# Moving Boundary Finite Element Fluid Flow Simulation Methods for Marine Propulsion Systems

# Finite Elemente Methoden mit bewegten Rändern für Strömungssimulationen von maritimen Antriebssystemen

Von der Fakultät für Maschinenwesen der Rheinisch-Westfälischen Technischen Hochschule Aachen zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften genehmigte Dissertation

vorgelegt von
Eric Borrmann

Berichter: Univ.-Prof. Marek Behr, Ph. D.
Univ.-Prof. Dr.-Ing. Holger Schüttrumpf
Tag der mündlichen Prüfung: 24. Juni 2016

# Vorwort

Die vorliegende Arbeit ist ein Ergebnis meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Computergestützte Analyse Technischer Systeme (CATS) der RWTH Aachen. Zu allererst möchte ich denjenigen Menschen danken, welche zu dem hervorragenden Ruf der RWTH Aachen beigetragen haben und immer noch beitragen. Zu diesen Menschen gehört insbesondere mein Doktorvater, Herr Professor Marek Behr, Ph.D. Dank ihm ist die RWTH Aachen hinsichtlich der Wissenschaft im Bereich des Hochleistungsrechnens und der Fluidsimulationen mit finiten Elementen bereichert worden. Ich bin sehr dankbar, dass er mir die Möglichkeit gegeben hat, in diesem Gebiet forschen und meine Doktorarbeit erstellen zu können. Dank seiner großen Unterstützung vor allem in der Einarbeitungsphase, in welcher er mir und meinen Kollegen die Teilnahme an Vorlesungen ermöglichte, hatte ich die Chance, mich tief in das für mich neue Wissenschaftsgebiet einzuarbeiten. Danken möchte ich ihm auch für die Freiheit bei der Auswahl des Promotionsthemas, was mich besonders motivierte.

Besonderer Dank gilt auch meinen Kollegen vom Lehrstuhl CATS, welche mir immer mit gutem Rat zur Seite standen. Hier möchte ich besonders Mike Nicolai danken, welcher von Anfang an ein tiefes Verständnis für das Kompilieren und Rechnen auf Großrechnern in Jülich hatte und dies auch immer an seine Kollegen vermittelte.

Weiterhin möchte ich mich bei allen Studenten bedanken, mit denen ich im Rahmen der Betreuung von Studien- und Diplomarbeiten zusammen arbeiten durfte und die mich bei meiner täglichen Arbeit am Lehrstuhl CATS unterstützten.

Zu guter Letzt möchte ich mich natürlich bei meiner Familie bedanken, welche mich zu jeder Zeit unterstützte und mir Sicherheit gab.

Aachen, im September 2015 Eric Borrmann

# Summary

In this thesis, a moving boundary finite element fluid flow simulation method for a propulsion system of a Voith Water Tractor (VWT) consisting of two Voith Schneider propellers (VSP) and a nozzle plate is developed. In order to perform a fluid flow simulation around the moving parts of the VWT, such as the counterrotating VSPs and the single blades of each VSP, a moving finite element mesh is constructed. A Shear-Slip Mesh Update Method (SSMUM) is used to follow the motion of each VSP whereas an Elastic Mesh Update Method (EMUM) captures the rotation of each single blade of a VSP.

The EMUM approach regards the finite element mesh as an elastic solid and solves the linear elasticity equation for the mesh node motions. Therefore, the element deformation cannot be controlled explicitly and this can lead to tangling elements in regions of high relative motion of neighbouring elements. This is especially the case in a critical region between the bottom of the blades and the nozzle plate, which is situated just 2.6 centimetres below the blades of the VSPs. While trying to partly prescribe the EMUM node motion in the critical region in order to prevent the tangling of elements, a Concentric Shell Mesh Motion (CSMM) approach is worked out in parallel.

The CSMM approach is designed such that the motion of every mesh node is explicitly prescribed and tangling of an element of the mesh is impossible. However, the CSMM concept demands a higher effort in programming the mesh generation compared to the EMUM approach. In order to directly compare these two alternative mesh motion concepts, a comparative refinement study is performed wherein an unsteady Navier-Stokes flow around a fixed blade of the VSP is calculated. Compared to the EMUM concept, worse elements in the critical region below the blades are created with the CSMM concept. This contributes to a worse convergence behaviour of the solution of the Navier-Stokes equations. Therefore, the EMUM approach is selected for the full ten blade VWT proplulsion system unsteady Navier-Stokes simulation.

A refinement study is performed for the unsteady Navier-Stokes simulation around the VWT proplulsion system as well. Here, similar mesh resolutions around the blade compared to the single blade refinement study are used. The two finest meshes show an average force value of each single blade and also a characteristic of each force value that agree in the range of five to eight percent. The average thrust of a single blade varies by up to twelve percent compared to the measured and simulated data by Voith.

Beside the flow simulation of the VWT propulsion system, a method for the simulation of a free-surface flow simulation around a hull of the VWT is presented. According to this method the surface of the hull is divided into single subareas. The geometry of the subareas are approximated by the use of artificial neural networks. Using this approach, the implementation of a complicated CAD spline based ship hull geometry can be avoided.

# Zusammenfassung

In der vorliegenden Arbeit wird eine Methode zur Fluidsimulation eines Antriebssystems eines Voith Wassertraktors (VWT) entwickelt. Das Antriebssystem besteht aus zwei Voith-Schneider-Propellern (VSP) und einer Schutzplatte. Um eine Fluidsimulation einer Strömung um die beweglichen Teile des VWT, wie die gegenläufig rotierenden VSP und die einzelnen Rotorblätter jedes VSP, durchzuführen, wird ein bewegliches Netz aus finiten Elementen konstruiert. Für eine Simulation einer Drehung eines einzelnen VSP wird eine Shear-Slip Mesh Update Methode (SSMUM) verwendet. Eine Simulation einer Drehung einzelner Rotorblätter der VSP erfolgt mithilfe einer Elastic Mesh Update Methode (EMUM).

Bei der Anwendung des EMUM-Konzeptes werden das Finite-Elemente-Netz als ein elastischer Festkörper betrachtet und Gleichungen nach der linearen Elastizitätstheorie gelöst. Dadurch sind Verformungen von Elementen des Netzes nicht explizit kontrollierbar, was zu degenerierten Elementen in Bereichen führen kann, wo eine hohe relative Bewegung von benachbarten Elementen herrscht. Dies ist insbesondere in einem kritischen Bereich zwischen den Unterseiten der Rotorblätter und der Düsenplatte der Fall, welche nur 2,6 cm unterhalb der Rotorblätter angeordnet ist. Während versucht wird, in dem kritischen Bereich zum Teil Knotenbewegungen vorzugegeben, um eine Degeneration von Elementen zu verhindern, wird zusätzlich ein Ansatz einer Concentric Shell Mesh Motion (CSMM) verfolgt.

Das CSMM-Konzept ist so konzipiert, dass eine Bewegung jedes einzelnen Knotens des Netzes explizit vorgegeben ist und eine Degeneration eines Elementes unmöglich ist. Jedoch erfordert das CSMM-Konzept einen im Vergleich zum EMUM-Konzept höheren Programmieraufwand für die Netzgenerierung. Das CSMM-Konzept wird mit dem EMUM-Konzept im Rahmen einer Netzverfeinerungsstudie verglichen. Bei dieser Studie wird eine instationäre Navier-Stokes-Strömung um ein einzelnes verdrehtes Rotorblatt eines VSPs berechnet. Im Vergleich zum EMUM-Konzept werden bei der Anwendung des CSMM-Konzeptes Elemente mit einer schlechteren Qualität in dem kritischen Bereich unterhalb der Rotorblätter erzeugt. Dies verursacht ein schlechteres Konvergenzverhalten einer Lösung der Navier-Stokes-Gleichungen. Deshalb wird für die Fluidsimulation des kompletten Antriebssystems des VWT mit zehn Rotorblättern das EMUM-Konzept ausgewählt.

Um die instationäre Navier-Stokes-Strömung um das VWT Antriebssystem zu berechnen, wird eine weitere Netzverfeinerungsstudie durchgeführt. Hierbei werden ähnliche Netzauflösungen in den Bereichen um die Rotorblätter wie bei der Netzverfeinerungsstudie zur Strömung um das einzelne verdrehte Rotorblatt verwendet. Die berechneten Strömungsfelder, welche mit den zwei am höchsten aufgelösten Netzen erzielt werden, stimmen annähernd überein. Eine jeweils mit diesen Netzen berechnete durchschnittliche

Kraft jeweils eines einzelnen Rotorblattes sowie ein jeweiliger zeitlicher Verlauf dieser Kraft zeigen Übereinstimmungen im Bereich von fünf bis acht Prozent. Die durchschnittliche Kraft eines Rotorblattes weicht um bis zu zwölf Prozent von Mess- und Simulationsdaten aus dem Hause Voith ab.

Neben der Strömungssimulation des VWT Antriebssystems wird eine Methode zur Berechnung einer freien Oberfläche um einen Schiffsrumpf des VWT beschrieben. Dabei wird die Oberfläche des Schiffsrumpfes in einzelne Teilflächen aufgeteilt und eine jeweilige Geometrie der Teilflächen mithilfe von künstlichen Neuronalen Netzen approximiert. Durch diesen Ansatz kann auf eine Implementierung einer mithilfe von Splinefunktionen beschriebenen komplexen Schiffrumpfgeometrie verzichtet werden.

# Contents

# 1 Introduction

In the last decades, huge advances have been made by computational fluid dynamics (CFD) and its application. On one hand, computing resources have expanded much in facilities, from in-house clusters in companies, over big clusters in many universities, to the development of supercomputing centers such as in Jülich or Stuttgart, just to name those in Germany. On the other hand, engineering research and development activities have become more complex compared to those twenty years ago. By now, most machines and applications in the field of mechanical engineering have reached a high level of complexity. The functions and behaviour of complex machines in various operating conditions can only be fully understood and optimized when these machines are tested in these conditions. Such tests must include all possible parameters and their modifications with respect to their influence on the total outcome, such as fuel consumption. Taking a propulsion system of a ship with only seven varying parameters, e.g., the number of propeller blades, the dimensions of length, width and height, the profile, the rotation per minute of the propeller blades, and the velocity of the propelled ship, one can imagine a huge number of possible parameter value combinations. Given, e.g., the ship's velocity, the optimal design of the propeller shape and its rotation speed would require an enormous number of tests to optimize the whole system to operate at that velocity. Limited time and budget conditions usually restrict these optimization processes to only a reduced number of tests, such that only a very tiny amount of the possible parameter value combinations could be captured by the testing. In such cases, the use of fluid flow simulations can considerably contribute to simulate additional tests in the parameter space where no real tests could have been performed due to limited time. Error sources due to scaling effects when performing the tests can be detected as well by performing additional fluid flow simulations. CFD simulation is not a straightforward task, depending strongly on the complexity of the flow and the object exposed to the flow. This becomes evident in this thesis, which reports on experiences in the fluid flow simulation around a Voith Water Tractor (VWT) propulsion system. On one hand, the thesis presents the necessary theory to understand the author's simulation approach; on the other it focuses on the individual intermediate steps and the associated difficulties on the road to a fluid flow simulation around a VWT propulsion system consisting of two counterrotating Voith Schneider propellers and a nozzle plate.

Chapter 2 describes the physical governing equations, which are simulated by the fluid flow simulation. These are the Navier-Stokes equations wherein water is treated as an incompressible fluid. In Chapter 3, a general overview of propeller propulsion systems for ships is given, including a glimpse of the evolution of propeller design, starting from a design known as the Archimedian screw to the propulsive propeller of today. In Chapter 4, the Voith Schneider propeller (VSP) is presented. The unbreakable will of Ernst

Schneider contributed to the innovation of this kind of propulsion system. The main idea of Ernst Schneider was to build a ship propulsion system with blades performing a similar motion as the fins of a swimming catfish. How he managed to transfer his ideas to a mechanical reality is also explained in Chapter 4.

Chapter 5 explains the function of the Voith Water Tractor propulsion system, consisting of two counterrotating VSPs and a nozzle plate. The nozzle plate is situated very close to the single VSP such that only a small gap between the nozzle plate and the VSP is built and thus a good duct performance is achieved. However, the small gap causes difficulties when a fluid flow simulation in the region between the moving VSPs and the static nozzle plate is performed. On one hand, moving boundaries that are situated very close to each other present difficulties in the mesh generation and the mesh motion. On the other hand, the solution of a fluid flow problem with moving boundaries can be difficult, depending on the applied discretization scheme. Therefore, a suitable mesh moving scheme has to be chosen in combination with an adequate discretization approach. In Chapter 6, the basic discretization methods are described, including the finite difference method, the finite volume method, the finite element method and the space-time finite element method.

In Chapter 7, the common moving boundary methods are presented. The disadvantages of the application of fixed mesh methods to moving boundary problems are described briefly. In a more detailed view different moving mesh methods are explained. For each method it is shown how a moving interface can be tracked.

The sliding, the clicking and the chimera mesh methods are presented, as well as the arbitrary Lagrangian-Eulerian method. In addition, the shear-slip mesh update method (SSMUM) and the elastic mesh update method (EMUM) are explained. The shear-slip mesh update method is very adequate to capture infinite rotational motions, whereas the elastic mesh update method is very useful to capture elastic deformations like squeezing or stretching. For the elastic mesh update method, different mesh adaptation algorithms are presented.

In Chapter 8 it is described how the SSMUM moving boundary technique is applied to the VWT propulsion system. Herein, each single VSP of the VWT propulsion system is encapsulated by a rotating SSMUM beaker mesh. The description of the SSMUM application comprises not only the construction of the single moving SSMUM mesh elements, but also the intermediate steps of the mesh generation: starting from a CAD-data file of the VWT propulsion system, via the surface meshes of the blades of each single VSP and the nozzle plate of the VWT created with the mesh generation software *Gridgen*, to the final volume mesh including the SSMUM elements. Hereby, each single data format that is used to create the final volume mesh is mentioned and a tool chain is given to show which tools are used to convert the data from each data format to another.

In Chapter 9, the application of the EMUM is described. The EMUM approach is applied to capture the rotational motion of the single blades of each VSP by surrounding SSMUM beaker mesh deformation elements. With the application of the EMUM technique, the deformation of each mesh element is not prescribed explicitly but is gained implicitly by solving the elasticity equilibrium equation for the mesh. The boundary conditions are given by the individual blade motions of the VSP blade surfaces inside

the rotating SSMUM beaker mesh. The large relative motion between a single blade and the SSMUM beaker has to be compensated by an elastic mesh deformation. This elastic mesh deformation is extremely complex, as there is only a tiny deforming computational mesh between the tips of the blades and the SSMUM beaker. This computational mesh cannot be enlarged, as the nozzle blade is situated only 2.6 cm away from the tips of the blades. Inside that tiny computational mesh, tangling of highly deformed elements occurs often, such that a fluid flow simulation on such a mesh cannot be performed. Several different surrounding meshes and their corresponding deformations are generated and tested to achieve a proper deforming mesh motion without tangling.

Because of the risk of tangling elements, a second mesh deformation technique, the Concentric Shell Mesh Motion (CSMM), is developed to capture the rotational motion of the single blades of a VSP in Chapter 10. Herein the nodes of the computational mesh around a blade are situated on moving concentric shells. The motion of every node on a shell is prescribed explicitly. By that mesh design and mesh motion around the blades, the phenomenon of tangling elements can be prevented.

In Chapter 10, several different shell designs are presented. To be able to chose the best mesh deformation algorithm for the mesh deformation around the single blades of a VSP, a tool for the evaluation of the mesh quality is presented in Chapter 11. For the fluid flow simulation, the best possible mesh quality is required, as a low mesh quality results in a low convergence rate of the fluid flow simulation on that mesh. Chapter 11 describes how the mesh evaluation tool is used to improve the mesh deformation design around a single blade with respect to the EMUM technique.

To compare the convergence behaviour of the two different methods, the EMUM and the SSMUM method, a detailed refinement study of the fluid flow around a single blade is performed in chapter 12. Here, the single blade is deflected such that the finite elements below the blade get in their worst possible position regarding the simulation of one full rotation of one VSP. The result of the refinement study suggests which of the two mesh moving concepts is most suitable for the full ten blade simulation including the nozzle plate.

Here, the main selection criteria is the convergence behaviour of the in-house Navier-Stokes Solver (XNS) on the different meshes. In Chapter 13 the best mesh concept is applied to the simulation of the fluid flow around the complete VSP propulsion system. In that context, another refinement study is performed and the resulting flow fields obtained from each mesh are discussed. These results are compared to the experimental and the simulation data given by Voith. Subsequently, the deviation of the results from the reference data is discussed.

In the view of a fluid flow simulation comprising not only the VSP propulsion system but also the hull of the Voith Water Tractor, an interface tracking scheme to follow the wave motion along a ship hull is presented in Chapter 14. This scheme is based on neuronal networks and circumvents the difficulty of a representation of splines, which define the ship hull shape in the corresponding CAD model, in the software code. As a first test, the wave motion is visualized along a fictitious ship hull. Chapter 15 gives a summary and an outlook on further development.

# 2 Equations of Motion for Incompressible Fluid Flow

Before the subject of simulation, i.e. the propulsion system of the VWT, is presented, the mathematical equations used to express the physical motion of water will be introduced. These are the Navier-Stokes equations for incompressible fluids, also known as the equation for the conservation of momentum. Claude-Louis Navier and George Gabriel Stokes independently formulated the conservation of momentum for viscous fluids in the first half of the nineteenth century. Within the VSP simulation approach for the motion of water, water is treated as an incompressible fluid. The water molecules fill a tiny water domain $\Omega_t \subset R^{n_{sd}}$ bounded by $\Gamma_t$ at an instant of time $t \in (0, T)$, where $n_{sd}$ symbolizes the number of space dimensions. The bounded region $\Omega_t$ can be of arbitrary size and this generality is the key concept of the two most common spatial discretization methods. For all places inside the domain $\Omega_t$ the physical quantities like the velocity and the pressure, $\mathbf{u}(\mathbf{x}, t)$ and $p(\mathbf{x}, t)$, fulfil the conservation of mass and impulse at any instant of time $t \in (0, T)$. The mathematical equation maintaining the conservation of mass for water, which is equivalent to the condition of incompressibility of water, reads as follows:

$$\nabla \cdot \mathbf{u} = 0 \quad \text{on } \Omega_t \quad \forall t \in (0, T). \tag{2.1}$$

The equation for the conservation of momentum of water is:

$$\rho \left( \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} - \mathbf{f} \right) - \nabla \cdot \boldsymbol{\sigma} = \mathbf{0} \quad \text{on } \Omega_t \quad \forall t \in (0, T), \tag{2.2}$$

where $\rho$ is the constant fluid density and $\mathbf{f}(\mathbf{x}, t)$ can be regarded as the sum of all external forces , e.g., a gravitational force field. For water, the stress tensor $\boldsymbol{\sigma}$ expresses on the one hand the forces acting among the water particles due to friction, which can be ascertained in all directions. On the other hand, the pressure only acts normally to a water domain surface. Taking this into account, the stress tensor for water can be written as:

$$\boldsymbol{\sigma} = -p\mathbf{I} + \mathbf{T}, \tag{2.3}$$

$$\mathbf{T} = 2\mu\boldsymbol{\varepsilon}(\mathbf{u}), \tag{2.4}$$

where the viscous frictional part is defined by the viscosity of water $\mu$ and the strain rate tensor $\boldsymbol{\varepsilon}(\mathbf{u})$, which quantifies the variations of all the three velocity components in all three physical directions:

$$\boldsymbol{\varepsilon}(\mathbf{u}) = \tfrac{1}{2}(\nabla \mathbf{u} + (\nabla \mathbf{u})^T). \tag{2.5}$$

Fluids with a constant kinematic viscosity like water are named Newtonian fluids.

According to the application of each fluid flow simulation special boundary conditions have to be described at the boundary $\Gamma_t$. These can refer either to the value of the velocity, called Dirichlet-type boundary condition $g_d$, in any direction or the stress in any direction, called Neumann-type boundary condition $h_d$:

$$\mathbf{u} \cdot \boldsymbol{e}_d = g_d \quad \text{on } (\boldsymbol{\Gamma_t})_{\boldsymbol{g},d} \ , \quad d = 1...n_{sd}, \tag{2.6}$$

$$\mathbf{n} \cdot \boldsymbol{\sigma} \cdot \boldsymbol{e}_d = h_d \quad \text{on } (\boldsymbol{\Gamma_t})_{\boldsymbol{h},d} \ , \quad d = 1...n_{sd}, \tag{2.7}$$

where $(\Gamma_t)_{\boldsymbol{g},d}$ , and $(\Gamma_t)_{\boldsymbol{h},d}$ , are complementary subsets of the boundary $\Gamma_t$, considering each spatial dimension separately, and $\boldsymbol{e}_d$ is a basis in $R^{n_{sd}}$. Assuming the basis coincides with the local directions normal and tangent to the boundary, then for a typical inflow boundary condition all three dimensions of $\Gamma_t$ are of Dirichlet-type, so that certain values are assigned to each velocity direction on the inflow boundary section. Regarding a typical outflow boundary condition, the stresses in all directions are set to zero or, but more uncommon, to another value, so that all three dimensions of $\Gamma_t$ are of Neumann-type. A further condition for solving a Navier-Stokes equation system is that the initial velocity values must represent a divergence-free velocity field specified over the entire domain:

$$\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0, \quad \nabla \cdot \mathbf{u}_0 = 0 \quad \text{on } \Omega_0. \tag{2.8}$$

# 3 Ship Propulsion Systems

This chapter gives an overview of the evolution of ship propeller design and its application to propeller propulsion concepts for ships. Probably Archimedes of Syracuse (c. 250 BC) and Leonardo da Vinci (c. 1500) were not aware of the physics described by the Navier-Stokes equations. However, they must have had the idea that a lateral force on a specially shaped body was induced by moving the body through the water, and that this lateral force could propel a ship. Both had similar ideas, namely that the special shape of that said body should rotate around an axis parallel to the propulsive direction. Their designs served as examples for ship propulsion engineers many years later.

In 1681, Robert Hooke followed an idea similar to the concept of da Vinci. Hooke's propeller comprised wooden vanes geared to a central shaft, see Figure 3.1(a). The Archimedean screw, however, was designed to be a screw-shaped body attached to a rotating axle. In 1752, Alexis-Jean-Pierre Paucton picked up the concept of the Archimedean screw, see Figure 3.1(b), when he competed with other famous mathematicians and scientists of Europe, like d'Alembert, Euler and Bernoulli, in a research contest in the field of naval architecture, organized by the Académie des Sciences in Paris. Bernoulli's propeller wheel, suggested in 1752, can be seen in Figure 3.1(c).

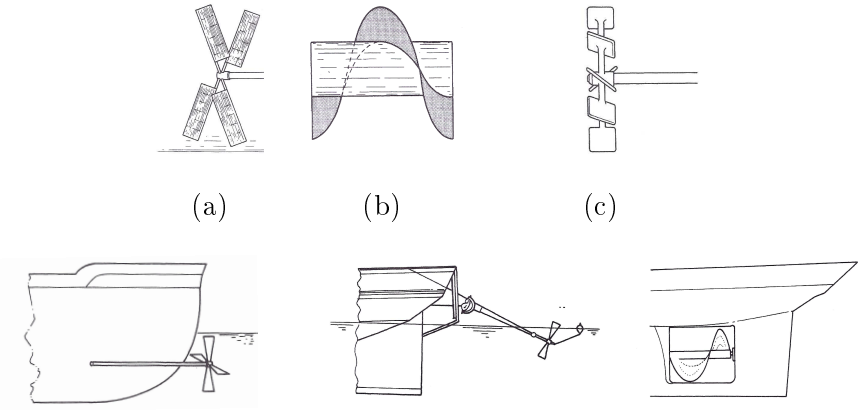



Figure 3.1: The early history of propeller designs: (a) Hooke's screw propeller (1683); (b) Archimedean screw of Paucton (1752); (c) Bernoulli's propeller wheel (1752); (d) Bramah's screw propeller design (1785); (e) Shorter's propulsion system (1802) and (f) Smith's Archimedean screw design (1839)[Car07].

Thirty years later, Joseph Bramah designed a concept for a screw propeller arranged

at the stern of a ship, which is shown in Figure 3.1(d). Different to Bernoulli's propeller wheel, it comprised a smaller number of blades. Edward Shorter varied this concept slightly, see Figure 3.1(e), and succeeded to propel the transport ship Doncaster in Gibraltar.

Further research in propeller design led not only to contra-rotating propellers, suggested by Ericsson, but also to another experiment with a propeller, which was again similar to the Archimedean screw, performed by Francis Petit Smith. During a test with that Archimedean screw concept in 1837, the screw broke such that only half of the screw remained on the propeller shaft. Immediately after the accident, the ship accelerated. As a consequence, Smith designed an improved Archimedean screw propeller with a shorter length. The outcome can be seen in Figure 3.1(f).

In parallel, Lowes developed novel propeller designs, which were also based on the Archimedean screw design, but had even a shorter length than those designs discovered by Smith. In 1838, Lowes patented a propeller comprising one or more blades, where each blade was a portion of a curve, which if continued would produce a screw. That propeller design can be regarded as the first screwed propeller design. In 1840, the first cargo ship was propelled with that screwed propeller concept. Afterwards, further research in propeller design focused on the shape and number of the blades.

In 1868, another milestone in propeller history was made by H.B. Young, who patented a method for changing the pitch of the propeller blades, and by R. Griffiths, who invented the adjustable pitch propeller. Adjustable pitch propellers are advantageous because the thrust can be changed rapidly by adjusting the angle of attack of a single blade of the propeller.

Today, the research of propeller design is still focused on the optimal shape of the propeller, especially the propeller blade. Most ships are individual in their design and therefore the flow around the propeller induced by the moving ship hull is unique. As such, often an individual propeller has to be developed for a single ship. As a consequence, there is no optimal propeller design or propeller propulsion system in general.

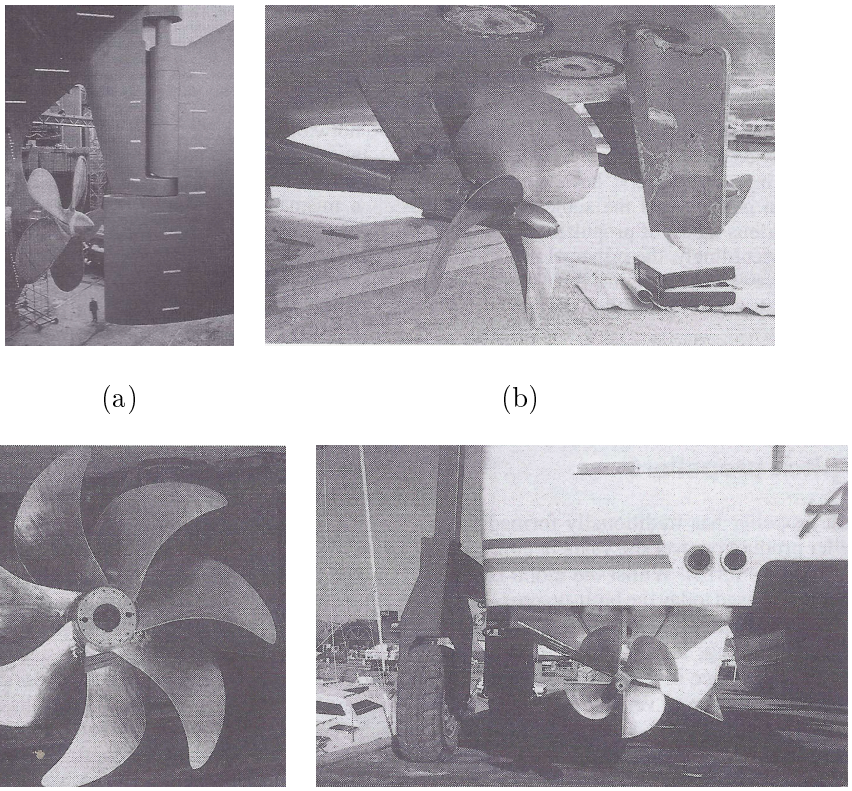
In these days, propeller propulsion systems are usually designed not only to aim at good efficiency, but also to control cavitation. In the following, the basic concepts of ship propeller propulsion systems are introduced.

## 3.1 Fixed Pitch Propellers

The most common ship propulsion system of today is a mono-block fixed pitch propeller. Contrary to the built-up propeller, the mono-block propeller is cast in one casting. The built-up propeller was used in the past for two reasons. Firstly, it was difficult to achieve appropriate large castings. Secondly, a built-up propeller allowed for blade pitch adjustment after it was built-up. Adjusting the blade pitch by trail and error was a necessity in the early years of the last century to improve the propeller performance. However, built-up propellers mostly comprise a larger boss radius than their fixed pitch counterpart. As a larger boss radius can increase the danger of cavitation in the blade root sections, and the casting technology has advanced significantly in the last decades, the

mono-block propeller design is favoured today.

The mono-block propellers of today can be used in almost every ship type, from small power-boats to large container ships. The propeller design differs considerably depending on its application, as shown in Figure 3.2. For example, a large four-bladed propeller is typically used to propel a bulk carrier, as shown in Figure 3.2(a). A propeller of a high-speed patrol craft propeller is presented in Figure 3.2(b). A highly skewed propeller, as shown in Figure 3.2(c), is commonly used for merchant and naval ships. An example of a surface-piercing propeller of a cruiser boat is given in Figure 3.2(d).



(a) (b)

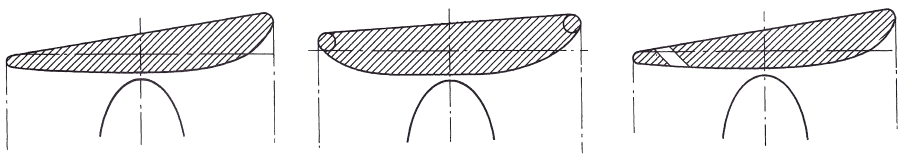
(c) (d)

Figure 3.2: Different propeller designs according to their applications: (a) large four-bladed propeller for a bulk carrier; (b) high-speed patrol craft propeller; (c) seven-bladed balanced high-screw design for merchant and naval ships and (d) surface piercing propeller of a cruiser boat [Car07].

## 3.2 Ducted Propellers

Ducted propellers are used in cases where a ship propulsion system is required to generate high thrust at low speeds, especially in towing situations. Ducted propellers consist of

a fixed pitch or controllable pitch propeller and an annular duct. The duct has an aerofoil cross-section, mostly symmetric with respect to the propeller axis. Compared to a standard non-ducted propeller, the propeller blades are modified in a way that accounts for the interaction with the flow field, which is generated by the duct. At zero ship speed the duct can contribute up to fifty percent of the thrust of the propeller propulsion system [Car07]. With increasing ship speed the contribution of the duct is less and can even lessen the whole propulsion system performance [Car07]. The design of the duct shape depends on the application. One of the most common duct designs is the Wageningen 19A form (cf. the accelerating duct in Figure 3.3(a)). It is designed to perform an efficient acceleration when cruising ahead. However, for a tug boat application the duct should be shaped such that the duct performance is efficient in both cruising directions. Such a bidirectional duct shape is the Wageningen No. 37 form (shown with the 'pull-push' duct in Figure 3.3(b)). Of course, the astern performance is increased at expense of the ahead cruising performance. Another design feature to improve the astern performance of a duct is the Hannan slot, which is shown in figure 3.3(c). When cruising ahead, the shape of the Hannan slot allows the water to pass through the duct in a similar way as through a duct with the Wageningen 19A shape. However, when cruising backwards, the slot induces a water flow through the slot, such that the effective hydraulic shape of the Hannan slotted duct resembles the hydraulic shape of the duct with the Wageningen No. 37 form.



(a) (b) (c)

Figure 3.3: Duct types: (a) accelerating duct; (b) 'pull-push' duct and (c) Hannan slotted duct [Car07].

## 3.3 Azimuthing Propellers

To enhance the manoeuvrability of a ship, it can be equipped with an azimuthing propeller, which can turn its rotational axis relative to the ship hull. This capability of manoeuvring is essential for ice breakers and very useful for cruise ships in landing operations. Azimuthing propellers can be realized as non-ducted and ducted propellers. An azimuthing ducted propeller is shown in Figure 3.4. Some types of azimuthing propellers can even perform a full rotation around their vertical axis.

Figure 3.4: Azimuthing ducted propeller [Car07].

## 3.4 Controllable Pitch Propellers

As mentioned above, controllable pitch propellers (CPP) allow an adjustment of the pitch of the blades. The adjustment is mostly performed with a hydraulic mechanism. An example is given in Figure 3.5. Here, the hydraulic mechanism can move the pull-push rod, which is situated in the propelling shaft. The amount of oil supplied to one of the two channels can be controlled by a distribution box. The corresponding motion on the rod causes an adjustment of the angle of attack of the blades, which are fastened to the rod with the securing arrangement shown in Figure 3.5. The advantages of controllable pitch propellers are numerous. First of all, the engine speed can be kept at constant speed, which offers the opportunity to work out an engine design that is optimal for one operating point. These kinds of engine designs result in a higher efficiency with less environmental pollution. Moreover, with respect to vibration issues, the hull design is easier to develop when the engine is mostly running at constant speed. In addition to that, the thrust of a controllable pitch propeller can be adjusted very easily and faster compared to a fixed pitch propeller, where the engine speed has to adjust to the required thrust. This is especially advantageous in dynamic position situations. Some CPP designs even include the opportunity to feather the blades of a CPP, which is very common with double-ended ferries or with small war ships.

Aside from all the research on the classic propeller concept, other ideas of ship propulsion arise from time to time. Among these are waterjet propulsion, the magnetohydrodynamic propulsion and propulsion systems including cycloidal propellers. One of the first cycloidal propellers is the Kirsten-Boeing propeller, invented in the 1920s. The most popular cycloidal propeller is the Voith-Schneider propeller. Its history and functionality is described in the following.

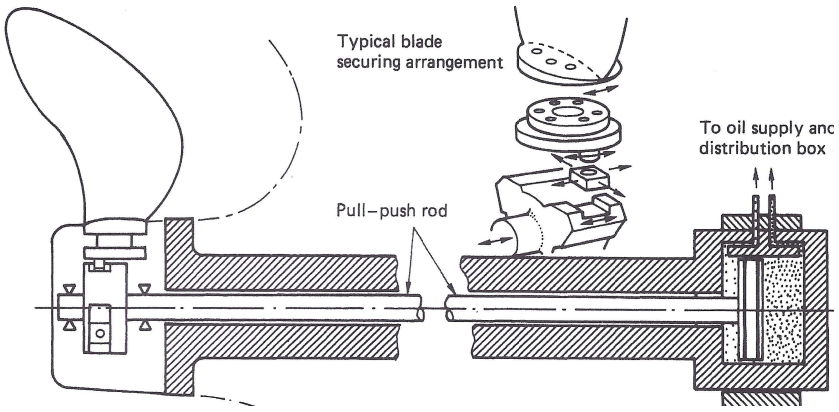
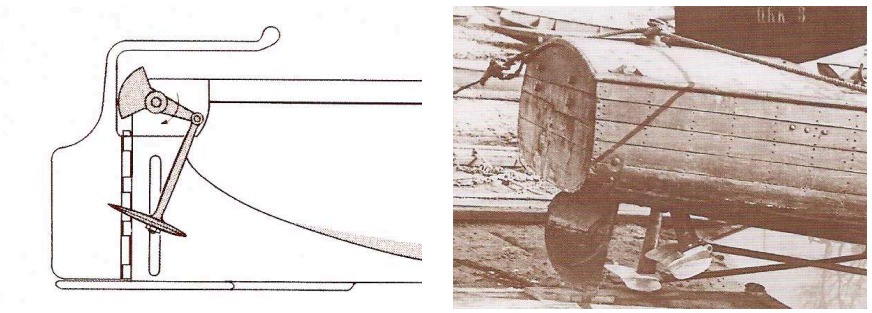



Figure 3.5: Controllable pitch mechanism [Car07].

# 4 The Voith Schneider Propeller

The innovation of the Voith-Schneider propeller (VSP) is owed to Ernst Schneider and his unceasing inventive genius. Already in the time of his studies of mechanical engineering, he wanted to contribute innovation to the world of engineering, especially in the field of ship propulsion. His early ideas lead him to a concept of a ship propeller, whose blades comprised a profile similar to a wing of a bird. With his invention he contacted Johann Matthäus Voith, the founder of the Voith company of today. Though J. M. Voith was interested in that wing type profile, Ernst Schneider had to finance on his own the first experiments to prove the efficiency of this propulsion system. The fact that he gave private lessons to finance his experiments is a proof of Ernst Schneider's passion to innovation. Comparative tests with a conventional propeller indicated that the new propeller could perform better than the conventional one. These results motivated the Voith company to perform its own tests with that wing-type profile propeller system. Unfortunately, these test results did not prove satisfactory. In retrospect, Ernst Schneider had to confess that the conventional propeller was wrongly dimensioned to give a reasonable chance to his invention. But Ernst Schneider did not give up, constantly serving new ideas to the Voith company to enhance the performance of his propulsion concept; e.g., he suggested to use a casing around the blades to achieve a nozzle effect. As the Voith company did not want to support these kinds of tests, Schneider tested the casing concept at the TH Wien with the engineer Zwerina and the support of Prof. Richard Knoller. In 1925, Zwerina showed Ernst Schneider an article about the catfish propulsion system. At that time, beside the catfish propulsion system no other ship propulsion system with a parallel blade axis was known to Ernst Schneider. In the same way as he had copied the shape of bird's wing, this time Ernst Schneider was interested in the idea of copying the motion of a fish's fin. The idea of the catfish drive is to move the blades of the drive such that their motion corresponds to the motion of a catfish fin. The principle of the catfish propulsion system can be seen in Figure 4.1(a). The rotation of the axis indicated by the arrow is transferred into a horizontal motion of the fin inside the horizontal slot combined with a swinging motion of the fin. An application of the catfish drive is shown in Figure 4.1(b).

Investigating the catfish drive, Ernst Schneider discovered that the error angle of a blade of a catfish drive reached up to thirty degrees compared to the motion a catfish fin would perform. This was due to the kinematic drive of the blades. Furthermore, this drive was not adjustable. Given these facts, Ernst Schneider started to modify the catfish drive in the following way. Firstly, the size of the blades was reduced such that the theoretical error angle was reduced. In a second step, he arranged the blades on a circle, which is called blade circle in the following. This step is considered to be the most crucial modification to the catfish drive. The blades should move on a circular path, such



(a) (b)

Figure 4.1: The catfish propulsion system: (a) principle; (b) application [BJ02].

that the normals of each blade traversed a single point region [BJ02]. The third step of the invention was done by Ernst Schneider during the preparation of his experiments. To construct the propeller he replaced the said point region with a single point. By doing so, Ernst Schneider created a crucial innovation, the control point of the VSP. In Figure 4.2, the normals of each blade are labeled $PN$ and the point of intersection of these normals is the point $N$, which is the control point of the VSP. By moving the control point, e.g., from position $N$ to $N^*$, the angles of attack of each single blade can be changed simultaneously. Therefore, the motion of the control point offered the feasibility to control the VSP as a whole. Figure 4.2 shows the adjustment of the normals of each blade according to a motion of the control point to the positions $N'$ and $N^*$.

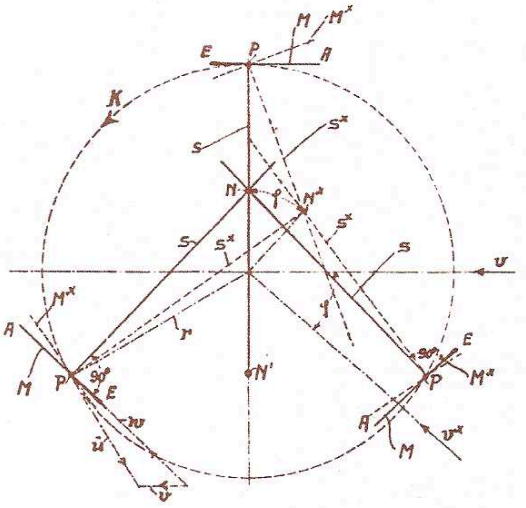
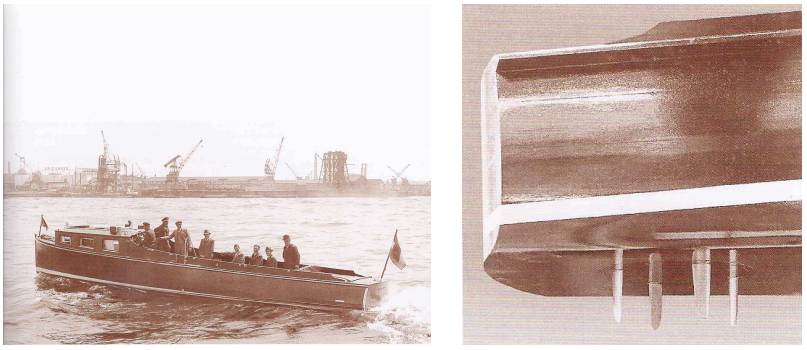


Figure 4.2: The control point of the VSP [BJ02].

As the motion of the control point could be performed in any direction, not only

the force of the propeller thrust but also the direction of the thrust could be changed during the operation of the VSP. The direction of the thrust could be even inverted. These characteristics of the VSP differentiated it from the known screw propellers. This motivated Ernst Schneider to present his invention to the Voith company. Before, he patented this special kind of ship propulsion system with the title "Schaufelrad" and asked his former professor Karl Kobes to render an expert opinion on this invention. He approved the claimed characteristics of the Schneider propulsion system according to the patent. At Voith, this new propulsion system was assessed as well by three experts. Though their opinions deviated, Walter Voith, the son of J.M. Voith, believed that this innovation of Ernst Schneider was an important one and initiated the Voith company's research of the VSP. So a team was set up to build various prototypes to test the efficiency of the VSP and to create a steering kinematic mechanism for steering the blades according to the motion of the control point. The research covered many tests, including the variation of the number of the blades, the blade shape, the steering mechanism and also the location of the propeller with respect to the ship hull. The position in relation to the ship hull was found to be crucial. Water tank tests showed that efficiency was strongly reduced if the propeller worked in a mixture of air and water. Due to this, the position of the rotating plate where the single blades were attached to, was changed to a position, where it was below the water surface. In addition, the blade shafts had to be sealed.

The VSP was tested in a ship for the first time in 1927, propelling the test boat 'Torqueo', which is shown in Figure 4.3(a). The VSP arranged below the stern of the 'Torqueo' is shown in Figure 4.3(b). The tests with the 'Torqueo' exceeded all expectations. The maneouvrability of the boat was so good, that a full turn could be performed with the boat within ten seconds. Due to these successful tests, the application of the VSP in a ship was born. Based on the innovation of Ernst Schneider, the engineers of the Voith company developed a ship propeller not known up to that time.



(a) (b)

Figure 4.3: The test boat Torqueo: (a) manoeuvring in operation; (b) VSP arragement below the stern [BJ02].

# 5 The Voith Water Tractor Propulsion System

A ship propulsion system consisting of a VSP is the interest of our fluid flow simulation. It is a propulsion system of the Voith-Water Tractor (VWT), which is one of the most famous tug boats. It consists of two Voith-Schneider propellers, a nozzle and a protection plate. As said in Chapter 4, the Voith-Schneider propeller (VSP) enables a ship to change its direction of thrust almost instantaneously. This is realized by a circulating plate at the bottom of the ship (cf. Figure 4.3(b)), on which several vertical blades are situated on a blade circle and rotate with that plate. During one full plate rotation, the blades change their angle according to their position (cf. Figure 5.1, left picture). The blade angle $\alpha$ of a single blade is measured from the tangent of the blade circle to the centre line of the blade. The different blade angles according to the varying positions of the blade result in a blade angle curve, which can vary in reference to the operating point of the ship. In the right picture of Figure 5.1 various blade angle curves are shown. The horizontal axis in the right picture of Figure 5.1 indicates the position of a single blade measured in degrees from 0 to 360 degrees. This degree range covers one full rotation of a single blade. The vertical axis indicates the blade angle $\alpha$, which is shown in the left picture of Figure 5.1. The blade angle curve varies according to the performance factor of the propeller, e.g., 0.4, 0.6 or 0.8. The right picture of Figure 5.1 shows three different blade angle curves corresponding to three different operating points, i.e., performance factors of the propeller. The operating points can be chosen such that the thrust is generated in any direction, even sidewards.

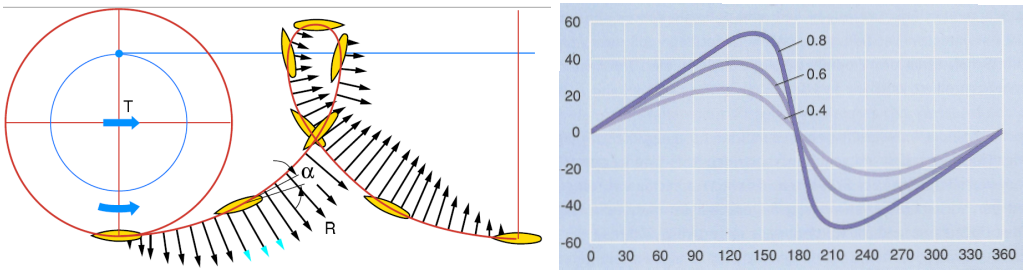



Figure 5.1: Blade angle curve of the VSP [BJ02].

That characteristic qualifies the VSP to be a very suitable propulsion element for a tug boat, which must be highly manoeuvrable to push or pull large container ships in any possible direction. The propulsion system of the Voith Water Tractor is equipped with two counterrotating VSPs, with five blades each, and a nozzle plate situated very

closely below these propellers (cf. Figure 5.2).

Figure 5.2: Voith Water Tractor propulsion system [RV05].

In this way, this plate works as a nozzle plate and also protects the propellers in shallow water. Regarding the complexity of the geometry, the most difficulties considering any simulation approach for the VWT propulsion system are represented by the small gap between the bottom tips of the blades and the nozzle plate, which counts just up to 2.6 cm (see Figure 5.3).

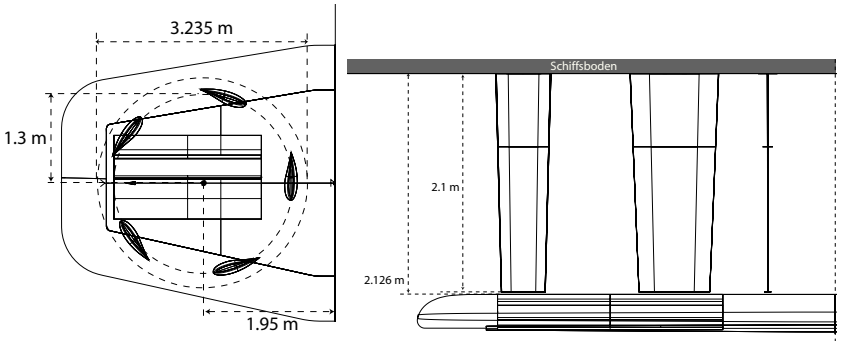



Figure 5.3: Measurements of the VSP propulsion system top and side view[RV05].

When the VSP is operated at a performance factor of 0.8, in order to generate full thrust, a single blade must be turned to a maximal blade angle $\alpha$ up to 55 degrees, according to the blade angle curve given in Figure 5.1.

During one full rotation of the VSP in this operation mode, the lower boundaries of the blades perform a huge relative motion to the nozzle plate boundary. The simulation method that handles the motion between moving boundaries is called a moving boundary scheme or method. Regarding the small gap between the blades and the nozzle plate, any successful fluid flow moving boundary approach must compensate these huge relative motions while still maintaining the conservation of mass and momentum in this zone, independently of the discretization method.

The mesh moving boundary schemes this paper proposes for the fluid flow simulation of the VSP propulsion system can be regarded as rather general and can be applied to

many kinds of propeller constructions. The closest application might be the simulation of standard controllable pitch propellers, which are also often surrounded by ducts or complex geometries close to the rotating blades (see Figure 5.4). The left picture of Figure 5.4 shows a good example of how rotating marine propellers are generally constructed closely to stationary parts of the ship hull. Similarly, this characteristic design feature can be observed at aircraft controllable pitch propellers (see the right picture of Figure 5.4).


Figure 5.4: Controllable pitch propellers of a Hurtigrutenschiff and a Hercules aircraft propeller [Car07].

# 6 Discretization Methods

This chapter will describe how the flow field around a moving VSP propulsion system, i.e., the velocities and the pressure, can be calculated. First, at any point of the region $\Omega_t$ at any time $t$, the region as well as the time have to be discretized, i.e., within the region $\Omega_{t_i}$ at a precise instant of time $t_i$, unique values of velocity $v(x_j, t_i)$ and pressure $p(x_j, t_i)$ have to be assigned to a unique position $x_j$ and its discrete environment. The division of the region $\Omega_{t_i}$ into many smaller subregions $\Omega(x_j, t_j)$ is called discretization; the distance between these subregions is called resolution. The difference between various discretization methods mainly arises from the manner the region is divided into smaller ones and the method how and on which position inside the single regions physical values like the velocities and the pressure, as well as their derivatives, are determined.

One principal aspect of the two most common discretization schemes, the finite volume and the finite element method, is that, because the Equations (2.1) and (2.2) are valid for any size of the volume $\Omega_{t_i}$ or of the corresponding subregions $\Omega(x_j, t_j)$ , the volume can therefore be arbitrary small, even infinitesimal $\Omega_{t_{i(h)}}$. In addition, the superposition of several generated equations considering volumes $\Omega_{t_{i(h)}}$ ensures the conservation of the total momentum and mass for the combined volume $\Omega_{t_i} = \sum \Omega_{t_{i(h)}}$. If the equations of mass and momentum conservation are written over each subvolume $\Omega_{t_{i(h)}}$, and the coupling among them is established, a system of equations can be derived solvable for each unique velocity and pressure value. In the following subchapters 6.1, 6.2 and 6.3, the three most important spatial discretization techniques are presented: the finite difference, the finite volume and the finite element method. The explanation of the finite difference and the finite volume method is a brief summary of what is described in [Oer95].

## 6.1 The Finite Difference Method

The finite-difference method is a spatial discretization method wherein the computational domain is subdivided into numerous intervals of equal size, a cartesian mesh:

$$(\xi_1)_i = \xi_{1,i} = i \cdot \triangle\xi_1 \quad ; \quad (\xi_2)_j = \xi_{2,j} = j \cdot \triangle\xi_2 \quad ; \quad (\xi_3)_k = \xi_{3,k} = k \cdot \triangle\xi_3, \qquad (6.1)$$

with $i, j, k$ being the indices along the coordinate axis and $\Delta\xi_1$, $\Delta\xi_2$, $\Delta\xi_3$ the corresponding step sizes. The physical entities like velocity and pressure are assigned on the mesh points of the cartesian mesh, its indices indicating its position like $u(\xi_1, \xi_2, \xi_3) = u_{i,j,k}$. At each grid point, differential operators are used in order to transfer the differential Equations 2.1 and 2.2 to an algebraic discretized equation system. By that, the number of unknown values of discretized entities is equal to the number of equations. Concerning the finite-difference method there can be used different discretization

schemes which can be distinguished by the differential operators used for the first and second derivatives of the physical entities. For the first derivatives, the three most common differential operators, the forward, backward and central difference operator, are shown in the following:

$$\frac{\partial u}{\partial \xi_1} \approx \frac{u_{i+1} - u_i}{\triangle \xi_1} \quad \text{forward differential operator,} \tag{6.2}$$

$$\frac{\partial u}{\partial \xi_1} \approx \frac{u_i - u_{i-1}}{\triangle \xi_1} \quad \text{backward differential operator,} \tag{6.3}$$

$$\frac{\partial u}{\partial \xi_1} \approx \frac{u_{i+1} - u_{i-1}}{2\triangle \xi_1} \quad \text{central differential operator.} \tag{6.4}$$

The difficulty in applying the finite-difference method is that usually the physical coordinate system embedding the object of simulation is defined in curvilinear coordinates and not in cartesian ones. However, the differential operators are defined in a cartesian coordinate system. So the derivatives of the differential operators formulated in the cartesian system must be converted to fit to the curvelinear coordinate system. Considering real applications with complex curved geometries, the transformation matrices resulting from this conversion are very complex and so the finite-difference scheme is rarely used for complex geometries [Oer95].

## 6.2 The Finite Volume Method

On the contrary, the finite-volume method is the most widely used spatial discretization scheme for the simulation of fluid flows, as it does not need such a complex transformation from the physical to the computational space. Herein, the physical space is directly transferred to the computational domain, which is divided into finite volumes or cells shaped as arbitrary triangles or quadrilaterals (2D) or tetrahedrons or hexahedrons (3D), whose edges (2D) or faces (3D) can easily adapt to the boundaries of the object to be simulated.

To explain how a differential equation is discretized with the finite-volume method, a simpler differential equation than the Navier-Stokes equation is considered, i.e., the differential equation expressing the conservation of momentum for a fluid, but without considering the forces acting on the fluid particles due to friction. For simplicity, the body force is assumed to be set to zero as well. This modified equation for a fluid occupying a volume $\Omega_t$ with its boundary $\Gamma_t$ then reads:

$$\rho \left( \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \frac{1}{\rho} \nabla p \right) = \mathbf{0} \quad \text{on } \Omega_t \quad \forall t \in (0, T), \tag{6.5}$$

or using the notation of the dyadic product:

$$\rho \left( \frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) + \frac{1}{\rho} \nabla p \right) = \mathbf{0} \quad \text{on } \Omega_t \quad \forall t \in (0, T). \tag{6.6}$$

In the first step of the finite-volume method, the differential equation is integrated over the volume (3D) of the whole domain, which gives:

$$\int_{\Omega_t} \rho \left( \frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) + \frac{1}{\rho} \nabla p \right) dV = \mathbf{0} \quad \text{on } \Omega_\mathrm{t} \quad \forall t \in (0, \mathrm{T}). \tag{6.7}$$

Applying the divergence theorem, Equation (6.7) modifies to:

$$\frac{\partial}{\partial t} \int_{\Omega_t} \rho \mathbf{u} \; dV + \int_{\Gamma_\mathrm{t}} \left( (\rho \, (\mathbf{u} \otimes \mathbf{u}) + p) \cdot \mathbf{n}_A \right) dA = \mathbf{0} \quad \text{on } \Omega_\mathrm{t} \quad \text{and} \quad \Gamma_\mathrm{t} \quad \forall t \in (0, \mathrm{T}). \tag{6.8}$$

For a simpler notation, the state vector $F_m$ is introduced, which is called the vector of convective fluxes in the direction of $m$:

$$\mathbf{F}_m = \begin{pmatrix} \rho \; u_m u_1 + \delta_{1m} p \\ \rho \; u_m u_2 + \delta_{2m} p \\ \rho \; u_m u_3 + \delta_{3m} p \end{pmatrix}; \quad \text{where } \delta_{ij} = 1 \text{ for } i = j \text{ and } \delta_{ij} = 0 \text{ for } i \neq j. \tag{6.9}$$

Considering the three-dimensional form of the momentum conservation (Equation 6.8), the value of a physical entity $\mathbf{u}$ inside a volume changes over time according to the fluxes crossing the volume surfaces like the following:

$$\frac{\partial}{\partial t} \int_{\Omega_t} \rho \mathbf{u} \; dV + \sum_{m=1}^{3} \int_{\Gamma_t} (\mathbf{F}_m \cdot \mathbf{n}_A) \, dA = 0 \quad \text{on } \Omega_\mathrm{t} \quad \text{and} \quad \Gamma_\mathrm{t} \quad \forall t \in (0, \mathrm{T}). \tag{6.10}$$

Equation (6.10) holds for any volume and so also for each cell of the computational mesh, and thus Equation (6.10) can be discretized for each cell of the computational domain. In the course of the discretization, the physical entities like velocity and pressure are assigned to the center of a cell, assuming that an entity is constant within a cell. Let us consider the computational mesh consisting of cubes. Then the integral part over the convective terms and the pressure can be transferred to a sum over the faces $A_1, A_2, ..., A_6$ of a cubic volume cell. The normal vector of each face is named $\mathbf{n_1}, \mathbf{n_2}, ..., \mathbf{n_6}$ respectively. For each cell, Equation (6.8) can be discretized to:

$$\frac{d}{dt} \rho \mathbf{u} \; V + \sum_{m=1}^{3} \sum_{l=1}^{6} (\mathbf{F}_{ml} \cdot \mathbf{n}_l \cdot A_l) = \mathbf{0}. \tag{6.11}$$

The flux $F_{ml}$ is the flux in the center of the face $l$ and is calculated via the mean value of the physical entities of the two cells touching the face $l$. If considering the computational mesh with only cubic cells, Equation (6.8) for cell $i, j, k$ reads:

$$\frac{d}{dt} \rho \mathbf{u}_{ijk} \, V_{ijk} + \sum_{m=1}^{3} \sum_{l=1}^{6} (\mathbf{F}_{ml} \cdot \mathbf{n}_l \cdot A_l)_{ijk} = \mathbf{0}, \tag{6.12}$$

where $V_{ijk}$ symbolizes the volume of cell $i, j, k$. After having built the discretized equations for each cell, an equation system is obtained where the single equations of each cell are coupled with those of the neighbouring cells via the fluxes because the incoming flux of one face of a cell is the same as the outgoing flux of the neighbouring cell sharing the same face.

## 6.3 The Finite Element Method

Originally, the finite-element method was mainly used in the field of structural mechanics. In recent years however, its application to fluid flow problems has become much more common, partially due to the possibility of tracking moving boundaries and interfaces in an easy manner. The concept of the finite-element method is very general and can be applied to various partial differential equations. To explain this concept, this chapter gives a rough summary of what is elaborately developed in [Hug87]. What can be considered as the first step is the formulation of the variational statement of the problem; the second step consists of the discretization of the weak formulation via finite element functions. To explain the basic ideas, the focus will be on a very simple differential equation, the Poisson equation. The strong formulation of the Poisson boundary-value problem, (S), can be given by:
Given $f : \bar{\Omega} \to R$ and the constants $g$ and $h$, find $u : \bar{\Omega} \to R$, such that

$$\begin{array}{lcll} u_{xx}(x) + f & = & 0 \quad \text{on} \quad \Omega, & \\ u(1) & = & g, & \\ -u_x(0) & = & h, \quad \text{where} \quad \Omega = ]0,1[ \quad \text{and} \quad \bar{\Omega} = [0,1]. & \end{array} \tag{6.13}$$

The first step of the finite element method is to convert the strong formulation to a weak formulation. This includes multiplying the strong formulation by a weighting function $w$ and integrating the resulting product over the domain $\bar{\Omega}$. Then the full weak form is:
Given $f : \bar{\Omega} \to R$, find $u : \bar{\Omega} \to R$, such that for all $w$ in a certain function space:

$$0 = -\int_0^1 w\,(u_{xx}(x) + f)\,dx. \tag{6.14}$$

The inherent advantage of this first step is that the integral over the product can be simplified by integrating by parts, such that a second derivative of one function, here $u(x)_{xx}$, is converted into two first derivatives of two functions $u_x$ and $w_x$, which will be easier to discretize. This step can be compared to the reduction of the second derivatives, applying the divergence theorem when using the finite volume method (see Equation 6.8). The integration by parts gives:

$$0 = \int_0^1 w_x(x) u_x(x)\,dx - \int_0^1 wf\,dx - [w u_x(x)]_0^1\,. \tag{6.15}$$

It must be mentioned that the function $w$ can be of any type as long as it fulfills the condition that it is zero on the Dirichlet part of the boundary, $w(1) = 0$. So, considering all admissible functions $w$, one usually speaks of a set of weighting functions. In order to prevent the integrals from being infinity, the requirement for the derivatives of the functions $u$ and $w$ is that they and their derivatives are square-integrable, that is:

$$\int_0^1 (u_x(x))^2\,dx < \infty, \tag{6.16}$$

$$\int_0^1 (w_x(x))^2\,dx < \infty. \tag{6.17}$$

Functions whose first derivatives are square-integrable belong to a space called $H^1$. The set of all possible solutions fulfilling 6.16 and the Dirichlet boundary condition in the strong formulation, here $u(1) = g$, is called the collection of trial solutions, $\mathcal{S}$. The set of all admissible weighting functions fulfilling 6.17 and the condition $w(1) = 0$, is called the collection of weighting functions, $\mathcal{V}$. The common mathematical notation including these properties for both the collection of trial solutions, $\mathcal{S}$, and the weighting functions $\mathcal{V}$ is given by:

$$\mathcal{S} = \left\{ u \mid u \in H^1, u\left(1\right) = g \right\} \quad \text{(trial solutions)}, \tag{6.18}$$

$$\mathcal{V} = \left\{ w \mid w \in H^1, w\left(1\right) = 0 \right\} \quad \text{(weighting functions)}. \tag{6.19}$$

Ensuring now that the trial and weighting functions fulfil Equations (6.18) and (6.19) respectively, the weak formulation can be stated as:
Given $f, g$ and $h$, as in Equation (6.13), find $u \in \mathcal{S}$ such that for all $w \in \mathcal{V}$

$$\int_0^1 w_x(x) u_x(x) dx = \int_0^1 wf\, dx - w\left(0\right) h. \tag{6.20}$$

In the literature the following expressions are used to simplify the writing:

$$a\left(w, u\right) = \int_0^1 w_x(x) u_x(x) dx, \tag{6.21}$$

$$\left(w, f\right) = \int_0^1 wf\, dx. \tag{6.22}$$

So the variational formulation of the Poisson problem writes:

$$a\left(w, u\right) = \left(w, f\right) + w\left(0\right) h. \tag{6.23}$$

For the forthcoming argumentation, it is important to state that the expressions $a\left(\cdot, \cdot\right)$ and $\left(\cdot, \cdot\right)$ are symmetric, that is:

$$a\left(u, v\right) = a\left(v, u\right), \tag{6.24}$$

$$\left(u, v\right) = \left(v, u\right), \tag{6.25}$$

and bilinear, that is:

$$a\left(c_1 u + c_2 v, w\right) = c_1 a\left(u, w\right) + c_2 a\left(v, w\right), \tag{6.26}$$

$$\left(c_1 u + c_2 v, w\right) = c_1\left(u, w\right) + c_2\left(v, w\right). \tag{6.27}$$

After the weak formulation is built up, the second step of the finite element method, the discretization of the variational problem via finite element functions, follows. In the 1D case, the space is divided into $n+1$ nodes and $n$ elements, where in 1D two nodes belong to one element and the elements fill the complete space defined by $\bar{\Omega}$. Then, to each node $A = 1, 2, ..., n+1$ the same shape function $N_A$, but a different discrete value $d_A$ according to the local value $u(x)$ of node $A$, is assigned. In the 1D example, one discrete value

is already given by the Dirichlet boundary condition $u(x = 1) = 1$, whereas the other $d_A's$ have to be calculated by solving the linear equation system derived from the finite element discretization. The discretization uses $n + 1$ shape functions and will result in $n$ unknown discrete values $d_A$. The shape functions $N_A(x)$ are designed such that $u(x)$ can be approximated as a sum of each nodal shape function $N_A$ multiplied by its nodal value $d_A$ at the position $x$. The discretized version of $u(x)$ is denoted $u^h(x)$ and can be written as:

$$u^h = \sum_{A=1}^{n} d_A N_A + g N_{n+1}, \tag{6.28}$$

$$N_{n+1}(1) = 1, \tag{6.29}$$

$$g^h = g N_{n+1}, \tag{6.30}$$

$$g^h(1) = g. \tag{6.31}$$

Usually, the shape functions are designed very simple, such that they have a value of one at the position of their node and the value of zero at the next node; in between they are linear in the simplest case. A piecewise linear shape function design of two nodes, $N_1$ and $N_2$, within one element for the 1D example would look like the following:

$$N_1 = 1 - \xi\ , \quad N_2 = \xi, \quad \xi = [0, 1]\,, \tag{6.32}$$

where $\xi$ is the local coordinate of one element, starting with the value 0 at the node number one of the element, and getting the value 1 at the node number two of the element. How the local coordinate system is transferred to the physical one, and that the partition of shape functions must be a partition of unity, can be read in [DH03].

Studying the design of the shape functions, it is evident that applying this partition of shape functions is just a simple linear interpolation algorithm in between and on the nodes. In the same way, the collection of weighting functions $w(x)$ are discretized to $w^h(x)$:

$$w^h = \sum_{B=1}^{n} c_B N_B, \tag{6.33}$$

where the shape functions $N_B$ can be the same as for the trial solutions (cf. the described Galerkin approach [DH03]) but do not have to be the same. After the discretization of the trial and weighting functions, the discretized form of the variational problem 1.4.10 can be expressed like:

$$a\left(\sum_{B=1}^{n} c_B N_B, \sum_{A=1}^{n} d_A N_A\right) = \left(\sum_{B=1}^{n} c_B N_B, f\right) + \left[\sum_{B=1}^{n} c_B N_B(0)\right] h - a\left(\sum_{B=1}^{n} c_B N_B, g N_{n+1}\right). \tag{6.34}$$

Using the bilinearity of $a(\cdot, \cdot)$ the coefficients $c_B$ can be excluded:

$$0 = \sum_{B=1}^{n} c_B G_B, \tag{6.35}$$

where

$$G_B = \sum_{A=1}^{n} [a\,(N_B, N_A)\,d_A - (N_B, f) - N_B\,(0)\,h + a\,(N_B, N_{n+1})\,g]. \tag{6.36}$$

Since $w$ can be arbitrary, just fulfilling $w \in \mathcal{V}$, $w^h(x)$ can be arbitrary. Just fulfilling $w^h \in \mathcal{V}^h$, the $c_B{}'s$ can also be arbitrary, from which it follows that each single $G_B$, $B = 1, 2, ..., n$ must be identically zero. Using standard numerical integration rules, an equation system can then be derived with $n$ equations and $n$ unknowns. A variational formulation for the Navier-Stokes equations will be given in the next section. The finite element discretization for the three-dimensional Navier-Stokes problem is much more complex and is not derived here. An elaborate derivation of the discretization of the Stokes flow problem is shown in [DH03]. Herein, the convective terms are not considered compared to the Navier-Stokes flow problem, but the handling of the finite element discretization in three dimensions becomes apparent.

## 6.4 Space-Time Finite Element Method

So far, only spatial discretization methods were introduced, but in order to complete the discretization of a partial differential equation with respect to time, a method for time discretization has to be chosen. A very widely used scheme to discretize first-order differential equations among the finite difference schemes with respect to time is the $\theta$ family of methods, which is explained in [DH03]. There are a huge variety of time discretization and integration methods that cannot be explained in this thesis. A very unique approach to handle time discretization is given by extending the concept of the spatial finite element method to the time dimension; this is then named the space-time finite element method. According to [DH03], this concept was already included by the time-discontinuous Galerkin method proposed by Jamet (1978) and Johnson et al. (1984). A space-time Galerkin/least-squares finite element formulation with fixed spatial domains was developed by Hughes et al. [HFH89], whereas a space-time finite element approach for a deformable spatial domain was firstly presented by Tezduyar, Behr, Liou and Mittal [TBL91, TBL92, TBML92]. An example of a space-time interpolation function over a space-time slab $]t^n, t^{n+1}[$ between two consecutive time steps, $t^n$ and $t^{n+1}$, could have the following form:

$$u^h(\boldsymbol{x}, t) = \sum_{A=1}^{n_{np}} N_A(\boldsymbol{x}) \left( (1-\theta)\, u_A^n + \theta u_A^{n+1} \right), \tag{6.37}$$

with $\theta = (t - t^n) / (t^{n+1} - t^n)$ being the time interpolation function where $N_A$ can be considered as a standard spatial shape function of node $A$, as in 6.28, and the values $u_A^n$ and $u_A^{n+1}$ are the values of node $A$ at the two consecutive time steps $t^n$ and $t^{n+1}$. Assuming that weighting functions in the time dimension can be created in a similar way as the time interpolation functions, one can create space-time finite elements that have $n_{sd} + 1$ dimensions, such that the multiplication with a weighting function and the consecutive integration can be done not only in the spatial, but also in the time dimension. Regarding a spatial one dimensional problem, two dimensional space-time elements would be necessary as shown in Figure 6.1. Here, already a moving one-dimensional mesh is shown, where $x$ is the spatial domain and the six nodes move along the x-axis when the time $t$ progresses from the time step $t^n$ to $t^{n+1}$. The local element coordinates $\xi$ and $\theta$ represent the reference coordinate system $R_\chi$ of one single space-time finite element.

The conservation equations of mass and momentum 2.1 and 2.2 are written neither over the material $R_X$ nor the spatial $R_x$ but over the reference domain $R_\chi$, which is given by the computational mesh. Doing so, the material time derivatives, which are formulated in the conservation laws, must be related to the time derivatives in the referential coordinate system. How this is done will be explained in Section 7.2.3. The mesh nodes can move arbitrarily, so an accurate following of moving boundaries is also possible. Usually the mesh nodes do not move differently in the time dimension. This is why the nodes in Figure 6.1 are orientated horizontally considering the time dimension. However, it must be mentioned that the mesh nodes could move differently in the time dimension, which is equal to considering different time resolutions in different areas of the mesh.

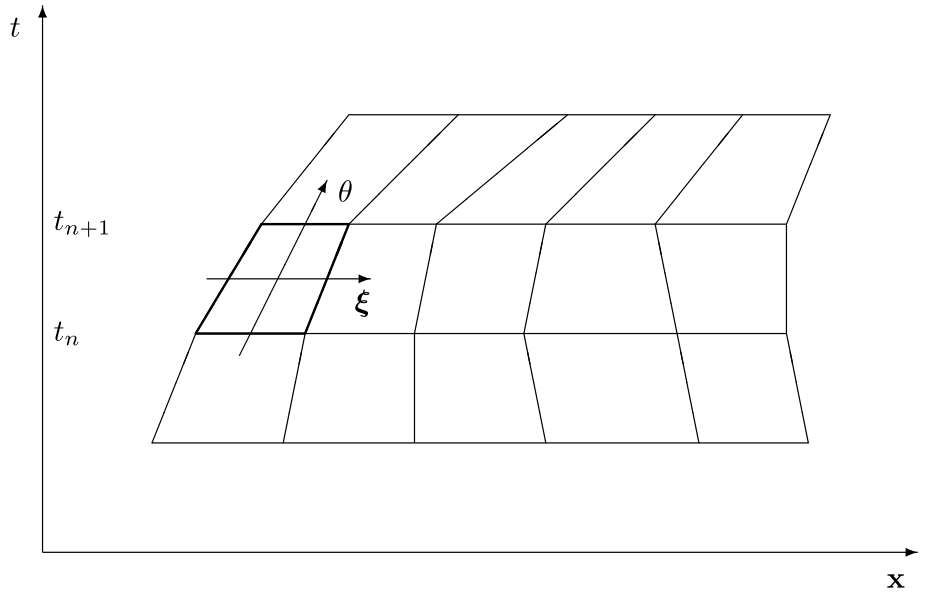



Figure 6.1: Space-time finite elements for a 1D-spatial problem [Beh92].

Considering the space-time finite element concept, the variational formulation of the Navier-Stokes equation as it was developed in [TBL92, TBML92] is presented in the following. First, the time interval $(0,T)$ needs to be divided into subintervals $I_n = (t_n, t_{n+1})$, where $t_n$ and $t_{n+1}$ belong to an ordered series of time levels $0 = t_0 < t_1 < ... < t_N = T$. For each time level, a corresponding domain occupied by the fluid $\Omega_n = \Omega_{t_n}$ and its boundary $\Gamma_n = \Gamma_{t_n}$ is assigned. In between these time levels, a space-time slab $Q_n$ is defined, which is enclosed by the surfaces $\Omega_n, \Omega_{n+1}$ and $P_n$, where $P_n$ is the surface described by the boundary $\Gamma_t$ as $t$ traverses $I_n$. A space-time slab with its surrounding boundaries is shown in Figure 6.2.

In the same way as $\Gamma_t$ is decomposed into a Neuman-type and a Dirichlet-type part, the surface $P_n$ can be divided into $(P_n)_{\boldsymbol{g}}$ and $(P_n)_{\boldsymbol{h}}$. For each space-time slab, the following finite element interpolation function spaces for the velocity and the pressure are defined:

$$\left(\mathcal{S}_{\mathbf{u}}^h\right)_n = \left\{\mathbf{u}^h \mid \mathbf{u}^h \in \left[H^{1h}\left(Q_n\right)\right]^{n_{sd}}, \mathbf{u}^h \doteq \mathbf{g}^h \quad \text{on } \left(P_n\right)_{\mathbf{g}}\right\}, \tag{6.38}$$

$$\left(\mathcal{V}_{\mathbf{u}}^h\right)_n = \left\{\mathbf{u}^h \mid \mathbf{u}^h \in \left[H^{1h}\left(Q_n\right)\right]^{n_{sd}}, \mathbf{u}^h \doteq \mathbf{0} \quad \text{on } \left(P_n\right)_{\mathbf{g}}\right\}, \tag{6.39}$$

$$\left(\mathcal{S}_p^h\right)_n = \left(\mathcal{V}_p^h\right)_n = \left\{p^h \mid p^h \in H^{1h}\left(Q_n\right)\right\}. \tag{6.40}$$

Similar to Equation (6.28) and Equation (6.37), a first order polynomial in space and time is used. Considering consecutive space-time slabs and its division into finite elements, the interpolation functions are continuous in space but discontinuous in time. Discontinuous approximations in time allow problems to be solved independently for each time slab instead of solving a global problem over the whole time domain [DH03]. In

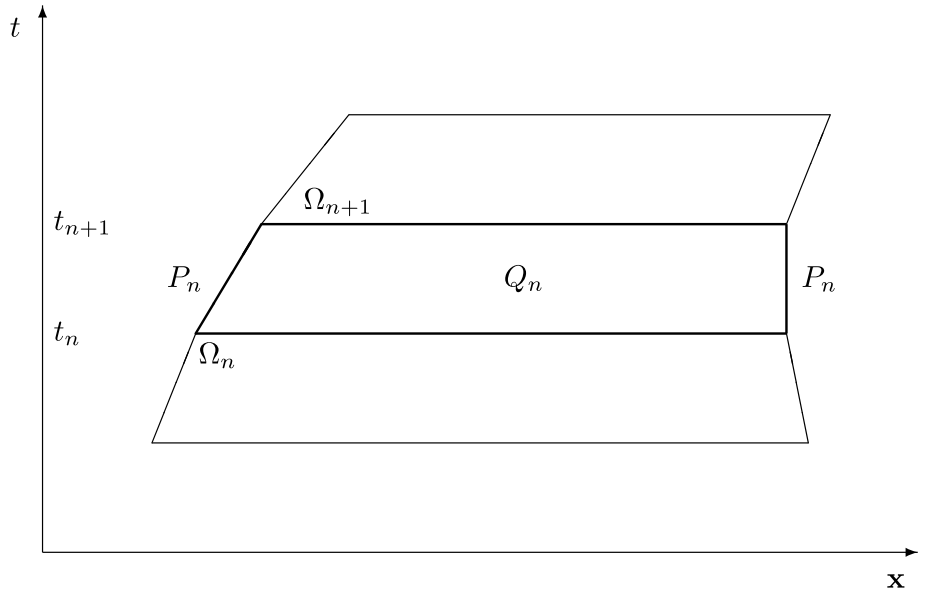



Figure 6.2: Space-time slab and its boundaries [Beh92].

doing so, the space-time finite element mesh can cope with just two different time levels. The stabilized space-time formulation for deforming domains can be written as follows: given $\left(\mathbf{u}^h\right)_n^-$, find $\mathbf{u}^h \in \left(\mathcal{S}_{\mathbf{u}}^h\right)_n$ and $p^h \in \left(\mathcal{S}_p^h\right)_n$ such that $\forall \mathbf{w}^h \in \left(\mathcal{V}_{\mathbf{u}}^h\right)_n$, $\forall q^h \in \left(\mathcal{V}_p^h\right)_n$:

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left( \frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \nabla \mathbf{u}^h - \mathbf{f} \right) dQ + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(p^h, \mathbf{u}^h) dQ \\
&\quad + \int_{Q_n} q^h \nabla \cdot \mathbf{u}^h dQ + \int_{\Omega_n} \left(\mathbf{w}^h\right)_n^+ \cdot \rho \left( \left(\mathbf{u}^h\right)_n^+ - \left(\mathbf{u}^h\right)_n^- \right) d\Omega \\
&\quad + \sum_{e=1}^{(n_{el})_n} \int_{Q_n^e} \tau_{mom} \frac{1}{\rho} \left[ \rho \left( \frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \nabla \mathbf{w}^h \right) - \nabla \cdot \boldsymbol{\sigma}(q^h, \mathbf{w}^h) \right] \\
&\qquad \cdot \left[ \rho \left( \frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \nabla \mathbf{u}^h - \mathbf{f} \right) - \nabla \cdot \boldsymbol{\sigma}(p^h, \mathbf{u}^h) \right] dQ \\
&\quad + \sum_{e=1}^{(n_{el})_n} \int_{Q_n^e} \tau_{cont} \nabla \cdot \mathbf{w}^h \rho \nabla \cdot \mathbf{u}^h dQ = \int_{(P_n)_h} \mathbf{w}^h \cdot \boldsymbol{h}^h dP.
\end{aligned} \tag{6.41}$$

The notational conventions used in 6.41 are given below:

$$\left(\mathbf{u}^h\right)_n^{\pm} = \lim_{\varepsilon \to 0} \mathbf{u}\left(t_n \pm \varepsilon\right), \tag{6.42}$$

$$\int_{Q_n} ... dQ = \int_{t_n}^{t_{n+1}} \int_{\Omega_t^h} ... d\Omega dt, \tag{6.43}$$

$$\int_{P_n} ... dP = \int_{t_n}^{t_{n+1}} \int_{\Gamma_t^h} ... d\Gamma dt. \tag{6.44}$$

As Equation (6.41) can be solved for each time slab independently, the overall solution can be calculated consecutively for all space time slabs $Q_1, Q_2, ..., Q_{N-1}$. The computations start with

$$\left(\mathbf{u}^h\right)_0^+ = \mathbf{u}_0. \tag{6.45}$$

Regarding the variational formulation 6.41, the first three terms of the left hand side, together with the right hand side, correspond to the standard Galerkin formulation of the problem. The fourth term weakly enforces the continuity of the velocity in time over the lower boundary $\Omega_n$ of the space-time slab $Q_n$. The fifth term is built by the least-squares form of the momentum equation and stabilizes as long as the residual of the momentum equation is different from zero. If the fifth term is not zero, an additional viscosity, also dependent on the stabilization parameter $\tau_{mom}$, is added to the water viscosity, which stabilizes the advection-dominated cases. If the residual is getting close to zero, this term vanishes and the consistency of the momentum equation is preserved. In addition, the sixth term stabilizes at high Reynolds numbers. The explanation of the stabilization schemes and its background can be read in [Beh92]. The time around the publishing date of [Beh92] can be regarded as the advent of the use of XNS, an executable Navier-Stokes solver, which discretizes all the terms of 6.41 with respect to time and space similar to the theory given in this and the previous chapter. The discretized terms are then linearized and the resulting linear equation system is solved iteratively. The solution concept of the linearized equations used in XNS, namely the GMRES algotithm, is also presented in [Beh92]. At the CATS institute at the RWTH Aachen, XNS has been extended for many different applications like blood flow simulations, maritime applications and optimization.

# 7 Moving Boundary Methods

Before deciding on a discretization method and a suitable moving boundary scheme for the fluid flow simulation of the Voith Water Tractor (VWT), a brief overview will be given about moving boundary schemes in general, along with a short glimpse at rotor and propeller fluid simulation approaches. Beforehand, the author would like to point out a quite common alternative to a moving boundary approach regarding propeller simulation. If the propeller is only surrounded by the computational domain and its boundaries, a rotating coordinate system can be applied to simulate the flow around the blades of a propeller. In [Dym08], this approach is used to calculate the pressure distribution on the suction and the pressure side of one propeller blade as well as the probability of cavition on the blade. Herein, only one blade is simulated and periodic boundary conditions on the circumference of the finite volume mesh are used in order to save computing resources. This simplification does not allow to simulate the mutual influence of the single blades and a wake simulation is also not possible. Since the blades of the VWT propulsion system move individually and the nozzle plate represents another rigid object inside the computational domain, an approach with a rotating coordinate system is not suitable [JPSU07].

## 7.1 Fixed Underlying Mesh

Moving boundary methods can be categorized into two classes according to [Gue06]. The first class comprises the methods using a fixed underlying grid, e.g., the Cartesian grid [MAB03], the immersed-boundary [Pes02, LL03] and the ficticious boundary [GPHDJ01] method. The disadvantage of these methods is that either the interface of the moving boundary is smeared in some way, as the exact boundary can only be approximated to a certain extent; or the mesh around the moving boundary is redesigned, which comes along with high computational effort. To give an example, the Cartesian-grid method according to [MAB03] detects where the moving boundary cuts the cells of the fixed grid and creates irregularly shaped cells, which are then also used for the discretization of the equations system. The detection of the cells to be recreated, as well as the remeshing in the boundary zones, can be considered as quite complex, and still the surrounding mesh of the moving boundary can hardly adjust as well to the moving boundary as the initial one created by a conventional mesh generator.

## 7.2 Moving Mesh

To achieve more accurate moving boundary simulations, boundary fitted conforming meshes that move exactly with the boundary must be used [Gue06]. Considering these moving mesh methods as a second class of moving boundary methods, the very common ones, e.g., the clicking mesh, the sliding mesh, the Chimera mesh and the general Arbitrary Lagrangian-Eulerian (ALE) approach, will be explained below.

### 7.2.1 Sliding and Clicking Mesh Method

The sliding and the clicking mesh methods are suitable when it comes to regular rotational motions of the boundaries. The principal idea of both methods is to divide the computational mesh into one part, which is fixed, and another moving part, which is attached to the moving boundary. In between, a proper interface has to be defined. Considering a rotating geometry, a proper interface consists of two concentric circles having the same number of nodes, with all nodes being situated in the same angular distance to each other (see Figure 7.1). The concept of the clicking mesh is to chose the time step size, such that according to the rotational speed the moving inner circle just rotates that its nodes reach exactly their successive nodes on the fixed outer circle within one time step (see Figure 7.1).

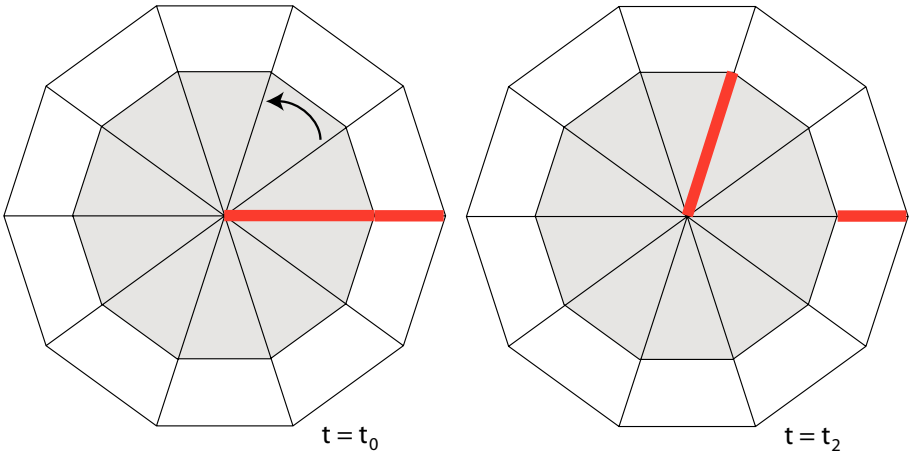



Figure 7.1: Clicking mesh concept showing two different positions of the inner rotating mesh.

This approach is often found in the context of finite volume schemes. Here, only the values inside the rotating grid need to be updated when continuing the simulation from one time step to the next one. The easy concept of a sliding mesh interface is followed up in [BWS98], where the simulation of the fluid flow is performed in a stirrer consisting of a stator (fixed mesh) and a rotor (rotating mesh). The inherent disadvantage is that the temporal refinement is directly linked with the spatial one. So regarding a fixed time step size, the mesh resolution on the interface needs to be adapted to the rotational speed, or

vice versa. Consequently, simulations with varying rotation speeds cannot be performed with the same mesh [Sie02, Grö00] nor can the ramp-up conditions be represented as an initially very low rotation speed. This would correspond to a time step size almost equal to infinity. In [BWS98] it is reported that for low Reynolds number simulations of the stirrer, very large time steps would have to be used, or the grid resolution in the azimuthal direction would become very high. The sliding mesh concept avoids this interdependence between the spatial and the time refinement. Here, the nodes of the moving and the fixed circle do not necessarily have to touch each other but can be situated independently (see Figure 7.2); even the number of nodes does not necessarily have to match [SB08], depending on the sliding mesh approach.

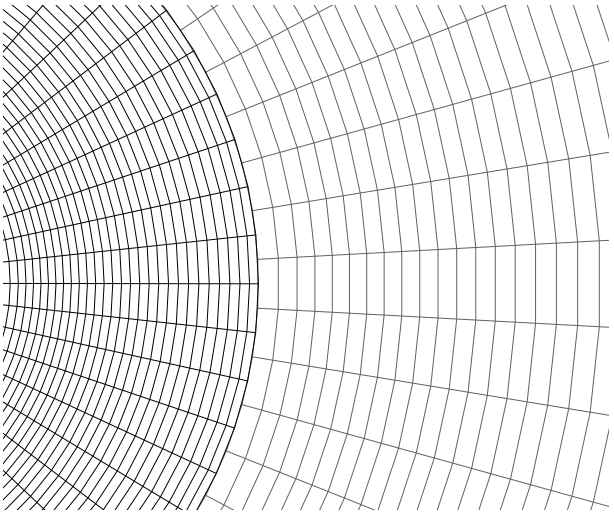


Figure 7.2: Section of a sliding mesh [SB08].

Of course, an interpolation algorithm has to be used to establish the communication at the interface between the fixed and the moving mesh. In [SB08], a sliding mesh approach implemented in a finite volume CFD code is applied to simulate the aerodynamics of a helicopter rotor-fuselage. It is warned in [SB08] that an interpolation algorithm at the interface can introduce numerical artefacts and can have effects on the overall solution quality.

### 7.2.2 Chimera Mesh Method

The most popular moving boundary technique in the field of finite volume and finite difference schemes is the Chimera mesh method. Originally, it derives from the idea to construct the most suitable meshes around single objects situated within one computational domain, leading to various component grids, which together fill the whole computational domain but overlap at many regions (see the red, green and blue meshes in Figure 7.3).

At the overlapping regions, interpolation algorithms are used to connect the infor-

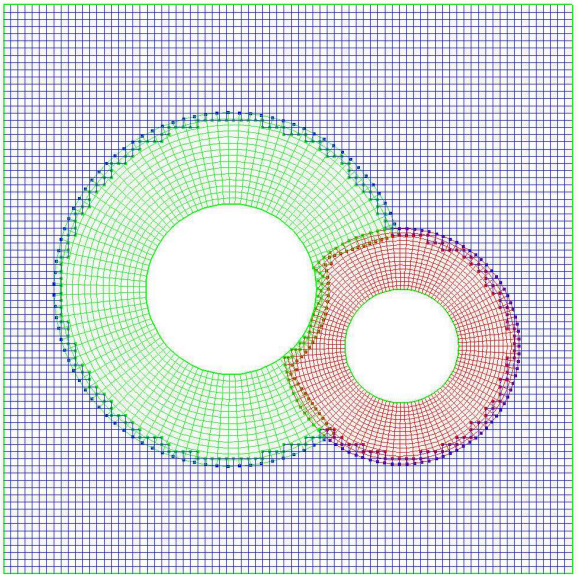


Figure 7.3: Chimera mesh [Gue06].

mation from one component mesh to another. Once a proper interpolation algorithm is found, it is also possible to move the component grids relatively to each other. Using that technique, not only simple rotations can be simulated but also rather complex motions like eccentric rotations or rotations super-positioned with translation.

Basically, the traditional overset composite grids can be established in three steps according to [Gue06]: geometry and component grids generation, an algorithm for determining how to cut holes in the component grids that overlap, and an algorithm for the interpolation of the field data between the meshes.

According to [Gue06], the main difficulty considering the Chimera approach is the data transfer between overlapping grids and how fluxes are treated at overlapping boundaries. Whereas simple interpolation methods just transfer the flow variables from a donor cell to the receptor cell; more accurately, a trilinear interpolation scheme can be used, in which several vertex points of the donor cell form interpolation stencil points for the receptor point. Interpolation algorithms that maintain conservation seem to be more sophisticated [Gue06]. Similarly to the sliding mesh concept, the Chimera approach is not free from possible numerical artefacts due to the interpolation at the interface between the component grids, which can finally influence the overall solution.

### 7.2.3 Arbitrary-Lagrangian Method

Aside from the clicking mesh approach, the methods described so far show drawbacks concerning the calculation of the fluid flow variables either at the moving boundaries due to an insufficiently exact representation of the boundary or at the interface between a moving and a fixed mesh where the fluxes have to be calculated accurately leading to

difficulties regarding mass or momentum conservation. These drawbacks are present due to unavoidable interpolation of non-matching nodes or vertices. In the finite element and finite difference context, the Arbitrary Lagrangian-Eulerian (ALE) formulation gives the opportunity to move the nodes of the computational mesh arbitrarily and therefore an interpolation can be avoided. The mesh can move in a Lagrangian way at the moving boundaries, whereas its motion can be arbitrary or even be halted elsewhere in the computational domain. In that way, the moving boundary can be represented most accurately and the presence of a sharp interface between a moving and a fixed grid is avoided. The ALE is realized by writing the equations of mass and momentum conservation 2.1 and 2.2 neither over the material domain $R_X$ nor the spatial one $R_x$, but over the reference domain $R_\chi$ that is given by the computational mesh. In addition, the computational mesh can move arbitrarily and its deformations are commonly defined by solving an elasticity equation considering the computational mesh as an elastic solid exposed to the deformations at the boundary given by the moving boundaries (see Section 7.2.5). If the computational mesh is moving, the material time derivatives that are used in the conservation laws must be related to the time derivatives in the referential coordinate system that belongs to the mesh. This relation is called the fundamental ALE equation in [DH03] and can be written as follows:

$$\frac{\partial f}{\partial t}\,|_X = \frac{\partial f}{\partial t}\,|_\chi + \frac{\partial f}{\partial \boldsymbol{x}} \cdot \boldsymbol{c} = \frac{\partial f}{\partial t}\,|_\chi + \boldsymbol{c} \cdot \nabla f, \tag{7.1}$$

where $c$ is the relative velocity between the material and the mesh. To interpret Equation (7.1), one can say that the variation of a physical quantity $f$ for a given particle $X$ is calculated by the sum of the local variation with respect to the reference $\chi$ and a convective contribution that includes the relative motion between the material and the reference system multiplied by the spatial gradients of $f$. With respect to the discretization, a method must be developed to relate the fluid values calculated at one node at the beginning of a time step to the values to be calculated at the same node at the end of the time step, including the displacement of the node within the time step. This can be done with a finite difference scheme, but is more accurately done by using the space-time concept explained in Section 6.4. This discretization algorithm in space and time accounts for the moving mesh within the discretization of one time step at two different time levels, saying that there is not only one spatial mesh where the partial differential equations are discretized on but two; one capturing the locations of the mesh nodes at the beginning of the time step, and another at the end of the time step. In between these two meshes representing two subsequent time levels, the space time is spanned like visualized in Figure 6.1 or 6.2 for a one-dimensional mesh in space. Within this space time, the linear time interpolation function is used according to Equation (6.37) to interpolate the flow field values between the two different time steps. In the space-time approach, the conservation equations of mass and momentum 2.1 and 2.2 are, similarly to the ALE approach, written neither over the material $R_X$ nor the spatial $R_x$ but over the reference domain $R_\chi$, which is given by the computational mesh. When transforming the flow variables from the material domain to the reference element domain, a Jacobian matrix is built where the relation shown in Equation (7.1) is included due to the character of

the space-time approach. Therefore, this part of the space-time method can be regarded as analogous to the ALE approach [MH97]. To apply the space-time method successfully in the context of moving boundaries, appropriate mesh moving schemes for the moving finite elements have to be chosen.

### 7.2.4 Shear-Slip Mesh Update Method

One mesh moving scheme that has already been applied successfully with the space-time method is the Shear-Slip Mesh Update Method (SSMUM). The SSMUM can be regarded as the finite element counterpart to the sliding mesh method in the field of finite volumes. Similarly, this method is ideal for the modelling of geometries that undergo large but regular deformations, such as the rotational motion of all the propeller blades of one Voith-Schneider propeller. The concept of the SSMUM is to create a special layer of elements, which can undergo 'shear' deformation, so that the relative motion between the moving and the static elements in the mesh can be compensated. In order to restrict the element deformation to a minimum, a frequent remeshing via the regeneration of the element connectivity is realized. The basic steps of the SSMUM concept are visualized in Figure 7.4. The upper nodes of the SSMUM layer are moving in both cases, the two dimensional (top row) and the three dimensional (bottom row) case, whereas the nodes of the lower layer are static.

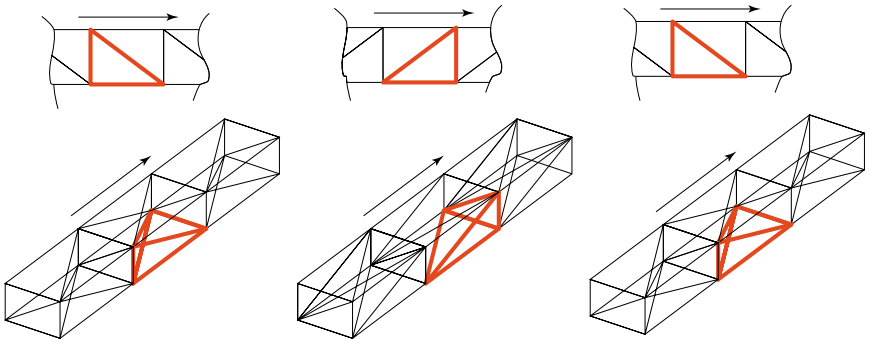


Figure 7.4: SSMUM concept [BT01].

From the left to the right the mesh deforms where the second column represents the moment of remeshing as the node positions of the new (deformed) and the original (undeformed) mesh overlap so that projection is not necessary. The reconnectivity changes such that the old upper node no longer belongs to the red triangle or tetrahedron respectively, but its ancestor node on the moving layer. Since only a small part of the overall connectivity is being regenerated, the computational cost is greatly reduced compared to approaches where a frequent remeshing of the whole mesh is necessary. In the context of the space-time finite element method applied to a 3D application, one has to imagine a four dimensional space-time element which consists of one tetrahedron at the bottom of a time slab and another one at the top of a time slab; then the space in

the fourth dimension between the two tetrahedrons is filled by interpolating the node velocity and pressure values between the lower and the upper time slab by a time interpolation function, for example that given in Equation (6.37). The SSMUM method together with the space-time finite element concept has been successfully employed in [BT99, BA03, BT01] where the applications range from the flow simulation around a rotating cylinder, squares, a stirrer over a centrifugal blood pump to a rotating propeller and a helicopter.

### 7.2.5 Elastic Mesh Update Method

When it comes to more general and not only rotating moving boundary applications, the Elastic Mesh Update Method, together with the ALE and the space-time finite element approach, offers excellent opportunities, as the mesh can absorb even irregular deformations. The basic idea is to regard the mesh as a fictitious elastic material occupying at an instant $t \in (0,T)$ a bounded region $\Omega_t \subset R^{n_{sd}}$, with boundary $\Gamma_t$. In that context, the displacements $\mathbf{v}(\mathbf{x},t)$ are governed by the linear elasticity equilibrium equation:

$$\nabla \cdot \sigma_{\#}(\mathbf{v}) = 0 \quad \text{on } \Omega_t \quad \forall t \in (0,T), \tag{7.2}$$

with the constitutive equation defined as:

$$\sigma_{\#}(\mathbf{v}) = \lambda(\operatorname{tr} \varepsilon_{\#}(\mathbf{v}))\mathbf{I} + 2\mu\varepsilon_{\#}(\mathbf{v}), \tag{7.3}$$

$$\varepsilon_{\#}(\mathbf{v}) = \frac{1}{2}(\nabla\mathbf{v} + (\nabla\mathbf{v})^T), \tag{7.4}$$

with $\lambda$ and $\mu$ being the Lamé constants. Let the displacements also be subject to the boundary conditions:

$$\mathbf{v} \cdot \mathbf{e}_d = g_{\#,d} \quad \text{on } (\Gamma_t)_{g_{\#,d}}, \quad d = 1, ..., n_{sd}, \tag{7.5}$$

$$\mathbf{n} \cdot \sigma_{\#} \cdot \mathbf{e}_d = h_{\#,d} \quad \text{on } (\Gamma_t)_{h_{\#,d}}, \quad d = 1, ..., n_{sd}, \tag{7.6}$$

where $(\Gamma_t)_{g_{\#,d}}$ and $(\Gamma_t)_{h_{\#,d}}$ are complementary subsets of $\Gamma_t$. A displacement field obtained with these equations leads to the new mesh coordinates $\mathbf{x}(t_{n+t})$ calculated through:

$$\mathbf{x}(t_{n+t}) = \mathbf{x}(t_n) + \mathbf{v} \quad \text{on } \Omega_t. \tag{7.7}$$

The elasticity equilibrium equation can be solved by the finite element method. The variational formulation of Equation (7.2) reads as follows: find $\mathbf{v}^h \in \left(\mathcal{S}^h_{\mathbf{v}}\right)_t$ such that $\forall \delta\mathbf{v}^h \in \left(\mathcal{V}^h_{\mathbf{v}}\right)_t$:

$$\int_{\Omega_t} \varepsilon_{\#}\left(\delta\mathbf{v}^h\right) : \sigma_{\#}\left(\mathbf{v}^h\right) d\Omega = \int_{(\Gamma_t)_{h_{\#}}} \delta\mathbf{v}^h \cdot \mathbf{h}^h_{\#} d\Gamma. \tag{7.8}$$

In most applications of the elastic mesh approach, the motion of the nodes on the whole boundary $\Gamma_t$ is explicitly known, so all the boundary nodes get a Dirichlet-type boundary condition. As a consequence, $(\Gamma_t)_{h_{\#,d}}$ is empty and so all the boundary integral above is

zero. Equation (7.3) itself assumes an isotropic and homogeneous linear material. This type of material behaviour, however, "introduces a high element distortion during the mesh updating process. Typically, elements close to the changing surfaces are constrained to modify their shape much more than those elements located far from these surfaces. This behaviour frequently leads to extremely distorted meshes near the boundaries and, in the limit, to not conforming meshes and intersection elements." [CBO00]. Therefore, it is proposed in [CBO00] to vary the Young's modulus for each element according to geometric criterion, such that elements closer to the moving boundary get stiffer than those further away. Beside that approach, it is suggested to stiffen smaller elements with regard to a previous analysis. Three kinds of analysis are proposed. One preparatory analysis determines the local strain of the elements assuming a homogeneous Young's modulus. According to these calculated strains, the Young's modulus of the elements are changed such that elements with higher strains obtain a higher Young's modulus. Thus, a smooth mesh deformation can be achieved in the final calculation, where the final local strain of the elements is calculated assuming an inhomogeneous Young's modulus. A second preparatory analysis relates the element strain energy density, and a third one the distortion energy density, to Young's modulus. The ball-vertex method used by [BDS05] applies a spring stiffness between nodes situated on one edge, which is inversely proportional to the edge length, so that short edges are stiffer than longer ones, which they report to be beneficial in the control of the local element deformation.

Tezduyar, Behr and co-workers modified the variational formulation of Equation (7.2) by dropping the Jacobian of the transformation of the integrals in Equation (7.8) from the physical domain $\mathbf{x}$ to the element domain $\xi$. The Jacobian of an element $e$ is $J^e = det\,(\partial \mathbf{x}/\partial \xi)^e$, with $\mathbf{x}$ and $\xi$ representing the physical and local element coordinates respectively. Normally, as a standard finite element method step, the integrals in Equation (7.8) are not calculated as a whole in the physical domain $\mathbf{x}$ but as a sum of integrals calculated within the single elements using the element domain $\xi$. Herein, the Jacobian has to be used in the following way:

$$\int_{\Omega_t} [...]\, d\Omega = \sum_e \int_\xi [...]^e\, J^e d\xi. \tag{7.9}$$

Excluding the element Jacobian has the effect that small elements get stiffer than the bigger ones. That approach was first used in [TBMJ92]. A further extension to this concept is the introduction of a scaling parameter that gives the ability to switch from the concept of excluding the Jacobian or not, or choosing any intermediate forms (see [ST02]).

In order to make smaller elements stiffer than larger ones, Hughes and Masud [MH97] multiply the right side of Equation (7.9) with the following parameter:

$$\tau^e = \frac{1 - \frac{A_{min}}{A_{max}}}{\frac{A_e}{A_{max}}}, \tag{7.10}$$

where $A_e, A_{max}$ and $A_{min}$ represent the areas of the current, the largest, and the smallest elements in the mesh. This approach is extended by Masud [MBK07] in considering compressibility effects. In a one-dimensional application this can be best described

as forcing the nodal displacements of one element to be less than the element length. This limit to the compressibility enters Equation (7.8) with a multiplier calculated with respect to the element area like Equation (7.10).

In the implementation of XNS, the concept of a scaling parameter is used to transition between including or excluding the element Jacobian in combination with a parameter, which scales the influence of element stiffening with respect to the element area similar to Equation (7.10). A very common benchmark among authors in the field of elastic mesh deformation algorithms is the pitching of an airfoil and the responsive deformation of the surrounding unstructured mesh. Whereas the numerical experiments of Chiandussi show zones of strongly squeezed elements and tangling at an airfoil pitch of about thirty degrees, the stiffening concept chosen by Masud shows almost perfectly shaped elements at a thirty degree pitch. The method chosen by Tezduyar seems to result in similar element quality compared to the concept of Masud, though it cannot be compared perfectly, as instead of an airfoil turned to thirty degrees a rigid beam is pitched up to forty-five degrees. The ball-vertex method only uses a very coarse mesh around the airfoil and the airfoil is just pitched up to about fifteen degrees, which makes a comparison to the other methods difficult. The XNS elastic mesh deforming approach has been successfully applied to various free-surface flow simulations [TBML92, BA02], where the moving waterline prescribes the Dirichlet boundary condition for the elastic mesh. In [BA02] the flow through a trapezoidal channel and in a trapezoidal tank is simulated with the EMUM approach; in [GBT99] the 2D flow past a spillway of a dam. As a final remark, it is important to mention that all the extensions regarding the stiffening of smaller elements introduce a nonlinear character to Equation (7.2), which results in a irreversible mesh deformation that is especially harmful when dealing with the simulation of various consecutive propeller rotations where a blade has to be pitched back and forth periodically. Regarding the application of the EMUM concept in 3D together with the space-time finite element method, one has also to think of the use of four dimensional space-time elements described at the end of Section 7.2.4.

### 7.2.6 Conclusion

Given an adequate mesh moving scheme, a finite element approach using the space-time method should be more accurate than fixed-mesh or semi-discrete approaches when simulating a fluid flow involving moving boundaries. Firstly, the moving boundary can be followed more precisely compared to fixed-mesh approaches. Secondly, the use of deformable finite elements together with a space-time approach allows to create a broader region separating the fully moving elements from the static elements; so the motion of the moving boundary can be smoothed out within the mesh. A Chimera or a sliding-mesh approach does not have such a smoothing region, as the moving elements directly touch the static elements. This increases the risk of numerical artefacts due to the interpolation at the interface (cf. the discussions in Sections 7.2.1 and 7.2.2). Last, the projection of the old solution to the mesh of a new time step becomes unnecessary when using a space-time method together with a deformable finite element approach. With regard to the time discretization, the space-time finite element concept can be regarded as more accurate than conventional time discretization schemes. Overall, the space-time finite element implementation of the variational formulation presented in Equation (6.41) is therefore considered as promising for the simulation of the fluid flow around the VSP propulsion system. The fluid flow solver, called XNS, has also shown to be highly parallelizable, which is an indispensable feature when simulating large fluid flow real world 3D applications. XNS has been used not only for the simulation of biofluid [TAB$^+$96], for the flow in ventricular assist devices [BBNP08] and therefore for the design of medical devices [BNP07], but also for fluid flow simulation past a spillway [GBT99]. The SSMUM and the EMUM approaches have both been successfully applied in these applications in combination with the space-time finite element approach (cf. Sections 7.2.4 and 7.2.5). However, the SSMUM has not yet been applied simultaneously with the EMUM. How the knowledge of these methods can be transferred to the simulation of the VWT propulsion system is shown in the forthcoming chapters. At first, the rotational motion of each propeller shall be captured by the SSMUM approach. The motion of the single blades with respect to the whole propeller rotation, using the EMUM and a customized approach, is treated thereafter.

# 8 Shear-Slip Mesh Update Method Applied to the Voith-Schneider Propeller Propulsion System

As mentioned in Section 7.2.4, the SSMUM method is well suited to capture rotating moving boundaries such as the two counterrotating VSP propellers of the VWT. Regarding the motion of the VWT system, the SSMUM layer should be shaped like a beaker around each VSP to cover all rotating parts, e.g., all the five blades of one VSP. In Figure 8.1, the SSMUM layer around the far VSP is shown while the other one is not displayed. Each of the two SSMUM layers has a thickness of one characteristic element length, as shown in the second row of Figure 7.4.

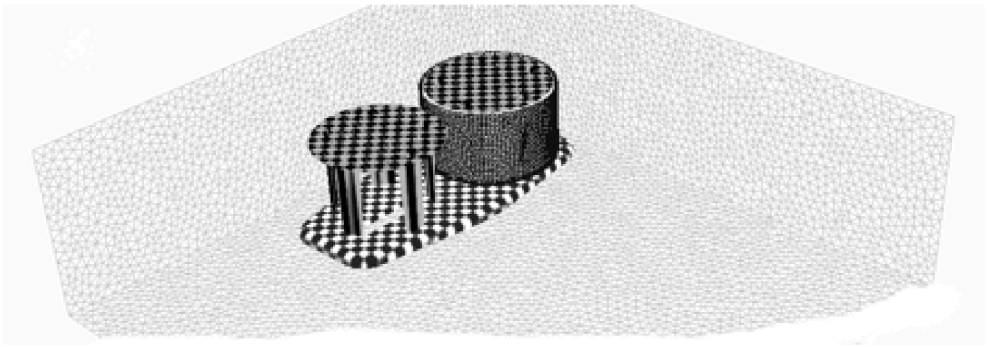


Figure 8.1: VWT propulsion system and one of the SSMUM layers.

In order to create such SSMUM layers, structured domains covering the rotating five blades of each VSP are constructed. In Figure 8.2, the beaker-shaped domain and the rotating plate with five blades are partly visualized. As the static counterparts, structured domains with the same topology but sized up by the width of the SSMUM layer are built attached to the static volume grid of the computational domain (see Figure 8.3). Given these two pairs of surface beaker domains, the SSMUM layer elements can be created between the space of the inner (moving) and the outer bigger (static) beakers. The SSMUM elements are designed prism by prism, where one prism is designed between two opposite triangles, one situated on the inner rotating (green) and another on the outer static (blue) beaker (see Figure 8.4). The space of the prism in between the green triangle and the red triangle is indicated by the red lines. After the prisms are constructed, every prism is divided into three tetrahedrons. It is important to mention that these tetrahedrons cannot be created arbitrarily inside one prism; the direction of

the moving beaker and the corresponding motion of each single tetrahedron has to be considered to avoid bad-quality SSMUM elements or even tangling in the curved zone at the bottom of the SSMUM layer during the propeller rotation.

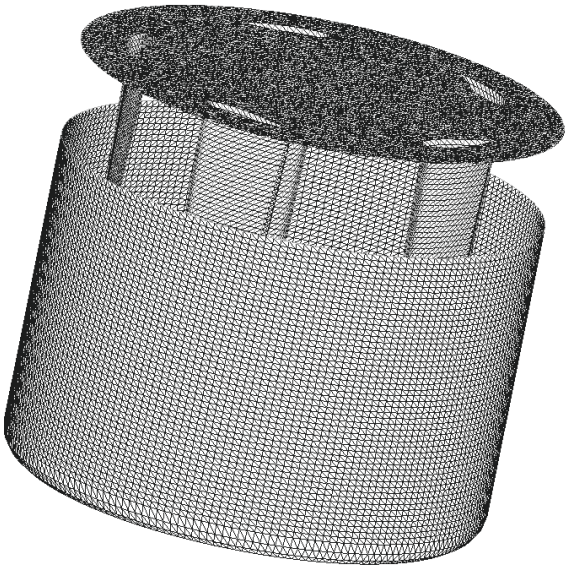


Figure 8.2: Beaker-shaped, structured domain around one VSP.

Considering the relative motion of the SSMUM beakers, the tetrahedrons should be constructed in such a way that the nodes within one tetrahedron are not separated by more than two edge lengths from each other. This must hold at any time of the rotation and for any tetrahedron inside any prism. A proper creation of tetrahedrons is shown in the Figures 8.5.

The corresponding correct relative motion of the inner moving beaker (green) to the outer static beaker (blue) is indicated by the red arrow. If the inner beaker moves to the right, none of the nodes of the moving triangle would be separated more than two edge lengths from any node of the static triangle (as can be seen in Figure 8.6).

However, if the inner moving beaker (green) moved to the left, the node at the right bottom vertex of the blue tetrahedron would be separated more than two edge lengths from the node attached to the moving beaker (green), resulting in a bad element quality of the blue tetrahedron.

In our mesh generation approach, the SSMUM layers are built after the two moving volume meshes of the two VSPs and the static volume mesh are generated. In that way, the programme building the SSMUM layers can then also connect the three volume grids shown in Figure 8.7, the two rotating propeller grids and the static grid, into one entire mesh ready for simulation.

During the whole process of the mesh generation for the VSP propulsion system, many

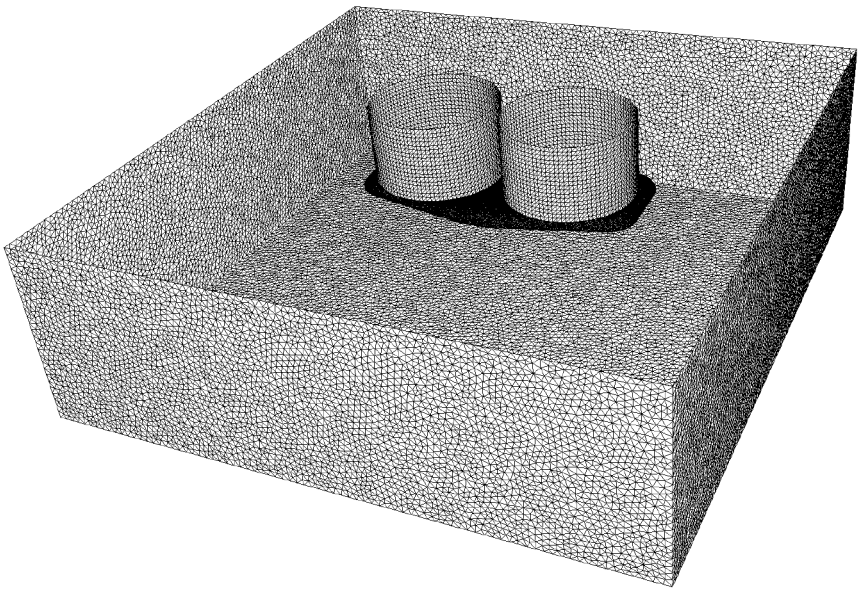


Figure 8.3: Beaker-shaped structured domains attached to the static mesh.

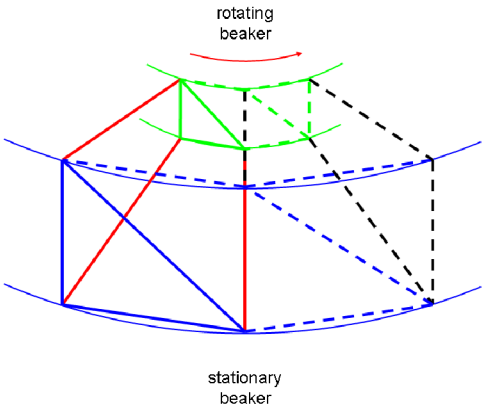
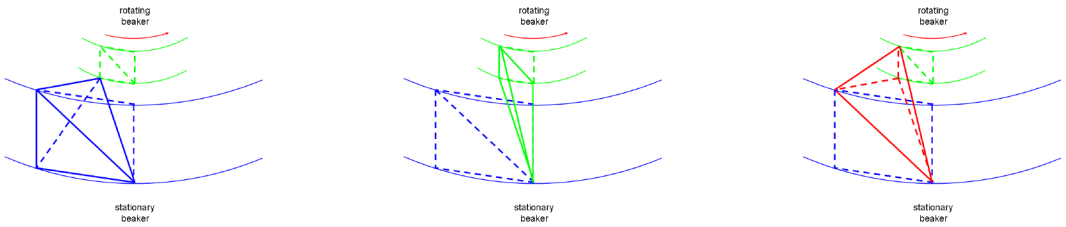



Figure 8.4: Connected opposite triangles on the inner (green) and the outer (blue) layer building one prism.

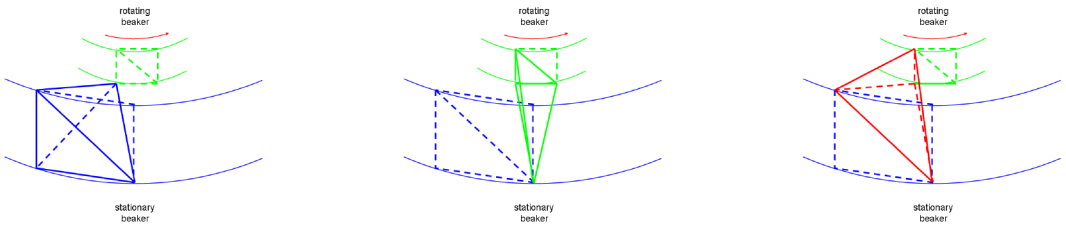
intermediate steps have to be taken and several data formats are used. These steps are visualized in Figure 8.8, with the data format also given after each intermediate step.

Firstly, the CAD data file of the VSP propulsion system is imported into *Gridgen* via the IGES format. IGES, abbrevation for *Initial Graphics Exchange Specification*, is a manufacturer-independent format for a standardized exchange among CAD programmes.




blue tetrahedron green tetrahedron red tetrahedron

Figure 8.5: Division of a prism into three tetrahedrons.




blue tetrahedron green tetrahedron red tetrahedron

Figure 8.6: Deforming tetrahedrons during an inner beaker rotation.

It is saved in a text-based (ASCII) or in a binary format. After the import, the surface meshes are created on the geometric surfaces of the blades and the protection plate, as well as on the boundaries of the computational domain. These triangular surface meshes are exported via a text-based UCD format (.grd). This format is converted by the Fortran programme *grd2mode* into the Surface95 Mesh Generator format (.mod.e), which can be read by *MeshView* and *makelayer*. The Fortran programme *makelayer* uses this format to create the beaker-shaped surface meshes around the geometric surface meshes of each VSP including the five blades and the rotating plate, where the blades are attached. As a static counterpart, *makelayer* also constructs two beakers attached to the stationary part of the mesh. The result is the three separated closed-surface meshes that can be seen in Figure 8.7. The format of these three meshes is then changed by the converter *me2stl* to the STL-format. By that, these meshes can be imported into *Gridgen*, where the volume meshes inside the three closed surface meshes are generated; thereafter, the reference node group (RNG) numbers of each surface entity is prescribed in *Gridgen*. The three volume meshes are then exported via the fieldview unstructured (FV-UNS) format. In order to join these three meshes easily, their format is changed into the MIXD format by the converter *fv2mixd*. Finally, the Fortran programme *comblayer* generates the two SSMUM layers between the three meshes as explained earlier, and exports a single mesh in the MIXD format, which can be read by XNS. Further details about each format and converter programme can be read in [Wal05].

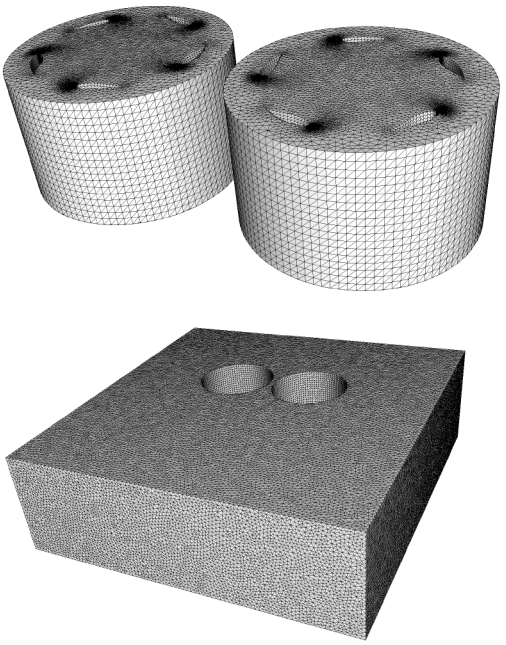


Figure 8.7: Volume meshes of the two VSP's (top) and the static volume mesh (bottom).

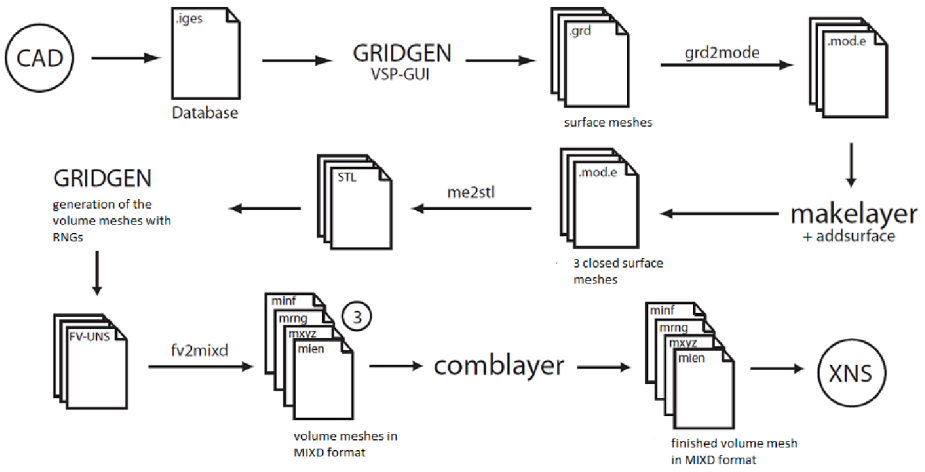



Figure 8.8: Toolchain of the mesh generation.

# 9 Elastic Mesh Update Method Applied to the Voith-Schneider Propeller Propulsion System

So far, the rotation of each single propeller can be accommodated by the SSMUM approach. One could also assume that the rotation of a single blade can be handled in the same way. However, due to the design and location of the supports of the nozzle plate (cf. Figures 5.2 and 5.3), there is not enough space to construct concentric beakers around a single blade; moreover, in a later simulation of the fluid flow around the whole propulsion system, the supports have to be integrated. Therefore, the SSMUM concept cannot be applied in this case. As described in Section 7.2.5, the Elastic Mesh Update Method can be used to prescribe an elastic motion of the mesh nodes given certain moving boundaries. For the application to the VSP propulsion system, the whole interior mesh of the inner rotating SSMUM beaker will be considered as a fictitious elastic material. The Dirichlet boundary conditions $g_{\#,d}$ are given by the location of the mesh nodes on the single blades inside each SSMUM beaker. The node displacements $\mathbf{v}$ are therefore a superposition of the rotation of one propeller as a whole and the rotation of one single blade around its axis according to the blade angle curve given in Figure 5.1. The two pictures of a very coarse mesh in Figure 9.1 show the combined application of the SSMUM and the EMUM concept for the two VSP's of the VWT propulsion system. For the left VSP, only the top domain (yellow) of the deforming mesh is shown; for the right VSP, the top domain (red) of the deforming mesh and the SSMUM beaker (yellow) is visualized. The bottom of the ship is placed on the said top domains and a slip condition is used for the nodes of these domains, except for the nodes attached to the moving blades, to each of which a Dirichlet boundary condition is assigned.

The left picture of Figure 9.1 shows the two counterrotating VSPs at a first position, the right picture at a second subsequent position. The right VSP rotates anticlockwise from the first to the second position, the left VSP clockwise. A red bar indicates an extension of a virtual line connecting the centre of rotation of the right VSP with the axis of the blade, which is located to the lower right most corner of the left picture of Figure 9.1. The distance between the black and the red bar in the right picture indicates the segment of the circle covered by that blade, while the right VSP is moving from the first to the second position. During that motion, the deflection of that blade changes from about fifty degrees to zero degree. The other blades also change their deflection during that propeller motion.

In the plane of the top domains, the mesh deformation looks well distributed and the five blades motion of each VSP is fully compensated. So far, the protection plate is not

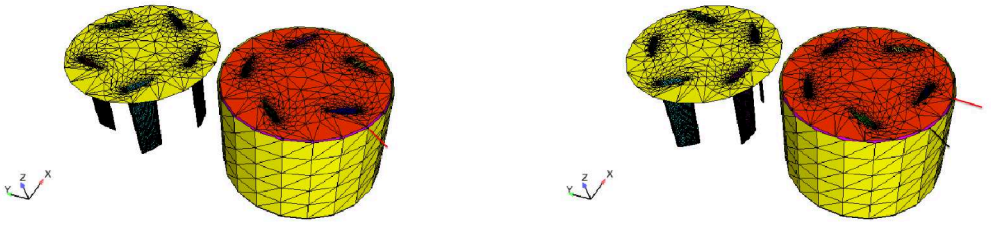
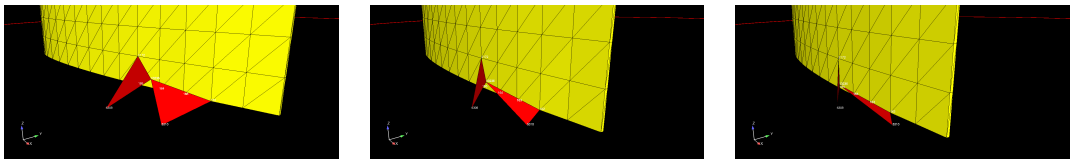


Figure 9.1: Elastic mesh deformation of two surface meshes at two different positions of two counterrotating VSPs.

integrated into the mesh, as direct application of the EMUM approach generates mesh tangling after a one degree turning of the blade in the small horizontal gap between the bottom tips of the blades and the inner SSMUM beaker. As the SSMUM beaker is fixed relative to the axis of the blade, the horizontal length of the blade measures about half a meter, and the gap between the bottom of the blade and the protection plate measures only 2.6 cm (cf. Figure 5.3), especially the elements close to the tips at the bottom of the blade need to perform strong deformations, which can lead to tangling of these elements. Regarding the tangling, the region between the bottom of the blade and the inner SSMUM beaker is called the critical zone in the following. In Figure 9.2, the tangling of an element in that critical zone is visualized. From time step one to time step three the blade turns in such a way that the trailing edge of the blade moves to the bottom part of the picture. By that, the top node of the right red tetrahedron pushes the neighbouring node of the left red element to the left. By that motion, this node of the left element approaches the plane spanned by the three other nodes of the left tetrahedron. Due to the movement, this node touches this plane at time step three and this results in a flat tetrahedron, which is explained as tangling in the following. A tangling element inside the computational mesh deteriorates the mesh integrity and aborts the simulation. As integration over a zero or even a negative volume of an element is not possible, or more precisely, the Jacobian of a tangling element becomes negative, the contribution to the whole integration process then becomes negative. Tangling predominantly occurs in zones of high mesh deformation and when the initial elements are not optimally shaped. This is especially the case in the critical zone between the blade bottom and the SSMUM beaker, as mentioned above. The limited horizontal space in that critical zone not only induces a problematic mesh deformation, but furthermore does not allow to create optimally shaped tetrahedrons in that critical zone. Regular tetrahedrons in that zone would be so small that the meshes of all the critical zones below the ten blades of the VWT would contribute with a number of approximately twenty million elements to the number of elements of the whole mesh. Due to limited computing and memory resources, such a

mesh size cannot be used for a fluid flow simulation.

In the hope to avoid tangling, all possible parameter combinations influencing the element stiffening are tested, both by varying the impact of the element Jacobians as well as the element areas as described in Chapter 7.2.5. Unfortunately, these modifications to the mesh deformation calculation do not show any improvement. As the implemented mesh moving scheme is very general, it cannot take advantage of the fact that the nodes on the bottom of the blade only move on a horizontal plane. This strictly horizontal motion is a simplification to an arbitrarily moving boundary. One could of course restrict the node motion within the mesh to be only horizontal, but this does not prevent tangling in general. Instead, one can use the information about the moving boundary to change the mesh design such that tangling cannot occur. So, it was decided to construct the upcoming mesh approaches such that the tetrahedrons in that critical zone below the blades have all their four nodes situated only on two different horizontal levels. In this way, a node cannot pierce through the plane that is spanned by the other three nodes of a tetrahedron assuming the motion of the nodes of that tedrahedron is prescribed to be only horizontal.



time step=1 time step=2 time step=3

Figure 9.2: Tangling of an element in the critical zone at the blade bottom.

In [JMBF01], the elastic mesh motion is also analyzed and special restrictions for the nodes on moving parts are set to prevent tangling. For example, mesh nodes on the surface of a valve stem of an engine piston lead most probably to tangling when they were prescribed to be attached to the stem. However, allowing them to slip on the stem surface prevented tangling.

Our first approach uses horizontal elliptic planes parallel to the moving direction where the nodes of one tetrahedron are situated only on two different elliptic planes, and their motion is prescribed explicitly in the form of Dirichlet boundary conditions. For geometric reasons, circular planes cannot be chosen, as the diameter of the inner SSMUM beaker is restricted in a way that it fits inside the space given by the supports of the nozzle plate, as explained before. That is why a horizontal circular plane with its centre at the point of rotation of a blade, capturing just the trailing edge of the blade, would cut into the biggest allowed SSMUM beaker. The motion of the nodes on each single elliptic plane are prescribed in such a way that the angular velocity of the nodes goes down from top to bottom. By that, tangling is prevented between the blades and the inner SSMUM beaker, and the blade can be turned up to fifty degrees. Then, however, tangling occurs at the borders of the elliptic planes, as the node motion outside the planes is not prescribed and the elastic mesh has to compensate again strong relative deformations, this time in the region outside the ellipsoids (see Figure 9.3).

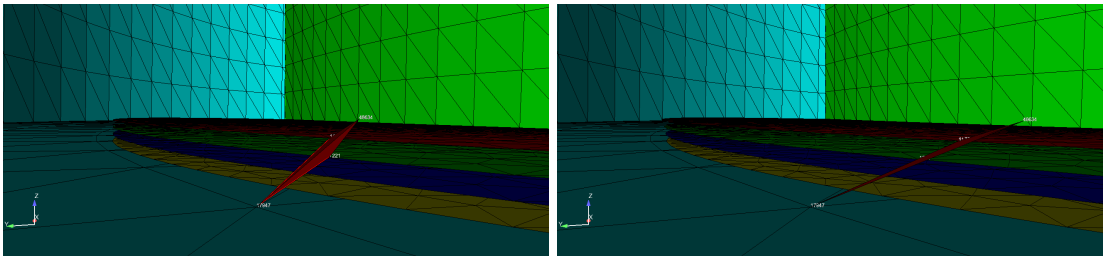
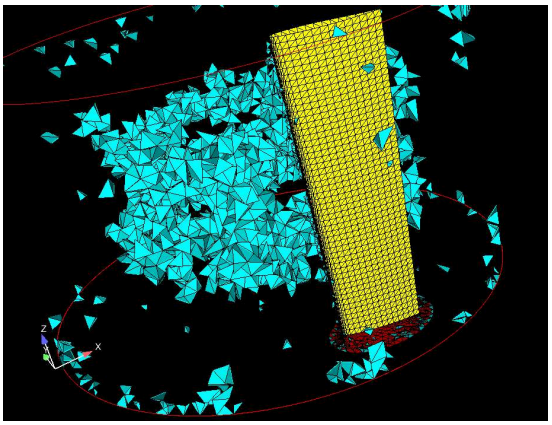


Figure 9.3: Tangling at the borders of the ellipsoids.

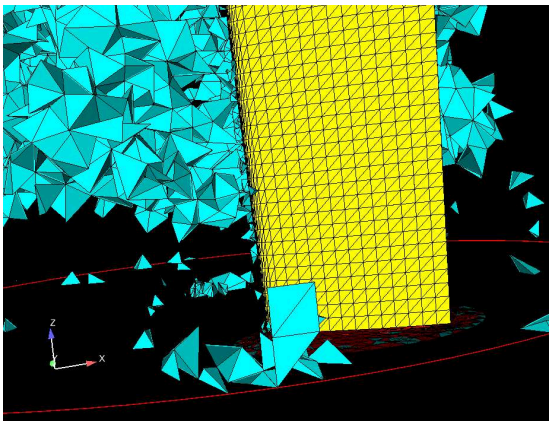
One option to solve this is prescribing the angular velocity of the nodes within one plane in such a way that the innermost nodes exactly follow the turning of the blade, and the nodes further away from the blade have a reduced angular velocity. As the planes are elliptic, the angular velocity of a node within a plane would have to be related to its distance from the axis of the blade and to its angular position on the elliptic plane. This approach is complicated to program, and the basic idea of the EMUM approach is to gain the necessary displacements of the nodes by solving the elasticity equilibrium equation and not by the formulation of complex mesh-moving functions for each single node of the mesh. Another option is trying to influence the distribution of shear in the critical region with additional domains, which can affect the resulting deformation. These additional domains can build a shearing body within the elastic mesh, which, for example, leads to a smoothening of a high shearing rate close to the blade and to a lower shearing rate close to the SSMUM beaker. Such a shearing body (red) is shown in Figure 9.4. This body is built by the small bottom domain of the blade, a wider domain closely situated at the SSMUM beaker, and third domain connecting the small domain with the wider one. The motion of all nodes within this body are prescribed to be horizontal during an elastic mesh motion. The blue elements in Figure 9.4 are elements far away from the critical zone of mesh deformation at the blade bottom. It is evident that these elements do not undergo strong deformations during the elastic mesh motion. These blue elements all preserve the shape of almost regular tetrahedrons.

Through this approach, a turning angle of a single blade of 72 degrees can be achieved, which is a deflection higher than the required maximal blade angle of the important blade angle curves to model for the fluid simulation. On the other hand, this concept shows tangling when turning the blade several times back and forth.

For that reason, big horizontal discs are constructed covering the whole space between all the five blades of one propeller and the inner SSMUM beaker as shown in Figure 9.5. The motion of their nodes are also restricted to be only horizontal. A mesh moving strategy based on these parallel horizontal discs, called EMUM discs in the following, is also free from tangling up to 72 degrees, but in addition, it is possible to turn the blade various times back and forth, which is a necessity when simulating several full rotations of the two VSP's. In Figure 9.5, the mesh deformation of the discs can be seen from the top and on a vertical cut.

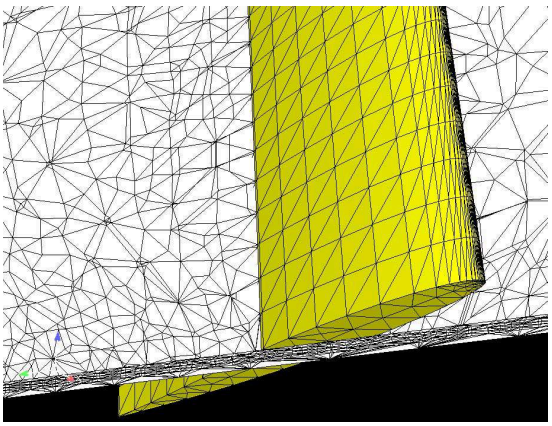


20 degree turn

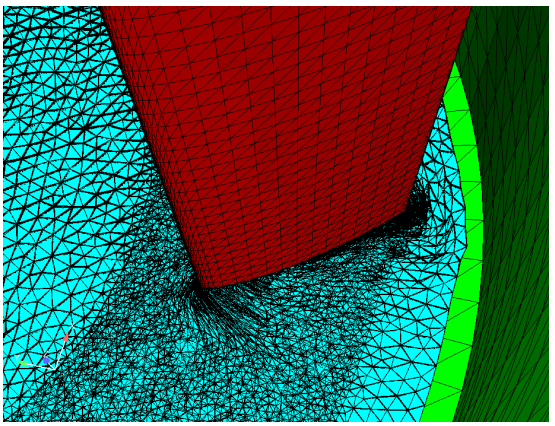


40 degree turn

Figure 9.4: Twisting ellipsoid as a shearing body (red) with increasing diameter.



Lateral View



Top View

Figure 9.5: EMUM and horizontal discs.

As described at the end of Chapter 7.2.5, it can be observed that the mesh deformation of the EMUM approach is not fully reversible. In Figure 9.6, a different position of the mesh nodes, especially around the trailing edge, can be observed when the initial mesh (left picture) is compared to the mesh after having the blade turned one time back and forth (right picture). An almost reversible mesh deformation algorithm is important if various rotations of a propeller have to be simulated. An irreversible elastic mesh deformation has the inherent danger to tangle at an unpredictable moment. This could mean that the simulation cannot be continued and must be restarted interpolating the velocity and pressure values of the tangled mesh to a new mesh. As this diminishes the advantages given through the space-time finite element approach, the inherent danger of tangling does not favour the elastic mesh approach. The easy application of the EMUM approach on the contrary does favour it.

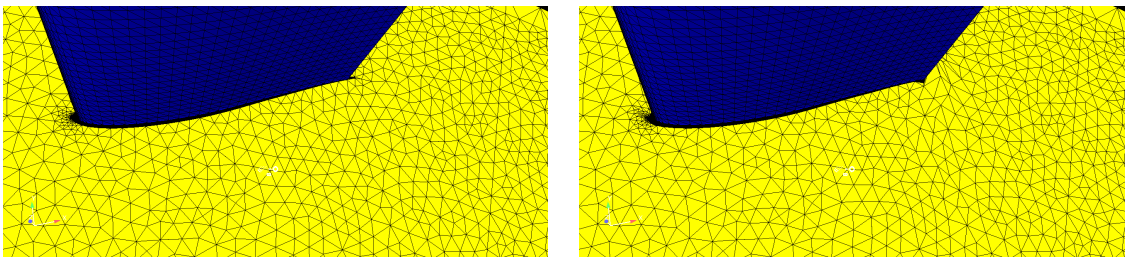


Figure 9.6: Irreversible mesh motion of the EMUM concept.

# 10 Concentric Shell Motion Method

Due to the uncertainties of the EMUM approach regarding the danger of tangling when turning the blade to the maximal blade angle, considering a mesh deformation with five blades inside the rotating mesh of one propeller as well as the issue of non-reversibility, a second approach is followed by developing the Concentric Shell Motion Method (CSMM). The aim of this concept is to prescribe the motion of every single node explicitly. To achieve this, each single blade is embedded into various increasing concentric shells, which rotate relatively to each other. These shells cover the blade completely from the bottom region between the blades and the inner SSMUM beaker up to the ship hull. As all the shells have the same two dimensional grid topology consisting of triangles, the opposite nodes of two neighbouring shells are used to construct tetrahedrons in the volume between these shells, similarly to the construction of the SSMUM elements between the inner and the outer SSMUM beaker. The grid motion is realized in such a way that the mesh between the blade and the innermost shell is moving with the same angular velocity as the blade, and the outer shells move with a decreasing angular velocity; the outermost shell remains fixed. Considering the shape of the shells, the most intuitive form is the elliptical one. A cylindrical shell shape is not possible, as the space around a single blade is restricted by the diameter of the inner SSMUM beaker due to the supports of the nozzle plate. Finding the appropriate shell shape is equal to reducing the squeezing of the elements between the shells to the minimum, so that the blade can be turned sufficiently without the tangling of the shell elements. By means of a custom visualization tool, several different elliptical shapes are tested; with the best one an angle of 48 degree can be achieved. The Figures 10.1 and 10.2 show triangular elements that represent the horizontal faces of the tetrahedrons between two neighbouring shells. It has to be noticed that the initial constant direction of triangulation in the two-dimensional mesh favours the anticlockwise turning (see Figure 10.1), while the clockwise turning provokes very flat elements and almost tangling (cf. Figure 10.2).

In the next step, the shell shape is adapted so that the zone of mesh deformation has a similar outline to the area traced by the blade whilst turning.

This butterfly shape, see Figure 10.3, only enables a 42 degree turning and is therefore discarded, but it helps develop ideas for the design of the final balloon shape shown in Figure 10.4. In the first place, a decreasing number of nodes from the inner shell to the next outer one results in a better element quality in the round zones of the mesh. Secondly, the closer distance of the leading edge of the blade to the centre of blade rotation compared to the trailing edge allows the frontal rounded parts of the shells to have a bigger central angle, which is the angle enclosed by two neighbouring nodes of one shell with the centre of the blade rotation. By that, the deformation in that zone is almost as optimal as it would be with circular shells. The shells in the zone of the

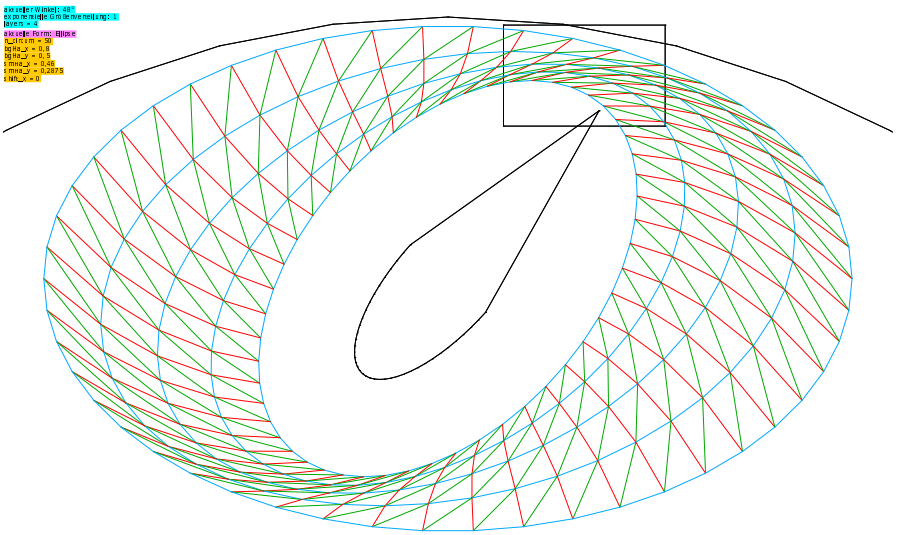


Figure 10.1: Asymmetric triangulation and anti-clockwise turning.

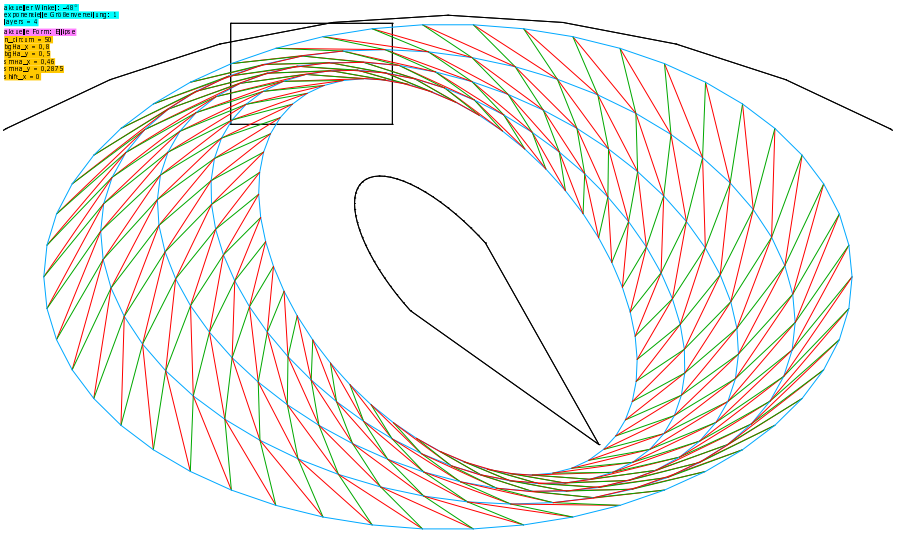


Figure 10.2: Asymmetric triangulation and clockwise turning.

trailing edge are designed similarly to the butterfly shape in that area because of the greater distance to the centre of rotation, e.g., bigger radii and smaller central angles are used. The left and the right circle sections are joined by a section of horizontal lines. In addition, the mesh topology of the final ballon shape is symmetric to the x-axis to ensure the same mesh quality when turning either clockwise or anticlockwise.

This kind of shell shape can be turned up to 63 degrees, shown in Figure 10.5, in both

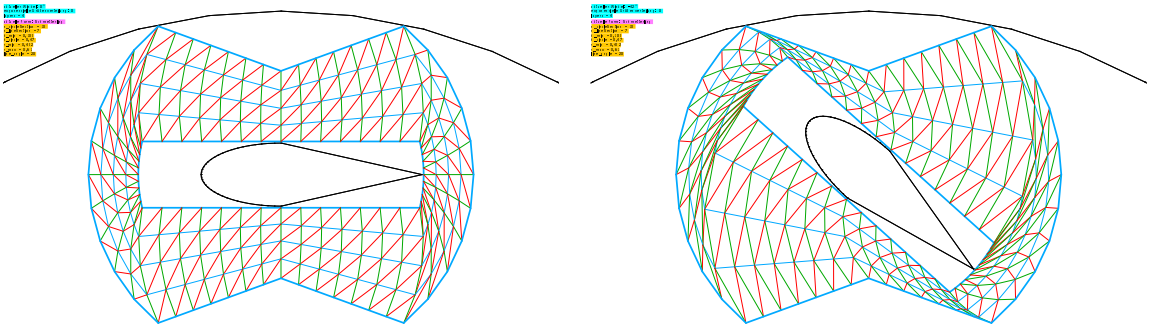


Figure 10.3: Butterfly shape.

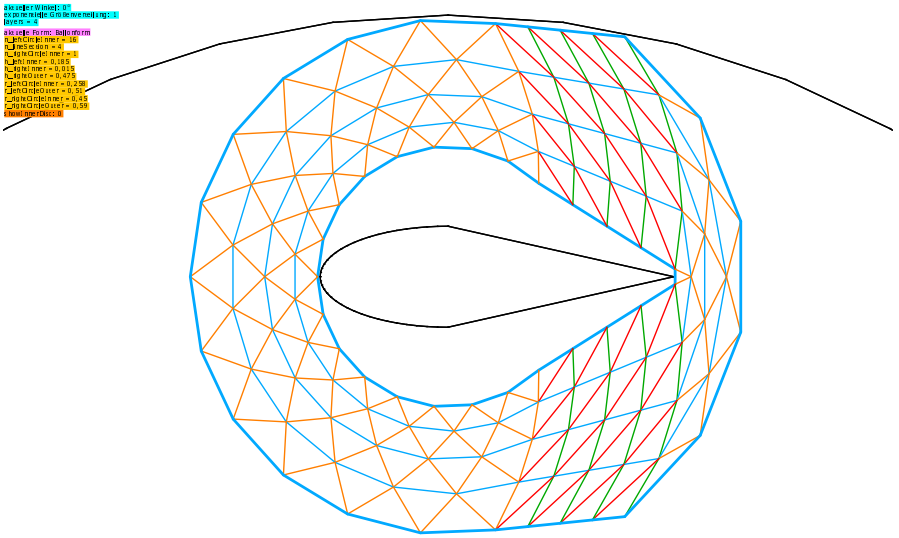


Figure 10.4: Final balloon shape.

directions and is regarded as the final shape for the Concentric Shell Method. Allowing a flexible mesh generation for upcoming fluid flow simulations, several parameters of this final shape are programmed to be input mesh parameters. These are the number of edges of the left inner circle section, the horizontal line section and the right inner circle section, which can be found in Figure 10.7 as *nleftCircleInner, nlineSection, nrightCircleInner* respectively. Considering the figure plane comprising a vertical and a horizontal dimension, further parameters are *hleftInner*, which can be identified as the vertical distance of the rightmost nodes of the left circle section of the innermost shell to the centre of blade rotation, and *hrightInner* and *hrightOuter*, which correspond each to the vertical distances of the leftmost nodes of the right circle section of the innermost shell and the outermost shell respectively. The parameters *rleftCircleInner, rleftCircleOuter, rrightCircleInner* and *rrightCircleOuter* represent the inner and outer radii of the left and right

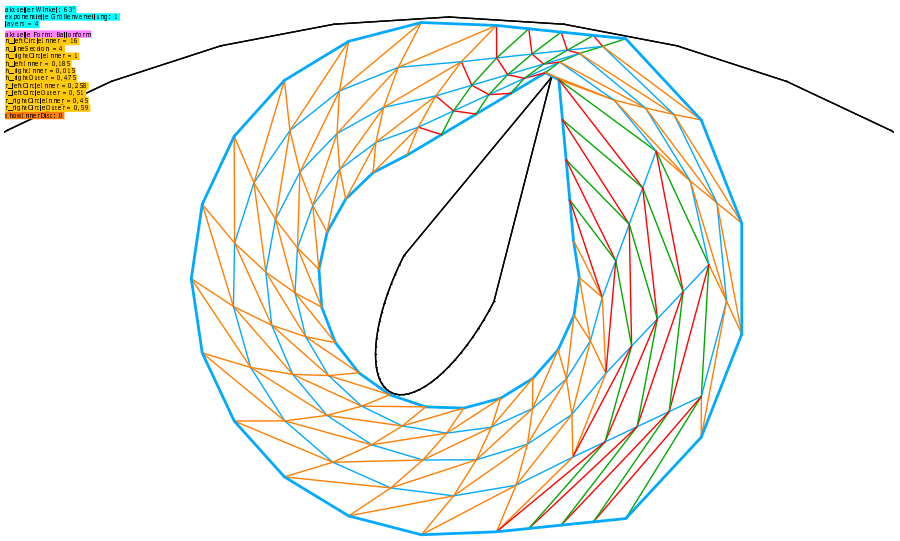


Figure 10.5: 63 degrees turn with the balloon shape.

circle sections respectively. The left circle sections show a decreasing number of nodes from the inner to the outer shells, so that two neighbouring nodes on one shell always enclose the same central angle with the axis of the blade. On the right circle sections, an increasing number of nodes on the shells suggests a better element quality. To complete the mesh generation around the blade, a topology for the region between the bottom of the blades and the inner SSMUM beaker must be defined. The above described shape parameters are kept. In order to achieve a good element quality in the innermost zone near the centre of rotation, the number of nodes is reduced by half from one ring of nodes to the next inner one (see Figure 10.6).

In the following, the volume mesh generation between the blade surface and the inner SSMUM beaker is briefly described. At first, the shells around a blade are created, all with an identical topology. The topology of the outermost shell is shown in Figure 10.7. The nodes (red and black numbers) and the triangles (blue numbers) are enumerated in an ascending order, starting from a node located in the uppermost plane and going further in an anticlockwise direction. After the nodes are enumerated in the upper plane, the nodes that are situated in the next lower plane are enumerated. As a result, there exist several rings of enumerated nodes for each shell, each ring located on a single horizontal plane. The triangles of the outer and the inner shell are enumerated as well. Furthermore, to each triangle three nodes are assigned, e.g., to the triangle number one the nodes one, nineteen and eighteen are assigned. Given these triangles on the outer and inner shell, these domains can be used to generate a mesh of tetrahedrons between the outer shell and the inner SSMUM beaker and the inner shell and the blade surface using Gridgen.

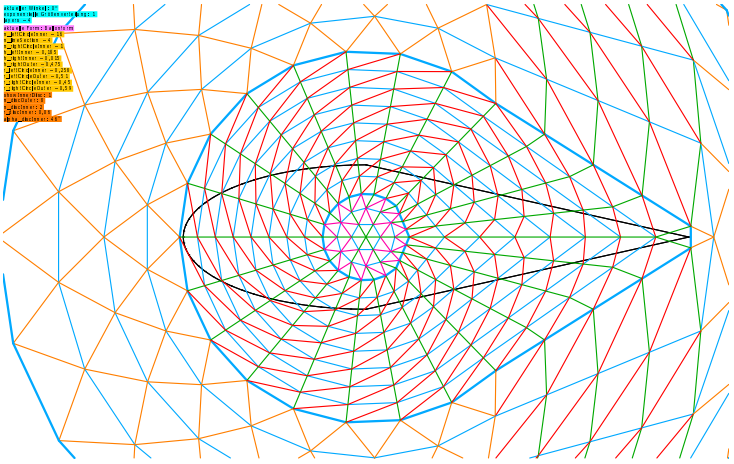


Figure 10.6: Balloon shape in the region below the blade.

For the volume mesh generation between the shells, an in-house software is used. Given four rings of enumerated nodes, an upper and a lower ring on the inner shell (red) and an upper and a lower ring on the outer shell (blue) (cf. Figure 10.8), prisms between two neighbouring shells can be constructed. Figure 10.8 shows how the nodes on these four rings are assigned to the prisms. The array "index" stores the number of six nodes for the creation of one prism. In the upper picture of Figure 10.8, the red prism is generated by the nodes 302, 4, 3, 322, 24 and 23. There are two different types of prisms, one with their rectangular face on the inner shell and those with their rectangular face on the outer shell (see Figure 10.8). For the generation of another prism having its rectangular face on the inner shell, all numbers of the stored values in the array "index" just have to be added by one. The initial numbers of the nodes are given by the *lineOffset* value for each ring of nodes. In order to create a prism having its rectangular face on the outer shell, the values of the array "index" have to be adapted. Within the software code, this is done by changing the Boolean *innerTr* from "true" to "false".

Out of one prism three tetrahedrons are built similar to the mesh generation of the SSMUM approch.

Due to the changing geometry of two neighbouring shells in the region below the blade, a different division of the prisms is applied and can be read in [CET08]. A top view of four surrounding shells around a turned blade is shown in Figure 10.9.

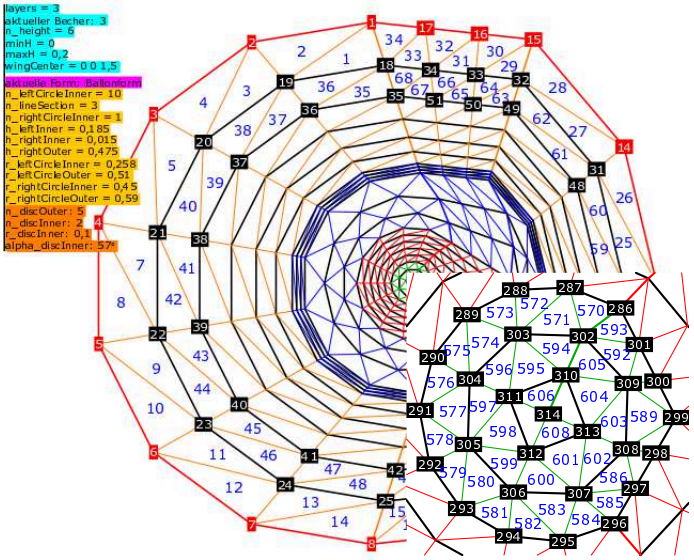



Figure 10.7: Outer shell with enumerated nodes and triangles.

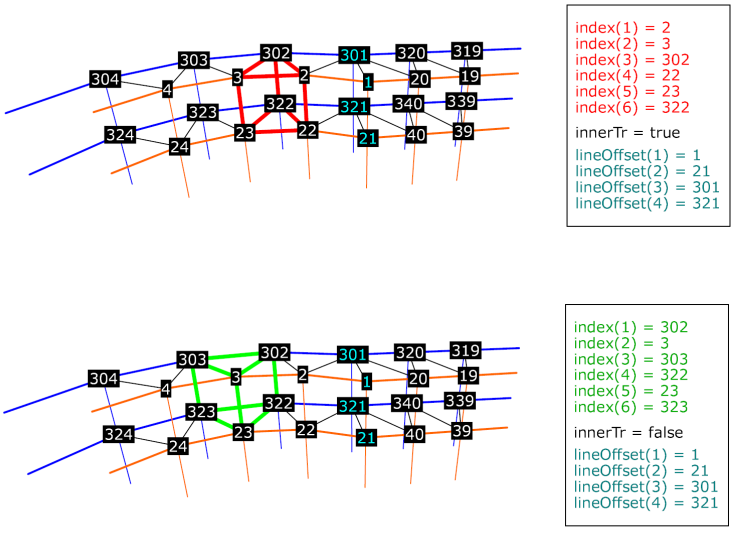



Figure 10.8: Generation of prisms between the shells.

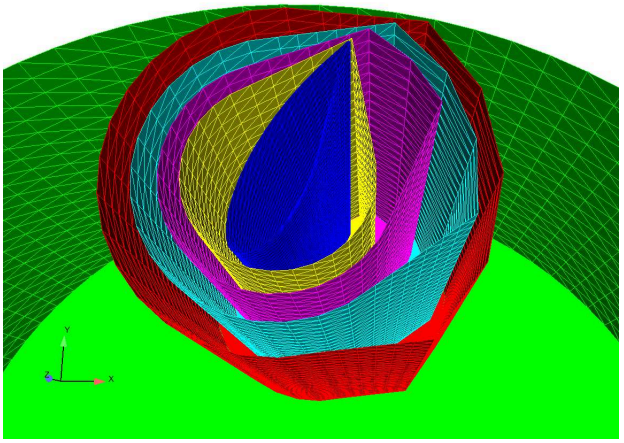


Figure 10.9: Shells around one blade turned at 63°.

# 11 Mesh Quality Evaluation

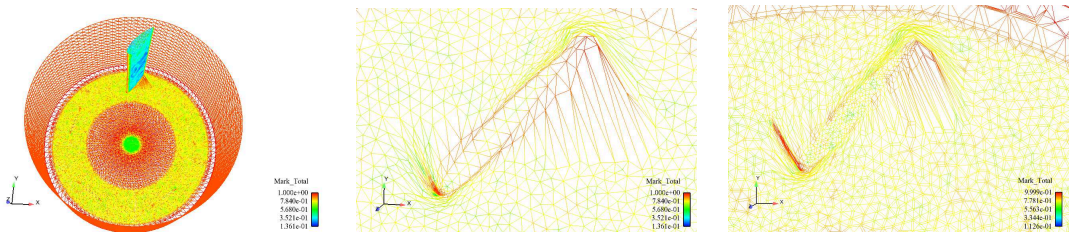
Given the two different mesh moving schemes for the motion of the single blades of a VSP, the feasibility of these approaches regarding their application to the simulation of the fluid flow have to be compared. One of the most important issues when dealing with deforming finite element simulations is the mesh quality in the deformation zones. If the element quality is low, the convergence rate is low, even to a degree that a flow solution cannot be obtained. In section 11.1 the mesh quality criteria for the evaluation of the mesh quality is presented. This critera is not only used to assess the final mesh quality but also to assess intermediate mesh adjustments during the mesh generation process, especially during the EMUM mesh generation process. Such an example is given in section 11.2. Herein, the goal is to compensate the mesh deformation around the blade and keeping an element quality of the deformed elements which is as good as possible as well as reducing the number of elements.

## 11.1 Mesh Quality Criteria

In [ESW05], a proof can be found that the norm of the discretization error is directly bounded by the inverse of the minimum angles of the triangles of a mesh, given a finite element approximation of the Poisson equation. To achieve a similar a priori discretization error bound for the finite element discretization of the Navier-Stokes equation is generally regarded as mathematically complex [ESW05] and is not within the scope of this thesis. However, it can be assumed that the integration rules that are implemented for optimally shaped elements become more and more imprecise the more an element degenerates. So, the challenge of the finite element mesh moving approach for the VSP propulsion system is to keep the average deformation of the elements in the critical zone between the blades and the nozzle plate as low as possible to ensure convergence of the fluid flow solution. Basically, the average quality of these critical elements must be kept as good as possible. For the evaluation of the element quality, the standard criteria taken from the Gridgen mesh generation software are applied. These are the radius ratio, the angle ratio and the mean ratio criterion. The radius ratio is calculated by dividing the inner radius of a tetrahedron by the third of its outer radius; the angle ratio accounts for each deviation of the twelve angles of the considered tetrahedron compared to an optimal tetrahedron, where the absolute value of the greatest deviation from the optimal angle is a measure for the quality of the element.

The mean ratio criterion depends on the eigenvalues of the matrix that maps the element under examination to a perfect element whose edges all have equal lengths. All criteria are formulated such that a perfect element gets a value of zero and the worst

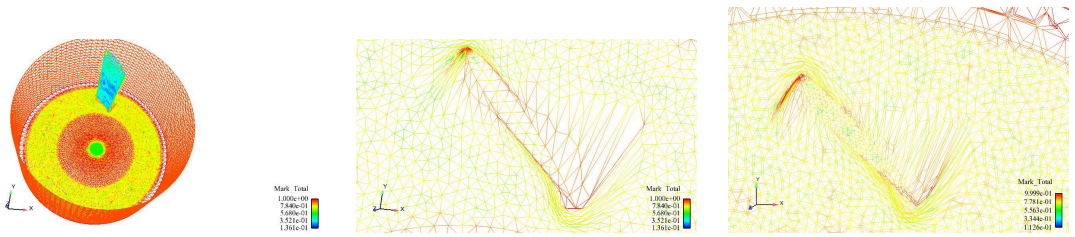
possible element obtains the value one. These values are calculated in a Fortran routine and visualized in Ensight, so that scripting and faster viewing is possible. As each of the three single criteria evaluates some critical elements with a slightly different score, a safe way to make all badly shaped elements visible is to compute the mean value out of the three single values. This value is called MarkTotal and shall be used to compare all upcoming mesh concepts regarding their element qualities. For example, the combined EMUM disc approach can be discussed focusing on the MarkTotal values of the mesh. In Figure 11.1, the most critical elements such as the strongly deformed elements in the horizontal planes at the level of the bottom tip of the blade and one millimeter below the blade and the SSMUM elements are shown.



General view Cut z=0.026m Cut z=0.025m

Figure 11.1: Zones of critical mesh quality at a blade angle of $+55\,^{\circ}$ degrees.

From the "MarkTotal" value of at least 0.8 one can observe that the elements in both horizontal planes are highly deformed around the trailing and the leading edge of the blade due to the large distance from the center of rotation. The bad quality of the SSMUM elements with a MarkTotal value of almost one derives from a very small distance between the inner and the outer SSMUM beaker. Initially, these radii were chosen in that way in order to leave as much space as possible for the EMUM deformation and also for the construction of the shells considering the CSMM method. Clearly, the mesh quality of the SSMUM elements suggests that the mesh has to be improved in the SSMUM layer. Figure 11.2 shall emphasize that the element quality is independent of the direction of the blade turning as the mesh quality is similar for both a positive and a negative blade angle turn of 55 degrees.



General view Cut z=0.026m Cut z=0.025m

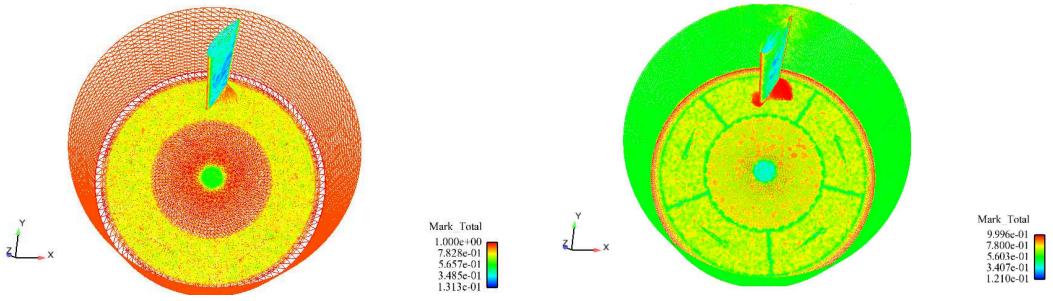
Figure 11.2: Zones of critical mesh quality at a blade angle of $-55\,^{\circ}$ degrees.

## 11.2 Evaluation of Various SSMUM-EMUM Mesh Concepts

In this chapter, the main steps improving the mesh quality of the EMUM disc approach are presented. The first SSMUM-EMUM mesh configuration shown in Figure 11.1 provides as much space as possible for the deformation of the EMUM elements as explained in the last section. For that reason, and because the closely situated supports of the nozzle plate prescribe a maximal possible radius for the outer SSMUM beaker, the SSMUM elements are squeezed. Observing that the strong deformation does not spread to the most outer radius of the EMUM discs, a second mesh concept is developed which widens up the SSMUM elements. The quality of the SSMUM elements is improved as well by increasing the number of SSMUM elements on the circumference. However, the higher mesh resolution, and at the same time different mesh topology of the new inner SSMUM layer compared to the lowest EMUM disc, causes the grid generation between these layers to become very problematic. This is due to fact that the finer mesh of the SSMUM beaker does not match well the coarser mesh of the EMUM disc, especially given the small distance between the two different layers of mesh nodes. For that reason, the lower disc is arranged further away from the inner SSMUM beaker. Furthermore doubling the domain mesh resolution in the plane of all the EMUM discs leads to a better element quality between the inner SSMUM layer and the bottom EMUM disc, especially away from the zone of high element deformation. This new mesh design should also explore how the EMUM deformation is solved for only two discs instead of four. By that, the total number of elements can be reduced significantly, and furthermore, by reducing the number of discs, a higher distance between the discs can be realized, which together with the higher node resolution on the discs results in a better element quality in the space between the discs. Comparing the four-disc with the two-disc mesh configuration in Figure 11.3, many differences regarding the element quality can be observed. Firstly, the SSMUM elements improve their "MarkTotal" grading from at least 0.8 down to below 0.6, and secondly, the elements in the zone where the deformation is not strong on the EMUM discs also improve their quality from a MarkTotal value of about 0.78 to about 0.7 and less. In Figure 11.4, the element quality of the first disc attached to the bottom of the blade (disc 1) and the disc below (disc 2) is shown for the two concepts. Because the two disc approach has a higher mesh resolution on the discs and an increased distance between the two discs compared to that of the four-disc concept, the elements are generally less squeezed between the discs and this ends up in a better element quality.

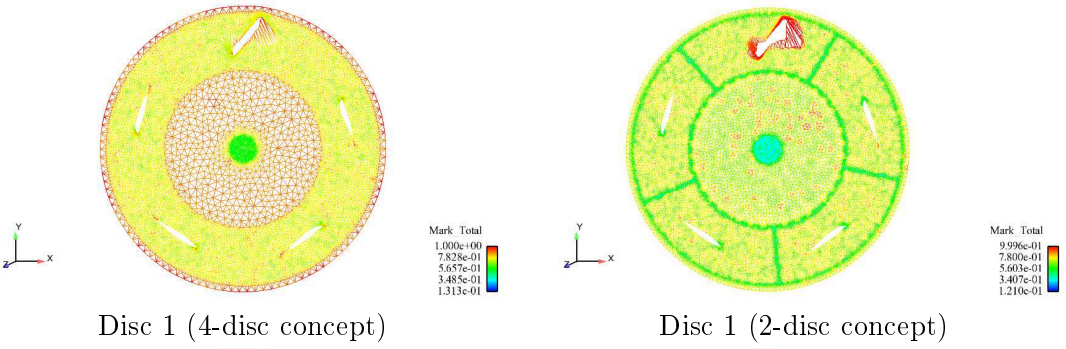
On the other hand, the two disc concept shows elements around the blade with a worse quality. Here it becomes evident that a higher number of discs between the bottom of the blade and the inner SSMUM beaker enables the EMUM solver to maintain a higher element quality in the region of strong deformation. A zoom of the most critical region of the mesh deformation around the bottom of the blade and its corresponding element quality is presented in Figure 11.5 .

For the final EMUM mesh configuration, all the observations regarding the achievement of a better element quality in the critical zones are considered. In addition, the number of elements is reduced where it is possible. So, in the region of little or no deformation the number of elements is reduced by decreasing the mesh resolution in these zones.

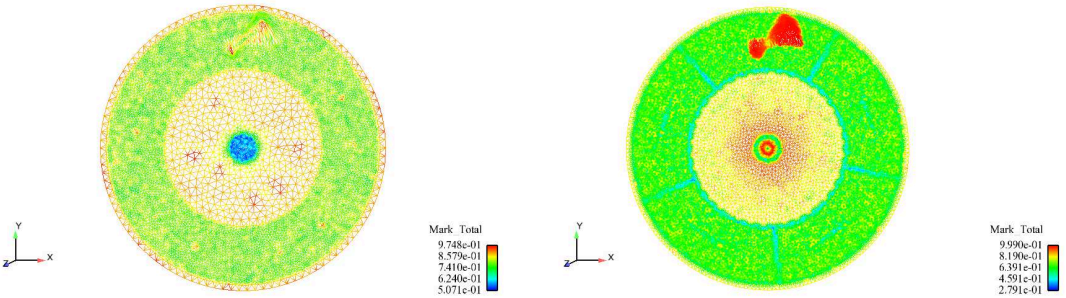



4-disc concept 2-disc concept

Figure 11.3: Refinement of the SSMUM elements.



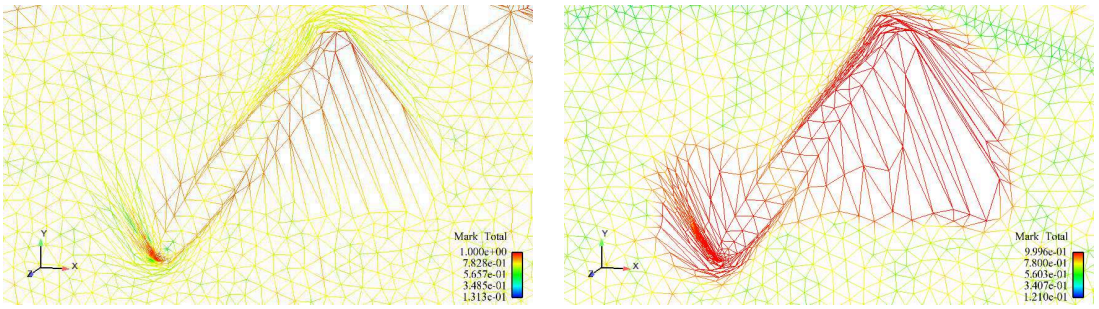

Disc 1 (4-disc concept) Disc 1 (2-disc concept)




Disc 2 (4-disc concept) Disc 2 (2-disc concept)

Figure 11.4: Disc refinement.

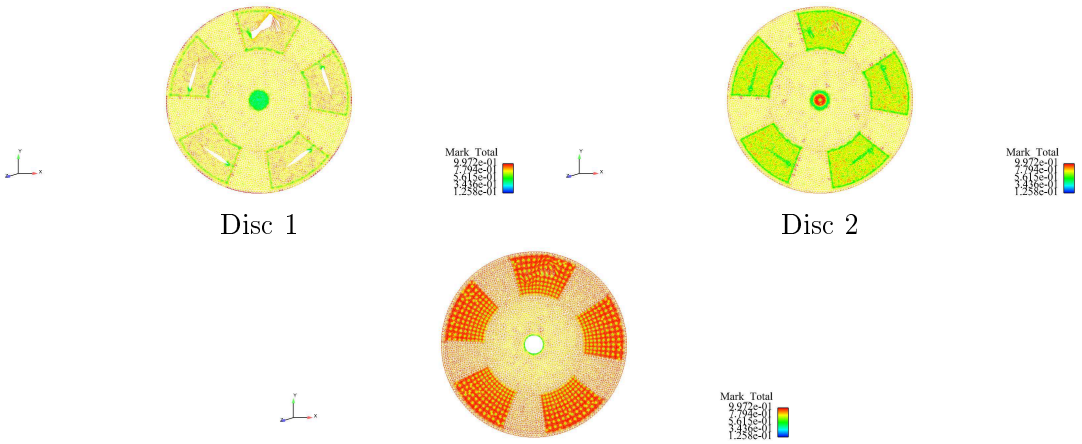
Furthermore, the number of discs is set to three in order to enable the EMUM solver to maintain a better quality of the elements in the region below the bottom of the blade compared to the two disc approach. The high resolution around the blades is kept as well,




Zoom disc 1 (4-disc concept) Zoom disc 1 (2-disc concept)

Figure 11.5: Deformation on the top disc around the bottom of the blade.

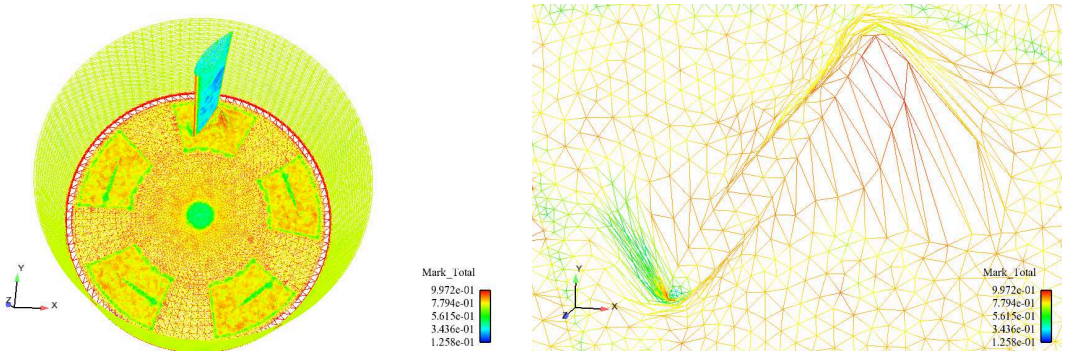
because the aspect ratio of these elements is better the higher the resolution of the disc mesh is, and a higher element quality of initial non-deformed elements is a prerequisite to a better element quality in the deformed state. The high number of SSMUM elements in the circumference is reduced again in order to position the lowest of the three discs closer to the inner SSMUM beaker, which gives more space for the EMUM deformation. The fact that the deformation does not reach the outer radii of the discs allows a smaller diameter of the inner SSMUM beaker. This gives enough possibility to improve the element quality of the SSMUM layer. The quality of the final SSMUM-EMUM mesh configuration can be seen in Figures 11.6 and 11.7.




Disc 1 Disc 2

Disc 3

Figure 11.6: Element quality on the discs in the final mesh configuration.




Overview Zoom disc 1

Figure 11.7: Element quality of the final mesh configuration.

The overall element quality of the final mesh configuration is better in the SSMUM region and in the close region of the bottom of the blade compared to the four disc and the two disc approach. On the bottom of the SSMUM layer and on the bottom EMUM disc no improvement can be achieved due to the different mesh topologies of the lowest EMUM disc and the inner SSMUM beaker and their very close distance to each other. It has to be emphasized that the mesh quality, especially at the leading and trailing edge of the blade, is also improved when using the final mesh configuration. In addition to that, the three disc concept reduces the number of elements by the factor of two compared to the four-disc concept.

# 12 Comparison of the CSMM and the EMUM Methods in the Context of a Refinement Study

The last two chapters showed how to develop two different mesh moving approaches, the CSMM and the EMUM concept, to account for the blade motion inside the rotating SSMUM beaker mesh, ensuring lack of tangling and good-quality mesh motion at the same time. In this chapter, these two approaches are compared regarding their feasibility in the context of a refinement study of the flow around a single deflected blade. The aim of this chapter is to provide arguments for either the CSMM or the EMUM concept application when setting up the mesh for the final ten-blade Voith Schneider propeller flow simulation. These two methods should be evaluated according to their feasibility regarding a practicable mesh generation involving various mesh resolutions on the blade surface and their smooth transition to the coarser outer zones of the entire mesh. In addition to this detailed comparison of these two mesh moving approaches, the outcome of two independent refinement studies of two different mesh structures can provide more knowledge about the discretization error due to the resolution of the mesh and the mesh quality. Depending on the results, the fluid flow simulation could be regarded as mesh-independent in a twofold sense, independent of mesh resolution and mesh structure.

## 12.1 Scope of the Refinement Study

In the course of this refinement study, the absolute value of the force exerted by one single blade of the VSP propulsion system without the nozzle plate on the fluid should be determined; the nozzle plate is not included in the refinement study due to limited computing resources. The fluid enters the computational domain with a velocity of 1 m/sec at the inflow, which corresponds to the left wall in Figure 12.1, and hits the blade, which is turned to 55 degrees according to the maximal angle of blade deflection considering the blade angle curve for the operating point "open water speed" given by Voith, compare Figure 5.1. In this way, the worst possible element quality that can be encountered during a full ten-blade simulation is taken into account if an almost reversible mesh EMUM deformation during various consecutive VSP rotations is assumed. In the scope of the refinement study, meshes with continuously increasing resolutions are created and the above described fluid flow simulation is performed on each mesh. In the course of these simulations, the absolute values of any fluid flow variables and especially the force exerted by a single deflected blade of the VSP propulsion system on the fluid should show a continuously decreasing deviation with an increasing mesh resolution.

By that, an estimation of the discretization error of the specific fluid flow simulation with respect to the mesh resolution can be made. Taking these errors into account, the most adequate mesh resolution for the full ten-blade VSP propulsion system fluid flow simulation considering the computing resources and the expected discretization error can be chosen.

## 12.2 Mesh Design

For both methods, the CSMM and the EMUM, four different resolutions on the blade surface are designed. Due to limited computing resources, only the flow around a single blade instead of all ten blades of the propulsion system is examined. Compared to the mesh concept in Figure 8.7, the surrounding mesh volume is reduced, whereas the rest of the entire mesh concept is kept, e.g., the SSMUM beakers, the EMUM discs and the shells of the CSMM method, see Figure 10.9. By that, the observations made during the single blade simulations can be transferred to the full ten blade fluid flow simulations. In order to resolve the fluid flow, which evolves closely to the blade surface as well, the mesh resolution there is chosen to be the most fine of the whole mesh. To save computing resources, the resolution at the exterior zones of the mesh is set coarser; the coarsest at the bounding walls of the computational domain. In between the interior zones of the mesh around the blade and the exterior zones close to the bounding walls, the transition of the mesh resolutions should be as smooth as possible. The common mesh design, here for the blade surface resolution of 40 mm, can be seen in Figure 12.1, the customized one for both the EMUM and the CSMM approach in Figure 12.2.

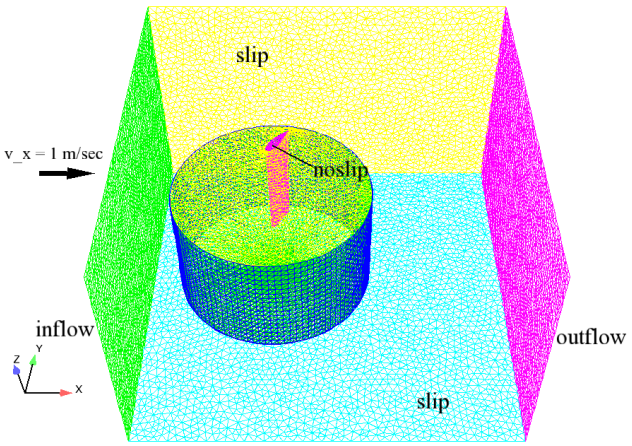
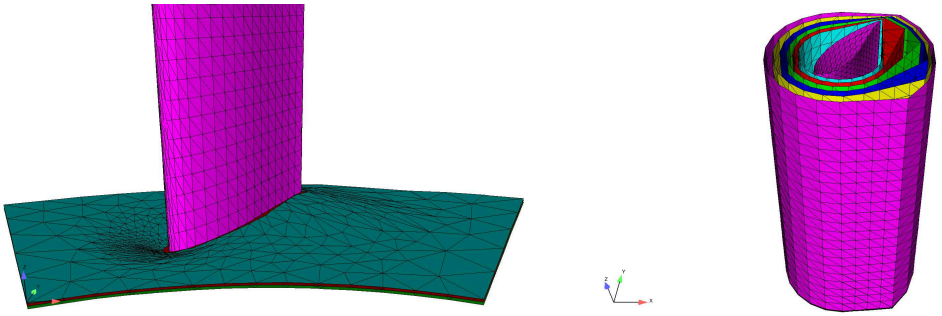



Figure 12.1: Common mesh design for the refinement study.

The meshes are labelled according to their resolution on the blade surface, e.g. EMUM.05 or CSMM.40 would correspond to an EMUM mesh with a blade surface node distance of 5 mm and a CSMM mesh with a node distance of 40 mm on the blade.



EMUM mesh CSMM mesh

Figure 12.2: Customized mesh designs for the refinement study.

In a further refinement study, the influence of the squeezed and distorted elements between the bottom part of the blade and the interior SSMUM beaker on the resulting fluid flow is examined. This is realized by comparing the results of the first refinement study with the results of a refinement study where the elements below the bottom of the blade have a better quality. To achieve this, these meshes are constructed with arbitrarily large space for the elements below the blade. The SSMUM beakers are not excluded from the mesh, but they are changed in shape in order to prescribe the mesh resolution in region around the blade to aim a smoother distribution of the mesh resolution from the fine blade surface to the coarser outer regions of the mesh. The common mesh concept of the second refinement study can be seen in Figure 12.3. These meshes are called modified meshes in the following; e.g., EMUM.20.mod would label a modified EMUM mesh with a blade surface mesh resolution of 20 mm.

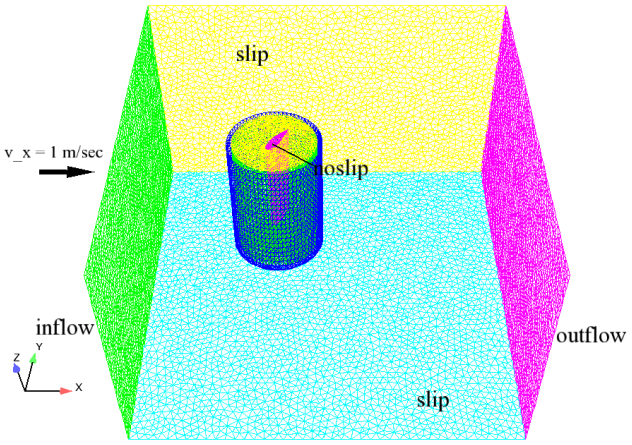



Figure 12.3: Common modified mesh design for the refinement study.

## 12.3 General Mesh Parameters

In order to realize different mesh resolutions for the above described mesh designs, various surface domain and mesh volume solver parameters are used, which are explained in the following. As mentioned, four different mesh resolutions on the blade surface have to be created, so the node distances (dsWings) of 40, 20, 10 and 5 mm are chosen. Corresponding to these four different mesh resolutions on the blade, the resolutions on the SSMUM beakers and on the outer domains of the volume (dsBox) are adapted. The topology of the SSMUM beakers is varied by the parameters ncircum, which is equal to the number of nodes around the circumferential, ncylinder, which is equal to the number of nodes along the vertical part of the SSMUM beaker, and ncone, which is equal to the number of nodal rings on the horizontal bottom of the SSMUM beakers. All these surface mesh parameters are shown in Figure 12.4. Once the surface mesh parameters are set,

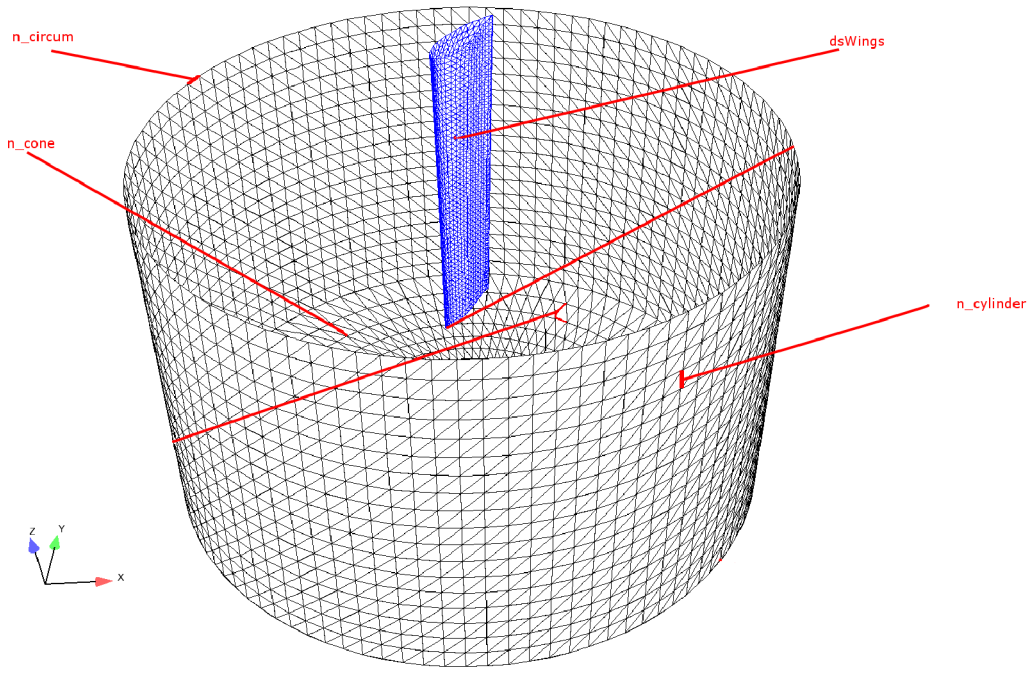
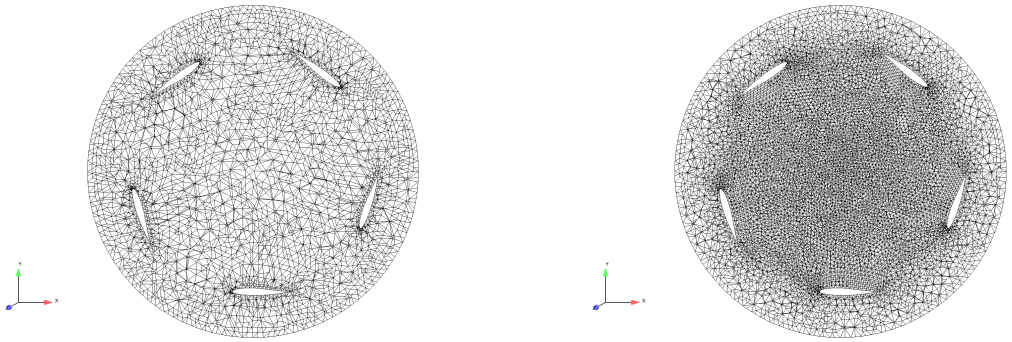



Figure 12.4: General mesh parameters.

the parameters for the volume mesh generation can be selected. The most important one is the parameter boundary decay, which weighs the influence of the bounding surface mesh resolutions on the inner parts of a volume mesh. The influence is maximal when this parameter is set equal to one and minimal when it is set equal to zero. A good example how this parameter affects the mesh resolution is given in Figure 12.5. Using a boundary decay value of 0.25, the highly resolved blade surface does not influence the inner parts of the volume mesh, whereas a value of 1.0 results in a highly refined inner volume mesh. The parameter boundary decay can be used in the same way when setting

the influence of the resolution of bounding lines, defined as connectors in Gridgen, of a surface mesh to the resulting inner resolution of that surface mesh.



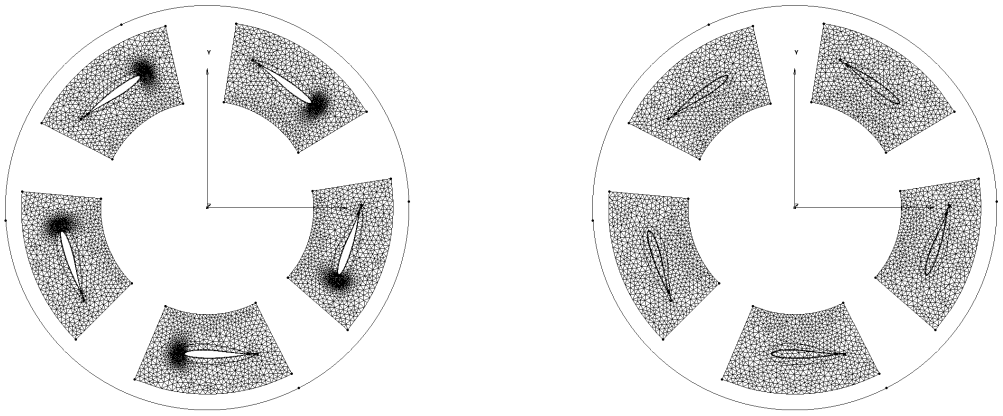
Boundary decay = 0.25 Boundary decay = 1.0

Figure 12.5: Influence of the mesh parameter boundary decay.

## 12.4 EMUM Meshes

For the refinement study, the EMUM mesh concept presented in Figures 11.6 and 11.7 is slightly modified. As mentioned in the last paragraph, the resolution of the SSMUM beaker is adapted to the resolution on the blade. The EMUM discs are then modified in such a way that the transition of the resolution between the bottom EMUM disc and the SSMUM beaker is as smooth as possible. This is realized by setting the resolution of the connectors of the bottom EMUM discs equal to the resolution on the SSMUM beaker. In order to create a smooth transition to the blade surface resolution, the parameter boundary decay is set to one when creating the top EMUM disc domain. As a consequence, the meshes of the top and the bottom EMUM disc differ considerably (see Figure 12.6).

Another modification to the EMUM concept shown in Figure 12.6 includes the extension of the highly resolved zones around the blade. This is a necessity as the smaller zones of the former version cause tangling with finer meshes when turning the blade. The number of EMUM discs, however, is not changed and remains three. As the described variations of the EMUM concept are independent of the blade resolutions, the mesh can be generated via a Glyph script, and so the whole mesh generation for any new resolution on the blade is easy, but the surface mesh parameters have to chosen carefully and often iteratively in order to achieve smooth transitions in the mesh resolution. It can be seen in Figure 12.7 that the transition from the fine region around the blade to the coarser zones of the SSMUM beaker is quite smooth, for both the blade resolution of dsWings equal to 40mm and 20mm. The finer meshes however present a quite dense mesh around the blade and quite a hard transition from the dense mesh to the coarser outer mesh. In Figure 12.8, a horizontal cut at the height of the blade bottom visualizes quite well the element quality in that region. The quality in that critical region is almost independent



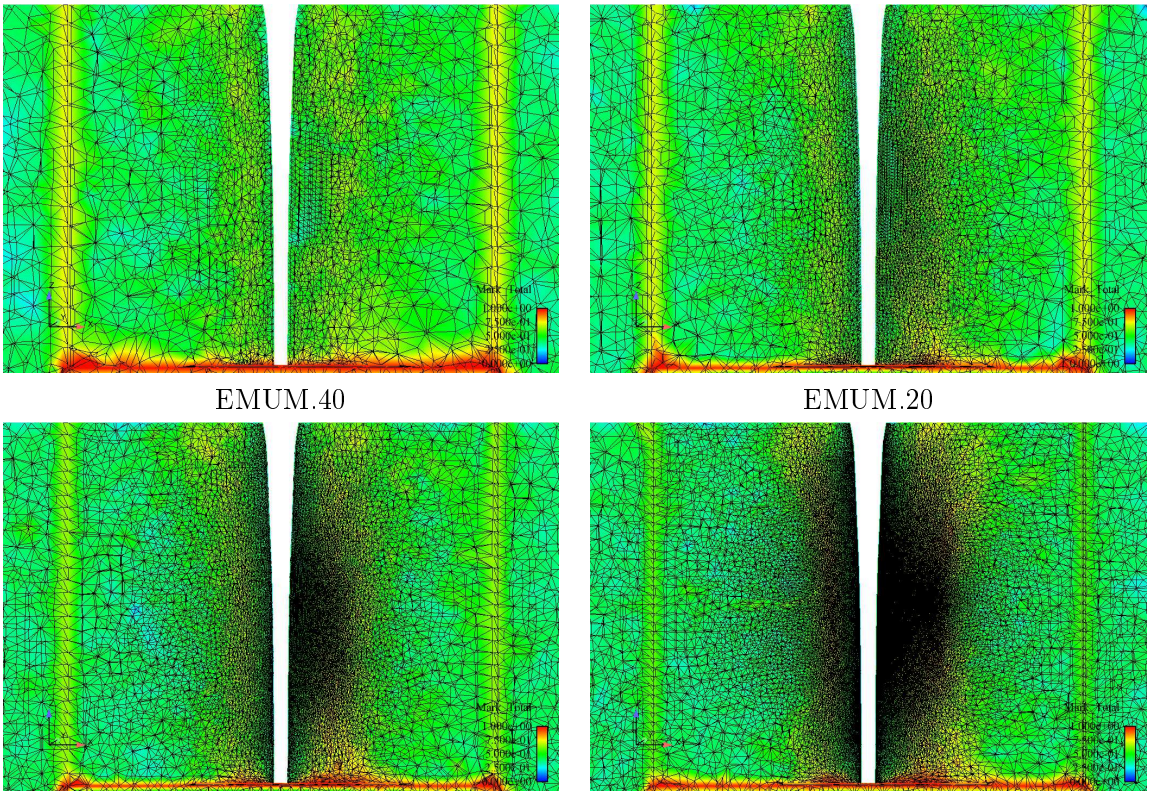
Top EMUM disc Bottom EMUM disc

Figure 12.6: New EMUM discs.

of the blade resolution. At least, it is not getting worse with a higher resolution on the blade surface. Instead, the coarsest blade resolution of dsWings equal to 40 mm generates worse elements in the EMUM discs. With the same vertical node distance between the EMUM discs, the aspect ratio is worse, in this case because of the bigger horizontal element size caused by the bigger node distances on the blade. Compared to the element deformation obtained by the deformation algorithm proposed by Chiandussi [CBO00] shown in Figure 12.9, this approach shows similar squeezing at the trailing edge, but the foil pitch shown in Figure 12.8 is almost doubled.

A comparison to the numerical experiments done by Masud [MBK07] as shown in Figure 12.10 demonstrates the element quality in the horizontal plane to be better than that shown in Figure 12.8. However, it must be mentioned that considering the application shown in 12.10 the horizontal space to absorb the deformation is much bigger than in the VSP application with the limited space for the deforming mesh around the blades. Furthermore, the pitch in the VSP application is almost doubled, fifty-five degrees compared to thirty degrees and lastly, a 3D deformation is considered here compared to the 2D applications in the aforementioned literature. Nevertheless, a study including all possible parameter variations regarding the element stiffening factors as described in Chapter 7.2.5 was carried out for the VSP application but did not show any improvement of the element quality.

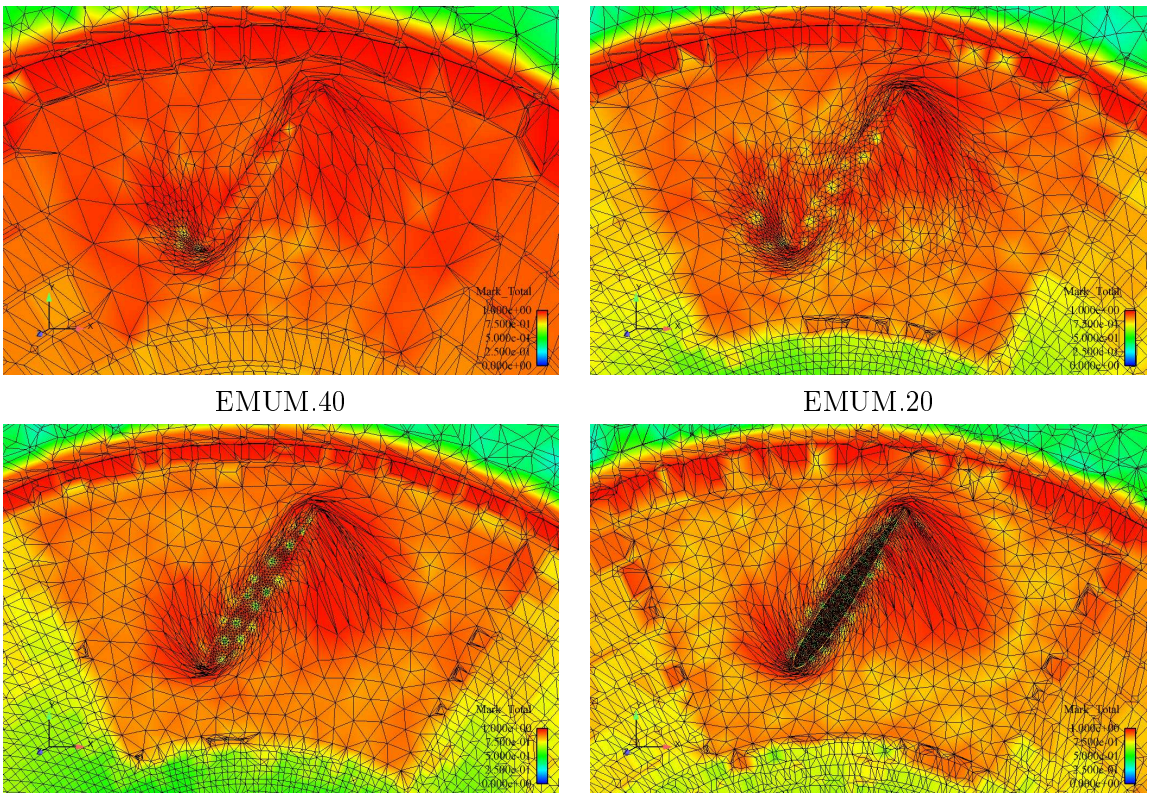
The figures in Subsections 16.1.1 and 16.1.2 of the appendix will show the final EMUM meshes and their mesh generation parameters used for the refinement study. The horizontal cuts taken at the middle of the blade as well as the zoomed mesh area at the bottom of the blade again show quite smooth transitions between the regions of different mesh resolutions.



EMUM.40 EMUM.20

EMUM.10 EMUM.05

Figure 12.7: Vertical element distribution.




EMUM.40 EMUM.20

EMUM.10 EMUM.05

Figure 12.8: Horizontal element distribution at the blade bottom.

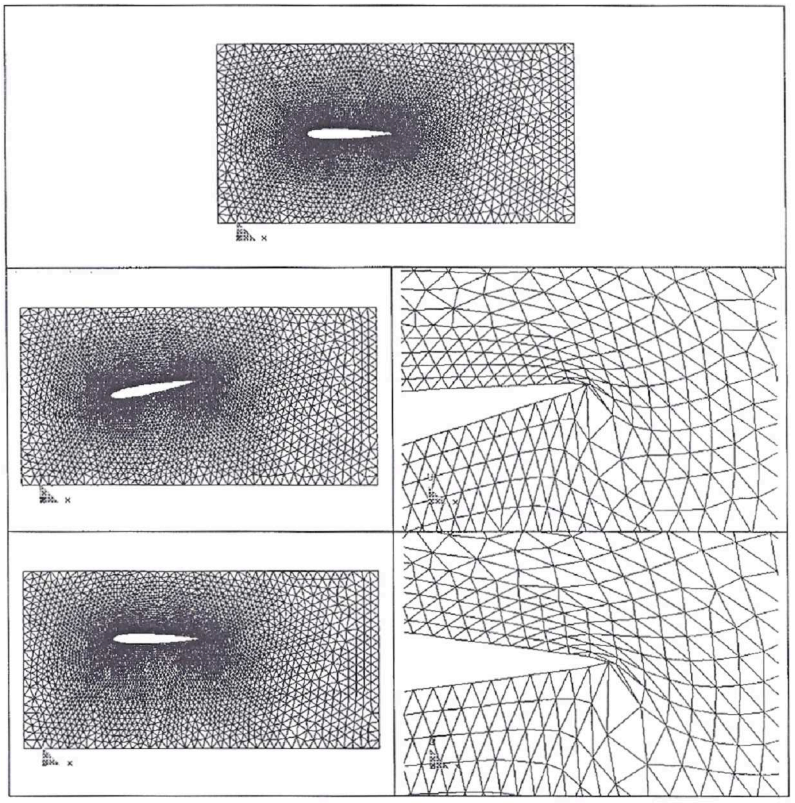


Figure 12.9: Element deformation at the trailing edge of a NACA0012 aerofoil [CBO00].

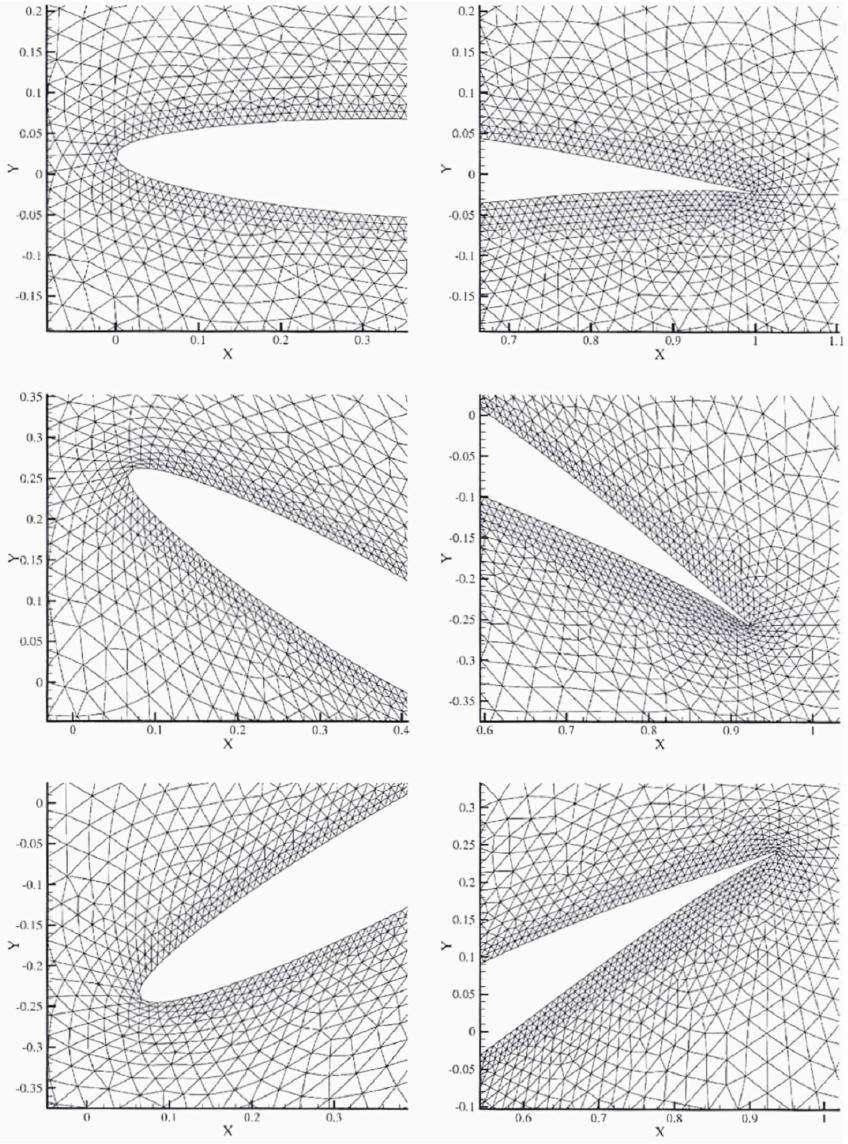
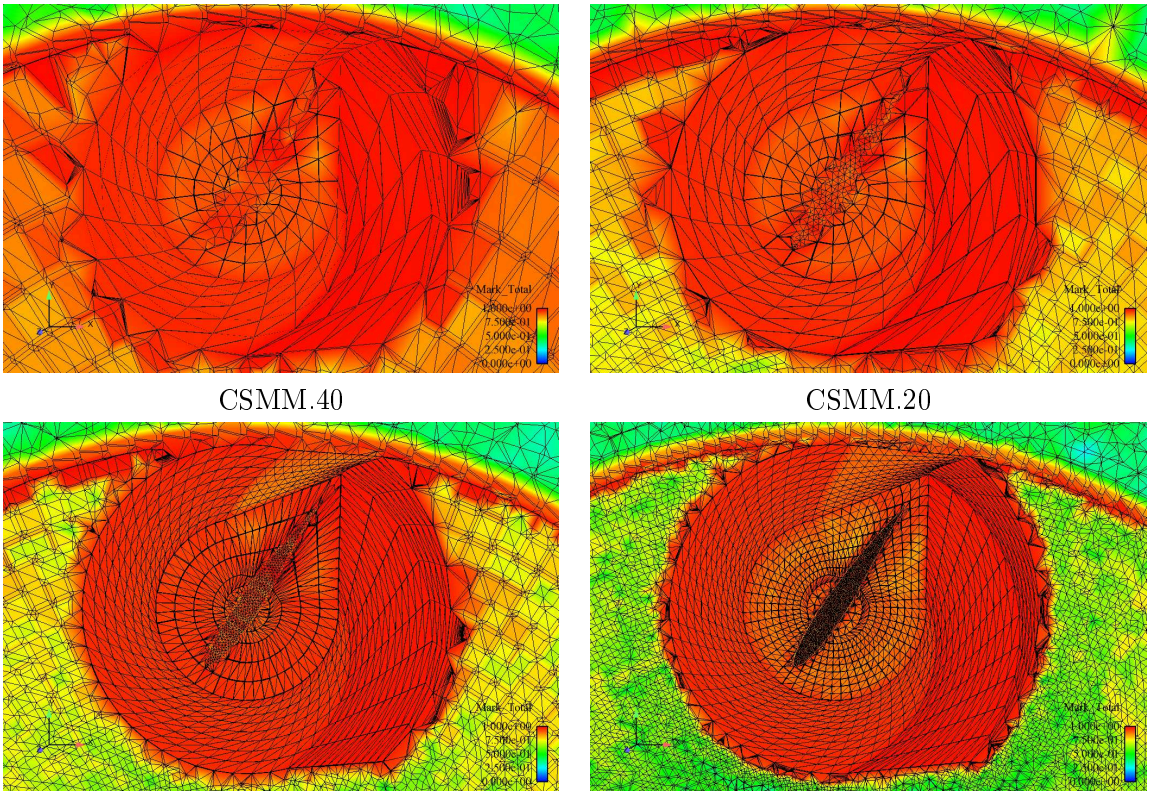


Figure 12.10: Zoomed view of the tip and tail of an airfoil at various stages of motion [MBK07].

## 12.5 CSMM Meshes

Considering the CSMM approach, a smooth distribution of the mesh resolution inside the volume mesh is much more difficult. As consecutive finer resolutions on the blade surface require a finer resolution in the closer regions of the blade, the number of shells around the blade have to be increased. But a higher number of shells automatically creates flatter elements in the critical zone between the bottom of the blade and the inner SSMUM beaker. For all four dsWings values (5 mm, 10 mm, 20 mm and 40 mm) this results in a worse element quality in this region, indicated by a mainly red zone referring to a MarkTotal value close to one in the horizontal plane at the bottom of the blade shown in Figure 12.11. On the contrary, the EMUM approach does not result in such bad quality elements (cf. Figure 12.8). As an alternative, the number of shells of the CSMM approach could be kept constant for different dsWings values while constructing the shells closer to each other, but very closely situated shells produce tangling when assuming a constant relative rotation between the shells.



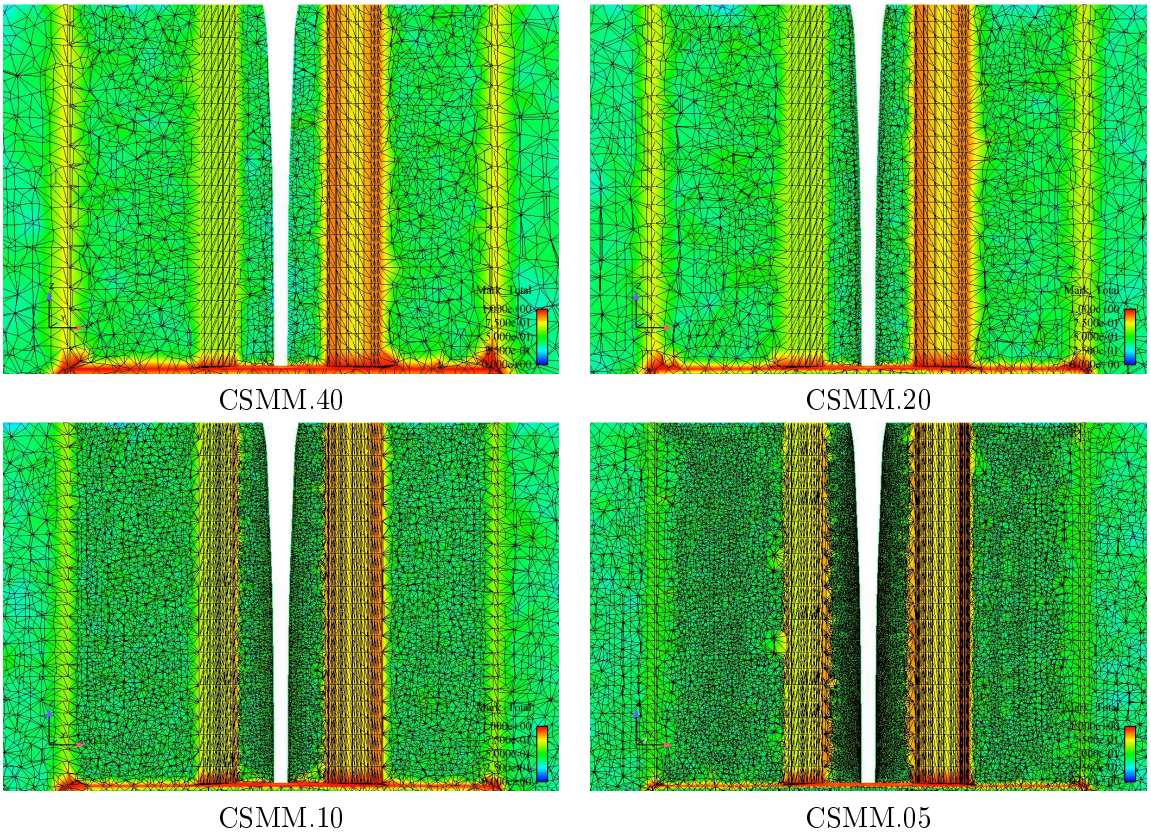
CSMM.40 CSMM.20

CSMM.10 CSMM.05

Figure 12.11: Horizontal element distribution and quality at the blade bottom.

Another inherent disadvantage of the CSMM concept implemented so far is that the vertical resolution of the shells cannot be changed from an inner to the next outer shell.

By that, the shells constitute an obstacle for the smooth distribution of the mesh resolution from the fine part at the blade to the coarser part at the inner SSMUM beaker; this is shown by the vertical planes through the mesh in Figure 12.12.



CSMM.40 CSMM.20

CSMM.10 CSMM.05

Figure 12.12: Vertical element distribution.

The mostly yellow coloured shell elements in the whole region from the top of the blade to its bottom indicate that the single elements inside the shells get more deformed and have a worse quality than the elements of the EMUM concept in that zone; compare Figure 12.12 with Figure 12.7. In addition, the mesh generation between the inner shell and the blade surface is difficult, as it is hardly possible to create a smooth transition from the fine resolution on the blade surface to the coarser resolution of the most inner shell. A boundary decay value of one gives too much influence to the coarse resolution on the inner shell, which results in too coarse meshes in the closer region around the blade; compare the left picture of Figure 12.13. Various combinations of boundary decay values create meshes with too many elements and considerably big holes, as seen in the right picture of Figure 12.13, and therefore must be discarded. Overall, these parts of the CSMM meshes take a lot of time to find the proper mesh generation parameter set varying the values of boundary decay. The same accounts for the mesh between the outer shells and the inner SSMUM beaker. In Subsections 16.1.3 and 16.1.4 of the appendix,

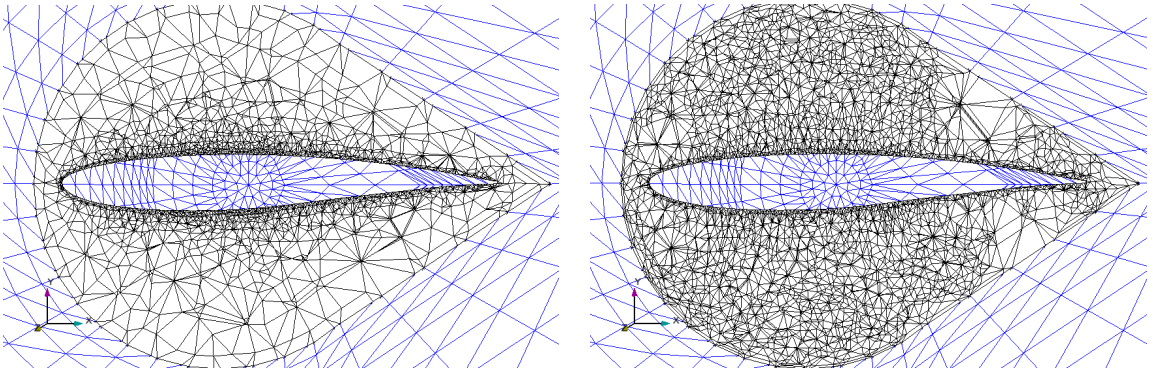


Figure 12.13: Element distribution between the blade and the inner shell.

the final CSMM meshes and their mesh generation parameters, which are used for the refinement study, will be shown.

## 12.6 The CSMM and the EMUM Method Applied to a Flow Simulation in the Context of a Refinement Study

The previous section shows that the CSMM approach is much more difficult to handle when it comes to the creation of smooth meshes with good quality elements. In this section, the sixteen meshes of both methods as generated in Sections 12.4 and 12.5 are used to simulate the flow around one turned blade as described in Sections 12.1 and 12.2. The fluid enters the computational domain with an x-velocity of 1 m/sec at the inflow, hits the blade and leaves the computational domain at the outflow, the right wall of Figure 12.1. There is a no-slip condition on the blade and a slip condition on the side walls as well as on the upper and the lower wall of the bounding domain. On the outflow, a no-stress condition is applied according to [DH03] :

$$-p + 2\nu \frac{\partial v_n}{\partial n} = t_n = 0, \tag{12.1}$$

$$\nu \left( \frac{\partial v_\tau}{\partial n} + \frac{\partial v_n}{\partial \tau} \right) = t_\tau = 0. \tag{12.2}$$

As the time-dependent Navier-Stokes problem requires an initial velocity field that must be divergence free, see Equation 2.8 in Chapter 2, a stationary Stokes problem needs to be solved in the first place, which can be formulated as the following:

$$-\nu \nabla^2 \mathbf{u} + \nabla p = \mathbf{0} \quad \text{in } \Omega \quad \text{(equilibrium)}, \tag{12.3}$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega \quad \text{(incompressibility)}, \tag{12.4}$$

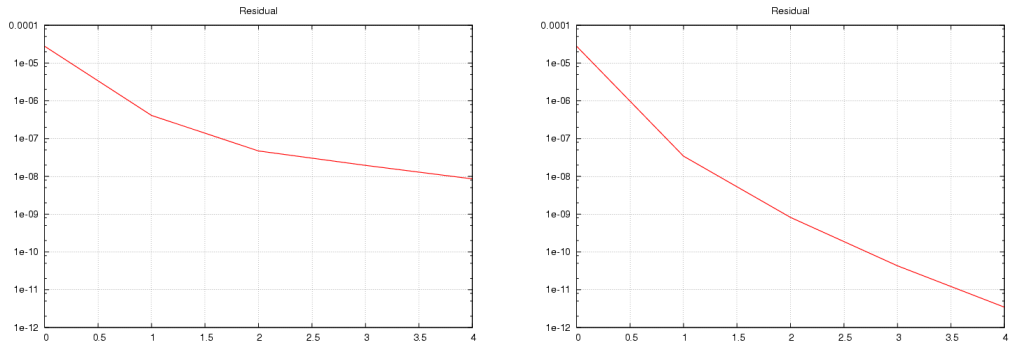
$$\mathbf{u} = \mathbf{u}_D \quad \text{on } \Gamma_D \quad \text{(Dirichlet b.c.)}, \tag{12.5}$$

$$-p\mathbf{n} + \nu \left( \mathbf{n} \cdot \nabla \right) \mathbf{u} = \mathbf{t} \quad \text{on } \Gamma_N \quad \text{(Neumann b.c.)}. \tag{12.6}$$

The resulting divergence free velocity field of the Stokes problem is then used as an initial flow field for the Navier-Stokes problem. However, as the Stokes solution generally varies too much from the real Navier-Stokes solution, a direct Navier-Stokes simulation of the water flow with the real water viscosity of $\nu = 10^{-6} \frac{\mathrm{m}^2}{\mathrm{s}}$ would become too unstable, and therefore a convergence of that type of computation is rather difficult. For that reason, intermediate consecutive Navier-Stokes simulations with decreasing viscosities such as $\nu = 10^{-2} \frac{\mathrm{m}^2}{\mathrm{s}}$ and $\nu = 10^{-4} \frac{\mathrm{m}^2}{\mathrm{s}}$ are performed, where the resulting flow field of a simulation with a higher viscosity serves as an initial flow field for next flow simulation with a lower viscosity. In order to equally compare the four different meshes of each mesh moving method, the EMUM and the CSMM approach, the Krylov space size (ninner) for the solver of the linearized equation system within one single nonlinear iteration (nit) within one time step is equal for all meshes and only varies depending on the value of the viscosity. The number of nonlinear iterations used within one time step is the same as well for each mesh. Only when the required computational time gets too high, and when there is no convergence detectable, a simulation is aborted in the sense that the number of nonlinear iterations is reduced. Of course, convergence problems can be solved if the

Krylov space size is increased, but this is not guaranteed and the computational cost is high as the computational time increases more than linearly with the Krylov space size.

In Figure 12.14 a case is shown where an augmentation of the Krylov space size improves significantly the convergence rate. The left and the right chart of Figure 12.14 show each a plot of the residual of a nonlinear equation system over four single nonlinear iterations, which are performed to simulate one time step of a fluid flow simulation. The left chart corresponds to a simulation where a Krylov space size of 250 is used to solve the linearized equation system within each nonlinear equation, whereas the right chart corresponds to a simulation where a Krylov space size of 500 is used to solve the linearized equation system within each nonlinear equation. As it can be seen, the convergence behaviour improves significantly by the augmentation of the Krylov space size, as the residual of the fourth nonlinear iteration declines from $10^{-6}$ to below $10^{-11}$ by doubling the Krylov space size.




Krylov space size 250 Krylov space size 500

Figure 12.14: Convergence behaviour with varying Krylov space sizes over four nonlinear iterations.

The corresponding values for the number of nonlinear iterations and the Krylov space size for each single simulation step regarding all the different meshes are chosen such that convergence is guaranteed in most cases and are presented in Table 12.1. The above described boundary conditions are the same for each mesh and each single simulation. Also, the number of time steps (nts) are the same for each simulation.

| type of simulation | `nit` | `nts` | `ninner` |
|---|---|---|---|
| Stokes | 10 | 1 | 500 |
| Navier-Stokes ($\nu = 10^{-2}\frac{\mathrm{m}^2}{\mathrm{s}}$) | 10 | 1 | 500 |
| Navier-Stokes ($\nu = 10^{-4}\frac{\mathrm{m}^2}{\mathrm{s}}$) | 10 | 1 | 1000 |
| Navier-Stokes ($\nu = 10^{-6}\frac{\mathrm{m}^2}{\mathrm{s}}$) | 25 | 1 | 500 |
| Navier-Stokes ($\nu = 10^{-6}\frac{\mathrm{m}^2}{\mathrm{s}}$) | 3 | 50 | 500 |

Table 12.1: Simulation solver parameters.

According to Table 12.1, a Stokes flow simulation is performed, then an intermediate one-time-step unsteady Navier-Stokes simulation with a viscosity of $\nu = 10^{-2}\frac{\mathrm{m}^2}{\mathrm{s}}$ and a time step size of 0.1 sec is done, then the viscosity is decreased in another two successive intermediate simulations to reach the water viscosity of $\nu = 10^{-6}\frac{\mathrm{m}^2}{\mathrm{s}}$. Finally, a fifty-time-step unsteady Navier-Stokes simulation is performed in order to calculate the value of the forces applied from the turned blade on the fluid over the time. The time step size is 0.1 sec and constant for all unsteady simulations.

The left charts of the following Figures 12.15, 12.16, 12.17 and 12.18 show the forces exerted from the blade on the fluid in all three dimensions over the last fifty time steps of the unsteady Navier-Stokes simulation with the water viscosity $\nu = 10^{-6}\frac{\mathrm{m}^2}{\mathrm{s}}$ calculated on the sixteen meshes. The force values orientated in the x-direction, y-direction and z-direction are visualized by the red, green and blue charts respectively.

The presented results of all sixteen meshes also include the convergence chart of the nonlinear residual of each simulation on the single meshes, including the Stokes flow simulation, the unsteady Navier-Stokes simulation with a viscosity of $\nu = 10^{-2}\frac{\mathrm{m}^2}{\mathrm{s}}$, $\nu = 10^{-4}\frac{\mathrm{m}^2}{\mathrm{s}}$ and $\nu = 10^{-6}\frac{\mathrm{m}^2}{\mathrm{s}}$ shown in right chart of Figures 12.15, 12.16, 12.17 and 12.18. A peak in the chart corresponds to the start of a new intermediate simulation or a new time step.

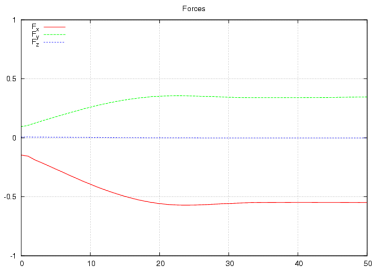
Comparing the simulations on the EMUM meshes according to Figures 12.15 and 12.16, the curve of each component of the blade force shows a similar behaviour. From an initial lower force in the x- and y-direction resulting from the unsteady Navier-Stokes simulation with the viscosity $\nu = 10^{-4}\frac{\mathrm{m}^2}{\mathrm{s}}$, the absolute values of the force components increase and reach their maximum between the 20th and the 22nd time step; the values can be considered as nearly converged at the last time step. All intermediate simulations start from a residual value higher than $10^{-6}$ and terminate at a residual value lower than $10^{-10}$, which we regard as a satisfying convergence behaviour. During the fifty-time-step simulations, the residuals do not decrease that much, because there are only 3 nonlinear iterations performed within one time step due to limited computing resources. To ensure that three nonlinear iterations are sufficient the calculated force values of two different simultations of seven time steps are compared; one done with three nonlinear iterations per time step and another with nine. It becomes apparent that the difference is negligible.

Looking at the results of the simulations on the CSMM meshes according to the Figures 12.17 and 12.18, an oscillating curve of the value of the force components can be detected in most of the cases. Obviously, the resulting nonlinear equation system of the unsteady Navier-Stokes equation is unstable and can therefore not be solved correctly. In those cases, already the Stokes flow cannot be computed with a sufficiently small residual. An augmentation of the Krylov space size from five hundred to one thousand does not show any improvement. However, the continuing Navier-Stokes simulations are performed with the unconverged Stokes flow fields as an initial flow field in the hope that the equation system is getting more stable in the course of several time steps. In fact, the values of the force components converge for all CSMM meshes and the residual values for the fifty time step simulations are similar to the ones performed on the EMUM meshes.

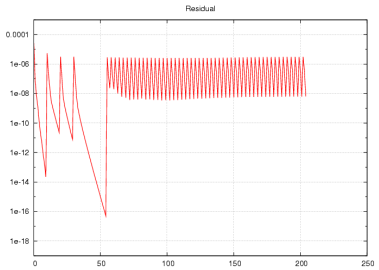
The high instability of the Navier-Stokes equation system computed on the CSMM

meshes probably derives from the zones where a smooth distribution of the element sizes in the mesh cannot be achieved. Such zones occur at the transition from the inner SSMUM beaker to the outer shell (e.g., zones one and three indicated in Figure 12.19) and at the transition at the inner shell to the mesh around the blade (zone two).

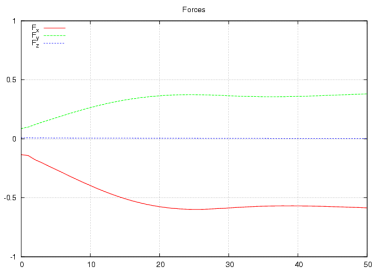
### 12.6.1 EMUM Mesh Simulation Results



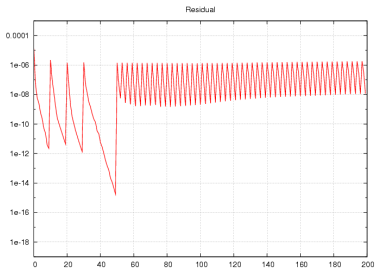

Forces EMUM.40



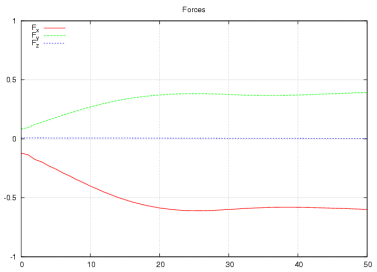

Residual EMUM.40



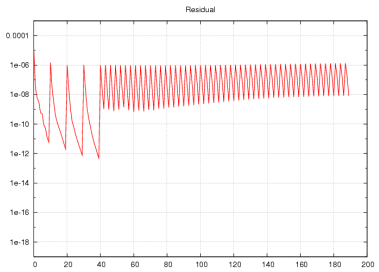

Forces EMUM.20



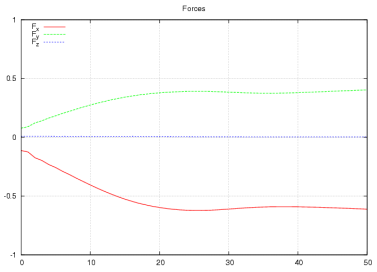

Residual EMUM.20



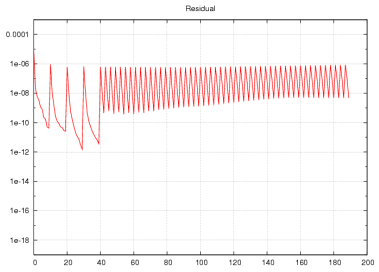

Forces EMUM.10




Residual EMUM.10



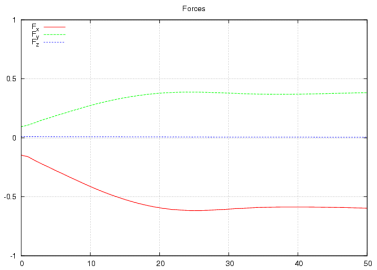

Forces EMUM.05



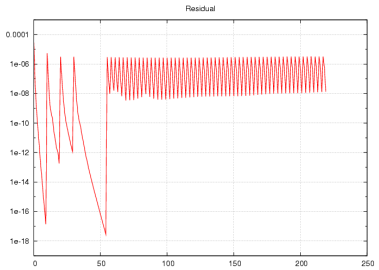

Residual EMUM.05

Figure 12.15: EMUM meshes: Calculated forces of the blade and residuals of the flow solution.

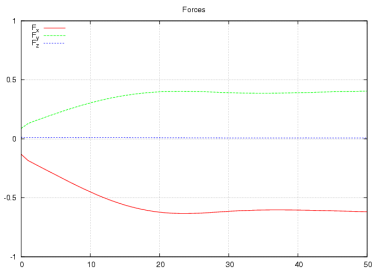
### 12.6.2 EMUM Modified Mesh Simulation Results



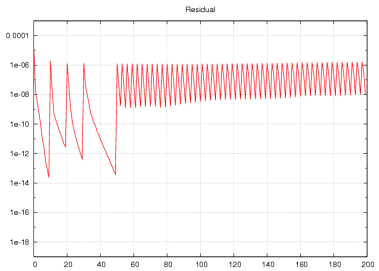

Forces EMUM.40.mod



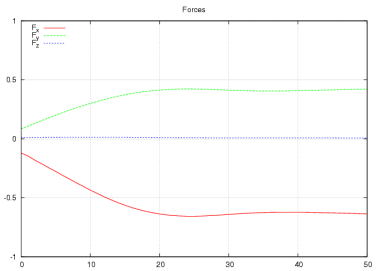

Residual EMUM.40.mod



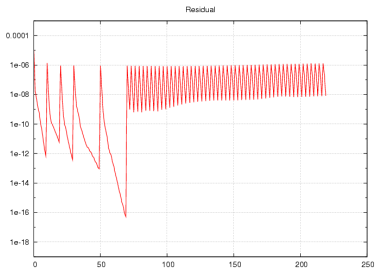

Forces EMUM.20.mod



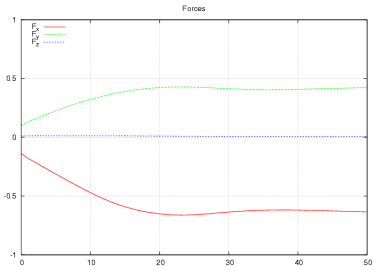

Residual EMUM.20.mod



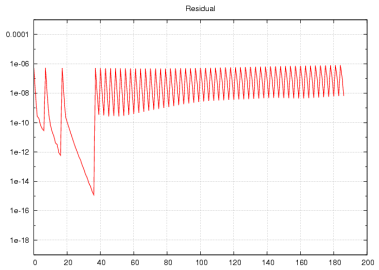

Forces EMUM.10.mod




Residual EMUM.10.mod



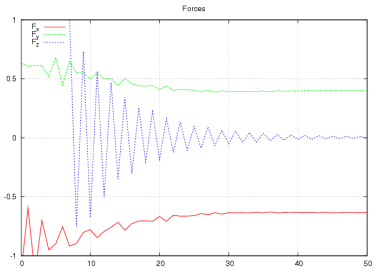

Forces EMUM.05.mod



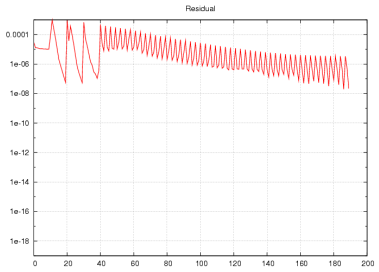

Residual EMUM.05.mod

Figure 12.16: EMUM modified meshes: Calculated forces of the blade and residuals of the flow solution.

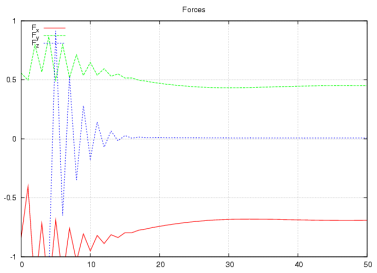
### 12.6.3 CSMM Mesh Simulation Results



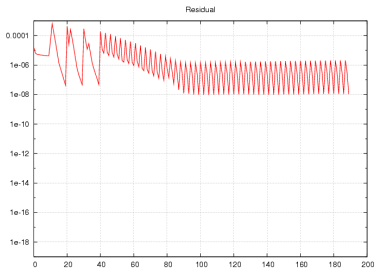

Forces CSMM.40



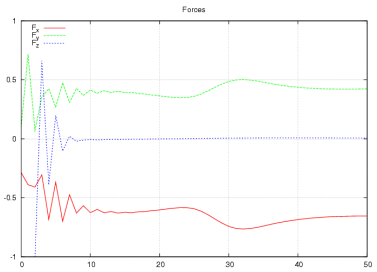

Residual CSMM.40



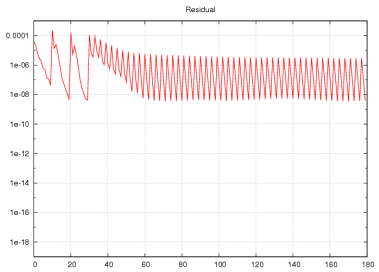

Forces CSMM.20



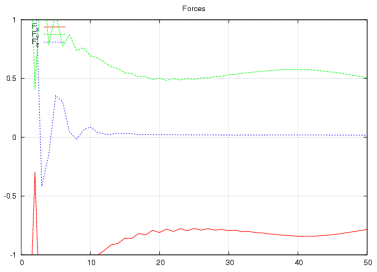

Residual CSMM.20



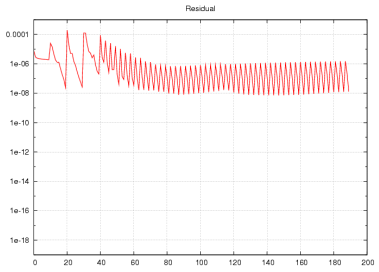

Forces CSMM.10




Residual CSMM.10



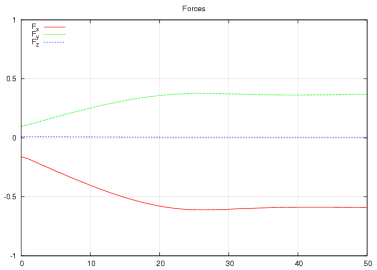

Forces CSMM.05



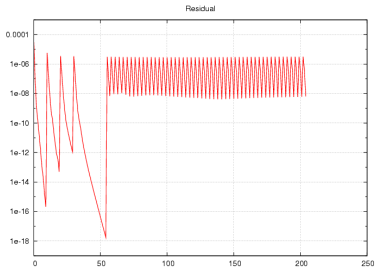

Residual CSMM.05

Figure 12.17: CSMM meshes: Calculated forces of the blade and residuals of the flow solution.

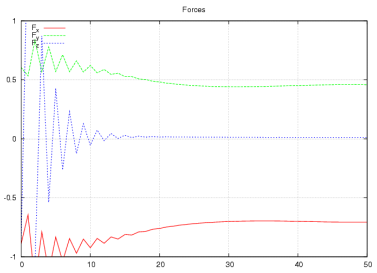
### 12.6.4 CSMM Modified Mesh Simulation Results



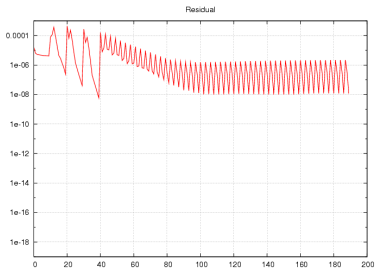

Forces CSMM.40.mod



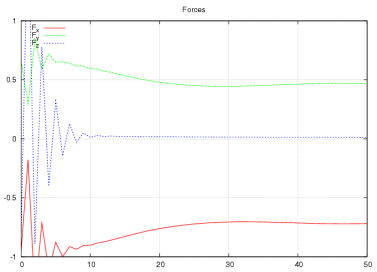

Residual CSMM.40.mod



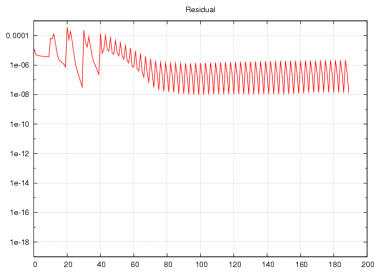

Forces CSMM.20.mod



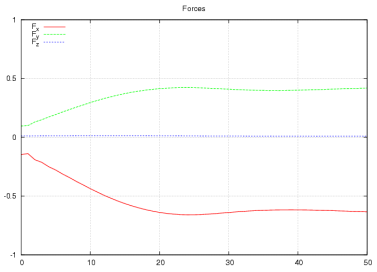

Residual CSMM.20.mod



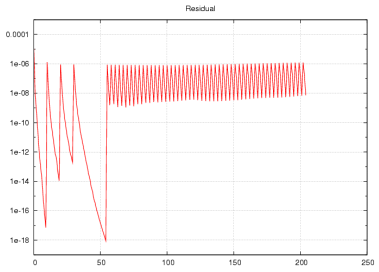

Forces CSMM.10.mod




Residual CSMM.10.mod




Forces CSMM.05.mod




Residual CSMM.05.mod

Figure 12.18: CSMM modified meshes: Calculated forces of the blade and residuals of the flow solution.

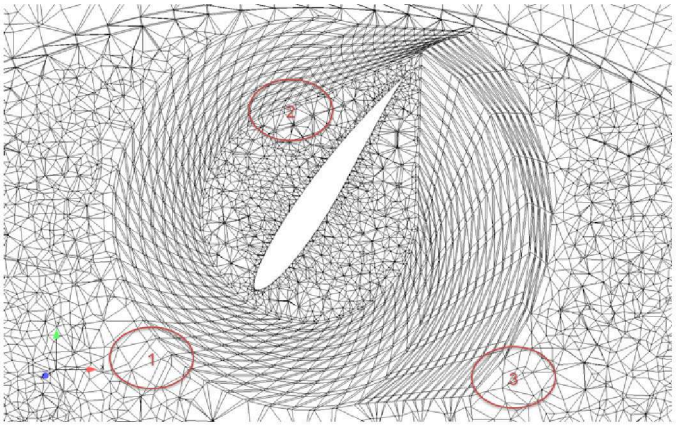
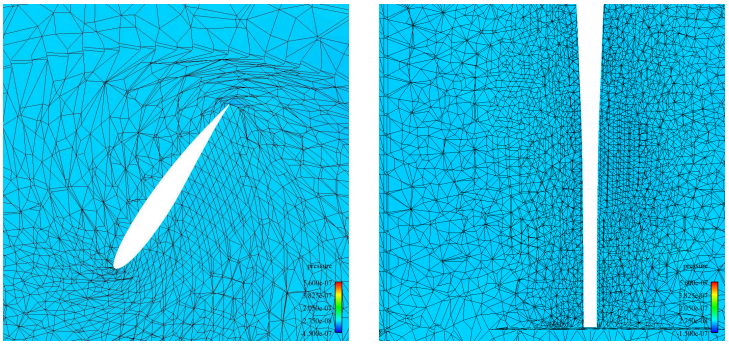
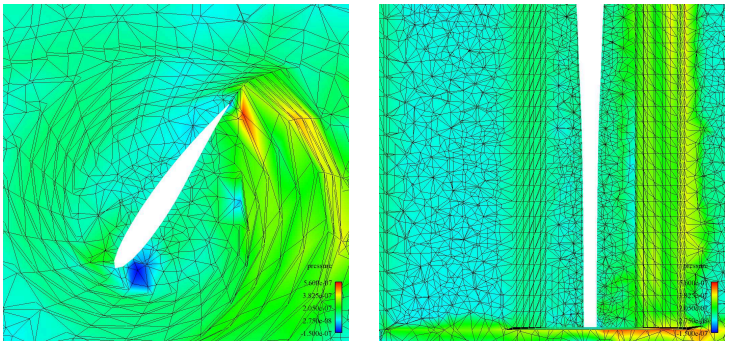



Figure 12.19: Non-conforming zones 1, 2 and 3 of the CSMM approach.

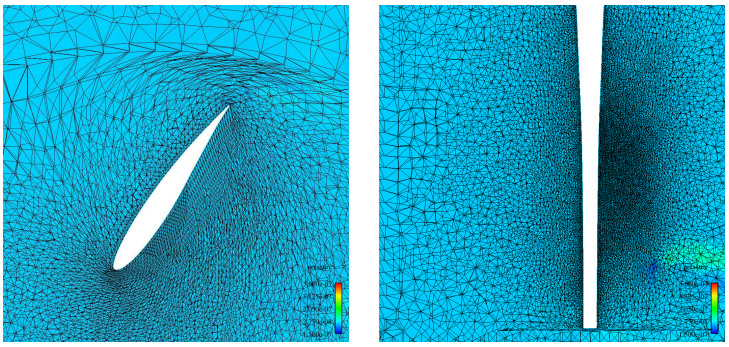
In addition to these zones, the region between the bottom of the blade and the inner SSMUM beaker is a non-conforming one regarding a smooth element size distribution (see Figure 12.11). The direct relationship between the element quality and the local element residuals of a mesh is evident when Figures 12.11, 12.12, 12.8 and 12.7 are compared with Figure 12.20. Studying these figures, the element quality of the EMUM meshes EMUM.40 and EMUM.10 in the region below the gap (figure 12.8) and in a horizontal plane in the midth height of the blade (Figure 12.7), and similarly for the CSMM meshes CSMM.40 and CSMM.10 (Figures 12.11 and 12.12) can be compared to the local element residuals of the pressure values at the 50th time step of the Navier-Stokes simulation (Figure 12.20). In all mesh regions where the CSMM meshes show a worse element quality than the EMUM meshes, the local element residuals of the pressure values are higher. Whereas for the coarser meshes (EMUM.40 and CSMM.40) this is directly evident and showing strongly red and blue coloured elements inside the CSMM meshes, it must be stressed that residuals for the finer CSMM meshes are also about one hundred times higher in the regions of the shells, but the residuals have a highly negative value and are not as visible as if they had both highly positive and negative values as in case of the coarser meshes.



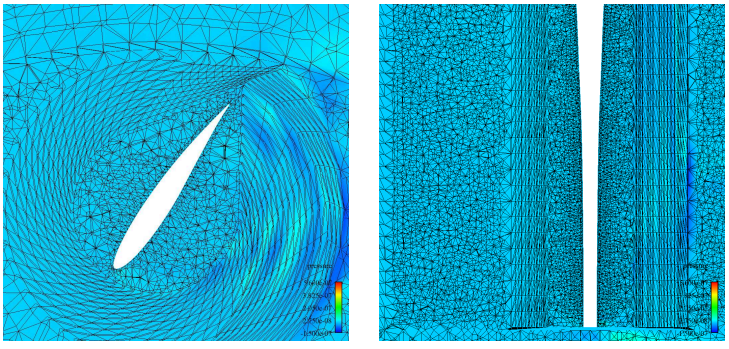
EMUM.40



CSMM.40



EMUM.10



CSMM.10

Figure 12.20: Residuals of the pressure, EMUM.40/10 and CSMM.40/10.

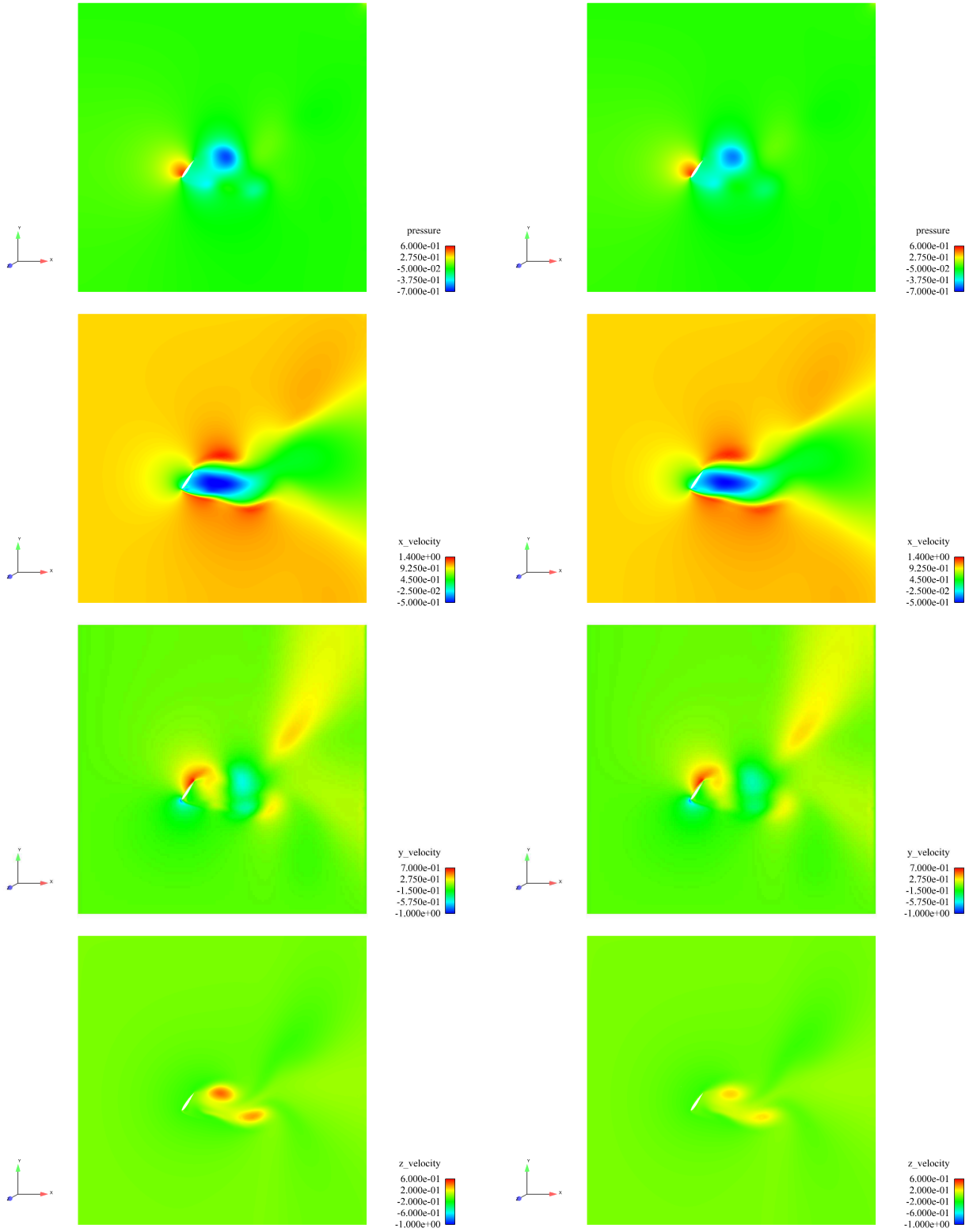
In addition, the resulting flow fields are compared. As the EMUM mesh simulations show a significantly better convergence behaviour, mostly the flow results of these grids are visualized. The flow field at the last time step, e.g., after 5.3 seconds, of the unsteady Navier-Stokes simulation calculated on the two finest EMUM meshes, EMUM.05 and EMUM.10, are compared. The flow pattern is practically the same, see Figure 12.21, except the velocity in the z-direction behind the blade. The z-velocities calculated with the EMUM.05 mesh are about twice as high as those calculated with the EMUM.10 mesh, which indicates that the vortices behind the blade are significantly better resolved by the EMUM.05 mesh. Compared to the simulations of the EMUM.40 mesh in Figure 12.22, these z-velocities are about three times higher, which indicates a successively more accurate calculation of the vortices in the wake of the blade when increasing the mesh resolution.

The remaining flow field calculated via the EMUM.40 mesh also differs considerably from the flow fields of the two finest meshes, taking into account all flow variables, x-, y- and z-velocities and the pressure, which is not visualized here. In comparison with the results of the finest modified EMUM mesh, EMUM.05.mod, the EMUM.05 mesh simulation shows only some minor differences (see Figure 12.23). Considering the visualization of the z-velocity, the small vortex furthest away from the blade is resolved by the EMUM.05 mesh simulation but not from the one performed with the EMUM.05.mod mesh. Also, the minimal and maximal values of the z-velocites are higher than those of the EMUM.05 mesh simulation.

Looking at the flow fields generated via the CSMM meshes, there cannot be found any similarities when comparing the resulting flow fields of the simulations with an increasing mesh resolution (cf. Figure 12.24). The flow pattern resulting from the finest CSMM mesh simulation does not resemble the one from the next coarser CSMM mesh and so forth.

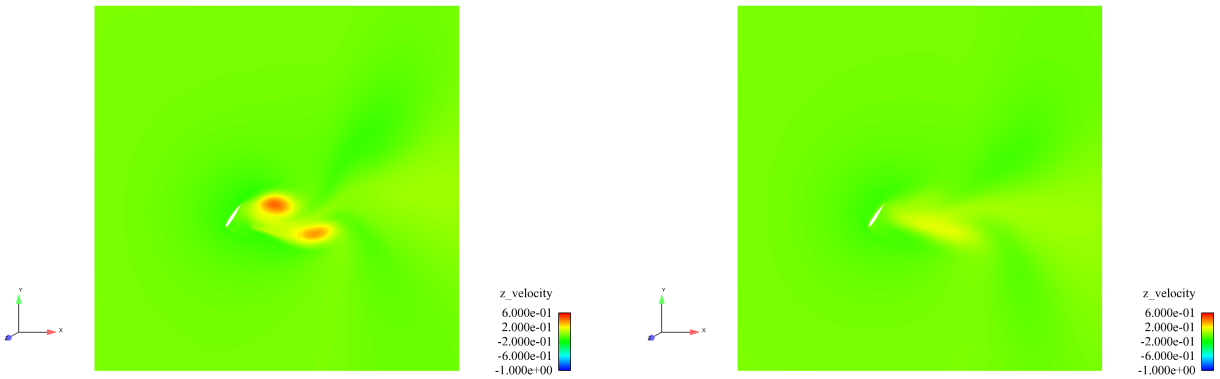
Also the flow field of the finest CSMM mesh does not resemble the one of the finest EMUM mesh (cf. Figure 12.25), e.g., the vortices behind the blade resolved by the EMUM.05 mesh are not resolved by the CSMM.05 mesh.

In addition, the coarsest CSMM mesh CSMM.40 shows an oscillating pressure around the blade over the computed time, which again gives hint that there are stability problems of the CSMM approach due to bad element qualities around the blade (see Figure 12.26).



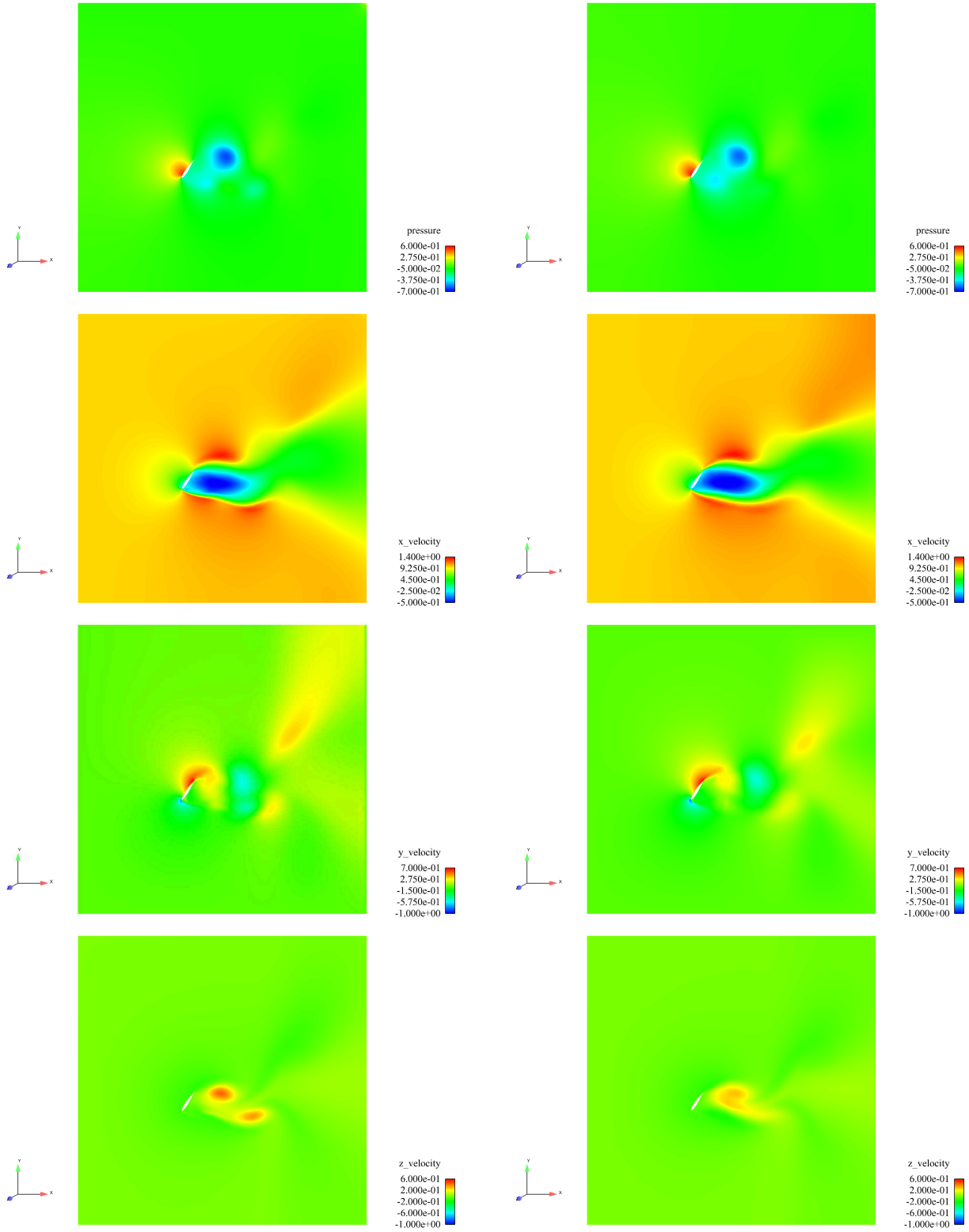

EMUM.05 EMUM.10

Figure 12.21: Flow field after 5.3 sec.




EMUM.05 EMUM.40

Figure 12.22: Comparison of the z-velocity downstream of the blade.




EMUM.05 EMUM.05mod

Figure 12.23: Flow field after 5.3 sec.

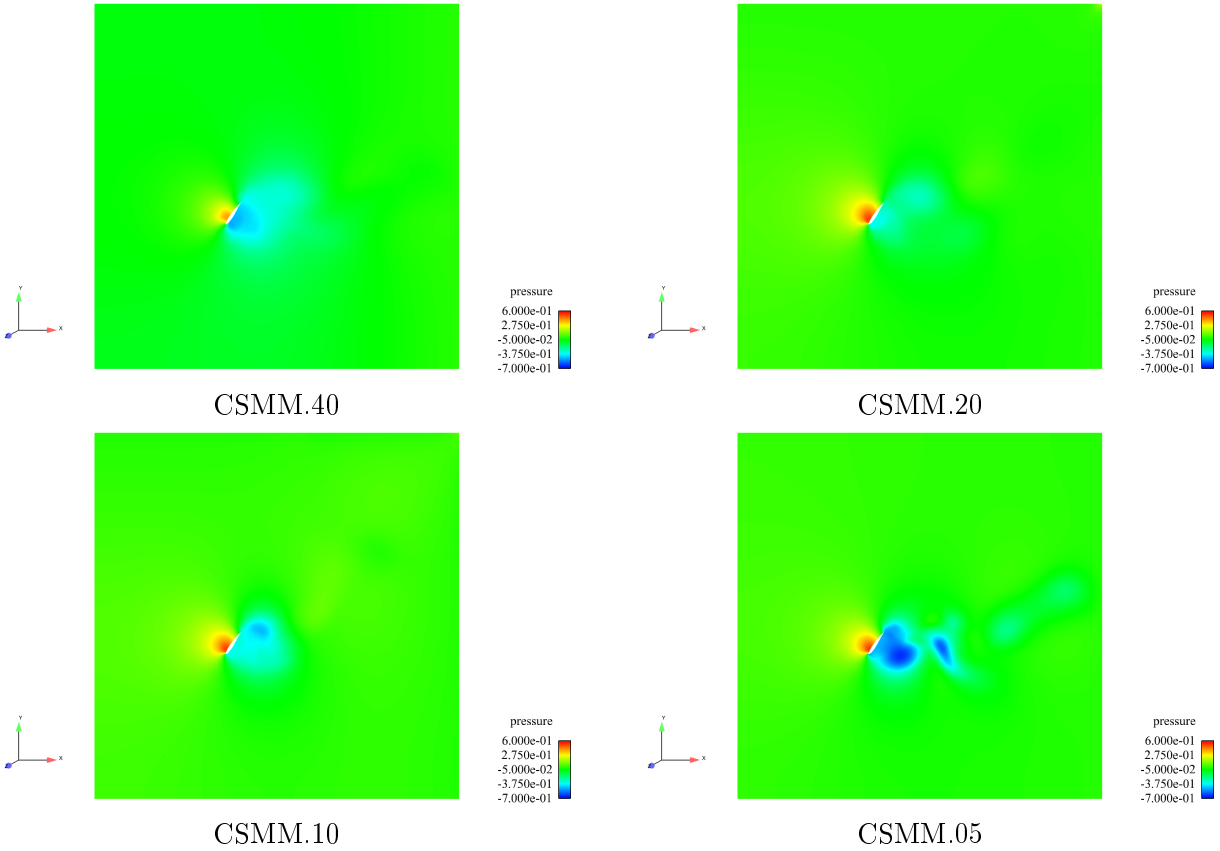



Figure 12.24: CSMM pressure field after 5.3 sec.

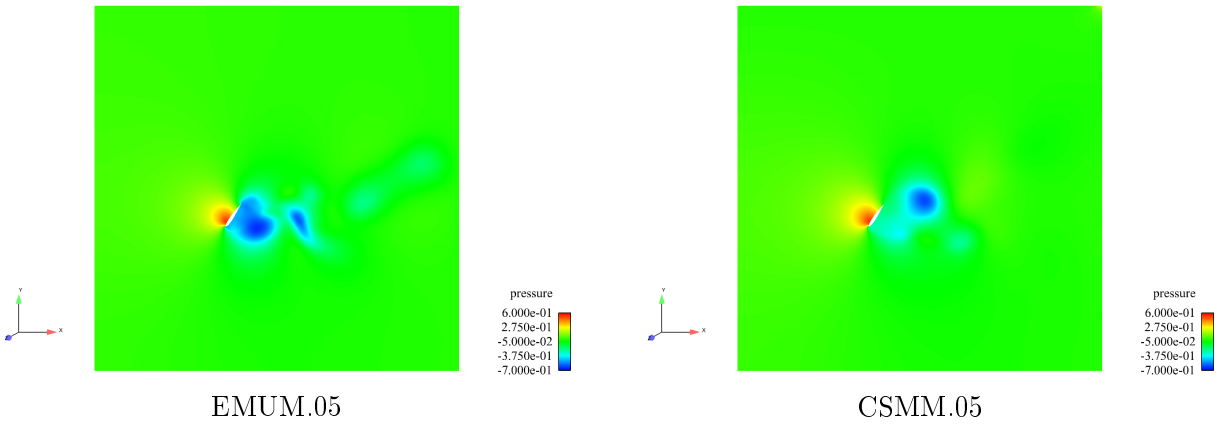



Figure 12.25: Comparison of the pressure field after 5.3 sec.

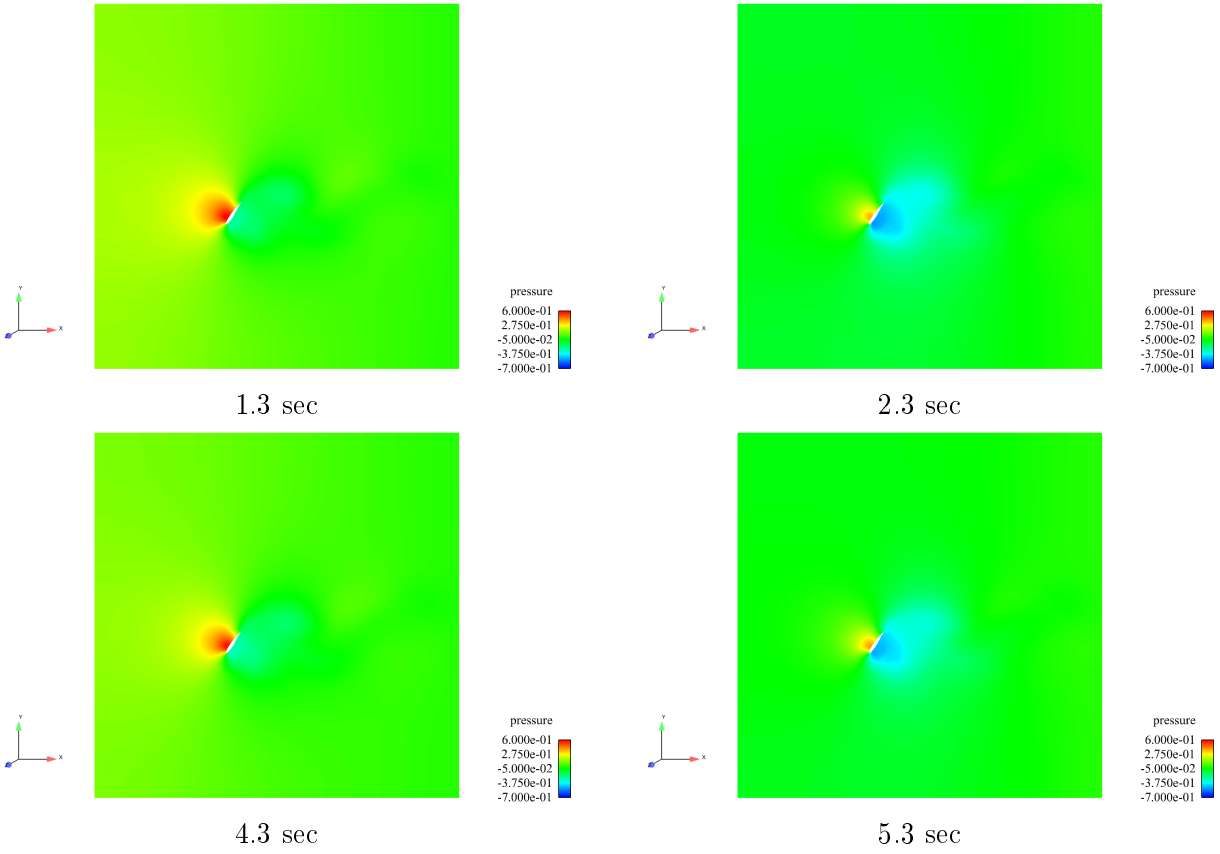



Figure 12.26: CSMM : quasi-periodic pressure field.

## 12.7 Comparison of the Blade Forces

In this chapter, the values of the blade force components calculated via each Navier-Stokes simulation of the fluid flow around the blade are compared in order to determine the relationship between the mesh size and its effect on the resulting fluid flow solution. The viscosity of the simulated fluid is that of water, $\nu = 10^{-6}\frac{\mathrm{m}^2}{\mathrm{s}}$. In Table 12.2, the absolute forces exerted from the blade on the fluid are listed, as well as the deviation of each force component value with respect to the value of the previous coarser mesh. The bold mesh labels indicate the simulations of the CSMM approach with insufficient convergence behaviour.

| **mesh** | $F_x$ | $F_y$ | $F_z$ | $\Delta F_x$ | $\Delta F_y$ | $\Delta F_z$ |
|---|---|---|---|---|---|---|
| EMUM.40 | $-0,549$ | $0,346$ | $-0,0015$ | | | |
| EMUM.20 | $-0,585$ | $0,380$ | $0,0027$ | $6,5\%$ | $9,9\%$ | $-280\%$ |
| EMUM.10 | $-0,598$ | $0,393$ | $0,0028$ | $2,2\%$ | $3,2\%$ | $3,7\%$ |
| EMUM.05 | $-0,611$ | $0,403$ | $0,0030$ | $2,1\%$ | $2,6\%$ | $11\%$ |
| EMUM.40.mod | $-0,596$ | $0,383$ | $0,0051$ | | | |
| EMUM.20.mod | $-0,618$ | $0,405$ | $0,0068$ | $3,6\%$ | $5,7\%$ | $34\%$ |
| EMUM.10.mod | $-0,636$ | $0,421$ | $0,0074$ | $2,8\%$ | $4,0\%$ | $7,9\%$ |
| EMUM.05.mod | $-0,635$ | $0,422$ | $0,0063$ | $-0,16\%$ | $0,23\%$ | $-15\%$ |
| **CSMM.40** | $-0,633$ | $0,398$ | $-0,0045$ | | | |
| **CSMM.20** | $-0,691$ | $0,449$ | $0,0064$ | $6,6\%$ | $12\%$ | $242\%$ |
| **CSMM.10** | $-0,655$ | $0,423$ | $0,0075$ | $-9,2\%$ | $-5,8\%$ | $17\%$ |
| **CSMM.05** | $-0,784$ | $0,508$ | $0,0175$ | $19\%$ | $19\%$ | $135\%$ |
| CSMM.40.mod | $-0,592$ | $0,369$ | $0,0031$ | | | |
| **CSMM.20.mod** | $-0,706$ | $0,458$ | $0,0110$ | $19\%$ | $24\%$ | $259\%$ |
| **CSMM.10.mod** | $-0,719$ | $0,466$ | $0,0120$ | $1,8\%$ | $1,8\%$ | $9,0\%$ |
| CSMM.05.mod | $-0,634$ | $0,419$ | $0,0088$ | $-11\%$ | $-10\%$ | $-27\%$ |

Table 12.2: Forces from the blade on the fluid after 5.3 sec.

Considering the EMUM meshes, a decreasing deviation of the force components can be observed when the mesh refinement is increased. This holds for the normal meshes as well as for the modified ones. The simulations on the CSMM meshes, however, do not show a decreasing deviation but even an increasing one. Therefore, the CSMM meshes cannot be used to determine the proper mesh resolution. Taking the results of the EMUM simulations into account, already coarser meshes seem to solve the Navier-Stokes problem quite well (see line 2 and 3 of Table 12.2). To see this, the deviations of the values achieved with the simulations on the coarser meshes EMUM.40 and EMUM.20 are compared to the values achieved with the simulation on the finest mesh EMUM.05 in line 2 and 3 of Table 12.3. Comparing the EMUM meshes with the modified EMUM meshes, the deviations of the force values comparing a coarse mesh with the next finer mesh are less with respect to the modified EMUM meshes. Even the deviation comparing

the resulting force values of the EMUM.40.mod mesh with the EMUM.05.mod mesh are smaller than the deviations comparing the results of the EMUM.40 mesh with the results of the EMUM.20 mesh (see line 4 of Table 12.3 and line 3 of Table 12.2). In the cases where the CSMM meshes show a satisfying convergence, the CSMM.40.mod and CSMM.05.mod meshes, the results gained with these meshes deviate only very little from those gained with the EMUM counterpart meshes (see line 5 and 6 of Table 12.3).

| **mesh** | $\Delta F_x$ | $\lvert\Delta F_x\rvert$ | $\Delta F_y$ | $\lvert\Delta F_y\rvert$ | $\Delta F_z$ | $\lvert\Delta F_z\rvert$ |
|---|---|---|---|---|---|---|
| EMUM.05 - EMUM.40 | 10% | 0,062 | 14% | 0,057 | 50% | 0,0015 |
| EMUM.05 - EMUM.20 | 4,4% | 0,026 | 5,5% | 0,023 | 10% | 0,0003 |
| EMUM.05.mod - EMUM.40.mod | 6,1% | 0,039 | 9,2% | 0,039 | 18% | 0,0011 |
| EMUM.40.mod - CSMM.40.mod | −0,6% | −0,004 | 3,6% | 0,014 | 39% | 0,002 |
| EMUM.05.mod - CSMM.05.mod | 0,2% | 0,001 | 2,6% | 0,011 | 42% | 0,0026 |

Table 12.3: Comparison of the force value deviations among selected EMUM meshes.

The influence of the squeezed elements below the blade on the resulting values of the force components can be judged looking at Table 12.4, where the results of the modified EMUM meshes are compared against the results achieved with the normal EMUM meshes.

| **mesh** | $\Delta F_x$ | $\lvert\Delta F_x\rvert$ | $\Delta F_y$ | $\lvert\Delta F_y\rvert$ | $\Delta F_z$ | $\lvert\Delta F_z\rvert$ |
|---|---|---|---|---|---|---|
| EMUM.40.mod - EMUM.40 | 7,8% | 0,047 | 11% | 0,037 | 129% | 0,0066 |
| EMUM.20.mod - EMUM.20 | 5,3% | 0,033 | 6,1% | 0,025 | 57% | 0,0039 |
| EMUM.10.mod - EMUM.10 | 5,9% | 0,038 | 6,6% | 0,028 | 62% | 0,0045 |
| EMUM.05.mod - EMUM.05 | 3,8% | 0,024 | 4,5% | 0,019 | 52% | 0,0032 |

Table 12.4: Deviations of the force values comparing EMUM and modified EMUM meshes.

It can be concluded that the influence of the bad element quality lessens with an increasing resolution in the critical region between the bottom of the blade and the inner SSMUM beaker. Considering the results of the simulations with the modified EMUM meshes, it can be assumed that the flow solution gained on the EMUM.05.mod mesh is the closest to the real fluid flow, as the element quality is the best inside the modified meshes. To have a glimpse on the deviations from the force values taken from that best simulation, Table 12.5 is presented where all results gained from the non-modified EMUM meshes are compared with the results of the EMUM.05.mod mesh.

Overall, the results of the EMUM approach are much more convincing, as the blade forces really converge with an increasing mesh resolution around the blade. In addition, the influence of the squeezed elements diminishes with a higher refinement. For the CSMM concept, this is neither the case for normal nor the modified meshes. To conclude, the EMUM approach can be regarded as more suitable for the ten blade VWT fluid flow simulation.

| **Mesh** | $\Delta F_x$ | $\lvert\Delta F_x\rvert$ | $\Delta F_y$ | $\lvert\Delta F_y\rvert$ | $\Delta F_z$ | $\lvert\Delta F_z\rvert$ |
|---|---|---|---|---|---|---|
| EMUM.05.mod - EMUM.40 | 14% | 0,086 | 18% | 0,076 | 124% | 0,0077 |
| EMUM.05.mod - EMUM.20 | 7,8% | 0,050 | 10% | 0,042 | 56% | 0,0035 |
| EMUM.05.mod - EMUM.10 | 5,8% | 0,037 | 6,9% | 0,029 | 55% | 0,0034 |
| EMUM.05.mod - EMUM.05 | 3,8% | 0,024 | 4,5% | 0,019 | 52% | 0,0032 |

Table 12.5: Deviations of the force values compared to the results of mesh EMUM.05.mod.

# 13 Simulation of the VWT Propulsion System

According to the results of the last chapter, the EMUM approach is chosen for the fluid simulation of the operating point "open water speed", considering the complete ten blade propulsion system of the Voith Water Tractor including the nozzle plate. The speed of the entire VWT at that operating point is 7.2 m/sec, and the two Voith-Schneider propellers counterrotate at a speed of 80.2 rotations per minute. The corresponding blade angle curve for each of the ten blades was shown in Figure 5.1. The boundary conditions are the same as those used in the single blade refinement study, except for the inflow and the outflow; these are swapped and the inflow x-velocity is -7.2 m/sec. Considering this simulation, the water streams then from the upstream to the downstream side of the domain. The VWT propulsion system is placed closer to the upstream side (inflow) (see figure 13.4). By that, the simulation domain downstream of the VWT is enlarged in order to better resolve the wake behind the VWT. Worth mentioning is that the velocity of each single surface node of the blades must be calculated in order to prescribe a no-slip boundary condition on the blade surfaces. Also for the ten blade VWT simulation a refinement study is performed. The range of the mesh resolution on the blade surfaces from the coarsest resolution of 40 mm to the finest resolution of 5 mm seems to be suitable, as the force components in the x- and y-direction differ only less than three percent comparing the results of the two finest meshes of the single blade refinement study. All the mesh parameters used for the mesh generation of the single blade meshes are kept (cf. Section 12.3 and 12.4). The mesh in the closer region of the blades is shown for all the four meshes in Figure 13.1.

Regarding the simulations on these meshes, a similar set of intermediate simulations, starting with the Stokes flow simulation and a continuation with decreasing kinematic viscosities, is not successful, as finding a proper set of solver parameters for a converging flow simulation for each intermediate simulation has shown to be too much effort. However, as we observed convergence of the fluid flow simulation in an earlier simulation, where only the SSMUM approach was applied to the VWT propulsion system, we used that simulation setup including the same solver parameters for the simulation including the SSMUM and EMUM approach. Eventually, that direct Navier-Stokes simulation with the water viscosity $\nu = 10^{-6}\frac{\mathrm{m}^2}{\mathrm{s}}$ using a Krylov space size of 200 and ten nonlinear iterations within one time step, converges after about 130 time steps. The time step size is 0.002078 seconds and by that each propeller rotates by one degree within one time step. This holds for all the four different mesh sizes. Figure 13.2 shows the residual chart of each single nonlinear iteration during the simulation on the finest mesh. Here, the convergence behaviour improves significantly after seventy time steps, which corresponds

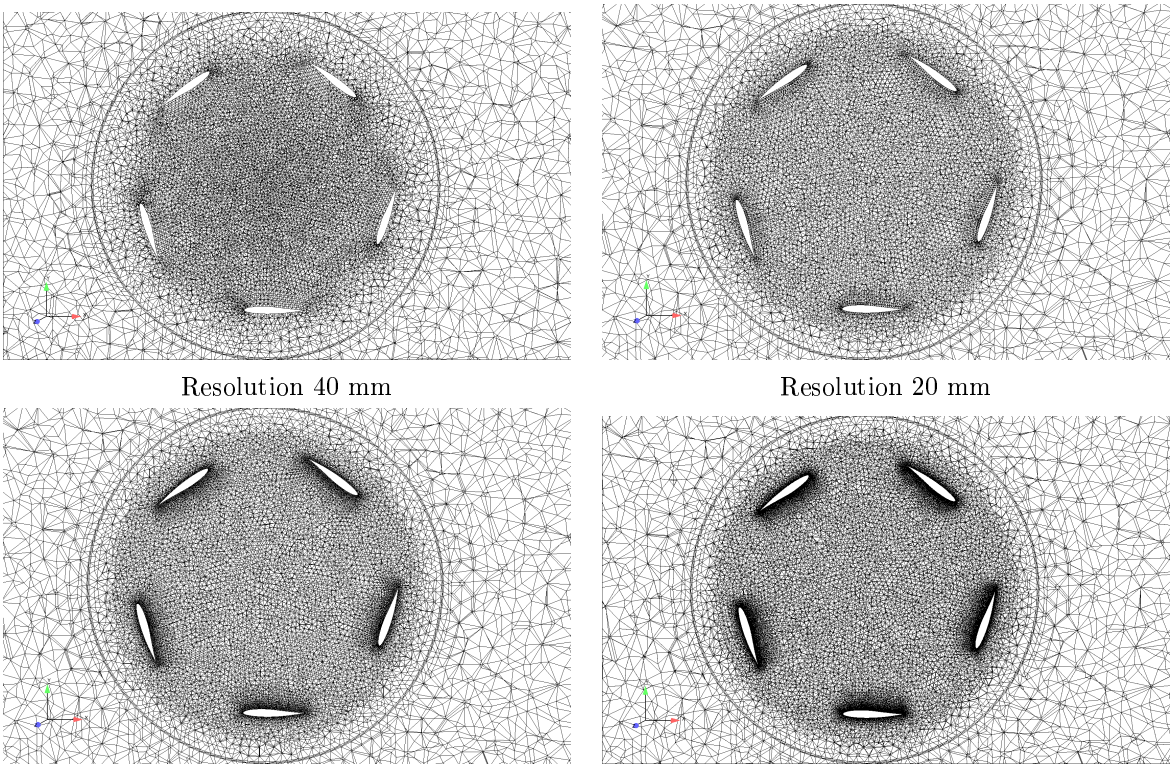



Figure 13.1: Four different resolutions of the 10 blade mesh.

to nonlinear iteration number seven hundred as ten nonlinear iterations are performed per time step.

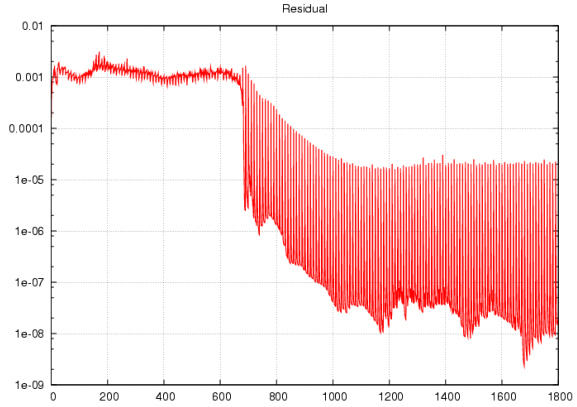



Figure 13.2: Residual over the nonlinear iterations.

On all four meshes, two full rotations where simulated and after one rotation already a

periodic behaviour can be observed, as the values of the force exerted from the blades on the fluid already show a periodic behaviour after one rotation; cf. Figure 13.3 where the graph of the force of one single blade exerted on the fluid is shown from time step 130, at which the solution on all four meshes converges. The time step number 130 corresponds to a 130 degree rotation. The position of one blade at zero degree is considered as the position of the blade closest to the symmetry line on the left picture of Figure 5.3.

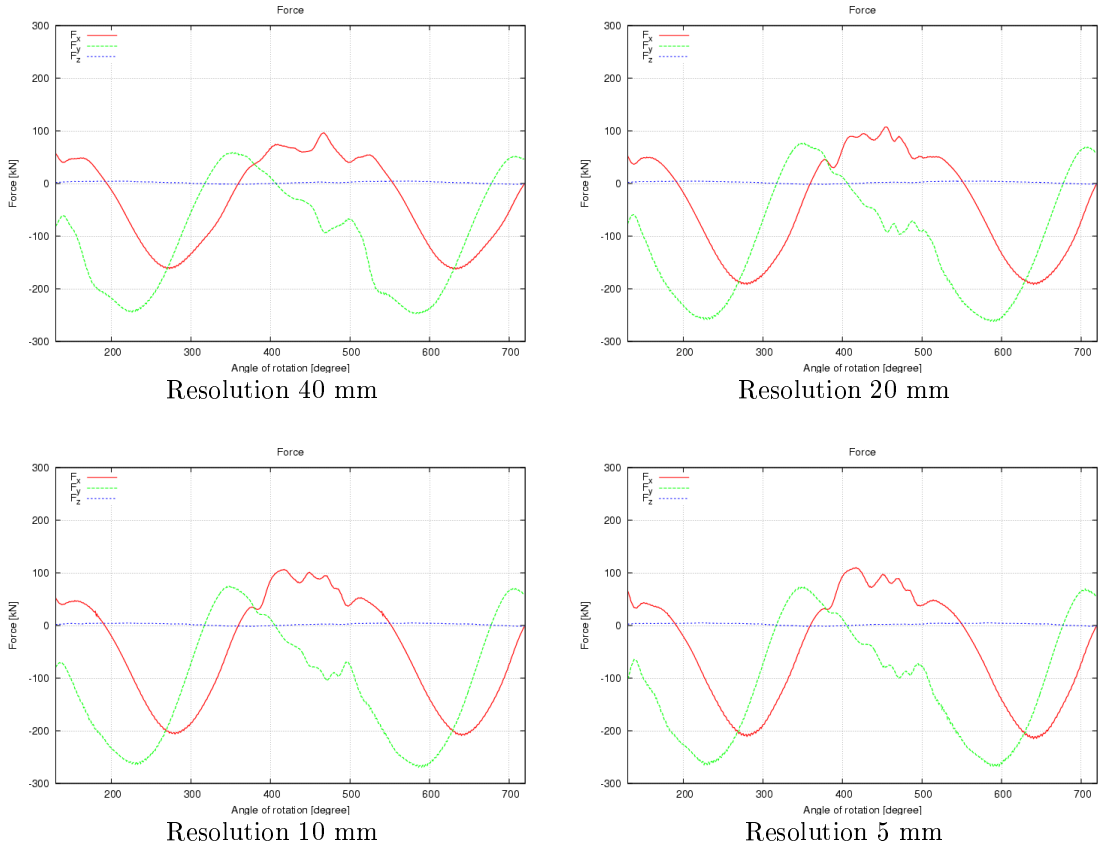



Figure 13.3: Graph of the force components of a single blade.

For further investigation of Figure 13.3, the characteristics of the x-component of the force exerted from the blade on the fluid between the 400 degrees and the 540 degree position of the blade will be examined. In the course of that part of the propeller rotation, a strong deviation of the force characteristics is visible comparing the three coarsest meshes to each other. On the contrary, the two finest grids hardly show any deviation considering the chart of the force values. Nevertheless, the two finest meshes show some minor differences regarding the calculated flow field after two full propeller rotations. The pressure field at the low pressure sides of the blades differ slightly looking at a horizontal cut of the computational domain (cf. Figure 13.4), and looking at the vertical cut some differences in the x-velocity values at the near blade surface can be seen.

A further refined mesh simulation was not possible due to limited computing resources. The most important figure resulting from the simulation of the VWT propulsion system is the averaged thrust generated by a single blade of a VSP during one full rotation of a VWT propeller. This value is comparable with the values measured by Voith in water tank experiments, real experiments and their simulation with the CFD software COMET. In Table 13.1, the calculated averaged forces of each single blade for all the four different meshes are presented. The units of these forces is kN. It can be seen that the resembling force values of the two finest meshes within each time step result in a similar averaged thrust generated by the single blades during one full propeller rotation (see Table 13.1).

| Blade / Mesh resolution | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 40 mm | 16 | 17 | 16 | 17 | 17 | 16 | 17 | 16 | 16 | 16 |
| 20 mm | 22 | 22 | 21 | 22 | 21 | 21 | 23 | 22 | 22 | 21 |
| 10 mm | 25 | 24 | 25 | 25 | 24 | 24 | 24 | 25 | 25 | 24 |
| 5 mm | 27 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 |

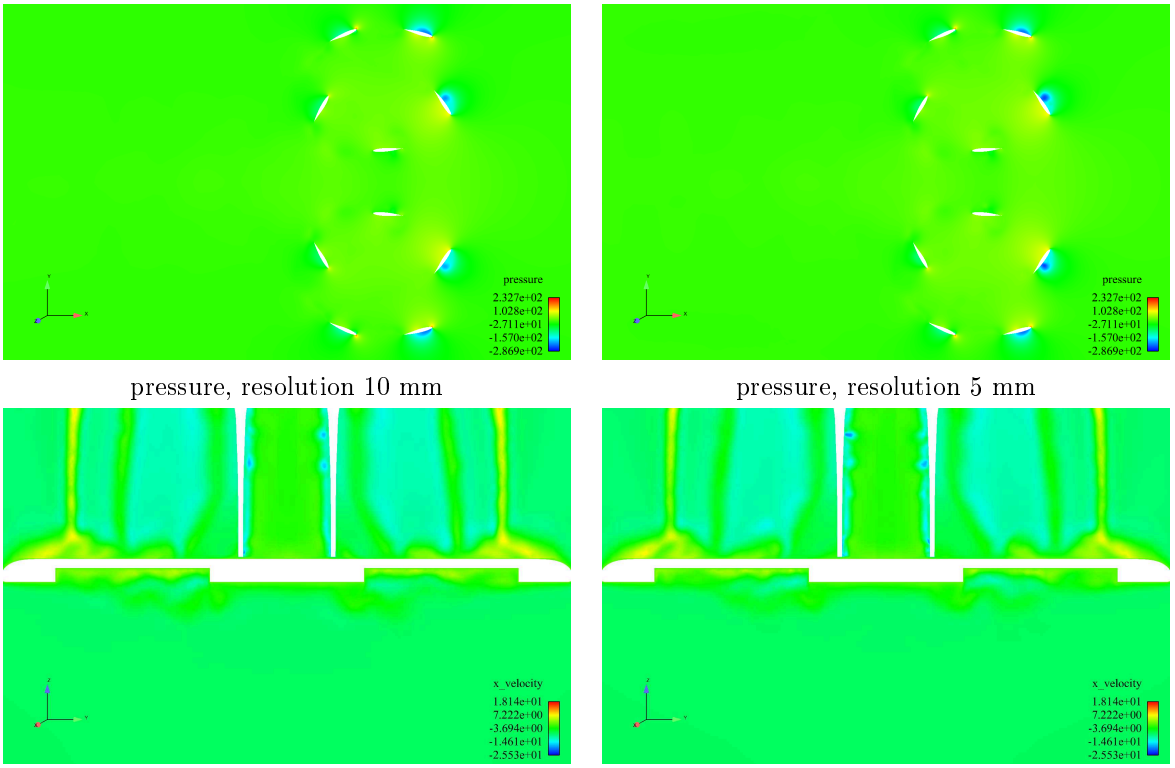
Table 13.1: Averaged forces of each single blade for the four different mesh resolutions.

The measured averaged thrust of one single blade measured by Voith is 23 kN, which corresponds to a deviation of about 12 per cent compared to the thrust values simulated with the finest mesh. This deviation we regard as a good result, but according to Voith the characteristics of the force values differ considerably compared to the measured ones and the curves obtained by the COMET fluid flow simulations done by Voith.

Precisely, the blades generate too much thrust in the negative x-direction in the positions where they move against the motion of the ship to be propelled. In Figure 13.3, these positions would correspond to the region from a 360 degree position to a 540 degree position of a single blade. Referring to Figure 13.4, it was suggested that the pressure distributions around the blades at the 72 and the 144 degrees position, which corresponds to the 432 degree and the 504 degree position in Figure 13.3, show that the blades at these positions would generate a total force on the ship in the negative x-direction. However, according to the experiments and flow simulations at Voith, the blades generate at these positions a positive total force on the ship. In Figure 13.3, it can be seen that the blades generate a positive force on the fluid in the positions between the 72 and the 144 degree position. This corresponds to a thrust contribution in the negative x-direction by the blades moving from the 72 to the 144 degree position or from the 432 to the 504 degree position respectively.

Obviously, the direct flow around the blades is not simulated like it is with the CFD flow solver used at Voith and how the experimental data gained by Voith suggests. One main effect is assumed to be responsible for this deviation.

Different to commercial CFD codes like COMET or ANSYS, the flow solver XNS does not yet account for the modelling of a boundary layer close to walls where a noslip boundary condition has to be set due to the wall friction, as on a propeller blade. Com-




pressure, resolution 10 mm

pressure, resolution 5 mm

x-velocity, resolution 10 mm

x-velocity, resolution 5 mm

Figure 13.4: Pressure and x-velocity distribution of the propulsion system.

mercial CFD codes do this in using the universal wall log law. The universal wall log law stems from the boundary layer theory and concepts of Ludwig Prandtl introduced in 1904, which are regarded as a milestone in the history of fluid dynamics [Sch04]. His concepts were followed up by Karman and Blasius modelling the velocity distribution in the turbulent boundary layer. Given the resulting formulas of their work, it is possible to approximate the thickness of the turbulent boundary layer, the distance from the wall where the turbulent layer joins the viscous sublayer and the shear stress in the turbulent layer. Within commercial CFD codes, the universal wall log law is used to place the closest grid node of the mesh next to the wall still inside the logarithmic wall log law region of the turbulent boundary layer. The codes then determine the wall shear stress at this position according to the universal wall log law, which is then imposed as boundary condition at the first grid points next to the wall. By that the difficult resolution of the near-wall turbulent flow is avoided [OK95]. The boundary layer concepts developed by Prandtl, Karman and Blasius use various dimensionless variables to determine the thickness of the turbulent boundary layer and of the viscous sublayer. These variables will be employed in the rest of this section but will not be explained fully. A detailed derivation of the universal wall log law and its use to determine the velocity distribution in the turbulent layer and its thickness can be reviewed in [Sch04]. In the following, it will be

shown that it is impossible to resolve the turbulent boundary layer by fine meshes around the blade of the VWT propulsion system given the computational resources. Using the resulting formulas of the boundary layer concepts presented in [Sch04] it will be shown that the grid spacing next to the blade surface would be too small, such that the meshes with ten blades would become too big if the turbulent boundary layer was resolved by the fluid flow simulation. As the first grid point would need to be placed at the border of the viscous sublayer to the turbulent boundary layer, the following is set:

$$\frac{\bar{u}}{u_*} = y^+ = 1, \tag{13.1}$$

with $\bar{u}$ being the velocity in the viscous sublayer and $u_*$ the friction velocity inside the viscous sublayer, defined as

$$u_* = \sqrt{\frac{\tau_w}{\rho}}, \tag{13.2}$$

where $\tau_w$ is defined as the shear stress at the wall, which is assumed to be equal to the shear stress in the viscous sublayer, as this layer is very thin [Sch04]. The $y^+$ is the dimensionless distance to the wall defined by:

$$y^+ = \frac{y u_*}{\nu}, \tag{13.3}$$

with $y$ being the distance to the wall whereas $y$ is equal to one at the position where the viscous sublayer adjoins the turbulent boundary layer. Given these formulas and the derived formulas in [Sch04] for the friction coefficient $c_f$ as well as the formula for the wall shear stress, the distance $y$ between the wall and the position where the viscous sublayer adjoins the turbulent boundary layer can be calculated. At this distance to the wall $y$, the first grid point must be positioned if the fluid dynamics effects inside the turbulent boundary layer have to be resolved without using the universal wall log law. The wall shear stress is related to the friction coefficient $c_f$ :

$$c_f = \frac{\tau_w}{\frac{1}{2}\varrho U_\infty^2}, \tag{13.4}$$

where $U_\infty$ is the free stream velocity outside the boundary layer. Using the boundary layer theory given in [Sch04], $c_f$ can be expressed as:

$$c_f = \frac{0.072}{(Re_L)^{\frac{1}{5}}}. \tag{13.5}$$

Setting Equation (13.4) equal to (13.5) yields the following term for the wall shear stress:

$$\tau_w = \frac{0.072}{(Re_L)^{\frac{1}{5}}} \frac{1}{2} \varrho U_\infty^2. \tag{13.6}$$

Replacing the friction velocity with the wall shear stress and the density according to Equation (13.2) and solving Equation (13.3) for $y$ with $y^+ = 1$ yields:

$$y = \sqrt{\frac{\rho\nu^2}{\tau_w}} = \sqrt{\frac{2(Re_L)^{\frac{1}{5}}\nu^2}{0.072U_\infty^2}} \approx 10^{-6}m, \tag{13.7}$$

considering a blade velocity of about $10\frac{m}{sec}$ at a rotation speed of 80.2 rotations per minute and a blade length of $0.5m$. That calculated node distance $y$ would correspond to a mesh resolution five thousand times finer than the resolution of the finest mesh used for the simulation of the whole VWT propulsion system so far, and the mesh size of that mesh is already about ten million elements. Given the computing resources of today, it is evident that an adequate turbulence model must be employed, which proves to be beyond the scope of this thesis.

# 14 Ship Hull Approximation with Neural Networks

So far, the described mesh moving algorithms allow simulating a flow around the VWT propulsion system. As the XNS flow solver has also successfully been used to simulate 3D free-surface flows of a trapezoidal channel [BA02], a spillway of a dam of the Ohio River [Beh00, Beh01], and around a circular cylinder [GBT99], the goal is to extend the free-surface capabilities of XNS so that a free-surface motion along the hull of the VWT, which is shown in Figure 14.1, can be simulated. Conversely to the CSMM, EMUM and

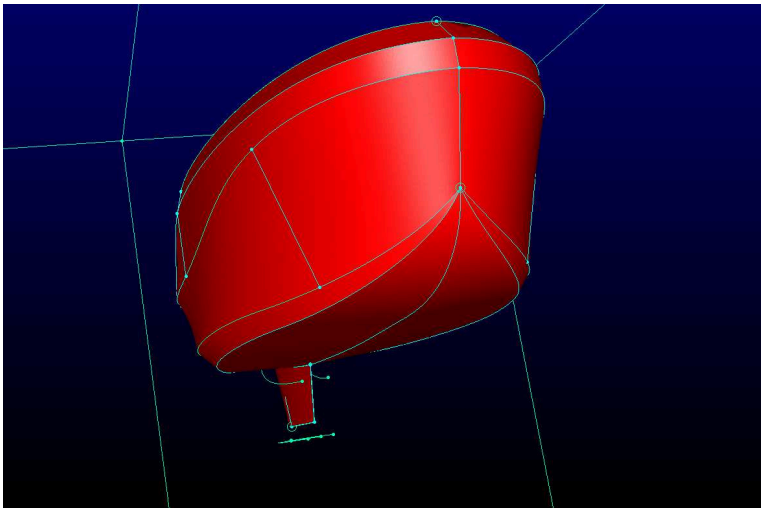


Figure 14.1: Hull of the Voith Water Tractor.

SSMUM methods, where the mesh nodes on the moving boundaries do not move relative to the boundary, a free-surface finite element fluid flow simulation of a ship hull needs to apply a mesh moving technique where mesh nodes can move arbitrarily along the ship hull according to the calculated wave touching the ship hull. As a consequence, to follow exactly the air water interface, the surface geometry where the free surface moves along must be known exactly in terms of a function. Regarding the implementation of XNS, this function has to be coded inside the programme. So far, the functions for the tracking of a free-surface motion along straight lines or planes like the boundaries of a trapezoidal channel and along semi-cylindrical shapes are implemented. The representation of the shape of a ship hull gets very complicated if a one to one implementation of the CAD splines is carried out. This chapter presents an easy and rather unconventional approach to enable a finite element flow simulation around a ship hull. Generally, the surface

function originating from the design process can be represented by an IGES file, NURBS patches or in many other ways. Assuming an internal different geometry representation within a finite element flow solver, a conversion of the given design format of the ship geometry to the customized format of the flow solver must be made. However, there are finite element fluid solver approaches based on NURBS shape functions where such a conversion is obsolete (see [BH08]). In the cases where a conversion is a necessity, the following approach should give a good example how to deal with a conversion of an IGES ship hull format to a simpler geometry representation. In the context of a ship hull simulation, the first key point of this approach is that most of the ship hull part shown in Figure 14.1 can be described with a convex function, where one pair of x and y values matches only one z value, especially in the vicinity of the waterline. Therefore, a direct mapping from the x- and y-coordinates to the z-coordinate is possible. This function is called hull function in the following. With such a given hull function, the motion of the waves along the ship hull can be easily calculated. In order to explain how a wave motion can be tracked, only a section of a very simple ship hull shape is considered, compare the blue body in Figure 14.2. Within this chosen section, the mapping from the x- and y-coordinates to the z-coordinate can be done with one single convex function, especially in the zone close to the waterline, which is indicated with red points in Figure 14.2.

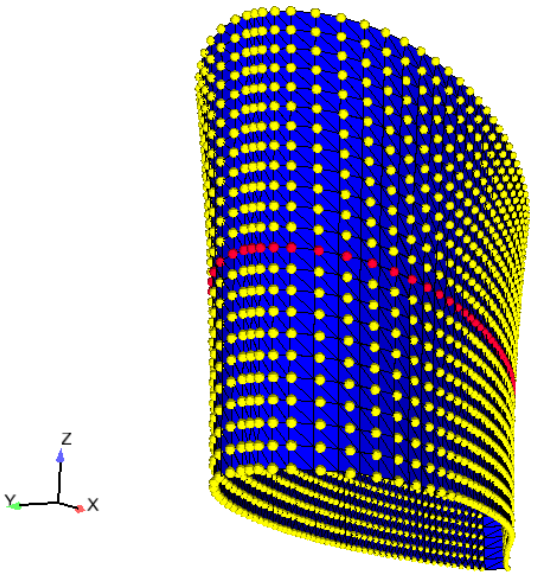



Figure 14.2: Section of a simple ship hull shape.

In the zone close to the waterline, for each node on the water surface attached to the ship hull a given $\Delta z$ is computed by the elevation equation and can then be transferred to a $\Delta x$ and $\Delta y$ using the normal vector on the ship hull. The normal vector can be calculated via the hull function at any position on the hull. A second key point of our approach is the generation of the hull function based on the IGES file. Here the IGES data is used to create a surface mesh in GRIDGEN, so that an arbitrarily large set of surface mesh

nodes and their coordinates can be exported. This data set of mesh node coordinates is then used to fit the parameters of a neural network based hull function. Generally, any type of function can be used to fit the mesh node data set, e.g., polynomials. However, polynomials are known for oscillations in the region between and further away from the given data points. Neural networks, however, are known for their smooth representation of high-dimensional functions and for their compact representation of data, which is important when the limitation of computer memory resources is considered. Therefore, neural networks are applied for the generation of the hull function.

Though neural networks are widely used in all fields of technical applications as well as in the ship design process, the method of fitting the geometry data of a ship hull is not known to the author. A very common application of neural networks, however, is the fitting of CFD simulation input data to the calculated CFD output data. For example, a flow around several ship hull designs, which are parametrized, is simulated, and the corresponding drag of the hull designs are calculated by conventional CFD software. The ship hull parameter values serve as input and the calculated drag as output data for the fitting of the neural network. Based on this specific use of neural networks, an optimization framework where a neural network replaces an extensive CFD computer model is described in [BSHS07]. In [ZHS02], neural networks are used to predict the damage in a ship structure, and in [Kou03], to predict the wash and ship resistance.

The concept of neural networks can be regarded in two ways. From the mathematical point of view, an assembled function consisting of sums and single subfunctions like hyperbolic tangents is fitted to a given data set by varying the parameters of the subfunctions and the weights of the single sums. From the biological point of view, artificial neural networks (ANN) can be considered as a simple copy from what is known on how the human brain might work. It is believed that the human brain is powerful because of its massive parallelism combined with a complex connectivity on the one hand and the simple operations of single neurons on the other hand [Zel94].

Within an ANN, the simple single operation of a neuron is translated from the biological archetype, where chemical processes trigger an electrical pulse, to very simple mathematical operations like a hyperbolic tangents, a multiplication and a summation. The high connectivity among the neurons is modelled by connecting the mathematical output of an artificial neuron to the inputs of many other neurons. There are many different types of ANNs, but in this chapter only the architecture and algorithm of the most commonly used [Zel94] is presented; a feed-forward multi-layer perceptron (MLP) neural network. Applied to the data-fitting of the hull function, the MLP in this case has two input neurons corresponding to the x- and y-values of one mesh node and one output neuron corresponding to the z-value of the mesh node. In between, there are many layers of neurons, which have direct connection to the neurons of the next layer but not to the further ones (cf. Figure 14.3).

How an input value passes through the neural network and creates an output value is explained in the following. As the calculation of the output value, the z-coordinate in this example, requires some vector-matrix multiplications, firstly the sizes of these vectors and matrices are given in Tables 14.1 and 14.2 below:

At first, the input values x and y are passed each separately to two input neurons, the x-

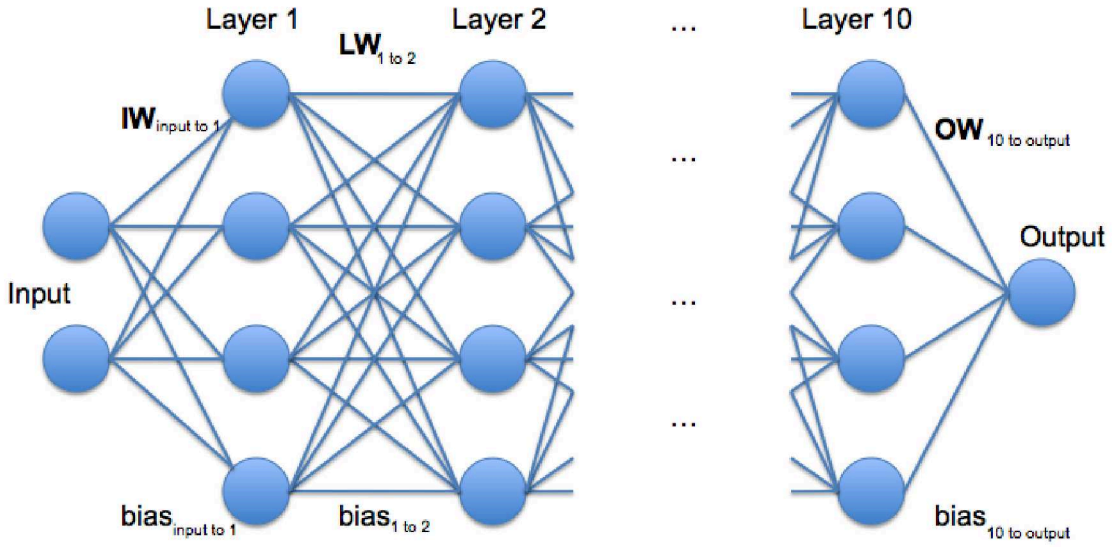



Figure 14.3: Architecture of a multi-layer perceptron neural network.

value to the upper one and the y-value to the lower one (see Figure 14.3). In a forthcoming matrix-vector operation and summation, the input values $\underline{temp}$ are calculated for the neurons of the first hidden layer, Layer 1 in Figure 14.3. The vector $\underline{bias}_{input\to 1}$ indicates the bias vector. The bias vector $\underline{bias}_{input\to 1}$ includes the single bias that has to be added to the matrix-vector operation in order to pass the $\underline{temp}$ values from the input layer to the first hidden layer:

$$\underline{temp} = \underline{bias}_{\text{input}\to 1} + \mathbf{IW} \cdot \begin{pmatrix} x \\ y \end{pmatrix}, \tag{14.1}$$

$$OutL_{1,i} = \tanh\left(temp_i\right). \tag{14.2}$$

Then, in each neuron $i$ of the first hidden layer, the hyperbolic tangent of the single scalar value $temp_i$ is evaluated. The output values of each neuron $OutL_{1,i}$ are then gathered to the input vector for the next hidden layer $\underline{OutL}_1$ and multiplied by the matrix $\mathbf{LW}_{1\to 2}$, weighting the outputs of each neuron from the preceding layer individually:

$$\underline{temp} = \underline{bias}_{1\to 2} + \mathbf{LW}_{1\to 2} \cdot \underline{OutL}_1, \tag{14.3}$$

| Abbr. | Name | Size | |
|---|---|---|---|
| **IW** | Input weights | $10 \times 2$ | |
| **LW** | Layer weights | $10 \times 10$ | from layer $j$ to layer $j+1$ |
| **OW** | Output weights | $1 \times 10$ | |

Table 14.1: Dimensions of the neural network weights.

| Input of the neural network | Output of the neural network | Output for a hidden layer |
|---|---|---|
| $\begin{pmatrix} x \\ y \end{pmatrix}$ $(2 \times 1)$ | $z$ $(1 \times 1)$ | $\underline{OutL}$ $(10 \times 1)$ |

Table 14.2: Dimensions of the neural network input and output vectors.

$$OutL_{2,i} = \tanh(temp_i). \tag{14.4}$$

A general conversion from the values of one hidden layer $j$ to the next hidden layer $j+1$ is:

$$\underline{temp} = \underline{bias}_{j \to j+1} + \mathbf{LW}_{j \to j+1} \cdot \underline{OutL}_j, \tag{14.5}$$

$$OutL_{j+1\ i} = \tanh(temp_i). \tag{14.6}$$

The output is then generated by a vector times vector operation:

$$z = bias_{10 \to \text{output}} + \mathbf{OW} \cdot \underline{OutL}_{10}. \tag{14.7}$$

The process of fitting the ANN according to the data set consisting of hundreds of mesh node coordinates is called training. Herein, the entries, also called weights, of the matrices **IW** and **LW** and the output weight vector **OW** are adapted such that all input pairs of x- and y-values produce the lowest residual $E$, that is the sum of the square of each deviation of $z_{NN}$ from the correct single output $z_{data}$ as possible:

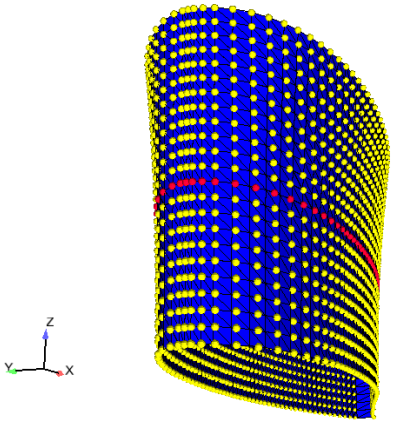
$$E = \frac{1}{2} \sum (z_{NN} - z_{data})^2. \tag{14.8}$$

A simple adaptation mechanism $\Delta w_{LW_{ij}}$ for each weight $w_{LW_{ij}}$ within a weight matrix $LW$ is based on the derivative of the square of the output error with respect to each single weight variation $dw_{LW_{ij}}$, and can be formulated like the following:

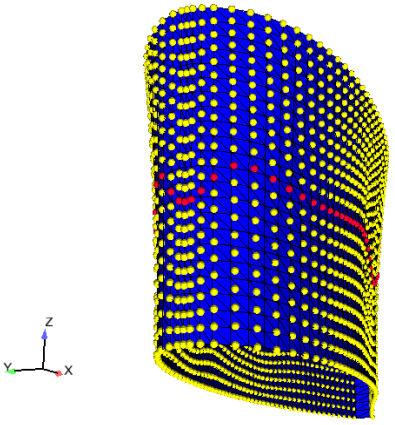
$$\Delta w_{LW_{ij}} = -\lambda \frac{dE}{dw_{LW_{ij}}}, \tag{14.9}$$

where $\lambda$ is an overall training parameter, which can be adapted manually according to the performance. There are many variations of this simple fitting algorithm; typically, the more sophisticated ones prevent oscillations during the training phase and overfitting. Overfitting occurs when the training parameters are too strongly adapted to the training data, which then leads to a bad performance in the region where there is no training data available; this is also called a bad generalization performance. In [Ney00] it is proven

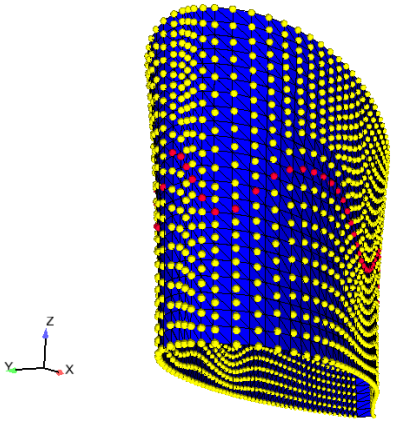
that a training algorithm for neural network can optimize the network so that at best the conditional probabilities of the training events can be met exactly, where one data set, in this case the presence of one x-, y- and z-value is considered as an event where the value z has a certain likelihood under the condition that the values x and y are given. For the shown geometry, a 10-layer MLP with 10 neurons in each hidden layer can be trained in that way that the maximal single error of all of the data points $z_{NN} - z_{data}$ is less than 0.01 percent. Therefore, the normal vector of each node can be calculated precisely so that a wave motion along the ship hull can be computed without crossing node paths on the hull (see Figure 14.4), where a wave motion (red nodes) along an imaginary ship hull is followed over four time steps. In the case of extreme wave motions along the hull of the VWT, where the waterline reaches the uppermost part of the hull, a direct mapping from the x- and y-coordinates to a z-coordinate for the entire hull shape is not possible. Then the hull must be subdivided in various parts so that a direct mapping is possible for the single parts of the hull. The training and testing of the neural networks for single parts of the hull showed good results. Therefore, this method is regarded as very effective for the embedding of ship hull data into flow simulation.



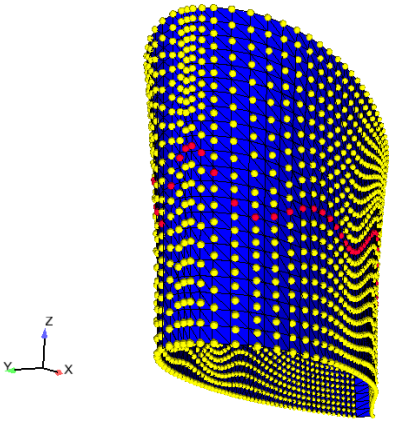

Wave along ship hull at time step 1.




Wave along ship hull at time step 2.




Wave along ship hull at time step 3.




Wave along ship hull at time step 4.

Figure 14.4: Wave along a fictitious ship hull.

# 15 Summary and Outlook

In this thesis, a mesh-moving strategy was developed for an unsteady Navier-Stokes fluid flow simulation around the propulsion system of a Voith Water Tractor (VWT) consisting of two Voith Schneider propellers (VSP) and a nozzle plate. In the beginning, the governing equations of water in motion, which is modelled as an incompressible fluid, were introduced, along with a brief description of the three most common spatial discretization approaches and an explanation of the space-time finite element discretization concept. Furthermore, it was shown that ensuring mass and momentum conservation and following a moving boundary within the fluid domain at the same time is a challenging task within the field of CFD. Given the broad variety of mesh moving schemes and also the choice of the discretization approaches, it was argued that, for the fluid flow simulation around the VWT propulsion system, the in-house space-time finite element fluid solver promises to have the highest accuracy when tracking the interface of the moving blades compared to alternatives like a solver combined with a moving Chimera mesh concept.

Based on already implemented mesh moving schemes like the Shear-Slip Update Mesh method (SSMUM) and the Elastic Mesh Update Method (EMUM), the necessary steps were followed to apply these methods to the given CFD 3D real-world problem. Whereas the application of the SSMUM approach did not reveal major difficulties, that of the EMUM approach did. The EMUM approach regards the finite element mesh as an elastic solid and solves the linear elasticity equation for the node motion. Therefore, the element deformation cannot be controlled explicitly and this can lead to tangling elements in regions of high relative motion of neighbouring elements, especially in a critical region between the bottom of the blades and the nozzle plate, which is situated just 2.6 centimetres below the blades of the VSP. There, the EMUM algorithm even produced tangling elements when only turning a single blade of a VSP a few degrees. It is important to mention that the generation of various meshes with different resolutions and the application of varying mesh motion parameters did not show a significant improvement of this EMUM behaviour.

While trying to partly prescribe the EMUM node motion in the critical region, a Concentric Shell Mesh Motion (CSMM) approach was worked out in parallel, where the motion of every mesh node was explicitly prescribed. On the one hand, the CSMM concept demands a high effort in programming the whole mesh generation, whereas the mesh motion algorithm is the simplest and cannot tangle. On the other hand, the EMUM approach goes along with an easier mesh generation, as commercial mesh generation tools like Gridgen can be used, but this method always shows an inherent danger of tangling and unpredictable mesh motion when a blade is turned back and forth several times. In order to directly compare these two alternative mesh motion concepts, a comparative refinement study was performed wherein the unsteady Navier-Stokes flow around a fixed

blade was calculated. For both methods, meshes with consecutively finer resolutions are created around a single blade of a VSP and then deformed when the blade was turned to the maximal angle of attack, regarding one full VSP rotation at the operation point of maximal thrust when the VWT runs on open water speed. The comparison between the EMUM and the CSMM approach revealed that the CSMM concept created worse elements in the critical region below the blade bottom and was very hard to handle when it came to the creation of smooth transitions from the most finest mesh resolution at the blade surface to the coarser zones of the mesh further away from the blade. This resulted in an overall worse mesh quality of the CSMM concept, and as a result the fluid flow refinement study performed with consecutively finer CSMM meshes did not show convergent force values exerted on the blade. In addition, when solving the fluid flow on most of the CSMM meshes, spurious oscillations of pressure could be observed in the first unsteady simulations due to the bad element quality, and for some meshes, a convergent fluid flow simulation could never be achieved.

On the contrary, the refinement study on the EMUM meshes showed a converging behaviour of the forces on the blade and that the two most finest meshes showed force values which varied only less than three percent from each other. Additionally, a second study proved that, considering the finest meshes of the EMUM concept, the highly deformed elements between the blade bottom and the nozzle plate did not influence the calculated force values more than four percent. For these reasons, the EMUM approach was selected for the full ten blade VWT proplulsion system fluid flow simulation.

Here again, a refinement study was performed with the similar mesh resolutions around the blade, as it was done with the single blade refinement study. Again, the two finest meshes showed average force values and also characteristics of the force values that agree very well. The force values within the horizontal plane parallel to the main fluid flow direction did not deviate more than five percent at any time, considering a simulation where the two VSPs were turned several full rotations. The average thrust of a single blade varied by only twelve percent compared to the measured and simulated data by Voith, but the characteristics of the force values deviated significantly according to Voith.

A probable reason for these strong deviations might be that the boundary layer, which evolves at the blade surfaces as a result of surface friction and attachment of the water molecules, was not resolved properly, even with the finest meshes used in the refinement studies. Using the boundary layer theory based on the work of Prandtl, Karman and Blasius, it was shown that it is essential to use the universal wall log law when simulating the fluid flow around the VWT propulsion system, considering the given computing resources. According to the author, the implementation of a wall log law function would be the next promising step of a further development of the XNS solver towards a more accurate simulation of the flow around the VWT propulsion system. Fortunately, the difficulties regarding a reliable mesh motion approach are overcome as the EMUM approach combined with the SSMUM and the space-time finite element discretization has proven to be robust and show good convergence behaviour.

Beside the flow simulation of the VWT propulsion system, first ideas about a free-surface flow simulation around the hull of the VWT were presented. The difficult task of representing the ship hull shape within the fluid flow solver could be overcome by the use

of artificial neural networks and a very special division of the ship hull, such that each geometric part can be approximated with a simple function, which maps two independent coordinates to the third one within a 3D space. Using this approach, the implementation of the complicated CAD spline based ship hull geometry could be avoided.

# 16 Appendix - Refinement Study Meshes

## 16.1 Meshes for the Single Blade Refinement Study

### 16.1.1 EMUM Meshes

| | |
|---|---|
| # elements | 1204352 |
| dsWings | 40 |
| dsBox | 200 |
| boundary_decay | 1,0 |
| n_circum | 100 |
| n_cylinder | 18 |
| n_cone | 25 |
| ratio | 2,0 |

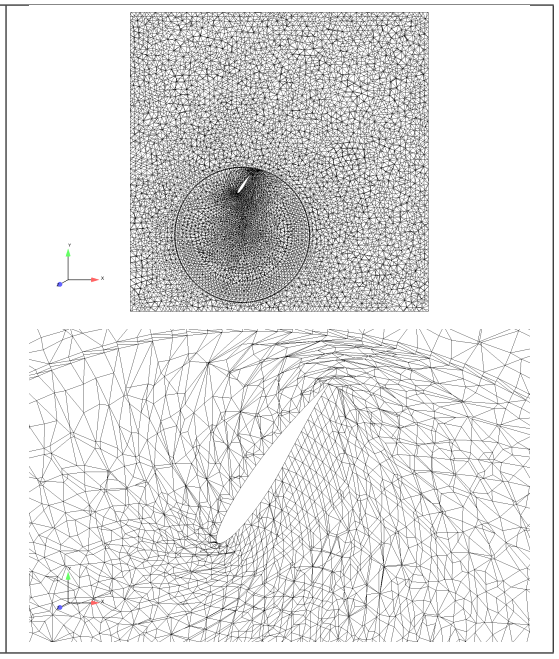


Table 16.1: EMUM.40 mesh and mesh generation parameters

| | |
|---|---|
| # elements | 2959924 |
| `dsWings` | 20 |
| `dsBox` | 150 |
| `boundary_decay` | 1,0 |
| `n_circum` | 120 |
| `n_cylinder` | 20 |
| `n_cone` | 50 |
| `ratio` | 1,0 |

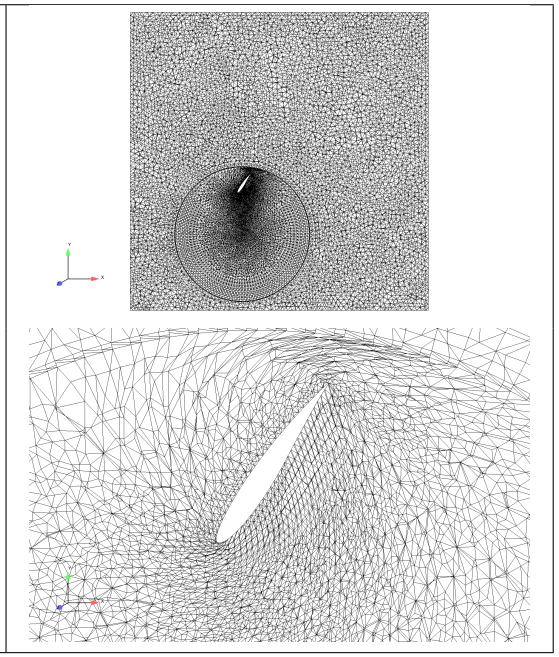


Table 16.2: EMUM.20 mesh and mesh generation parameters

| | |
|---|---|
| # elements | 5382674 |
| `dsWings` | 10 |
| `dsBox` | 150 |
| `boundary_decay` | 1,0 |
| `n_circum` | 140 |
| `n_cylinder` | 30 |
| `n_cone` | 60 |
| `ratio` | 1,0 |

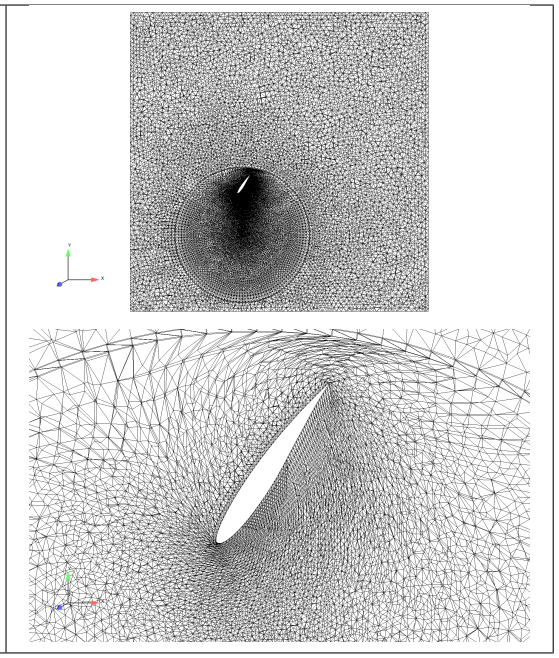


Table 16.3: EMUM.10 mesh and mesh generation parameters

| | |
|---|---|
| # elements | 12653669 |
| `dsWings` | 5 |
| `dsBox` | 150 |
| `boundary_decay` | 1,0 |
| `n_circum` | 180 |
| `n_cylinder` | 40 |
| `n_cone` | 65 |
| `ratio` | 2,0 |

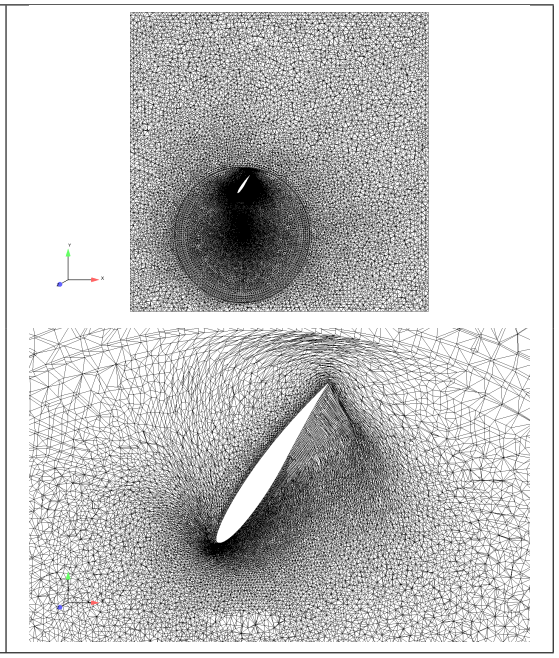


Table 16.4: EMUM.05 mesh and mesh generation parameters

### 16.1.2 Modified EMUM Meshes

| | |
|---|---|
| # elements | 1204665 |
| dsWings | 40 |
| dsBox | 200 |
| boundary_decay | 1,0 |
| n_circum | 70 |
| n_cylinder | 20 |
| n_cone | 40 |
| ratio | 1,0 |

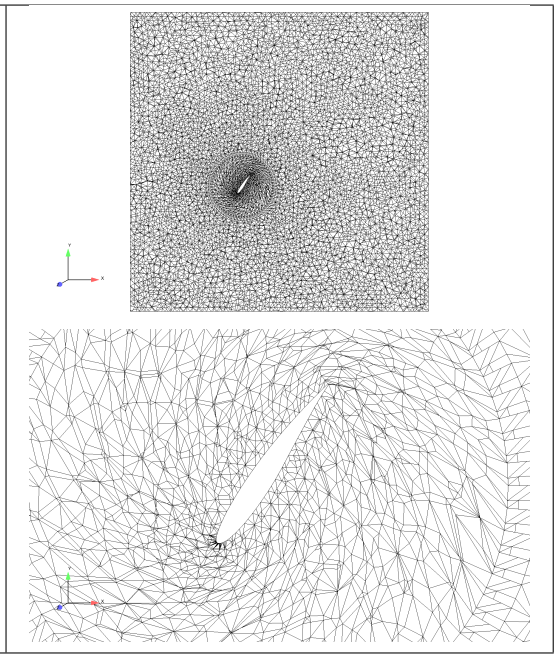


Table 16.5: EMUM.40.mod mesh and mesh generation parameters

| | |
|---|---|
| # elements | 2661962 |
| dsWings | 20 |
| dsBox | 150 |
| boundary_decay | 1,0 |
| n_circum | 100 |
| n_cylinder | 25 |
| n_cone | 40 |
| ratio | 1,0 |

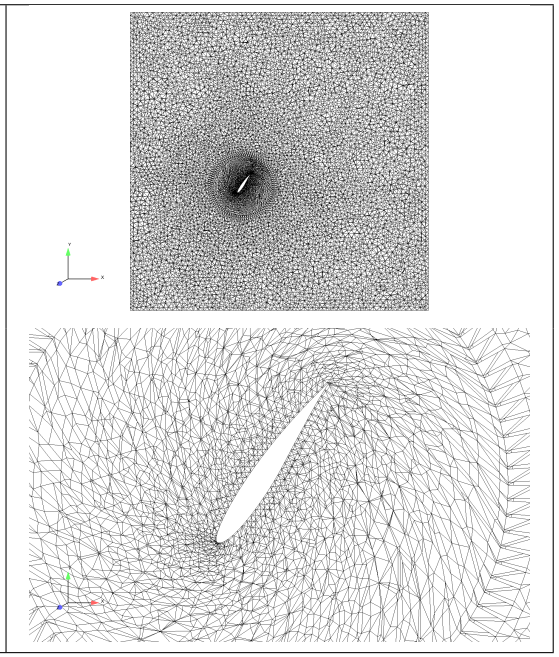


Table 16.6: EMUM.20.mod mesh and mesh generation parameters

| | |
|---|---|
| # elements | 4630671 |
| dsWings | 10 |
| dsBox | 150 |
| boundary_decay | 1, 0 |
| n_circum | 120 |
| n_cylinder | 30 |
| n_cone | 60 |
| ratio | 1, 0 |

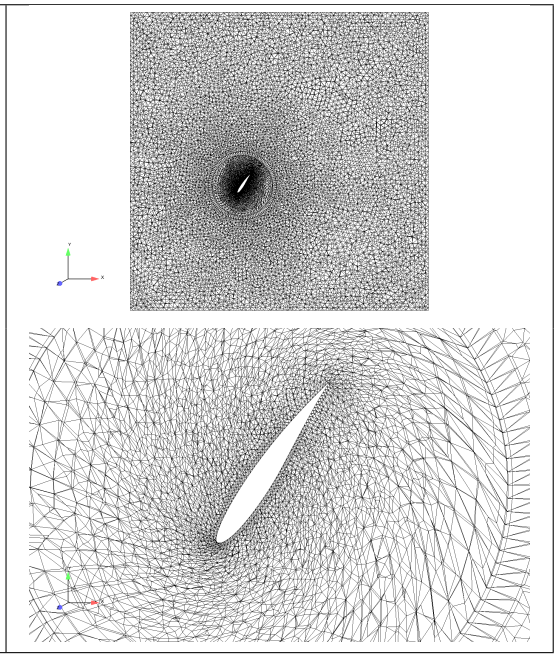


Table 16.7: EMUM.10.mod mesh and mesh generation parameters

| | |
|---|---|
| # elements | 12457247 |
| dsWings | 5 |
| dsBox | 150 |
| boundary_decay | 1, 0 |
| n_circum | 150 |
| n_cylinder | 40 |
| n_cone | 90 |
| ratio | 1, 0 |

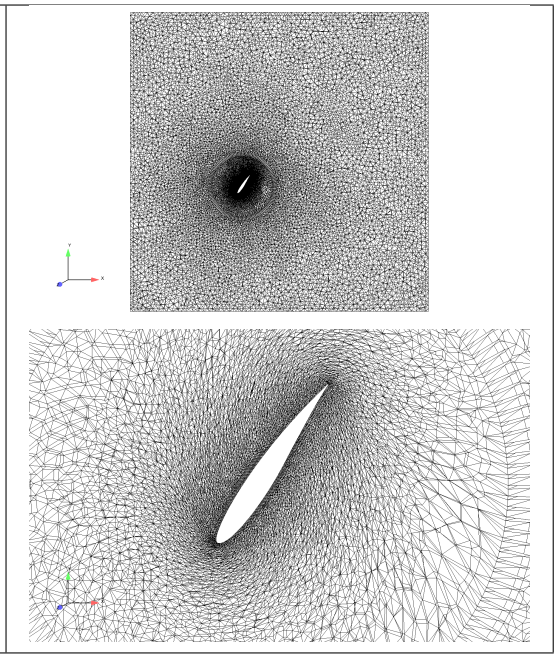


Table 16.8: EMUM.05.mod mesh and mesh generation parameters

### 16.1.3 CSMM Meshes

| | |
|---|---|
| # elements | 1204352 |
| dsWings | 40 |
| dsBox | 200 |
| boundary_decay | 1, 0 |
| n_circum | 100 |
| n_cylinder | 18 |
| n_cone | 25 |
| ratio | 2, 0 |

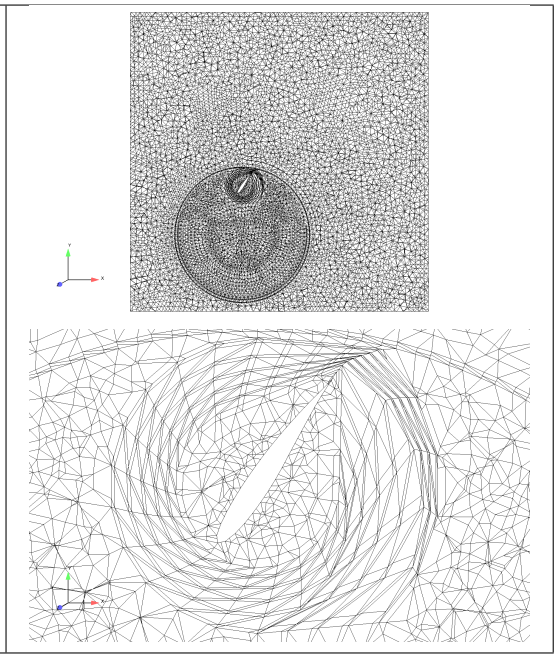


Table 16.9: CSMM.40 mesh and mesh generation parameters

| | |
|---|---|
| # elements | 2959924 |
| dsWings | 20 |
| dsBox | 150 |
| boundary_decay | 1, 0 |
| n_circum | 120 |
| n_cylinder | 20 |
| n_cone | 50 |
| ratio | 1, 0 |

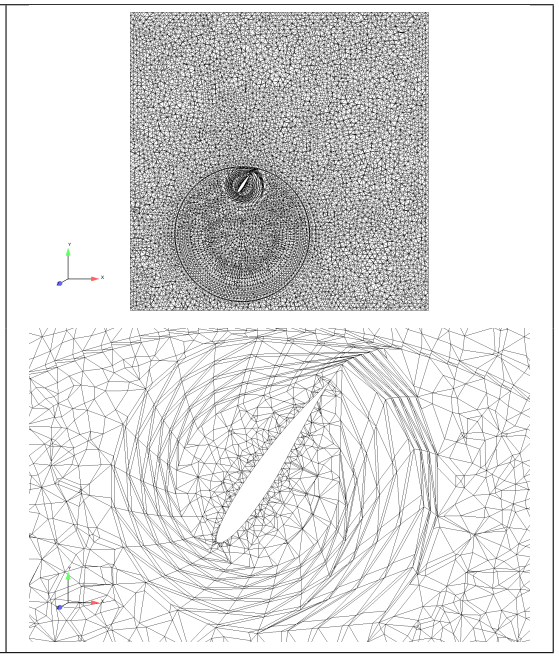


Table 16.10: CSMM.20 mesh and mesh generation parameters

| | |
|---|---|
| # elements | 5382674 |
| dsWings | 10 |
| dsBox | 150 |
| boundary_decay | 1,0 |
| n_circum | 140 |
| n_cylinder | 30 |
| n_cone | 60 |
| ratio | 1,0 |

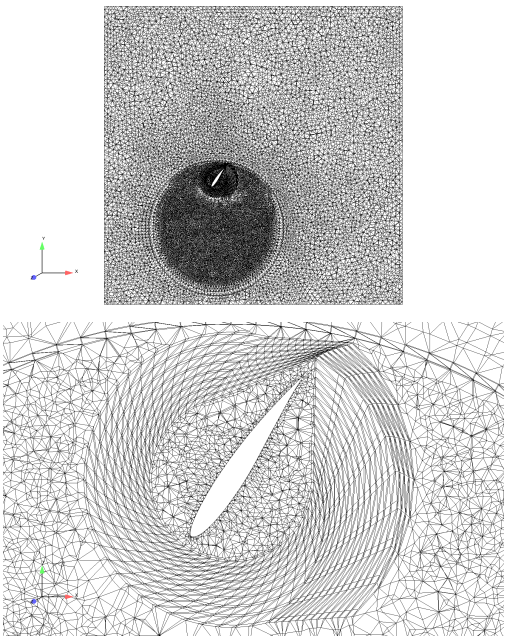


Table 16.11: CSMM.10 mesh and mesh generation parameters

| | |
|---|---|
| # elements | 12653669 |
| dsWings | 5 |
| dsBox | 150 |
| boundary_decay | 1,0 |
| n_circum | 180 |
| n_cylinder | 40 |
| n_cone | 65 |
| ratio | 2,0 |

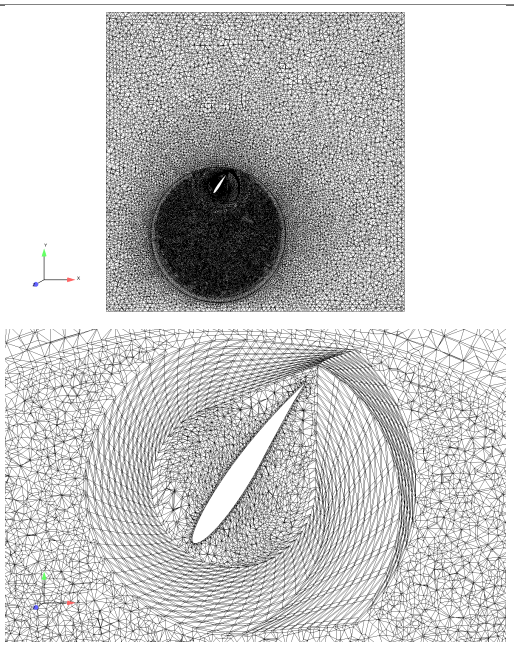


Table 16.12: CSMM.05 mesh and mesh generation parameters

### 16.1.4 Modified CSMM Meshes

| | |
|---|---|
| # elements | 1204665 |
| dsWings | 40 |
| dsBox | 200 |
| boundary_decay | 1,0 |
| n_circum | 70 |
| n_cylinder | 20 |
| n_cone | 40 |
| ratio | 1,0 |

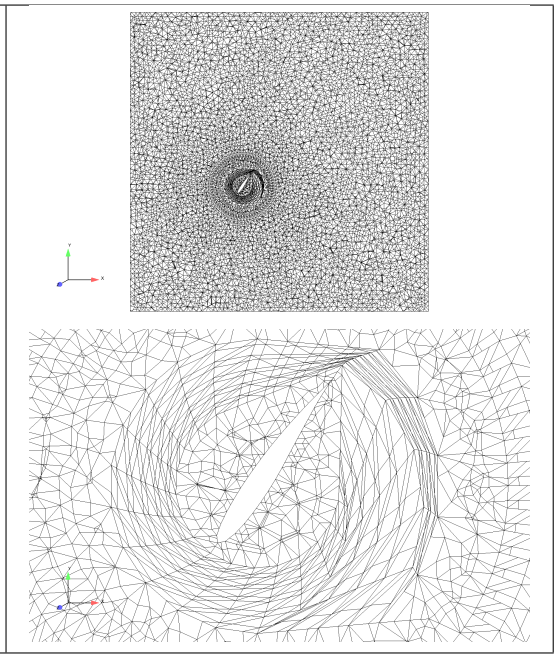


Table 16.13: CSMM.40.mod mesh and mesh generation parameters

| | |
|---|---|
| # elements | 2661962 |
| dsWings | 20 |
| dsBox | 150 |
| boundary_decay | 1,0 |
| n_circum | 100 |
| n_cylinder | 25 |
| n_cone | 40 |
| ratio | 1,0 |

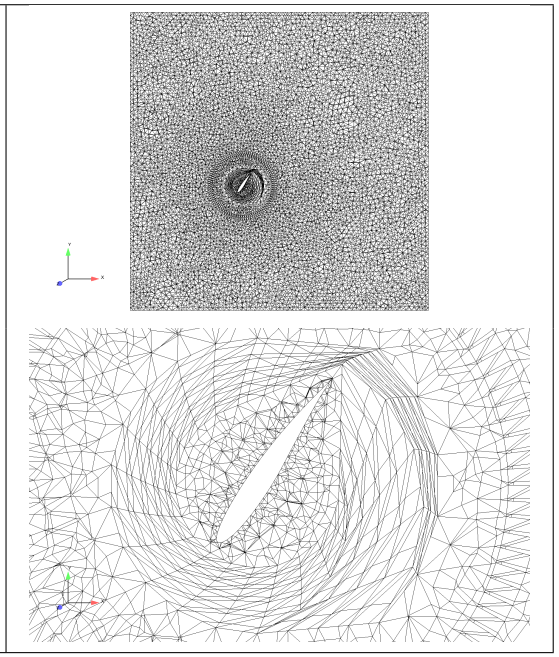


Table 16.14: CSMM.20.mod mesh and mesh generation parameters

| | |
|---|---|
| # elements | 4630671 |
| dsWings | 10 |
| dsBox | 150 |
| boundary_decay | 1,0 |
| n_circum | 120 |
| n_cylinder | 30 |
| n_cone | 60 |
| ratio | 1,0 |

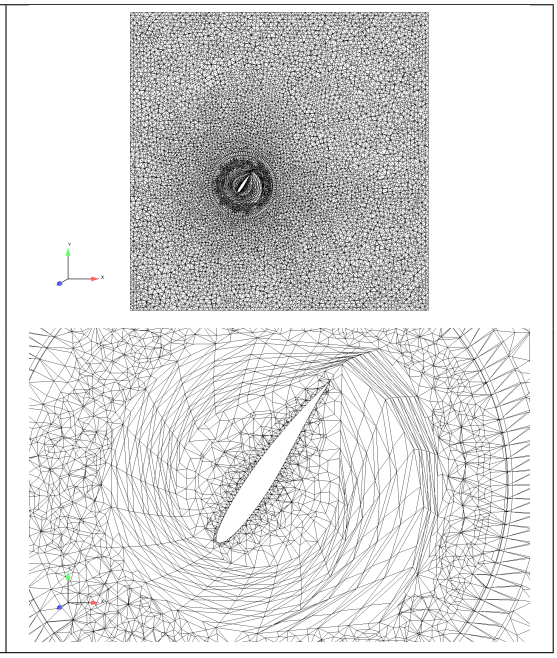


Table 16.15: CSMM.10.mod mesh and mesh generation parameters

| | |
|---|---|
| # elements | 12457247 |
| dsWings | 5 |
| dsBox | 150 |
| boundary_decay | 1,0 |
| n_circum | 150 |
| n_cylinder | 40 |
| n_cone | 90 |
| ratio | 1,0 |

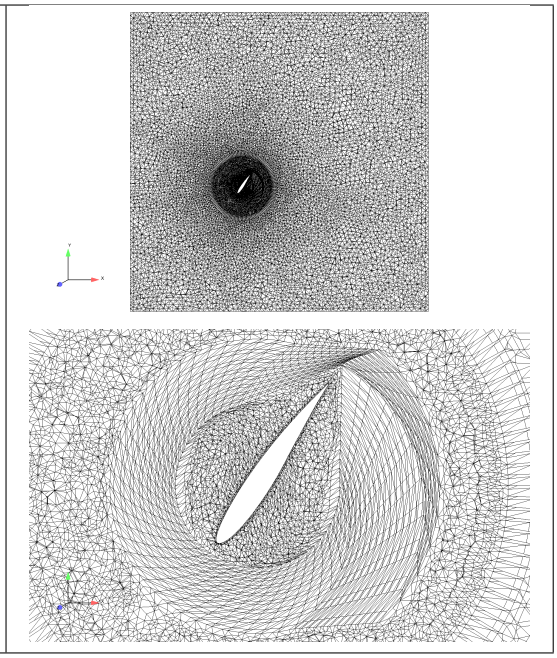


Table 16.16: CSMM.05.mod mesh and mesh generation parameters

# List of Figures

# List of Tables

# Bibliography


[BA02] M. Behr and F. Abraham. *Free-surface flow simulations in the presence of inclined walls.* Computer Methods in Applied Mechanics and Engineering 191, 5467-5483, 2002.

[BA03] M. Behr and D. Arora. *Shear-slip mesh update method: Implementation and applications.* Computer Methods in Biomechanics and Biomedical Engineering 6, 113-123, 2003.

[BBNP08] M. Behbahani, M. Behr, M. Nicolai, and M. Probst. *Towards Shape Optimization for Ventricular Assist Devices Using Parallel Stabilized FEM.* Proceedings of the NIC Symposium 2008 NIC Series, 325-331, 2008.

[BDS05] C. L. Bottasso, D. Detomi, and R. Serra. *The ball-vertex method: a new simple spring analogy method for unstructured dynamic meshes.* Computer Methods in Applied Mechanics and Engineering 194, 4244-4264, 2005.

[Beh92] M. Behr. *Stabilized finite element methods for incompressible flows with emphasis on moving boundaries and interfaces.* PhD Thesis Unitversity of Minesota, 1992.

[Beh00] M. Behr. *Stabilized space-time FEM and its applications to free-surface flows.* Proceedings of the Conference on Computational Engineering and Science 5, 2000.

[Beh01] M. Behr. *Stabilized space-time finite element formulations for free-surface flows.* Communications for Numerical Methods in Engineering 11, 813-819, 2001.

[BH08] Y. Bazilevs and T. J. R. Hughes. *NURBS-based isogeometric analysis for the computation of flows about rotating components.* Springer New York, 2008.

[BJ02] W. Fork B. Jürgens. *Faszination Voith-Schneider-Propeller Geschichte und Technik.* Koehlers Verlagsgesellschaft mbH Hamburg, 2002.

[BNP07] M. Behr, M. Nicolai, and M. Probst. *Efficient Parallel Simulations in Support of Medical Device Design, in Parallel Computing: Architectures, Algorithms and Applications.* Proceedings of the International Conference ParCo 2007 NIC Series, 19-28, 2007.

[BSHS07] E. Besnard, A. Schmitz, H. Hefazi, and R. Shinde. *Constructive neural networks and their application to ship multidisciplinary design optimization.* Journal of Ship Research 51, 297-312, 2007.

[BT99] M. Behr and T. Tezduyar. *Shear-slip mesh update method.* Computer Methods in Applied Mechanics and Engineering 174, 261-274, 1999.

[BT01] M. Behr and T. Tezduyar. *Shear-slip mesh update in 3D computation of complex flow problems with rotating mechanical components.* Computer Methods in Applied Mechanics and Engineering 190, 3189-3200, 2001.

[BWS98] M. Boehm, K. Wechsler, and M. Schaefer. *A parallel moving grid multigrid method for flow simulation in rotor-stator configurations.* Int. J. Numer. Meth. 42,175-189, 1998.

[Car07] J. Carlton. *Marine Propellers and Propulsion.* Butterworth-Heinemann, Elsevier, 2007.

[CBO00] G. Chiandussi, G. Bugeda, and E. Onate. *A simple method for automatic update of finite element meshes.* Communications in Numerical Methods in Engineering 16, 1-19, 2000.

[CET08] Callum Corbett, Christian Ewald, and Robin Tomcin. *Netzverformungsmethoden für den Voith-Schneider-Propeller.* CATS, Projektarbeit, RWTH Aachen, 2008.

[DH03] J. Donea and A. Huerta. *Finite element methods for flow problems.* John Wiley and Sons Ltd, 2003.

[Dym08] P. Dymarski. *Computations of the propeller open water characteristics using the SOLAGA computer program. Predictions of the cavitation phenomenon.* Archives of Civil and Mechanical Engineering, 2008.

[ESW05] H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics.* Oxford university press, 2005.

[GBT99] I. Gueler, M. Behr, and T. E. Tezduyar. *Parallel finite element computation of free-surface flows.* Computational Mechanics 23, 117-123, 1999.

[GPHDJ01] R. Glowinski, T. Pan, T. Hesla, and J. Periaux D. Joseph. *Fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies: Applications to particulate flow.* J. Comput. Phys. 169, 363-426, 2001.

[Grö00] T. Grönsfelder. *Numerische Verfahren zur Berechnung von CFD-Problemen mit rotierenden Einbauten.* Technische Universität Darmstadt, 2000.

[Gue06] Joel Guerrero. *Overset composite grids for the simulation of complex moving geometries.* MASCOT06 Poster Session, 2006.

[HFH89] T. J. R. Hughes, L. P. Franca, and G. M. Hulbert. *A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective-diffusive equations.* Computer Methods in Applied Mechanics and Engineering 73, 173-189, 1989.

[Hug87] Thomas J. R. Hughes. *The finite element method.* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1987.

[JMBF01] Z. Johan, A. C. M. Moraes, J. C. Buell, and R. M. Ferencz. *In-cylinder cold flow simulatin using a finite element method.* Computer Methods in Applied Mechanics and Engineering 190, 3069-3080, 2001.

[JPSU07] D. Juergens, M. Palm, S. Singer, and K. Urban. *Numerical optimization of the Voith-Schneider Propeller.* ZAMM, 2007.

[Kou03] K. Koushan. *Automatic hull form optimisation towards lower resistance and wash using artificial intelligence.* Fast2003 Conference, 2003.

[LL03] L. Lee and R. Leveque. *An immersed interface method for incompressible Navier-Stokes equations.* SIAM Journal of Scientific Computation 25, 832-856, 2003.

[MAB03] S. Murman, M. Aftosmis, and M. Berger. *Implicit approaches for moving boundaries in a 3-D cartesian method.* AIAA, 2003.

[MBK07] A. Masud, M. Bhanabhagvanwala, and R. A. Khurram. *An adaptive mesh rezoning scheme for moving boundary flows and fluid-structure interaction.* Computers and Fluids 36, 77-91, 2007.

[MH97] A. Masud and T. Hughes. *A space-time Galerkin/least-squares finite element formulation of the Navier-Stokes equations for moving domain problems.* Computer Methods in Applied Mechanics and Engineering 146, 91-126, 1997.

[Ney00] H. Ney. *Mustererkennung und neuronale Netze.* RWTH Aachen Vorlesungsskript, 2000.

[Oer95] H. Oertel. *Numerische Strömungsmechanik.* Springer-Verlag Berlin Heidelberg New York, 1995.

[OK95] K. Oh and S. Kang. *Numerical calculation of the viscous flow around a propeller shaft configuration.* International Journal for Numerical Methods in Fluids 21, 1-13, 1995.

[Pes02] C. S. Peskin. *The immersed boundary method.* Acta Numerica, 2002.

[RV05] Research and Development Department Voith. *General information about the Voith-Schneider Propeller.* 2005.

[SB08] R. Steijl and G. Barakos. *Sliding mesh algorithm for CFD analysis of helicopter rotor fuselage aerodynamics.* International Journal for Numerical Methods in Fluids 58, 527-549, 2008.

[Sch04] W. Schroeder. *Fluidmechanik.* Aachener Beiträge zur Strömungsmechanik, RWTH Vorlesungsskript, 2004.

[Sie02] R. Sieber. *Numerische Simulation technischer Strömungen mit Fluid-Struktur-Kopplung.* Technische Universität Darmstadt, Url = http://elib.tu-darmstadt.de/diss/000261/, 2002.

[ST02] K. Stein and T. Tezduyar. *Advanced mesh update techniques for problems involving large displacements.* Fifth World Congress on Computational Mechanics, 2002.

[TAB+96] T.E. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, V. Kalro, and M. Litke. *High performance computing techniques for flow simulations.* John Wiley and Sons, 363-398, 1996.

[TBL91] T. Tezduyar, M. Behr, and J. Liou. *Stabilized finite element formulations for incompressible flow computations.* Advances in Applied Mechanics 28, 1-44, 1991.

[TBL92] T.E. Tezduyar, M. Behr, and J. Liou. *A new strategy for finite element computations involving moving boundaries and interfaces - the deforming-spatial-domain/space-time procedure: I. The concept and the preliminary tests.* Computer Methods in Applied Mechanics and Engineering 94, 339-351, 1992.

[TBMJ92] T. Tezduyar, M. Behr, S. Mittal, and A. Johnson. *Computation of unsteady incompressible flows with the finite element methods - space-time formulations, iterative strategies and massively parallel implementations.* New Methods in Transient Analysis, 1992.

[TBML92] T.E. Tezduyar, M. Behr, S. Mittal, and J. Liou. *A new strategy for finite element computations involving moving boundaries and interfaces - the deforming-spatial-domain/space-time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders.* Computer Methods in Applied Mechanics and Engineering 94, 353-371, 1992.

[Wal05] C. Waluga. *Gittergenerierung für den Voith-Schneider-Propeller.* CATS, Projektarbeit, RWTH Aachen, 2005.

[Zel94] A. Zell. *Simulation neuronaler Netze.* ISBN 3-486-24350-0, 1994.

[ZHS02] A. Zubaydi, M. R. Haddara, and A. S. J. Swamidas. *Damage identification in a ship's structure using neural networks.* Ocean Engineering 29, 1187-1200, 2002.