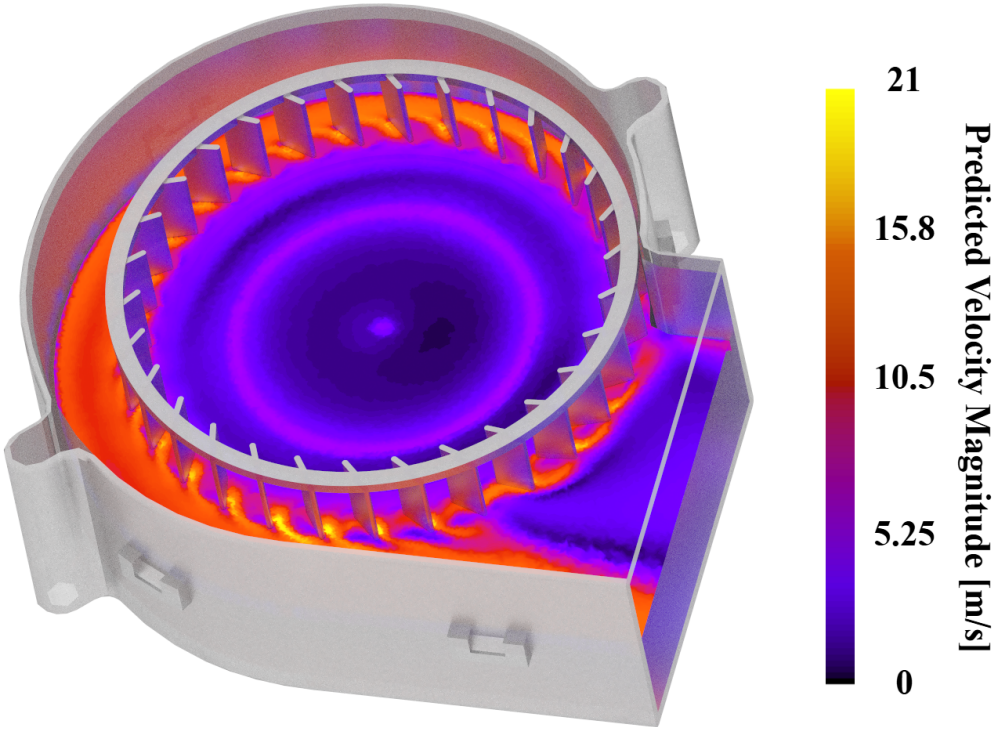
CHALMERS
UNIVERSITY OF TECHNOLOGY



CFD-Informed Neural Networks for Centrifugal Fan Design

Combining Simulation and Deep Learning to Predict the Performance of Parametrically Varied Fan Blades

TME131: Project in Applied Mechanics

CHRISTIAN BENSRYD
ALEXANDER KARLSSON
NELLIE KARLSSON
BERKEN SERBÜLENT
KARTHIK TAPASI HIMANTH

Department of Mechanics and Maritime Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025

TME131: Project in Applied Mechanics 2025

CFD-Informed Neural Networks for Centrifugal Fan Design

Combining Simulation and Deep Learning to Predict the Performance of Parametrically Varied Fan Blades

TME131: Project in Applied Mechanics

CHRISTIAN BENSRYD
ALEXANDER KARLSSON
NELLIE KARLSSON
BERKEN SERBÜLENT
KARTHIK TAPASI HIMANTH

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mechanics and Maritime Sciences
Division of Fluid Dynamics

Chalmers University of Technology
University of Gothenburg

Gothenburg, Sweden 2025

CFD-Informed Neural Networks for Centrifugal Fan Design
Combining Simulation and Deep Learning to Predict the Performance of Parametrically Varied Fan Blades

TME131: Project in Applied Mechanics
CHRISTIAN BENSRYD

ALEXANDER KARLSSON

NELLIE KARLSSON

BERKEN SERBÜLENT

KARTHIK TAPASI HIMANTH

Examiner: Håkan Johansson, Dario Maggiolo, Department of Mechanics and Maritime Sciences
Supervisors: Blanca Gonzalez Lozano, Anthony Vivek, André Fransson, AFRY AB

TME131 - 2025
Department of Mechanics and Maritime Sciences
Division of Fluid Dynamics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Render of the velocity magnitude, predicted by the developed surrogate model, for 30 blades with an angle of attack of 55 degrees, illustrated in Siemens Simcenter Star-CCM+.

Typeset in LaTeX
Gothenburg, Sweden 2025

CFD-Informed Neural Networks for Centrifugal Fan Design

CHRISTIAN BENSRYD, ALEXANDER KARLSSON, NELLIE KARLSSON, BERKEN SERBÜLENT, KARTHIK TAPASI HIMANTH

Department of Mechanics and Maritime Sciences
Division of Fluid Dynamics
Chalmers University of Technology

Abstract

This project presents the development of a surrogate model for centrifugal fan performance, focusing on the Same Sky CBM-97S series DC fan. The objective is to develop a validated CFD model of the centrifugal fan and to use the model to train neural networks that can predict the performance of varying blade geometries. To achieve this a steady state CFD model was developed in STAR-CCM+ using $k-\omega$ turbulence model and a moving reference frame to simulate the rotating impeller. After validation against experimental data, a simplified fan geometry was parametrized, and a dataset consisting of 200 simulations with varying blade count and blade angle of attack was generated.

Two different supervised Neural Networks were then trained: the first predicts the pressure and velocity fields across a 2D plane section, while the other estimates performance parameters such as outlet mass flow and pressure based on the geometric inputs. Both models demonstrated high accuracy. The flow field model achieved R^2 values above 0.94. The fan performance model showed mass flow and pressure predictions with errors below 6%.

The project shows that the neural networks trained on the CFD simulation data are able to accurately predict a two dimensional flow field and performance variables. Future work may involve extending to 3D field predictions using a physics-informed neural network, incorporating additional parameters such as blade length, and optimizing network architecture for enhanced performance.

Keywords: STAR-CCM+, Surrogate Model, Neural Networks, Machine Learning, Computational Fluid Dynamics, Centrifugal Fan, Performance Prediction.

Contents

Nomenclature

Abbreviations

BC	Boundary Condition
BI	Backwards-inclined
CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
FC	Forwards-curved
FVM	Finite Volume
GPU	Graphics Processing Unit
MAE	Mean Absolute Error
ML	Machine Learning
MRF	Moving Reference Frame
MSE	Mean Square Error
NN	Neural Network
RANS	Reynolds-averaged Navier-Stokes
ReLu	Rectified Linear Unit

Variables

μ	Dynamic viscosity
ν	Kinematic viscosity
ω	Specific dissipation rate
ρ	Fluid density
ε	Turbulent kinetic energy dissipation rate
$\vec{\Omega}$	Angular velocity vector
$\vec{r}$	Position vector
$\vec{u}_I$	Absolute velocity in inertial frame
$\vec{u}_R$	Relative velocity in rotating frame
F_1	Blending function between k–ω and k–ε models
f_i	Body force per unit mass in the i^{th} direction
k	Turbulent kinetic energy
p	Pressure
p	Static pressure
t	Time
v_i	Velocity component in the i^{th} direction
x_i	Spatial coordinate in the i^{th} direction

List of Figures

List of Tables

1
Introduction

1.1 Background

Centrifugal fans are integral components in various industrial and consumer applications, such as HVAC systems, electronic cooling, and ventilation systems. Their aerodynamic performance significantly influences system efficiency, energy consumption, and noise levels. Optimizing the blade geometry of these fans is crucial for enhancing overall performance, reducing operational costs, and minimizing environmental impact [1, 2].

Traditionally, the design and optimization of fan blades have relied on experimental testing and iterative computational fluid dynamics (CFD) simulations. CFD offers detailed insights into complex flow phenomena such as turbulence, vortices, and pressure gradients, but is computationally intensive, especially when multiple design iterations are required. This makes comprehensive design space exploration both time-consuming and expensive. [2]

To address these challenges, combining machine learning (ML) with CFD simulations could potentially be used to reduce computational cost. By training ML models on datasets generated from CFD results, it becomes possible to predict flow behavior and performance metrics across a range of geometries without the need for new simulations each time. This approach enables rapid evaluation of design variants and supports more efficient optimization workflows. [3]

1.2 Purpose & Problem Description

The primary objective of this study is to predict the behavior of a simplified fan, based on the Same Sky CBM-97S series DC centrifugal fan [4]. By combining validated CFD simulations with supervised machine learning, the aim is to develop a framework for predicting the aerodynamic behavior and performance of a fan with different blade configurations.

The specific goals of the study are as follows:

- Perform CFD simulations of the centrifugal fan and validate the results against experimental data provided in the manufacturer's datasheet. The validation is based on comparing the static pressure versus airflow curve obtained from simulations with the corresponding experimental results.
- Parameterize key geometric characteristics of the fan, such as number of blades and blade angle, to enable systematic design variation.
- Generate a dataset from the CFD simulations that can be used to train a supervised machine learning model.
- Develop a neural network based surrogate model capable of predicting detailed

pressure and velocity fields in a 2D fan domain for a wide range of blade geometries, enabling fast flow field estimation.
- Develop a neural network based surrogate model capable of predicting the average static pressure and mass flow at the fan outlet for a set of given blade parameters.

Ultimately, this study aims to integrate CFD and machine learning to enable rapid and reliable performance prediction of fan designs, thereby reducing the time and resources needed for iterative CFD evaluations.

1.3 Delimitation

As this project is part of a course, its duration is limited to one study period (9 weeks). Due to these time constraints, in-depth investigations into optimal CFD solver settings won't be prioritized. Instead, the focus will be on developing a validated simulation model for machine learning and geometry optimization.

To ensure feasibility, the simulation will be restricted to a single turbulence model, as more detailed simulations are too computationally expensive. Computational resources are provided by Chalmers e-Commons e-Infrastructure group (C3SE) funded by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) through the Swedish User and Project Repository (SUPR) [5, 6]. Since these resources have a maximum time budget, simulation and training runtime must be minimized. This will be done using a soft convergence criteria, and only the effects of number of blades and angle of attack will be investigated.

For the validation process, a simplified CAD model will be used. The simplified model will be used to reduce mesh size, and thus the complexity and simulation time. It will exclude cables and be sealed to ensure a single inlet and outlet. For the Machine Learning part, the impeller will be further simplified with straight blades to reduce the number of variables involved in the parameterizations, further reducing the complexity of the problem.

Machine Learning encompasses various approaches. We have chosen supervised learning since the goal is to predict fan performance based on a given set of parameters. While reinforcement learning is more suited for direct design optimization, our focus is performance prediction. Although supervised learning requires large datasets for training, it aligns well with our intended outcome.

2

Theory

This chapter provides a brief overview of fundamental physical concepts underlying the physical model used in the CFD simulations. It also includes the performance theory of centrifugal fans, and the theoretical background of machine learning parameters, including the different archetypes of machine learning.

2.1 Computational Methods

The main governing equations relevant to the CFD simulations are the incompressible conservation of mass and the steady incompressible momentum equation:

$$\frac{\partial v_i}{\partial x_i} = 0 \tag{2.1}$$

$$\rho \frac{dv_i}{dt} = -\frac{\partial p}{\partial x_i} + \mu \frac{\partial^2 v_i}{\partial x_j \partial x_j} + \rho f_i. \tag{2.2}$$

The velocities in the governing equations can be decomposed into a mean and a fluctuating part resulting in the Reynolds Averaged Navier-Stokes (RANS) equation. [7]

2.1.1 CFD & FVM

Computational Fluid Dynamics (CFD) is a numerical approach for analyzing fluid flows. In this project, the Finite Volume Method (FVM) is used to solve the governing equations. The domain is divided into control volumes, and the equations are integrated over each volume. Using Gauss's divergence theorem, volume integrals are converted into surface integrals. Applying this to the continuity and momentum equations, Equations 2.1 and 2.2, gives their integral forms as:

$$\oint_S v_i n_i \, dA = 0 \tag{2.3}$$

$$\frac{d}{dt} \int_V \rho v_i \, dV + \oint_S \rho v_i v_j n_j \, dA = -\oint_S p n_i \, dA + \oint_S \mu \frac{\partial v_i}{\partial x_j} n_j \, dA + \int_V \rho f_i \, dV. \tag{2.4}$$

The equations are discretized using finite control volumes, allowing them to be solved. In FVM, variables like pressure and velocity are stored at cell centers, while surface integrals require face values. Interpolation schemes are then used to estimate these values. In this project, a second-order upwind scheme is used, where the face value is approximated using upstream neighboring cells.

2.1.2 MRF - Moving Reference Frame

The Moving Reference Frame (MRF) method allows steady-state simulation of rotating boundaries like fans by applying a rotating coordinate system to a stationary mesh.

This removes the the need to use a transient rotating mesh. The momentum equation in the rotating frame, assuming constant angular velocity, becomes:

$$\frac{\partial \vec{u}_R}{\partial t} + 2\vec{\Omega} \times \vec{u}_R + \vec{\Omega} \times (\vec{\Omega} \times \vec{r}) = -\nabla \left(\frac{p}{\rho}\right) + \nu \nabla^2 \vec{u}_R \tag{2.5}$$

The additional terms represent Coriolis and centrifugal forces, which account for the rotating frame's non-inertial effects. [8]

2.1.3 The $k - \omega$ SST Turbulence Model

The k–ω Shear Stress Transport (SST) turbulence model is a turbulence model applied when solving flows using the RANS equations. The model is able to accurately predict boundary layer behavior, flow separation, and adverse pressure gradients [9]. The SST model blends the standard $k - \omega$ formulation near walls with the $k - \varepsilon$ model in the free-stream region. This is achieved using a blending function F_1, which depends on local flow variables and wall distance. Further formulation details of the blending function and model coefficients can be found in [7].

2.2 Performance of Centrifugal Fans

A centrifugal fan consists of a rotor or impeller that operates through a combination of centrifugal force and airflow deflection. As the fan rotates, the air between the blades is propelled outward by centrifugal force. All centrifugal fans utilize this outward flow for airflow generation but the influence of blade deflection is most prominent in forward-curved (FC) blade designs. Three of the main fan blade designs, presented in Figure 2.1, are: a) backward-inclined (BI), b) radial, and c) forward-curved (FC) blade designs. The BI design is typically constructed with a straight, curved, or airfoil-shaped blade. [10]

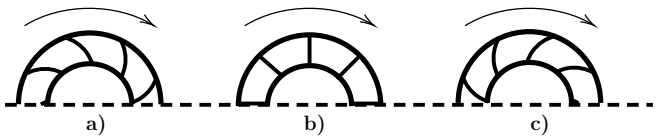


Figure 2.1: Backwards-inclined, radial and forwards-curved centrifugal blade configurations.

The Same Sky CBM-97S series DC centrifugal fan is a forward-curved (FC) fan. FC fans are known for delivering higher volume flow and higher static pressure than other blade designs at the same size and rotational speed, though this comes at the cost of operating efficiency. Their lower operating speeds make them desirable for cooling electronics and other equipment where low vibrations are preferable. To accommodate the large airflow, the ratio between the inner diameter d_1 and the outer diameter d_2 is large, typically between 0.75 and 0.9, leading to a narrow annular space for the blades. With a narrow blade space, a high number of blades between 24 and 64 is used. The lower efficiency of FC fans results from a high degree of flow redirection between the blades, which causes flow separation and turbulence. As a result, refining

the aerodynamic shape is less critical in an FC fan, with a typical blade geometry consisting of a smooth curve or circular arc. FC fans are advantageous at small scales, where lower operating speeds and high airflow are prioritized over efficiency. However, their performance degrades with increased size, where other options like BI fans are preferred. [10]

2.3 Machine Learning

This section presents the general theory behind how a machine learning (ML) model is constructed and how different parameters will affect its performance.

2.3.1 Neural Network

A neural network (NN) consists of multiple artificial neurons interconnected and organized into layers. These neurons feed data forward with each neuron receiving signals from connected neurons. The network consists of an input layer that distributes the input to the active layers, one or more hidden layers that process the information, and an output layer, as shown in Figure 2.2.

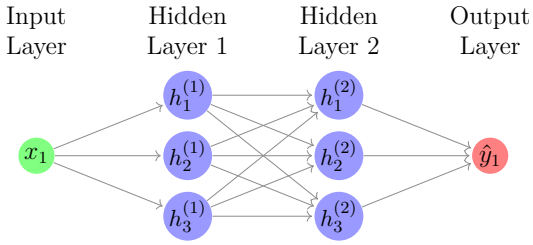


Figure 2.2: Simple neural network.

The output from a neuron j to the next is computed from the inputs x_i using a mathematical model. The output is obtained by applying an activation function f to a weighted sum of the inputs. Each x_i input is multiplied by a weight w_i and a bias term b_j is added before applying the function [11]:

$$y_j = f(z_j) = f\left(\sum_{i=1}^{n} w_{ij}x_i + b_j\right). \tag{2.6}$$

In addition to the basic feedforward neural networks, there are several specialized types of neural networks designed to handle different kinds of data and tasks. Convolutional neural networks (CNN) uses convolution operations to process the data and are often used in image analysis and recognition. Graph neural networks (GNN) has graphs as inputs and uses pairwise message passing to update the nodes iteratively.

2.3.2 Activation Function

The choice of activation functions is a key characterization of a neural network and determines how signals are transformed and propagated through the network. Activation functions can be linear, however non linear activation functions like the sigmoid function and hyperbolic functions are commonly used to enable learning for complex

patterns. However, these functions can lead to the vanishing gradient problem for inputs near zero. This problem is addressed by the Rectified Linear Unit (ReLu) function:

$$f_{ReLu} = \max(0, z) \tag{2.7}$$

ReLU outputs zero for all negative input values, which introduces non-linearity into the model, improves training efficiency, enables the network to learn more complex patterns, and helps mitigate the vanishing gradient problem during training. [11]

2.3.3 Supervised Learning

Different approaches to the architecture of ML models have different applications. Supervised learning relies on labeled datasets to train models that can predict outcomes and recognize patterns. This approach results in high accuracy and reliable models at the cost of sufficiently large and high-quality datasets.

2.3.4 Loss Function

Quantifying the difference between predicted outputs and the actual outputs of the ML model is done through a loss function. To minimize the prediction error the model utilizes the calculated loss to change the weights, which in effect reduces the prediction error. Various types of loss functions can be used and they all have different applications.

Mean Square Error (MSE) takes the average of the the square between the predicted and actual values, Equation 2.8, treating all values equal leading to fast convergence when the error is small. For large errors it is penalized since the square magnifies the impact of the error.

$$\mathcal{L}_{MSE} = \frac{1}{N}\sum_{i=1}^{N}\Big(y_i - f(x_i)\Big)^2 \tag{2.8}$$

Mean Absolute Error (MAE) is well suited for handling outliers in given datasets. It uses the absolute value of the difference between the actual and predicted data, Equation 2.9, resulting in a robust method when datasets have large outliers which would otherwise affect the loss greatly.

$$\mathcal{L}_{MAE} = \frac{1}{N}\sum_{i=1}^{N}\Big|y_i - f(x_i)\Big| \tag{2.9}$$

Additional contributions to the loss functions such as e.g. continuity loss and momentum loss, would enable the neural network to account for the physics involved in the flow. This is called a physics-informed neural network (PINN) where the continuity equations and momentum equations are directly implemented in the loss function.

2.3.5 Training Procedure

Proper scaling of input features and targets is a crucial step in neural network modeling, affecting stability and speed of the learning process. Neural networks are sensitive

to the relative scale of the inputs and the optimization process can become biased toward a feature with larger range. This bias can cause the network to under fit smaller-scale features. This is particularly important in physical modeling, where different features have different units and magnitudes.

Training starts with forward propagation where the input is added with the weights and the neuron outputs are calculated using the activation function. After the forwards propagation, back propagation is performed to update the network's hyperparameters. The aim for the back propagation is to regulate the weights to minimize the loss calculated with the loss function. During back-propagation optimization algorithms, such as gradient descent, are applied to the gradient of the loss function to update the weights towards local optimum [11].

Performing a forward and backward propagation on every sample dataset is defined as an epoch. When the number of epochs is set too high the model can become prone to overfitting. Overfitting is when the models predictions are accurate for training data, but inaccurate for new data. If the number of epochs are too low it can lead to underfitting giving inaccurate predictions.

Batching is a technique used during training aiming at reducing generalization and overfitting. Batching divides the dataset into smaller groups called batches which can include data from different datasets or simulations for diverse set of data. This can improve the models ability to generalize across varying data or simulation outcomes. A large batch enables faster and more stable training due to the efficient use of the graphics processing unit (GPU) and smoother estimation of the gradients. This however, requires more memory at the cost of the models ability to generalize. Smaller batch sizes use less memory and can improve the models ability to generalize at the cost of slower training and less stability.

The validation of the models predictions can be used to evaluate the models performance on data not included in its training. This can be used to tune the models hyperparameters e.g. learning rate, batch size, number of layers and neurons to develop the ability to generalize and reduce overfitting.

3

Methodology

3.1 Geometry Simplifications

The fan model was parameterized in the STAR-CCM+ CAD environment, where the blade geometry was simplified by changing to a straight blade design, see Figure 3.1. The parametrization included the angle of attack, number of blades and blade length. In the simulations the angle of attack and the number of blades were altered. The angle of attack ranged from $0 - 45°$ in increments of $5°$ and the number of blades were altered from $22 - 60$ in increments of 2 for a total of 200 simulations.[8]

(a) Validation impeller geometry.

(b) Simplified impeller geometry.

(c) Full simplified impeller geometry.

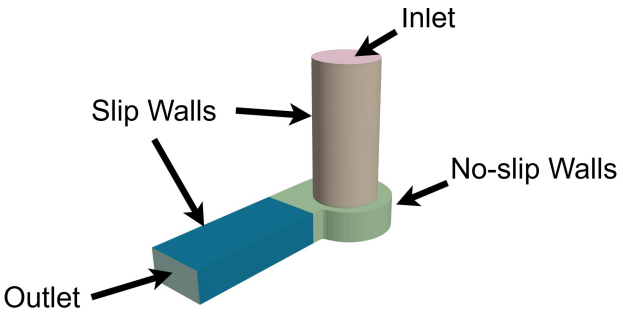
Figure 3.1: Original impeller geometry and the simplified geometry.

3.2 CFD Setup

To simulate the rotating fan, a Moving Reference Frame (MRF) was employed. The geometry was divided in to a static and a rotating region with interfaces between them. The following boundary conditions (BCs) were used.

Surface	BC
Inlet	Stagnation Inlet
Inlet Tube	Slip Wall
Casing	No-Slip Wall
Impeller	No-Slip Wall
Outlet Tube	Slip Wall
Outlet	Pressure Outlet

(a) Boundary conditions



(b) Simulation Domain

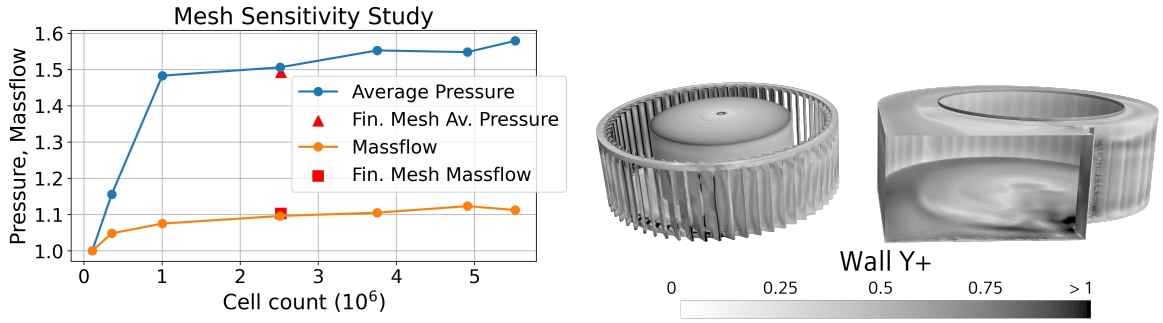
Figure 3.2: Boundary conditions and simulation domain.

To avoid over-constraining in the simulation, the rotation-rate was set in the rotating region, while the inlet and outlet pressure were set to ambient conditions. The simula-

tion model used Reynolds-averaged Navier-Stokes (RANS) and the SST-Menter $k-\omega$ turbulence model with all y^+ wall treatment and a target y^+ of below 1. The physics were set to incompressible with a constant density and a coupled solver. A second-order discretization with an implicit solver was used, as it provided a good balance between iteration time, accuracy and convergence speed. Furthermore a quadratic constitutive relation was selected to account for the anisotropic turbulence caused by the high rotational rate. This was combined with curvature and compressibility corrections, leading to a more stable and accurate solution following guidelines.[8]

3.2.1 Mesh-study

During the mesh configuration a mesh sensitivity study was performed to assess the mesh independence of the simulation results. The study was carried out using the validation geometry, which includes the original impeller and hub. Initially, mesh refinement were achieved by varying a global base size from a coarse to a fine mesh. The mass flow and average pressure results for different cell counts, with pressure measured at the fan outlet, are displayed in Figure 3.3a.



(a) Mesh study of mass flow and pressure. Red markers show final mesh values.

(b) Wall y^+ distribution on the surface of the impeller (left) and the case (right).

Figure 3.3: Mesh and near-wall resolution analysis.

Following the mesh study, the mesh was manually altered by applying different base sizes for the rotating and stationary region. To improve mesh resolution near the blades a finer base size was used in the rotating region, as it was important to maintain sufficient mesh quality when increasing the blade count. To reduce the overall cell count and maintain computational efficiency, a coarser mesh was used in the non-rotating regions where high resolution was less critical. Attention was given to achieving a near-wall resolution of $y^+ < 1$, in order to accurately capture boundary layer effects. The y^+ distribution is shown in Figure 3.3b where the majority of cells exhibit a value less than 1. Given the limited computational resources available for the project, minimizing simulation costs was prioritized. The mesh settings were kept constant for all parametric geometries, but the total cell count varied with blade count, as more blades required finer meshing around additional surfaces, resulting in a higher cell count.

3.2.2 Mesh Setup

The volume mesh was constructed with polyhedral cells, with custom controls applied around the blades and no-slip walls to accurately capture the boundary layer. The final mesh settings are displayed in Tables 3.1 and 3.2. The total cell count for the modified geometry varied between 2.5 million and 3.2 million cells, depending on impeller geometry. The stationary and the rotating regions used different values for base size, prism layer configuration, and minimum surface size. The global mesh settings for both regions are summarized in Table 3.1.

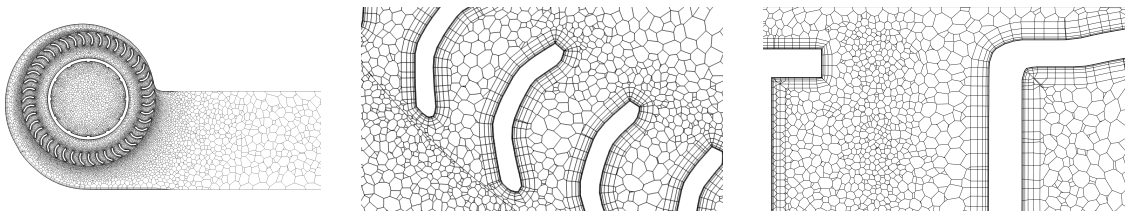
Table 3.1: Global mesh settings for stationary and rotating regions.

Parameter	**Stationary Region**	**Rotating Region**
Base size	4.25 mm	3.75 mm
Minimum surface size	15% of base	5% of base
Prism layers	5 layers	6 layers
Prism layer thickness ratio	20.0	16.0
Total prism layer thickness	1.2 mm	1.1 mm

Table 3.2: Local Mesh Refinements

Region	**Boundary**	**# of Prism Layers**	**Prism Layer Thickness Ratio**	**Target Surface Size**	**Total Prism Thickness**
Stationary	Cylinder top/bottom	Disabled	–	1.0 mm	–
	Cylinder wall	Disabled	–	0.25 mm	–
	Inlet/outlet	Disabled	–	8.0 mm	–
	Case sidewalls	Global	23	90% base	1.3 mm
Rotating	Blades	Global	16	50% base	0.65 mm
	Cylinder	Disabled	–	1.0 mm	–
	Hub	Global	Global	80% base	Global

Local mesh refinements for specific boundaries within each region, including prism layer settings and surface sizing, are detailed in Table 3.2. In order to create smooth transitions between the boundaries, the minimum surfaces sizes were also changed. The final volume mesh for the validation can be observed in Figure 3.4.



(a) Volume mesh on the XY-Plane.

(b) Close-up the XY-Plane.

(c) Close-up on the YZ-Plane.

Figure 3.4: Full volume mesh for the validation geometry displayed on the XY-plane, with a zoomed view on the blades on the XY and YZ-plane sections.

3.2.3 Validation of Physics Model

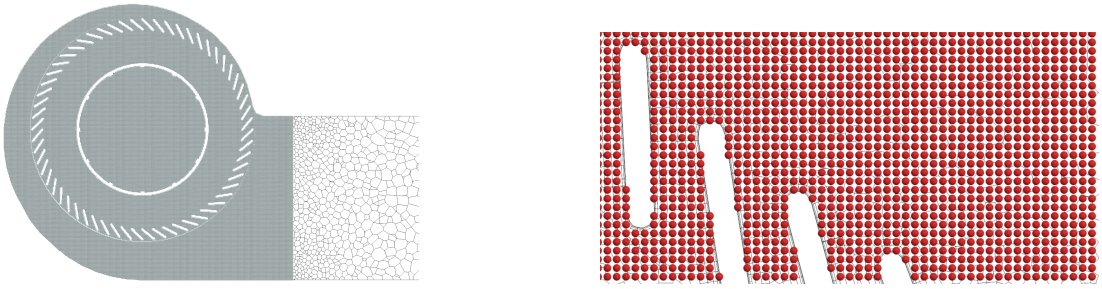
The fan was validated by comparison to the provided fan performance curve from the Sky CBM-97S series DC fan. The curve measures the relation between mass flow and static pressure for a fan. The experimental fan curve was normalized and then compared to normalized data from CFD simulations. The simulations were set up with a prescribed mass flow inlet, a pressure outlet and a rotational rate of 2500 RPM's. By varying the inlet mass flow from $0 - 0.9$ g/s, and measuring the pressure at the fan outlet, a similar curve to the experimental data was created. The results for the fan curve comparisons are presented in the Results, Chapter 4.1.

3.3 Surrogate Model for Pressure- & Velocity Fields

One objectives of this study is to predict pressure and velocity fields in a 2D fluid domain using a neural network that generalizes across various fan blade geometries. The neural network was implemented in Python, using PyTorch and scikit-learn libraries. The full codes are available in the *Code Repository*.

3.3.1 Data Preprocessing

Data was extracted from the STAR-CCM+ simulations on a fixed 350×350 grid of equidistant probes. Using the same grid for all simulations ensures that the model always sees data from consistent spatial locations, even when the domain changes with different impeller geometries. The grid used for sampling is illustrated in Figures 3.5a and 3.5b, magnified for visualization. The raw data contained only the nodes within the fluid domain for a given 2D-slice, each including three spatial coordinates and four field values; pressure and three velocity components. Figure 3.6 shows histograms of velocity magnitude and pressure across all simulations and for a single case, illustrating the data distribution.



(a) Grid used for sampling. **(b)** Close up of sampling grid

Figure 3.5: Grid used for samlpling data from the CFD simulation

To address the problem with missing solid nodes, a file containing all possible nodes was merged with simulation data, introducing an `is_fluid` mask that marks fluid nodes with `1` and solid nodes with `0`. This masking allows downstream processes to distinguish fluid from solid nodes. As solid nodes lack pressure and velocity values, they present a challenge for machine learning, particularly because their locations vary

across simulations due to differences in blade design.

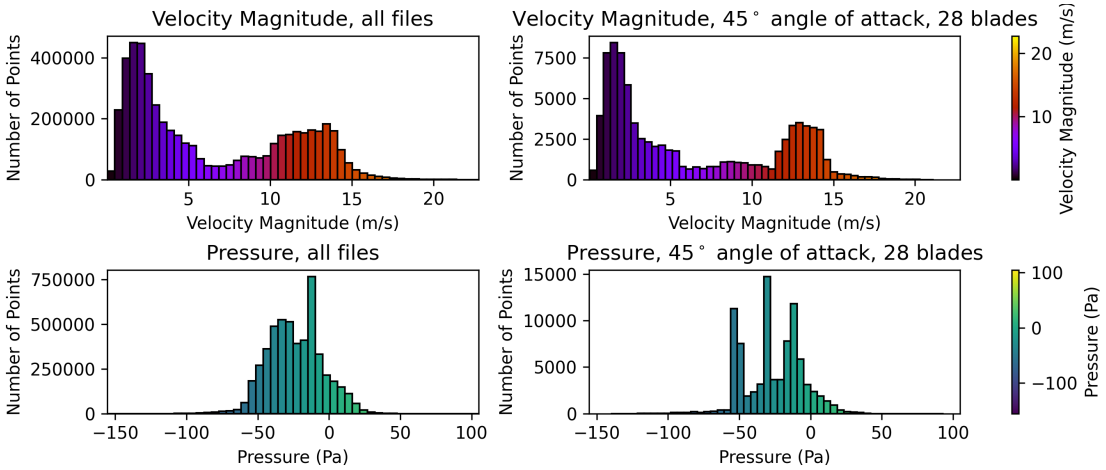


Figure 3.6: Histogram for the velocity magnitude and pressure for all nodes and for a single simulation file, with 45° angle of attack and 28 blades.

A further complication is that extreme pressure and velocity values typically occur near the fluid and solid interface. If the missing solid values were set to zero, sharp gradients would occur at the interface, making it even more difficult for the neural network to learn effectively. To mitigate this, data was interpolated into the solid regions, creating a smooth transition between fluid and solid. An interpolation example is shown in Figure 3.7.

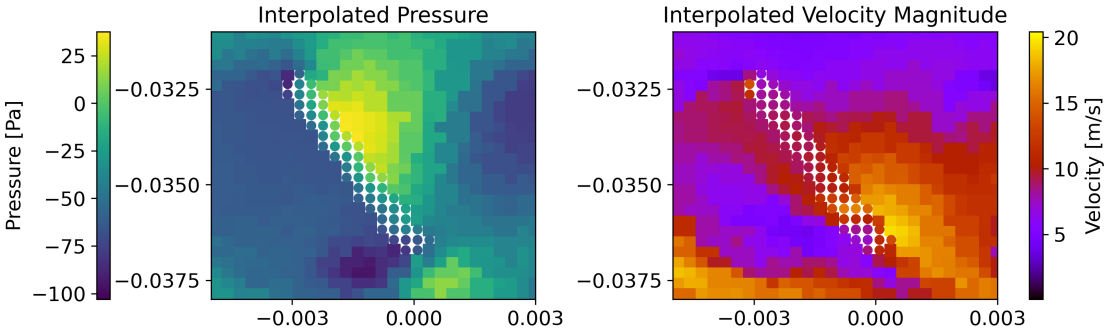


Figure 3.7: The interpolated values for pressure and velocity magnitude for solid is shown. The interpolated nodal values are marked with circles.

In addition to this, the three simulation parameters were extracted from file names and added as columns. Each dataset row represents a node with seven input features (coordinates, simulation parameters and `is_fluid`) and four target features (pressure and three velocity components).

Proper scaling of input features and targets is a crucial step in neural network modeling. To address this, `StandardScaler` from the `scikit-learn` library were fitted for all variables. Each column is scaled independently of the others resulting in columns with a zero mean and a standard deviation of one. Each scalar was fitted across all

training data to ensure that the transformation reflected the full range.

Lastly, a total of 200 CFD simulation files were conducted with varying geometries. Out of these, five cases were randomly selected and set aside as true validation cases, ensuring they remained unseen by the neural network during training, making them suitable for final evaluation of the model's performance. The remaining 195 files were used to train the model, with the data split into two sets: 80% allocated for training the model, and 20% reserved for internal validation. This split allows the model to learn from most of the data, while the validation set is used to tune hyperparameters and assess performance during training, helping improve generalization and robustness.

3.3.2 Data Loading & Batching

To construct effective batches for neural network training, files were preprocessed in parallel, stacked, and nodes were then randomly sampled. This ensured that each batch contained a diverse set of geometries, parameter values, and field distributions. Since each node included spatial coordinates, the neural network learned how the target values related to spatial position, making it possible to mix nodes from different simulation files. It also prevented the model from memorizing simulation specific patterns, thereby reducing overfitting.

Stacking hundreds of simulation files, each containing about 100000 nodes, resulted in large memory demands. On systems with sufficient memory, such as a computer cluster, stacking all simulation at once was possible. However, if the training were to be performed on a machine with limited system memory, an option to load and process the files in smaller chunks was implemented. This chunking approach would allow any number of simulations to be handled, with batches drawn from the currently loaded subset before proceeding to the next chunk. To further reduce overfitting, the files included in each chunk is randomized for every epoch. However, this approach would be less efficient and slower, as the data would need to be preprocessed, scaled, and loaded into memory in each epoch.

3.3.3 Model Architecture

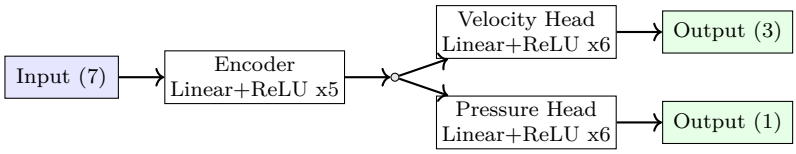


Figure 3.8: Architecture of the neural network model used for prediction of pressure- and velocity fields.

The neural network was designed to jointly predict both pressure and velocity fields from the input features at each node. To achieve this, a multi-head architecture was implemented, consisting of a shared encoder followed by two specialized output heads, as seen in Figure 3.8. The encoder comprises fully connected layers with ReLU activations, chosen for its computational efficiency, non-linearity and ability to mitigating

vanishing gradients during training. After the encoder, the network branched into two separate heads, one which predicted the scalar pressure and the other that predicted the three velocity components. Although pressure and velocity are related through the governing equations of fluid dynamics, they represent different physical properties with different complexities. By the shared encoder, the model learned features relevant to both properties, improving the coupling between pressure and velocity, while the separate heads allowed for a more task specific refinement.

3.3.4 Loss Function & Optimization

The neural network was trained using a loss function defined as the sum of MSE for both pressure and velocity. Empirical tests showed that MSE consistently provided accurate and stable results compared to other loss functions. The loss was computed over all nodes, including both fluid and solid cells. Although solid cells lack physical meaning, including them in the loss encouraged smooth and consistent predictions throughout the domain, helping the model avoid sharp discontinuities at fluid-solid interfaces and improving generalization.

Model optimization used the AdamW optimizer with a weight decay of 1×10^{-4} and an initial learning rate of 1×10^{-3}. AdamW provides adaptive learning rates and efficient convergence, with a decoupled weight decay mechanism. Weight decay acts as a form of regularization by adding a penalty to large parameter values, thereby helping to prevent overfitting and encouraging the model to learn simpler, more generalizable patterns. To further improve training stability, a `ReduceLROnPlateau` scheduler was used to automatically lower the learning rate when validation loss plateaued, helping the model fine-tune its parameters and escape local minima. A batch size of 1024 was chosen because it was computationally efficient and resulted in good model performance. A suitable number of epochs was determined by monitoring training and validation losses, ensuring convergence without overfitting.

3.3.5 Validation

As mentioned earlier, five of the 200 CFD simulation files were randomly selected and set aside as true validation cases. The pre-trained model was then used to predict the flow fields for these validation cases, and the results was compared to the reference CFD data for evaluation. Performance is assessed using MSE, MAE and coefficient of determination (R^2) on the normalized data, calculated for pressure, velocity, and total error. R^2 quantifies the proportion of variance in the observed data that the model can predict, indicating how well the model fits the data, with 1 signifying a perfect prediction. All metrics and loss functions are evaluated only on fluid nodes, since predictions on solid nodes have no physical meaning.

3.4 Surrogate Model for Fan Performance

The project involves the construction of a neural network model for predicting key flow quantities namely, *static pressure* and *mass flow rate* based on fan blade geometry parameters. The network is designed to generalize across a wide variety of geometrical configurations. It was implemented in `PyTorch`, trained using simulation data from

STAR-CCM+, and validated on unseen designs. The complete codebase can be found in the *Code Repository*.

3.4.1 Data Preprocessing

The dataset used for training and validation contained simulation data for various 2D fan blade configurations. The input features included angle of attack, number of blades and blade length, while the target variables were average static pressure and mass flow. Each input and output variable was independently scaled using the `MinMaxScaler` from `scikit-learn`, ensuring values between 0 and 1. Separate scalers were fitted for the input vector, pressure, and mass flow to allow for consistent inverse transformation during prediction. A random split (80% training, 20% validation) was used to prepare the dataset.

3.4.2 Model Architecture

As previously described, the neural network was developed to predict static pressure and mass flow rate based on the fan blade geometry. To achieve this, a multi-head architecture was employed, comprising a shared encoder and two distinct output branches, as seen in Figure 3.9.

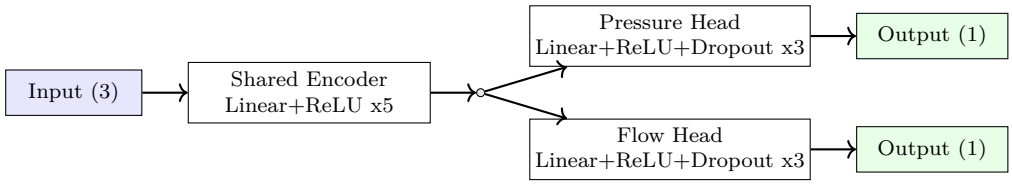


Figure 3.9: Architecture of the neural network model used for fan performance.

A shared encoder, composed of fully connected layers with `ReLU` activations and dropout, was used to improve training stability and prevent overfitting. This encoder transforms the three input geometry parameters into an internal representation. From this internal representation, two separate output heads are used: one for static pressure and one for mass flow rate. This design allows the model to share common features while specializing each head for its respective target variable.

3.4.3 Loss Function & Optimization

To balance learning between mass flow and pressure, a weighted combination of their MAE losses was applied, assigning 60% weight to pressure and 40% to mass flow. MAE measures the average absolute difference between predicted and actual values, making it robust to outliers and easy to interpret. Pressure is given higher weight due to its greater sensitivity to geometry changes, helping the model focus on capturing pressure-related behavior more accurately. The model is optimized using the Adam optimizer with weight decay for regularization. A learning rate scheduler (`ReduceLROnPlateau`) is applied to reduce the learning rate when validation loss stops improving, aiding smooth convergence.

3.4.4 Validation

Five CFD simulation cases were set aside as true validation and were not used during training of the NN. The trained model was then used to predict static pressure and mass flow rate for these unseen geometries, and the predictions were compared with the corresponding CFD results. Performance was evaluated on the normalized data using MSE, MAE, and the prediction error in percentage, calculated separately for static pressure and mass flow rate.

4

Results

4.1 CFD Simulation

A comparison of pressure-mass flow curves between provided experimental data and data from CFD simulations is presented in Figure 4.1. The pressure is normalized against its own value at 0 mass flow. The pressure was measured as an average of the inlet pressure. The experimental and simulated curves are not equal but they follow similar trends: static pressure decreases with the mass flow rate. Noticeably the simulations do not reach a pressure of 0 at the mass flow where the experimental data does.

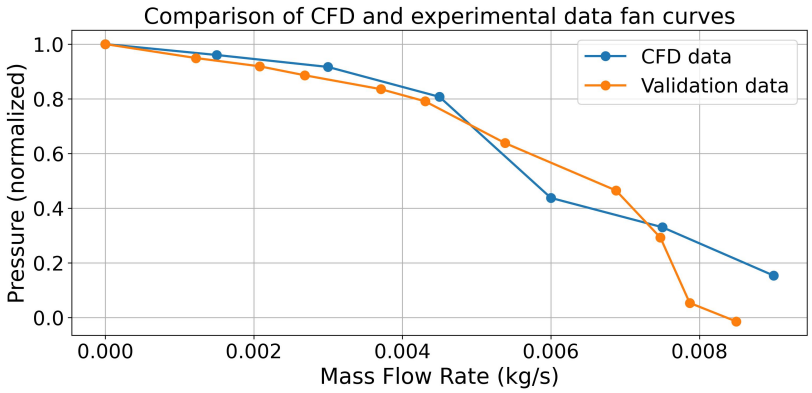


Figure 4.1: Comparison of pressure-mass flow fan curves.

4.2 Surrogate Model for Pressure- & Velocity Fields

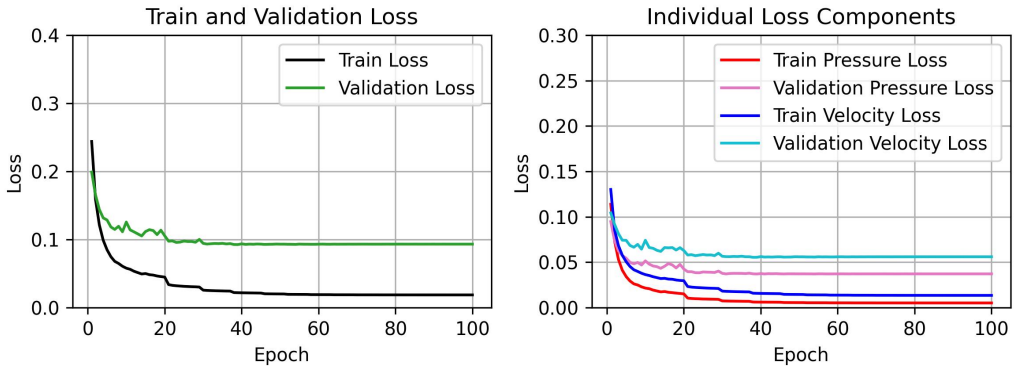


Figure 4.2: The loss over epochs for training of neural network for pressure and velocity fields.

The training and validation loss curves in Figure 4.2 shows that the model converges steadily, with both losses decreasing and stabilizing as training progresses. The close

alignment between training and validation loss curves suggests that the model generalizes well to unseen data and does not suffer from overfitting. Upon inspection of the individual loss curves, the pressure loss is consistently lower than the velocity loss, showing that the model predicts pressure more accurately than velocity. It can also be seen that the lowest validation loss was found at epoch 39 with 0.021 for training loss and 0.093 for validation loss, hence this model was used. After this point, the validation loss stabilized while the training loss continued to decrease, indicating that further training would likely lead to overfitting and reduced generalization.

Table 4.1: Validation performance of the neural network on unseen simulations. The table shows MSE, MAE, and coefficient of determination (R^2) for pressure, velocity, and total predictions.

Angle of attack	Number of blades	MSE			MAE			R^2	
		Tot	P	V	Tot	P	V	P	V
45.0°	28	0.095	0.046	0.050	0.244	0.124	0.120	0.96	0.94
55.0°	30	0.066	0.036	0.030	0.227	0.130	0.097	0.95	0.96
60.0°	26	0.058	0.022	0.036	0.189	0.084	0.105	0.97	0.96
65.0°	38	0.071	0.027	0.044	0.188	0.086	0.102	0.97	0.95
85.0°	54	0.057	0.017	0.040	0.193	0.085	0.108	0.98	0.96

Table 4.1 presents the results for five different validation cases that is unseen by the model, each with varying angles of attack and blade numbers. The lowest total loss is observed for the case with a 60° angle of attack and 26 blades, while the highest total loss occurs for the case with a 45° angle and 28 blades. Similar to the losses during training, the pressure losses is less than the velocity losses. Furthermore, the MSE values for these validation files are very similar to the validation losses observed during neural network training. This close correspondence indicates that the model performs consistently on new, unseen data, demonstrating good generalization ability.

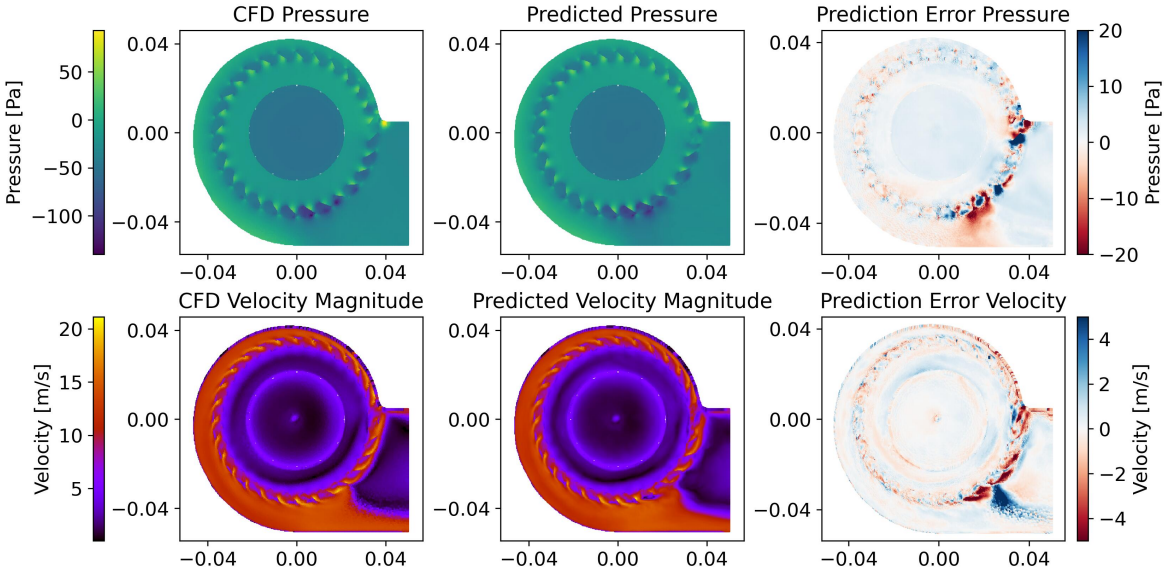


Figure 4.3: Worst prediction, 45° angle of attack and 28 blades.

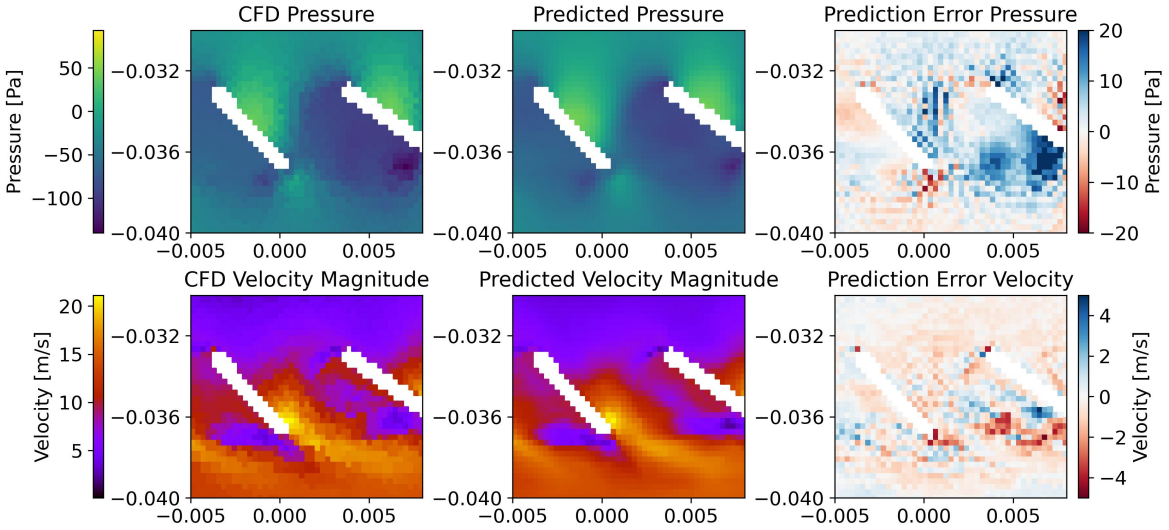


Figure 4.4: Zoom in for worst prediction, 45° angle of attack and 28 blades.

Figures 4.3 and 4.4 display the worst-case prediction for the 45° angle of attack and 28 blades. The model captures the overall structure of the pressure and velocity fields, but larger discrepancies are visible near the fluid and solid interface and in regions with high velocity gradients. The zoomed-in view highlights these areas, where the largest errors occur close to the blade surfaces and in the region close to the outlet. In contrast, Figures 4.5 and 4.6 show the best-case prediction for the 60° angle of attack and 26 blades. Here, the predicted fields closely match the reference simulation, but similar but smaller discrepancies are visible similar as the worst case.

Even in the worst-case scenario, the model's predictions remain remarkably accurate. The overall structure and main flow features are well captured, and the quantitative difference between the worst and best cases is relatively small. This shows that the model is robust across different blade geometries, consistently delivering high quality predictions even in more challenging cases.

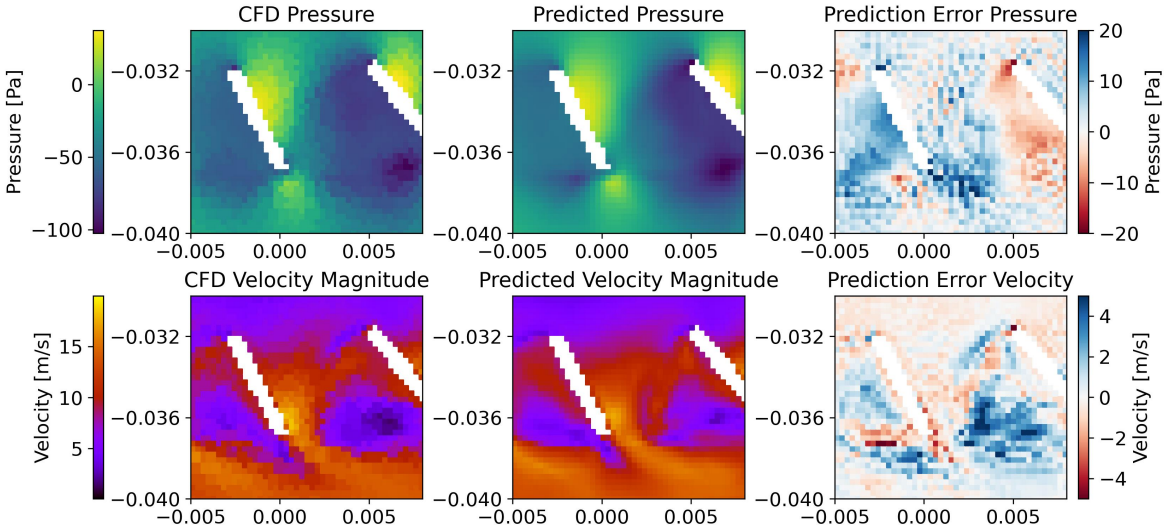


Figure 4.6: Zoom in for best prediction, 60° angle of attack and 26 blades.

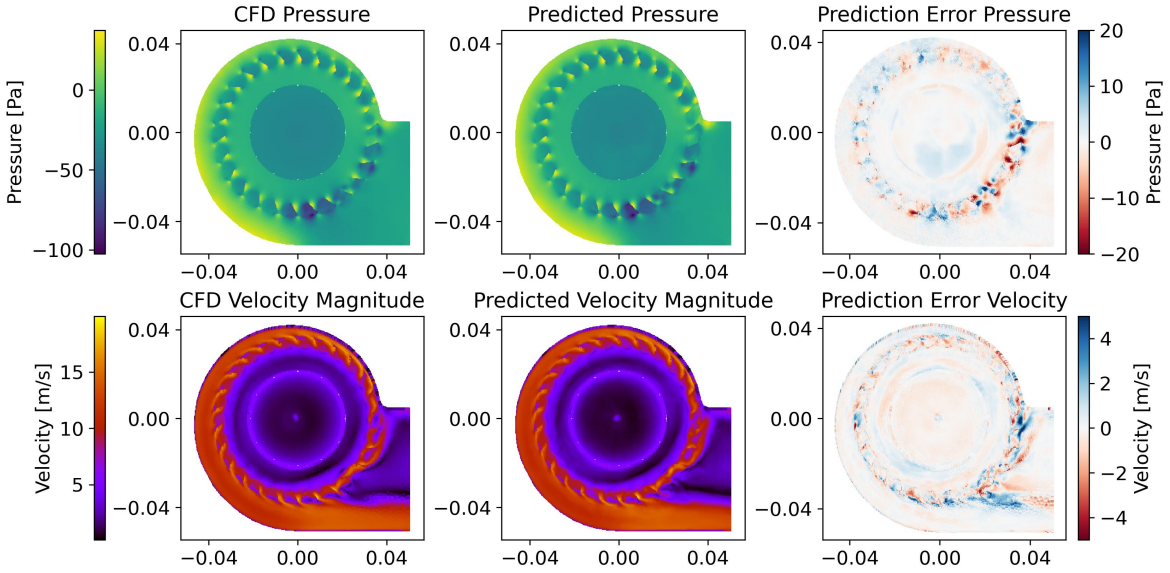


Figure 4.5: Best prediction, 60° angle of attack and 26 blades.

4.3 Surrogate Model for Fan Performance

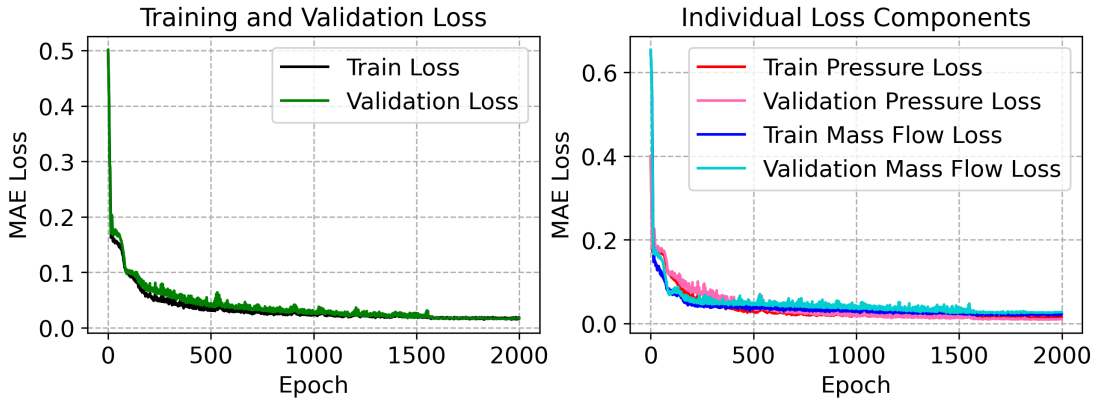


Figure 4.7: The loss over epochs for training of neural network for performance.

The neural network model is trained to predict static pressure and mass flow rate using the geometric input parameters: angle of attack (AoA), number of blades (nBlades) and blade length (BladeL). Figure 4.7 show that the total training and validation losses steadily decreases over 2000 epochs, with minimal difference between them, indicating strong generalization. The individual loss components reveal that pressure and mass flow predictions converge smoothly, with slight variations.

Figures 4.8 and 4.9 presents a comparison between the NN model predictions and CFD simulation data for mass flow rate and static pressure, respectively. For both quantities, the predicted values closely match the CFD results, demonstrating the model's ability to capture key performance trends. Although minor deviations are present in some cases, the model overall good agreement and reliably estimates fan performance across varying input parameters. Additionally, Figure 4.8 indicates that

mass flow rate generally increases with the number of blades and for angles of attack between 60° to 80°.

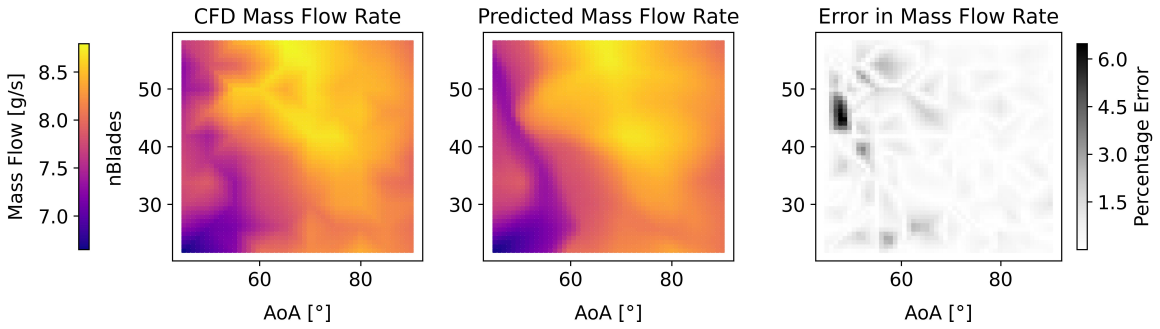


Figure 4.8: Comparison of predicted mass flow rates across the parameter space defined by number of blades and angle of attack.

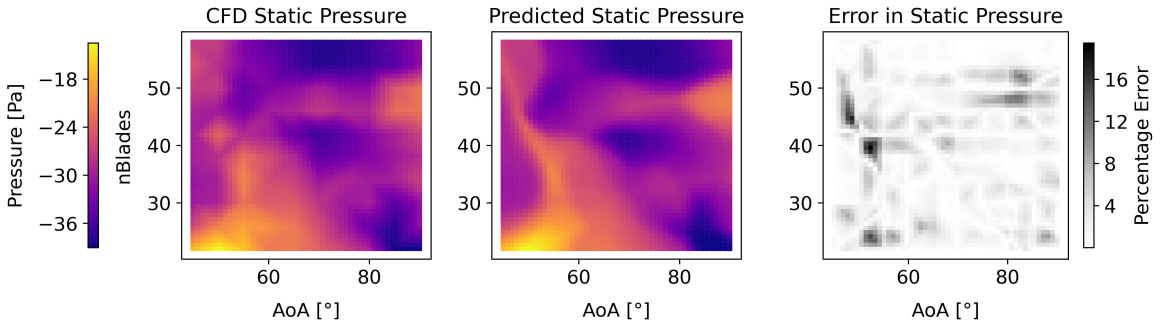


Figure 4.9: Comparison of predicted static pressure rates across the parameter space defined by number of blades and angle of attack.

To evaluate the performance of the trained neural network, predictions of five unseen validation cases are summarized in Table 4.2, which shows the prediction errors for static pressure and mass flow rate. These geometries were not included in the training set and were selected to test the model's prediction capability. The model maintains low error levels across all cases, with most static pressure predictions within 6% and mass flow rate errors remaining well below 2%. Overall, the model demonstrates strong prediction capability and reliable performance on new input combinations.

Table 4.2: Validation performance of the neural network on unseen simulations.

Angle of attack	**Number of blades**	**MAE**		**MSE**		**% Error**	
		P	**M**	**P**	**M**	**P**	**M**
45.0°	28	0.029	0.038	0.0008	0.0014	2.48	1.08
55.0°	30	0.002	0.003	4.00×10^{-6}	1.40×10^{-5}	0.23	0.10
60.0°	26	0.126	0.143	0.0160	0.0207	5.88	4.11
65.0°	38	0.074	0.068	0.0055	0.0046	4.09	1.82
85.0°	54	0.008	0.062	6.00×10^{-5}	0.0039	0.83	1.59

5

Discussion

5.1 CFD Simulation Accuracy

The base of the project relies on accurate CFD simulations. The choice of mesh was motivated by the mesh independence study in Chapter 3.2.1 where a mesh on the coarser side was chosen to keep computational costs low. It was determined that the final mesh offered a reasonable trade-off between cell count and numerical accuracy, producing results that were sufficiently close to the finer meshes.

The accuracy of the physics model was determined by validating the simulation results with a physical test setup. Due to uncertainties in the experimental setup and the inability to fully recreate its physics in the numerical simulations, the values were normalized. This shifts the focus from validating based on absolute values to comparing the trend for operation under comparable conditions. The validation provided in Figure 4.1 shows that the pressure-mass flow trend for the simulations agree with the provided experimental data. The agreement fits better for lower mass flows; however, the conclusion is that the physical model is sufficiently accurate. When altering the geometry for the final simulation, it was assumed that the new geometry could be simulated using the same physics model while maintaining physical accuracy. To further ensure that this is the case a physical model would have to be built. In further studies with greater computational resources, validation could be carried out using transient simulations to assess whether time-based fluctuations and behaviors affect the results.

5.1.1 Simplifications of the Geometry

FC or forward-inclined fans are typically only designed with straight blades [10]. Therefore, the use of straight blades in this fan model represents a geometric simplification that may limit direct comparison with existing fan designs. While the simplification may impact the performance of the simulated fan compared to conventional designs, the performance trends should still reflect those of a model with curved blades. Further applications of the research methods used in this project could be to extended to a curved configuration, enhancing the functionality of the model for comparison with industry-standard designs.

5.2 Surrogate Model for Pressure- & Velocity Fields

When constructing a surrogate model for pressure and velocity fields, the quality and structure of input data are critical, as consistent and meaningful inputs directly determine the model's predictive capability. A central challenge was handling varying blade geometries across simulations, which introduced inconsistencies in the spatial representation of the fluid domain. Ideally, the NN would infer blade positions solely from

geometric parameters and predict flow fields without explicit spatial guidance. However, this proved infeasible due to the coarse mesh, which lacked sufficient geometric detail for the NN to reliably learn spatial relationships between blade configurations and flow patterns. To address this, explicit spatial information was provided via a mask indicating fluid or solid nodes. This significantly improved the model's ability to correlate blade locations with flow features, compensating for the limited mesh resolution. While refining the mesh or using more complex architectures could enhance geometric learning, these options were computationally prohibitive with available resources.

Additionally, since all nodes were included as NN inputs, special consideration was needed for solid nodes, which varied with geometry and could not be ignored. Assigning NaN or zero values to these nodes quickly reduced the model's ability to generalize. Interpolating values for solid nodes improved this, but it is important to note that the quality of interpolation directly affects the neural network's training and accuracy-poor interpolation can introduce additional errors.

Depending on the stage of development, the model was trained and evaluated using both smaller subsets of simulation files and the full dataset. Early in the development process, smaller subsets were used to rapidly prototype and compare different network architectures, which allowed for more efficient experimentation given the resource constraints. As the development progressed, the most promising configurations were further refined and thoroughly tested using the complete set of simulation files to ensure robust performance. Due to these resource limitations, more complex neural network architectures such as CNN and GNN were not pursued. Instead, a variety of simple feedforward network architectures were systematically explored. Similarly, the choice of loss function significantly impacts model behavior, for example, minimizing MSE does not always yield the same results as minimizing MAE. While the current NN shows strong results, alternative architectures, activation functions, or loss functions, especially with more data, could potentially lead to even better performance.

The current surrogate model demonstrates strong predictive performance for pressure and velocity fields across most of the domain. However, the largest errors are typically observed near boundaries, particularly around the blade surfaces and interfaces. This can be attributed to the complex physics occurring within boundary layers, where rapid changes in flow properties take place over very short distances. Given the fixed and relatively coarse grid, the placement of nodes in relation to the boundary layer can vary significantly between simulations, sometimes a node may fall within the boundary layer, sometimes just outside, or even inside the solid region. This variability makes it challenging for the neural network to learn consistent patterns, resulting in higher prediction errors in these regions. Despite this, the model performs well overall, but further improvements in sampling grid resolution or specialized treatment of boundary regions could help reduce these localized errors.

Several promising directions exist for future work. Extending the model to a three-dimensional domain would enable the use of physics-informed neural networks (PINN), where the momentum and continuity equations can be directly implemented in the loss

function during training, further coupling pressure and velocity. Additionally, convolutional neural networks (CNN) and graph neural networks (GNN) as mentioned in Chapter 2.3.1 could be explored to better capture spatial relationships and resolve complex boundaries, potentially reducing the need for explicit masking. Another important improvement would be to include the third input parameter, blade length, in the training process, something that was not done in the current study due to the limited resources. Furthermore, training a classifier NN to automatically identify fluid and solid regions from raw CFD data could streamline preprocessing and improve generalizability for new geometries. While the explicit `is_fluid` mask remains a practical solution for the current model, these future developments could significantly enhance both accuracy and flexibility.

5.3 Surrogate Model for Fan Performance

The NN model demonstrated good predictive accuracy, with pressure errors mostly under 10% and mass flow rate errors typically below 5%. This performance suggests that the selected input parameters, i.e., angle of attack, number of blades and blade length carry sufficient information to estimate the performance. However, the ranges of these input parameters can be adjusted. The surrogate model results discussed in Section 4.3 show that mass flow rate increases with a higher number of blades. This suggests that extending the dataset, to include more blades beyond the current range, could further improve the model's ability to explore designs targeting higher mass flow rates. The results were broadly in line with expectations, although slightly higher pressure errors were observed in some regions with relatively high static pressure. These deviations may be due to local nonlinear effects or data sparsity in certain regions of the design parameters. This sparsity could be a consequence of the random split used for training and validation, which may led to uneven training of certain geometrical combination.

The neural network used in this project is a deeper architecture with multiple hidden layers and separate output branches for pressure and flow prediction. This multi-head design contributed to more stable training and better output by allowing the model to specialize in learning each target. The use of an MAE-based loss function, with increased weight on pressure, likely helped the model prioritize learning where prediction errors were more critical. Although this architecture is more computationally intensive to train, it offers greater flexibility and capacity to learn complex nonlinear relationships. This makes the model suitable for applications where high accuracy is required across a wide range of design inputs, and potentially improves performance when extended to higher-dimensional or more realistic datasets.

In this study, the model was trained using data generated from simplified, straight blade geometries operating at a fixed rotational speed. As a result, it does not currently account for the influence of blade curvature, variable RPM, or more complex flow conditions. To enhance the model's generalization capability, future work could focus on generating datasets with curved blades, varying operating conditions, and more diverse geometrical parameters. Additionally, incorporating physics based constraints or hybrid modeling techniques could further improve prediction.

6
Conclusion

The provided fan geometry was simulated in STAR-CCM+ using RANS and MRF. A pressure-mass flow fan curve was obtained from simulations, in which the mass flows rates were varied, was compared to experimental data. From the comparison it was concluded that the CFD model represents the physical system with sufficient accuracy. The geometry for the fan impeller was simplified, with straight blades, and parametrized allowing quick alteration of number of blades, angle of attack and blade length between simulations. Several simulations with varying number of blades and blade angle of attacks were preformed to collect data for the neural network training.

One of the main objectives of this study was to develop a surrogate model capable of fast and accurate predictions of pressure- and velocity fields in a 2D fan domain, reducing the need for time-consuming CFD simulations and allowing faster design changes. The results demonstrate that the neural network model generalizes well across a variety of blade geometries, consistently achieving high accuracy on unseen validation cases, as indicated by low MSE and MAE values and R^2 scores. The analysis shows that the model is able to capture essential features of the pressure- and velocity fields, with the largest discrepancies located in regions near blades and areas with steep gradients. Even in the worst-case scenarios, the surrogate model maintains robust performance, demonstrating consistent generalization and reliability. These findings confirm that the approach is well suited for fast performance evaluation in fan design. Although the present model is not perfect, as well as restricted to 2D domains and a fixed set of geometric parameters, the methodology provides a solid foundation for future work.

A second neural network model was developed to predict the performance of centrifugal fans using basic geometric parameters. The model was trained on CFD results and was able to accurately predict both static pressure and mass flow rate with minimal errors. A shared encoder and task-specific heads for pressure and mass flow rate enabled efficient multi-output learning. The weighted loss function was designed to give more importance to sensitive metrics, such as pressure in our case, which helped improve the prediction accuracy. Overall, it presents a fast and reliable alternative to CFD for design evaluation and lays the foundation for future optimization tasks.

Bibliography

[1] V. Kadambi and M. Prasad, *Turbomachinery*, 3rd. New Academic Science, 2015, ISBN: 9781781830826.

[2] S. L. Dixon and C. A. Hall, *Fluid Mechanics and Thermodynamics of Turbomachinery*, 7th. Butterworth-Heinemann, 2014, ISBN: 9780124159549.

[3] F. Meng, L. Wang, W. Ming, and H. Zhang, "Aerodynamics optimization of multi-blade centrifugal fan based on extreme learning machine surrogate model and particle swarm optimization algorithm," *Metals*, vol. 13, no. 7, p. 1222, 2023. DOI: 10.3390/met13071222.

[4] Same Sky Devices, *Cbm-97s series dc centrifugal blower*, https://www.sameskydevices.com/product/thermal-management/dc-fans/centrifugal-blowers/cbm-979433s-125-467, 2025.

[5] *Vera*, Accessed: 2025-04-27, Chalmers University of Technology, Chalmers e-Commons e-Infrastructure group. [Online]. Available: https://www.c3se.chalmers.se/about/Vera/.

[6] *About us*, Accessed 2025-04-27, National Academic Infrastructure for Supercomputing in Sweden. [Online]. Available: https://www.naiss.se/about-us/.

[7] L. Davidson, *Fluid mechanics, turbulent flow and turbulence modeling*. Gothenburg, Sweden: Chalmers University of Technology, 2025. [Online]. Available: https://www.tfd.chalmers.se/~lada/.

[8] Siemens Digital Industries Software, *Simcenter star-ccm+ documentation*, Accessed: 2025-04-16, Siemens Digital Industries Software, 2025. [Online]. Available: https://docs.sw.siemens.com/documentation/external/PL20200805113346338/en-US/userManual/userguide/html/index.html#page/connect%2Fsplash.html.

[9] F. R. Menter, M. Kuntz, and R. Langtry, "Ten years of industrial experience with the sst turbulence model," *Heat and Mass Transfer*, 2003, Available at https://www.researchgate.net/publication/228742295.

[10] F. P. Bleier, *Fan Handbook: Selection, Application, and Design*. McGraw-Hill, 1998, ISBN: 0-07-005933-0.

[11] C. Forssén and A. Ekström, *Bayesian inference and machine learning*, Accessed: 2025-04-27, 2023. [Online]. Available: %5Curl%7Bhttps://cforssen.gitlab.io/tif385-book/content/MachineLearning/NeuralNet.html#learning-algorithm%7D.

Code Repository

https://github.com/