Norwegian University
of Life Sciences

Master's Thesis 2016 30 ECTS
Department of Mathematical Sciences and Technology

Computational Fluid Dynamic (CFD) Study Investigating Wave Loads on a Torus at Deep Sea

Håvard Blålid Refvik
Environmental Physics and Renewable Energy

Preface

This thesis is submitted as fulfilment of a Master of Science degree in Environmental Physics and Renewable Energy at Norwegian University of Life Sciences.

The project was done under the supervision of Prof. Tor Anders Nygård at Norwegian University of Life Sciences, whose knowledge and expertise were of great help realising this thesis. I would also like to express my gratitude to Luca Oggiano at Institute for Energy Technology (IFE), who together with Prof. Nygård helped me set up a plan to validate the numerical wave tank correctly.

My special thanks also goes to Simen Strømsøyen at Stadt Towing Tank, who provided valuable help troubleshooting.

Finally I would like to thank the other students in room *TF 211* for countless discussions and the constantly fresh-brewed coffee.

Håvard B. Refvik
Ås, December 2016

Abstract

In this thesis work has been done to validate the open-source CFD-software *OpenFOAM*® and its additional package *waves2Foam* in two dimensions, by comparing numerically obtained vertical forces from waves and heave periods with experimental results. The aim of this validation process was to establish a foundation for further investigation in three dimensions, so that trustworthy simulations of a large torus at deep sea could be performed.

By implementing waves from 5th order Stokes theory, scaled from well documented experiments, an acceptable mesh and setup was found after a series of grid dependency studies. A similar grid dependency study was performed preparing for simulations measuring the heave period, showing that the numerical results closely followed the trend of the experimental results.

Finally, a simulation of the same waves was done in three dimensions, measuring the forces and moments from waves on a halfway-submerged torus. This resulted in plots showing some of the expected disturbances from interference within the torus, as well as some very interesting pictures showing how interference caused the creation of a breaking wave within the torus.

Samandrag

I denne masteroppgåva har det først vore arbeida for å validere CFD-programvara *OpenFOAM*® med tilhøyrande tilleggspakke *waves2Foam* i to dimensjonar. Dette har vore gjort ved å samanlikne vertikale krefter frå bølgjer og eigenperiode frå dropptestar med eksperimentelle data. Målet med denne valideringsprosessen var å lage eit fundament for vidare undersøkingar i tre dimensjonar, så pålitelege simuleringar av ein torus på djupt hav kunne bli utført.

Ved å implementere bølgjer frå 5. ordens Stokes teori, skalert frå vel dokumenterte eksperimenter, vart eit nøyaktigheitsstudie utført for å finne eit akseptabelt domene for bølgjetestar. For dropptestar vart ei same type nøyaktigheitsstudie gjort, som viste at trenden i dei numeriske resultata følgde dei eksperimentelle tett.

Til slutt vart ei simulering av bølgjer gjort i tre dimensjonar, med måling av krefter og moment frå bølgjer på ein halvvegs nedsenka torus. Dette resulterte i grafar kor den forventa forstyrringa frå interferens på innsida av torusen blir vist, i tillegg til nokre svært interessante bilete av korleis interferensen byggjer opp ei brytande bølgje på innsida av torusen.

Contents

Chapter 1

Introduction

Computational fluid dynamics, or *CFD* from here on out, is a way to numerically solve fluid problems, such as moving fluids in interaction with objects. It supplements experimental tests by allowing scientists and engineers to extract whatever data they need from simulations, wherever desirable. When simulations are set up correctly, CFD can therefore be a powerful extension to both the design and validation process of new products.

Whereas physical experiments in a wave tank involve time-consuming manufacturing, which can be expensive when several designs are to be tested, modifying a model in a functioning numerical tank is done quickly by one engineer alone. If time is of the essence, CFD also has the advantage of allowing the user to run several tests simultaneously, provided sufficient computational resources are available. However, CFD-analysis is not able to completely replace physical wave tanks. The legitimacy of the results from CFD heavily depend on both the accuracy of the underlying equations and the numerical setup, so a comparison with similar cases or final tests in a physical wave tank is still necessary.

In this thesis, CFD-analysis is used to investigate a floating torus on deep ocean. This is a geometry commonly used for structures such as fish farms in shallow fjords, which for several reasons wish to move further out at sea. It is also a geometry with a large potential for use in multiple types of wave energy converters and Spar-Torus Combinations (STCs). For these reasons, it is of great interest to gain more knowledge about the forces a torus will experience from waves at sea. Another unique and interesting aspect for a torus is the shape of waves in its "pool" and how these will affect both the forces and the heave period measured.

The open-source software *OpenFOAM® 2.4.0* was chosen for performing the simulations, primarily because it is an interesting project and completely free alternative to the very expensive commercial software on the market. By having an open-source code, it invites users to continue developing the program and create additional packages as they like. Today there are

several user-made wave-simulation toolboxes available for OpenFOAM. Among these, a tool box called *waves2Foam*, developed by Jacobsen et al. (2012) has been showing great potential and was chosen to be investigated further in this thesis.

To conclude, the aim of this thesis is as follows:

"To investigate if OpenFOAM and the additional package waves2Foam simulates waves and forces in a satisfactory manner, and if so, use this package to investigate heave period and the effect of wave induced forces on a torus in a numerical wave tank."

1.1 Layout of Thesis

Beginning in Chapter 2, an introduction to the theoretical background of computational fluid dynamics is given, in addition to a few scaling and inertia calculations necessary for the simulations in subsequent chapters. Chapter 3 builds on this theory to briefly describe OpenFOAM and the solvers used for the CFD-problems in this thesis, as well as giving an overview of other important programs used.

To make a three-dimensional analysis of a large structure at sea trustworthy, it is essential to verify the setup gradually to minimise the errors. In this thesis this is done as validation effort in 2D for two different full scale simulations, one part preparing for a fixed torus in waves and one for a heave test of the same torus in still water. Both these paths of validation processes are described in Chapter 4 and compared to experimental results. Based on these validation runs in 2D, results of full scale 3D simulations are presented in Chapter 5.

Summary and discussions are combined in Chapter 6, where results from simulations are investigated systematically to point out both the results worthy of continued work and fields of improvement. Finally, a conclusion is made in Chapter 7, rounding up the thesis.

1.2 Limitations

As often happens when setting ambitious goals in time-limited projects, a combination of lack of experience with OpenFOAM and deadline-day approaching too quickly led to some setbacks regarding results. Although heave simulations in two dimensions yielded the expected response, the move from 2D to 3D turned out not to be trivial. Despite trying several varieties of the mesh, as well as modifying initial and solver parameters, the heave tests of a torus in three dimensions either crashed or gave an unnatural response. As this problem was not solved in time for new

simulations to be run and results discussed in this thesis, the part with 3D-simulations of the heave of a torus was moved to Appendix B.

Chapter 2

Theoretical Background

This chapter will give a brief review of the theoretical underlying concepts of computational fluid dynamics and simulation of propagating waves, as well as describing the scaling methods and inertia calculations used throughout this thesis. The fluid dynamic equations and theories used are mainly obtained from Finnemore and Franzini (2002), Dias and Bridges (2006) and Anderson Jr. (1995). Constants and acronyms used are provided in appendix A.

The coordinate system shown in Figure 2.1 is used throughout the thesis, unless specified otherwise.

2.1 Governing Equations

Simply put, CFD is a technique to numerically solve the *Navier-Stokes Equations* (2.1-2.3), a set of second-order, nonlinear, partial differential equations, obtained by applying *Newton's law of*

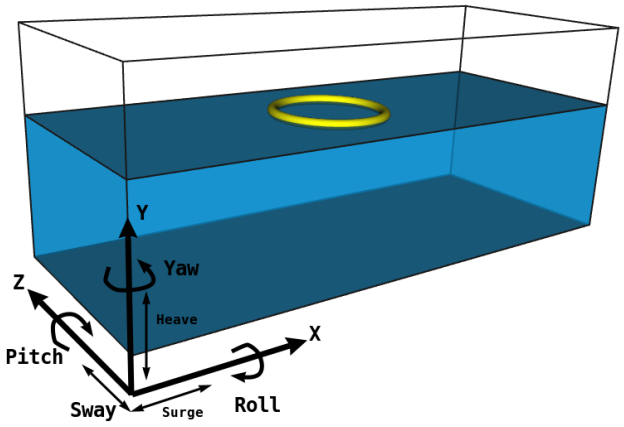


Figure 2.1: Coordinate system used in this thesis, where the inlet is closest to origin.

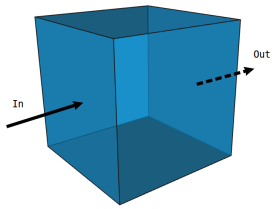


Figure 2.2: Illustration of a cell filled with water.

motion to a fluid element such as the cell seen in Figure 2.2. By calculating these equations, the motion of the fluid element can be described for one-, two- and three-dimensional flow. Although a general solution to them has never been found, the equations can be solved both analytically and numerically for specific scenarios. For an incompressible fluid with constant viscosity, the Navier-Stokes equations can be expressed in the following way in Cartesian coordinates:

$$-\frac{\partial p}{\partial x}+\mu\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}\right)=\rho\left(\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}+w\frac{\partial u}{\partial z}\right) \tag{2.1}$$

$$-\frac{\partial p}{\partial y}+\mu\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}+\frac{\partial^2 v}{\partial z^2}\right)-\rho g=\rho\left(\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}+w\frac{\partial v}{\partial z}\right) \tag{2.2}$$

$$-\frac{\partial p}{\partial z}+\mu\left(\frac{\partial^2 w}{\partial x^2}+\frac{\partial^2 w}{\partial y^2}+\frac{\partial^2 w}{\partial z^2}\right)=\rho\left(\frac{\partial w}{\partial t}+u\frac{\partial w}{\partial x}+v\frac{\partial w}{\partial y}+w\frac{\partial w}{\partial z}\right) \tag{2.3}$$

Where u, v and w are the velocity components in $x-$, $y-$ and z-directions respectively, t is time, ρ is the density of the fluid, p is the pressure, g is the acceleration of gravity in y-direction and μ is the dynamic viscosity of the fluid.

Another key concept to CFD is that the mass of the fluid flow is conserved, which can be described for incompressible flow by the differential equation of continuity:

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}=0 \tag{2.4}$$

Together, the Navier-Stokes equations (2.1-2.3) and continuity equation (2.4) accurately describe viscous, incompressible fluid flow. In ocean engineering applications such as this one, where we are looking at water waves and massive floating objects, the viscous effects of the fluid can be neglected in the equations above (Dias and Bridges (2006)). To further simplify the analysis of ocean waves, it is common to assume that the flow is irrotational ($\omega_x=\omega_y=\omega_z=0$) for non-breaking, deep ocean waves, so that the velocity potential function ϕ (2.5) can be used

to reduce the number of unknowns in the continuity equation (2.4).

$$u = \frac{\partial \phi}{\partial x}, \mathrm{v} = \frac{\partial \phi}{\partial y} \text{and } \mathrm{w} = \frac{\partial \phi}{\partial z} \tag{2.5}$$

Inserting the velocity components in equation (2.5) from the velocity potential function into the continuity equation, we end up with an alternative continuity equation for potential flow:

$$\nabla^2 \phi = \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + \frac{\partial^2 \phi}{\partial z^2} = 0 \tag{2.6}$$

The last equation can be recognised as the *Laplace's equation*, named after the French mathematician Pierre Simon de Laplace. It is a linear equation in terms of the potential function ϕ, where the boundary coefficients need to be specified for all surfaces of the numerical domain, as well as models within it. The boundary conditions and domain used in this analysis are explained in Section 2.2.

2.1.1 Volume of Fluid (VOF) Method

VOF is a numerical technique to track and localise the interface between different fluids, developed by Hirt and Nichols (1981) and later modified by Berberović et al. (2009). When using the VOF method, a phase fraction α is used to determine the amount of each fluid in each of the cells. For a floating object at sea, the cells containing water only will have an α-value equal to 1, while those that only contain air will have an α-value equal 0. At the interface, the α-value of the surface cells will usually be somewhere in between. This is because the VOF method calculates an approximation of the free surface, rather than defining an exact water level, which could lead to instabilities and disturbances. An example of two fluids divided using the VOF method is shown in Figure 2.3, where the water filled cells with $\alpha = 1$ are red and the ones with $\alpha = 0$ are blue.

The volume, phase fraction equation for the phase fraction α is described as:

$$\frac{\partial \alpha}{\partial t} + \Delta(\alpha u) = 0 \tag{2.7}$$

where t is time and u is velocity. By determination of the density of the air/water mixture within each cell using Equation 2.8, the phase fraction α is used to solve the Navier-Stokes equations.

$$\rho = \alpha \rho_{water} + (1 - \alpha)\rho_{air} \tag{2.8}$$

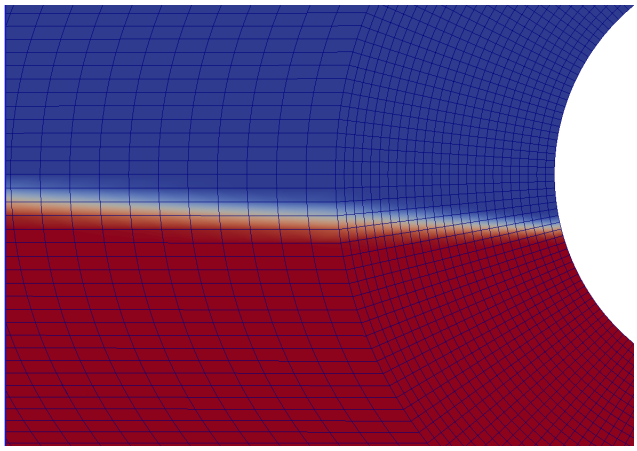


Figure 2.3: Visualisation from ParaView of the phase fraction close to a oscillating cylinder. The red zone represents $\alpha = 1$ and blue zone $\alpha = 0$, while the graded and white part are areas with $0 < \alpha < 1$.

where ρ_{water} is the density of water and ρ_{air} is the density of air.

2.1.2 Finite Volume Method (FVM)

FVM is a discretisation technique for partial differential equations. In OpenFOAM this is utilised by dividing the domain into cells, where the flux in and out of a cell is equal. This way parameters like velocity, pressure and α-value can be calculated at the centre of each cell and be interpolated between them.

2.2 Numerical Wave Tank

In physical wave tanks the most common wave generators are either flap or piston type, for which the movement of a paddle induces motion in a pool to create waves. On the opposite side of the tank, a beach is placed to kill off waves by forcing them to break. Creating waves numerically with the same type of wave maker would require a large amount of computational resources to simulate the moving paddle, as well as knowledge of how to match the paddles motion with the desired output of waves. A more efficient way to create waves in CFD is therefore to take advantage of an analytical wave model and use a region of the pool to let the waves build up numerically in accordance with a user-specified wave theory. In the wave generation package chosen here, these regions are called relaxation zones and can be set up either as wave generation zones or beach zones.

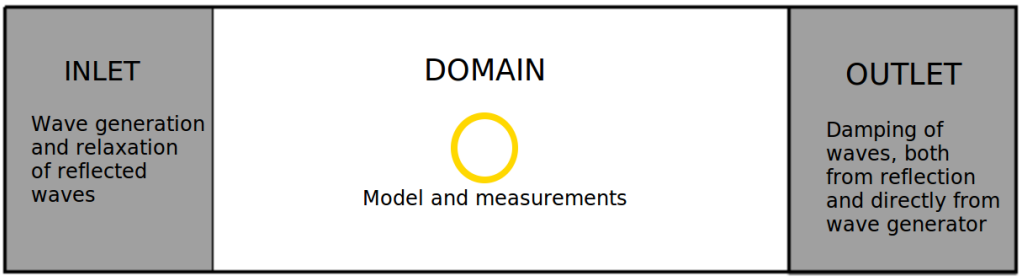


Figure 2.4: Schematic of a numerical wave tank seen from birds eye view. Not to scale.

2.2.1 Relaxation Zones

In this thesis, relaxation zones from the package *Waves2Foam* by Jacobsen et al. (2012) are implemented both at the inlet and outlet to generate waves and absorb both generated and reflected waves in the tank. To effectively do this, the relaxation function $\sigma(\epsilon)$ shown in Equation (2.9) is used, with a shape factor of 3.5.

$$\sigma(\epsilon) = 1 - \frac{exp(\epsilon^{3.5} - 1)}{exp(1) - 1} \tag{2.9}$$

$\sigma(\epsilon)$ is calculated for each cell in the relaxation zone visualised in Figure 2.4, where ϵ is defined such that $\sigma(\epsilon)$ decreases from 1 on the border between the domain and relaxation zone to 0 by the far end of the inlet or outlet.

Using this relaxation function, the velocity u or water fraction α is calculated in the relaxation zone for every time step, using Equation (2.10). In this equation θ represents either u or α, as both variables are calculated using this equation.

$$\theta(t) = (1 - \sigma)\theta_{target} + \sigma\theta_{computed} \tag{2.10}$$

In Equation (2.10) θ_{target} is specified by the wave theory implemented in the inlet zone, while it in the outlet zone is specified as either wet or dry cells for calm water. $\theta_{computed}$ are the values of α and u found by the VOF and Navier-Stokes equation for the previous time step.

2.3 Linear Wave Theory

Stokes 1st order wave theory, also known as *Airy-* or *linear wave theory* is a classical, still widely used wave theory. Although higher-order wave theories derived from Stokes 1st are more accurate for steep waves, Stoke 1st tends to work very well for deep-water waves when wave length

λ is much greater than wave height H. Some of its main assumptions are:

- The fluid is inviscid, incompressible, continuous and homogeneous.
- Surface tension and the Coriolis effect is neglected.
- The flow is irrotational
- The pressure is uniform and constant at the free surface
- Fixed and impenetrable bottom

For distance x and time t, the free surface elevation η of a two-dimensional Airy wave in positive x-direction can be calculated as:

$$\eta = \frac{H}{2} cos(kx - \omega + \theta) \tag{2.11}$$

where $k = 2\pi/\lambda$ is the wave number, ω is the angular frequency and θ is the initial phase. The latter is only included when there is interaction from other waves.

2.4 Nonlinear Wave Theory

As the Airy wave theory is only applicable for waves of low amplitudes, it has been necessary to develop higher order theories of the 1st order Stokes theory using geometric series. *Fifth order Stokes* theory, described by - amongst others - Skjelbreia and Hendrickson (1960) and Fenton (1985), offers accuracy for a wider range of wave steepness and height. This wave theory applies to an inviscid, incompressible fluid, and comes pre-installed with the waves2Foam package used in this thesis to generate waves. As it according to Skjelbreia and Hendrickson (1960) is generally applicable to deep ocean waves of finite depths, it was chosen for this thesis.

2.5 Scaling

To be able to compare results between the full-scale numerical tests and experimental results it is crucial to scale the model correctly. As the models used in both experiments and the numerical validation tests in Chapter 4 are of identical, cylindrical shape, they are said to have *geometric similarity* (Finnemore and Franzini (2002)). For geometrically similar cases, a method called *Froude scaling* can be used to determine wavelengths, periods and domain size for the full scale tests, based on data from the experiments. The scaling factor used is called the length

scale ratio L_R, which in this case is the ratio between the diameter of the full scale cylinder L_F and the diameter of the model L_M:

$$L_R = \frac{L_F}{L_M} \tag{2.12}$$

Relevant parameters used for Froude scaling are shown in Equations (2.13-2.16).

$$\text{Time:} \qquad T_F = \sqrt{L_R} T_M \tag{2.13}$$

$$\text{Mass:} \qquad M_F = L_R^3 \rho_R M_M \tag{2.14}$$

$$\text{Acceleration:} \qquad a_F = a_M \tag{2.15}$$

$$\text{Force:} \qquad F_F = L_R^3 \rho_F g F_M \tag{2.16}$$

Here, the density factor $\rho_R = \frac{\rho_F}{\rho_M}$, ρ_M is usually defined as the density of fresh water, in this case $\rho_M = 998 \frac{kg}{m^3}$ and $\rho_F = 1027 \frac{kg}{m^3}$ is the density of salt water.

Furthermore, a series of dimensionless parameters are commonly used to describe the relationship between an object and the waves and domain it is surrounded by. The following parameters are used by Dixon et al. (1979):

$$\text{Relative force:} \qquad F' = \frac{F}{\rho g \pi r^2 l} \tag{2.17}$$

$$\text{Relative amplitude:} \qquad a' = \frac{a}{D} \tag{2.18}$$

$$\text{Relative wavelength:} \qquad \lambda' = \frac{\lambda}{D} \tag{2.19}$$

2.6 The Model

Both the cylinder used for validation purposes in Chapter 4 and a 3D-model of a torus centred around the y-axis used in Chapter 5 were created using the open-source CAD-software FreeCAD(2016). For simplicity, the torus was chosen be halfway submerged, so that half of the

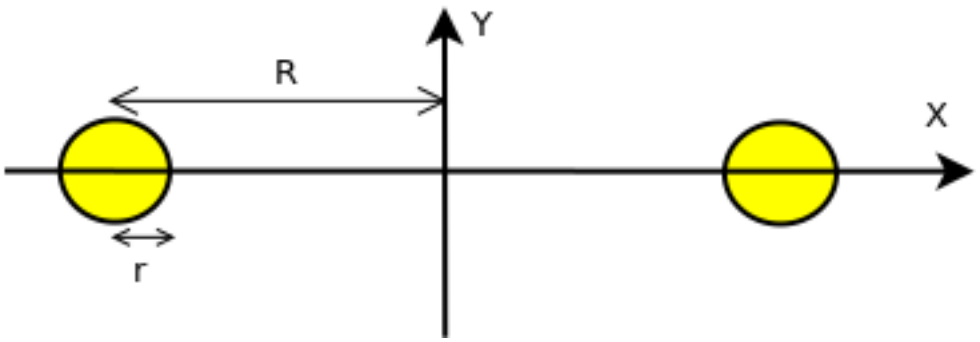


Figure 2.5: Cross-sectional view of the torus, where the large and small radii are illustrated. Dimensions is not to scale.

volume of the torus was below water level. This allowed a simple calculation of the mass of the torus, by multiplying the displaced water volume $\forbidden{V}_{\text{displaced water}}$ with the density of the water. As mentioned in Section 2.5, salt water with density equal to $\rho_{sw} = 1027\frac{kg}{m^3}$ was chosen for the full scale calculations. The mass M of the torus is then calculated using Equation (2.20), where the cross-sectional radius r and large radius R are illustrated in Figure 2.5.

$$M = \rho_{sw}\forbidden{V}_{\text{displaced water}} = \rho_{sw}\forbidden{V}_{torus}\frac{1}{2} = \rho_{sw}\pi^2 R r^2 \tag{2.20}$$

For the heave tests of a floating torus done in Chapter 5, knowledge about the inertia of the torus around all axis is necessary. When dealing with a simple, symmetrical object such as a torus, the inertia can easily be calculated without other tools than a calculator. For a torus in the coordinate system shown in Figure 2.5, the inertia I around each axis can be calculated using Equations (2.21) and (2.22).

$$I_Y = (\frac{3}{4}r^2 + R^2)M \tag{2.21}$$

$$I_X = I_Z = (\frac{5}{8}r^2 + \frac{1}{2}R^2)M \tag{2.22}$$

where M is the mass of the torus, R is the large radius and r is the radius of the cross-sectional cylinder, as shown in figure 2.5.

Based on the information provided in this chapter, the next few chapters will explain the software and methods used throughout the thesis.

Chapter 3

Software and Procedure

3.1 OpenFOAM®

Open Field Operation and Manipulation is an open source library developed by OpenCFD Ltd., created to solve fluid and heat problems numerically. It consists of a large number of customisable numerical solvers for CFD problems, as well as both pre- and post-processing utilities. In addition to being licence free and compact, one of its greatest advantages compared to other CFD solving software is that the user can customise the publicly available source code, so that the solvers and boundary conditions fit the users needs. This has allowed for a vast number of specialised additions, such as the package waves2Foam used in this thesis.

OpenFOAM does not come with a graphical user interface, so all setup is done in text files in the project directory and run from the terminal. Due to this the learning curve of OpenFOAM is much steeper than most other CFD-software. The amount of documentation and available support is also limited compared to commercial software, although aid and hints for troubleshooting can be found in online forums such as CFD-Online(2016). Once the user has gained some experience setting up the cases, most processes can be automated so that modifications and rerunning simulations can be done quickly. Most of the information used to build each case in this thesis was found in the official OpenFOAM manuals Users(2016) and Programmer's(2016). Rather than setting up new cases from scratch, the tutorials *DTCHull* and *floatingObject* included in the OpenFOAM installation provided a good starting point for the creation of each case.

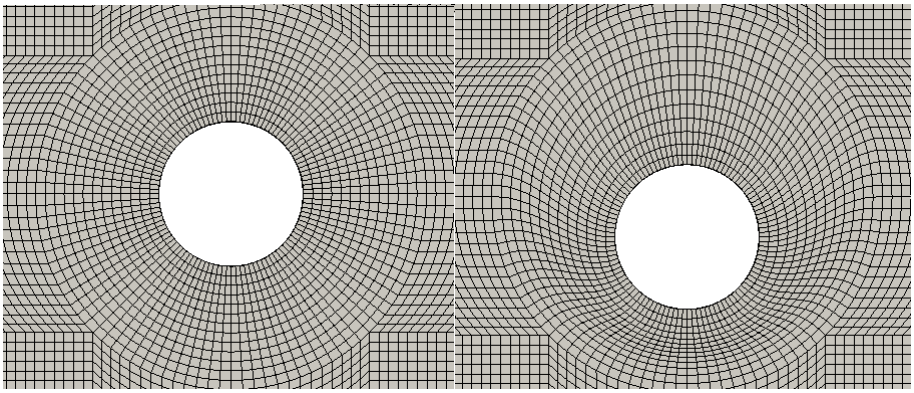


Figure 3.1: Mesh deformation of a dynamic mesh around a cylinder. The left side of the figure is the initial configuration and the right side shows the deformed mesh, adjusted to move with the motion of the body. Notice how only a finite region of the mesh is deformed.

3.1.1 Waves2Foam

To perform the CFD analysis in this thesis, the aforementioned additional package waves2FOAM was installed together with OpenFOAM 2.4.0. Waves2Foam lets the user generate, measure and absorb waves in a numerical wave-tank. It was developed by Jacobsen et al. (2012) and is based on the `interFoam` and `interDyMFoam` solvers provided in OpenFOAM. `interFoam` and `waveFoam` are solvers for two incompressible fluids and can be solved for dynamic meshes such as a freely floating torus with the options `interDyMFoam` and `waveDyMFoam`. A *mesh* is a commonly used term for the grid shaped connection between different cells, seen in Figure 3.1. All the mentioned solvers track the interface between the two fluids and interaction between fluids and structures using the VOF- and FVM-methods described in Sections 2.1.1 and 2.1.2.

For moving models, regions of the mesh need to change from one time step to the next, as illustrated in Figure 3.1. This is solved using `interDyMFoam` and `waveDyMFoam` by stretching and compressing the cells either in the entire domain or within an finite region surrounding the moving object. The size of the moving region and the weight, centre of mass and inertia of the object can all be specified easily in the same dictionary `dynamicMeshDict` when moving from a static to a dynamic mesh.

3.1.2 Boundary Conditions

In order to solve the governing equations at the surface of each object or at the edge of the domain, boundary conditions must be specified for all cells facing them. An important word to take note of in this regard is the term *faces*, which is defined as the side of the cell creating a border of either the domain or geometry. An example of a face is the side of a cell shaping the

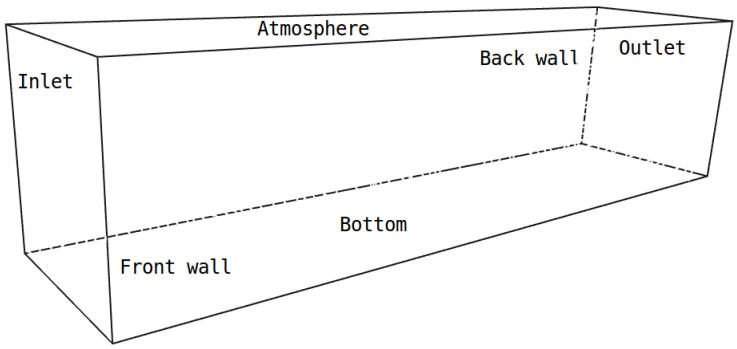


Figure 3.2: Schematic representation of the computational domain where boundary surfaces are named.

cylinder in Figure 3.1. There are plenty of different boundary conditions available for the variety of uses of OpenFOAM, but only the ones used in this thesis will be described here.

For simulations in 1D and 2D there is an option to choose a face to be *empty*, which is a way to force OpenFOAM to solve the problem in 2D. This is a "go-around"-method used because OpenFOAM always generate geometries in three dimensions. In the 2D-simulations where the entire mesh is constructed using `blockMesh`, the sides were chosen to be the type *empty*. Another common boundary condition used in this thesis is *fixedValue*, whereas the name implies, the value of either a vectorial or scalar variable is specified at the boundary.

symmetryPlane, *slip* and *zeroGradient* are different methods of choosing the normal gradient of the face to be zero, with each their unique specifications. *symmetryPlane* is a boundary condition usually applied to faces where symmetry exists in the geometry and the flow field is in three dimensions. It is a commonly used boundary condition on the mirror border when a symmetrical mesh is split in half before simulation to reduce costs. An example is a numerical drag test of a ship, where the flow field and geometry is identical on both sides. By splitting the mesh in two in width direction, the number of cells is reduced by 50% while maintaining the accuracy. In this thesis, *symmetryPlane* patch was used to make OpenFOAM treat the mesh as a 2D domain, despite the fact that parts of the mesh refined by `snappyHexMesh` was in 3D. This because it does not interfere with the flow. The bottom boundary - and side walls for simulations in 3D - were chosen as patch-types *zeroGradient* for pressure and *slip* condition for velocity.

At the top of the domain, the *atmosphere* is specified to have a boundary type called *totalPressure*. This patch makes the total pressure constant.

The boundary condition used for the inlet is a special type provided with the waves2Foam-package called *waveAlpha*, where the α field and velocity u is specified according to the chosen wave theory and wave parameters defined by the user.

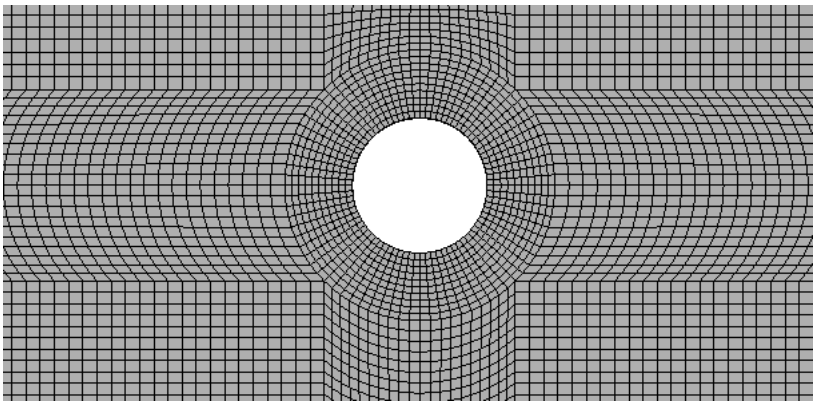


Figure 3.3: A section of the 2D mesh created by `blockMesh`.

Turbulence Model

OpenFOAM supports both Reynolds-Averaged Simulation (RAS) and Large-Eddy Simulation (LES) turbulence models for incompressible and compressible flows respectively. In this thesis, viscous effects were ignored by not implementing a turbulence model at all and rather applying a slip condition to the torus and walls. This simplifies both the setup of each case and potential troubleshooting, while preserving a reasonably accurate result assuming that the forces are dominated by inertia.

3.2 Mesh

An accurate mesh is essential to obtain the best solution possible. To define a domain for the tests, an initial basic mesh was created using the mesh generator `blockMesh`. As implied by its name, `blockMesh` splits the defined domain into a user specified number of blocks. Relying on a single dictionary called `blockMeshDict`, it can be modified however the user desires.

For the 2D-simulations done in Chapter 4, `blockMesh` was used to create both the domain and the cylinder within it. In later chapters, where a 3D-domain was needed, a rectangular domain was created with `blockMesh`, before the mesh modifier `snappyHexMesh` was used to hollow out and refine the region around the torus.

As described in Chapter 5.3 of the OpenFOAM Users Manual, each block in the `blockMeshDict` is defined by its edge points, and the position of these points decide the shape of the block. The block is later split into a defined number of cells in each direction. Further modifications can be done by grading the mesh, gradually increasing the length of each cell in the specified direction. In addition to straight lines, `blockMesh` also allows the user to define arcs between the edge points. Both features can be seen in Figure 3.3. Utilising arcs and blocks, most shapes in the 2D

domain can be created using `blockMesh`. However, for 3D-cases and especially curvatures in 3D-space, meshing in `blockMesh` alone is a tedious task.

3.2.1 SnappyHexMesh

After the initial mesh was generated using `blockMesh` the aforementioned mesh generator `snappyHexMesh` was used to adapt the mesh to the geometry of the floating torus. By carefully editing `snappyHexMeshDict`, the dictionary controlling the mesh generator, new cells were added, removed and reshaped to best fit the structure. `snappyHexMesh` can both import *STL*-files and create user-specified simple geometries to be hollowed out or refined. As it refines in three directions by standard, a mesh refined using `snappyHexMesh` will be three-dimensional.

3.3 Post-Processing in ParaView

ParaView from Kitware (2016) is an open-source post-processing software in which OpenFOAM is very well integrated. By using simple commands in the terminal, meshes, simulations or problematic faces and cells can be visualised in 3D in a ParaView window. Being among the most commonly used open-source post-processing software, user-documentation is easy to find and assistance provided in online forums or in the manual Kitware (2016).

All figures except for plots and schematics in this thesis are screenshots from either ParaView 4.1.0 or the open source CAD-program FreeCAD(2016), version 0.16.

3.4 MATLAB®

All plots in this thesis were made using MATLAB R2016b from MathWorks (2016) by either directly extracting data from log files or using preprocessed data from terminal scripts. MATLAB is a powerful analytical programming language and software package, widely used by engineers and scientists. Although being commercial software it is free for most students, so combined with great documentation and previous experience with the language it was decided to use MATLAB to analyse forces, momentum and surface elevations.

3.5 Computational Resources

In this thesis the 2D-simulations consisted of maximum a few hundred thousand cells, so that the total number of cells when expanding to the 3D-domain would not be too computationally

expensive. Because of this limitation it was possible to perform the 2D-simulations on a laptop with 8GB RAM and Intel's 8-virtual-core i7-3630QM processor. Although this is not an especially powerful setup in the world of CFD, it was considered sufficient to perform analyses in 2D.

In Chapter 5 several millions of cells were needed, so it would have taken an 8-core laptop weeks too complete a single simulation. After failing to compile the waves2Foam package on a few external HPC-resources, OpenFOAM 2.4.0 and waves2Foam were both successfully installed on an Ubuntu 14.01 server instance on EC2 by Amazon(2016). EC2 is a highly customisable HPC-resource that lets the user modify the cluster needed based on a pay per hour policy. Being one of the largest providers of HPC-computing, there is also plentiful documentation to be found on Amazons web pages and lots of help to be found in forums.

Chapter 4

Validation of Waves and Numerical Wave Tank in Two Dimensions

4.1 Procedure

In this chapter, several simulations were done in two dimensions to verify that the heave period and the interactions between waves and a small, cylindrical section of the torus were correctly calculated using OpenFOAM. To do so the validation process was split in four parts, where the forces or centre of mass-displacement were compared to experimental results. The procedure used in this validation chapter is similar to that of several other validation reports on wave generation, amongst others a master thesis by Bruinsma (2016) and paper by Westphalen et al. (2009).

The order of the validation procedures was as follows:

1. Wave propagation
2. Wave induced forces on a fixed cylinder
3. Heave test of cylinder
4. Mesh modifications

4.2 Wave Induced Vertical Forces on a Fixed Cylinder

In a research paper by Dixon et al. (1979) a comparison of forces from waves on fixed horizontal cylinders was made between theory and experiments, with the purpose of finding an equation

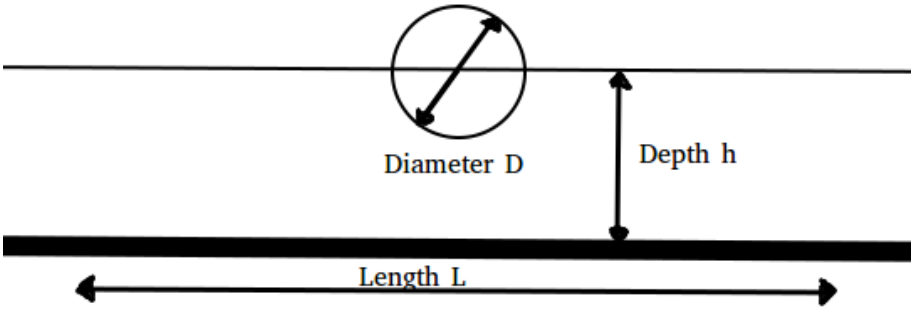


Figure 4.1: Schematic of the numerical wave tank.

	Dixon	Numerical	scale
Diameter	0.100m	3.70m	37.0
Wave height	0.100m	3.70m	37.0
Wave period	0.986s	6.00s	6.09

Table 4.1: Table of wave parameters, scaled from experiments by Dixon et al. (1979).

to predict the vertical forces from waves. In his work, several experiments were done with different wave heights and relative depths of the cylinder, in a domain illustrated in Figure 4.1. However, as this thesis investigates forces on a halfway-submerged torus, only an experiment with a halfway submerged cylinder was considered.

4.2.1 Propagation of Waves Generated by Waves2Foam

To check that the waves at the location of the cylinder were correct, Waves2Foams built-in wave probes were used to log the wave height in an empty pool. By placing probes at the planned centre of the cylinder, wave heights were measured and saved to a file, before being plotted using MATLAB. A 5th order Stokes wave with a wave height of 3.70m and a period of 6.00s is shown in Figure 4.2. This is a scaled up version of one of the waves from Dixon et al. (1979), where the experimental wave height is 0.100m and period of 0.986s for a cylinder with diameter 0.100m. Based on the dimensionless relative wavelength of 15.6 from the same experiment, the full-scale numerical wavelength was calculated to be 57.8m. Remaining parameters are described in table 4.1.

As can be seen in the plot in Figure 4.2, the wave probe located at the planned position of the cylinder measured a nice propagation throughout the domain without large changes in amplitude and period.

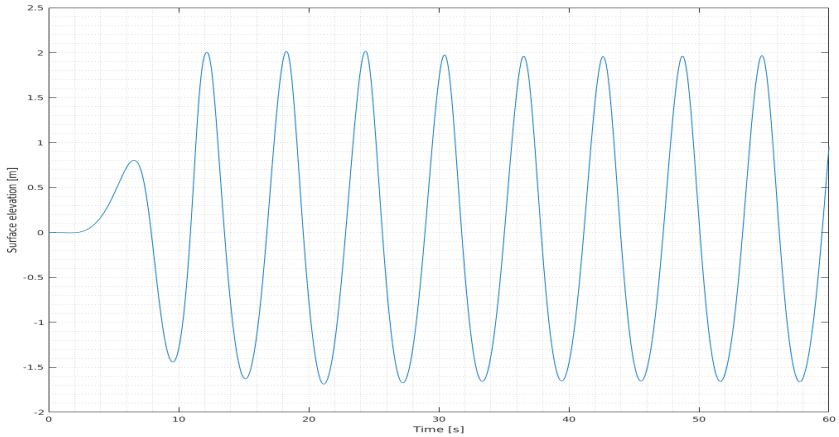


Figure 4.2: Propagation of a 5th order Stokes wave with wave height 3.7m and a period of 6s.

4.2.2 Forces on a Fixed Cylinder in Waves

After validating that the waves propagated in a satisfactory manner, a cylinder of diameter $3.70m$ was placed in the domain, at the same position the probe had previously been. To limit costs of computation and avoid *edge-effects* from the ends of the cylinder, a two-dimensional domain similar to the one in Figure 4.3 was used. Here, the walls on each side were defined as *empty*-patches, which were described in Section 3.1.2.

The mesh used was made purely using the `blockMesh` utility described in Section 3.2. In Figure 4.3, a circular mesh can be seen surrounding the hollowed-out cylinder, before it is split into blocks. For simplicity, the diameter of the outer, circular region was chosen to be exactly two cylinder diameters and to ensure accuracy the width and height of the cells were chosen so that the shape of them closely resembled squares. The numerical wave tank had a length of $L = 4\lambda$ and depth of $d = 5D$.

A comparison between the vertical forces calculated by OpenFOAM and the experimental results from Dixon et al. (1979) can be seen in Figure 4.4, where three different refinement levels of the mesh are plotted. Based on these results it was decided to continue using a mesh with 12 cells per cylinder diameter (cpcd) - in this case $\Delta x = \Delta y = 0.300m$ - as a minimum when measuring forces from waves on circular objects.

4.3 Heave of a Floating Cylinder

To simulate the motion of a floating torus and measure its heave period, it was first necessary to verify the numerical domain and motions of a dynamic mesh. This was done by investigating a cylinder in 2D, but now comparing results of a numerical drop test with experimental results

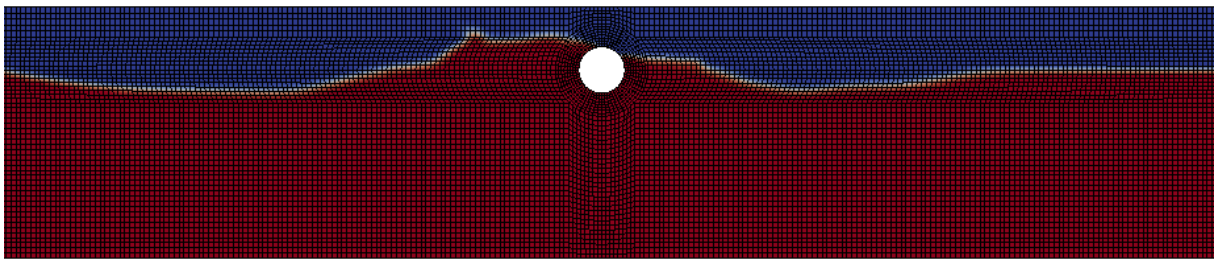


Figure 4.3: Numerical wave tank constructed using `blockMesh`, with a cell size of 12 cells per cylinder diameter.

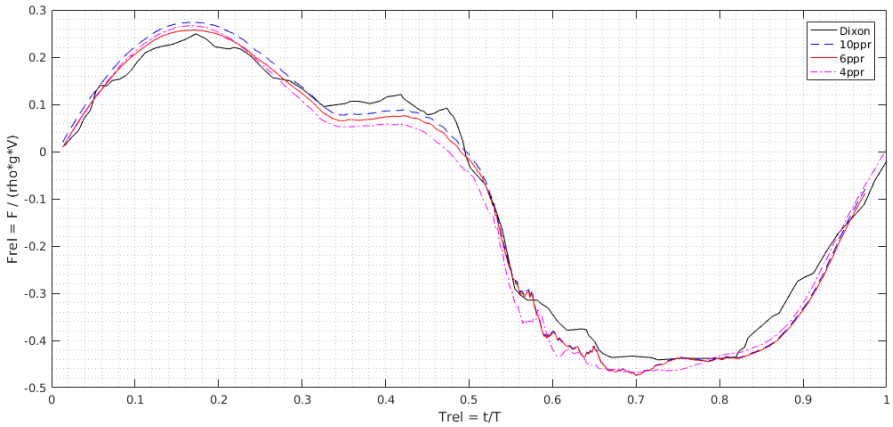


Figure 4.4: Comparison between the experimental results obtained by Dixon et al. (1979) and numerical results obtained by different levels of mesh refinement.

	Experiment by Ito	Numerical	Scale
Diameter	0.1524m	3.700m	24.28
Initial elevation	0.02540	0.6167m	24.28
Wave tank length	6.000m	222m	37.00
Wave tank depth	1.240m	33m	26.61

Table 4.2: Table of wave tank parameters for the simulation of a numerical heave test. Comparison between Ito's physical wave tank and the numerical wave tank used here.

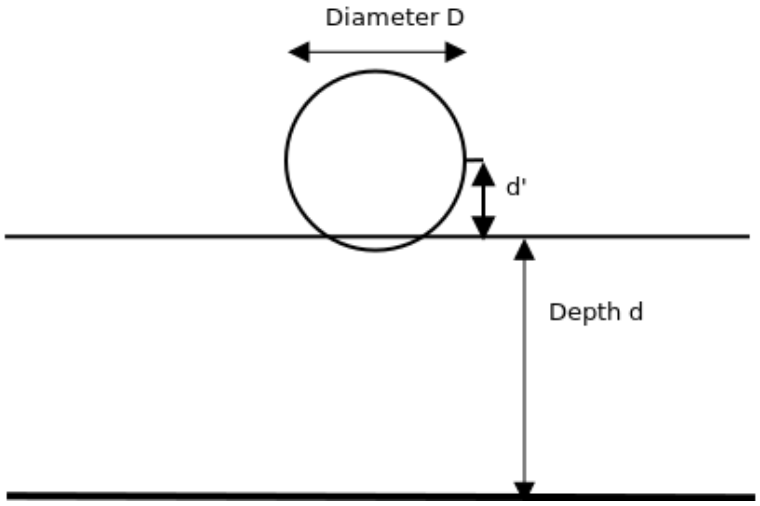


Figure 4.5: Schematic of the heave tests done on a cylinder in 2D. Be aware that the dimensions are not proportional.

from a drop test in a wave tank by Ito (1977). Comparison with experimental data made it possible to verify whether OpenFOAM correctly calculated the effects of added mass and that the movement of the mesh was captured in a satisfiable manner. In the experiment by Ito (1977), several drop tests were performed in a physical wave tank, from which the vertical position of the cylinder was measured and logged. Using Froude scaling, a scaling parameter was found based on the diameter of the numerical cylinder and the diameter of the one used in the experiments. Using this scaling factor, the initial numerical elevation d' was calculated to be $0.6167m$. The rest of the parameters used can be found in Table 4.2 and a schematic of the domain is shown in Figure 4.5.

Ito's experiment took place in a 6.00m long, 1.24m deep wave tank, using a cylinder of diameter 0.1524m. Although using regular Froude scaling the numerical tank would calculate the wave tank to have a length of $148m$, an extra-long wave tank of $222m = 60D$ was chosen to be certain that no reflection would come from the walls in each end.

A comparison between the results of the numerical drop test and the experiment by Ito (1977) is shown in Figure 4.6. This plot shows that the relative period in the numerical test was

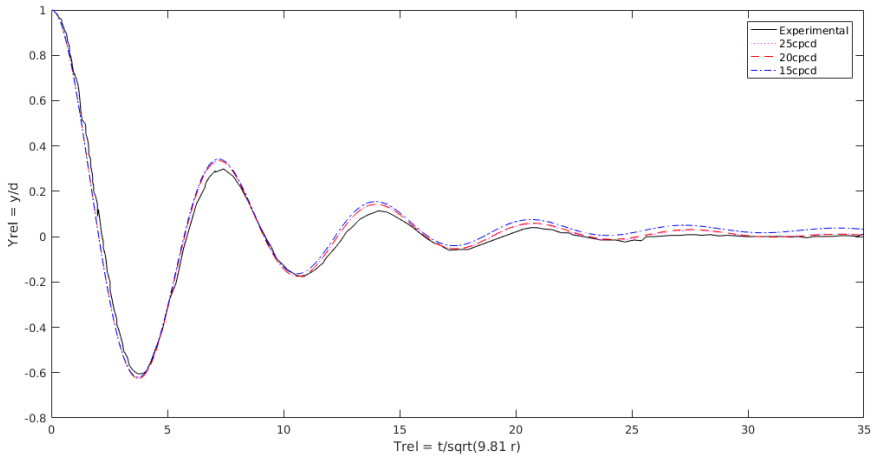


Figure 4.6: Comparison between the numerical heave test and the experimental test performed by Ito, using relative amplitudes and relative time.

quite similar to that of the experimental one. However, it seemed to have some negative drifting over time compared to the experiment. The relative amplitudes showed a close similarity below the water surface, while the positive amplitudes were further apart. This means that the setup and/or OpenFOAM did not perfectly capture the effects of added mass or surface tension of the fluid. As time did not allow further investigation of the issue and the general trend of the simulation seemed to be quite accurate, it was nevertheless decided that this setup with a resolution of 20 cpcd was sufficiently precise to begin heave tests of the torus in 3D.

4.4 Preparation for Simulations in Three Dimensions

After the domain, cell sizes and waves were validated in 2D, new versions of these meshes had to be made for the 3D cases.

4.4.1 From BlockMesh to SnappyHexMesh

When moving from 2D to 3D creating a mesh using `blockMesh` alone will rarely suffice. As described in Section 3.2, a full domain was created using `blockMesh`, before it was modified to fit the input geometry using `snappyHexMesh`. To ensure that this change of mesh did not effect the solutions negatively, a comparison was made between the waved induced vertical forces on a cylinder using `blockMesh` alone and in combination with `snappyHexMesh`. The structural difference between the two types of mesh is displayed in Figure 4.7.

The results from the simulation using the two different meshes is shown in Figure 4.8. The

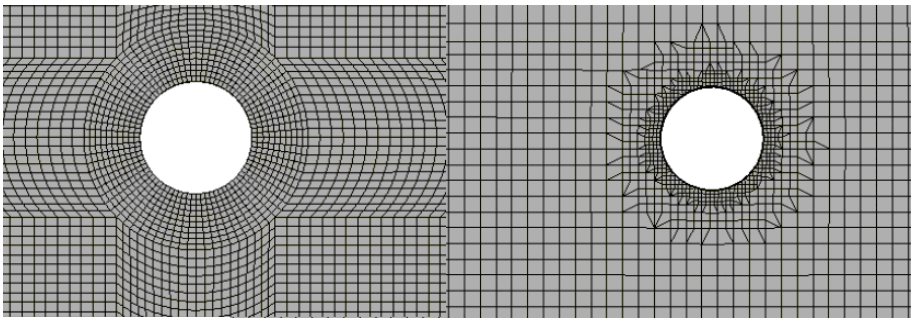


Figure 4.7: Illustration of two different methods of meshing. Cylinder and domain made with `blockMesh` on the left side VS domain made using `blockMesh` with a cylinder hollowed out using `snappyHexMesh` on the right. In this figure the mesh modified by `snappyHexMesh` has a much coarser background mesh, but with two levels of refinement the surface layer is even finer than in the picture to the left.

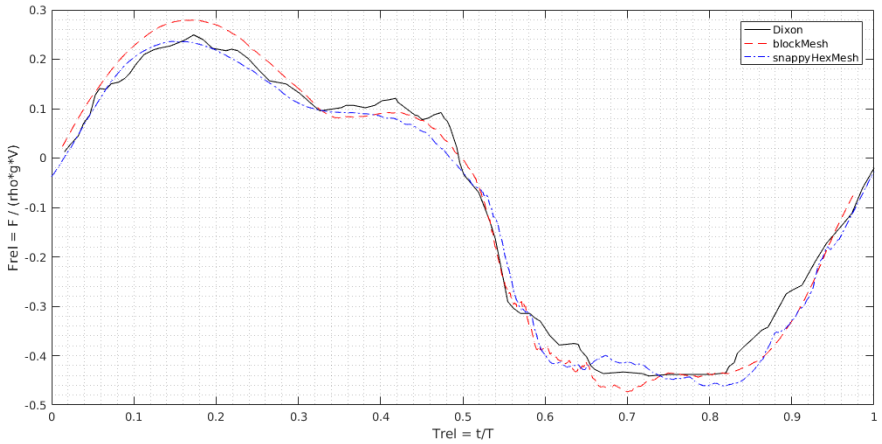


Figure 4.8: Comparison between the relative results from experiments by Dixon, numerical results obtained by `blockMesh` alone and numerical results refining the cylinder using `snappyHexMesh`.

difference here is small; in fact the cylinder hollowed out by `snappyHexMesh` seems to yield a more accurate solution than from `blockMesh` alone. This was assumed to be because of a small extension of the inlet relaxation zone, as well as an increased number of cells in proximity of the cylinder. As the result was at least as good as the one made using `blockMesh` and showed no large increase in accuracy when increasing the number of cells in the mesh, no further investigation was done to find out what caused the improvement.

4.4.2 Further Modifications of the Mesh

A critical point of all CFD simulations is the cost/time used for the computations. As a decrease in the number of cells generally decreases the time needed for computation, it was of interest

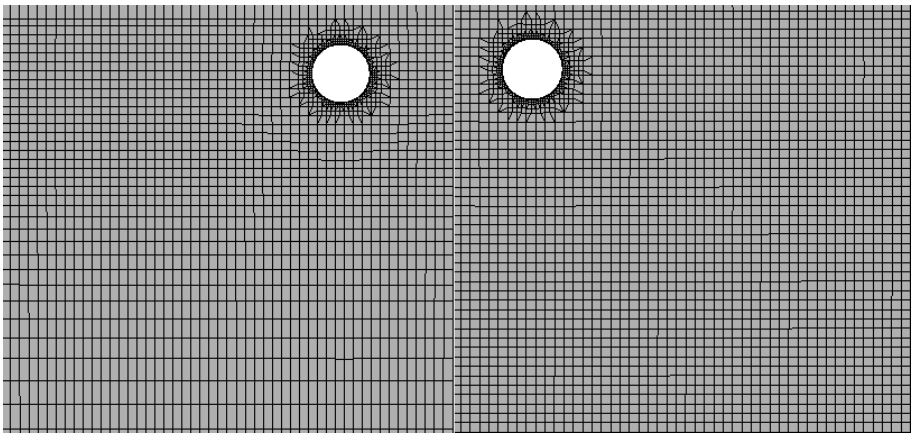


Figure 4.9: Visualisation of a graded mesh compared to a non-graded mesh. The grading is done vertically in the y-direction, with a factor of 0.25.

to reduce the number as far as possible without decreasing the accuracy. Using the grading method described in Section 3.2 the number of cells was reduced in regions of less importance such as the atmosphere and bottom, by gradually reducing the number of cells from central regions towards the less critical ones. A picture showing a graded versus a non-graded mesh is shown in Figure 4.9.

When the mesh was graded the number of cells was reduced by 23.5% (see Table 4.3), even though the most refined region surrounding the cylinder was slightly expanded. To verify that this did not affect the results, the plots in Figure 4.10 were made to compare the accuracy of the two meshes. As the effects of the modifications and grading in this case were small, in addition to reducing the number of cells considerably, it was decided to continue using a graded mesh in the 3D-scenario.

Mesh type	Number of cells
`blockMesh`	13 312
`snappyHexMesh` not graded	15 326
`snappyHexMesh` graded	11 721

Table 4.3: Overview of number of cells for each modification of the mesh.

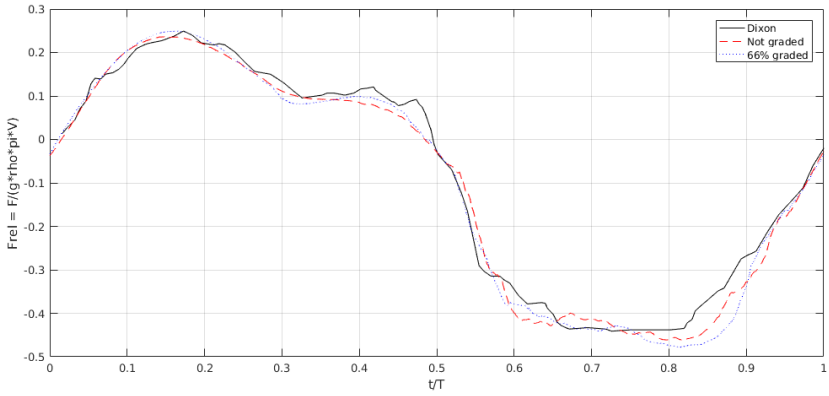


Figure 4.10: Comparison between results from Dixon's experiment and results from simulations with regular and a slightly refined graded mesh.

Chapter 5

Results From Simulations in Three Dimensions

In this chapter, the setup and results from the simulations performed on the complete 3D-model of a torus is shown.

For the halfway submerged torus of large radius $R = 35m$ and small radius $r = 1.85m$ shown in Figure **??**, the moment of inertia around each axis was found by solving the inertia equations in Section 2.6. Resulting data for the torus can be found in Table 5.1.

5.1 Waves on Fixed Model

After simulating waves on a cylinder using both a 2D domain made with `blockMesh` and a thinly sliced 3D domain with `snappyHexMesh`, the results were considered sufficiently accurate for

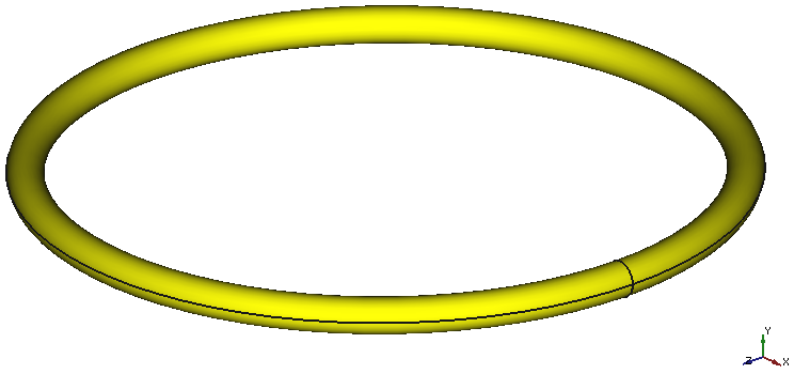


Figure 5.1: CAD drawing of the torus made in FreeCAD. Axis system can be found in the bottom right part of the picture

	Constant	Value	Unit
Large radius	R	35.00	m
Cylinder radius	r	1.850	m
Mass	M	$1.214 \cdot 10^6$	$\frac{kg}{m^3}$
Inertia around X-axis	I_X	$1.490 \cdot 10^9$	$kg \cdot m^3$
Inertia around Y-axis	I_Y	$0.7460 \cdot 10^9$	$kg \cdot m^3$
Inertia around Z-axis	I_Z	$1.490 \cdot 10^9$	$kg \cdot m^3$

Table 5.1: Size, weight and inertia parameters of the full size torus, halfway submerged in salt-water of density $1027 \frac{kg}{m^3}$.

	Value	Unit
Length	285	m
Width	210	m
Depth	37	m
Number of cells	3 723 152	

Table 5.2: Wave tank parameters for a fixed, halfway submerged torus.

simulations of a complete torus to begin. A domain with the dimensions defined in Table 5.2 was created using a more complexly graded `blockMesh`, splitting the domain into 27 blocks, three in each direction. To perform a simulation of a fixed torus at deep sea, the depth d has to be larger than half of the wavelength. It was here chosen to be $d = 10D = 37m$.

The mesh used for the simulations is shown in Figures 5.3 - 5.4, where the three-dimensional grading made in `blockMesh` is visible. The grading was first performed in the z-direction and the y-direction, gradually increasing the length of the cells in both directions away from the torus, horizontally and vertically. This was done both to minimise the number of cells and to numerically dissipate waves towards the borders of the domain. As can be seen in the same figures, grading in the x-direction is slightly different. To avoid dissipation of the generated waves, the cell size in x-direction was kept constant until past the back end of the torus, where grading was again used towards the outlet of the domain.

After the graded domain was constructed, an inner cylindrical region surrounding the torus was refined using *refineMesh*, before the meshing tool `snappyHexMesh` was used to cut the shape of a torus out of the mesh. Using `snappyHexMesh`'s own refinement-tools, another level of refinement was added within a region of 1 meter surrounding the torus, as well as in the centre of it to capture the motion of waves both around the torus and in the pool accurately.

When the mesh consisting of 3 723 152 cells had been completed, the same 5th order Stokes waves as in Section 4.2.1 with a wave height of $H = 3.7$m and period $T = 6$s were applied to the three-dimensional numerical wave tank. The relaxation zone at the inlet had a length

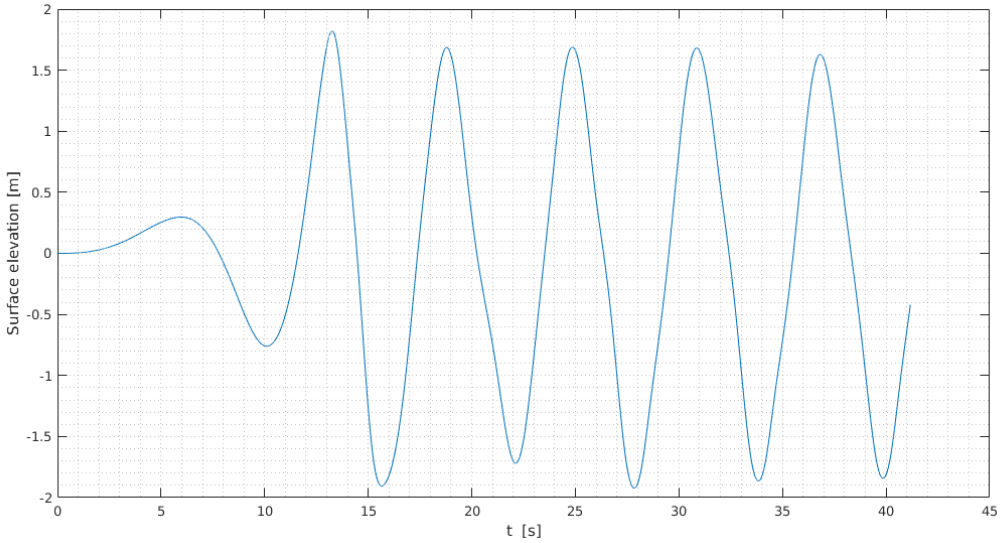


Figure 5.2: Surface elevation during the simulation of the fixed torus in waves. Wave probe was placed 15m from the torus, in direction of the wave maker.

of 1.5λ, while the one at the outlet had a length of 0.75λ. To be able to draw a connection between forces on the torus and the waves inducing them, a wave probe was placed in front of the torus, 15m closer to the inlet. A plot of the surface elevation measured by this probe during the simulation can be seen in Figure 5.2.

Pictures showing some of the interaction between the torus and waves are shown in Figure 5.5 on page 33, while pitch momentum and the most important force measurements from the simulation is shown in Figure 5.6 on page 34.

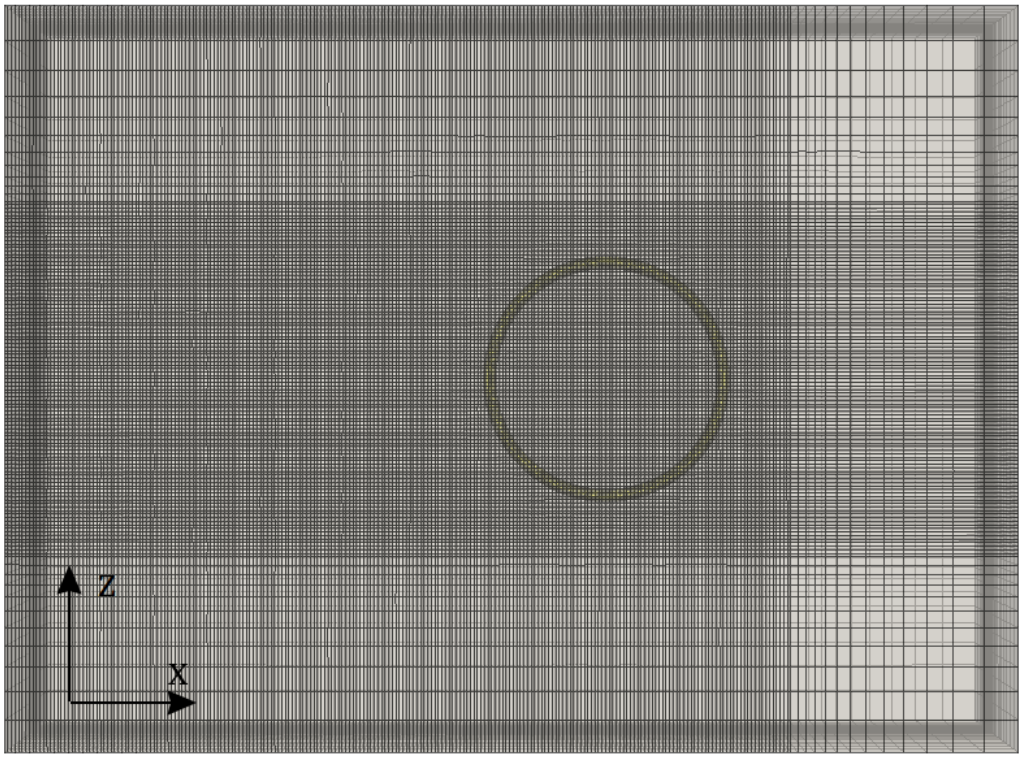


Figure 5.3: Birds eye view showing the graded mesh in x- and z-direction.

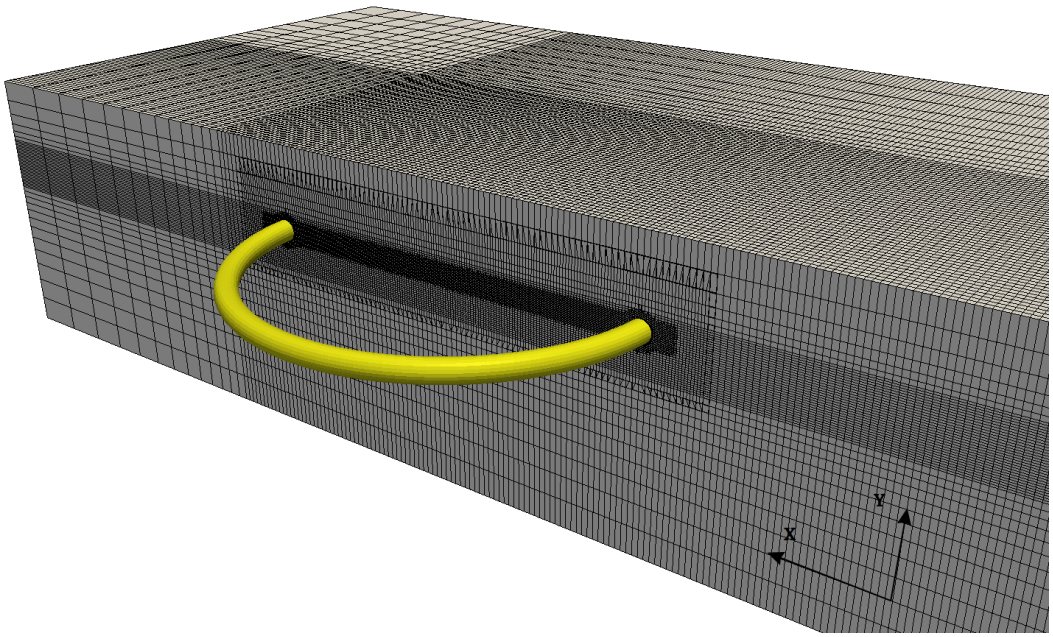


Figure 5.4: Refined zones surrounding the fixed torus used for simulations of waves. View from the front corner of a domain split in half at $z = 0$. Notice the grading in the horizontal x-direction behind the torus and in y-direction towards the atmosphere and bottom

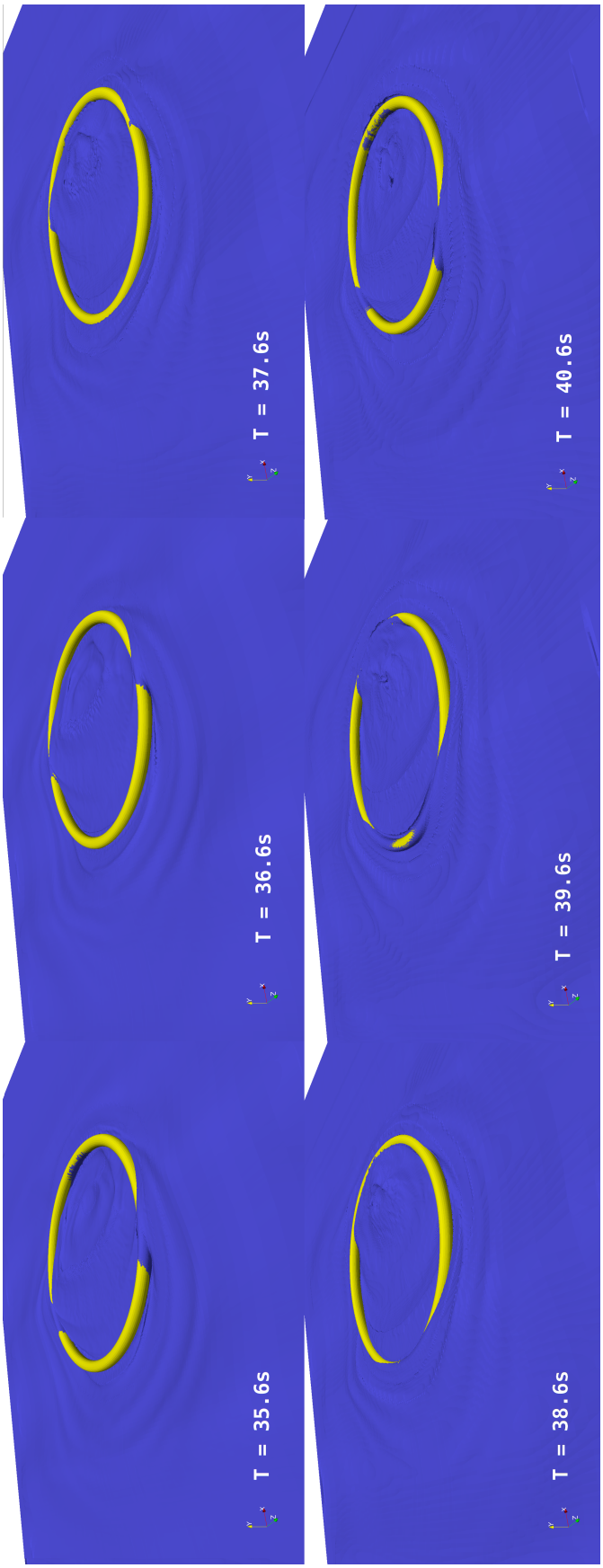


Figure 5.5: Wave and reflection development from t = 35.6s to t = 40.6s.

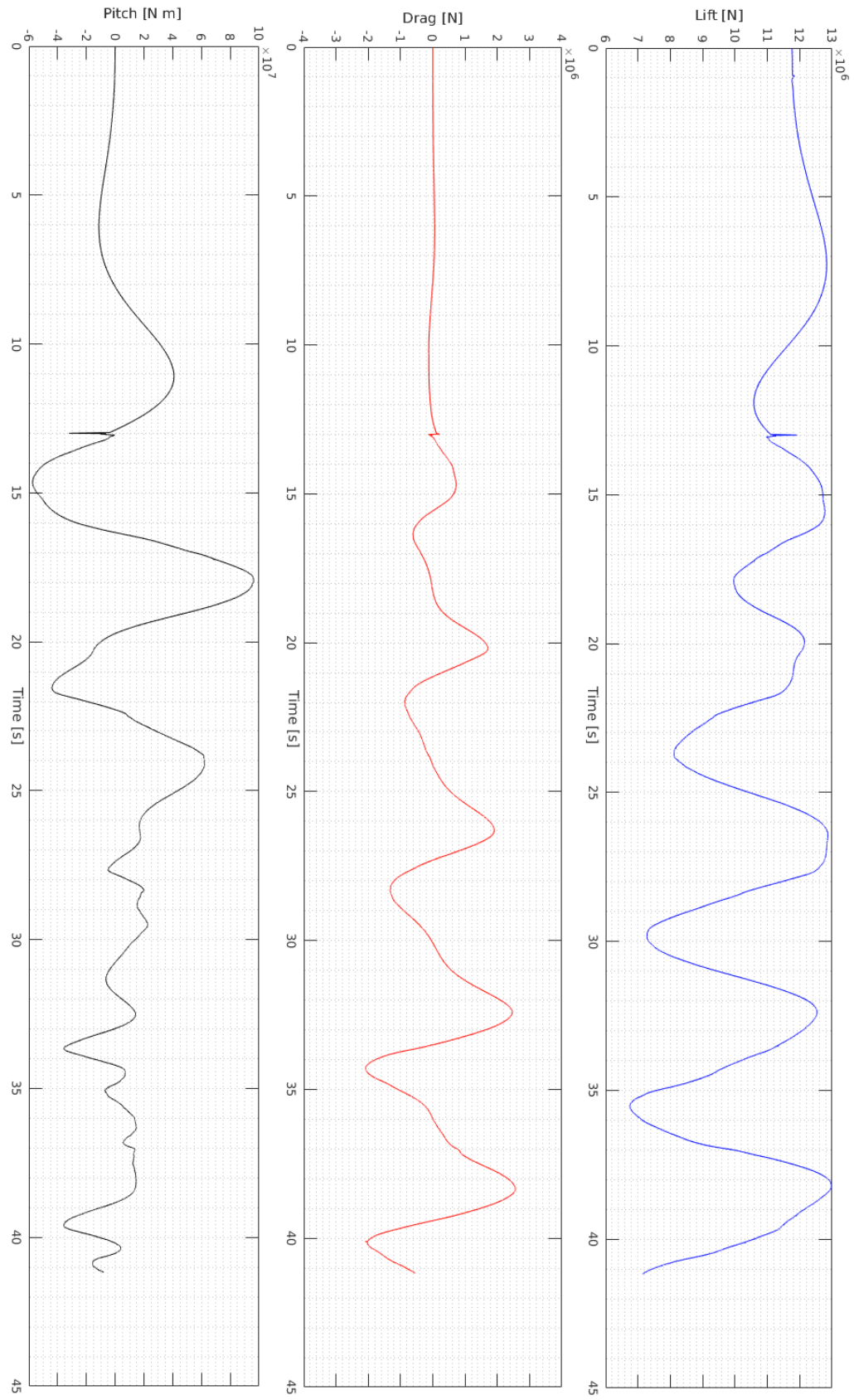


Figure 5.6: Resulting forces in drag and heave, as well as resulting moment in pitch.

Chapter 6

Summary and Discussion

In the present chapters a validation procedure has been followed to investigate how well OpenFOAM 2.4.0 and the additional wave generation and absorption package waves2Foam can deal with numerically difficult phenomena such as an induced motion of waves and moving objects in a multiphase domain. For simplification, viscous effects along all boundaries were neglected in this work.

As can be seen in Section 4.2, the vertical force from waves of relative amplitude $a' = 1$ were captured quite accurately by a simple 2D-domain made using `blockMesh`. When refining the mesh in the region around the cylinder using a combination of `blockMesh` and `snappyHexMesh` in Section 4.4.2, an even closer resemblance was seen between experimental and numerical results. As to be expected when replicating physical experiments numerically there are some minor differences, however, the positive amplitude and trend seem to be correct. The negative amplitude seems to show a larger deviation from Dixon et al. (1979)'s experiment than the positive one, but as the total accuracy was reasonably high, it was decided that the wave generation of waves2Foam performed well enough to use it on the torus in three-dimensions.

Numerical heave tests performed in 2D were described in Section 4.3. In this section numerical results were compared to experimental results from Ito (1977), to see how OpenFOAM handled dynamic meshes and captured the effects of added mass. Although the numerical period nearly fitted perfectly, the amplitude showed some distance to the experimental results, mainly in its positive peaks. As the difference in period was not great and amplitude below water surface was very similar to the experimental one, the setup was still considered sufficiently precise to use it in 3D for this thesis.

One thing to note in the numerical heave tests is that the lowest resolution plot seemed to take longer than the finer ones reaching equilibrium around the water surface. When using the same mass and densities as the more refined simulations, this behaviour is assumed

to arise because the less refined setup struggles defining the exact free surface level. As can be seen in Figure 2.3 in Section 2.1.1, the interface between water and air can be quite diffuse. This diffusivity can be greatly reduced as done in this case by increasing the number of cells in the region of the interface, obviously to the cost of increased computational time. Another option to minimise the thickness of this diffuse layer is to adjust the `cAlpha` value found in the `system/fvSolution` file in the case directory. This is a factor used to control the compression of the interface, where 0 corresponds to no compression, 1 is the standard conservative compression and values larger than 1 will enhance the compression of the interface. However, as a detailed study of adjusting `cAlpha` is necessary to find its effects on both accuracy and stability, the standard value of `cAlpha` = 1 was chosen for continued use, together with a mesh with cell size 20 cpcd.

After the validation in 2D was finished, a three-dimensional simulation of waves on a fixed torus was performed in Chapter 5. After about 13s the length of the time-steps started to rapidly decrease to time steps smaller than $10^{-20}s$. By decreasing `deltaT` manually in `controlDict` (the governing file for control of OpenFOAM simulations) from 10^{-3}s to 10^{-5}s, this error seemed to be resolved and the simulation continued until about $t = 41s$, almost 7 wave periods. Although no error messages appeared after 13s, the large decrease in time-steps and sudden peak in pressure gives hints that there were some underlying problems during this time. However, as manually decreasing the time-steps alone made the situation solvable, it is assumed that the results are more or less correct. By visually inspecting the total run in paraVIEW, no unphysical events were spotted. A snippet of the simulation from t = 35.6s to t= 40.6s is shown in Figure 5.5.

Considering the fact that nothing but fluids were motion in the simulation, it is expected that the issue was related to calculations of the free water surface. Alternative ways to avoid or fix the issue could therefore have been to modify the solvers used in `fvSolution`, so that a larger number of `MULES`- or `nAlpha`-correctors were used to correct the position of the water surface.

Assuming the results from the simulation can still be trusted, the plot in Figure 5.6 is giving away some very interesting information. Ignoring the peak in lift-forces at t = 13s, a period starting around the third amplitude at t = 20s is showing a great resemblance with the shape of the measurement from forces on a cylinder in section 4.2.2. Although it is not enough to verify the results, it is an indication that the setup might be correct and that further investigation of forces on cylinders at different angles should be performed. Additionally it seems like the lift-forces on the torus started oscillating around a lower mid-level after the waves had reached it, meaning that the mean sea level probably had decreased. This theory is verified by investigating the surface elevation plotted in Figure 5.2, measured 15m in front of the torus, where a slightly

lowered mean sea level can be observed.

The shape from the plot of the drag forces looks quite like expected, with forces gradually building up as the wave is building up by the front end of the torus. The relationship between waves building up and drag forces increasing is also shown by the fact that the lift forces are at their maximum around the same time as the drag forces are.

The results for measured momentum in pitch are a bit more difficult to describe, but the plotted momentum seems to stabilise at a more "messy" frequency after the first few wave periods have passed. Looking at the interfering waves within the torus in Figure 5.6, the shorter periods and smaller amplitudes in pitch-momentum is making sense, as no large momentum was building up among all the smaller waves. It should also be noted that a breaking wave was discovered at the centre-back of the torus, seen in Figure 5.6 at time 39.6s. This is a phenomena of great importance, as the amplitude of the wave seems to be of as big amplitude as the waves outside the torus. If the torus is placed to defend an inner region against waves, a breaking wave within it will be critical.

Simultaneously as waves on a fixed torus were ran in the HPC-cluster, a heave test of a floating torus was attempted. However, as the torus either crashed or started showing unphysical behaviour during the simulations, no useful results could be retrieved. The entire section describing the heave simulations on a torus and discussion of what went wrong is therefore moved to Appendix B.

6.1 Further Work

- Obviously, the first priority of continued work would be to solve the problems described in Appendix B and perform a heave test of a freely floating torus. This would provide valuable information about the heave period of the torus and motions of waves created within it.

- During the work on this thesis there was not enough time to produce a complete validation of the waves2Foam package for different wave heights and periods, accompanied with wave loadings on differently shaped geometries. A thorough validation such as that would be very beneficial for waves2Foam to become a generally trusted wave generation tool, reducing the number of validation procedures needed to be performed by the user before measuring wave loads on 3D-structures.

- After simulations of a fixed torus in waves and heave of a freely floating torus a natural next step is to perform simulations of waves on a floating, but moored torus. The steps

performed in this thesis are merely preceding, validating steps to learn more about the forces imposed on a torus, whereas simulations of a moored floating torus will be directly applicable for industries interested in such a design.

- Waves2Foam has a built-in option to implement circular relaxation zones. For a torus and other similar geometries, a circular domain would be a great fit, since the cells in each corner of the domain can be removed. However, when the domain is as heavily graded as the 3D-cases in this thesis, the number of cells removed will be moderate. A more important, but less obvious advantage is that the entire domain could be made circular using `blockMesh`, both simplifying the setup and imposing a mesh where almost all cells are of the same geometry, leading away from the torus. On a general basis, such a mesh made out of similar, hexahedral cells will yield a higher computational efficiency and less probability of error than an unstructured mesh combining cells of different sizes and shapes.

- Yet another option for possibly higher computational efficiency and more accurate solutions in the heave tests would be to use an *overset-grid*, rather than a regular dynamic mesh to cope with the movements of the model. Using overset-grids is a way to avoid deformation of cells when objects are moving, by using multiple layers of mesh. The near-body meshes will move with the body, while the off-body, background meshes shaping the domain are stationary. The overset-grid method is described by amongst others Chandar (2015), where a heave test comparing with Ito (1977) is performed. Another example of showing the potential of using overset-grids is attached in Appendix C, in the form of an attempted simulation of forced oscillation of a cylinder. Due to time-limitations, an effort was not made to implement overset-grids in this thesis.

Chapter 7

Conclusion

The work in this thesis was performed with the intention of first developing and validating a numerical wave tank, using the open-source CFD-software OpenFOAM. It was shown that the additional wave generation tool waves2Foam provided a sufficiently accurate simulation of waves and that OpenFOAM calculated the vertical forces from waves on a cylinder quite well.

Another validation run of a heave test of a cylinder showed similarly promising results, although a slightly increased amplitude above surface level was observed. The reason for this was not investigated directly, nor found. The conclusion was nonetheless that OpenFOAM and waves2FOAM were sufficiently accurate for usage in simulations with a three-dimensional structure in this thesis.

When performing simulations in 3D, a full-scale numerical wave tank was constructed around a large diameter torus, from which forces were measured and interfering waves observed. The forces on the torus in drag had a visible periodic curve, illustrating the periodic nature of the waves. Lift forces on the torus showed a less periodic solution with an initially close resemblance in shape to forces from waves on a cylinder. However, after a few periods of waves had passed the torus, the force-plot looked more periodic, although it was now oscillating around a lower middle point, indicating that the mean surface level was lower than before.

In the momentum-plot and pictures from the simulation the interference between waves and reflected waves within the torus were clearly visible. Another important observation from these results is a breaking wave within the torus, made from reflected waves. Before building a large, wave sensitive structure using a torus such as this, the reflected breaking wave is a phenomena of great importance, needing to be further investigated.

Bibliography

Amazon. EC2, HPC-service from Amazon Web Services: `https://aws.amazon.com/ec2`.

Anderson Jr., J. D. (1995). *Computational Fluid Dynamics, The basics with applications*. McGraw-Hill. ISBN: 978-0070016859.

Berberović, E., van Hinsberg, N. P., Jakirlić, S., Roisman, I. V., and Tropea, C. (2009). Drop impact onto a liquid layer of finite thickness: Dynamics of the cavity evolution. *PHYSICAL REVIEW*, 79.

Bruinsma, N. (2016). Validation and application of a fully nonlinear numerical wave tank. Master's thesis, Delft University of Technology.

CFD-Online. Online forum found at: `http://www.cfd-online.com/Forums/openfoam/`.

Chandar, D. (2015). Seamless integration of an overset grid framework for openfoam: The opera library. *OpenFoam User conference*, (3).

Dias, F. and Bridges, T. J. (2006). The numerical computation of freely propagating time-dependent irrotational water waves. *The Japan Society of Fluid Mechanics and Elsevier B.V*, pages 803–829.

Dixon, A. G., Greated, A. C., and Salter, S. H. (1979). Wave forces on partially submerged cylinders. *Journal of Waterway, Port, coastal and Ocean Division*, 105(4):421–438.

Fenton, J. D. (1985). A fifth-order stokes theory for steady waves. *J. Waterway Port Coastal and Ocean Engineering*, (111):216–234.

Finnemore, E. J. and Franzini, J. B. (2002). *Fluid Mechanics with Engineering Applications*. McGraw-Hill, 10th edition. ISBN: 978-0-07-112196-5.

FreeCAD. *FreeCAD, A MANUAL*. Written by Yorik Van Havre and the FreeCAD community. Software and manual can be found at `http://www.freecadweb.org/`.

Hirt, C. W. and Nichols, B. D. (1981). Volume of fluid (vof) method for the dynamics of free boundaries. *Journal of computational physics*, 39:201–225.

Ito, S. (1977). Study of the transient heave oscillation of a floating cylinder. Master's thesis, Massachusetts institute of technology.

Jacobsen, N. G., Fuhrman, D. R., and Fredsøe, J. (2012). A Wave Generation Toolbox for the Open-Source CFD Library: OpenFoam® . *Int. J. Numerl. Meth. Fluids*, 70(9):1073–1088.

Kitware (2016). *The ParaView Guide, Community Edition*. Written by Utkarsh Ayachit. Software and manual can be found at `http://www.paraview.org/`.

MathWorks (2016). Matlab. version 9.1.0.441655 (R2016b). The MathWorks Inc., Natick, Massachusetts.

OpenCFD. Version 2.4.0, `http://www.openfoam.com/`.

Programmer's. Guide, openfoam. Written by Christopher J. Greenshields, CFD Direct Ltd. Software and manual found at `http://openfoam.org/`.

Skjelbreia, L. and Hendrickson, J. (1960). Fifth order gravity wave theory. *Coastal Engineering Proceedings*, 1(7):184–196.

Users. Guide, openfoam. Written by Christopher J. Greenshields, CFD Direct Ltd. Software and manual found at `http://openfoam.org/`.

Westphalen, J., Greaves, D. M., Williams, C. K., Taylor, P. H., Causon, D. M., Mingham, C. G., Hu, Z. Z., Stansby, P. K., Rogers, B. D., and Omidvar, P. (2009). Extreme wave loading on offshore wave energy devices using cfd: a hierarchical team approach. *Proceedings of the 8th European Wave and Tidal Energy Conference, Uppsala, Sweden, 2009*.

Yu, Y. and Ursell, F. (1961). Surface waves generated by an oscillating circular cylinder on water of finite depth: theory and experiment. *Journal of fluid mechanics*, 11(4):529–.

Appendix A

Summarised Information

A.1 Constants

g	Acceleration of gravity in z-direction	-9.81 m/s²
d	Basin depth	
L	Basin length	
D	Diameter of cylinder or cylindrical torus section	
r	Radius of cylindrical torus section	
R	Larger radius of torus	
T	Period	
α	Water volume fraction	
H	Wave height	
a	Wave amplitude	
ω	Angular wave frequency	
λ	Wave length	
η	Surface elevation	
θ	Inital wave phase	
k	Wave number	
∇	$i\frac{d}{dx} + j\frac{d}{dy} + k\frac{d}{dz}$	
ρ	Density	
μ	Dynamic viscosity	
p	Pressure	
ϕ	Velocity potential function	
u, v, w	Velocity components in x-, y- and z-direction	
I_x, I_y, I_z	Moment in x-, y- and z-direction	
M	Mass	
L_R	Length scale ratio	
$\forall$	Volume	

A.2 Acronyms

CFD	Computational Fluid Dynamics
CAD	Computer-Aided Design
OpenFOAM	Open Source Field Operation and Manipulation
HPC	High-Performance Computing
EC2	Elastic Compute Cloud
FVM	Finite Volume Method
VOF	Volume of Fluid Method
DyM	Dynamic Mesh
cpcd	Cells Per Cylinder Diameter
MULES	Multi-Dimensional Limiter for Explicit Solution (part of VOF)

A.3 Superscripts

$'$	Relative value

Appendix B

Heave Test of Torus

Even though a 222m long domain seemed long enough for the two-dimensional tests on a cylinder in Section 4.3, interactions between waves within the torus and their effect on it were of great interest. Seeing as this area was close to 70m in both x- and z-direction it was decided to increase the length of the domain to 888m to ensure more time for the motions of the torus to settle. Because of the symmetrical nature of the torus the `blockMesh` from Section 5.1 could simply be changed to a square in the xz-plane. Again a coarsely graded domain was created using `blockMesh`, before a cylindrical region surrounding the torus was refined. In the end the shape of the torus was hollowed out of the mesh using `snappyHexMesh`, while also adding an extra level of refinement surrounding the torus and pool within it, using the same methods as for the fixed torus in Chapter 5.

Still using the same initial elevation $d' = D/6 = 0.61667m$ as in Section 4.3, the torus's displacement, forces and moments were planned to be measured over time. After a failed attempt on a mesh with 6 million cells and a highly refined water surface, a coarser grid was made to search for errors in the setup. It was then attempted to run the case both with the `waveDyMFoam`-solver with numerical beaches on all four sides and with the `interDyMFoam`-solver using *wall*-patches and *slip*-condition, without any luck.

The coarse mesh is shown below in pictures B.1 and B.2, with the number of cells for each mesh modification shown in Table B.1. The `checkMesh` utility provided with OpenFOAM yielded no errors or warnings, had a max non-orthogonality of 69 and max skewness of 1.5.

One of the attempts to troubleshoot was a simulation where the torus started halfway submerged, at initial elevation $d' = 0$. This test was performed to see if the torus stabilised around the free surface level, without any unexpected movements or forces. The displacement of the body over time is shown in Figure B.3 where the torus is shown to be sinking for the first few seconds. Unfortunately, the results from the middle of the run were overwritten during a reini-

	Number of cells
`blockMesh`	282 240
`RefineMesh` once	655 704
`snappyHexMesh`	1 887 816

Table B.1: Cell numbers for each level of mesh modification and refinement

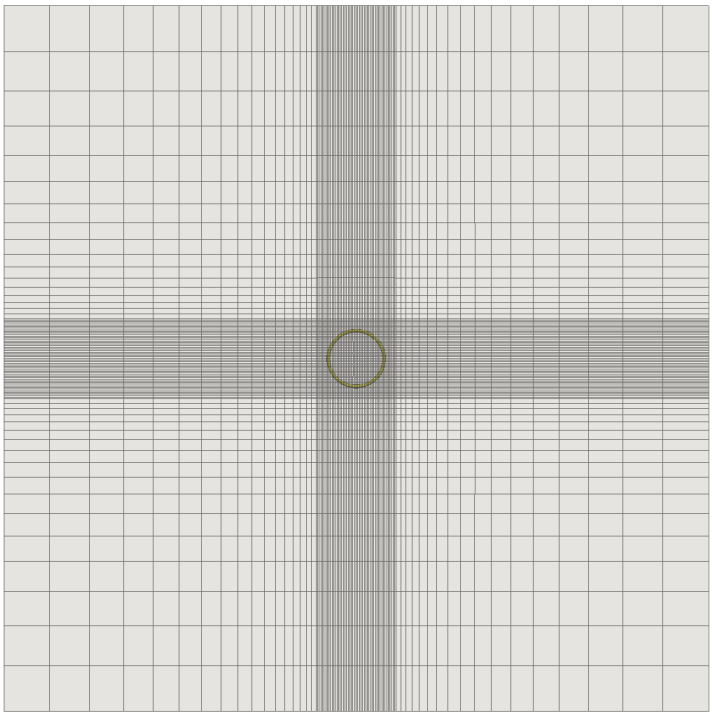


Figure B.1: View from above showing the very coarse grid and grading in x- and z-direction

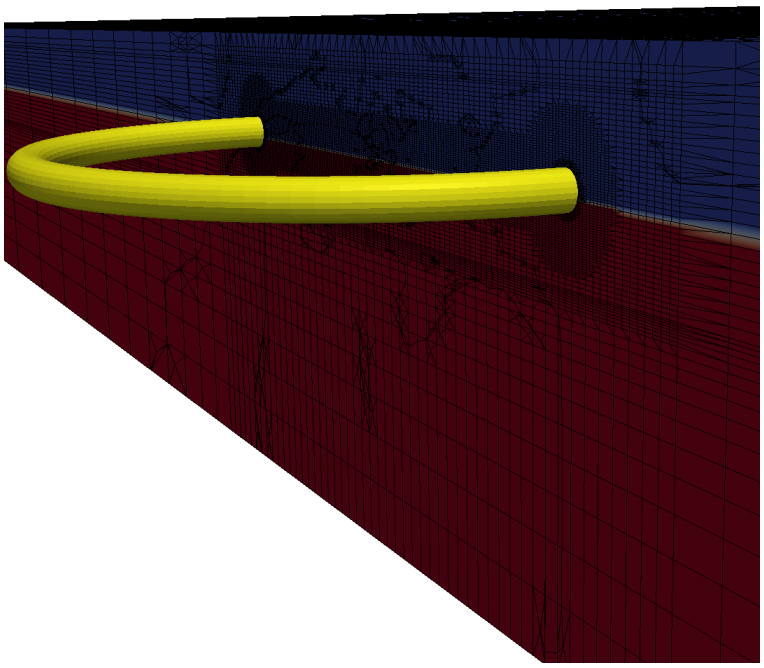


Figure B.2: View showing the coarse outer grading, with a variable water level depending on the cell size in y-direction.

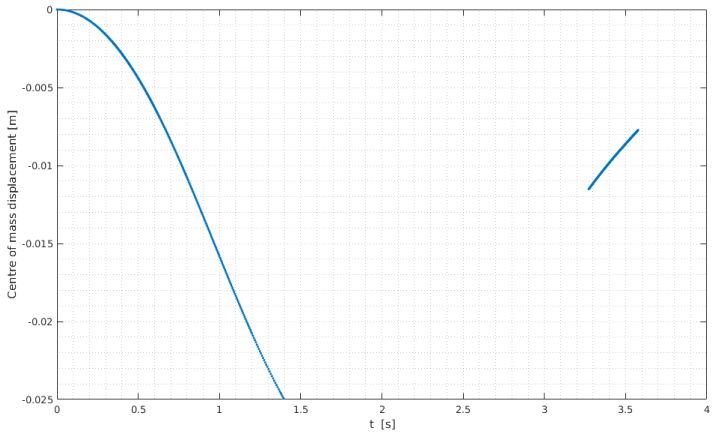


Figure B.3: Plot showing the centre of mass diplacement pr time for the free floating torus. The measurement in the middle was overwritten and is therefore not shown.

tialisation of the simulation, but the torus stopped descending after a few seconds, before rising again as indicated in the figure. When further investigating the coarse grid in ParaVIEW, shown in Figure B.2, the large cells in vertical direction is a probable cause of this displacement, as the water itself will take quite a while to settle. It did however not crash as the simulations with an initial displacement of 0.61667m did.

Other attempts were setting `deltaT` down to vales as low as 10^{-8}s, increasing both `nAlphaCorrectors` and the `PIMPLE`-solvers `nOuterCorrectors` to numbers as high as 4 and trying both `smoothSolver` and `PBiCG`-solver for velocity and α calculation. None of the attempts managed to get the simulation to run for more than approximately one second. Unfortunately, these setbacks led to the thesis being finished before a solution could be found. Given more time, the next attempt would be to refine the free water surface for the very coarse grid, so that no initial movement of fluids occurred. Then a step-wise increase in initial elevation of the torus would be performed, hopefully resulting in the discovery of what went wrong.

Appendix C

Forced Vertical Movement of Cylinder

Another way to verify that the VOF-solver of OpenFOAM correctly describes fluid-object interaction is to force the movement of the cylinder vertically, so that it oscillates around the free surface level. This was attempted as an additional verification method to be used in Chapter 4, but because of incorrect results the entire section was moved here.

Forced, oscillating movement has previously been investigated experimentally by Yu and Ursell (1961). Half of their experiments were done on a cylinder similar to the one used by Ito, described in Section 4.3. The cylinder, which also was chosen to be compared with in this section, has a diameter of 0.1524m (6 inches). After also picking the experimental results from tests with a depth of 0.58m to compare with, the remaining parameters are presented in Table C.1. To compare with the experimental results it was decided to study the amplitude of the waves when the waves reached a stable height, to see how close it was to the relative amplitude R_A measured by Yu and Ursell (1961). In that report, R_A is defined as:

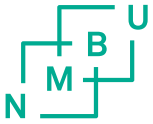
$$R_A = \frac{\text{Stabilised wave amplitude}}{\text{Amplitude of oscillating cylinder}} \tag{C.1}$$

	Yu & Ursell
Diameter	0.1524m
Amplitude	0.006150m
Angular frequency ω	6.830 rad/s
Wave tank length	30m
Wave tank depth	0.58m
Relative amplitude R_A	0.451

Table C.1: Table showing the numerical test parameters compared to those of the experiment by Yu and Ursell (1961)

After multiple attempts with refined meshes, ramped up and/or reduced angular velocities, it was concluded that the dynamic meshing methods known to the author could not provide a sufficiently accurate simulation. Using forced movement in a mesh, rather than a freely moving, finite dynamic mesh region as in Section 4.3, the grid in large parts of the domain has to be able to move for the cylinder to oscillate. This movement leads to changes of the free water surface, as the cells around the water surface are moving vertically with the cylinder. In turn, this results in either disturbances or increased amplitudes of the waves created, making a comparison with experimental results difficult.

When the entire mesh moves, it also greatly increases the computational time used for each time-step. A direct move from the 2D-setup to 3D would therefore lead to a extremely computationally expensive simulation. The overset-grid method shortly described in Section 6.1 is expected to be able to solve these problems, as a static background mesh with wet cells would avoid the problem of an unstable water level. However as forced oscillation of a torus in 3D was not the goal of this thesis, implementing an overset-grid to add an additional verification step in 2D was not prioritised with the time available.



Norges miljø- og biovitenskapelig universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway