeds up to 200 epochs. The hyper-parameter λ of the loss function is set to 10 after searching for optimal parameters in [0, 100]. We set the initial learning rate at 1×10^{-3} with learning rate decay. The batch size is set to 32. As for

Table III: Comparison of 2D velocity field between *OpenFOAM* solver, baseline methods and our predictive model. The *diff.(1-2)* means the difference between the first row and the second row and the similar below.



activation functions, we use leaky ReLU functions with a slope of 0.2 in both the generator and the discriminator.

B. Data Preparation

A total number of 1525 different airfoil shapes from the UIUC database [21] are used in our paper. With a range of Reynolds numbers between [0.5, 5] million, and angles of attack in the range of $\pm$22.5 degrees, 1450 airfoil shapes are considered to generate 6400 training cases (400 for validation) for the cGAN network. The testing sets including 100 samples are produced with the rest 75 airfoil shapes in the same way. All the training CFD data (i.e., the ground-truth velocity) is generated with the open-source code *OpenFOAM* by solving RANS equations [22] using the *SimpleFoam* solver. Moreover, the simulated velocity fields and airfoil geometries are then mapped into a Cartesian grid with the size of 128x128.

C. Evaluation Metrics

We conduct comprehensive evaluations of the proposed framework in this section. The work of Afshar et al. [6] and Thuerey et al. [9] are introduced as baseline models and we call them *Endec* and *Unet*. To evaluate the accuracy of our predictive model we define:

1) MRE*:* : the mean relative error of velocity for the whole 2D flow fields:

$$\mathrm{MRE}=\sum_{\mathrm{l}=1}^{\mathrm{N}}\frac{\sum\limits_{\mathrm{i}=1}^{\mathrm{n_x}}\sum\limits_{\mathrm{j}=1}^{\mathrm{n_y}}\left(\left|\mathrm{u}_{\mathrm{ij}}^{\mathrm{l}}-\overline{\mathrm{u}}_{\mathrm{ij}}^{\mathrm{l}}\right|+\left|\mathrm{v}_{\mathrm{ij}}^{\mathrm{l}}-\overline{\mathrm{v}}_{\mathrm{ij}}^{\mathrm{l}}\right|\right)}{\mathrm{N}\sum\limits_{\mathrm{i}=1}^{\mathrm{n_x}}\sum\limits_{\mathrm{j}=1}^{\mathrm{n_y}}\left(\left|\mathrm{u}_{\mathrm{ij}}^{\mathrm{l}}\right|+\left|\mathrm{v}_{\mathrm{ij}}^{\mathrm{l}}\right|\right)} \tag{7}$$

2) $\mathrm{MRE_{RoI}}$*:* : the mean relative error of velocity for the regions of interest for the airfoil:

$$\mathrm{MRE_{RoI}}=\sum_{\mathrm{l}=1}^{\mathrm{N}}\frac{\sum\limits_{\mathrm{i}=1}^{\mathrm{n_s}}\left(\left|\mathrm{u}_{\mathrm{ij}}^{\mathrm{l}}-\overline{\mathrm{u}}_{\mathrm{ij}}^{\mathrm{l}}\right|+\left|\mathrm{v}_{\mathrm{ij}}^{\mathrm{l}}-\overline{\mathrm{v}}_{\mathrm{ij}}^{\mathrm{l}}\right|\right)}{\mathrm{N}\sum\limits_{\mathrm{i}=1}^{\mathrm{n_s}}\left(\left|\mathrm{u}_{\mathrm{ij}}^{\mathrm{l}}\right|+\left|\mathrm{v}_{\mathrm{ij}}^{\mathrm{l}}\right|\right)} \tag{8}$$

where N is the size of test dataset and l indicates a certain sample, n_x and n_y are the numbers of cells (pixels) along the x- and y-direction respectively, u and v are the velocity components of ground truth for the x- or y-direction, $\overline{u}$ and

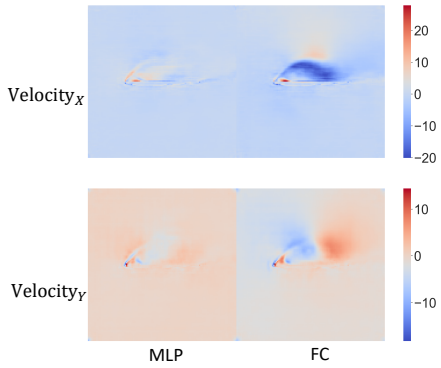
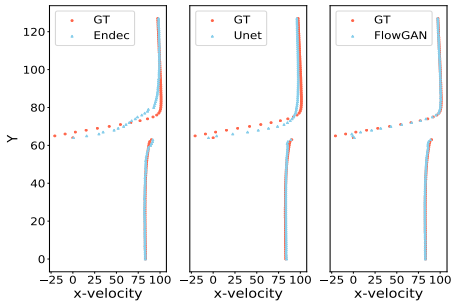
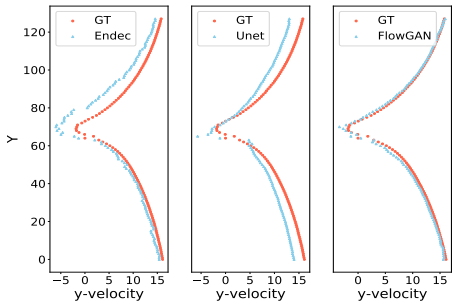


Figure 7: Difference of MLP and FC compared to ground truth.



(a) x-component



(b) y-component

Figure 8: Prediction of the x- and y-component velocity profile at the tail of airfoil *bw3* using different models compared with ground truth (GT).

$\overline{v}$ stand for the predicted velocity components accordingly. n_s is the number of cells in the *RoI* of the airfoil. Figure 5 shows an example of *RoI* defined by the white box, containing the domain of the car in our experiments. Note that the box is not fixed, and its size and location will adaptively change to contain the object geometry in the flow field.

VI. Experimental Results

A. Overall Results

Table II compares *Endec* and *Unet* to FlowGAN in terms of MRE, MRE_{RoI} and flexibility. For the prediction of the whole flow fields, FlowGAN outperforms its counterparts in both the velocity field for x- and y-directions. To give more details about the predictive accuracy of three models, the violin diagram Figure 6 describes the statistical distribution of MRE for all models on the whole test dataset. Here, the thick black line shows where 50% of the data locates, and the shape of the violin shows the data distribution. FlowGAN reduces MRE from 6.08% and 4.67% to 2.27%. We visualize several sample predictions in Table III. Intuitively, the predictions of our model are much closer to the CFD simulation results than *Endec* and *Unet*. The differences between the CFD simulation results and approximation models are shown in the last three columns, demonstrating the effectiveness of our network architecture.

We further compare the MRE in the region of interest (near the airfoil surface) where CFD experts are concerned because the velocity quantity changes fast in this region, and this area contains more useful information for aerodynamic design. Table II tells that MRE is much smaller than MRE_{RoI}, because it is more difficult for neural networks to predict accurately at the area near the airfoil surface compared with the region far away from the airfoil surface. The third column in Table II shows that FlowGAN gives a lower MRE_{RoI} for both x- and y-velocity compared to baselines, which indicates that FlowGAN is more capable of learning about boundary information and yields better performance.

B. The Impact of MLP

To verify the effect of MLP, we implement a counterpart for FlowGAN that uses an FC layer to integrate flow parameters (*Re* and *AoA*). Results on the test dataset show that FC gives a higher MRE than MLP with 5.44% to 2.66%. Figure 7 shows difference of predictions on airfoil *bw3* using MLP and FC compared with the ground truth. MLP delivers much fewer errors than its counterpart in both the velocity field for x- and y-directions, demonstrating the effectiveness of MLP for the integration of multisource data. Note that MLP increases the parameters of the network. Still, it is significant to enhance the representing power to learn from the flow parameters since these parameters have a global impact on the whole flow fields. Besides, we did additional experiments about the influence of the depth of MLP. The results indicate that 3-layer MLP is deep enough for the integration of two-way data, and more layers did not improve the results considerably.

C. Airfoil Wake Analysis

The flow field of airfoil wake is another significant area in CFD analysis and design. We conduct further evaluate the wake consistency of the models compared to CFD solver. A representative example is shown in Figure 8, which shows the x- and y-component velocity profile of the airfoil wakes (at the downstream location from the leading edge) of the CFD result and the predictions of models. The visual comparison shows that the predictions of our cGAN model

are in more agreement with the ground truth compared with *Endec* and *Unet*.

D. Flexibility Discussion

The *Unet* model is aimed at two fixed flow parameters, Re and AoA, and requires extra conversion from flow parameters to input feature map of freestream. This method is not general enough to solve flow field prediction problems with more flow parameters (e.g., the viscosity ν). The *Endec* model takes advantage of the FC layer to catenate the airfoil features and the flow control variables. While flexible, the previous results indicate that merely fusing the flow parameters to the networks like *Endec* may deliver imprecise predictions. FLOWGAN first refines airfoil parameters using a U_Net and then utilizes an MLP network to achieve the integration of airfoil parameters and flow parameters, outperforming the baselines on the large-scale 2D airfoil datasets. We argue FLOWGAN can be easily applied to more complicated flow conditions that contain more flow parameters.

VII. Conclusion and Future work

We have presented FLOWGAN, a novel cGAN framework for learning flow representation for CFD simulations under changing flow conditions. FLOWGAN takes as input an artificial image of airfoil shapes along with the Reynolds number and angle of attack to predict the solutions for the given boundary conditions and domain. It then uses an MLP to integrate the geometry parameters and flow parameters to generate the simulation outcomes. Experimental results show that FLOWGAN is highly accurate in predicting both the entier flow fields and regions of interest when compared to the state-of-the-art methods. Our future work will look into extending FLOWGAN to modeling 3D turbulent flows and incorporating the physical laws (e.g., conservation of mass and momentum).

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant No.61772542 and No.61972408.

References

[1] G. E. Forsythe and W. R. Wasow, *Finite-difference methods for partial differential equations /*, 1960.

[2] P. Constantin and C. Foias, *Navier-stokes equations*, 1988.

[3] N. Geneva and N. Zabaras, "Multi-fidelity generative deep learning turbulent flows," *ArXiv*, vol. abs/2006.04731, 2020.

[4] X. Guo, W. Li, and F. Iorio, "Convolutional neural networks for steady flow approximation," in *SIGKDD*, 2016.

[5] J. Ling, A. Kurzawski, and J. Templeton, "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance," *Journal of Fluid Mechanics*, vol. 807, pp. 155–166, 2016.

[6] S. Bhatnagar, Y. Afshar, S. Pan, K. Duraisamy, and S. Kaushik, "Prediction of aerodynamic flow fields using convolutional neural networks," *Computational Mechanics*, pp. 1–21, 2019.

[7] R. Wang, K. Kashinath, M. Mustafa, A. Albert, and R. Yu, "Towards physics-informed deep learning for turbulent flow prediction," in *NIPS*, 2019.

[8] J. Blazek, *Computational fluid dynamics: principles and applications*. Butterworth-Heinemann, 2015.

[9] N. Thuerey, K. Weissenow, L. Prantl, and X. Hu, "Deep learning methods for reynolds-averaged navier–stokes simulations of airfoil flows," *AIAA Journal*, pp. 1–12, 2019.

[10] V. Sekar and B. Khoo, "Fast flow field prediction over airfoils using deep learning approach," *Physics of Fluids*, vol. 31, p. 057103, 05 2019.

[11] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015.

[12] V. O. Balabanov, A. A. Giunta, O. Golovidov, B. Grossman, W. H. Mason, L. T. Watson, and R. T. Haftka, "Reasonable design space approach to response surface approximation," *Journal of Aircraft*, vol. 36, no. 1, pp. 308–315, 1999.

[13] D. D. Daberkow and D. N. Mavris, "New approaches to conceptual and preliminary aircraft design: A comparative assessment of a neural network formulation and a response surface methodology," 1998.

[14] M. Ahmed and N. Qin, "Surrogate-based aerodynamic design optimization: Use of surrogates in aerodynamic design optimization," *International Conference on Aerospace Sciences and Aviation Technology*, vol. 13, pp. 1–26, 05 2009.

[15] S. Lee and D. You, "Data-driven prediction of unsteady flow over a circular cylinder using deep learning," *Journal of Fluid Mechanics*, vol. 879, pp. 217–254, 2019.

[16] F. G. Schmitt, "About boussinesq's turbulent viscosity hypothesis: historical remarks and a direct evaluation of its validity," *Comptes Rendus Mécanique*, vol. 335, no. 9-10, 2007.

[17] W. Sutherland, "Lii. the viscosity of gases and molecular force," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 1893.

[18] P. Spalart and S. Allmaras, "A one-equation turbulence model for aerodynamic flows," *AIAA*, vol. 439, 01 1992.

[19] J. A. Samareh, J. A. Samareh, and B. B. Polynomial, "A survey of shape parameterization techniques," 1999.

[20] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *CVPR*, 2017.

[21] M. Selig, *UIUC airfoil data site*. Department of Aeronautical and Astronautical Engineering University of Illinois at Urbana-Champaign, 1996.

[22] S. S. Girimaji, "Partially-averaged navier-stokes model for turbulence: A reynolds-averaged navier-stokes to direct numerical simulation bridging method," 2006.