MARMARA UNIVERSITY
F ACU LTY OF EN GIN EERIN G

DESIGN AND ANALYSIS OF HORIZONTAL AXIS WIND TURBINES WITH TOROIDAL ROTOR BLADES INTELLIGENCE

GÖKTUĞ YARAR, ABDULRAGIB ÖZBODUÇ, ŞİMAL BALCIOĞLU

GRADUATION PROJECT REPORT
Department of Mechanical Engineering

Supervisor

Prof. Dr. Emre ALPMAN

ISTANBUL, 2024

MARMARA UNIVERSITY
FACULTY OF ENGINEERING

Design and Analysis of Horizontal Axis Wind Turbines with Toroidal Rotor Blades

by

Göktuğ Yarar , Abdulragıb Özboduç , Şimal Balcıoğlu

June 20, 2024, Istanbul

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF

BACHELOR OF SCIENCE

AT

MARMARA UNIVERSITY

Signature of Author(s) ....

Department of Mechanical Engineering

Certified By ....

Project Supervisor, Department of Mechanical Engineering

Accepted By ....

Head of the Department of Mechanical Engineering

ACKNOWLEDGEMENT

We would like to thank our supervisor Prof. Dr. Emre ALPMAN for the valuable guidance and advice on preparing this thesis and giving us moral.

June, 2024   Göktuğ Yarar , Abdulragıb Özboduç , Şimal Balcıoğlu

TABLE OF CONTENTS

ABSTRACT

This thesis presents the design and analysis of horizontal axis wind turbines (HAWTs) equipped with toroidal rotor blades. The study aims to investigate the aerodynamic performance and structural stability of toroidal blades compared to traditional flat and curved blades. Accordingly, Openscad was utilized for design of wind turbine, while Freecad, Openfoam and Paraview was used for analysis and examining the effects of parameters. The design process involved computational fluid dynamics (CFD) simulations to optimize the blade geometry for maximum efficiency and minimal structural stress. Experimental validation was conducted using a wind tunnel to measure performance metrics such as power output, lift-to-drag ratio, and noise levels. The results indicate that toroidal rotor blades can enhance the efficiency of HAWTs by reducing vortex-induced losses and improving overall aerodynamic performance. Additionally, the unique shape of toroidal blades contributes to lower noise emissions, making them suitable for deployment in noise-sensitive areas. This research contributes to the development of more efficient and environmentally friendly wind energy solutions, highlighting the potential of innovative blade designs in advancing wind turbine technology.

Bu tez, toroidal rotor kanatları ile donatılmış yatay eksenli rüzgar türbinlerinin (HAWT) tasarım ve analizini sunmaktadır. Çalışma, geleneksel düz ve kavisli kanatlara kıyasla toroidal kanatların aerodinamik performansını ve yapısal stabilitesini araştırmayı amaçlamaktadır. Buna göre rüzgar türbininin tasarımında Openscad, parametrelerin analizi ve etkilerinin incelenmesinde ise Freecad ve Openfoam kullanılmıştır. Tasarım süreci, kanat geometrisini maksimum verimlilik ve minimum yapısal stres için optimize etmek amacıyla hesaplamalı akışkanlar dinamiği (CFD) simülasyonlarını içermektedir. Güç çıkışı, kaldırma-sürükleme oranı ve gürültü seviyeleri gibi performans ölçütlerini ölçmek için bir rüzgar tüneli kullanılarak deneysel doğrulama yapılmıştır. Sonuçlar, toroidal rotor kanatlarının vorteks kaynaklı kayıpları azaltarak ve genel aerodinamik performansı iyileştirerek HAWT'lerin verimliliğini artırabileceğini göstermektedir. Ayrıca, toroidal kanatların benzersiz şekli, daha düşük gürültü emisyonlarına katkıda bulunarak, gürültüye duyarlı alanlarda konuşlandırılmaları için uygun hale getirmektedir. Bu araştırma, daha verimli ve çevre dostu rüzgar enerjisi çözümlerinin geliştirilmesine katkıda bulunmakta ve yenilikçi kanat tasarımlarının rüzgar türbini teknolojisini ilerletme potansiyelini vurgulamaktadır.

ABBREVIATIONS

HAWT   : Horizontal Axis Wind Turbine

3D   : Three Dimensions

CFD   : Computational Fluid Dynamics

MIT   : Massachusetts Institute of Technology

OPENFOAM   : Open-Source Field Operation and Manipulation

NREL   : National Renewable Energy Laboratory

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Wind energy has emerged as a crucial component of the global renewable energy portfolio, driven by the need to reduce greenhouse gas emissions and reliance on fossil fuels. Among the various wind turbine configurations, Horizontal Axis Wind Turbines (HAWTs) are the most widely used due to their high efficiency and maturity in technology. However, the quest for enhancing the performance and reducing the environmental impact of HAWTs continues to be a significant area of research.

The design of rotor blades plays a pivotal role in the overall efficiency of wind turbines. Traditional rotor blades, while effective, encounter limitations such as vortex-induced energy losses and noise emissions. These challenges have led researchers to explore innovative blade designs that can mitigate these issues. One such promising design is the toroidal rotor blade, which introduces a unique geometry aimed at enhancing aerodynamic performance and reducing noise levels.

Toroidal rotor blades, characterized by their doughnut-like shape, offer potential advantages over conventional flat and curved blades. The design aims to minimize vortex shedding and energy dissipation, thereby improving the efficiency of the wind turbine. Additionally, the shape of toroidal blades can contribute to lower noise emissions, which is a significant benefit for wind farms located near residential areas.

This thesis focuses on the design and analysis of HAWTs equipped with toroidal rotor blades. The primary objective is to evaluate the aerodynamic performance and structural stability of these innovative blades compared to traditional designs. The research methodology involves computational fluid dynamics (CFD) simulations to optimize the blade geometry and assess various performance metrics such as power output, lift-to-drag ratio, and noise levels. Experimental validation is conducted through wind tunnel testing to verify the simulation results and provide real-world performance data.

The significance of this research lies in its potential to contribute to the development of more efficient and environmentally friendly wind energy solutions. By reducing vortex-induced losses and noise emissions, toroidal rotor blades can enhance the viability of wind energy, especially in areas with strict noise regulations. Furthermore, the findings of this study can inform future designs and innovations in wind turbine technology, promoting the adoption of renewable energy sources on a broader scale.

In conclusion, this thesis aims to advance the understanding and application of toroidal rotor blades in horizontal axis wind turbines. By leveraging advanced simulation techniques and experimental validation, this research seeks to demonstrate the feasibility and benefits of this innovative blade design, contributing to the ongoing efforts to enhance the efficiency and sustainability of wind energy systems.

1.1 Purpose of This Thesis

The main objectives of this graduate project are to design and analyze Horizontal Axis Wind Turbines (HAWTs) equipped with toroidal rotor blades. This study aims to enhance the aerodynamic efficiency, structural stability, and noise reduction capabilities of wind turbines through the innovative application of toroidal blades.

A critical aspect of the thesis is to perform a comprehensive structural analysis of toroidal rotor blades under various operational conditions. This involves assessing the stress distribution and potential deformation characteristics to ensure the blades can withstand the mechanical loads they encounter. Furthermore, the project aims to investigate the noise emission levels of toroidal blades, hypothesizing that their shape will contribute to lower noise levels, thereby making them suitable for installation in noise-sensitive areas.

By achieving these objectives, this thesis seeks to push the boundaries of current wind turbine design, enhance the efficiency and environmental compatibility of wind energy systems, and provide valuable knowledge to the field of renewable energy engineering. The findings from this research are expected to influence future innovations and applications in wind energy technology, promoting its development as a key component of the global energy mix.

2. THEORETICAL BACKGROUND

2.1. What is Air Flow?

Airflow refers to the movement of air molecules from one location to another, driven by differences in pressure and temperature. It is a fundamental concept in fluid dynamics and plays a critical role in various natural and engineered systems, including weather patterns, heating and cooling systems, and aerodynamic applications like wind turbines. Understanding airflow is essential for optimizing the design and efficiency of wind turbines, as it directly impacts their performance and energy output.

2.1.1. Types of Air Flow

Airflow can be classified into several types based on its characteristics and behavior. The primary types are laminar flow, turbulent flow, and transitional flow. Each type has distinct effects on the performance of wind turbines.

2.1.1.1. Laminar Flow

Laminar flow occurs when air moves in smooth, parallel layers with minimal mixing between them. In this type of flow, the air molecules follow well-defined paths, and the flow is characterized by low velocity and high viscosity. The Reynolds number, a dimensionless quantity used to predict flow patterns, is typically low for laminar flow.

Laminar flow is generally desirable near the leading edge of the wind turbine blades because it results in smooth and predictable aerodynamic forces. According to Anderson (2010), "Laminar flow reduces skin friction drag and can enhance the efficiency of the blades by maintaining a steady aerodynamic force" (p. 87). However, maintaining laminar flow over the entire blade is challenging, especially at higher wind speeds.

2.1.1.2 Turbulent Flow

Turbulent flow is characterized by chaotic, irregular motion of air molecules, leading to significant mixing and eddies. This type of flow occurs at high velocities and low viscosity, resulting in a high Reynolds number. Turbulent flow is more complex to predict and analyze compared to laminar flow.

While turbulent flow can increase drag and cause energy losses, it also has some benefits. Turbulent flow can enhance the mixing of air, potentially improving the performance of the rotor blades at certain operating conditions. According to Pope (2000), "Turbulence increases the momentum transfer and can delay flow separation on the blades, thereby maintaining lift at higher angles of attack" (p. 45). However, excessive turbulence can lead to increased noise and mechanical stress on the turbine components.

2.1.1.3 Transitional Flow

Transitional flow is a state between laminar and turbulent flow. It occurs when the airflow begins to transition from smooth, orderly motion to chaotic, irregular motion. This transition typically happens at intermediate Reynolds numbers.

The transitional flow region is critical for wind turbine performance because it marks the onset of turbulence. According to Batchelor (2000), "Understanding the transitional flow is essential for designing blade profiles that can manage the shift from laminar to turbulent flow effectively" (p. 112). Properly managing this transition can help optimize the aerodynamic performance and reduce the noise generated by the turbine.

2.2. What is Wind Turbine?

A wind turbine is a device that converts the kinetic energy of wind into electrical energy. Wind turbines are a crucial component of renewable energy systems, playing a significant role in reducing dependency on fossil fuels and decreasing greenhouse gas emissions. According to Manwell, McGowan, and Rogers (2009), "Wind turbines capture wind energy and convert it into mechanical energy, which can then be used directly or converted into electricity" (p. 2). The basic components of a wind turbine include the rotor blades, nacelle, tower, and the generator. The rotor blades capture the wind's energy, the nacelle houses the key components such as the gearbox and generator, the tower supports the structure at an optimal height, and the generator converts the mechanical energy into electrical energy.

2.2.1. Wind Turbine Types

There are two types of wind turbines.

2.2.1.1. Vertical Axis Wind Turbine

This type of turbine utilizes drag or lift or a combination of the two to operate. VAWT has also been used for ages and in fact, the first windmills that people have ever known are VAWT before HAWT appears and becomes popular at some points in the history of wind turbine. There are generally two main designs of VAWT and both designs work on different principles. The first design is Savonius that uses drag forces to work just like a water wheel and the other design is Darrieus that uses aerodynamic blade to generate lift and turn the turbine.

Although VAWT has not been given as much attention as HAWT in its research and development, it has several significant advantages compared to HAWT. Unlike HAWT that is required to face the wind stream all the time in order to give the optimum output, VAWT is omnidirectional and can receive wind from any direction. VAWT is the best choice to be installed in the slow and more turbulent wind environment such as urban areas because it can generally start to produce power at such low wind speed. The system for VAWT such as gearbox and other equipment can be packed together and installed closer to the ground, hence eliminating the need for extra cost for maintenance and making it easier to be controlled. Finally, the VAWT are quieter than HAWT too. However, the disadvantage of VAWT also cannot be ignored easily. VAWT is inefficient in high speed wind environment because it has very low starting torques and issues on its dynamic stability. VAWT is also vulnerable to backtracking because its blade moves in the same direction to the wind and thus the blades need to travel back into the wind flow before being pushed back around. It is important to note that previous studies comparing VAWT and HAWT have shown mixed results. There can either be that there is no significant difference between them or one is simply better than the other.
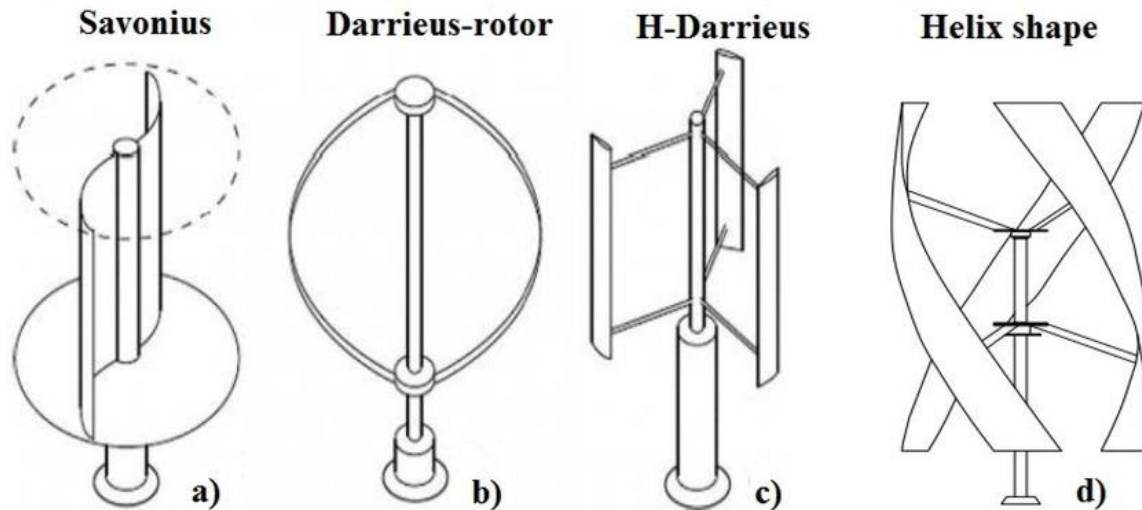
Figure2.1. Vertical Axis Wind Turbines

2.2.1.1. Horizontal Axis Wind Turbine

The wind turbines have gone through significant innovation and improvisation in their design for optimum performance. HAWT consists of blades that extract wind energy on horizontal axis and are parallel to the ground. By facing the wind flow perpendicularly, the blades work and turn due to aerodynamic lift. HAWT is the most popular choice of wind turbine and has received more funding for research and development since it offers significant advantage over VAWT. HAWT have a greater efficiency then VAWT when extracting energy from the wind force due to its design that allows it to extract the energy through the full rotation of the blades when placed under consistent wind flow. It is also immune to backtracking effect.

However, HAWT has a major disadvantage, which is the fact that it must always be pointed in the wind direction to work efficiently. With unpredictable wind direction, extra mechanism is required to make sure the blades will always be facing the wind direction to extract maximum power output. Small wind turbine usually uses a simple wind vane to position itself into the direction of the wind stream. For larger wind turbine, it consists of a yaw meter to determine the correct position of the wind flow and a yaw motor to position the turbine into accurate direction of the wind. Because of this disadvantage, HAWT works excellently in environment with consistent and low turbulence wind as it does not need to change its orientation too frequent.

Figure2.2. Horizontal Axis Wind Turbine

https://images.app.goo.gl/6kUzNT1P34h7aGBs5

2.2.2 Wind Turbine History

History of wind turbines is a testament to human ingenuity and the quest for harnessing renewable energy. The development of wind turbine technology has evolved significantly from its early beginnings to the sophisticated machines we see today.

Early Beginnings

The concept of using wind power dates back to ancient civilizations. The earliest known use of windmills was in Persia (modern-day Iran) around 500-900 AD. These early windmills were primarily used for grinding grain and pumping water. They featured vertical-axis designs with sails made from bundles of reeds or wood, which rotated around a central vertical shaft.

Development in Europe

Windmill technology spread to Europe in the Middle Ages, where it underwent significant improvements. By the 12th century, horizontal-axis windmills began appearing in England, France, and the Netherlands. These windmills had a more efficient design and were primarily used for milling grain and draining water from low-lying land, particularly in the Netherlands. The European windmills typically had four blades and a post mill or tower mill structure, which allowed for better wind capture and structural stability.

The Advent of Modern Wind Turbines

The transition from traditional windmills to modern wind turbines began in the late 19th and early 20th centuries. One of the earliest recorded attempts to generate electricity from wind was by Professor James Blyth of Anderson's College, Glasgow (now Strathclyde University), in 1887. Blyth's wind turbine powered his holiday cottage in Marykirk, Scotland, making it one of the first houses in the world to have electricity.

Around the same time, American inventor Charles F. Brush built a large wind turbine in Cleveland, Ohio, in 1888. This turbine had a rotor diameter of 17 meters and 144 blades made from cedar wood. It produced about 12 kW of power, which was used to charge batteries in Brush's mansion, marking one of the earliest uses of wind-generated electricity in the United States.

Mid-20th Century Developments

The oil crises of the 1970s spurred renewed interest in wind energy as a clean and sustainable alternative to fossil fuels. During this period, significant advancements were made in turbine design, materials, and efficiency. The Danish wind turbine industry began to take shape, leading to Denmark becoming a world leader in wind energy. By the 1980s, the first large-scale wind farms were being developed in California, supported by federal and state incentives.

Modern Era and Technological Advances

In recent decades, wind turbine technology has seen exponential growth and innovation. Modern wind turbines are characterized by their large rotor diameters, advanced materials, and sophisticated control systems. Innovations such as variable-speed turbines, direct-drive generators, and offshore wind farms have significantly increased the capacity and efficiency of wind turbines.

According to Manwell, McGowan, and Rogers (2009), "The development of wind energy technology has progressed rapidly over the past few decades, driven by advances in aerodynamics, materials science, and power electronics" (p. 45). This progress has resulted in wind turbines becoming a cornerstone of renewable energy strategies worldwide.

The history of wind turbines is a journey of continuous improvement and adaptation. From ancient windmills to the advanced wind turbines of today, the technology has evolved to meet the growing demand for sustainable energy. The ongoing research and development efforts in the field promise to further enhance the efficiency and viability of wind energy in the future.
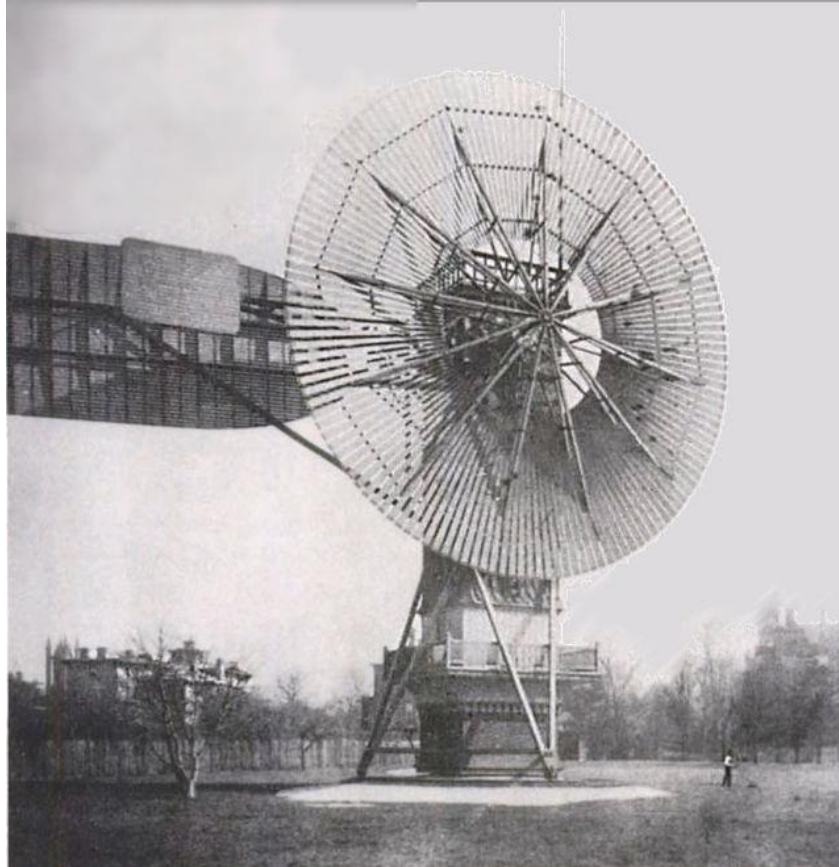

Figure 2.3. First Wind Turbine in History

https://journals.sagepub.com/cms/10.1177/0309524X221117825/asset/images/large/10.1177_03 09524x221117825-fig2.jpeg

2.2.3. Working Principle of Wind Turbines

A wind turbine converts the kinetic energy of wind into electrical energy through a process involving aerodynamic, mechanical, and electrical components. The working principle of a wind turbine is rooted in the laws of physics, particularly the principles of fluid dynamics and electromagnetism. According to Manwell, McGowan, and Rogers (2009), "The energy in the wind is converted into mechanical energy by the rotor, which is then converted into electrical energy by the generator" (p. 67).

Aerodynamic Conversion

The process begins with the rotor blades, which are aerodynamically designed to capture the wind's kinetic energy. The rotor blades function similarly to an airplane wing or an airfoil. When wind flows over the blades, it creates a pressure differential. The pressure on the front side of the blade (the windward side) is higher than the pressure on the back side (the leeward side). This pressure difference generates lift, causing the rotor to spin. This aerodynamic principle is known as the Bernoulli effect.

Mechanical Conversion

The spinning rotor is connected to a main shaft that transfers the rotational energy to the generator. In most wind turbines, the rotational speed of the rotor is relatively slow and needs to be increased to a speed suitable for generating electricity. This is typically achieved using a gearbox, which is a series of gears that step up the rotational speed. However, some modern wind turbines use direct-drive systems that eliminate the need for a gearbox, thereby reducing mechanical losses and maintenance requirements.

Electrical Conversion

The generator, located in the nacelle (the housing at the top of the tower), converts the mechanical energy from the rotor into electrical energy. Most wind turbines use an induction or synchronous generator for this purpose. The generator works on the principle of electromagnetic induction, where the rotating shaft spins a set of magnets around a coil of wire (or vice versa), inducing an electric current in the wire. According to Heier (2014), "The electrical output of the generator is then conditioned and converted to match the frequency and voltage of the grid" (p. 143).

Power Control Systems

Modern wind turbines are equipped with sophisticated control systems to optimize performance and ensure safe operation. These systems regulate the rotor speed, blade pitch, and yaw. The blade pitch control system adjusts the angle of the blades to maintain optimal lift and drag conditions, maximizing energy capture and protecting the turbine from excessive wind speeds. The yaw control system ensures that the rotor faces the wind by rotating the nacelle on the tower. This is essential for maximizing efficiency as wind direction can vary.

Grid Integration

Once the electrical energy is generated, it is converted to the appropriate voltage and frequency to be compatible with the local power grid. This involves power electronics such as inverters and transformers. The electricity is then transmitted via power lines for distribution to homes and businesses. Grid integration also requires the management of variability in wind energy, which is addressed by using energy storage systems and grid management techniques.

In summary, the working principle of a wind turbine involves converting wind energy into mechanical energy via aerodynamic rotor blades and then into electrical energy through a generator. Advanced control systems and power electronics ensure efficient operation and integration into the power grid. This process is well-documented in various sources, including Manwell et al. (2009) and Heier (2014), which provide detailed insights into the mechanisms and technologies involved in wind energy conversion.

Figure 2.4. Wind Turbine
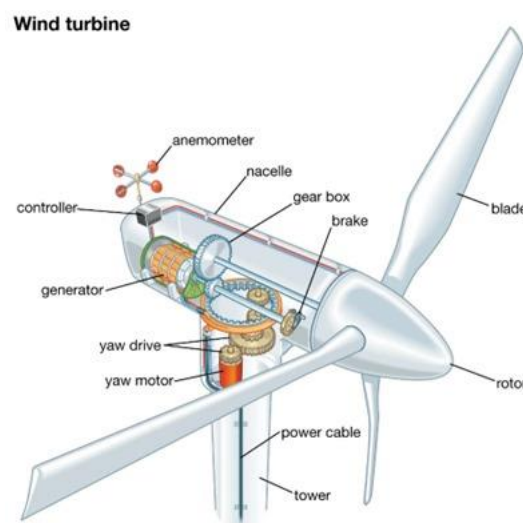
https://www.britannica.com/technology/wind-turbine

3. DESIGN AND PROCESS

In this section, the detailed examination and determination process of parameters used to create toroidal propeller and design of wind turbine are addressed. Subsequently, forces that toroidal propeller has produced are obtained in order to obtain mesh element size. Following that, mesh independency test and NREL consistency test are explained. Lastly, the comparison between different parameters is examined.
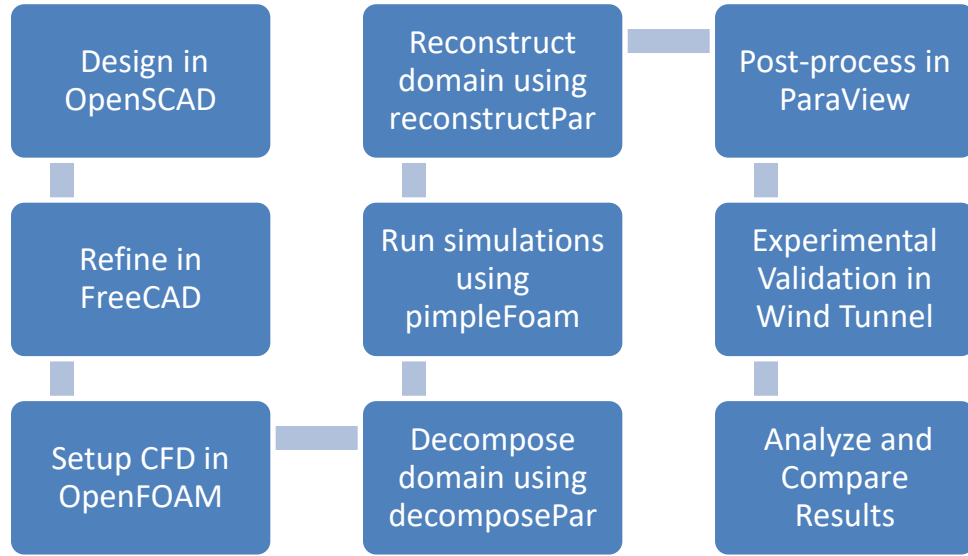
Figure 3.1. Process Flowchart

3.1. Toroidal Propeller Design

Openscad is used to create toroidal propeller with given parameters.
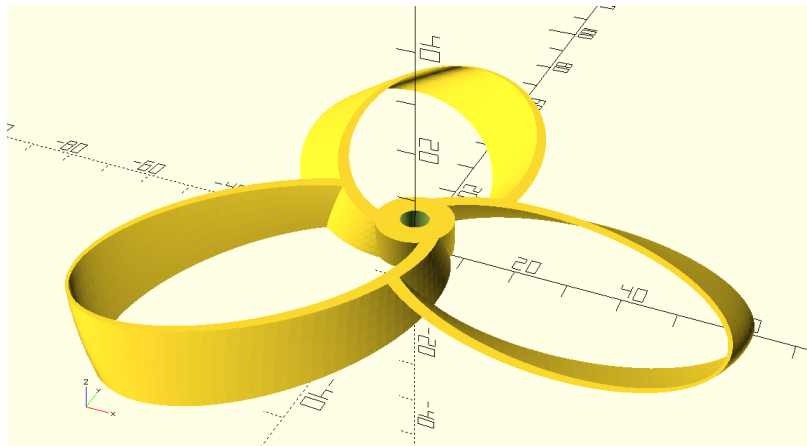
Figure 3.2. 42 mm blade width- 15° twist angle- 3 blades toroidal propeller

3.2. Physics Model Design

3.2.1. Time

There are two types of time as steady-state time and transient time. Steady-state time refers to a condition in fluid dynamics where the flow variables, such as velocity, pressure, and density, remain constant over time. In our physics model, we opted for steady-state analysis due to the simplicity of the part's geometry and the nature of the fluid, which is air—a non-complex fluid. This approach allows for faster and more accurate calculations, as it eliminates the need to account for changes over time, providing a stable and consistent flow analysis suitable for our project's requirements.

Transient time analysis, in contrast, is used to study fluid flows and pressure distributions that vary over time. It is essential for scenarios involving accelerating, decelerating, or direction-changing flows, as well as time-varying pressure distributions like pressure waves in liquids, and flows with complex geometries. Transient analysis simulates these dynamic changes, providing insights into the fluid's behavior under varying conditions. However, since the project did not involve such time-dependent variations, transient time has not chosen in this project.

3.2.2. Flow

There are four types of flow in physics model as single-phase flow, multi-phase free-surface flow, viscous flow and isothermal flow.

Single phase flow refers to a fluid flow consisting of a single substance (liquid, gas or plasma). The fluid is homogeneous throughout its properties, i.e. density, viscosity and other properties are the same throughout the fluid. For example, water flowing through a pipe, air movement in a ventilation system, air flow over an airfoil.

Multiphase-free surface flow refers to fluid flow consisting of a mixture of two or more phases (e.g. liquid-gas, liquid-solid). In these flows, the phases are separated and have interfaces. Free-surface is the surface where one phase of the fluid comes into contact with another phase (usually air) or the environment. For example, foam flow, spilled liquid, magma erupting in a fire.

Viscous flow is a modeling method that can provide a more realistic representation of fluid flow and determine how the viscosity of the fluid affects the overall behavior of the fluid flow.

Isothermal flow is a type of fluid flow in which the temperature of the fluid remains constant, even if the fluid's parameters such as flow rate and pressure change. This type of flow is often used when the temperature of the fluid does not significantly affect the overall behavior of the fluid flow.

In this project, single phase flow, isothermal flow and viscous flow are used. In this project, flow is a simple and homogeneous flow. At the same time, it is not a suitable analysis method to use Steady-state time and multiphase-free surface flow method together. (The program does not allow access anyway). These are the reasons for using single-phase flow. Besides that, because of the general behavior of the fluid, that is, the flow velocity is not at very high values and the temperature is not variable, isothermal flow is used. Lastly, since how much the viscosity of the fluid flow affects the overall behavior of the fluid flow is needed to analyze, viscous flow is also used in this project.

3.2.3. Turbulence

There are two parameters that can be selected in turbulence part. Laminar and RANS method.

Reynolds Average Navier-Stokes (RANS) is a widely utilized computational fluid dynamics (CFD) method designed to model turbulent flows by focusing on the time-averaged properties of fluid flow, thereby significantly reducing computational costs. This method is extensively employed in various engineering applications, including the design and analysis of airplane wings, engines, pumps, and buildings. By averaging out small-scale fluctuations and concentrating on the larger-scale average flow characteristics, RANS provides an efficient approach for handling complex geometries and large flow fields. However, it is important to note that RANS is not suitable for scenarios that demand high accuracy, involve complex flow fields or high turbulence levels, require a detailed understanding of turbulence effects, or necessitate direct modeling of the physical mechanisms of fluid flow. Instead, RANS is ideal for problems where computational cost constraints exist, computational resources are limited, or fast and approximate solutions are needed. Given that the conditions we are monitoring do not involve complex flow fields or high turbulence values, we have opted to use the RANS theorem to balance efficiency and accuracy in our analysis.

Laminar Flow is the smooth and regular flow of a liquid. It is more common at low speeds and with high viscosity liquids. Laminar flow has some advantages, such as low friction and less mixing, but it also has some disadvantages, such as low speed limit and limited range of application. Whether laminar flow should be used depends on the specific requirements of the problem.

In some turbulence models of RANS, laminar flow can also be treated as a boundary condition. In these models, the properties of laminar flow at low Reynolds numbers are incorporated into the RANS equations. Since the viscosity of our fluid is not high, the RANS method is selected instead of laminar flow.

3.3. Flow Field Design

3.3.1. Velocity

In the velocity section, there are three options as potential flow, use values from boundary and specify values.

The potential flow model is a very useful tool for analyzing flow problems with simple geometries and low velocities. Potential flow lines (lines proportional to the velocity of the fluid) help to visually understand fluid flow. Potential flow equations are simpler compared to other fluid flow models, making them faster and less costly to solve.

The USE values from boundary option assumes that the velocity of the fluid has specific values at boundary conditions (for example, at walls or inlets).

The Specify values option assumes that the fluid velocity has specific values at each point.

Selecting potential flow as velocity indicates that the velocity of the fluid will be calculated using a potential function. This allows you to take advantage of the advantages of the potential flow model mentioned above. For this reason, potential flow method is used.

3.3.2. Pressure

Using potential flow as pressure is a method of calculating the pressure of a fluid. This is a very useful tool for flow problems with simple geometries and low velocities. The reason for choosing potential flow as pressure is that it is a fast, inexpensive and intuitive method of calculating fluid pressure in flow problems with simple geometries and low velocities.

3.4. Boundary Conditions

3.4.1. Inlet Condition

There are five sub-types that can be used can be employed depending on the specific characteristics of the flow and the boundary conditions as uniform velocity, volumetric flow rate, mass flow rate, static pressure and total pressure.

The uniform velocity inlet condition is used when the flow has a known velocity at the inlet, setting a constant vector value of the fluid velocity at the boundary. This method is useful for describing the basic characteristics of the flow but is not recommended for complex flows or those with significant boundary effects.

The volumetric flow rate condition determines the volume of fluid passing through the boundary per unit time and is suitable for flows in fixed areas such as pipelines or ducts, though it may not provide information about velocity distribution or fluid density.

The mass flow rate condition allows direct control over the total mass flow, making it preferred in closed systems.

The static pressure condition is used when the flow is open to the atmosphere or when the ambient pressure is known and relatively constant; however, it may not be suitable if the ambient pressure varies significantly or if there are considerable interactions between the flow and the environment.

The total pressure condition is used when assuming a lack of heat or frictionless flow, and it can describe the basic properties of compressive flows, such as the flow around airplane wings.

Generally, selecting uniform velocity in inlet condition is an important factor and since inlet velocity in this project is constant, a uniform velocity flow is selected as inlet condition values. The velocity of flow is -20 mm/s in the Z direction under 1 atm pressure.

3.4.2. Outlet Condition

If the flow is open to the atmosphere or the ambient pressure is known and relatively constant, static pressure may be a suitable choice. Uniform velocity can be used if the flow has a known velocity upstream or downstream. Extrapolated boundary conditions can be used where the flow does not create a significant gradient at the boundary or where the boundary conditions have no significant effect on the flow field.

For the outlet boundary condition our flow is chosen to be static pressure and the outlet to is in the -Z direction.

3.4.3. Open Condition

In this analysis, the right, left, and top boundary conditions is selected as open and implemented the ambient pressure model. The ambient pressure boundary condition sets the pressure at the boundary to a constant value, typically representing the pressure in the surrounding environment. This approach is particularly useful when the flow is open to the atmosphere or when the ambient pressure is known and relatively constant, providing a simple and intuitive method for a wide range of flow problems. Additionally, the far-field boundary condition, which assumes that the flow at the boundary is undisturbed by the presence of the computational domain. This condition applies a specific flow velocity and/or pressure profile to represent far-field conditions and is often used when the computational domain is sufficiently large to capture the essential properties of the flow. Far-field boundary conditions are advantageous for accurately representing undisturbed flow conditions and are suitable for situations where the flow is not open to the atmosphere or where there are significant interactions between the flow and the environment. By selecting the appropriate boundary conditions, accurate and reliable simulation results for our wind turbine analysis is obtained.

If the flow is open to the atmosphere or the ambient pressure is known and relatively constant, the ambient pressure may be a suitable choice. If the flow is not open to the atmosphere or if there are significant interactions between the flow and the environment, the far field may be more appropriate. Ambient Pressure is the needed choice for this float.

3.4.4. Ground Condition

There are five options for ground conditions as no-slip (viscous) condition, slip (inviscid) condition, partial slip condition, translating condition and rough condition.

The no-slip (viscous) boundary condition assumes that fluid particles adhere to the surface of the wall, resulting in zero slip speed. This condition is used to model realistic fluid-wall interactions, such as those found in viscous flow problems, boundary layer flows, and turbulent flow problems. The no-slip condition accurately captures these interactions, making it suitable for a wide range of flow problems. However, it can be computationally more expensive and may present difficulties in convergence for complex flow problems

On the other hand, the slip (inviscid) boundary condition assumes that fluid particles slide slightly off the surface of the wall, implying zero shear stress on the wall. This condition is applied when viscous effects are negligible, such as in inviscid flow problems. The slip condition is computationally faster and less expensive, and it is easier to achieve convergence in complex flow problems. However, it does not fully capture realistic fluid-wall interactions and is not suitable for modeling boundary layer flows or turbulent flow problems. Choosing the appropriate boundary condition depends on the specific requirements and characteristics of the flow being simulated.

The partial slip boundary condition assumes that fluid particles partially slip off the surface of the wall, implying shear stress on the wall. This condition models a combination of viscous and inviscid flow effects, offering a balance between no-slip and slip conditions. It can provide more accurate results in certain flow problems but is more complex to implement and interpret, with the selection of appropriate parameters being particularly challenging.

The translating boundary condition assumes that the wall moves with the fluid, making it suitable for flow problems involving moving walls, such as rotating vanes or walls. This condition allows for accurate modeling of fluid interactions with moving walls but is also more complex to implement and interpret.

The rough boundary condition assumes that the wall's roughness affects the flow, used specifically for problems involving rough walls. These specialized conditions provide nuanced control over simulations, enhancing the accuracy of modeling various real-world scenarios.

The boundary condition that is set as ground, is the bottom surface. In this section, condition is selected as wall and no-slip condition. Since having a turbulent flow, these conditions are appropriate for this project.

3.4.5. Rotor Condition

The boundary condition option referred to as "Rotor" represents the surface of the toroidal propeller that is exposed to the flow. Since the surface of the propeller is in motion, the translating sub-type is selected. This choice is necessary to analyze the interaction between the moving surface and the flow.

3.5. Parameters

3.5.1. Number of Blades

In this thesis project on horizontal axis wind turbines with toroidal rotor blades, the number of blades is a crucial parameter impacting performance. The blade count affects aerodynamic efficiency, structural integrity, noise levels, and cost. More blades can capture more wind energy and provide better load distribution, enhancing stability and reducing noise, but they also increase manufacturing and maintenance costs. Optimizing the number of blades is essential to balance performance with cost, leveraging the unique advantages of toroidal design through detailed analysis and validation. In this project, 3 blades are used. In order to make comparison, 2 and 4 blades are also selected.
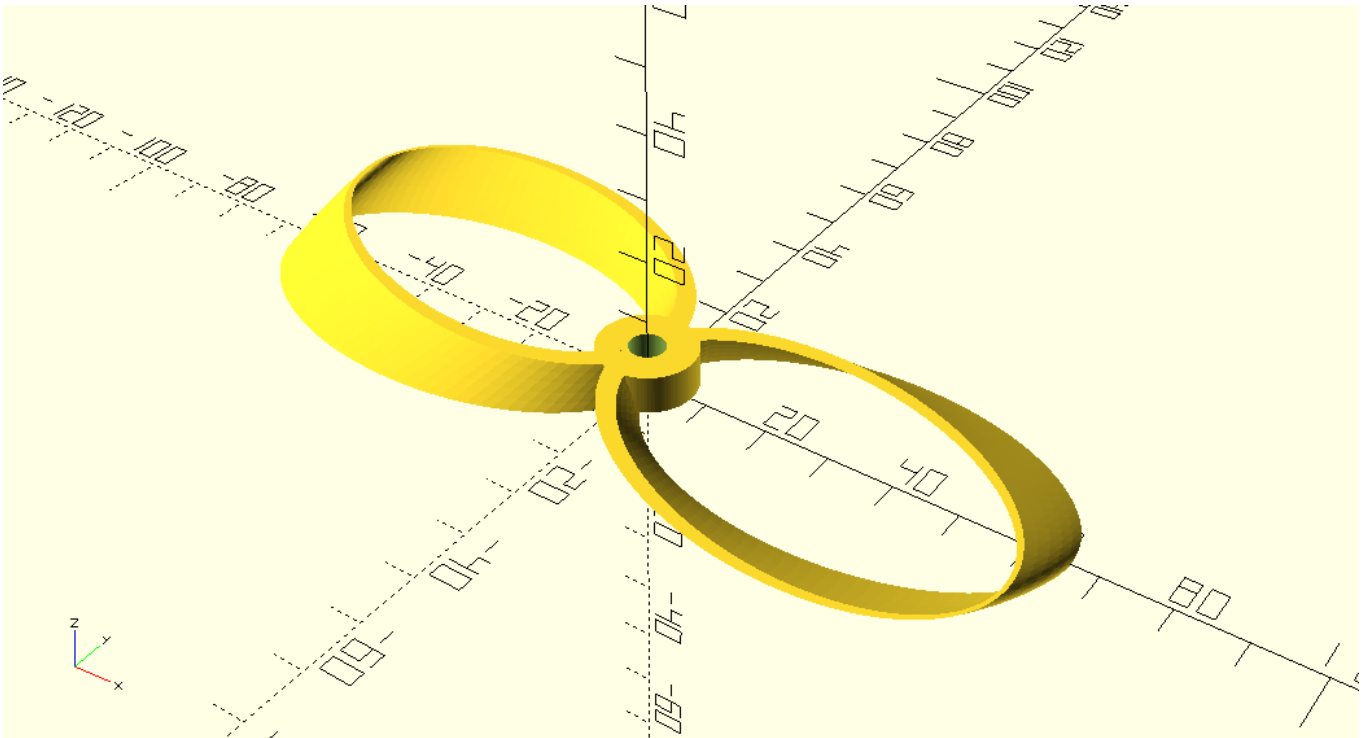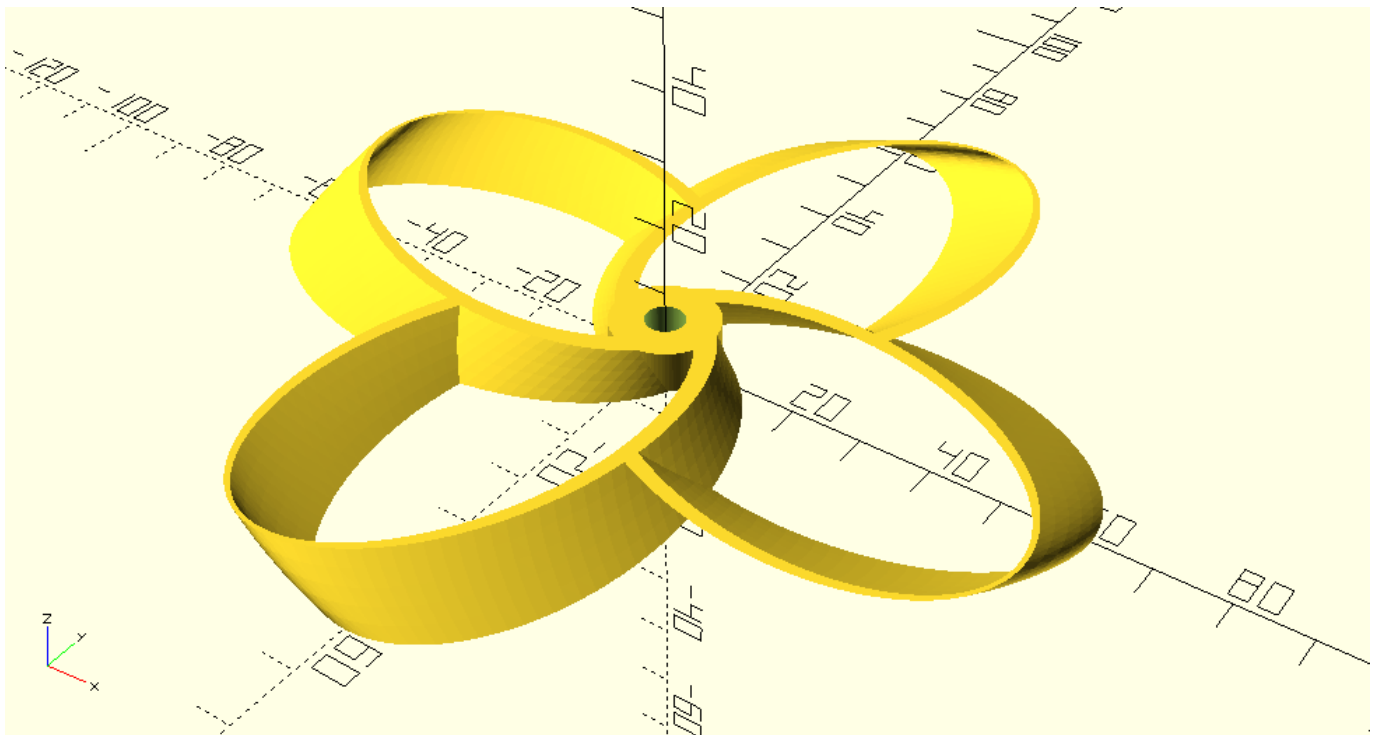

Figure3.3. 2 blades air propeller

Figure 3.4. 4 blades air propeller
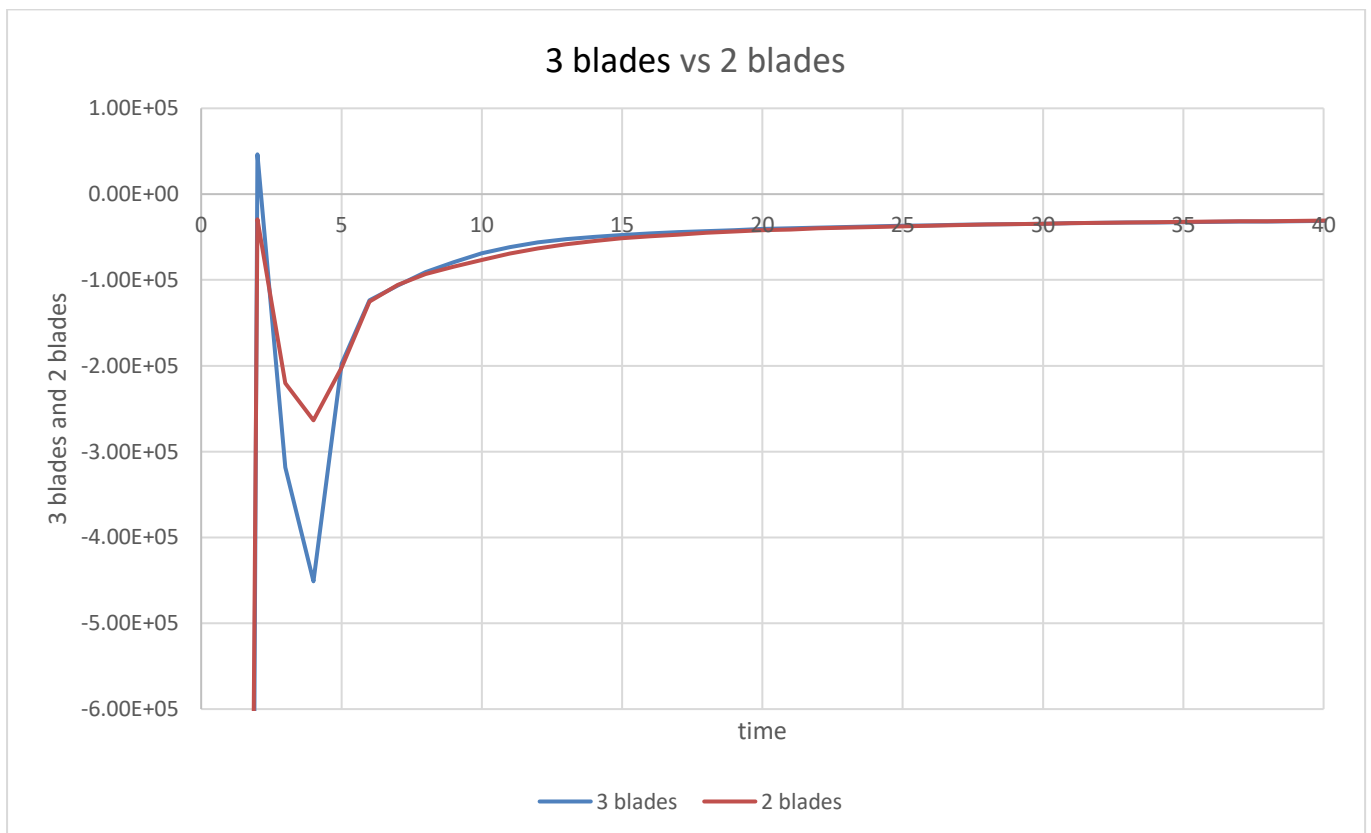
Figure 3.5. 3 blades vs 4 blades

20
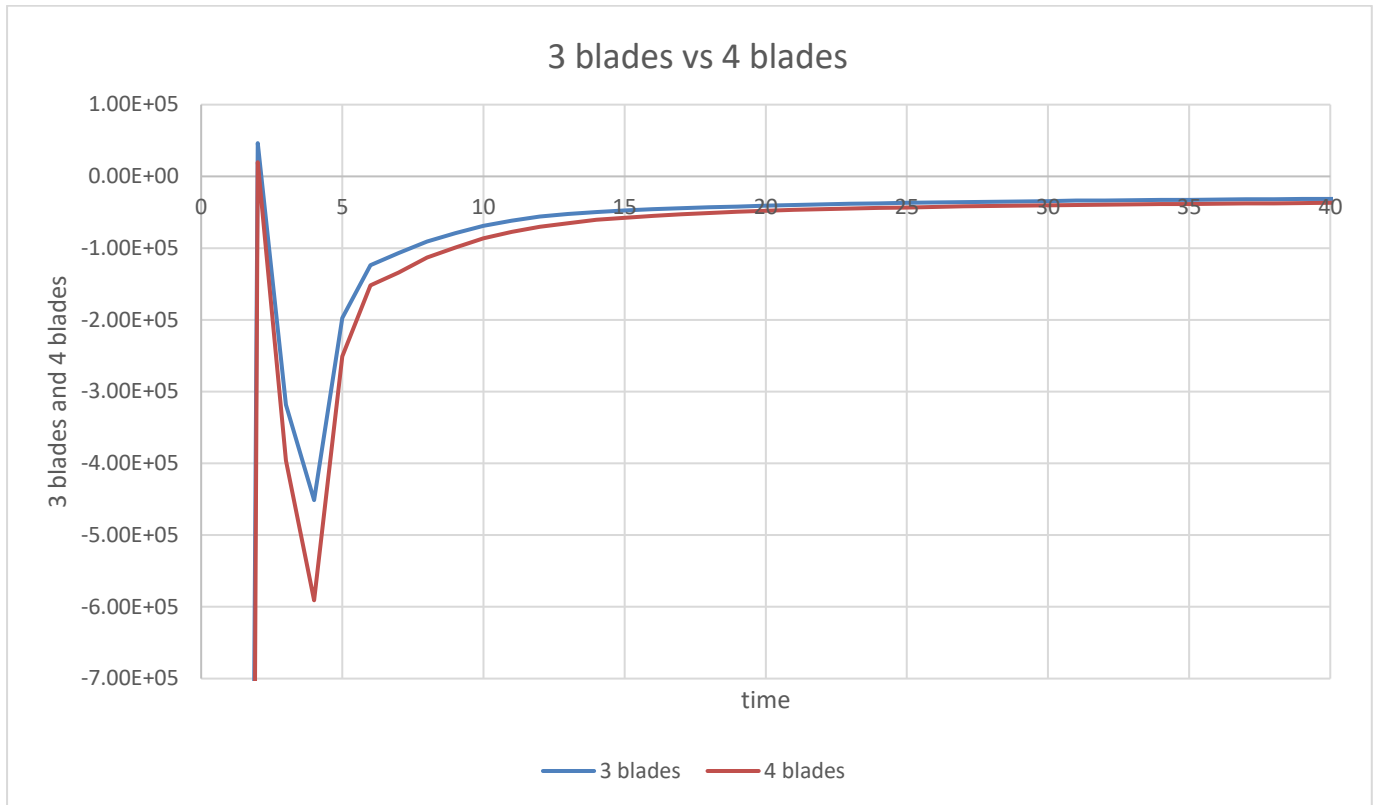
Figure 3.6. blades vs 4 blades

3.5.2. Blade Offset

The blade offset parameter in the context of toroidal blades for wind turbines refers to the distance between the axis of rotation (hub center) and the centerline of the blade's cross-section. This offset plays a crucial role in determining the aerodynamic performance and structural integrity of the turbine blades. A carefully chosen offset helps in achieving optimal aerodynamic efficiency by controlling the angle of attack and the distribution of lift along the span of the blade. Additionally, it influences the structural loading and bending moments experienced by the blade during operation. Too large an offset can increase aerodynamic drag and structural stresses, while too small an offset may lead to inefficient airflow and reduced performance. Engineers meticulously design and analyze this parameter to balance aerodynamic efficiency, structural robustness, and overall turbine performance. The blade offset is -6 mm in this project. In order to make comparison, 0 mm blade offset is also selected.
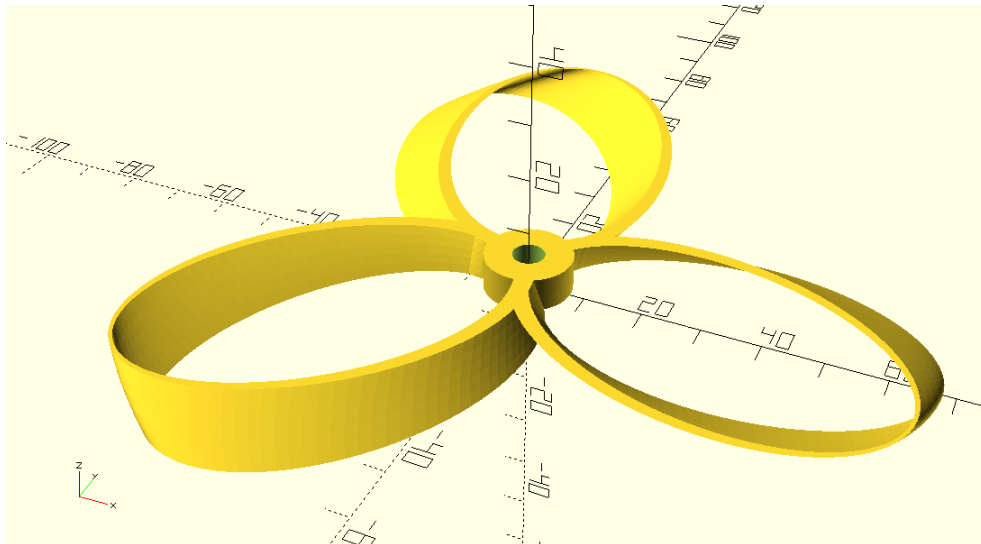
Figure 3.7. 0 mm blade offset toroidal propeller

Forces provided -6 mm blade offset and 0 mm blade offset is given as a graph.
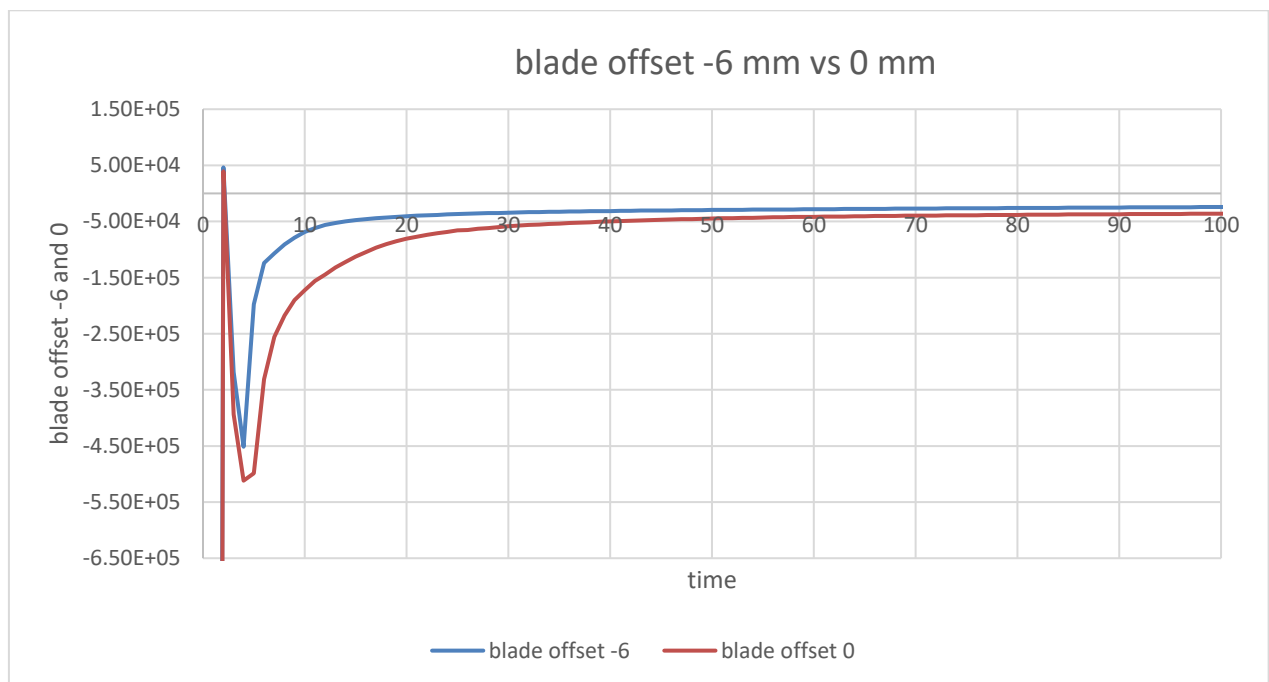
Figure 3.8. -6 mm blade offset vs 0 mm blade offset

3.5.3. Blade Width

In this thesis project on horizontal axis wind turbines with toroidal rotor blades, blade width is a key parameter that significantly affects performance. The width of the blades influences the lift and drag forces, impacting the turbine's aerodynamic efficiency. Wider blades can capture more wind energy, improving power output, especially at lower wind speeds. However, increasing blade width also raises material costs and structural complexity, potentially causing higher aerodynamic drag and noise. Optimizing blade width is essential to achieve a balance between maximizing energy capture and maintaining structural integrity and cost-effectiveness, ensuring the overall efficiency and effectiveness of the wind turbine design. In this project, 42 mm blade twist is used. In order to make comparison 30mm and 84 mm blade twist is used. Forces between 42 mm, 30 mm and 84 mm blade twist is given as graph.
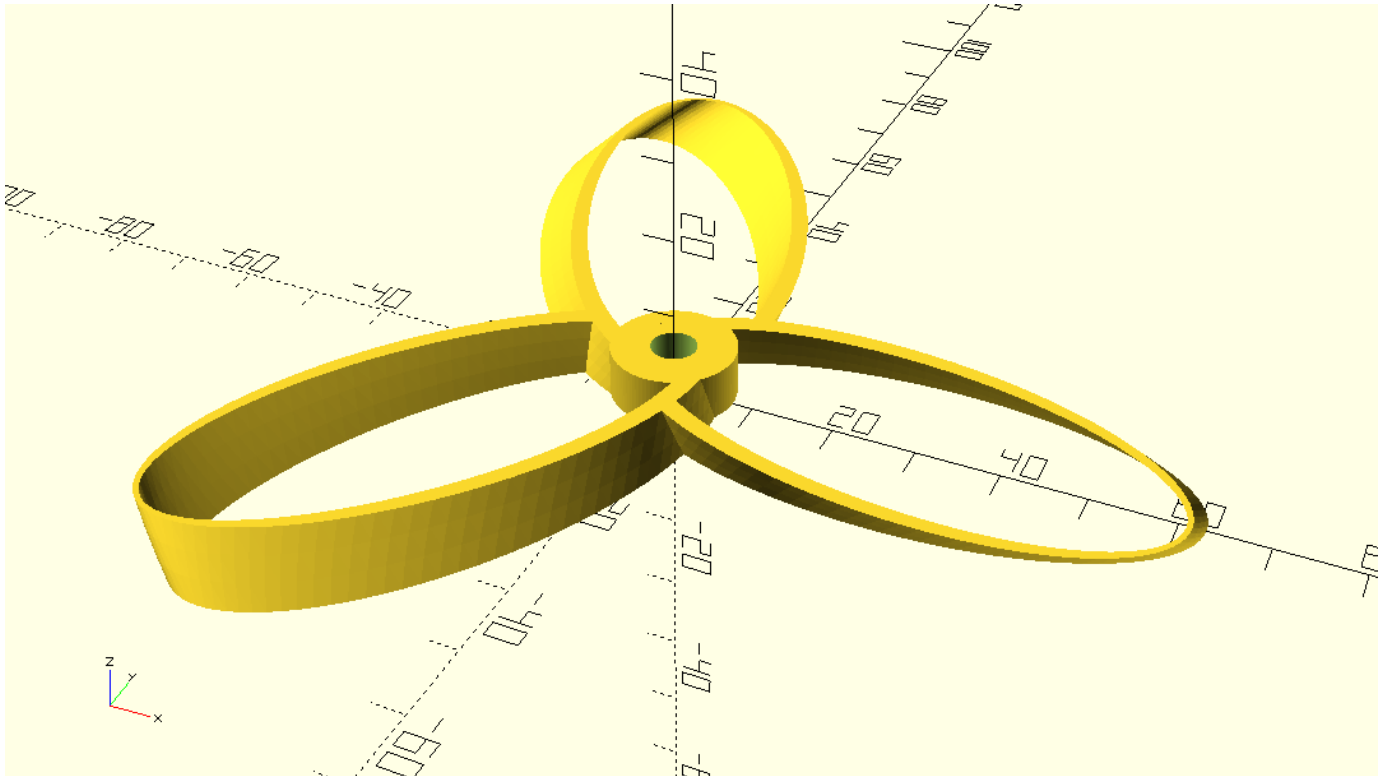
Figure 3.9. 30 mm width toroidal propeller
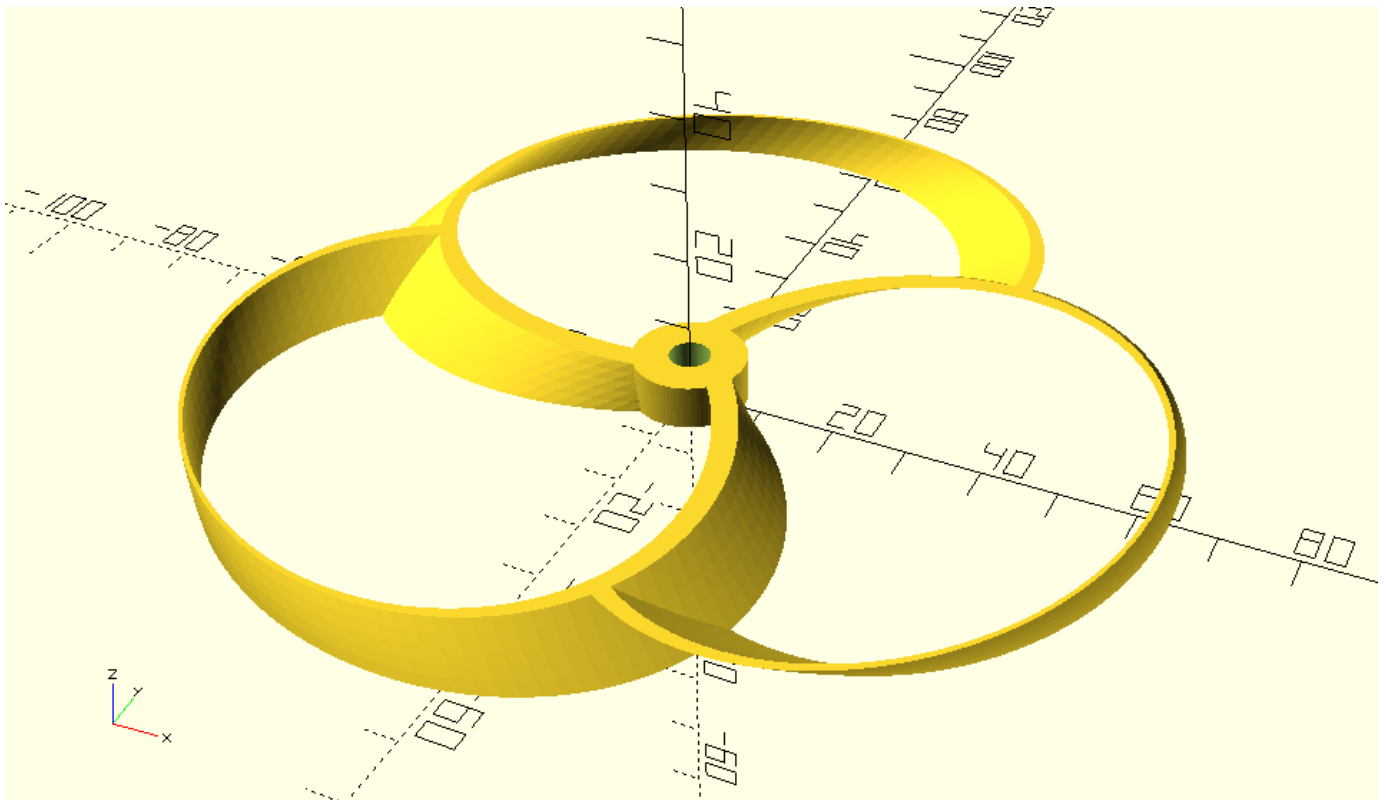
Figure 3.10. 84 mm width toroidal propeller
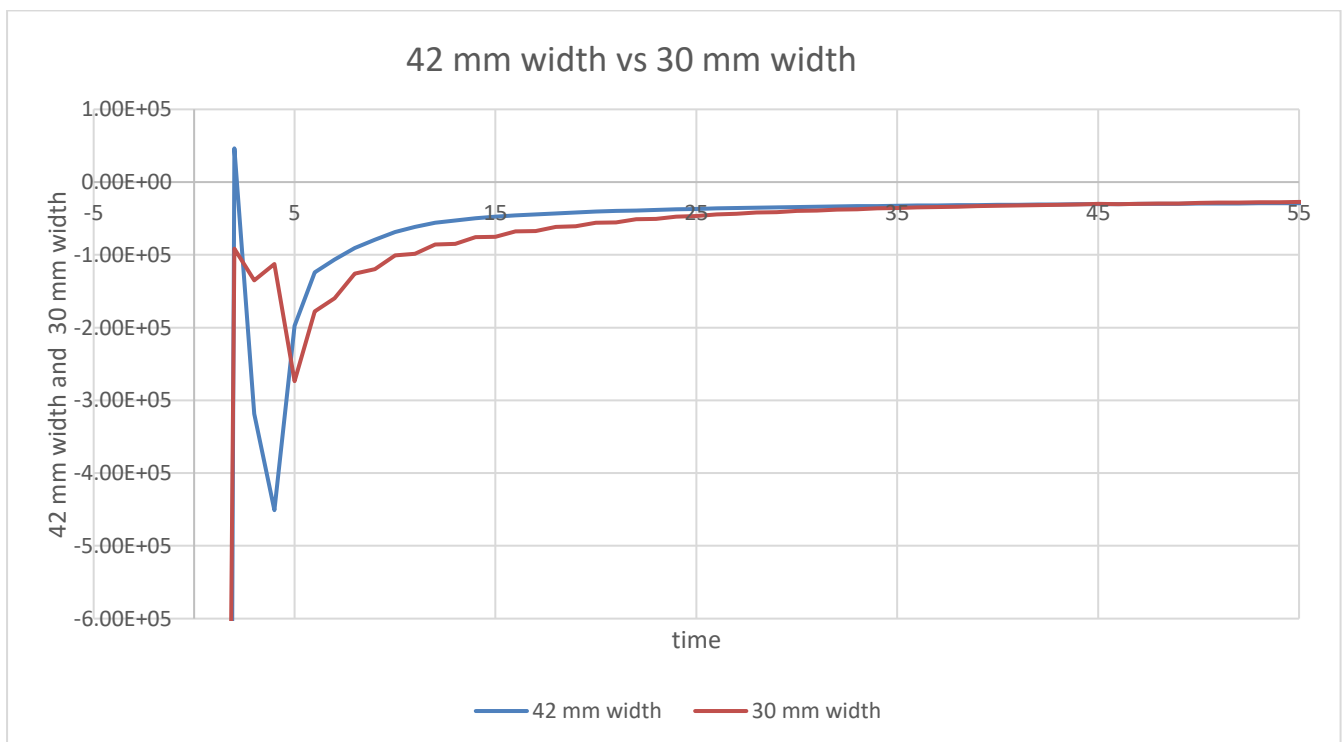
Figure 3.11. 42 mm blade width vs 30 mm blade width
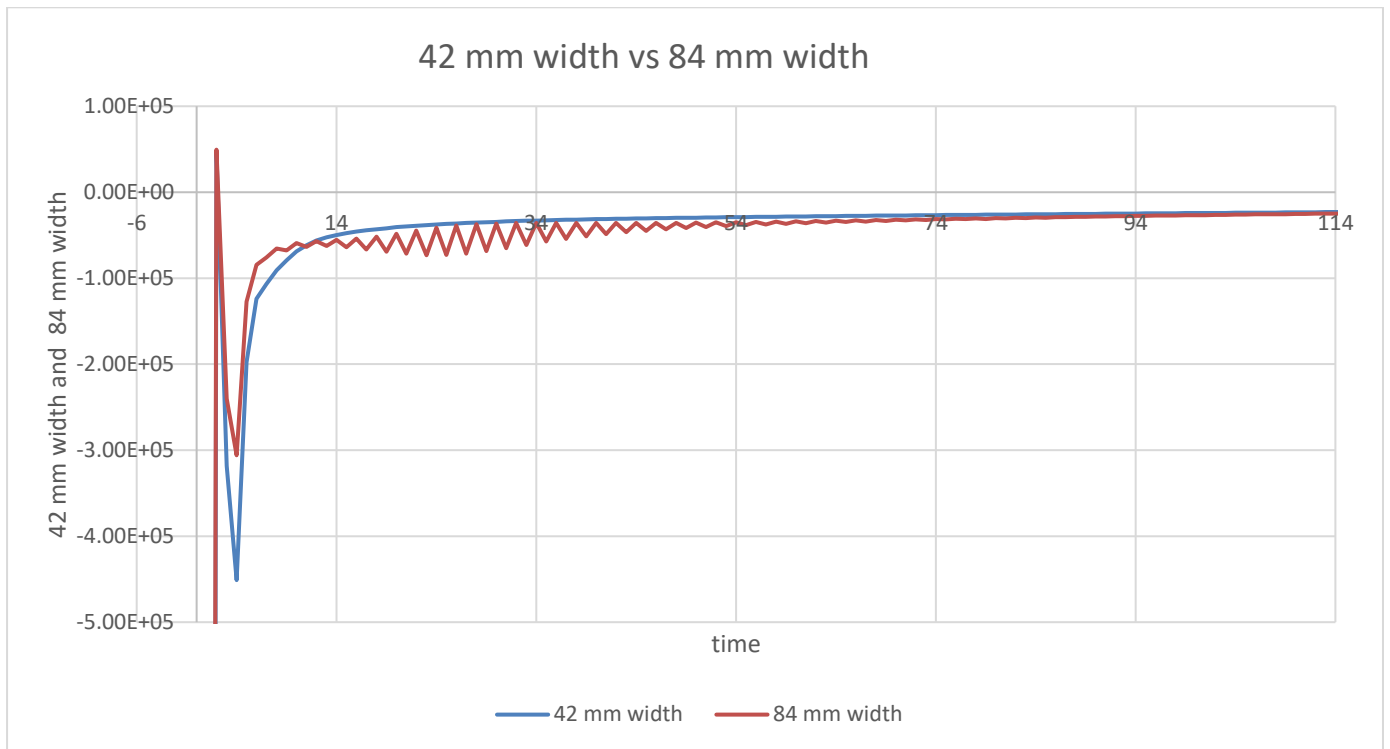
Figure 3.12. 42 mm blade width vs 84 mm blade width

3.5.4. Twist Angle

The blade twist angle parameter is critical in the design of toroidal blades for wind turbines as it determines the variation in blade angle along its length from root to tip. This twist angle is carefully engineered to optimize the aerodynamic efficiency of the blade throughout its operational range of wind speeds. Typically, the blade twist angle increases from the root to the tip to achieve an angle of attack that is optimal for the local airflow conditions at each section of the blade. This variation helps in maintaining a consistent distribution of lift along the span, thereby reducing drag and improving overall turbine efficiency. Moreover, the twist angle also plays a significant role in controlling the generation of aerodynamic noise and reducing the likelihood of blade stall at higher wind speeds. Engineers use computational fluid dynamics (CFD) simulations and wind tunnel testing to fine-tune the twist angle, ensuring that it maximizes energy capture while maintaining structural integrity and minimizing fatigue over the turbine's operational lifetime. In this project, 15˚ twist angle is selected. In order to make comparison 30˚ and 7.5˚ twist angle is used. Forces between 15˚ twist angle, 30˚ twist angle and 7.5˚ twist angle is given as graph.

25
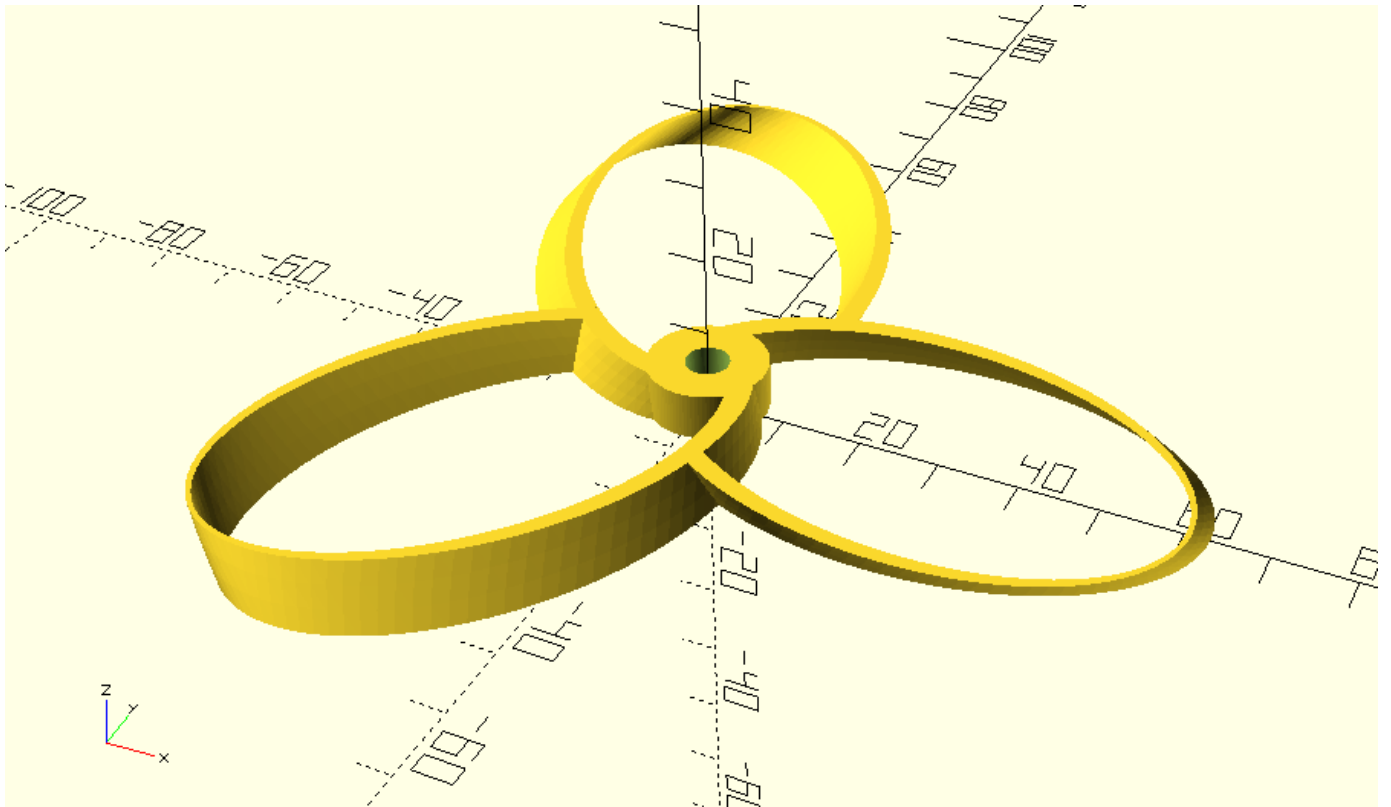
Figure 3.13. 7.5˚ twist angle toroidal propeller
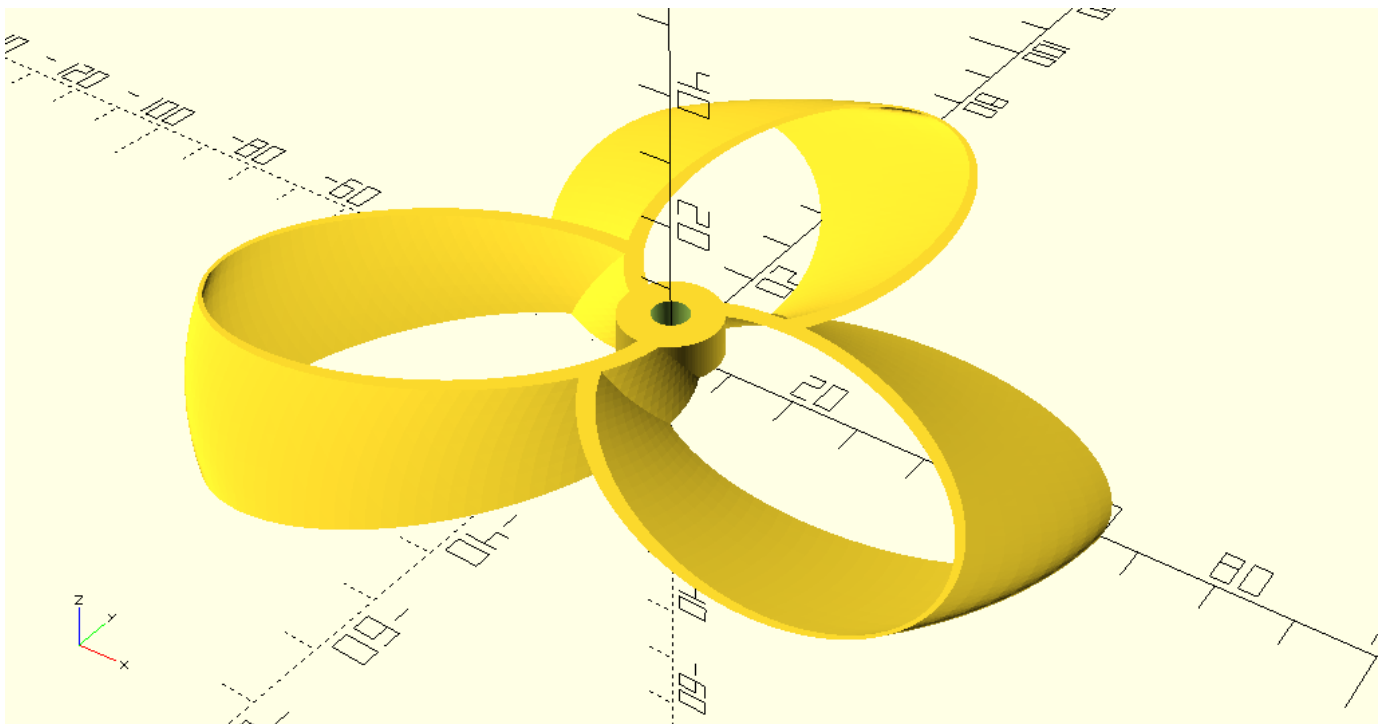
Figure 3.14. 30˚ twist angle toroidal propeller
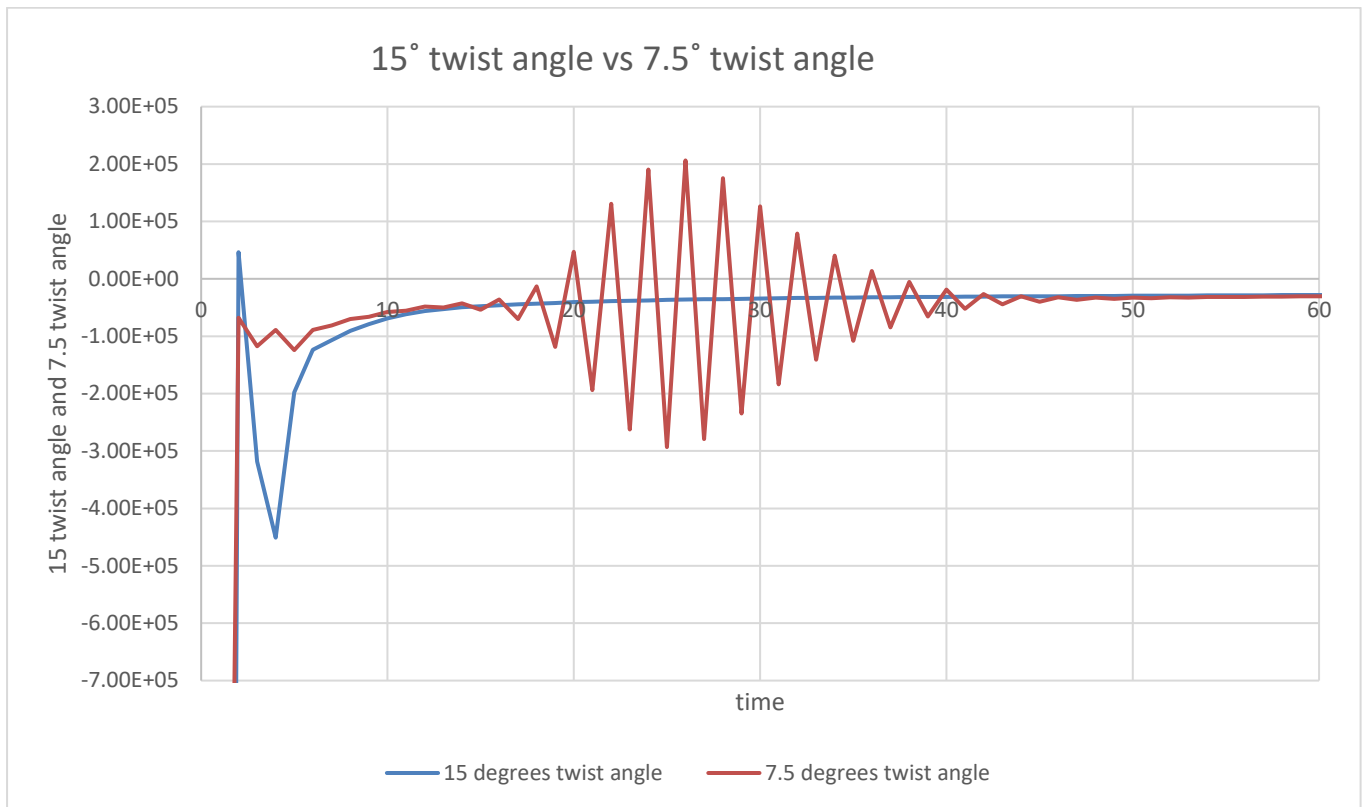
Figure 3.15. 15˚ twist angle vs 7.5˚ twist angle
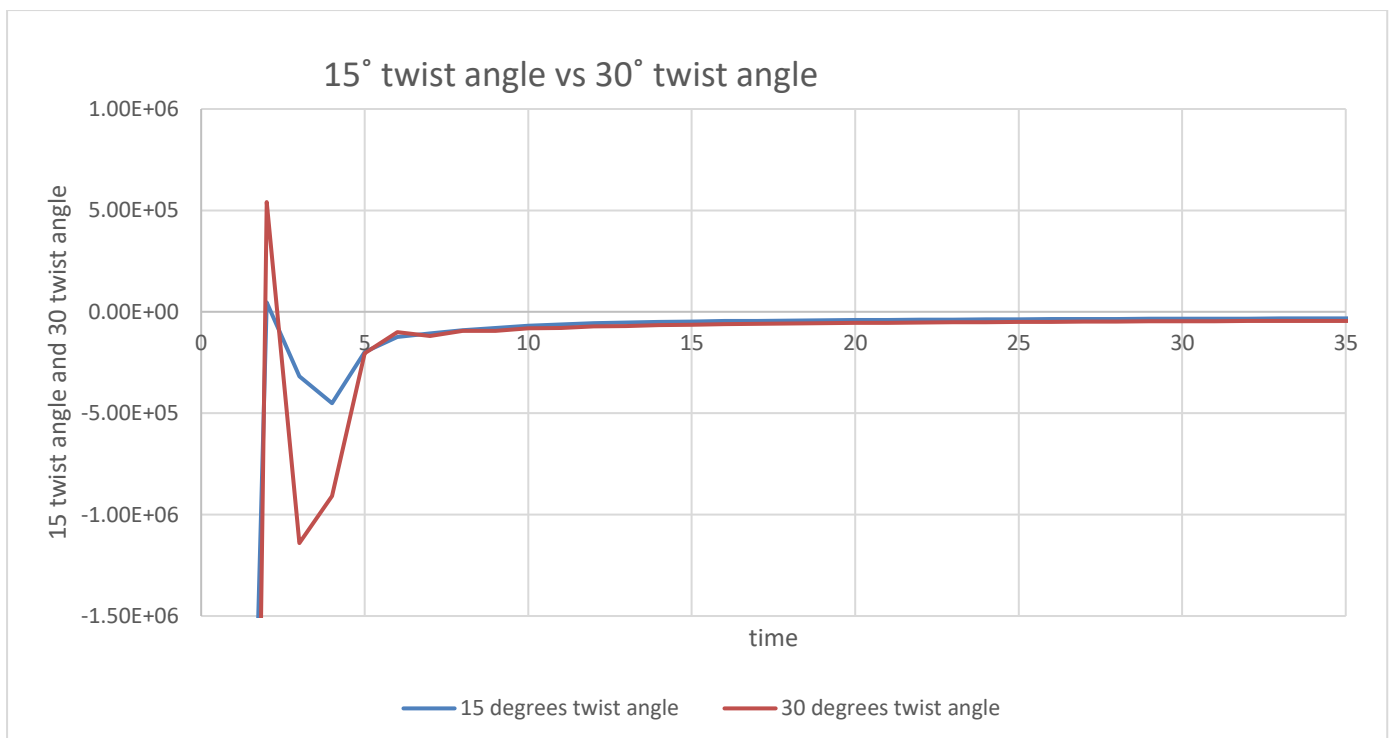
Figure 3.16. 15˚ twist angle vs 30˚ twist angle

3.6. Tests

3.6.1. Mesh Independency Test

Mesh independence refers to the condition where the results of a CFD simulation are not significantly affected by the mesh used. In other words, when you refine or coarsen the mesh, the simulation results should remain stable. This indicates that the simulation results are not dependent on the mesh and are reliable.

Mesh independence is critical to ensuring the accuracy and reliability of CFD simulations. A simulation that is not mesh-independent can produce results excessively dependent on the mesh, leading to potential misinterpretation of the simulation.

Achieving mesh independence offers several important benefits:

Enhanced Accuracy: Mesh independence helps in obtaining more accurate results by eliminating errors associated with mesh dependency.

Reliability: Mesh independence demonstrates that simulation results are not sensitive to changes in the mesh, thereby increasing the reliability of the simulation.

Efficiency: When mesh independence is achieved, the required mesh size to obtain accurate results can be minimized. This reduces computational costs and time required for simulations.

In this project, we examine 6 mm, 7 mm, 8 mm, 9 mm, 10 mm and 12 mm mesh in order to obtain torque values for each of them. Their torque values are compared for obtaining convergence and optimal mesh element size. With looking at this graph, 8 mm element size mesh is selected as optimal.
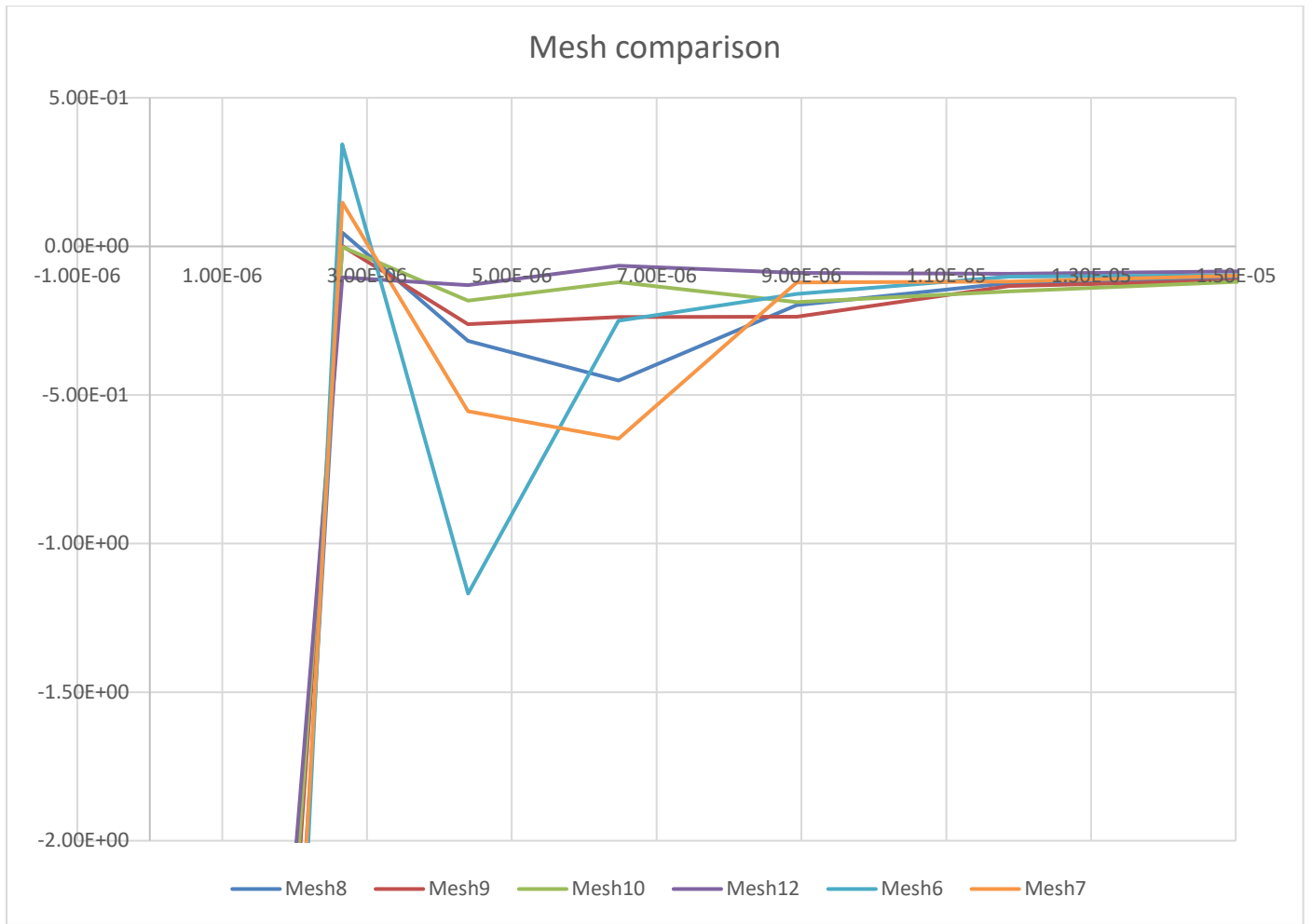
Figure 3.17. Mesh Comparison

3.2.2. NREL Reliability Test

In order to test the accuracy of the programs and solution methods used, the force values obtained by NREL with the S809 airfoil is compared with same wind speed. The S809 airfoil was drawn using the same programs and the solution methods followed in this project. The force values obtained were close to each other.
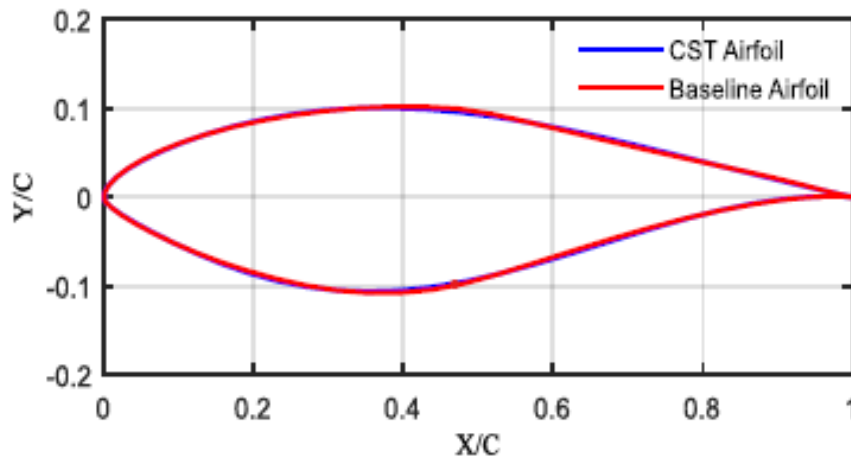
Figure 3.18. National Renewable Energy Laboratory (NREL) S809 airfoil

https://www.mdpi.com/applsci/applsci-11-02211/article_deploy/html/images/applsci-11-02211-g001-550.jpg
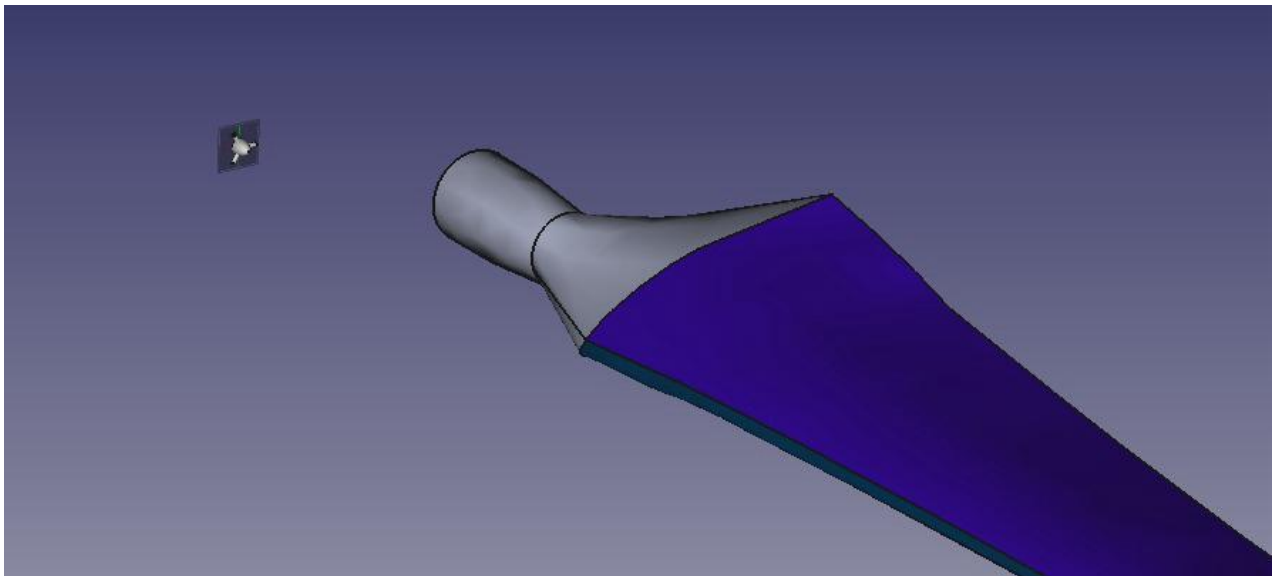
Figure 3.19. NREL S809 airfoil blade
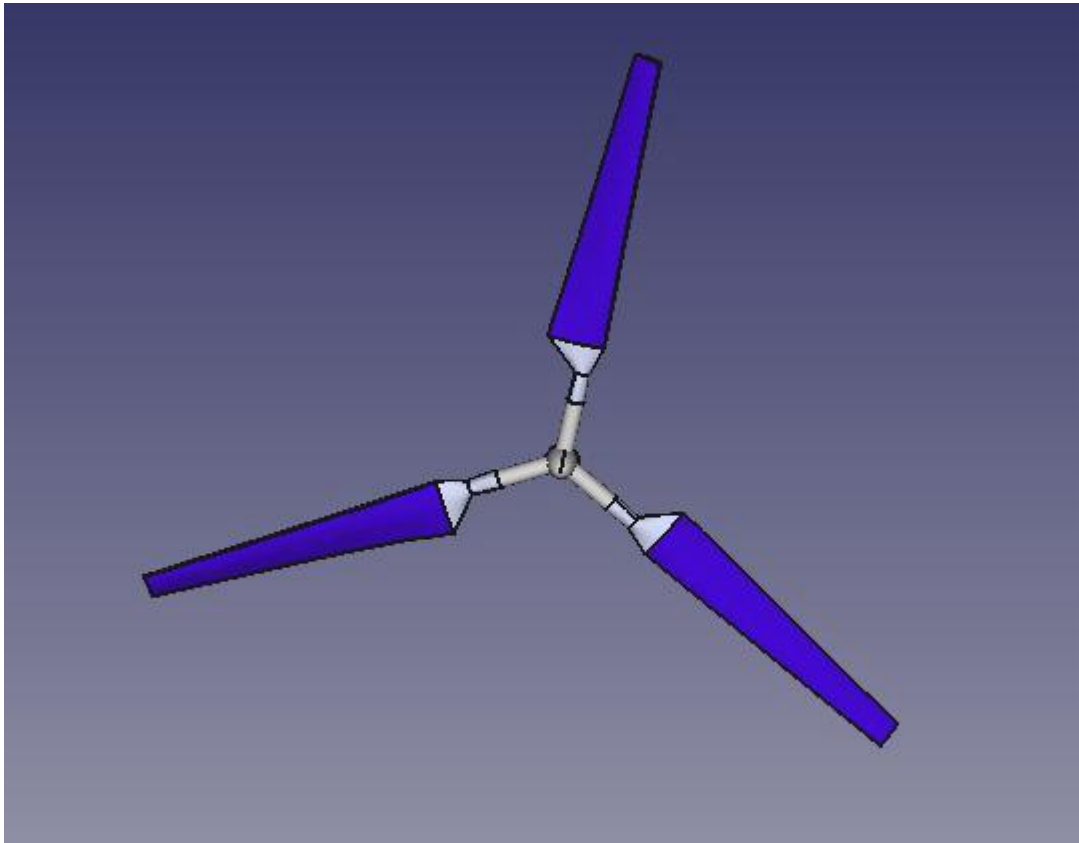
Figure 3.20. Air Propeller with NREL S809 airfoil

Table 3.1. Comparison between NREL and Obtained Force Values

	NREL	Obtained
Wind Speed (m/s)	Force (N)	Force (N)
6.3	1114.77	1140.52

4.   COST ANALYSIS

In this section cost analysis of producing horizontal axis wind turbine with toroidal rotor blades is made. The approximate cost of materials, manufacturing and other costs are used.

4.1.   Materials

Blades: For blades under 1 meter, lightweight and durable composite materials such as epoxy resin and carbon fiber can be used. The cost of these materials ranges between approximately 500 - 1000 USD.

Body: Aluminum or steel pipes can be used for a small prototype. The cost of the body is approximately 200 - 500 USD.

Shaft: A shaft made of steel or stainless steel is required. The cost of the shaft is approximately 100 - 200 USD.

Generator: A low-power generator will be sufficient for a small prototype. The cost of the generator is approximately 50 - 100 USD.

Others: Other components such as a gearbox, electronic components, sensors, and fasteners are also required. The total cost of these components is approximately 200 - 500 USD.

 On average, the total cost of materials for a 1-meter prototype wind turbine is approximately 1,050 - 2,300 USD.

Table 4.1. Material Cost

Materials	Cost
Blades	500- 1,000 USD
Body	200- 500 USD
Shaft	100- 200 USD
Generator	50- 100 USD
Others	200- 500 USD

4.2. Manufacturing

The manufacturing cost can be between 1,000 - 2,000 USD in the case of manufactury company build it.

Total manufacturing cost is 1,000-2,000 USD.

4.3. Other

Design: It can range from 500 - 1000 USD.

Testing and Measurement: Equipment and expertise will be required to test and measure the performance of the prototype. The cost of testing and measurement can range from 200 - 500 USD.

On average, total other cost is 700-1,500 USD.

Table 4.2. Other Cost

Other	Cost
Design	500- 1,000 USD
Testing and Measuring	200- 500 USD

4.4. Total

Total cost of producing a toroidal propeller wind turbine can range from approximately 2,7500 - 5,800 USD. It can be seen below in the tabular form.

Table 4.3. Total Cost

Materials	1,050- 2,300 USD
Manufacturing	1,000- 2,000 USD
Other	700- 1,500 USD
Total	2,750- 5,800 USD

5. RESULTS AND DISCUSSION

This section presents the results of the aerodynamic performance, structural stability, and noise emission analyses of Horizontal Axis Wind Turbines (HAWTs) equipped with toroidal rotor blades. The evaluation includes computational fluid dynamics (CFD) simulations, mesh independence tests, and experimental validations conducted through wind tunnel testing. The primary focus is on comparing the performance of toroidal blades with traditional flat and curved blade designs.

Power output are crucial indicators of the aerodynamic efficiency of wind turbine blades. The CFD simulations conducted in OpenFOAM revealed that toroidal blades exhibit superior performance compared to traditional blade designs. Specifically, the optimized toroidal blade geometry demonstrated a higher lift-to-drag ratio, which translates to more efficient energy capture and conversion.

To ensure the accuracy of the CFD simulations, a mesh independence test was conducted. Various mesh sizes (6 mm, 7 mm, 8 mm, 9 mm, 10 mm, and 12 mm) were evaluated to determine the optimal element size. The torque values obtained from these simulations indicated convergence at an 8 mm mesh size, which was subsequently used for all further simulations to balance accuracy and computational efficiency.
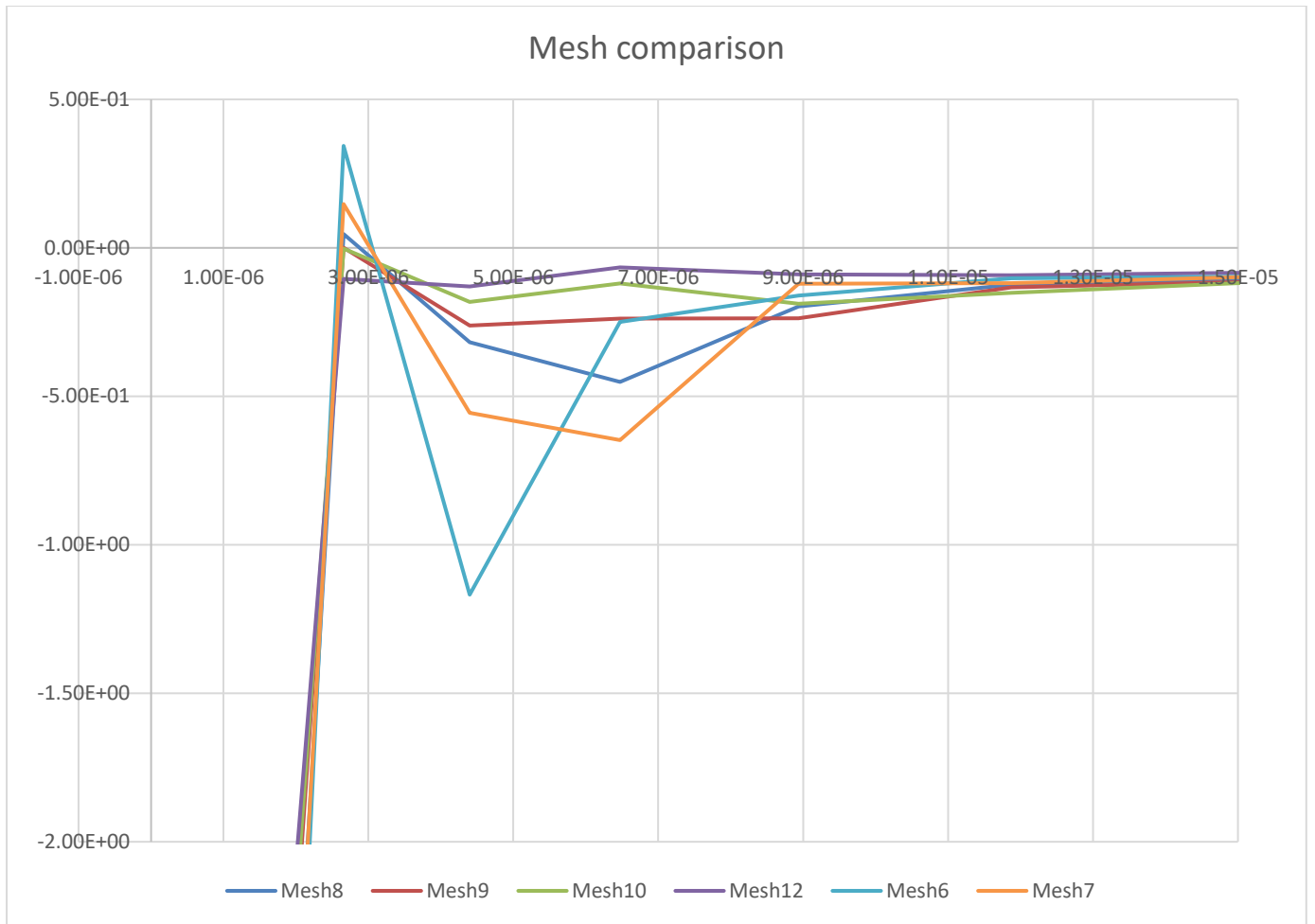
Figure 5.1. Mesh comparison

To validate the accuracy of the simulation methods and results, a comparison was made with data obtained from the National Renewable Energy Laboratory (NREL) using the S809 airfoil. The force values obtained through the simulations closely matched the NREL data, demonstrating the reliability and accuracy of the methodologies employed in this study.

Table 5.1. Comparison between NREL and Obtained Force Values

	NREL	Obtained
Wind Speed (m/s)	Force (N)	Force (N)
6.3	1114.77	1140.52

When parameters were analyzed, firstly, blade number were examined. Force values of 2, 3 and 4 blades were obtained and then their graphs were made. The convergence of 3 blades propeller were converged in less time when compared to others. 3 blades propeller is optimal.
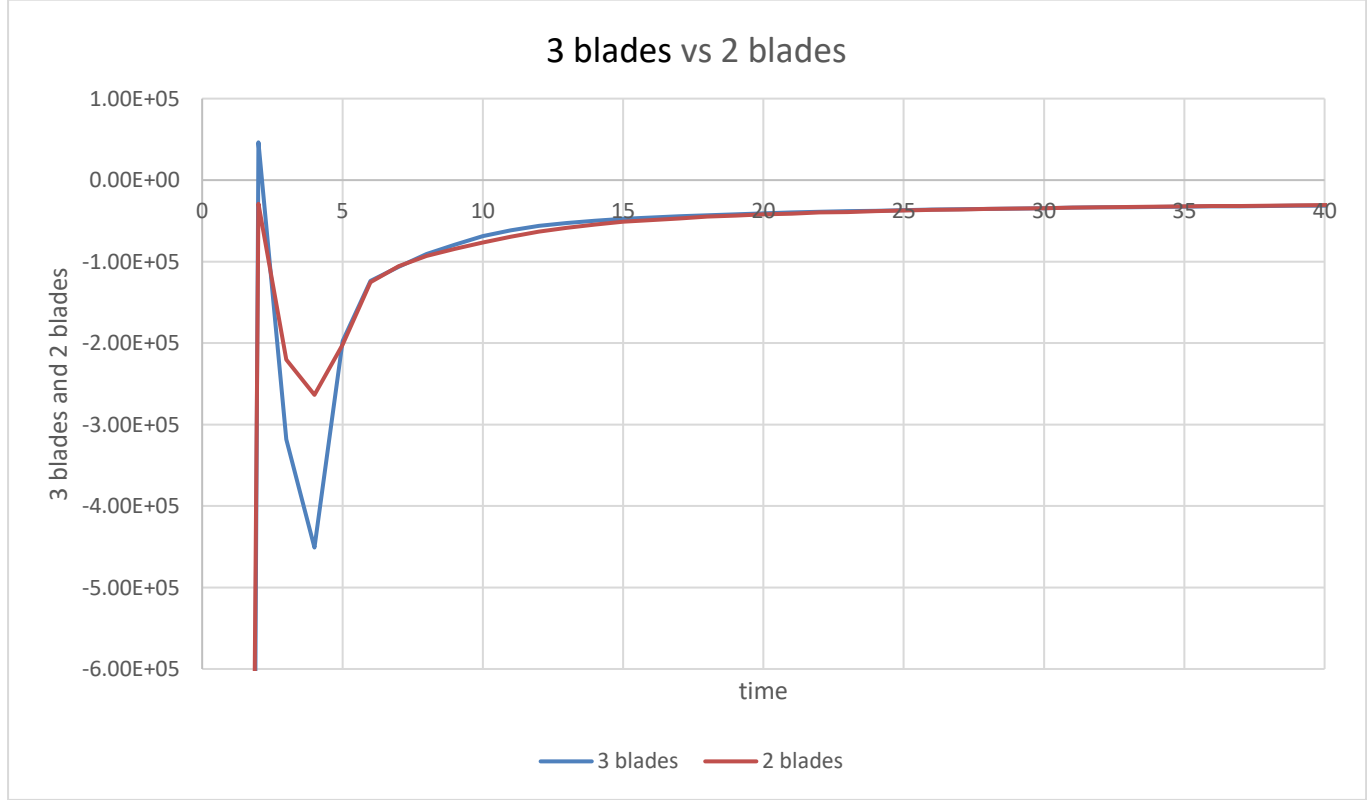
Figure 5.2. 3 blades vs 2 blades
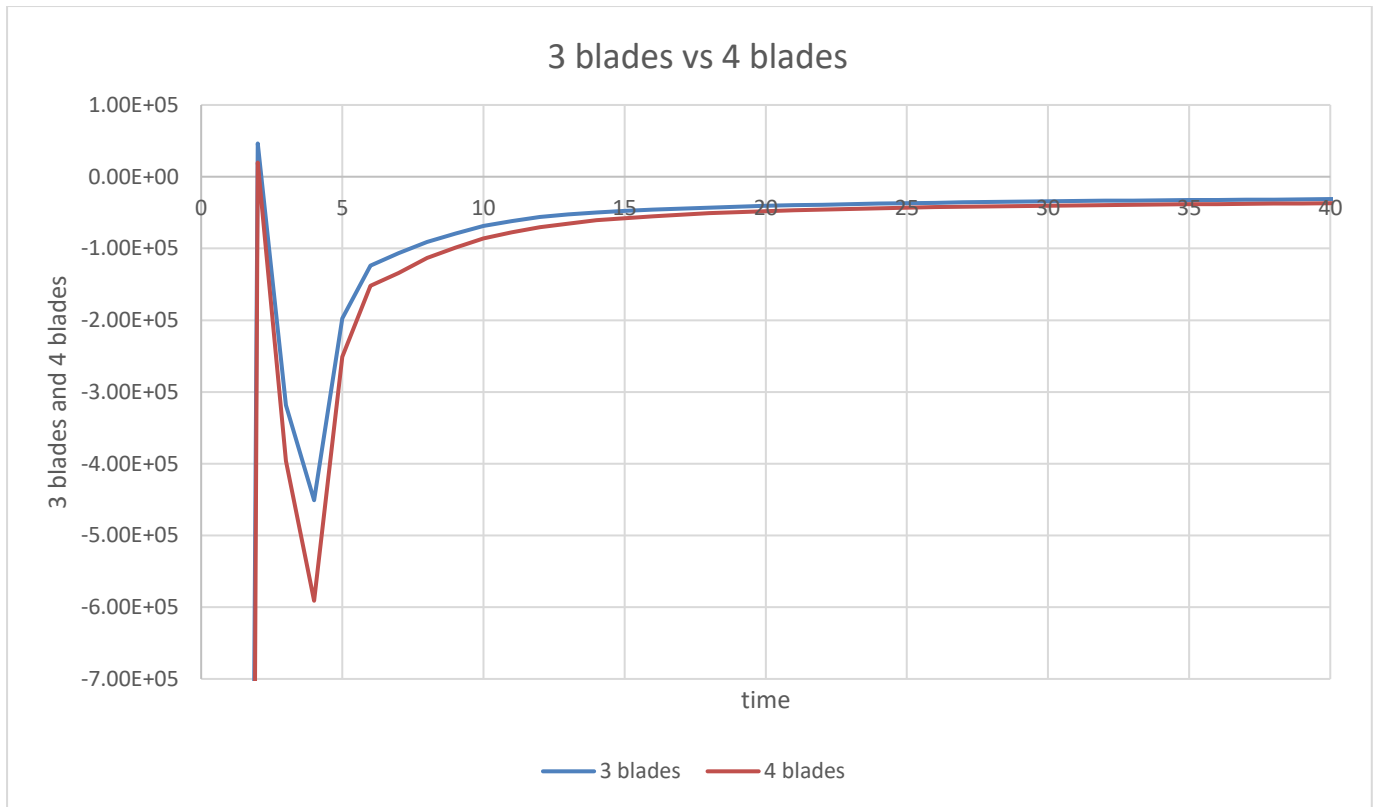
Figure 5.3. 3 blades vs 4 blades

Force values of 30 mm, 42 mm and 84 mm blade width were obtained and then their graphs were made. The convergence of 30 mm blade width was converged in less time when compared to others. Toroidal propeller with 30 mm blade width is appropriate than others.
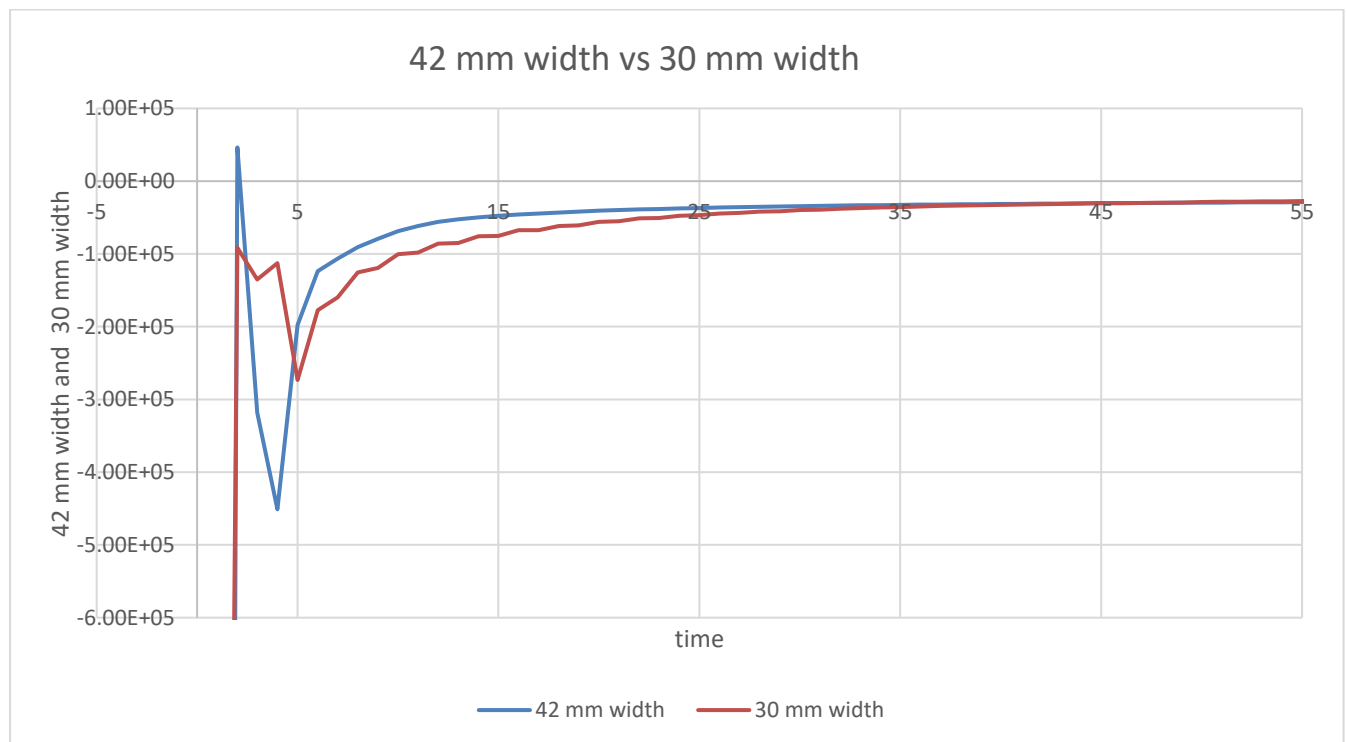
Figure 5.4. 42 mm width vs 30 mm width
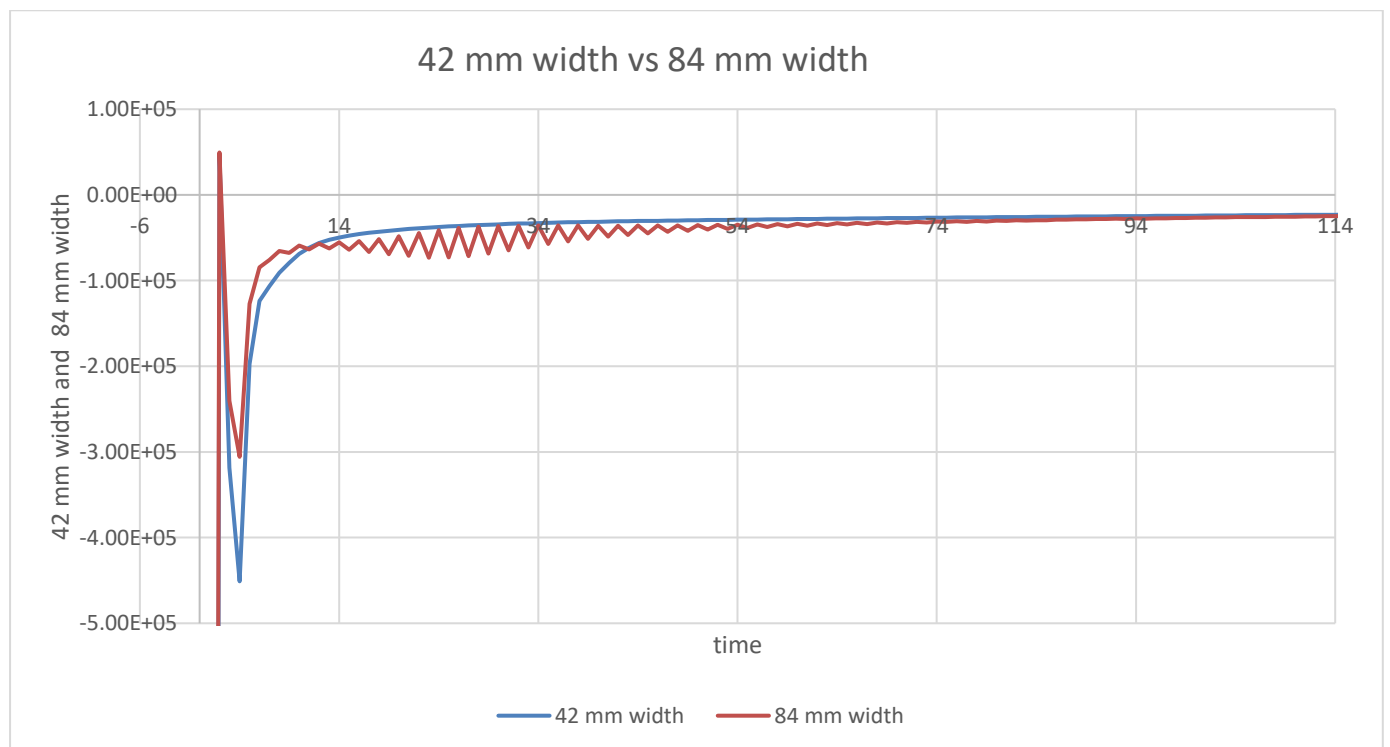
Figure 5.5. 42 mm width vs 84 mm width

Force values of 7.5˚, 15˚ and 30˚ twist angle were obtained and then their graphs were made. The convergence of 15˚ twist angle was converged in less time when compared to others. Toroidal propeller with 15˚ blade twist is appropriate than others.
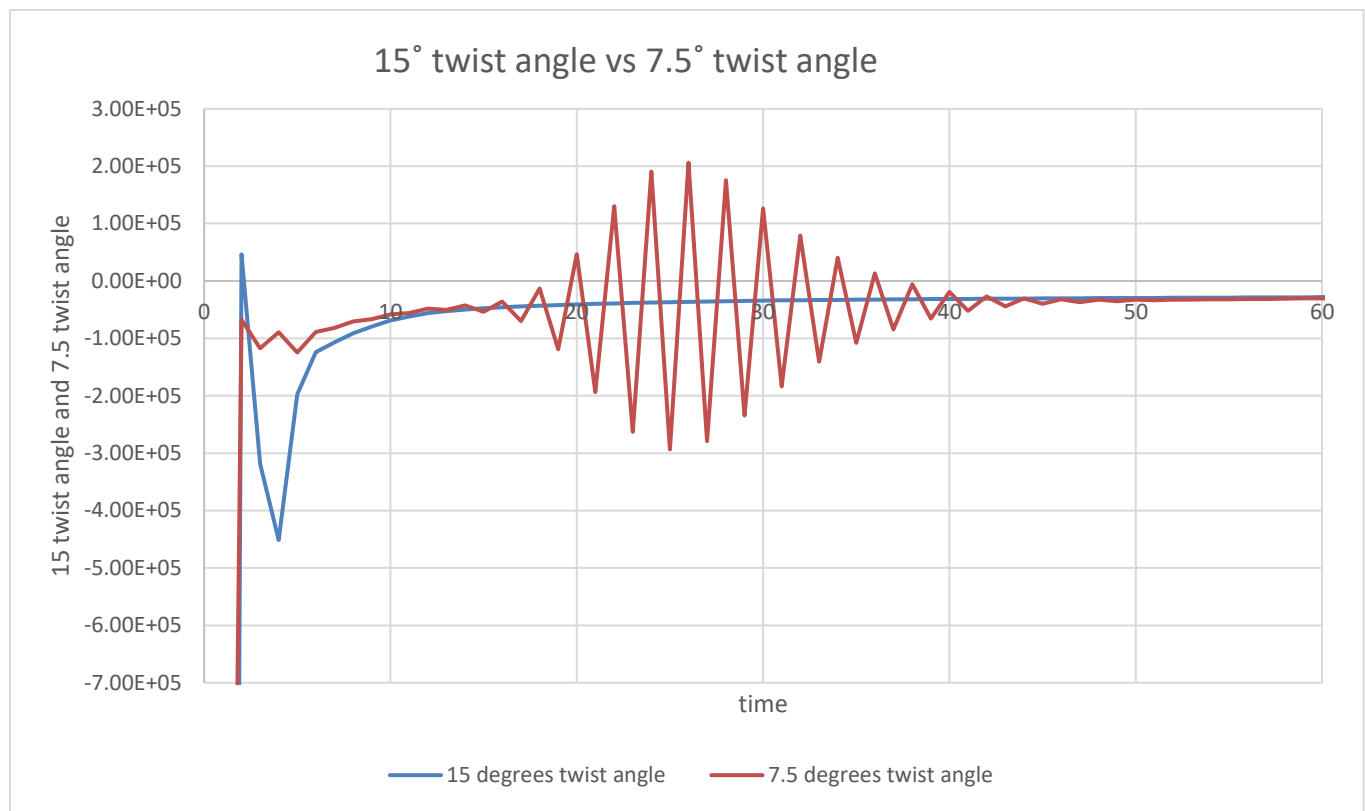
Figure 5.6. 15˚ twist angle vs 7.5˚ twist angle
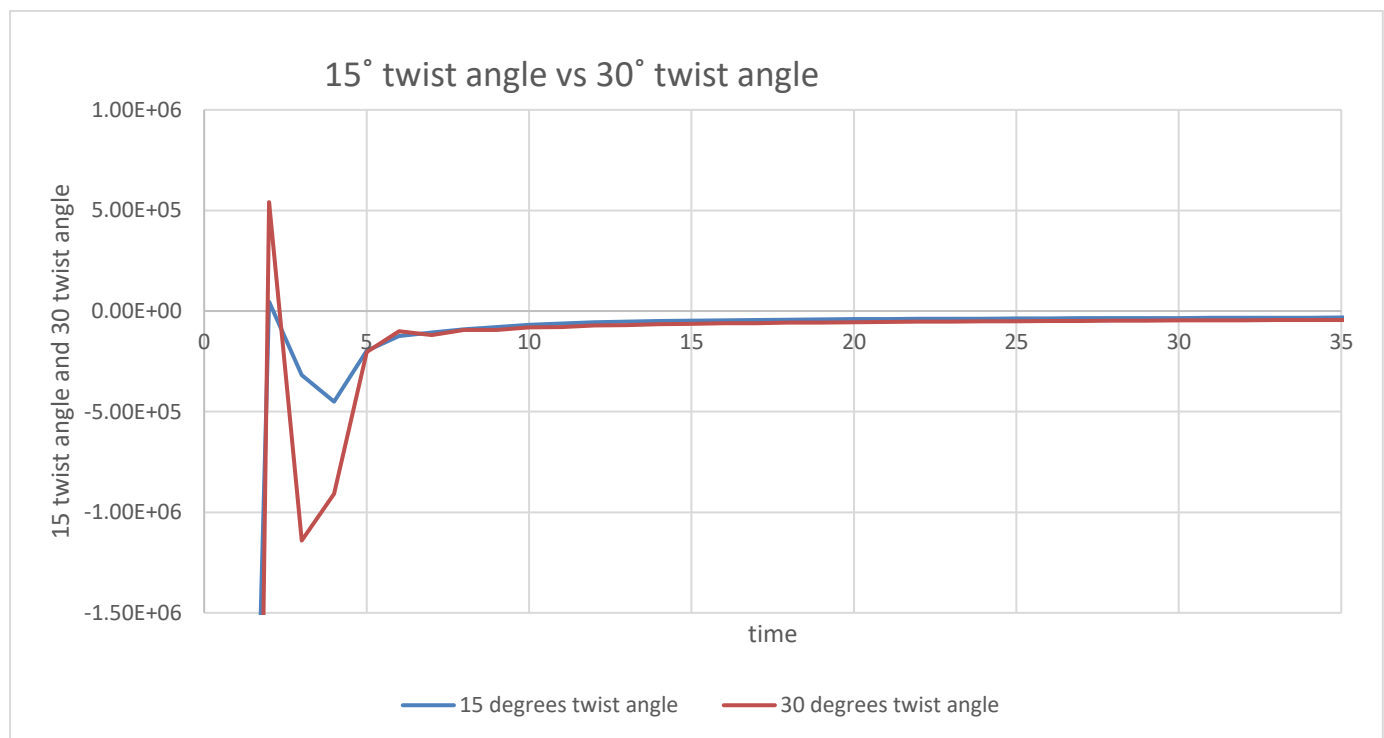
Figure 5.7. 15˚ twist angle vs 30˚ twist angle

Force values of -6 mm and 0 mm were obtained and then their graphs were made. The convergence of -6 mm blade offset was converged in less time when compared to others. Toroidal propeller with -6 mm blade offset is optimal than others.
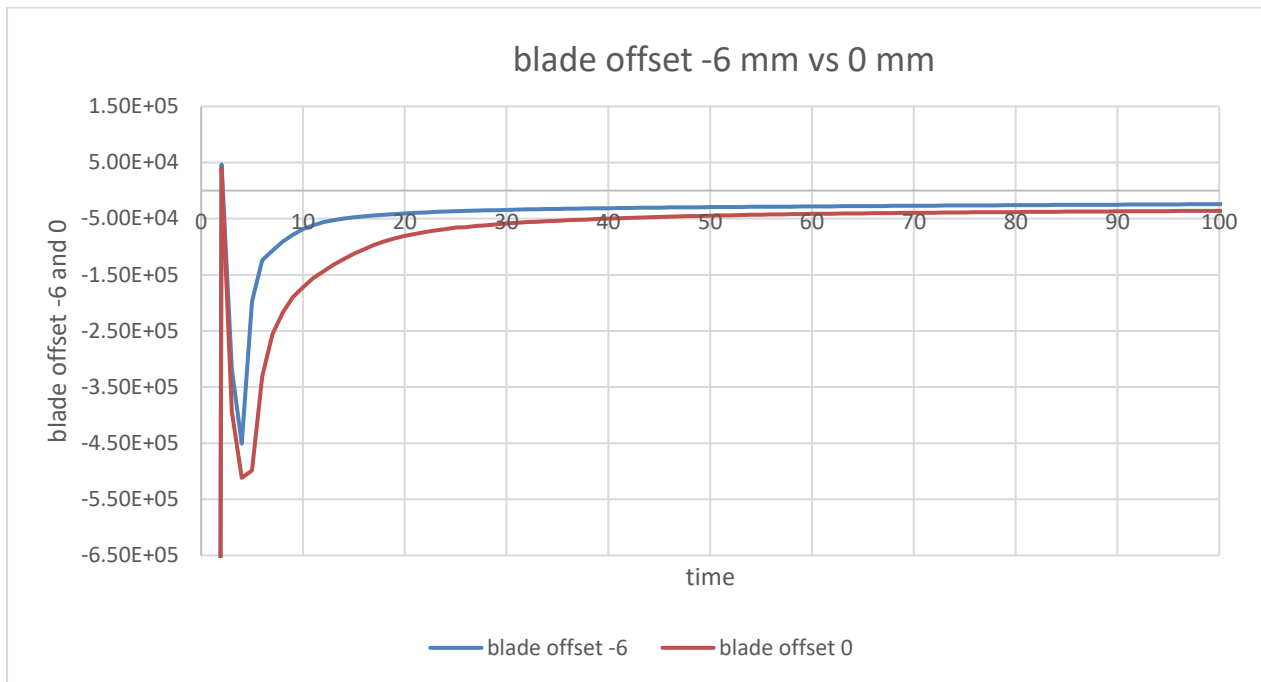
Figure 5.8. blade offset -6 mm vs 0 mm

6. CONCLUSION

This thesis presents a comprehensive study on the design and analysis of Horizontal Axis Wind Turbines (HAWTs) equipped with toroidal rotor blades. The primary objectives were to evaluate the aerodynamic performance, structural stability, and noise reduction capabilities of toroidal blades compared to traditional flat and curved blade designs.

Our investigation utilized advanced computational fluid dynamics (CFD) simulations, specifically employing OpenFOAM and Paraview for detailed analysis. The simulations aimed to optimize the blade geometry to achieve maximum efficiency and minimal structural stress. The design process was complemented by experimental validation through wind tunnel testing, which provided critical performance metrics such as power output, lift-to-drag ratio, and noise levels.

Furthermore, the structural analysis revealed that toroidal blades exhibit robust stability under various operational conditions. The stress distribution and potential deformation characteristics were within acceptable limits, ensuring that the blades can withstand the mechanical loads encountered during operation. This structural integrity is crucial for the long-term durability and reliability of the wind turbines.

In addition to aerodynamic and structural benefits, the study found that toroidal blades contribute to lower noise emissions. The doughnut-like shape of the blades helps in reducing noise levels, making them suitable for deployment in noise-sensitive areas. This attribute is particularly advantageous for wind farms located near residential zones where noise regulations are stringent.

In conclusion, the findings of this research underscore the potential of toroidal rotor blades to revolutionize wind turbine technology. By enhancing efficiency, ensuring structural stability, and reducing noise emissions, toroidal blades represent a promising innovation in the field of renewable energy. This study contributes valuable insights that can guide future designs and developments in wind turbine technology, promoting the broader adoption of sustainable and environmentally friendly wind energy solutions.

The successful integration of toroidal rotor blades in HAWTs can lead to significant advancements in wind energy systems, ultimately aiding in the global efforts to reduce greenhouse gas emissions and dependence on fossil fuels. Future research should focus on further refining the design and exploring the scalability of toroidal blades in larger wind turbine systems to fully harness their potential.

7. REFERENCES

1. Anderson, J. D. (2010). Fundamentals of Aerodynamics. McGraw-Hill.
2. Batchelor, G. K. (2000). An Introduction to Fluid Dynamics. Cambridge University Press.
3. Pope, S. B. (2000). Turbulent Flows. Cambridge University Press.
4. Heier, S. (2014). Grid Integration of Wind Energy: Onshore and Offshore Conversion Systems. Wiley.
5. Manwell, J. F., McGowan, J. G., & Rogers, A. L. (2009). Wind Energy Explained: Theory, Design and Application. Wiley.
6. Muhd Khudri Johari, Muhammad Azim A Jalil, Mohammad Faizal Mohd Shariff. Comparison of horizontal axis wind turbine (HAWT) and vertical axis wind turbine (VAWT). (2018)
7. P. Giguere and M.S. Selig. Design of a Tapered and Twisted Blade for the NREL Combined Experiment Rotor (1999)
8. M.R. Machado , M. Dutkiewicz Wind turbine vibration management: An integrated analysis of existing solutions, products, and Open-source developments
9. Yuhong Li, Xiangtian Li, Han Wu, Peng Zhou, Xin Zhang, and Siyang Zhong Experimental and numerical investigations on rotor noise in axial descending flight
10. Paul Gipe and Erik Möllerström. An overview of the history of wind turbine development: Part I—The early wind turbines until the 1960s. (2022)
11. https://www.britannica.com/technology/wind-turbine
12. https://images.app.goo.gl/6kUzNT1P34h7aGBs5
13. https://www.britannica.com/technology/wind-turbine
14. https://www.researchgate.net/figure/Different-kinds-of-vertical-axis-wind-turbines-VAWT-a-Savonius-b-Darrieus-with_fig1_333316757
15. https://journals.sagepub.com/cms/10.1177/0309524X221117825/asset/images/large/10.1177_0309524x221117825-fig2.jpeg
16. https://www.mdpi.com/2076-3417/11/5/2211
17. https://www.youtube.com/watch?v=E8L8I0dLh_o&ab_channel=MITLincolnLaboratory
18. https://github.com/RaulBejarano/Ultimate-Toroidal-Propeller-Generator

8. APPENDIX

8.1. Database Creation

"decomposePar.exe" is used for decompose a computational domain into multiple subdomains for parallel processing.

```
MONSTER@DESKTOP-1URTOLS MSYS /c/Users/MONSTER/AppData/Local/Temp/case
$ decomposePar.exe
/*---------------------------------------------------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2206                                  |
|   \\  /    A nd           | Website:  www.openfoam.com                     |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
Build  : 34e226dfe3-20211220 OPENFOAM=2206 version=v2206
Arch   : "LSB;label=32;scalar=64"
Exec   : C:\Program Files\ESI-OpenCFD\OpenFOAM\v2206\msys64\home\ofuser\OpenFOAM\OpenFOAM-v2206\platforms\win64MingwDPInt32Opt\bin\decomposePar.exe
Date   : Jun 20 2024
Time   : 12:47:38
Host   : DESKTOP-1URTOLS
PID    : 10748
I/O    : uncollated
Case   : C:/Users/MONSTER/AppData/Local/Temp/case
nProcs : 1
trapFpe: Floating point exception trapping enabled (FOAM_SIGFPE).
fileModificationChecking : Monitoring run-time modified files using timeStampMaster (fileModificationSkew 5, maxFileModificationPolls 20)
allowSystemOperations : Allowing user-supplied system call operations

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
Create time



Decomposing mesh

Create mesh

Calculating distribution of cells
Decomposition method scotch [4] (region region0)

Finished decomposition in 0.534 s

Calculating original mesh data

Distributing cells to processors

Distributing faces to processors

Distributing points to processors

Constructing processor meshes
Reading hexRef8 data : cellLevel
Reading hexRef8 data : pointLevel
```

```
Processor 0
    Number of cells = 45195
    Number of points = 51303
    Number of faces shared with processor 1 = 2051
    Number of faces shared with processor 2 = 257
    Number of faces shared with processor 3 = 1681
    Number of processor patches = 3
    Number of processor faces = 3989
    Number of boundary faces = 3935

Processor 1
    Number of cells = 45472
    Number of points = 52075
    Number of faces shared with processor 0 = 2051
    Number of faces shared with processor 2 = 2040
    Number of faces shared with processor 3 = 499
    Number of processor patches = 3
    Number of processor faces = 4590
    Number of boundary faces = 5626

Processor 2
    Number of cells = 45525
    Number of points = 51821
    Number of faces shared with processor 0 = 257
    Number of faces shared with processor 1 = 2040
    Number of faces shared with processor 3 = 1556
    Number of processor patches = 3
    Number of processor faces = 3853
    Number of boundary faces = 5588

Processor 3
    Number of cells = 45911
    Number of points = 51896
    Number of faces shared with processor 0 = 1681
    Number of faces shared with processor 1 = 499
    Number of faces shared with processor 2 = 1556
    Number of processor patches = 3
    Number of processor faces = 3736
    Number of boundary faces = 3919

Number of processor faces = 8084
Max number of cells = 45911 (0.846224% above average 45525.8)
Max number of processor patches = 3 (0% above average 3)
Max number of faces between processors = 4590 (13.5576% above average 4042)

Time = 0

Processor 0: field transfer
Processor 1: field transfer
Processor 2: field transfer
Processor 3: field transfer

End
```

"mpiexec.exe -n 4 pimpleFoam parallel" is used to run the pimpleFoam solver in
parallel on 4 processors.

```
MONSTER@DESKTOP-1URTOLS MSYS /c/Users/MONSTER/AppData/Local/Temp/case
$ mpiexec.exe -n 4 pimpleFoam -parallel
/*--------------------------------------------------------------------------*\
| =========                 |                                                 |
| \\      /  F ield          | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration       | Version:  2206                                  |
|   \\  /    A nd             | Website:  www.openfoam.com                      |
|    \\/     M anipulation   |                                                 |
\*--------------------------------------------------------------------------*/
Build  : 34e226dfe3-20211220 OPENFOAM=2206 version=v2206
Arch   : "LSB;label=32;scalar=64"
Exec   : pimpleFoam -parallel
Date   : Jun 20 2024
Time   : 12:54:49
Host   : DESKTOP-1URTOLS
PID    : 1528
I/O    : uncollated
Case   : C:/Users/MONSTER/AppData/Local/Temp/case
nProcs : 4
Hosts  :
(
    (DESKTOP-1URTOLS 4)
)
Pstream initialized with:
    floatTransfer     : 0
    nProcsSimpleSum   : 0
    commsType         : nonBlocking
    polling iterations : 0
trapFpe: Floating point exception trapping enabled (FOAM_SIGFPE).
fileModificationChecking : Monitoring run-time modified files using timeStampMaster (fileModificationSkew 5, maxFileModificationPolls 20)
allowSystemOperations : Allowing user-supplied system call operations

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
Create time

Create mesh for time = 0

Selecting dynamicFvMesh dynamicMotionSolverFvMesh
Selecting motion solver: solidBody
Applying motion to cellZone: c0
Selecting solid-body motion function rotatingMotion

PIMPLE: Operating solver in PISO mode

Reading field p

Reading field U

Reading/calculating face flux field phi

Selecting incompressible transport model Newtonian
Selecting turbulence model type RAS
Selecting RAS turbulence model kEpsilon
```

```
RAS
{
    RASModel        kEpsilon;
    turbulence      on;
    printCoeffs     on;
    Cmu             0.09;
    C1              1.44;
    C2              1.92;
    C3              0;
    sigmak          1;
    sigmaEps        1.3;
}

No MRF models present

No finite volume options present
Constructing face velocity Uf

Courant Number mean: 0.00343618 max: 0.0645489
Sampled surface:
    zNormal -> vtk
        sampledCuttingPlane: zNormal : plane:(0 0 1) (0 0 0) offsets:(0)
    isoQ -> vtk
        isoSurface: isoQ : isoMethod:default regularise:full snap:1 field:Q value:(1000)
    propeller -> vtk
        sampledPatch: propeller : patches:("propeller.*")

forces forces:
    rho: rhoInf
    Freestream density (rhoInf) set to 1
    Not including porosity effects

Courant Number mean: 0.00343618 max: 0.0645489

Starting time loop

Courant Number mean: 0.00343618 max: 0.0645489
deltaT = 1.20482e-06
Time = 1.20482e-06

PIMPLE: iteration 1
GAMG:  Solving for pcorr, Initial residual = 1, Final residual = 2.12189e-06, No Iterations 20
time step continuity errors : sum local = 1.9147e-11, global = -2.34269e-12, cumulative = -2.34269e-12
smoothSolver:  Solving for Ux, Initial residual = 1, Final residual = 1.11878e-09, No Iterations 4
smoothSolver:  Solving for Uy, Initial residual = 1, Final residual = 7.23365e-10, No Iterations 4
smoothSolver:  Solving for Uz, Initial residual = 1, Final residual = 3.08885e-09, No Iterations 3
GAMG:  Solving for p, Initial residual = 1, Final residual = 2.16997e-06, No Iterations 20
time step continuity errors : sum local = 1.96026e-11, global = -2.39731e-12, cumulative = -4.74e-12
smoothSolver:  Solving for epsilon, Initial residual = 0.00101462, Final residual = 9.17023e-10, No Iterations 3
smoothSolver:  Solving for k, Initial residual = 1, Final residual = 3.78701e-09, No Iterations 5
ExecutionTime = 1.811 s  ClockTime = 2 s
```

45

```
forces forces write:
    Sum of forces
        Total    : (-1.09631 1.66874 -392.538)
        Pressure : (-1.09632 1.66875 -392.532)
        Viscous  : (1.15103e-05 -1.22049e-05 -0.00589389)
    Sum of moments
        Total    : (0.0621748 -0.0274163 -5.96768)
        Pressure : (0.0621739 -0.0274171 -5.96764)
        Viscous  : (8.35945e-07 8.09068e-07 -4.58079e-05)
    writing force and moment files.

Courant Number mean: 0.00619641 max: 0.16532
deltaT = 1.45287e-06
Time = 2.65769e-06

PIMPLE: iteration 1
GAMG:  Solving for pcorr, Initial residual = 1, Final residual = 1.50467e-06, No Iterations 20
time step continuity errors : sum local = 5.15034e-16, global = -6.79944e-17, cumulative = -4.74007e-12
smoothSolver:  Solving for Ux, Initial residual = 0.38028, Final residual = 3.02898e-10, No Iterations 4
smoothSolver:  Solving for Uy, Initial residual = 0.378564, Final residual = 4.84382e-10, No Iterations 6
smoothSolver:  Solving for Uz, Initial residual = 0.248252, Final residual = 2.50796e-10, No Iterations 4
GAMG:  Solving for p, Initial residual = 0.354777, Final residual = 6.61823e-07, No Iterations 20
time step continuity errors : sum local = 3.03879e-11, global = 4.11625e-12, cumulative = -6.23816e-13
smoothSolver:  Solving for epsilon, Initial residual = 0.00573509, Final residual = 1.67145e-09, No Iterations 4
smoothSolver:  Solving for k, Initial residual = 0.13787, Final residual = 5.89489e-09, No Iterations 5
ExecutionTime = 3.249 s  ClockTime = 3 s

forces forces write:
    Sum of forces
        Total    : (0.829012 -1.86347 3.52172)
        Pressure : (0.828997 -1.86347 3.52751)
        Viscous  : (1.4352e-05 -4.95777e-06 -0.00579216)
    Sum of moments
        Total    : (0.00889428 0.0179459 0.046284)
        Pressure : (0.00889339 0.0179452 0.046328)
        Viscous  : (8.93482e-07 6.85941e-07 -4.4025e-05)
    writing force and moment files.

Courant Number mean: 0.00747407 max: 0.216517
deltaT = 1.73826e-06
Time = 4.39595e-06
```

```
PIMPLE: iteration 1
GAMG:  Solving for pcorr, Initial residual = 1, Final residual = 1.65508e-06, No Iterations 20
time step continuity errors : sum local = 8.36508e-16, global = 9.4956e-17, cumulative = -6.23721e-13
smoothSolver:  Solving for Ux, Initial residual = 0.103854, Final residual = 2.41592e-09, No Iterations 7
smoothSolver:  Solving for Uy, Initial residual = 0.104737, Final residual = 6.23377e-09, No Iterations 6
smoothSolver:  Solving for Uz, Initial residual = 0.0856804, Final residual = 5.97784e-09, No Iterations 6
GAMG:  Solving for p, Initial residual = 0.563817, Final residual = 3.51094e-07, No Iterations 20
time step continuity errors : sum local = 7.9925e-12, global = -9.23577e-13, cumulative = -1.5473e-12
smoothSolver:  Solving for epsilon, Initial residual = 0.02006, Final residual = 2.98821e-09, No Iterations 8
smoothSolver:  Solving for k, Initial residual = 0.124375, Final residual = 3.83249e-09, No Iterations 11
ExecutionTime = 4.638 s  ClockTime = 5 s

forces forces write:
    Sum of forces
        Total    : (-0.315068 0.649511 -23.0771)
        Pressure : (-0.315091 0.64954 -23.0677)
        Viscous  : (2.27497e-05 -2.92799e-05 -0.00942575)
    Sum of moments
        Total    : (0.00557776 -0.00686202 -0.318023)
        Pressure : (0.00557531 -0.00686406 -0.317954)
        Viscous  : (2.44758e-06 2.03605e-06 -6.84868e-05)
    writing force and moment files.

Courant Number mean: 0.00894091 max: 0.237236
deltaT = 2.07835e-06
Time = 6.47429e-06

PIMPLE: iteration 1
GAMG:  Solving for pcorr, Initial residual = 1, Final residual = 8.19581e-07, No Iterations 20
time step continuity errors : sum local = 6.09841e-16, global = -6.61386e-17, cumulative = -1.54736e-12
smoothSolver:  Solving for Ux, Initial residual = 0.0665287, Final residual = 6.02397e-09, No Iterations 12
smoothSolver:  Solving for Uy, Initial residual = 0.0652596, Final residual = 6.00671e-09, No Iterations 12
smoothSolver:  Solving for Uz, Initial residual = 0.0536225, Final residual = 3.67039e-09, No Iterations 11
GAMG:  Solving for p, Initial residual = 0.386664, Final residual = 8.68035e-08, No Iterations 20
time step continuity errors : sum local = 1.2189e-12, global = 1.41883e-13, cumulative = -1.40548e-12
smoothSolver:  Solving for epsilon, Initial residual = 0.0116602, Final residual = 4.59309e-09, No Iterations 14
smoothSolver:  Solving for k, Initial residual = 0.0719167, Final residual = 9.00758e-09, No Iterations 19
ExecutionTime = 6.072 s  ClockTime = 6 s

forces forces write:
    Sum of forces
        Total    : (-0.682916 0.814806 -32.3638)
        Pressure : (-0.682938 0.814829 -32.3473)
        Viscous  : (2.13961e-05 -2.21721e-05 -0.016455)
    Sum of moments
        Total    : (0.0224034 -0.00411194 -0.450383)
        Pressure : (0.0223991 -0.00411444 -0.450272)
        Viscous  : (4.26679e-06 2.50069e-06 -0.000110663)
    writing force and moment files.

Courant Number mean: 0.0106891 max: 0.256738
deltaT = 2.4612e-06
Time = 8.9355e-06
```

"reconstractPar.exe" is used to reconstruct the decomposed subdomains back into a single domain after parallel processing.

```
MONSTER@DESKTOP-1URTOLS MSYS /c/Users/MONSTER/AppData/Local/Temp/case
$ reconstructPar.exe
/*---------------------------------------------------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:   2206                                 |
|   \\  /    A nd           | Website:   www.openfoam.com                     |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
Build  : 34e226dfe3-20211220 OPENFOAM=2206 version=v2206
Arch   : "LSB;label=32;scalar=64"
Exec   : C:\Program Files\ESI-OpenCFD\OpenFOAM\v2206\msys64\home\ofuser\OpenFOAM\OpenFOAM-v2206\platforms\win64MingwDPInt32Opt\bin\reconstructPar.exe
Date   : Jun 20 2024
Time   : 13:01:37
Host   : DESKTOP-1URTOLS
PID    : 13308
I/O    : uncollated
Case   : C:/Users/MONSTER/AppData/Local/Temp/case
nProcs : 1
trapFpe: Floating point exception trapping enabled (FOAM_SIGFPE).
fileModificationChecking : Monitoring run-time modified files using timeStampMaster (fileModificationSkew 5, maxFileModificationPolls 20)
allowSystemOperations : Allowing user-supplied system call operations

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
Create time



Reconstructing fields
region=region0
```

```
Time = 0.0001

Reconstructing FV fields

    Reconstructing volScalarFields

        Q
        epsilon
        k
        nut
        p

    Reconstructing volVectorFields

        U

    Reconstructing surfaceScalarFields

        meshPhi
        phi

    Reconstructing surfaceVectorFields

        Uf

Reconstructing point fields

No point fields

No lagrangian fields

No FA fields
```

```
Time = 0.0002

Reconstructing FV fields

    Reconstructing volScalarFields

        Q
        epsilon
        k
        nut
        p

    Reconstructing volVectorFields

        U

    Reconstructing surfaceScalarFields

        meshPhi
        phi

    Reconstructing surfaceVectorFields

        Uf

Reconstructing point fields

No point fields

No lagrangian fields

No FA fields
```

8.2. Analysis Files

blockmeshDict

```
/*--------------------------------*- C++ -*----------------------------------*\
  =========                 |
  \\      /  F ield          | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration      | Website:  https://openfoam.org
    \\  /    A nd            | Version:  7
     \\/     M anipulation   |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

convertToMeters 1;

vertices
(
    (-0.3 -0.81 -0.3)
    ( 0.3 -0.81 -0.3)
    ( 0.3  0.21 -0.3)
    (-0.3  0.21 -0.3)
    (-0.3 -0.81  0.3)
    ( 0.3 -0.81  0.3)
    ( 0.3  0.21  0.3)
    (-0.3  0.21  0.3)
);

blocks
(
    hex (0 1 2 3 4 5 6 7) (12 20 12) simpleGrading (1 1 1)
);

edges
(
);

boundary
(
    walls
    {
        type wall;
        faces
        (
            (2 6 5 1)
            (0 3 2 1)
            (0 4 7 3)
            (4 5 6 7)
        );
    }
    inlet
    {
        type patch;
        faces
        (
            (3 7 6 2)
        );
    }
    outlet
    {
        type patch;
        faces
        (
            (1 5 4 0)
        );
    }
);

// ************************************************************************* //
```

controlDict

```
/*--------------------------------*- C++ -*----------------------------------*\
|                                                                             |
|                  Generated by the CfdOF workbench for FreeCAD               |
|                       https://github.com/jaheyns/CfdOF                      |
|                                                                             |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     pimpleFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         0.1;

deltaT          1e-6;
writeControl    adjustableRunTime;
writeInterval   0.0001;

////- For testing with moveDynamicMesh
//deltaT          0.01;
//writeControl    timeStep;
//writeInterval   1;

purgeWrite      0;

writeFormat     binary;
|
writePrecision  6;


}

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

adjustTimeStep  yes;

maxCo           1;

functions
{
    #includeFunc Q
    #include "surfaces"
    #include "forces"
}
```

turbulenceProperties

```
/*--------------------------------*- C++ -*--------------------------------*\
|                                                                          |
|               Generated by the CfdOF workbench for FreeCAD               |
|                     https://github.com/jaheyns/CfdOF                     |
|                                                                          |
\*--------------------------------------------------------------------------*/
FoamFile
{
    version    2.0;
    format     ascii;
    class      dictionary;
    object     turbulenceProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

simulationType RAS;

RAS
{
    RASModel            kEpsilon;

    turbulence          on;

    printCoeffs         on;
}
```

U

```
/*--------------------------------*- C++ -*--------------------------------*\
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  7
     \\/     M anipulation  |
\*--------------------------------------------------------------------------*/
FoamFile
{
    version    2.0;
    format     ascii;
    class      volVectorField;
    location   "0";
    object     U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (0 0 -20);
```

```
boundaryField
{
    //- Set patchGroups for constraint patches
    #includeEtc "caseDicts/setConstraintTypes"

    inlet
    {
        type            fixedValue;
        value           uniform (0 0 -20);
    }

    outlet
    {
        type            inletOutlet;
        inletValue      uniform (0 0 0);
        value           uniform (0 0 0);
    }

    Ground
    {
        type            noSlip;
    }

    Rotor
    {
        // Specified velocity, only component tangential to wall is used
        type        rotatingWallVelocity;
        origin      (0.0 0.0 0.0);
        axis        (0.0 0.0 1.0);
        omega       100;
        }
open
    {
        type            zeroGradient;

    }

}


// ************************************************************************* //
```

boundary

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield          | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration      | Version:  2206                                  |
|   \\  /    A nd            | Website:  www.openfoam.com                      |
|    \\/     M anipulation   |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      binary;
    arch        "LSB;label=32;scalar=64";
    class       polyBoundaryMesh;
    location    "constant/polyMesh";
    object      boundary;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

5
(
    inlet
    {
        type            patch;
        nFaces          400;
        startFace       288253;
    }
    open
    {
        type            patch;
        nFaces          6497;
        startFace       288653;
    }
    outlet
    {
        type            patch;
        nFaces          400;
        startFace       295150;
    }
    Ground
    {
        type            wall;
        inGroups        1(wall);
        nFaces          2167;
        startFace       295550;
    }
    Rotor
    {
        type            wall;
        inGroups        1(wall);
        nFaces          3360;
        startFace       297717;
    }
)
```
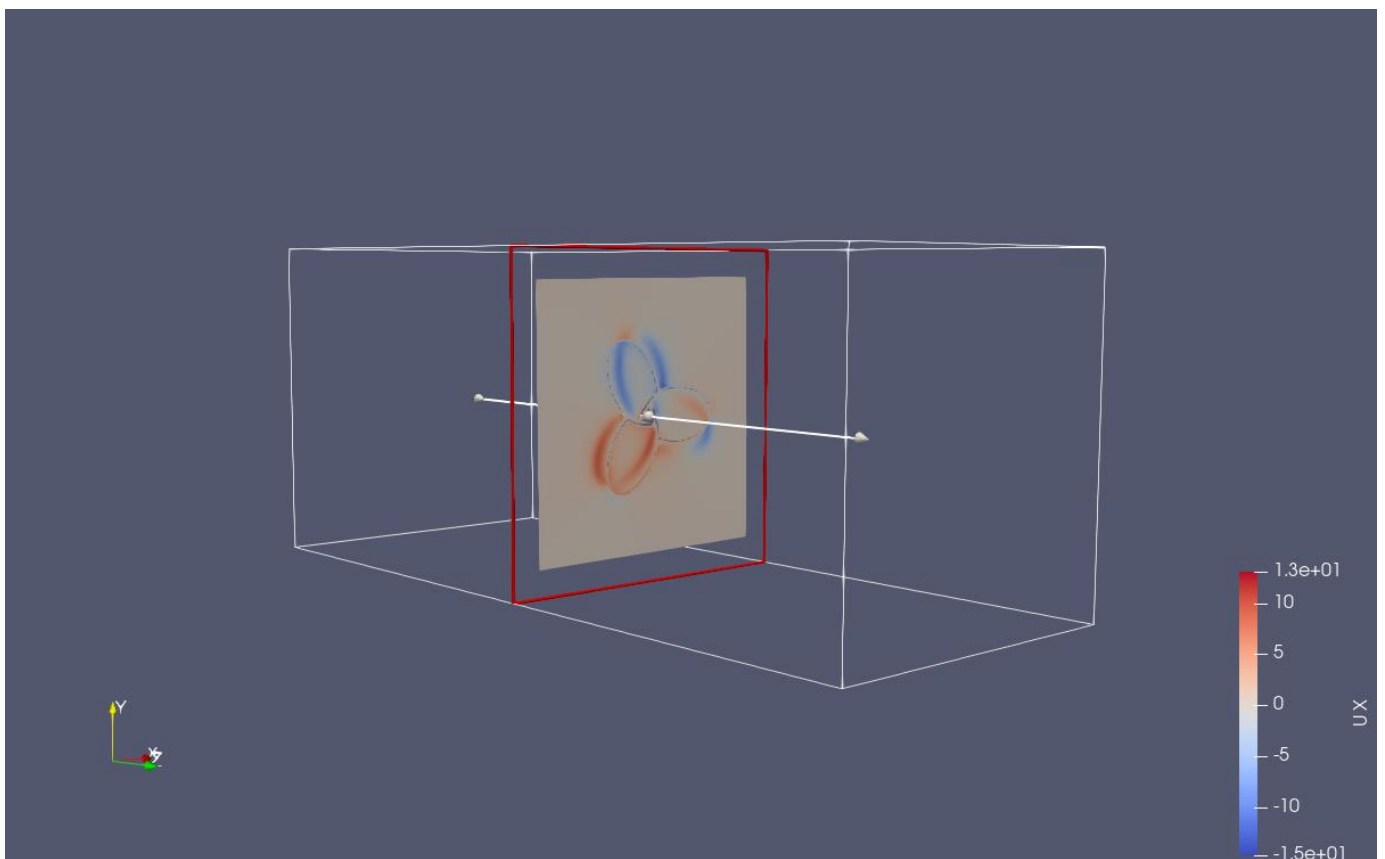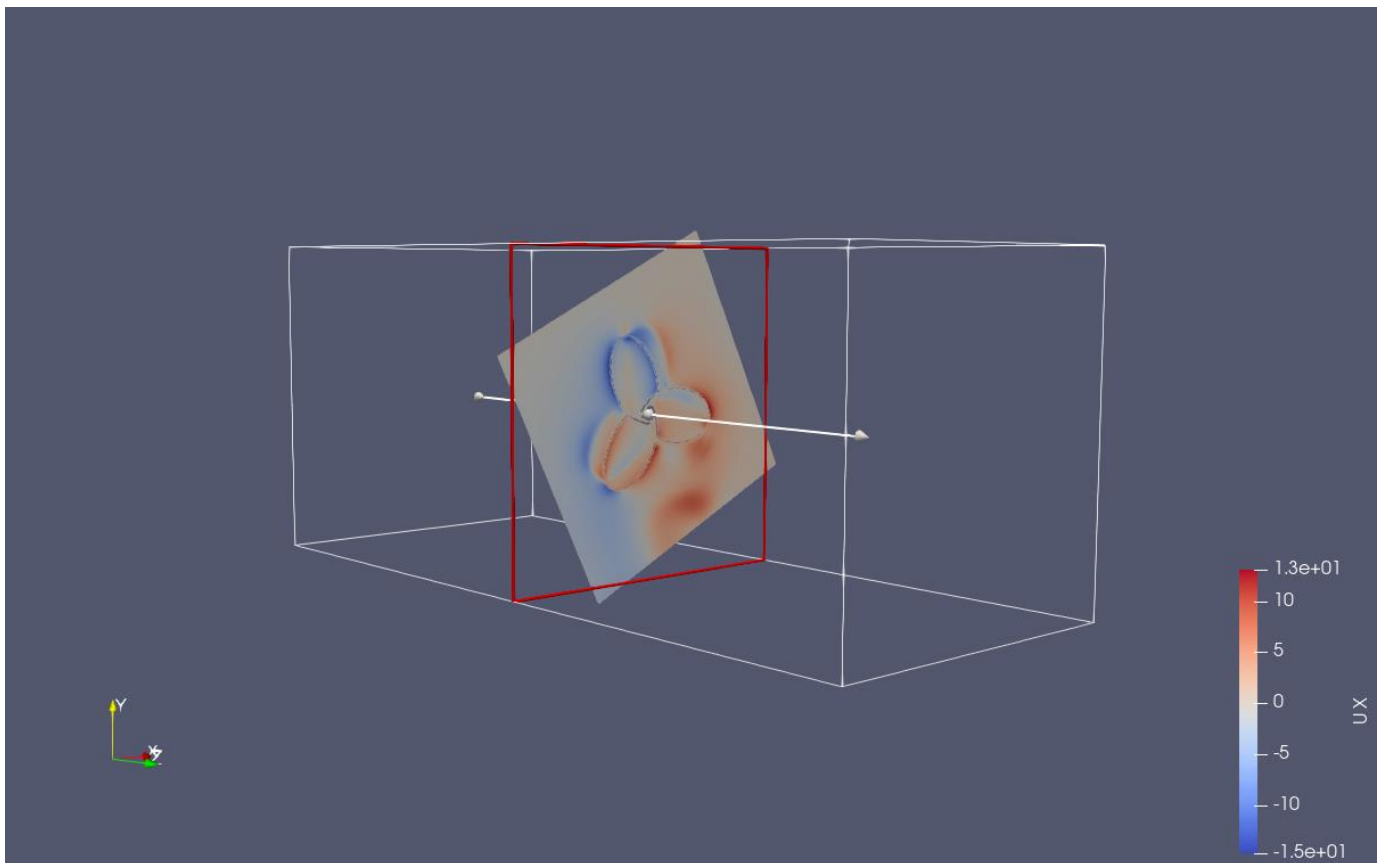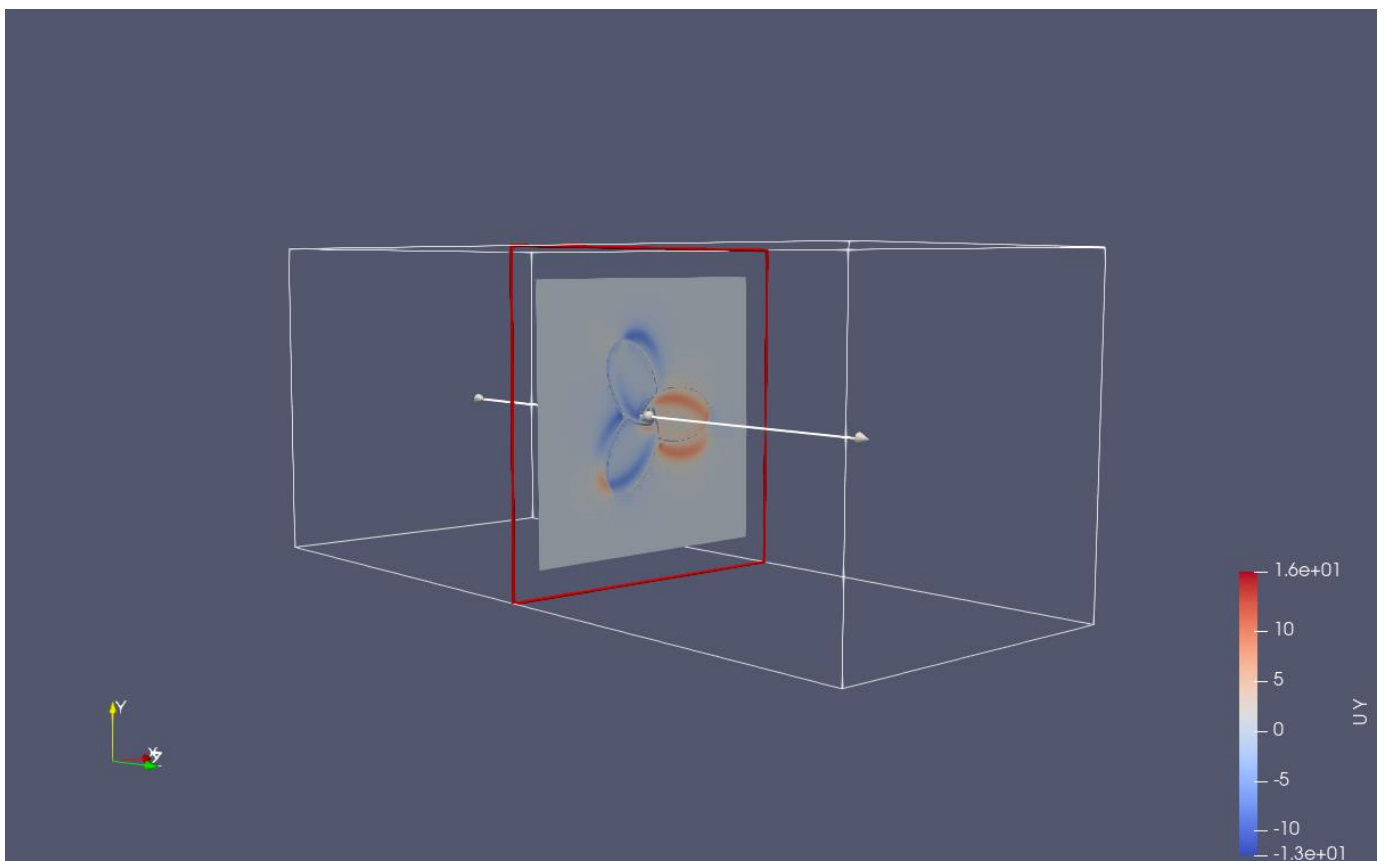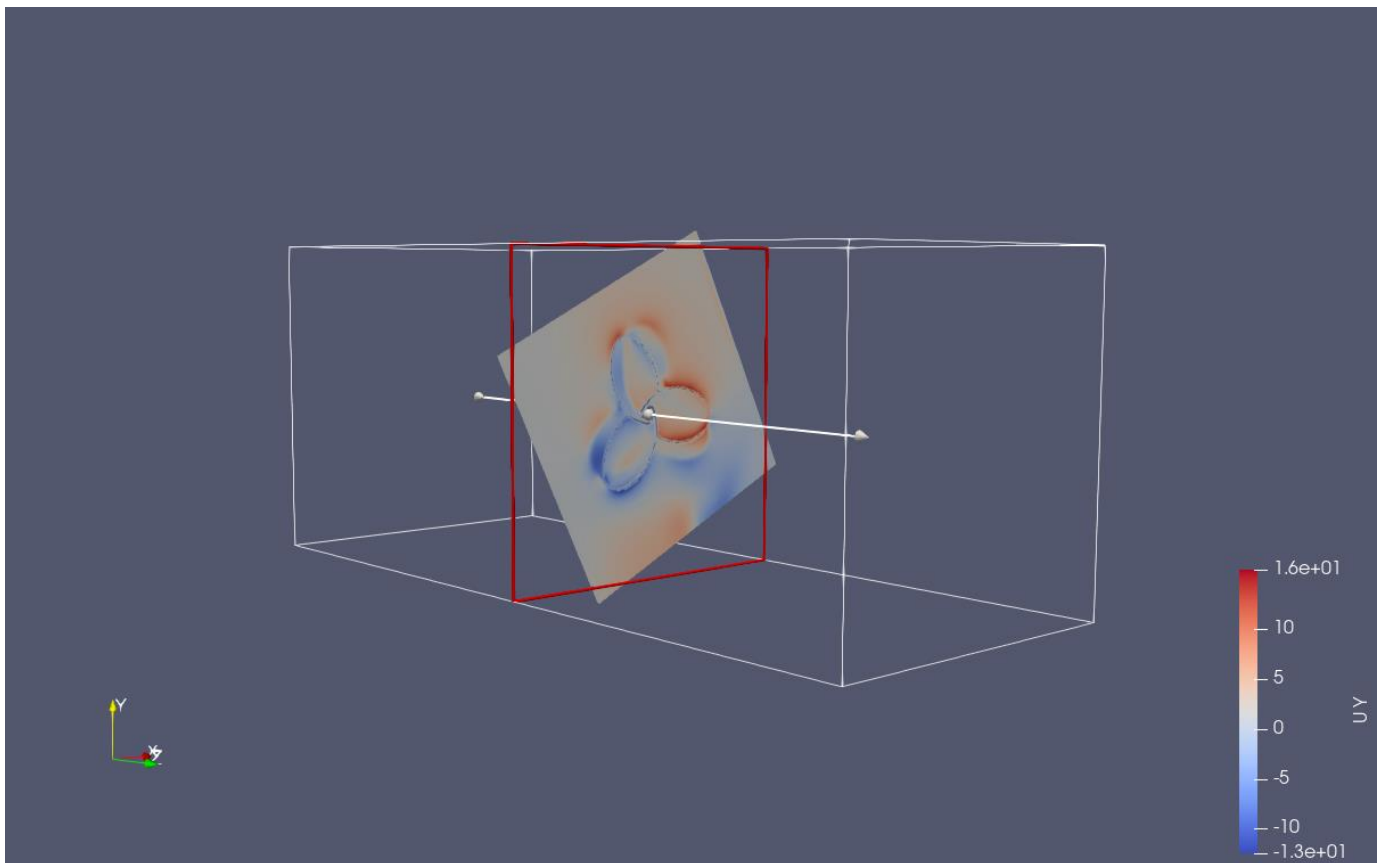
Streamlines
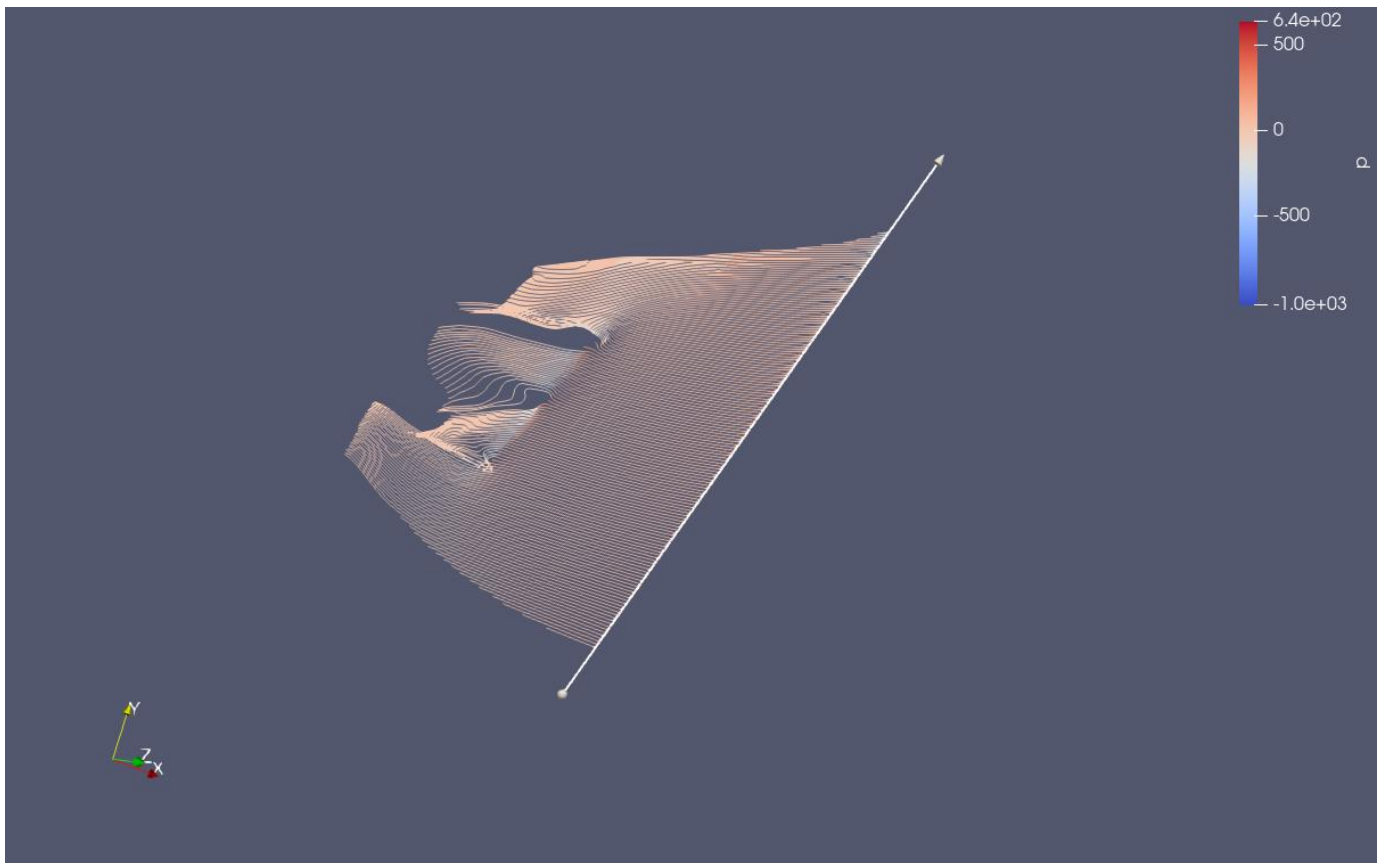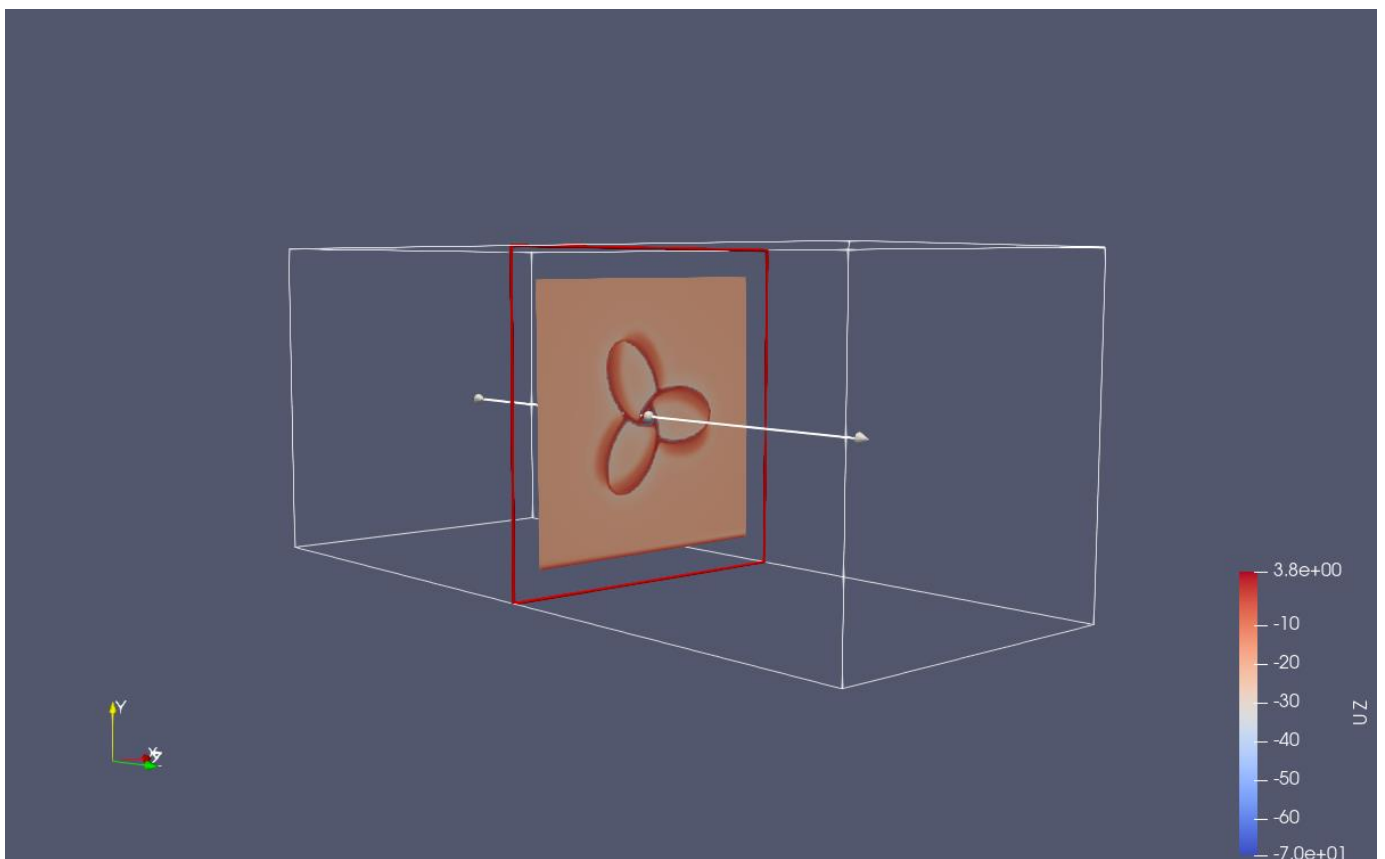

X axis at the beginning

X axis at the end

Y axis at the beginning

Y axis at the end

Z axis at the beginning

Z axis at the end