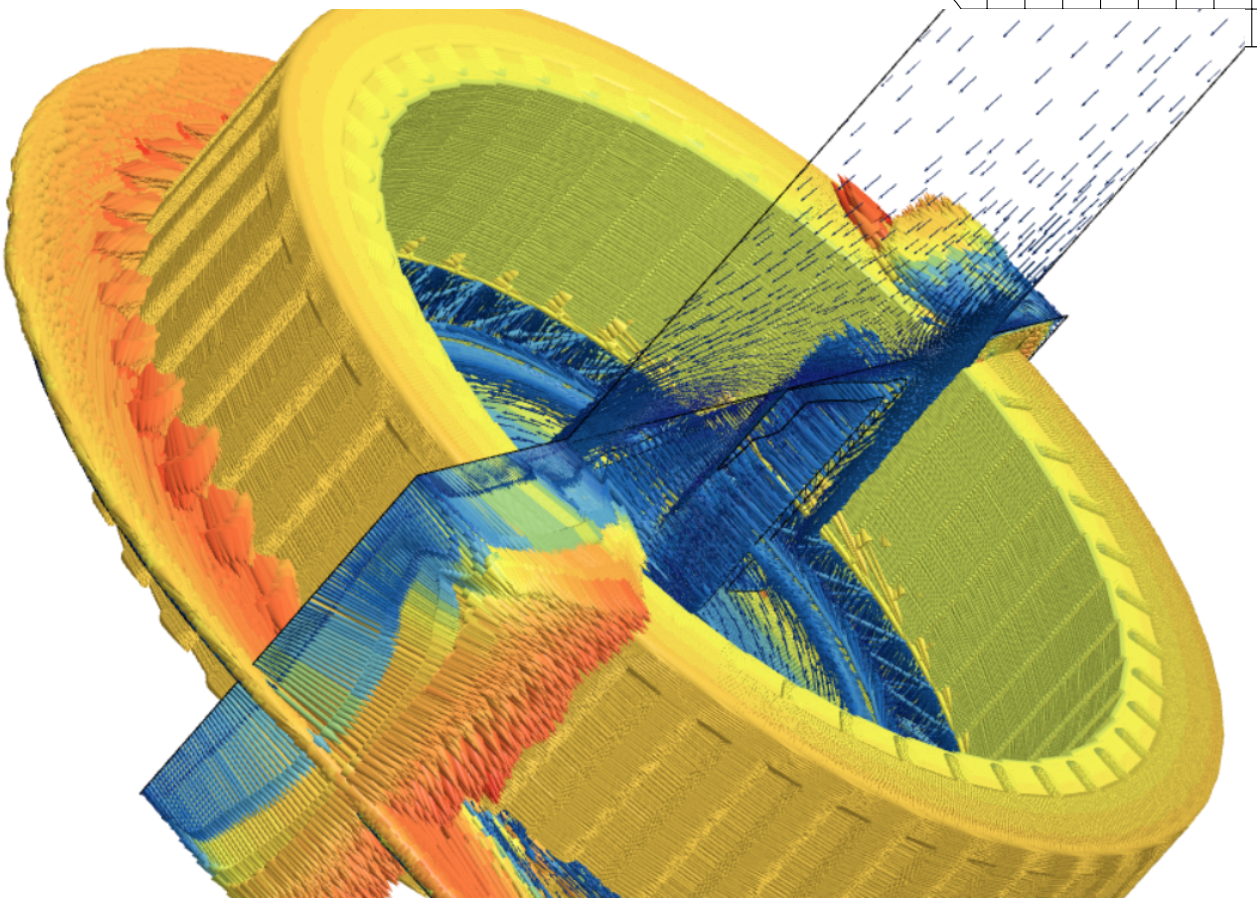
CHALMERS



Optimization of Fan Blade Design Using CFD and Reinforcement Learning

Course Project in Fluid Mechanics

ARON OLOFSSON
EESHAN GANLA
JAKKA SAI SRINIVASA MANIDEEP
JAKOB OLSSON
KOGANTI NAGA SAI RAMU

Department of Applied Mechanics
Division of Fluid Mechanics
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
Course Project 2025:01

Optimization of Fan Blade Design Using CFD and Reinforcement Learning
Course Project in Fluid Mechanics
ARON OLOFSSON
EESHAN GANLA
JAKKA SAI SRINIVASA MANIDEEP
JAKOB OLSSON
KOGANTI NAGA SAI RAMU
Department of Applied Mechanics
Division of Fluid Mechanics
Chalmers University of Technology

Abstract

The design of fan blades has undergone significant advancements over the past century. However, it is important to consider whether conventional design approaches may unintentionally constrain the range of possible blade geometries. This paper investigates the potential of integrating Machine Learning and CFD to further improve and accelerate the fan blade design process.

To achieve this, the study focuses on two main tasks: developing an accurate numerical model of a centrifugal fan in STAR-CCM+ and creating a Reinforcement Learning (RL) framework which implements the STAR-CCM+ model to optimize the fan blade geometry. CFD simulations were performed using the $k-\omega$ SST solver, and a mesh convergence study was performed. The RL framework for blade optimization was based on the Deep Q-Network (DQN) algorithm, implemented in Python using Pytorch.

The CFD Model validation was carried out by comparing the performance curve obtained from STAR-CCM+ simulations with the manufacturer's fan curve. The results indicate that the developed model accurately predicts the flow field generated by the fan.

The static pressure rise across the fan serves as the primary performance metric for evaluating design improvements. The Reinforcement Learning (RL) approach successfully produced new and improved blade designs in each iteration. However, none of the generated designs outperformed the original fan blade. Despite this, the approach shows strong potential for improving blade design given more time and computational resources.

For future studies, additional methods can be incorporated to better evaluate blade design. For instance, investigating other Reinforcement Learning methods or alter the current environment to reduce its design constraints.

Keywords : Centrifugal Fan, Blade Geometry, Reinforcement Learning, DQN Algorithm, Computational Fluid Dynamics (CFD), Performance Optimization, Star CCM+

Acknowledgements

The completion of this project would not have been possible without the valuable support and expertise of several individuals. We would like to thank our supervisors Blanca Gonzalez Lozano, Anthony Jayanath Vivek and André Fransson at AFRY for their guidance and feedback throughout the project. The authors also thank Chalmers University of Technology for providing access to STAR-CCM+ and the necessary computational resources. Finally, we thank our sister team, Team Alpha, for their valuable collaboration throughout the project.

Contents

1 Introduction

Centrifugal fans are used in a wide range of applications, from cooling electronic devices to ventilation and HVAC systems. Among other things, their performance depends heavily on blade geometry, making design optimization crucial for improving efficiency [1].

Using Computational Fluid Dynamics (CFD) to model fan performance offers an efficient and cost-effective method for analyzing flow behavior. When combined with Machine Learning (ML), CFD becomes a powerful tool for automating the design optimization process. Moreover, this approach enables new, unbiased blade design choices that may not have been considered through conventional design strategies [2][3].

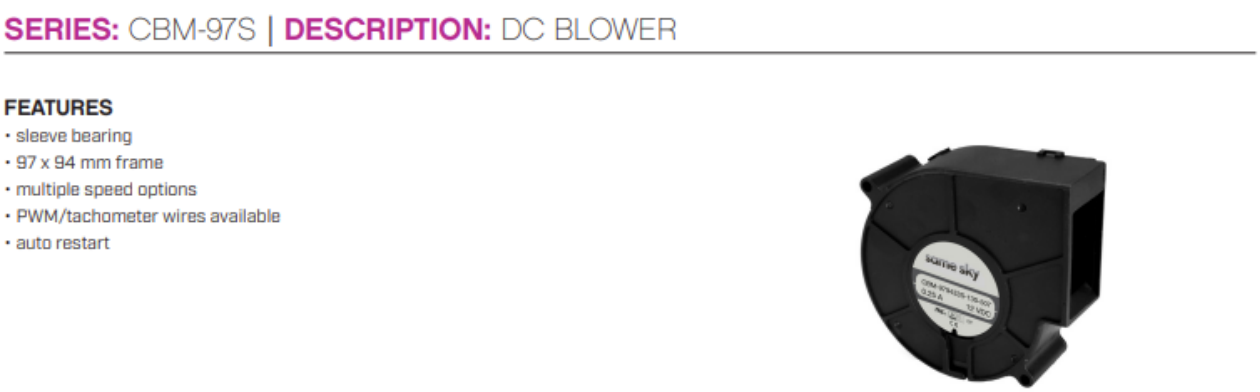


Figure 1.0.1: *CBM-97S DC Blower*

1.1 Project Definition

The objectives of the project are to develop a validate numerical model in STAR-CCM+ that accurately simulates the performance of a fan, and creating a Reinforcement Learning (RL) framework which implements STAR-CCM+ to optimize fan blade geometry. The fan simulated in the project is the CBM-97S DC blower by Same Sky. It can be seen in Fig. 1.0.1. The serial number of the fan is CBM-979433S-125-467 [4].

1.2 Purpose

The purpose of the project is to explore the possible advantages of using ML in combination with CFD software as a tool to optimize fan blade design. Improving the blade design can significantly increase the overall efficiency of the fan and reduce the design procedures.

1.3 Boundaries

The primary constraints of this project are the deadline for completion and the limited availability of cluster computing time for running numerical simulations.

1.4 Resource Inventory

The project requires STAR-CCM+ licenses for fan modeling and access to computing clusters to accelerate simulations, both of which will be provided by Chalmers. Additionally, Python, a commercial programming language, will be used to develop the machine learning code. All literary resources for preliminary studies will be sourced from the internet or the library.

2 Theory

2.1 Operating principle of Centrifugal Fans

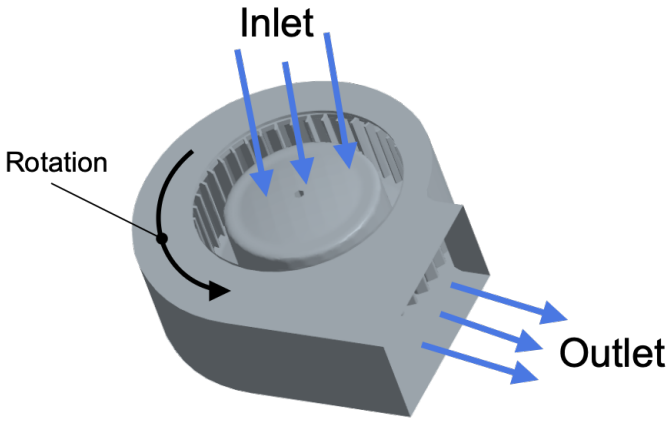


Figure 2.1.1: *Flow directions and rotational motion in a centrifugal fan.*

A centrifugal fan operates by air entering the fan perpendicular to the rotating impeller and exiting through a diffuser in the radial direction relative to the axis of rotation. As air enters through the impeller hub, it is accelerated by the rotating blades, which impart kinetic energy to the fluid. This kinetic energy is then partially converted into a static pressure increase as the airflow slows down within the diffuser [5]. As a result, the total pressure, defined as the sum of static and dynamic pressure,

$$P_{\text{total}} = P_{\text{static}} + P_{\text{dynamic}} = P_{\text{static}} + \frac{1}{2}\rho v^2, \tag{2.1.1}$$

increases across the fan.

The increase in static pressure across a fan is usually small enough that the air can be considered incompressible [6].

2.2 Performance Parameters

2.2.1 Fan Efficiency

The performance of a fan is measured by evaluating its efficiency, which can be done in a variety of ways. One common approach is to define the fan's *static efficiency*,

$$\text{Static Efficiency} = \frac{\Delta P_{\text{static}} \times \text{Airflow}}{10 \times \mathcal{P}}, \tag{2.2.1}$$

where $\mathcal{P}$ is the power required to drive the fan and ΔP_{static} is the static pressure rise across the fan. This equation implies that, for a given input power, increasing the static pressure rise or the airflow rate will improve the fan's static efficiency [5], [7].

By prescribing a constant airflow rate at the fan inlet, the optimization of different blade designs can be based on the resulting static pressure rise. In this approach, higher static pressure indicates a more efficient design under the given operating conditions.

2.2.2 Fan Curves and CFD Validation

Plots of static pressure versus airflow rate, called *fan curves*, are important for evaluating fan performance. They illustrate how the static pressure generated by the fan changes with varying airflow rates, typically decreasing as flow increases.

In CFD tools like STAR-CCM+, fan curves are produced by simulating fan performance across a range of airflow rates. Comparing these CFD-generated curves with manufacturer-provided fan curves helps verify the accuracy and reliability of the numerical model.

2.3 Fanblade design

The CBM-97S DC Blower is a centrifugal fan with Forward Curved Fan blades. The forward curvature results in large blade angles and larger mass flow than other centrifugal fans [8]. The fan blade can be parametrised according to five design parameters: The inlet blade angle β_1, outlet blade angle β_2, inner diameter D_2, outer diameter D_1 and blade thickness t, as shown in Fig. 2.3.1.

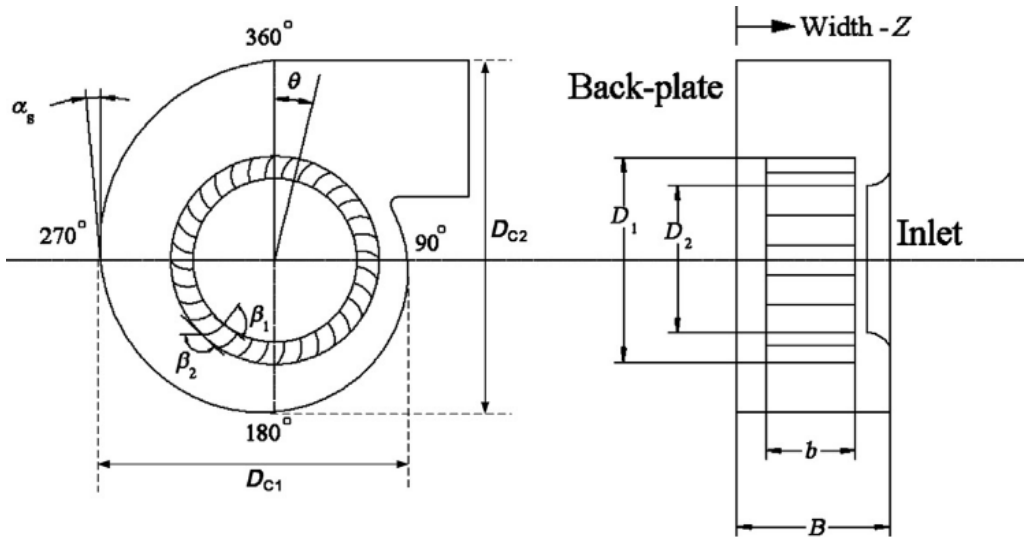


Figure 2.3.1: *Two-dimensional views of a forward-curved squirrel-cage fan including the blade design parameters. [8]*

2.4 Machine learning applied in shape optimization

Due to the increasing development in deep learning, aerodynamic shape optimization using machine learning has become progressively more common. Researchers from ASDL and the School of Aerospace Engineering, Georgia [9] investigated the use of Reinforcement Learning in the shape optimization of an airfoil. Reinforcement Learning proved beneficial for its data-driven and generative approach that does not require user-provided data.

2.4.1 Reinforcement learning

Reinforcement Learning (RL) is one of the three basic machine learning models together with unsupervised and supervised learning. RL focuses on striking a balance between exploration and exploitation in its environment to maximize the reward. When the agent is "exploring", it refers to discovering new solutions, and when it is "exploiting", it takes advantage of current knowledge to maximize rewards. RL has the capacity to identify unique and reliable design/order combinations to meet various conflicting objectives and return performance feedback in real time, doing so without tedious exhaustive simulations of architecture to achieve refinement. these machine learning method includes hyperparameters, which are values that are user specified that configures certain aspect like learning rate, convergence and stability.

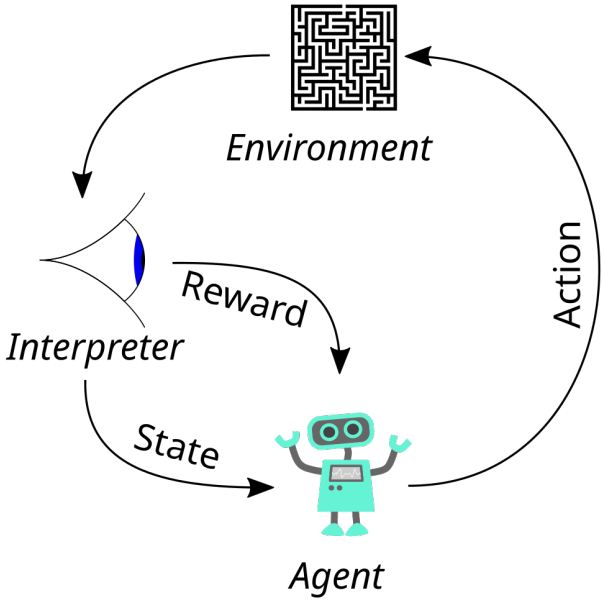


Figure 2.4.1: *Components in a typical Reinforcement Learning (RL) system. By Megajuice - Own work, CC0 [10]*

Common RL methods are Q-learning, which determines what action to take by evaluating q-values for each available action. Q-values are initially undetermined, but through interaction with the environment, that further updates these values by improving the choices that lead to higher rewards. In a standard Q-learning algorithm, these values are assigned in a Q-table. DQN instead predicts these Q-values using a convolutional neural network which works well for continuous state spaces. Figure 2.4.2 visually shows the difference between these two methods. DQN also includes improvements that come with the neural network like, Experience Replay that samples batches to learn and Target learning that stabilizes learning.

Experience replay, also known as Buffer replay, as said samples the agents state, action reward and keeps it as a memory, often in the form of tuples. This is what allows the agent to learn from previous experiences. This stabilizes the network by ensuring that the method is not highly revolved around correlation. It also improves data efficiency since experiences are used over and over. This is also good for when expensive action by possibly reducing the number of action required. This is essential for off-policy methods like DQN.

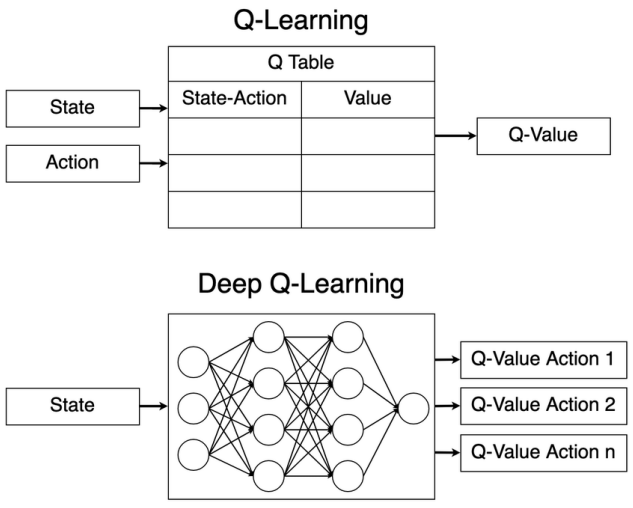


Figure 2.4.2: *Comparison between Q-learning and DQN [11]*

The biggest dilemma when it comes to Reinforcement Learning method is the balance between exploration

and exploitation [12]. Generally the agent wants to begin by exploring the environment and revert to exploiting after getting enough experience. One method often used is the epsilon-greedy algorithm. It is one of the most popular method because of its simplicity. It uses probability distribution to either make random or educated decision based on DQN. epsilon is defined as the probability of making a random action, meaning it is a value between zero and one. Where one is with 100 percent probability of making a random choice. to get the desired effect of exploration and exploitation discussed earlier, the epsilon needs to be decaying over iterations. This enables the agent to eventually gain enough experience to have evolved the Q-network to make the superior action every time.

3 Methodology

This chapter describes the procedures undertaken to complete the two main tasks set by the project; developing a validated CFD model that accurately simulates the flow field produced by the fan, and establishing a Reinforcement Learning (RL) framework for optimizing the fan blade geometry.

3.1 Developing the Validated CFD Model

The CFD simulations were performed using the commercial software Star CCM+, with the implicit steady-state solvers. The simulation framework modeled the blower's impeller and volute components, focusing on accurately capturing the mass flow and pressure distribution within the blower. An overview of the complete workflow is illustrated in Fig. 3.1.1, where each step in the process is briefly described chronologically in the following subsections.

3.1.1 CAD Model Acquisition & Cleanup

The CAD model of the fan was obtained from the manufacturer's website [3]. Before importing the geometry into STAR-CCM+, minor modifications were required to prepare the model for meshing and simulation. These modifications included the addition of extended inlet and outlet regions, closing any open surfaces, and smoothing sharp edges.

This pre-processing stage, often referred to as 'CAD cleanup,' is crucial for ensuring mesh quality in later steps and, by extension, simulation stability. In this case, it was carried out by a separate team working on a closely related project.

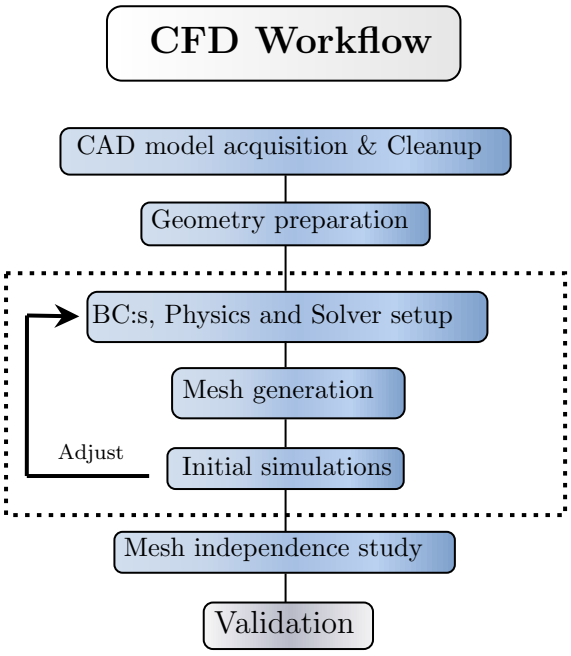


Figure 3.1.1: *Workflow used for CFD model setup and validation in STAR-CCM+.*

The computational domain and the corresponding region names used in the following subsections are presented in Figs. 3.1.2 and 3.1.3.

3.1.2 Geometry preparation

After importing the cleaned-up CAD model into STAR-CCM+, additional steps were required before assigning parts to regions. First, the internal volume of the model was extracted to isolate the air inside and around the fan. Then a cylindrical part was fitted around the rotating components and the fan blades were subtracted from this cylinder to define the air volume surrounding the blades. Finally, this cylindrical volume was subtracted from the overall internal volume. As a result, the geometry was divided into two parts: one stationary region that excludes the rotating components and one rotating region that contains them. The rotation of the fan was modeled using the Moving Reference Frame (MRF) approach applied to the rotating region. After the geometry was properly set up, the stationary and rotating parts were assigned to regions.

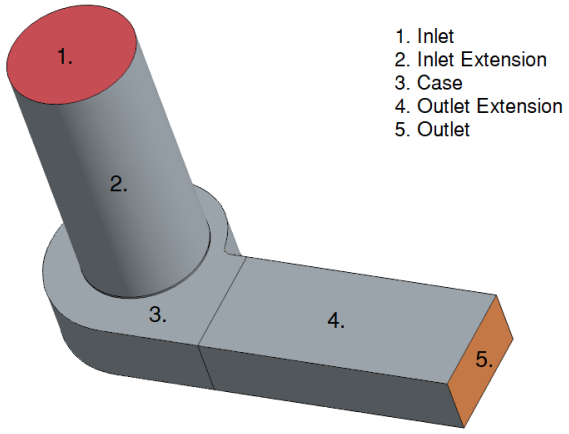


Figure 3.1.2: *Entire simulation domain.*

3.1.3 Numerical setup, Mesh and Initial simulations

The encircled steps in the CFD workflow diagram were carried out through an iterative process. After assigning parts to regions, this process included defining appropriate boundary conditions, selecting relevant physical models and solvers, generating the mesh, and running simulations until convergence.

Once the initial simulations were completed, the boundary conditions (BC:s), the physical conditions and the solver settings were reviewed and adjusted if the flow exhibits unexpected behavior. The mesh was refined in regions where complex flow features were observed.

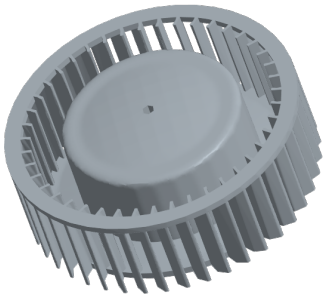


Figure 3.1.3: *The rotating impeller enclosed within the case, composed of the Hub and Blades.*

Additionally, appropriate stopping criteria were defined to ensure that the solution was both numerically converged and physically realistic. As residual convergence was not fully achieved, asymptotic limits of key metrics was used instead. Each stopping criteria was evaluated over 100 consecutive iterations, with convergence assessed based on the range between the minimum and maximum values during that interval. The following criteria were applied:

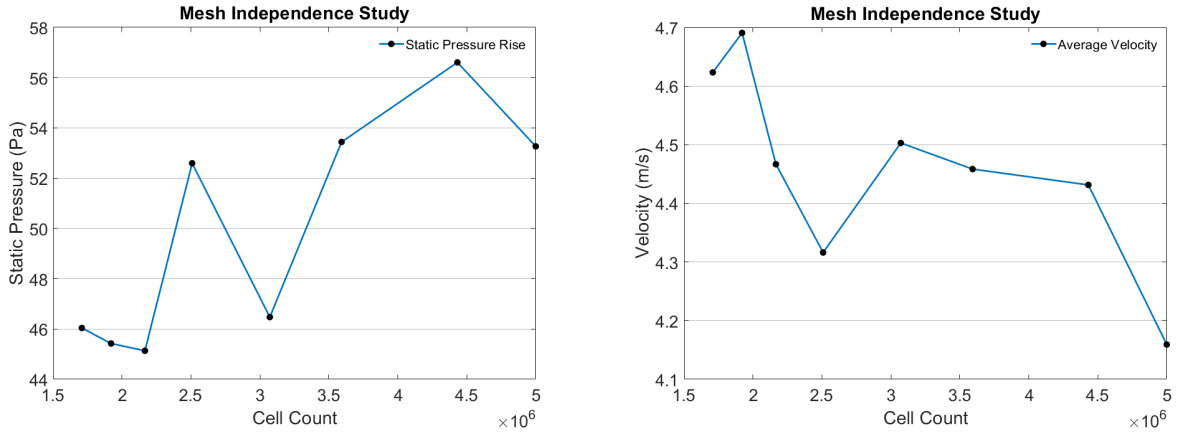
- Surface-averaged velocity on Outlet: $[\min, \max] < 0.1$ m/s
- Static pressure increase between inlet and outlet: $[\min, \max] < 0.1$ Pa
- Mass flow imbalance (Inlet vs Outlet): $[\min, \max] < 1 \times 10^{-4}$ kg/s

3.1.4 Mesh independence study

After a satisfactory flow field had been achieved, a mesh convergence study was conducted to determine the required mesh resolution. Using an overly fine mesh can significantly reduce computational efficiency without yielding meaningful improvements in accuracy. The study began with using a coarse mesh and running the simulation until convergence. The mesh was then progressively refined by increasing the number of cells, and the simulations were repeated.

Mesh convergence is typically achieved when further refinement leads to negligible changes in key output quantities, such as the average velocity across a plane or the static pressure rise across the fan. However,

in this case—as shown in Figure 3.1.4—the output quantities did not consistently converge toward a single value. Instead, slight variations were observed with each level of mesh refinement. The full results of the mesh refinement study are presented as a table in Appendix A. The variation in results could be attributed to several

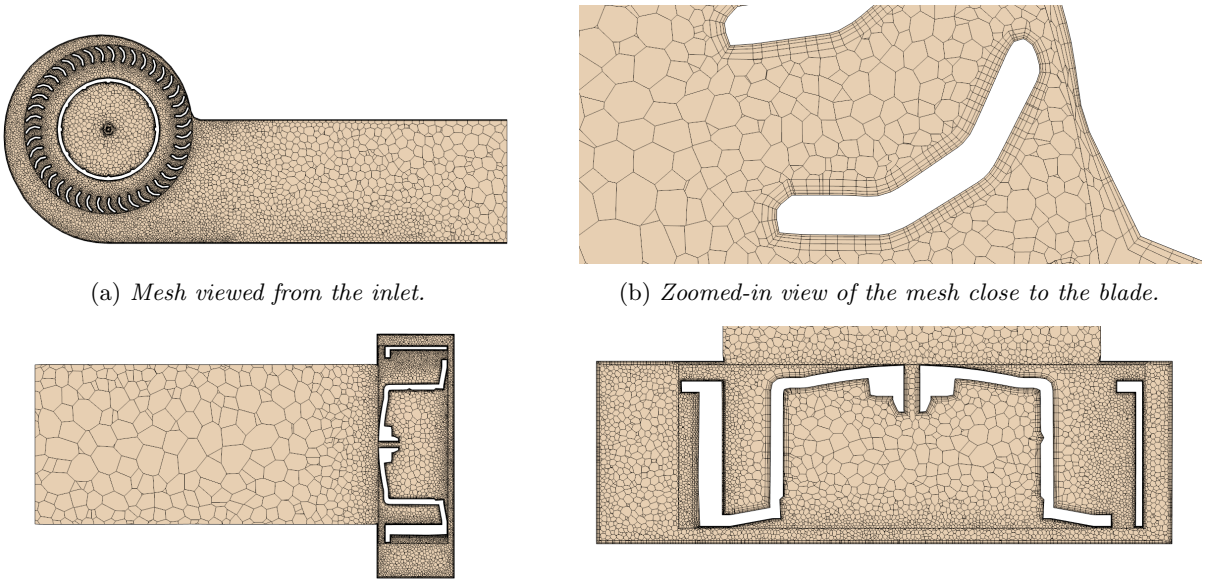


(a) *Static pressure rise across the domain.* (b) *Average outlet velocity.*

Figure 3.1.4: *Mesh independence study: comparison of static pressure rise and outlet velocity across different mesh densities.*

factors. Although the simulations are numerically converged, they were performed using a steady-state solver. This approach assumes a time-invariant flow field, which can be limiting given that the actual flow is turbulent and inherently unsteady. As a result, transient flow features—such as vortex shedding or local fluctuations in velocity and pressure—are averaged out, potentially introducing variability when comparing mesh refinements.

A mesh comprising 3,071,752 cells was selected for the validation simulations, as it accurately reproduced the manufacturer's fan curve, provided sufficient resolution around the blades, and maintained simulation times within an acceptable range. The mesh is illustrated in Fig. 3.1.5. While the actual mesh is three-dimensional, a two-dimensional projection is shown for improved clarity. The mesh itself consists of a combination of polyhedral and tetrahedral cells. Near-wall regions along the blade surfaces, hub, and case are resolved using prism layers to capture the boundary layer accurately. The surface-averaged wall y^+, which serves as a metric for near-wall resolution, is approximately 1.237 across the domain but <1 at the blades.



(a) *Mesh viewed from the inlet.* (b) *Zoomed-in view of the mesh close to the blade.*

(c) *Mesh viewed from the outlet.* (d) *Zoomed-in view of mesh viewed from the outlet.*

Figure 3.1.5: *The mesh used when validating the CFD model.*

3.1.5 Validation

To validate the simulation results, a series of simulations were conducted at different inlet volumetric flow rates. The resulting pressure rise across the fan was compared against the manufacturer's fan performance curve provided in the product datasheet. This comparison served to verify that the CFD model accurately captured the fan's behavior under varying operating conditions. Table 3.1.1 summarizes the selected physics models, solver settings, and boundary conditions applied during the CFD model validation. The k–ω SST turbulence model was chosen for its ability to accurately resolve boundary layer behavior near the wall. [H]

Table 3.1.1: Physics Models, Solver Settings and Boundary Conditions used for CFD Validation

Physics Models	**Solver Settings**	**Boundary Conditions**	
Coupled Flow	2nd Order	Inlet:	Velocity Inlet
$k-\omega$ SST	Implicit	Outlet:	Pressure Outlet
All y^+ Wall Treatment	Automatic CFL ≤ 5	Inlet Extension:	Slip Wall
Gradients	Expert Initialization	Outlet Extension:	Slip Wall
Solution Interpolation		Blades:	No-Slip Wall
Turbulent		Blade Hub:	No-Slip Wall
Steady		Casing:	No-Slip Wall
Constant Density			
Three Dimensional			

Remarks on Boundary Conditions

Two types of inlet boundary conditions were tested: a stagnation inlet and a velocity inlet.

The stagnation inlet represents still air being drawn into the fan by the suction created by the rotating blades. However, it imposes a fixed total pressure that does not respond dynamically to the suction generated by the fan, which can lead to unrealistic pressure distributions near the inlet.

In contrast, the velocity inlet allows the pressure to adjust naturally as the fan pulls air in [13]. Based on test simulations and these considerations, the velocity inlet was selected for the final setup.

3.2 Fan Blade Shape Optimization

The final machine learning program is a combined network of Python code, STAR-CCM+ simulations, Java macros, and bash codes. It consists of three individual Python codes that cooperate with each other. These codes are **RL.py**, **FanGeometryEnvironment.py** and **Curve_FC.py** (Appendix add here). The RL agent will recommend a redesign of the blade geometry, use CFD to determine the effect the redesign had on the fluid flow and use the CFD data to update the RL policy-favoring the configurations that produced greater rewards.

The **RL.py** code includes the DQN definition and the agent defined using the library Pytorch. It also serves as the base code that will run the entire procedure. More thoroughly, the code selects an action (increase or decrease the chosen parameter) and gets the reward and new state from the environment, samples to the replay buffer and optimizes the "policy". this iteratively runs steps until the maximum amounts of step is reach. after this it update the target from its optimization. **FanGeometryEnvironment.py** Creates the environment for the RL code. Its main function is to turn actions into the next state and returns the reward and it does this as shown in figure 3.2.2. Lastly the **Curve_FC.py** code simply creates the geometry and exports it as a ".csv" file. It is also used to save geometries.

3.2.1 Reinforcement Learning

The RL code moves through a vast design space through the agent's learning policy through a range of factors while customizing design parameters discussed in Section 2.3. The input parameters for the code are β_1, β_2 and thickness t. To assure that the fan blade is well defined, upper and lower boundaries are set for each of the variables. Since the simulations take a long time, actions become quite expensive. Therefore, some choices were made to suit our time span. Such choices were making the bigger step sizes and reducing the step size each episode. Gamma was reduced, which is a factor that helps reduce immediate reward bias. Reducing it makes

the RL converge faster with the risk of instability. If the simulation were not to converge, and for the DQN not to learn from nonphysical data, a penalty is given for when the solution has not converge within a maximum iteration window.

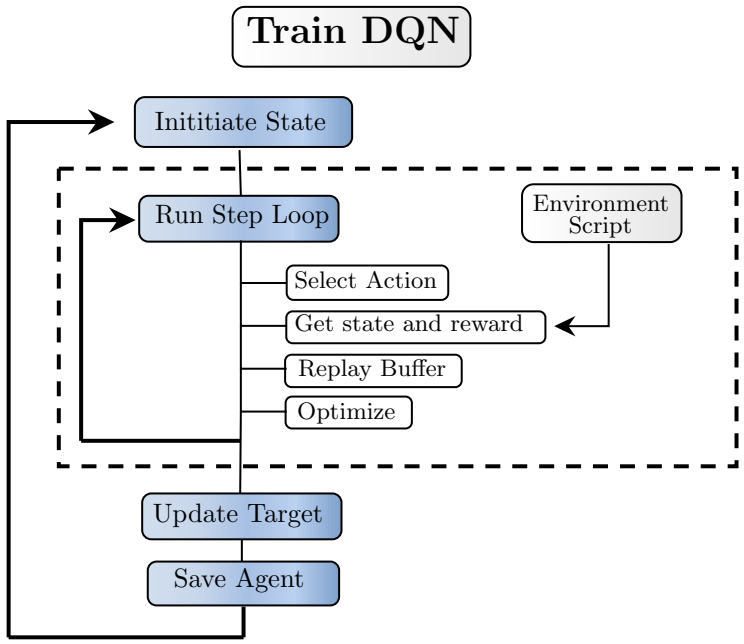


Figure 3.2.1: *Diagram of the DQN training process. The actions performed within the striped square corresponds to one step. The full loop corresponds to one complete training episode. For each episode a specified number of steps are performed.*

Another decision was made to make the Q-network more efficient was implementing Prioritized Experienced Replay (PER), which samples more important experience instead of random ones. It does this by only samping the action which give a higher TD error (temporal difference). This further enhances the code to reduce the amounts of required actions taken.

3.2.2 Adaptation of CFD Model for ML Optimization

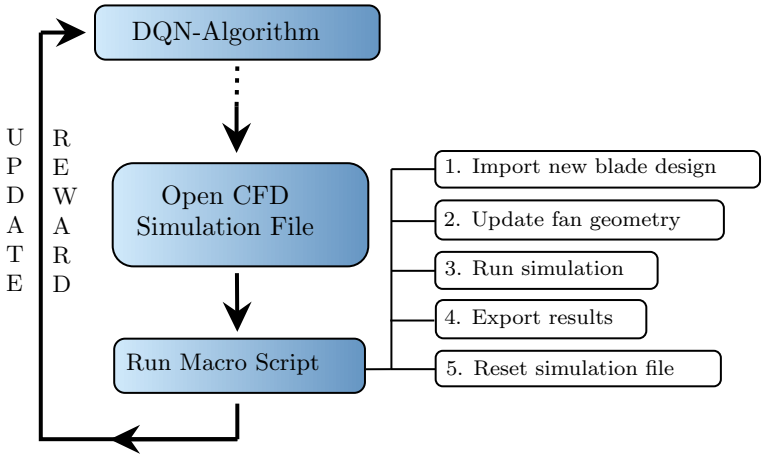


Figure 3.2.3: *Diagram of CFD-Based Fan Blade Optimization Loop*

The validated CFD model, developed using the methodology described in Section 3.1, served as the foundation for evaluating different blade geometries.

To couple the CFD model with the RL algorithm, a macro script was created in STAR-CCM+. The script

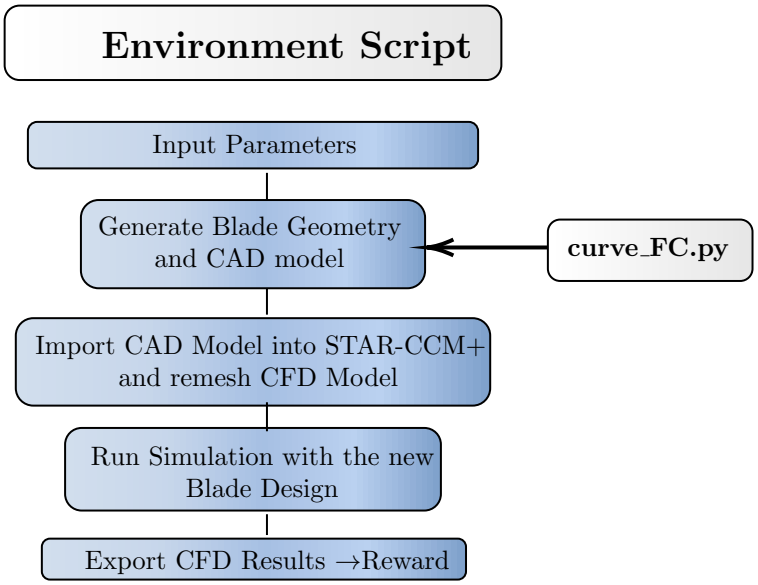


Figure 3.2.2: *Diagram demonstrating the actions performed during one step of the* **FanGeometryEnvironment.py** *script.*

automated the import of new blade geometries, executed simulations until convergence, and then reset the model to its initial state. After each run, key performance data were exported to a CSV file for updating the reward function in the DQN algorithm. This iterative process enabled each new blade design to be informed by the performance of previous designs, resulting in progressive improvement across RL episodes. The workflow is illustrated in Fig. 3.2.3. In order to execute the simulations in the code, bash files were created that ran the simulations with the macro.

To reduce simulation time for each new blade geometry, solver settings and stopping criteria were adjusted. Unlike the CFD validation simulations, these simulations did not require fully converged or highly accurate results—only sufficient accuracy to reliably assess relative performance differences between blade designs.

The following adjustments were made to accelerate the simulations:

- Stopping criteria were relaxed.
- Mesh cell count was increased.
- The fan was initialized directly at its final RPM, rather than ramping up gradually.
- Expert initialization was disabled.
- A convergence accelerator was enabled.

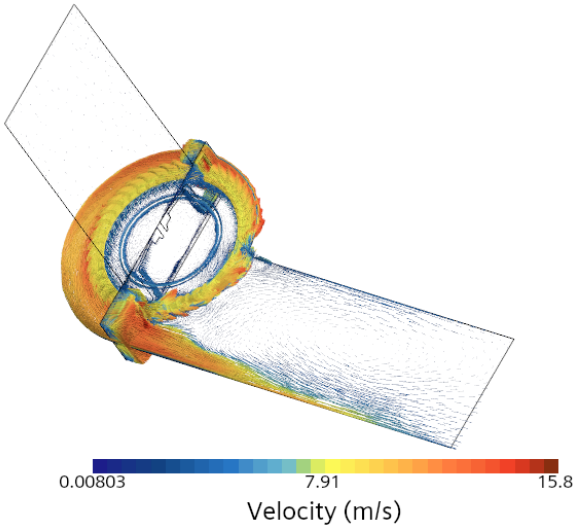
4 Results and Discussion

This chapter presents the results of the CFD model validation and the blade geometry optimization. Following the structure of the previous chapter, the CFD validation results are discussed first, followed by the outcomes of the blade geometry optimization.

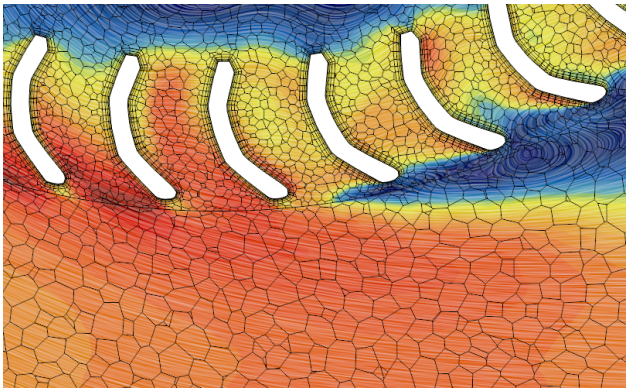
4.1 Fan Performance

To replicate the fan performance curve provided by the manufacturer, a series of simulations were performed across a range of volumetric flow rates. Since the flow field behavior was consistent across different inlet conditions, the results presented here correspond to a single representative volumetric flow rate. To aid visualization, flow variables are displayed on 2D planes that intersect the 3D domain, where key features are most clearly observed.

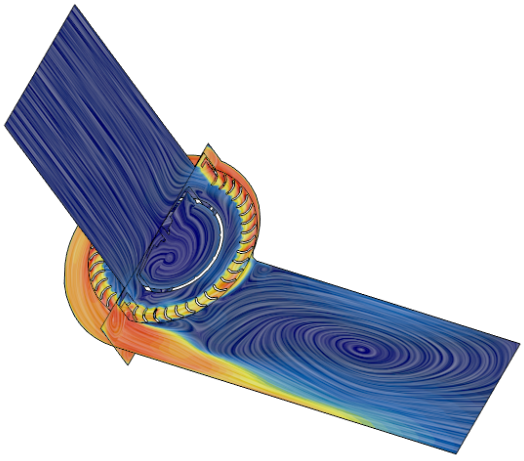
4.1.1 Velocity fields



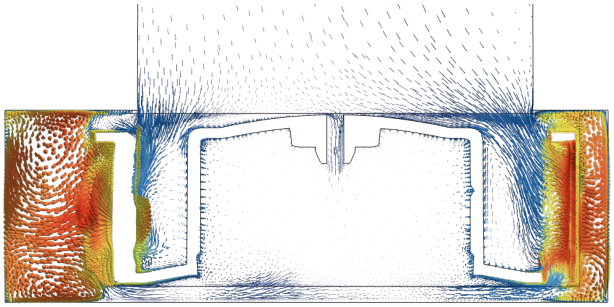
(a) *Isometric view showing velocity vectors.*



(b) *Close-up of the vector field inside the case as viewed from the outlet.*



(c) *Isometric view showing velocity streamlines.*



(d) *Zoomed-in view of blades with mesh.*

Figure 4.1.1: *Velocity field showing vectors, streamlines, and mesh around the blades.*

Figures 4.1.1a and 4.1.1b provide an overview of the velocity vector distribution within the centrifugal fan. The color map represents the velocity magnitude (in m/s), with red regions indicating areas of high velocity.

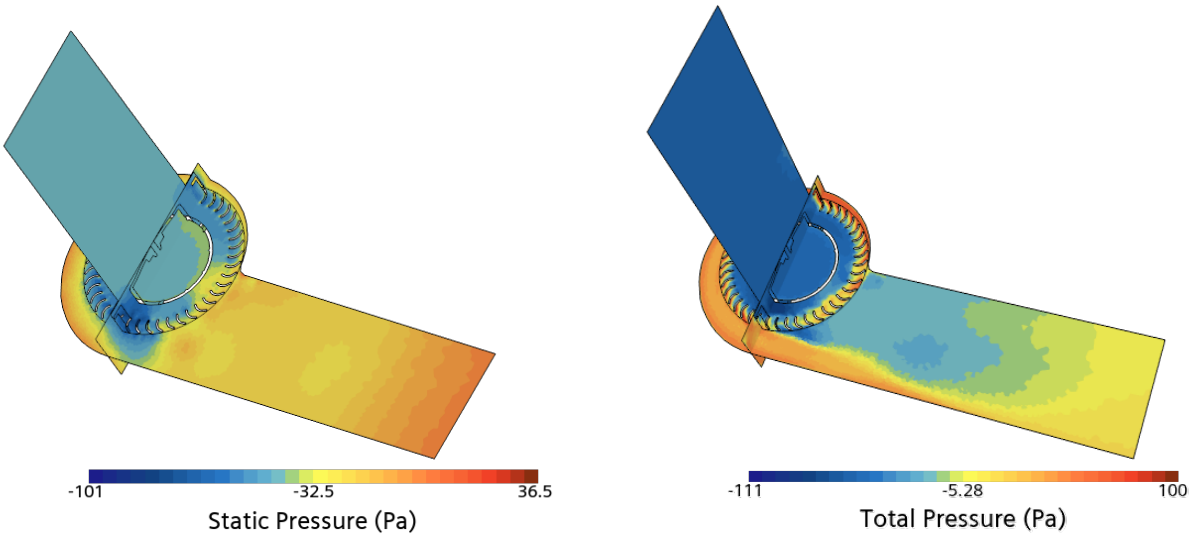
The results indicate that the high-velocity air exiting the fan is primarily directed toward one side of the outlet extension. In contrast, the flow in the remaining portion of the outlet exhibits significantly lower velocities, and a large recirculation zone forms in this region, as clearly observed in fig. 4.1.1b.

Figure 4.1.1c shows how the flow accelerates as it approaches the fan casing. The flow upstream of the casing begins at a relatively low velocity but accelerates near the blades due to the low-pressure region and suction effect generated by the rotating impeller. The flow becomes highly rotational in the narrow gap between the blades and the casing, indicating the presence of strong shear layers that impose significant loads on the blades.

Additionally, smaller-scale recirculation zones are observed near the blades, as shown in fig. 4.1.1d, which also displays the computational mesh. These localized flow structures emphasize the need for a sufficiently refined mesh near the blade surfaces to accurately resolve the complex flow phenomena in these critical regions.

Overall, the simulated flow field aligns well with physical expectations. The blades impart momentum to the incoming flow, generating high-velocity regions near the blade tips and casing, and producing a focused jet of air exiting downstream of the blades. Furthermore, the results highlight the turbulent nature of the flow within the casing and at the fan outlet.

4.1.2 Pressure Distribution



(a) *Static pressure distribution.* (b) *total pressure distribution.*

Figure 4.1.2: *Static and total pressure distribution.*

The static and total pressure distribution across the domain is presented in fig. 4.1.2. The lowest pressure is observed in the rotating region near the blades, which corresponds to the area of high flow acceleration close to the casing, as also seen in fig. 4.1.1d. This low-pressure region is a direct result of the suction effect generated by the rotating impeller.

As expected from theory, both static and total pressure increase across the fan. The dynamic pressure increases significantly in regions of high flow velocity.

While dynamic pressure varies sharply with velocity, the static pressure exhibits a more gradual and spatially uniform increase as the flow exits the fan casing. Notably, the static pressure at the inlet is below atmospheric pressure (indicated by negative values). This occurs due to the imposed outlet boundary condition, where the static pressure is fixed at atmospheric levels. To satisfy the required pressure rise across the fan, the inlet pressure decreases accordingly.

4.1.3 Fan Performance Curve

Figure 4.1.3 presents the fan performance curve, comparing experimental data provided by the manufacturer with the results obtained from CFD simulations. Static pressure (in mmH_2O) is plotted against air flow rate (in CFM).

The CFD results show reasonable agreement with the manufacturer's data. While the predicted values are of similar magnitude, the CFD model exhibits a more linear increase in static pressure as airflow decreases, in contrast to the slightly nonlinear trend observed in the experimental data.

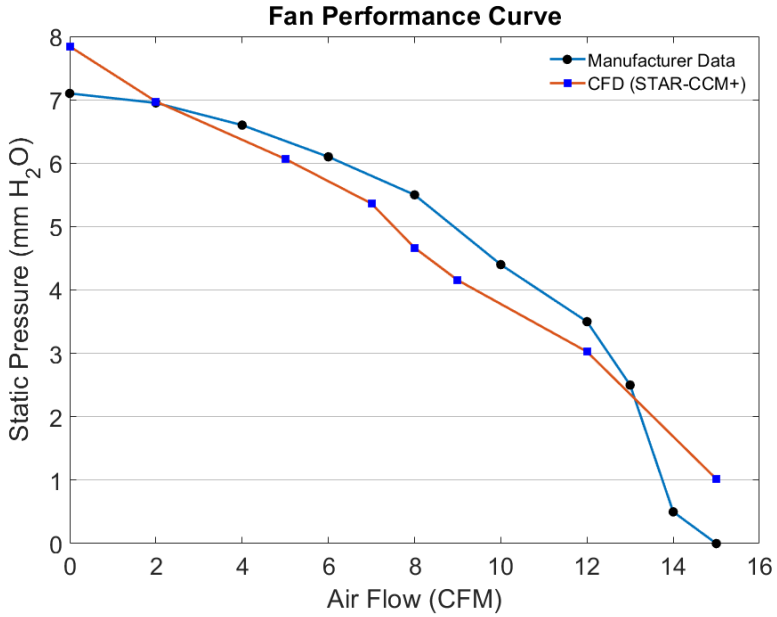


Figure 4.1.3: *Comparison of CFD and manufacturer fan performance curves.*

The most significant discrepancies appear at the extreme ends of the curve—particularly at flow rates 0 and 15 CFM. These deviations could arise from modeling inaccuracies such as mesh resolution, solver settings, or the choice of turbulence and physical models. Closer alignment would require detailed knowledge of the experimental setup, including the precise locations of static pressure measurements and testing procedures.

Despite these differences, the results provide reasonable agreement, indicating that the CFD model offers a sufficiently accurate approximation of the actual fan behavior.

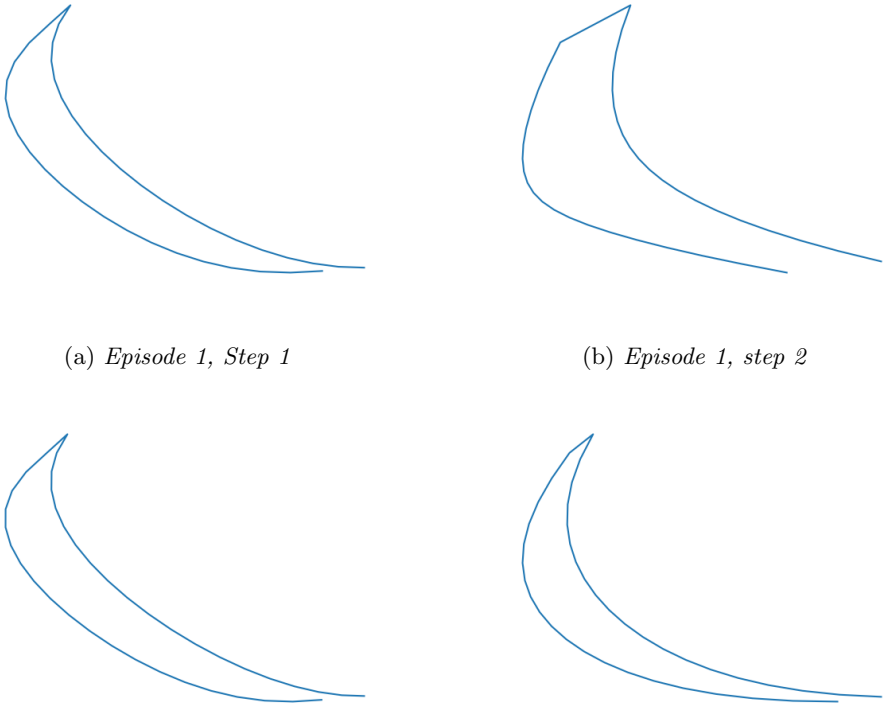
4.2 Shape optimization

Due to time limitations, insufficient amount of runs was made, making improvements minimal. Conclusions can still be made about the ML model regarding improvements and speculations about further applications.

4.2.1 Optimization Progress

A csv file was saved at each step, and after a few runs, it only saves for the last in each episode. These give a representation of the progression over the code. Over the first iterations, it was validated that small progressions with mostly randomized actions were taken. Figure **??** shows the first iterative steps taken and the given geometry state. These steps gave little in terms of results, and were mostly a way to see if the model would optimize for it. Examining figure 4.2.1a and figure 4.2.1b, the action produced on the blade was thickness. For many of the actions, the variation in thickness led to minimal changes in pressure difference. Compared to the difference between 4.2.1c and 4.2.1d where the angle was increased, which gave a more rewarding change in static pressure increase across the fan. Across all simulation the RL managed to increase the static pressure increase from 20 Pascals average to a pressure increase of 40 Pascals. There is a point to be made that the design process can be reduced by implementing Machine Learning that reduces the amount of simulations ran by using just automated CFD by itself. Theoretically this can be supported but no conclusion can be made

with current results.



(a) *Episode 1, Step 1*

(b) *Episode 1, step 2*

(c) *Episode 2, step 1*

(d) *Episode 2, Step 2*

Figure 4.2.1: *Show the iterations of the first few episodes.*

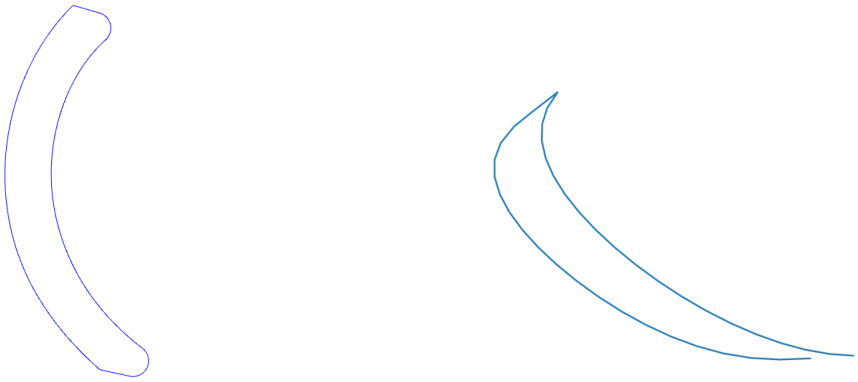
4.2.2 Comparison between Initial and Final Fan Blade Design

Comparing the original design with the best-performing fan blade obtained from the cluster task, it can be concluded that the current RL-based design remains insufficient. The CBM-97S DC blower fan blade achieves a static pressure of approximately 52.6 Pascals, as shown in Figure 4.1.3, whereas the RL-generated fan blade produces only 42 Pascals, as illustrated in Figure 4.2.4. When compared with Figure 4.1.2, it is evident that there is a significant variation in static pressure distribution between the original and the new fan blades. Relative to the amount of experience it gathered, it reached a state of similar class as the original design. It is highly probable that with continued iterative progression, the design will be able to surpass it.

4.2.3 Machine Learning method evaluation

Even with the limited time span, the DQN did suffice. It did show signs of convergence even if it was slow, but was moderately unstable after the whole cluster operation. As talked about in Section **??** a lot of hyperparameters were changed for quicker convergence, but were manually picked with respect to the restricted time. Hyperparameters such as gamma were considerably decreased compared to normal. This has made the impact of individual steps much more significant.

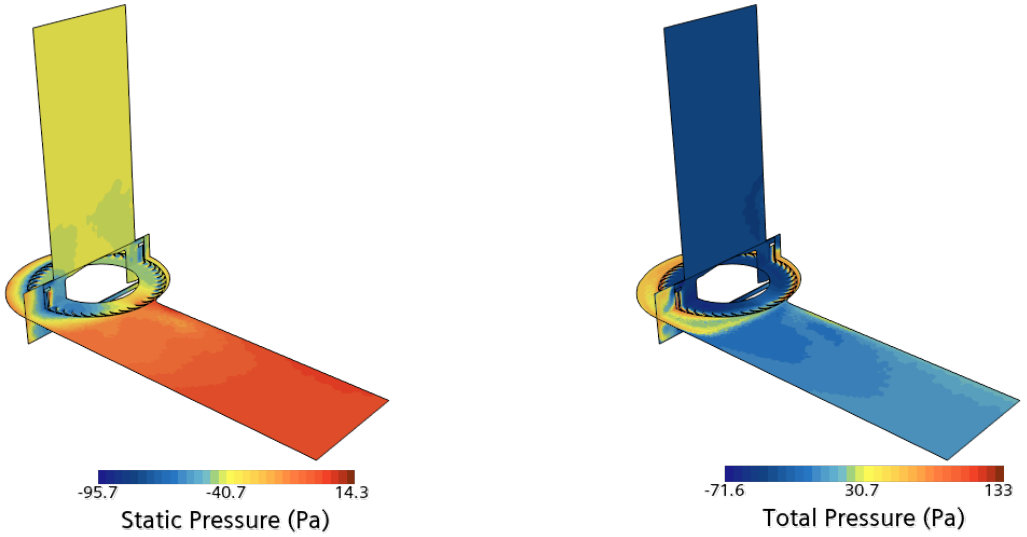
The report "A Reinforcement Learning approach to airfoil shape optimization" addresses the dilemma with DQN compared to other algorithms. The report says "DPG and DDPG can address both continuous action and state spaces and exhibit better stability and convergence properties compared to DQN [9]", which could be vital if the project does not get enough experience.



(a) *Original fan blade geometry, CBM-97S DC blower by Same Sky [4]*

(b) *Best-performing fan blade from the full cluster task*

Figure 4.2.2: *Comparison of original and optimized fan blade geometries*



(a) *Static pressure distribution.*

(b) *Total pressure distribution.*

Figure 4.2.3: *Static and total pressure distribution for the new Fan Blade.*

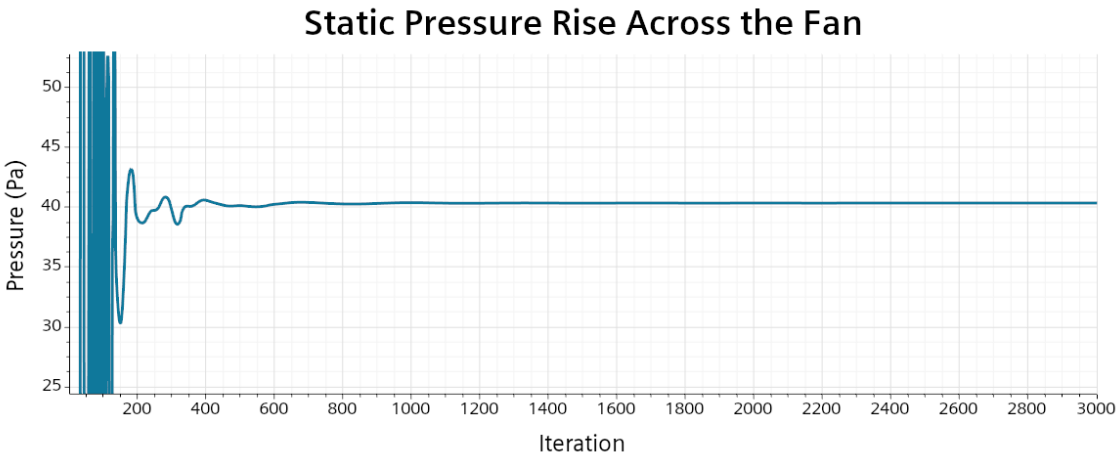


Figure 4.2.4: *Static Pressure difference between the inlet and the outlet for the new blade.*

4.2.4 Computational cost

The CFD simulations were computationally intensive, limiting the number of iterations possible within the project timeline and available cluster computing resources.

4.2.5 Future Improvement

During the development of the CFD model and RL framework, several areas for improvement were identified that could contribute to more accurate results. Some of those are discussed in this section.

The CFD simulations were performed using a steady-state solver. Exploring transient solvers with curvature correction could more accurately capture the inherently unsteady nature of the flow. A more comprehensive mesh convergence study should be conducted to ensure that the mesh quality is adequate and that the results are not influenced by insufficient mesh resolution. Testing alternative solver settings and turbulence models could help achieve closer agreement with the validation data.

As discussed in Section 4.2.3, attempting and investigating other RL algorithms might lead to better results. Changing the environment by putting less restriction on the actions would increase the solution space without restricting it to explore design decisions developed from already established methods. While this would increase the simulation run-time, the gain in possibilities could be beneficial. Enabling the agent to self-terminate instead of always reaching the maximum steps would be another improvement to the DQN algorithm. Further fine-tuning and optimization of the hyperparameters would certainly enhance the algorithm. Adjustment to the reward-functions could be a possible area to investigate for better results-it would spare the agent from doing unnecessary simulations that could possibly worsen the design during the well-educated episodes. Adding surrogate models and tweaking the DQN settings could shorten simulation times, thereby enabling more blade designs to be generated. Simulations on the data cluster could be made more effective by running more actions in parallel, thus accelerating the learning process. Additionally, running it for an extended period of time would certainly be advantageous.

The method for optimizing the blade geometry developed in this project uses the increase in static pressure rise across the fan as the only evaluation parameter. While this is in theory one valid way of estimating the efficiency of the fan, there are several more parameters that should be taken into account when designing the blade design. The choice of only using one optimization parameter was made to reduce computational complexity. The main objective of this project was to develop a functional model capable of improving blade design using CFD and Reinforcement Learning, rather than to produce the most physically optimal blade geometry.

5 Conclusion

The objectives of this project, as presented in this report, were twofold: first, to develop a numerical model in STAR-CCM+ that accurately replicates the performance of a real fan; and second, to create a reinforcement learning (RL) algorithm capable of optimizing the fan blade geometry for improved performance.

5.1 CFD Model

The results of the computational fluid dynamics (CFD) validation, along with their agreement with the fan performance curve provided by the manufacturer, suggest that the numerical model reliably predicts the flow field generated by the actual fan. However, obtaining closer agreement would require knowledge of the experimental setup and conditions.

The quality of a numerical CFD model can always be improved. With more time to improve the mesh and run simulations for a larger range of solver and physics settings, the model would certainly perform even better. However, given the time limitation that has been set for the project's completion, the model is considered to be satisfactory.

5.2 Optimizing Blade Geometry Using Reinforcement learning

The RL can be concluded as undetermined in terms of results, but theoretically it can practically guaranteed that RL will be the future in fan blade design and fields involving shape optimization. DQN has been proven effective at learning in a multi-dimensional space. The angles $\beta_1$andβ_2 proved more volatile compared to thickness and has a more significant influence on the pressure difference. Future improvement can be made covered in section 4.2.5 for example mitigating the environment constraints and so forth. Future research

A Mesh Convergence Study

Table A.1: Summary of velocity and pressure values at different mesh sizes. Velocity in unit (m/s) and pressure in unit (Pa).

Mesh Size (nr of cells)	Avg V at COP	Avg V at OP	Avg V at YNP	ΔP_{static}	ΔP_{total}
1,708,774	4.589	2.565	4.623	46.041	50.787
1,921,140	4.092	2.665	4.691	45.424	50.274
2,166,480	4.845	2.316	4.467	45.134	48.855
2,508,607	4.003	2.244	4.316	52.594	56.049
3,071,755	4.179	2.387	4.503	46.474	50.085
3,591,994	4.425	2.074	4.458	53.442	56.624
4,431,532	4.287	2.043	4.431	56.607	58.858
4,999,612	3.858	2.607	4.159	53.267	57.724

COP: Case Outlet Plane — a plane located near the casing, oriented with the same unit normal as the outlet plane.
OP: Outlet Plane — the downstream boundary of the domain where flow exits.
YNP: Y-Normal Plane — a plane perpendicular to the rotational axis of the blades.

Bibliography

[1] ACDC Fan. "Centrifugal fan design: Essential concepts for optimizing efficiency." Accessed: 2025-05-22. (2024), [Online]. Available: https://www.acdcecfan.com/centrifugal-fan-design/.

[2] M. Dineshkumar, S. Manickavel, K. Paraneetharan, and M. Rohith, "Cfd analysis and evaluation of performance for centrifugal fan using wind tunnel testing," *International Journal for Scientific and Advanced Research in Technology (IJSART)*, vol. 11, no. 3, pp. 131–132, 2025, Accessed: 2025-05-22. [Online]. Available: https://ijsart.com/public/storage/paper/pdf/IJSART113102769.pdf.

[3] B. D. Baloni, Y. Pathak, and S. Channiwala, *Centrifugal blower volute optimization based on taguchi method*, Unpublished manuscript or image reference, Accessed: 2025-04-01, 2025.

[4] SameSky Devices, *CBM-979433B-125-467 Centrifugal Blower*, https://www.sameskydevices.com/product/thermal-management/dc-fans/centrifugal-blowers/cbm-979433b-125-467, Accessed: 2025-04-01.

[5] U.S. Department of Energy, "Improving fan system performance: A sourcebook for industry," U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy, Tech. Rep., 2003, Accessed May 18, 2025. [Online]. Available: https://docs.nrel.gov/docs/fy03osti/29166.pdf.

[6] S. L. Dixon and C. A. Hall, *Fluid Mechanics and Thermodynamics of Turbomachinery*, 7th. Oxford: Butterworth-Heinemann, 2010, Accessed May 18, 2025, ISBN: 9781856177931. [Online]. Available: https://www.sciencedirect.com/book/9781856177931/fluid-mechanics-and-thermodynamics-of-turbomachinery.

[7] Woods Air Movement, *Power efficiency: How to choose the right fan*, Accessed: 2025-05-18, 2023. [Online]. Available: https://www.woodsairmovement.com/en-gb/news/power-efficiency-how-to-choose-the-right-fan/.

[8] N. Montazerin, G. Akbari, and M. Mahmoodi, "1 - general introduction of forward-curved squirrel-cage fan," in *Developments in Turbomachinery Flow*, N. Montazerin, G. Akbari, and M. Mahmoodi, Eds., Woodhead Publishing, 2015, pp. 1–23, ISBN: 978-1-78242-192-4. DOI: https://doi.org/10.1016/B978-1-78242-192-4.00001-4. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9781782421924000014.

[9] P. F. O. e. a. Dussauge T.P. Sung W., "A reinforcement learning approach to airfoil shape optimization.," *Scientific Reports*, 2023.

[10] Wikimedia Commons, *Reinforcement learning diagram*, Accessed: 2025-05-22, Apr. 2017. [Online]. Available: https://commons.wikimedia.org/w/index.php?curid=57895741.

[11] A. Sebastianelli, M. Tipaldi, S. Ullo, and L. Glielmo, "A deep q-learning based approach applied to the snake game," May 2021. DOI: 10.1109/MED51440.2021.9480232.

[12] P. O. Staff, "The exploration-exploitation dilemma: A multidisciplinary framework.," *PloS one, 10(3), e0119116.*, 2015. DOI: https://doi.org/10.1371/journal.pone.0119116.

[13] Simularge. "Exploring inlet boundary conditions in cfd." Accessed: 2025-05-18. (2023), [Online]. Available: https://www.simularge.com/blog/exploring-inlet-boundary-conditions-in-cfd.