NTNU
Norwegian University of Science and Technology
Department of Marine Technology

M.Sc.THESIS

Title:	**Delivered:**
Numerical investigations of Piston mode resonance in a moonpool using OpenFOAM	14.06.2010
	Availability: Open
Student: Jørgen Andreas Alsgaard	**Number of pages:** 68

Abstract:

Moonpool dynamics is a complex phenomenon, and is part of a series of problems referred to as gap problems involving entrapped fluid in a small gap. The fluid in these gaps has an infinite number of resonance modes; a piston mode resonance in heave, and an infinite number of sloshing modes. The piston mode is often the most troublesome, and must be considered when designing a moonpool.

A numerical verification is done on an experiment of a 2D section of a moonpool in shallow water. The numerical tool used will be OpenFOAM (http://www.openfoam.com/) by OpenCFD. The solver library interDyMFoam with dynamic meshing and free surface modeling using a Volume-Of-Fluid method will be used.

Four different configurations of the moonpool will be assessed numerically, but only two will be compared to experimental data. Of these four design alternatives, it will be seen if OpenFoam accurately can describe the physics involved.

Before assessing the moonpool case, a verification of a simpler case is done with the NACA0012 free surface experiment by Duncan (1983). This is done with interFoam.

Good quantitative agreements were found in all cases; the resonance frequency was found in all cases with 3 digit accuracy. Qualitative agreements were only acceptable in some cases, and 10-20% deficiency was experienced. A boundary conforming grid made 0-2% errors at medium mesh, but was unphysical when refined and was not used.

A huge problem was encountered, and not figured out: solutions were often worse instead of better when refining the grid. Possible explanations include numerical diffusion of the indicator variable α, insufficient numerically dissipation in the damping zone, reflection energy into the system and poor boundary layer conformity, with maximum of 8 elements within the boundary.

Keyword:

CFD, VOF, moonpool

Advisor:

Håvard Holm

Address:
NTNU
Department of Marine Technology
N-7491 Trondheim

Location
Marinteknisk Senter
O. Nielsens vei 10

Tel. +47 73 595501
Fax +47 73 595697

Numerical investigations of Piston mode resonance in a moonpool using OpenFOAM.

Abstract

Moonpool hydrodynamics is a complex phenomenon, and is part of a series of problems referred to as gap problems involving entrapped fluid in a small gap. The fluid in these gaps has an infinite number of resonance modes; a piston mode resonance in heave, and an infinite number of sloshing modes. The piston mode is often the most troublesome, and must be considered when designing a moonpool.

A numerical verification is done on an experiment of a 2D section of a moonpool in shallow water. The numerical tool used will be OpenFOAM (http://www.openfoam.com/) by OpenCFD. The solver library interDyMFoam with dynamic meshing and free surface modeling using a Volume-Of-Fluid method will be used.

Four different configurations of the moonpool will be assessed numerically, but only two will be compared to experimental data. Of these four design alternatives, it will be seen if OpenFoam accurately can describe the physics involved.

Before assessing the moonpool case, a verification of a simpler case is done with the NACA0012 free surface experiment by Duncan (1983). This is done with interFoam.

Good quantitative agreements were found in all cases; the resonance frequency was found in all cases with 3 digit accuracy. Qualitative agreements were only acceptable in some cases, and $10-20\%$ deficiency was experienced. A boundary conforming grid made $0-2\%$ errors at medium-coarse mesh, but was unphysical when refined.

A huge problem was encountered, and not figured out: solutions were often worse instead of better when refining the grid. Possible explanations include numerical diffusion of the indicator variable α, insufficient numerically dissipation in the damping zone, reflection energy into the system and poor boundary layer conformity, with maximum of 8 elements within the boundary.

Table of Contents

List of abbreviations

CFD	Computational Fluid Dynamics
CG	Conjugate Gradient method
DNS	Direct Numerical Simulation (of Navier Stokes equation)
LES	Large Eddy Simultion
LS	Level Set method
NS	Navier Stoke ’s equation
RANS	Raynolds Averaged Navier Stoke
VOF	Volume of Fluid method

List of commonly used variables

ϕ	Computational Fluid Dynamics
U	Velocity magnitude
ν	Kinematic viscosity
ρ	Density
α	Indicator variable of the Volume of Fluid method $[0 < \alpha < 1]$
v	Velocity vector

other variables then these will be described at mension, and is not referenced often.

Preface

This paper is the Master Thesis for Jørgen Andreas Alsgaard, as part of his Master of Science study at the faculty of Marine Technology at Norwegian University of Technology (NTNU).

First, I would like to address that very little official documentation available on the methods and solvers in OpenFOAM, which is very frustrating. Most of the knowledge on OpenFOAM was gained by doing the tutorials (which are also not very well documented), looking through the source code and searching the opencfd.com forum.

I would also like to mention that I filed an application, which was not granted to use the supercomputer of SINTEF. I already had permission to use the supercomputer of NTNU, but it lacked the OpenFOAM software, so it was of no use. Therefore all work was done on a regular desktop dual-core computer supplied by Hårvard Holm, as well as additional runs on my own laptop at the end of the period when I needed CPU-hours the most. Needless to say, DNS on a laptop and a single desktop is futile for the high demand grid quality for such mesh, and due to insufficient memory, convergence of the solution were not observed at any of the simulations in the present paper.

I would like to thank Håvard Holm for his help as my supervisor, in giving me the topic and the formulation for the thesis. Also, I would like to thank him for advice given throughout the semester regarding the rapport and the simulations. He was helpful whenever I had something I needed.

I would also like to send a huge thanks to Trygve Kristiansen for advice about almost anything regarding the thesis, especially much help was provided to understand the physics of the moonpool. I would also like to thank him for the supplying me with a MATLAB script that picked out average maxima and minima from a timeseries.

A thanks is also due to Anders Östman, for help with my OpenFOAM related question and question to the delivered code. A thanks is also due to Thomas Sauder, for writing the provided code.

Trondheim, 14.June 2010,

Jørgen A. Alsgaard

1. Introduction

To describe the purpose for this thesis, a bigger picture has to be painted first. This chapter first introduces some marine concepts and ways marine engineering has been done over the years in section 1.1. The concept of ship designing will be explained, and how advances with computer technology have revolutionized design is discussed. In light of how commercial industry and non-profit research has been driven and driving each other into new technologic advances, a comparison between the two different approaches to computational fluid dynamics is briefly discussed, in section 1.2. Finally, this current work tied together with what has been presented, so that it can be seen where it contributes to the bigger picture.

1.1. A general introduction to ship design and naval architecture

Like with most engineering, ship design is expensive. It is expensive because the projects are of considerable size, complexity is high and it often involves finding solutions far beyond what already exist. Seldom can one person do the job alone; experts from several fields of engineering are needed to cover the complex domain. Also tolerance for safety is extremely low, as the results of failure usually involve damage to personnel and even fatalities, not to mention loss of equipment for millions of dollars potentially.

1.1.1. A brief overview of the history of naval architecture

Ship industry has always been challenged to produce unique designs in a limited time. Naval architecture was thought of more as a craft than a science, as understanding of physics and analysis of the flow was not pushing the industry. Experience together with some limited model testing was ruling design. The demand for more scientific approach came with industrialization. With little knowledge of flow regimes, experimental results were the only thing to go by. Experience from designed ships was established, previous solutions could be combined into new solutions and shipbuilders started sharing experience. Over time databases of dimensionless empirical knowledge have been gathered, for instance the Taylor Standard Series where a series of tests performed with systematic variations of hull form (Taylor, 1943). Empiric formulas were established that took into account only a handful of important design parameters. Special strategies were established to take advantage of the empiric formulas. Later, these design rules was gathered in national and international rules, such as NORSOK in Norway.

1.2. Computational Fluid Dynamics – an aid in ship design?

In the early stages of design, i.e. in conceptual and preliminary design, a key factor for competiveness is how fast a design can be improved. That is, how to produce an optimized design faster than the competitors. This is dependent on how fast one iteration cycle. One

cycle starts from an initial hull form. The hull geometry is input in numerical flow calculation. Resulting wave loads, response amplitude operators and other load parameters is input to a FEM program to calculate force distribution on the hull. Now, the hull can be optimized to carry these forces better, completing the cycle. Running several cycles leads to, hopefully, an improved product. The trouble is however, that the conditions for the optimum change for each iteration. Since the choice of hull form will change the flow regime outside the boat, this can be regarded as a kind of optimization problem with an extremely complex target function and boundary conditions that are hard to express in a simple mathematical way. It theory there might be a unique optimum, but it is impossible to find in practice. Instead, the goal is to get as close as possible, by 'seemingly' unsystematic testing; where experience is the deciding.

Since decades of experience have been thoroughly incorporated into design rules, nearly complete design can be derived from a few main parameters in a short time (Cramer & Krüger, 1999). Because of this, design rules still is used and is a key value of the shipbuilding industry. Today however, computational methods are receiving more emphasis due to better accuracy and faster turnaround. It is crucial to be sure that CFD design tool lowers design times while also producing designs that are better than by traditional methods.

1.2.1. CFD design tools

For usage of a design tool to be effective, a designer must be able to work directly with decision making, so the design tool should ideally be self-sufficient. So far, the backbone of numerical design tools that are effectively used in industry is based on inviscid non-linear wave theory (Gorski, 2002). These tools come with automatic grid generation and can therefore produce improved design fast. Solvers based on viscous Reynolds-Averaged Navier Stokes (RANS) has yet to become self-sufficient, do to complexities in grid generation in 3D with rudders and propellers complicating geometry combined with free surface. Numerical simulation of free surface turbulent flows is a troublesome affair because of the presence of a material interface, whose location is one of the unknowns of the problem; as a matter of fact, the simulation of this interface is still a numerical research issue (Mascio, Broglia, & Muscari, 2007). Adapting the mesh to the moving free surface, either directly or indirectly, complicate matters of automatic mesh generation considerably. ITTC (1996) concluded that RANS codes had matured enough to be integrated into the design process for addressing issues associated with resistance and propulsion, while maneuvering and seakeeping was not included in this discussion, as the free surface needs to be more accurately described.

According to Cramer et al., a RANS tool could be considered economically useful if one cycle could be preformed within 24 hours (1999, p. 6). Such RANS code do exists. A tool with automatic pre- and post-processing was, with some success run on the wake field on with the stern of a ship (Krüger, Manzke, Rung, & Vorhölter, 2009). Although there was considerable over-prediction of the boundary field around the propeller compared to experimentally measured values, there was otherwise good qualitative agreements. Krüger et al. further claim this tool to be great help in design optimization, with as low as 3 hours runtime for a cycle on a single modern desktop computer. In this case however, simulation of

free surface is input a-priori as with linear potential theory. It is easy to see that while CFD has great potential, it also has challenges ahead.

In this paper, the use of numerical simulations of the free surface oscillations of ship motion will be investigated using OpenFOAM. OpenFOAM is a package of CFD solvers that are free to use, has open source code, is regularly updated and expanded by the community, and can be installed on a personal computer or set up on a network of computers taking advantage of parallel processing.

1.2.2. Numerical methods as a competitor to experimental data

Using a virtual wave tank is both cost cutting and time saving, especially in the early phase when lots of design options have to be filtered out. Time in a towing tank is more expensive than numerical simulations. Both model tests and computations have their advantages and disadvantages. While an experiment is considered more expensive and slow, it is more robust and it does not neglect any physical effects due to inadequate modeling. If fact, if the complexity of the problem grows, the accuracy of experiments remains constant. Some effects might not scale correctly, but can be corrected to full scale based on empiric data, which is often sufficient. Model tests usually provide resistance and powering curves, which are currently difficult for computations to reproduce, as well as providing detailed measurements in critical areas (Gorski, 2002).

For CFD, weaknesses include that either a great deal of simplification model has to be applied on the problem or extensive computational power is needed. Also, development of CFD codes is expensive, so only when a method is standardized is the numerical tank going to be cheaper then experiments. Strengths include providing an overall picture of the flow field early, which can aid in its understanding, as well as an economical means to evaluate and rank order hull modifications (Gorski, 2002). Another advantage is that long term extreme statistics can be generated at sub-speed; for instance 3-h maxima might only need 1 h run on a computer while it will always need full 3 hours in experiments.

1.2.3. Different perspectives of the designer and the researcher

The designer is not really interested in the flow phenomena, which for him is only boundary condition. It is completely opposite for a developer or researcher. A developer/researcher is interested in how well a method can predict the right physical behavior, at the correct location and correct values of forces and velocities. For a researcher, the hull, propeller or rudder is seen simply as input. How a designer and a developer or researcher approach the program differently might seem irrelevant, but since it is developer that makes the tools for the designer to use it is important to remember that they have entirely different goal of the same program. Designers are often needed to make designs for cases a program was not intended for, but none the less, he have to make the design. It is therefore important for researches to find the operational limits of the code, and the designer should

include these limits in the documentation, so that the designer knows what to expect when a tool is applied outside the intended scope of purpose.

The goal of the present work is to see how close CFD might get to the experimental data in a case where free-surface and viscous flow is of great importance. Hence, the approach of this paper will be that of a developer and a researcher, as it is the prediction of phenomena to a great deal of accuracy that is important, rather than what it might be used for. If this is successful, it might be a future expansion to today's design tools.

2. Moonpool designing challenges

So far, only a general discussion on designing issues with help of computational fluid dynamics has been addressed. This section will introduce the concept of a moonpool in section 2.1, and special physical phenomena attached to it. Then the disposition of the rest of the work is presented in section 2.2.

2.1. Moonpool

A moonpool is a feature of marine drilling-rigs and -ships, cable laying and other subsea operation vessels, as well as some marine research and underwater exploration vessels. It is a vertical opening passing through the seawater-air interface providing a sheltered environment to the water inside of the structure. The ideal location of a moonpool is in the center of floatation, where the rotational motions are minimized. This provides a protected environment allowing technicians or researchers to lower fragile instruments and tools into the sea while the ship is exposed to high seas, ice or otherwise unwelcoming conditions on the outside. Introduction of equipment is critical around the free surface, due to the dynamic wave energies exhibit a maxima in the proximity of the surface and with exponential decay with depth. Entry of drilling equipment inside the less weather exposed moonpool is a therefore a good way to precautions of the fragile equipment compared to being launched off-deck. The same goes with divers and small submersible craft to enter or leave the water easily.

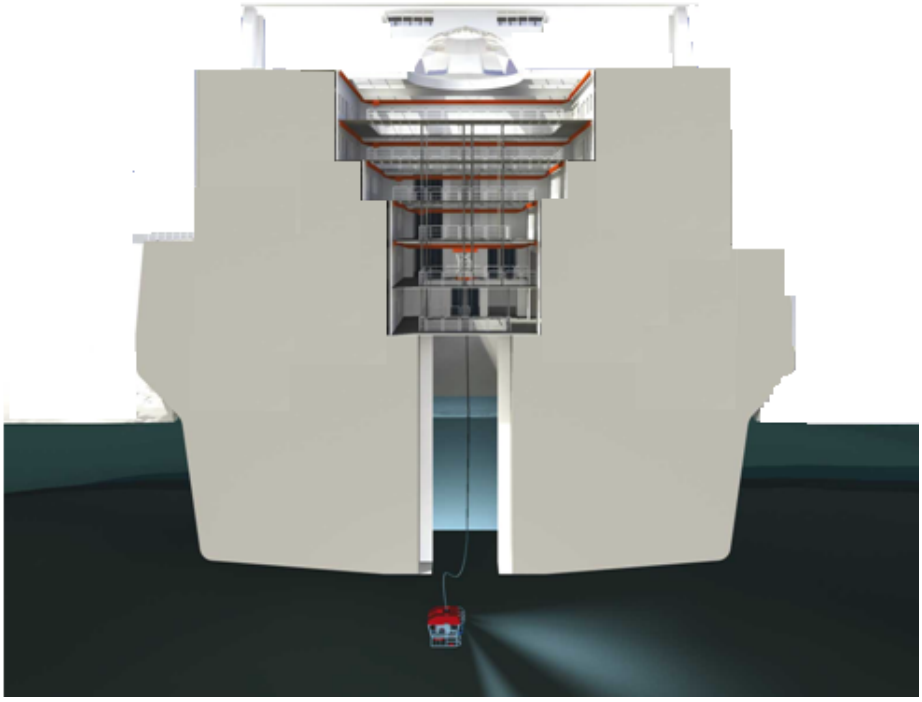


Figure 2-1: A ship with moonpool in the middle of a cross-section.

The moonpool will under normal circumstances deliver shelter, but if excited at resonance oscillations inside the moonpool may give larger waves than outside. Two types of motion occur, large vertical 'piston-mode' heave and sloshing modes in both directions in the water plane. At extreme cases the motion inside the moonpool may be up to ten times that of the outside. This dynamic magnification can cause slamming on diving bells or ROV that are launched, green water over the edge of the moonpool which can be dangerous for the crew, or can drastically increase the resistance of the vessel in transit conditions (Sadiq, Zhuang, & Liang, 2006). Piston mode is usually the most troublesome of the resonance modes and its effects most be taken into account in the design process (Molin, 2000). Therefore, this paper will focus on the piston-mode resonance.

Resonance in the moonpool can be excited through different mechanisms; incoming waves, through current, by forward speed or by motions of the vessel. It is found that the natural frequency of water inside the moonpool does not change with a change in the velocity rather it is dependent upon the physical properties of the fluid medium, shape and size of moonpool and the draught of water inside the moonpool (Faltinsen, Rognebakke, & Timokha, 2006). For these reasons forced heave will be the chosen approach to achieve resonance in the present study. The moonpool will be restricted from movement in any other degree of freedom in both experiments and in the numerical studies. This will limit the excitation of sloshing, which is outside the scope of this validation study.

2.2. Present work

Despite the research effort conducted on moonpool oscillation, most of the solutions were found and are still found and validated experimentally (Gaillarde & Cotteleer, 2005). As stated by Mascio et al. (2007) "The numerical simulation of free surface turbulent flows is a goal of paramount importance for the design and the optimization of ship hulls. Particular to this field of research is the presence of a material interface, whose location is one of the unknowns of the problem; as a matter of fact, the simulation of this interface is still a numerical research issue."

In the present paper several design of moonpools proposed by SINTEF are tested out experimentally, and validation is carried out by numerical simulation. In specific, the optimal shape of a moonpool to reduce the piston-like heave in resonance is searched for from a set of different design solutions presented a-priori (see Figure 6-2).The long term goal of the present work is to make numerical simulations as valid as experimental data when ranking design options, and make CFD the commercial de-facto standard in the early design phase, even for free surface turbulent flows.

2.2.1. Scope of the current work

The objectives of this work is (*i*) describe in detail VOF method as it is available in the general purpose open source libraries OpenFOAM for incompressible turbulent two-phase flow (*ii*) validation of the numerical results obtained for the free-surface with this approach in

a two dimensional study of NACA sample foil in a non-breaking case, and (*iii*) validation of piston-mode resonance for moonpool with four different configurations in two dimensional flow and comparison between numerical and experimental data.

2.2.2. Structure

The paper is organized as follows:

Chapter 1 is broad introduction to the field of designing in marine technology to set the scene, and paint a bigger picture. A discussion is made between experimental and numerical approach to design.

Chapter 2 give a introduction to this study, stating what should be done and how it should be done

Chapter 3 present the background theory of previous studies on similar topic, which are relevant for the further work.

Chapter 4 give an overview of on the physics involved, and present the governing equations that should be solved.

In Chapter 5 the application library OpenFOAM is presented. How discretization is done in OpenFOAM and choice of numerical scheme is presented, and strengths and weaknesses of choices are discussed.

Chapter 6 presents the experiment that were done in conjunction to this study. The setup is covered extensively. Some observations made during the experiment is discussed, which later will have an influence when comparing to numerical simulations.

Chapter 7 presents the numerical results, starting with a validation study on a NACA foil. Then all moonpool configurations is covered, and all relevant findings are discussed. A comparison analysis between experimental and numerical results completes the chapter, with focus on describing the discrepancy between results.

Finally, Chapter 8 concludes the work done in this Master's Thesis and point out topics for further investigation before given a closing summary.

Appendix A evaluate linear potential expressions of the moonpool resonance, while Appendix B gives a clip-book demonstration of a vortex shedding. Appendix C is the time series results from selected simulations.

3. Review of related research

This chapter gives a brief overview of research done in related areas, relevant to this study. First a summary of gap problems is given, before the use of numerical methods in similar problems is covered. Special interest is given to the Volume of fluid method, as it is implemented in the OpenFOAM code.

3.1. Gap resonance problems

Gap problems refer to a class of hydrodynamic problems where a semi-entrapped fluid produces a piston-like excitation in resonance. Gap problems include the analysis of flow in moonpool, between ships in side-by-side configuration, between a ship and an offshore loading terminal and flow between the hulls in a multihull configuration. The complex hydrodynamics of two floating bodies in close proximity has been a research topic for over two decades (Pauw, Huijsmans, & Voogt, 2007).

Prediction of forces and responses induced on floating bodies in waves is usually based on linear potential flow theory which is quite adequate for most wave frequency responses (Newman J. , 1977). However certain types of motion for which the linear potential flow damping is small can exhibit a strong resonant response to incident waves of an appropriate frequency. In these cases it is found that potential flow alone seriously over-predicts the response which is limited by further non-linear effects (Salvesen, Tuck, & Faltinsen, 1970). As linear theory only include linear wave diffraction as damping two major effects of damping is neglected: 1) Non-linear effects in the free-surface waves, such as breaking and 2) viscous effects. The major source of non-linear damping is due to separation of the flow at bilges of the keel leading to vortex shedding (Graham & Kendon, 2008). The resonance problem has been thoroughly treated within the framework of the linear potential theory of waves (Molin, 2001), (McIver P. , 2005), (Kuznetsov, Maz'ya, & Vainberg, Linear Water Waves. A Mathematical approach, 2002).

Chen (2005) presented a new lid method diffraction codes to suppress non-realistic high wave elevations between the two floating objects with use of linear diffraction codes. This method was verified on an experiment by Pauw et al. (2007) with two moored LNG tankers in a side-by-side, but showed it was difficult to obtain one unique value of the lid damping, leaving the method useless as help in a design process.

With regards to resonance behavior specific to moonpool, several studies have been conducted. Maisondieu et al. (2001) discussed experimental and numerical assessment in dynamics of moonpool. Some useful results are available in Molin (2001) and Tuck & Newman (2002), but focus is mainly on sloshing and using potential theory.

The possible existence of trapping modes and complex resonances for two-dimensional steady-state waves around two symmetric bodies in finite water depth was investigated by, Kuznetsov et al. (2001) and McIver (1996), (2005). However, the existence of trapped modes is a rare exception for actual moonpools including, most probably, two equal-sized free-

surface-piercing horizontal rectangular cylinders (Faltinsen, Rognebakke, & Timokha, 2006). Further references as well as computational results are available in Drobyshevski (2004).

3.2. Two-phase free surface modeling in CFD

In this study a moonpool in forced heave will be examined, where viscous wave damping is particularly important. An accurate model of the two-phase free-surface turbulent flow is essential to predict the resonance correctly. The problem is extendable to any form of floating structure where damping is significant. Many efforts are dedicated to describing free-surface flow with turbulence accurately. Starting with the conference in Gothenburg (Larsson, Bertram, & Stern, 2000) several marine hydrodynamic workshops have been held with focus on better accuracy of the numerical models used to simulate free-surface flow in CFD codes.

The interface between air and water is often called ‘free surface’. The reason for the "free" designation arises from the large difference in the densities of the gas and liquid. The inertia of the air can be ignored compared to that of the liquid. Hence the liquid moves independently, or freely. The only influence of the gas is the pressure it exerts on the liquid surface, which gives rise to surface tension.

This free-surface interface is difficult to resolve because the interface is subject to a number of unique physical phenomena, at scales much smaller than for the single-phase flows. For instance, a major difficulty is caused by the change in interface topology that occurs when a wave breaks. This complicates the physics and sharpens the requirements that a numerical method must satisfy in order to resolve the motion in a satisfactory way (Scardovelli & Zaleski, 1999).

Regardless of the method employed, there are three essential features required to properly model free surfaces according to (Flow Science):

- A scheme is needed to describe the shape and location of a surface,
- An algorithm is required to evolve the shape and location with time, and
- Free-surface boundary conditions must be applied at the surface.

In the following, the types of numerical approaches that have been used to model free surfaces will be briefly reviewed, indicating the advantages and disadvantages of each method.

3.2.1. Surface fitting by surface conforming grid

In the workshop held in Gothenburg (Larsson, Bertram, & Stern, 2000), it can be seen that most of the computations were performed by means of *surface fitting* approaches, i.e. with algorithms where the grid actually moves to fit the current position of the free surface. Although capable for excellent results in ideal cases (Wilson, Paterson, & Stern, 2000), it has limited applicability in practical cases where the surface deforms a lot. This is the case of breaking waves or with a complex hull geometry oscillating through the free surface. In the present study, the geometry is simple and wave breaking should not be dominant. However

moving mesh due to forced heave in addition to grid adapting to the free surface could easily result in too distorted mesh or it would need expensive topology changes.

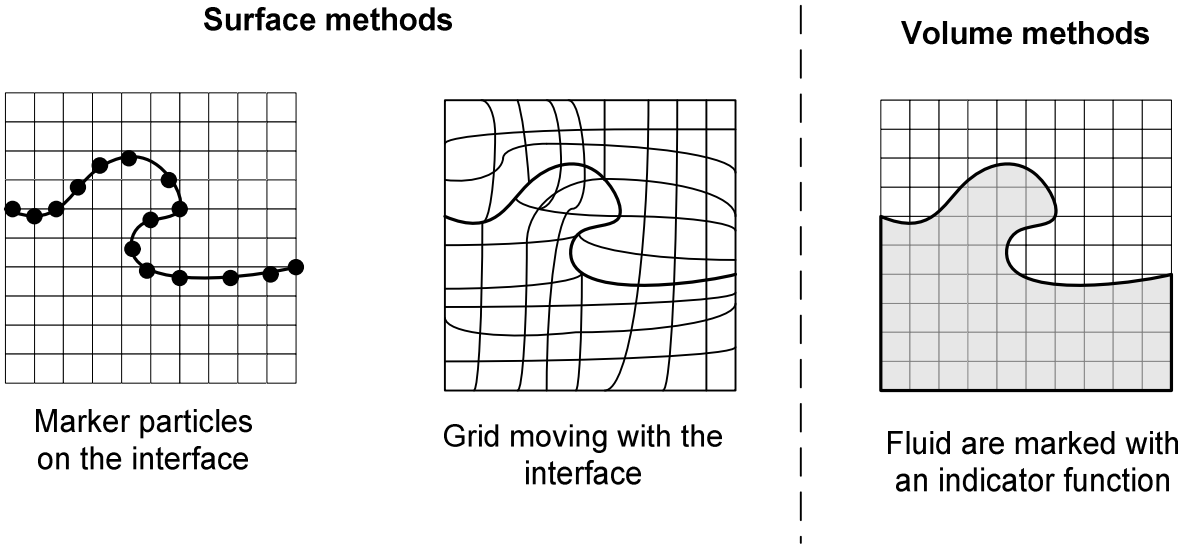


Figure 3-1: Different methods of representing the free surface interface

3.2.2. Monitoring surface with a fixed grid

To avoid the problem of surface conforming grid, the grid could be held fixed while the position of the free surface is monitored separately. Two different solutions exist which both have been improved and tested for numerical use since their introduction. The first technique introduces massless particles, and tracks these particles; hence the set of methods is known as *front-tracking.* The second major technique keeps track the surface through a scalar function, and captures the surface afterwards by using the scalar value. These methods are called *front-capturing*.

3.2.2.1. Surface tracking

The Marker-and-Cell method proposed by Harlow and Welch (1965) was one of the first attempts on front tracking. In this method, massless particles are distributed all over the computational domain, and move with the underlying flow. Afterwards the particles are used to identify each fluid and the interface is reconstructed by following the movement of these particles. More recently, Unverdi and Tryggvason (1992) have proposed a front-tracking method where particles, distributed only over the free surface interface and not the whole domain explicitly track the surface of discontinuity. The multi-phase fluid is then modeled as a single fluid with variable density and viscosity. Front-tracking methods have proved to be both accurate and robust, however show similar lacks as surface conforming grids. In order to prevent the clustering of marker particles, regridding algorithms must be employed;

moreover, special care is required when dealing with an interface that changes in topology, as in the case of a free surface that breaks (Unverdi & Tryggvason, 1992). Also, Marker-And-Cell method is known to have large overhead usage of memory.

3.2.2.2. Surface capture

In front-capturing approaches, no special procedure is required to model interfaces that change in topology, making it more powerful than previously mentioned techniques. Surface capture methods include *Volume-of-Fluid* method developed by Hirt and Nichols often known as the volume of fluid (VOF) method. It is presented by Nichols and Hirt (1975) and Noh and Woodward (1976), and further extended by Hirt and Nichols (1981). Since then, the VOF method has been improved significantly over the years (Rudman, 1997) (Rider & Kothe, 1998) Alternately, *Level Set* method was developed by Sussman, Smekerda and Osher (1994) and has likewise been subject to frequent updates.
Volume-of-Fluid method is a volume fraction function convected by the underlying fluid flow. In one fluid phase the function is set to 1, while 0 in the other phase. In regions where both phases are present, the value of this function represents the fraction of the volume occupied by the first fluid or phase; density and viscosity are then defined as a smooth function of the volume fraction. The major advantages of this technique lie in its mass conservation properties as a direct consequence of the development of an advection algorithm based on a discrete representation of the conservation law, as well as its capability in treating complex interface changes, such as breaking and reconnecting surfaces. To reasons that probably was chosen when OpenFOAM decided to implement VOF was the it relatively simply extends from two-dimensions to three-dimensions and that the scheme used is local, meaning it is relatively simple to implement these algorithms in parallel, in particular within the framework of domain decomposition techniques (Scardovelli & Zaleski, 1999). .

The major disadvantage of the VOF method lie in the difficulty to compute the geometric properties of the interface is stored implicitly in a set of scalars (Scardovelli & Zaleski, 1999). The curvature or the normal vector, which are needed when surface tension effects are taken into account, needs to process the data at the cost of added computational cost. For VOF methods, the interface reconstruction is still a research topic and moreover modeling errors stem from the difficulties to control the interface thickness, because of artificial spreading due to numerical diffusion.

In the Level Set approach, the interface is simulated by the evolution of the zero level of a smooth function, convected by the underlying flow. This approach allows the simulation of complex interfaces, including breaking or merging phenomena. Moreover, the computation of geometrical parameters of the interface is straightforward unlike with Volume-of-Fluid. Nevertheless, in classical Level Set approaches, both air and water are simulated as a single fluid whose properties vary continuously across the free surface hence volume conservation is not guaranteed. In order to get a stable algorithm, the change of fluid properties must be spread over several grid cells. This can be a source of significant loss of accuracy, especially in three-dimensional computations, because of the limited amount of cells at disposal in the

free surface region. Moreover, it has been shown by Sussman et al. (1994) that, even if a conservative algorithm is employed for the evolution of the level set function, this does not imply conservation of volume of each fluid

It is worth to mention that, in order to get both mass conservation properties for VOF and easy computation of geometric properties of the interface from LS, some authors have developed combined LS/VOF methods (see for instance (Sussman, Smekerda, & Osher, 1994). In spite of the above-mentioned problems, the increase in the use of front-capturing methods such as LS or VOF approach in the field of naval hydrodynamics is demonstrated by the number of authors that have used these methodologies to perform the steady and unsteady test cases proposed by the last workshop on CFD in ship hydrodynamics held in Tokyo in

The Level Set method loose volume fracture when a thin interface causes numerical diffraction. The imposition of a volume preservation constraint in the VOF method does not eliminate these errors, but instead changes their symptoms replacing mass loss with inaccurate mass motion leading to small pieces of fluid non-physically being ejected as jetsam or flotsam (Medina, 2008).

Since OpenFOAM only has a VOF method, there was no choice. For this case, a surface adapting grid would probably be a good choice as the surface elevation is only a few centimeter, the surface will not break or reconnect, and mixing of fluid will not happen.

4. Description of the flow problem

In this section, a mathematical representation of the simultaneous flow of two immiscible, incompressible, Newtonian fluids is presented. This mathematical representation includes surface tension, but neglects heat and compressibility of either of the two fluids. Since the volume of fluid method will be used, the continuity equations are given uniformly over the domain, with an indicator α variable correcting for difference in material properties between the phases.

4.1. Governing equations

The volume-of-fluid method is based on writing only a single set of conservation equations for the whole flow field. Let therefore the domain Ω be filled with two immiscible fluid, here it is air Ω_a and water Ω_w. Flow properties **v** and p are continuous throughout the domain, whereas material properties ν and ρ may contain a discontinuity across the interface. To keep track of the surface a volume fraction variable α is introduced. α is an indicator of how much of the current cell is occupied with water. Formally, it can be expressed as,

$$\alpha \begin{cases} 1 & for\ a\ point\ inside\ \Omega_w \\ 0 < \alpha < 1 & for\ a\ point\ in\ transition\ between\ fluids \\ 0 & for\ a\ point\ inside\ \Omega_a \end{cases} \qquad 4.1$$

$$\begin{aligned} \rho(x,t) &= \rho_w \alpha + \rho_a (1-\alpha) \\ \nu(x,t) &= \nu_w \alpha + \nu_a (1-\alpha) \end{aligned} \qquad 4.2$$

ρ_w, ν_w is the density and viscosity of water, while ρ_a, ν_a is that of air.

The general equations of fluid flow are given by the conservation laws of physics. These are represented here in a mathematical formulation, so that it can be discretized and solved numerically. The momentum equilibrium of equation 4.4 is also known as Navier Stokes equation, and can be given in several forms. The current formulation is taken from Ruche (2002). Since water can be considered incompressible, and pressure of air will be around atmospheric pressure, the fluids can safely be modeled as incompressible, giving a simpler expression for mass conservation. Also, conservation of the volume fraction is needed such that no volume fraction disappears thought the surface, this is given by equation 4.5.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \qquad 4.3$$

$$\frac{\partial \rho \mathbf{v}}{\partial t} + \rho \boldsymbol{a}_F + \nabla \cdot (\rho \mathbf{v}\mathbf{v}) = -\nabla p - \mathbf{g} \cdot \mathbf{x} \nabla \rho + (\nabla \cdot (\nu \nabla \mathbf{v}) + (\nabla \mathbf{v}) \cdot \nabla \nu) - \kappa \sigma \nabla \alpha \qquad 4.4$$

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \mathbf{v}) + \nabla \alpha (1 - \alpha) \mathbf{v}_{r\alpha} = 0 \qquad 4.5$$

here

$$\mathbf{v} = \frac{[\alpha \rho_w \mathbf{v}_w + (1 - \alpha) \rho_a \mathbf{v}_a]}{\rho} \qquad 4.6$$

is the mass velocity, $\boldsymbol{a}_f$ is the acceleration due to moving reference frame, κ is the curvature of the interface relative to the normal, σ is the surface tension and $\mathbf{v}_{r\alpha} = \mathbf{v}_w - \mathbf{v}_a$ is the relative velocity between the two phases at the surface. Equation *4.5* comes from Laplacian conservation of the volume fracture plus a last term; an artificial constructed term for compression of the surface. This compression term was introduced by Weller (2002) and presented further in the works of Rusche (2002). Its purpose is to minimize the numerical diffusion of the VOF method without being forced to use a compressive differencing scheme, as needed with for instance the CICSAM technique by Ubbink (1997). Both of these methods were implemented in OpenFOAM, but now only that of Weller remains as it proved to be more efficient.

The chosen compression term chosen originate the most common error in the VOF method: the numerical diffusion at the interface. According to (Medina, 2008) a relation between mass velocity **v** and volumetric velocity $\boldsymbol{\phi}$ can be given as,

$$\boldsymbol{v} = \boldsymbol{\phi} + \alpha(1 - \alpha) \frac{\rho_w - \rho_a}{\rho} \boldsymbol{\phi}_{r\alpha} \qquad 4.7$$

For a thin interface, the relative velocity between water and air at the interface vanishes, $\vec{u}_{r\alpha} \cong 0$. In this case, the phase indicator α changes abruptly from $\alpha = 1$ in the water phase to $\alpha = 0$ in the air phase, and **v** is purely convected to the volumetric flux $\boldsymbol{u}$. This causes spurious current at the interface where no apparent force is present. Several analysis of the spurious current has been conducted, see for instance Harvie et al. (2006).
First-order upwind scheme smears the interface too much and introduces artificial mixing of the two fluids. To avoid the numerical diffusion in the VOF method, one should employ second- and higher-order schemes, which again tend to produce over- and undershoots (Medina, 2008).

4.2. The physical resonance problem described by linear potential theory

If the flow can be assumed inviscid and irrotational the velocity may be expressed as a velocity potential $v = \nabla \phi$. When the velocity potential is inserted into the mass conservation equation 4.3, it is reduced to the Laplace equation, $\nabla^2 \phi = 0$ which is the continuity equation in potential theory. Also, Navier Stoke reduces to Bernoulli's equation,

$$\frac{1}{\rho}p + \frac{\partial \phi}{\partial t} + gz = C \qquad 4.8$$

In linear potential theory, the only form from damping comes from radiation waves, see. In the special case of a moonpool, a trapped mass act as a piston, which create radiation, as seen in *Figure 4-1*. In linear potential theory, the only form from damping comes from radiation waves, see *Figure 4-2*.

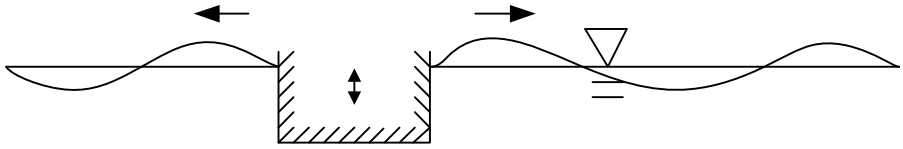


Figure 4-1: The radiation problem in linear potential theory, accounting for damping

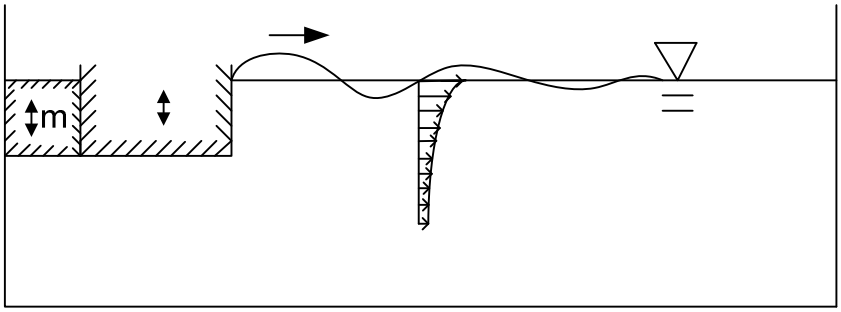


Figure 4-2: Radiation problem in the case of moonpool.

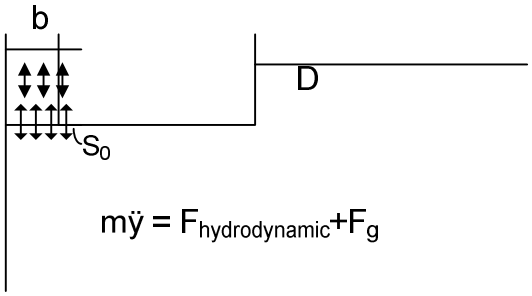


Figure 4-3: The piston-mode formulation of single degree oscillator.

Using Bernoulli and the free surface boundary condition the following expression can be found describing the resonance motion of a moonpool in Figure 4-3,

$$T_n = 2\pi\sqrt{\frac{m + A_{33}}{\rho g D}}\,, \qquad \rho h D\ddot{\eta} + \rho g D\eta = -\rho \int_{x=0}^{x=D} \frac{\partial \phi_R}{\partial t}(z = -h)\, dx$$

See Appendix A for how the expressions are found. For analytic values for A_{33}, see (Molin, 2001).

5. Outline of numerical solution

In the previous section, a transport equation was presented. To evaluate this computationally, the domain has to be discretized and the transport equation applied at the discrete points. With the discretization a number of numerical effects is introduced that may bias the solution. Choices of numerical schemes to minimize the effect of artifacts will be discussed. OpenFOAM will be presented, and who the choices are made in OpenFOAM is presented.

5.1. Discretization

To numerically evaluate the fluid transport equations 4.3-4.5 the flow domain requires representation as a set of finite discrete points; a mesh. Each term in the partial differential transport equations is evaluated in discrete points, using differencing schemes. This results in a system of coupled algebraic equations with dependencies between equations. Coupling of the equations requires simultaneous solution by means of a suitable solution algorithm.
The discretization method used in the present study is the well established finite volume method (Patankar, 1980), (LeVeque, 2002). In the finite volume method, the mesh is represented as volume elements, where the governing equations are applied as volume integrals over each cell. The divergence terms of the transport equations are converted to surface integrals, using the divergence theorem, see for instance (Kreyszig, pp. 506-509). The surface integrals are then evaluated as fluxes through the surfaces of each finite volume. Fluxes are preserved between the volume elements, as flux leaving one cell enters another. As a consequence, the finite volume method is conservative. However, numerically computation may destroy the conservation if all fluid flux in and out of a cell in one single time step. To prevent this instability, monitoring Courant number can prohibit fluid travelling too far per time step, see section 5.2.3.

5.1.1. Numerical schemes

For a detailed description of the discretization of the convection, diffusion, time derivative and source terms involved in equations 4.3-4.5, this is given by Jasak (1996), (2006). OpenFOAM uses this discretization. OpenFOAM further let the user directly choose numerical schemes for each term. The most basic scheme for the finite volume method is Gaussian integration with linear interpolation from the cell center to the cell face. For some terms special care must be chosen to limit numerical artifacts. Diverging terms are most likely to result in numerical diffusion, and should be chosen with convective specific schemes. These types of schemes are monotonic preserving, such that solutions don't grow with time and provide unphysical solution. The basic solver in this class is the Gaussian integration limited linear scheme, which requires a coefficient in range between 0 and 1 as the limiter. The value 1.0 is opted for best stability, at the cost of accuracy for lower values of limiter. For

instance the divergence term in Navier Stokes $\nabla \cdot (\rho \mathbf{vv})$ is found in OpenFOAM as *div(rho*phi,U),* in the '*fvSchemes'* file, see Figure 5-1. Here *phi* refers to $\phi = \mathbf{v}$**,** as it is rewritten by the divergence theorem to a flux, as described above. Stability is essential in this case, while accuracy comes second. Therefore, the keyword '*Gauss limitedLinearV 1.0*' in OpenFOAM gives Gaussian integration with 1.0 limited linear scheme.

Two other diverging terms were present in the transport equations of 4.5. These terms are cause of interface smearing which cause numerical diffusion, so extra importance is placed on robust and accurate scheme choice. The term $\nabla \cdot (\alpha \mathbf{v})$ is given by keyword *div(phi,alpha)*in OpenFOAM. Again stability is first priority, while accuracy is also important for the free surface. A vanLeer scheme is chosen. The van Leer scheme is second order in both time and space, and can involve solutions that exhibit discontinuity (van Leer, 1979). Since that the α-function abruptly leaps from 0 to 1 across the interface the van Leer scheme is a good choice, which also provide good accuracy. Finally $\nabla \alpha (1-\alpha)\boldsymbol{\phi}_{r\alpha}$ is given as *div(phirb,alpha),* and could similarly be chosen as van Leer. However, a specialized '*interfaceCompression'* scheme for this term is developed in OpenFOAM based on van Leer to give a smoother interface and limit the numerical diffusion. Using this scheme add to computational complexity, but is a good choice since the position of the surface is crucial to model the moonpool resonance correctly as well as reducing the spurious currents.

OpenFOAM has also added a multidimensional universal limiter for explicit solution, or MULES, that ensures boundedness of the phase fraction independent of underlying numerical scheme and mesh structure. The choices of schemes for convection are therefore not restricted to those that are strongly stable or bounded, e.g. upwind differencing (OpenFOAM).

The other discretized terms don't have as big numerical challenges, and could use simpler, faster schemes. Commonly employed schemes for $\delta/\delta t$ is first order implicit Euler scheme, which provide speed at acceptable accuracy, which is favored here. An alternative could be Crank-Nicholson which gives higher accuracy, however at the cost time and even stability in some cases. For the gradient discretization, i.e. ∇p it would be extremely unusual to select anything other than Gaussian integration on general interpolation schemes. In most cases linear interpolation from cell centers to cell faces is an effective choice, and is chosen here for its good tradeoff between speed and accuracy. The Laplacian operator, found in the viscous term $\nabla \cdot (\nu \nabla \mathrm{U})$, also uses Gaussian linear interpolation scheme, but need additional specification for face normal treatment. With explicit non-orthogonal correction of the surface normal, conservation is kept. The alternative is no correction to reduce cost. Again stability is the primary focus, so correction is used.

5.1.2. Solution algorithms

After the system is discretized to a set of algebraic equations, suitable solution algorithms must be chosen. The algebraic equation systems in the finite volume method are in general sparse, as each cell has a limited number of neighbors and few higher order schemes introducing next-neighbors are used. When the algebraic equations contain a lot of zero elements, iterative solvers are preferred to direct solution (Van de Velde, 1994). Iterative

solvers can ignore the zero elements in the linear equation system, hence get a nice speedup. Less memory is also needed, as only the non-zero elements needs to be stored unlike a direct solution using for instance LU-factorization. The drawback of iterative solvers is that nothing is known a priori about the number of iteration needed and there is no guarantee for convergence. The principle of iterative solvers is that an initial guess is gradually improved towards the correct solution. The most stable discretization schemes available was selected, so unphysical growth of the local solution can be avoided. Iterations stop when a predefined tolerance in term of residual is met. If the equation is $\boldsymbol{Ax} = \boldsymbol{b}$, the residual is given as by $\boldsymbol{Ax} - \boldsymbol{b} = r \neq 0$. After each iteration it is checked how close the result is to satisfying the equation exactly. If the residual is lower than specified tolerance, typically $tol = 10^{-6}$, the current $\boldsymbol{x}$ is accepted as the solution and iteration ends. Relative tolerance can also be used, and is defined as the ratio of the current residual over the initial residual, and is usually in the order $relTol = 10^{-3}$. This provides as an extra abort criterion. Relative tolerance is not used here, as nothing is known about the initial residual; hence the relative accuracy is not good enough. To make the iterative solvers converge faster, a pre-conditioned equation system is used to improve the conditional number of matrix A before iterations start. If the preconditioning is good, far less iteration is needed before tolerance is met as number of iterations scales with the conditional number for iterative solvers (see for instance (Kreyszig, 1999, pp. 906-912)).

The pressure solution was obtained by a linear conjugate gradient solver, coupled with a diagonal incomplete Cholesky preconditioner and tolerance 10^{-7}. The velocity solution was obtained with a linear bi-conjugate gradient solver, associated with an incomplete-LU preconditioner and tolerance 10^{-6}. Both unknown use the conjugate gradient (CG) solver, which is one of the fastest available iterative solvers, and is chosen for speed. The problem with CG method is that it only operates on symmetric matrices. While the pressure matrix exhibit symmetry can use the CG solver directly, the velocity matrix is asymmetric. A fictive, double-sized symmetric matrix must be created, hence the prefix *bi*-conjugate gradient method and the LU-preconditioner.

Since the pressure and velocity are coupled together by the governing equations neither can be solved alone. OpenFOAM's solution to this is to use a pressure-implicit split-operator (PISO) algorithm: First the pressure is solved, and then this pressure is used to improve the velocity. The updated velocity can then be used in the pressure again. This loop improves either attribute till a global solution is met. OpenFOAM recommends more than one but not more than four loops. Having in mind that a complete PISO loop is extremely time consuming, three iterations is chosen to get a good tradeoff between accuracy and speed.
Dynamic re-meshing is managed by cell stretching, no topology is changed. At every step, the mesh is checked for non-orthogonality and cell stretching, and is warning are printed if cells are too skew.

The main loop of the solver *interDyMFoam* looks like this:

- Check Courant number, and adjust time step if necessary
- Move the mesh, and get the frame acceleration
- Solve $\alpha - equation$ 4.5

- Smooth α field, and calculate new normal and curvature of surface
- Get a momentum prediction using the old value of flux
- PISO loop:
  - Predict flux
  - Construct and solve pressure equation
  - Correct fluxes
  - Reconstruct velocities

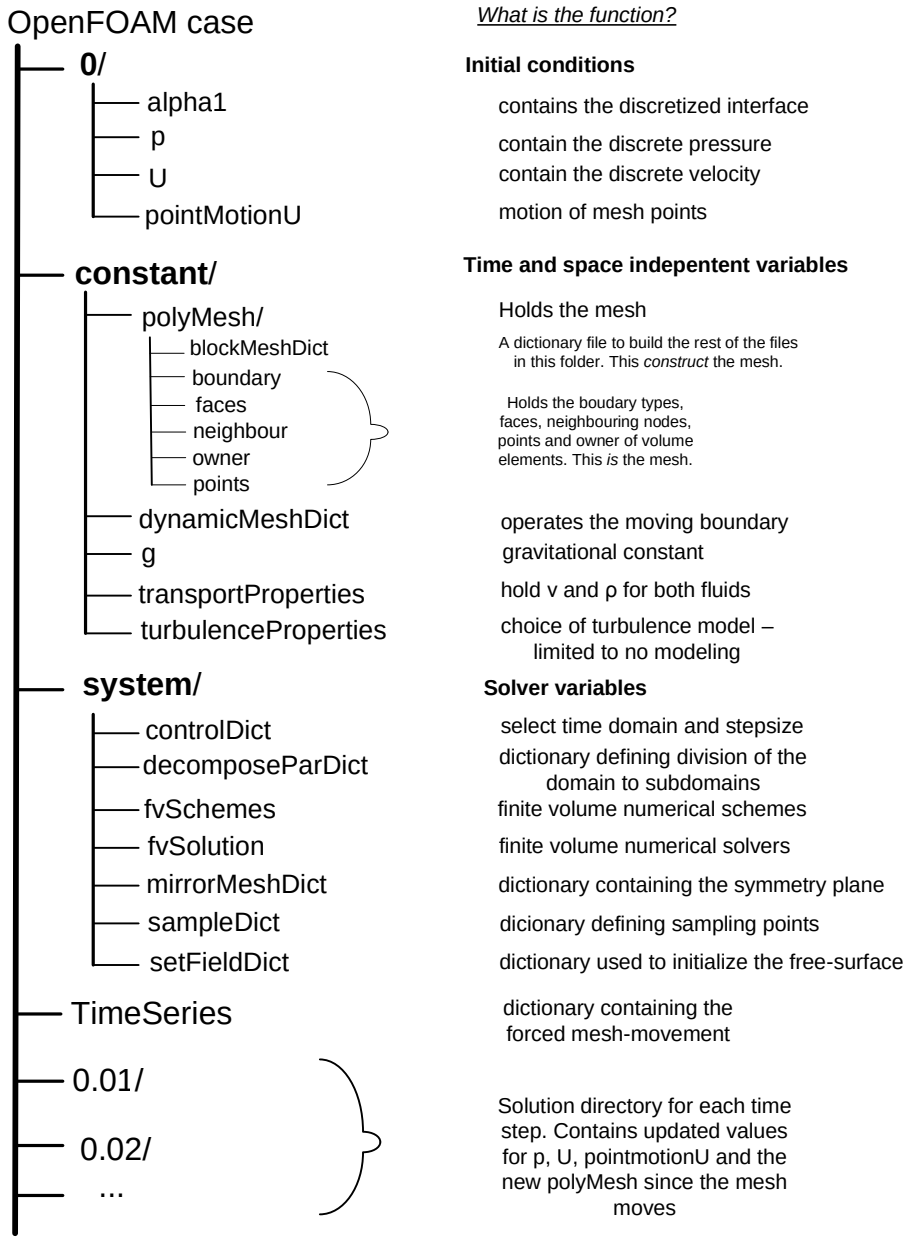


Figure 5-1: The structure of an OpenFOAM case

5.2. Physical properties of the flow and subsequent modeling choices

Some physical properties important for numerical simulation related to the discretization are discussed in the section. These include numerical wave absorption, turbulence schemes and Courant number.

5.2.1. Numerical wave beach

Numerical wave tanks have no standardized efficient approach for damping out waves, and a reoccurring problem is that wave reflects from the end of the domain. The obvious choice would be to mimic the beaches of the wave tank. The energy would be absorbed by wave breaking, which is difficult to capture accurately. This is especially true with long waves with low amplitude as in this study. It might work ok for other cases, but it is not the preferred way nonetheless as many mesh elements is needed to capture the breaking satisfactory, which is a waste of computational power without gain to the studied case. Currently, the general solution to numerical end zone damping is to include a far field large enough to let the waves die out naturally, and impose a zero pressure and velocity at the boundary. The trouble with this solution is including a large enough far-field. Again it is computationally expensive for seemingly no reason. If attempted to limit the size of the zone, a too sharp end zone would artificially introduce spurious pressure and velocities back into the domain. There have been some publications on numerical wave absorption methods. A review of methods is given by Romate (1992). One method known as absorbing layer put forth by Baker et al. (1981) introduces a dissipative pressure term into the free-surface equation, which is good at fairly high frequent waves. Maisondieu and Clement (1993) suggested a piston-mode wave flap boundary, which proved excellent at capturing small frequency waves. Clement (1996) proposed a combination of the two, capable at the whole specter of frequencies. Clement study was done for potential theory, but the ideas could be extended to viscous theory. Because OpenFOAM is open source this could easily be implemented, however, it is not known if VOF is extendable without further modifications, which might corrupt robustness. Instead a more crude, yet effective method is used in the following study: the far field changes abruptly to a coarse mesh in the horizontal direction, so the energy of the wave is numerically dissipated. The advantage is the simplicity and light numerical load, the backside is the mesh has to be controlled to be coarse enough to absorb the waves. Also, if attempted to increasing the expansion ratio to make the far field smaller, the result might be spurious waves if the expansion ratio is large enough.

5.2.2. Turbulence

Existing turbulence models, which have been mainly proposed for single phase flows, may not adequately represent the turbulence structure at the free-surface (Senocak & Iaccarino, 2005). The free surface treatment in OpenFOAM does not account for the effects of turbulence. This is a consequence of the fact that the Reynolds averaged approach to

turbulence modeling does not match the notion of an infinitesimally thin interface between air and water. As a consequence, all free surface simulations can be viewed as a direct numerical simulation (OpenFOAM). DNS is associated with certain requirements to the mesh size, such that all spatial and temporal scales must be resolved in the mesh to capture all turbulence. To accurately describe flow to the full extent of Navier Stokes, the smallest of Kolmogorov scales should be meshed. These scales are given as, (Pope, 2000)

$$\eta = \left(\frac{\nu^3}{\varepsilon}\right)^{\frac{1}{4}} \tag{5.1}$$

$$\varepsilon = \frac{U^3}{L} \tag{5.2}$$

where η is the smallest length scale for turbulence, ν is the viscosity, ε is the smallest energy dissipation scale, U the characteristic velocity of the fluid and L is the characteristic length. Letting L be the length of the domain, one would want $\mathrm{N} = L/\eta$ elements to capture all Kolmogorov scales in one direction. Inserting 5.2 into 5.1 gives

$$\eta = \left(\frac{\nu^3 L}{U^3}\right)^{\frac{1}{4}} \tag{5.3}$$

$$\mathrm{N} = \frac{L}{\eta} = L \Big/ \left(\frac{\nu^3 L}{U^3}\right)^{\frac{1}{4}} = \left(\frac{L^3\, U^3}{\nu^3}\right)^{\frac{1}{4}} = (Re)^{3/4} \tag{5.4}$$

Since the present study is limited to two dimensions, the least N^2 elements, meaning $N = (Re)^{6/4}$. In (Kristiansen T. , 2009, pp. 19-20) the following formula for Reynolds number is presented for an oscillating plate, and explained why it is approximately valid in this case.

$$Re = \frac{2\omega a^2}{\nu} \sim 10^3 - 10^4 \tag{5.5}$$

Here, $\omega = 2 - 9\, rad/s$ is the oscillation frequency, $a = 0.01 - 0.05m$ is the ambient amplitude. The ambient amplitude is dependent on the frequency, and for instance at resonance $Re \sim 2.5 \cdot 10^4$. This means two things. One: a grid of 4 million elements is needed to capture all turbulent scales. And two: the flow is governed by a laminar flow regime, with only very little local turbulence. $Re = 10^5$ is proposed as the transition point to turbulence for engineering purposes (Faltinsen, 1990, p. 238). The boundary layer is hence thought to by laminar, and given approximately as $\delta \cong 4.6\sqrt{2\nu/\omega} \sim 3 - 6mm$ (Kristiansen T. , 2009, p. 19). Hence the error of not capturing all Kolmogorov scales is not a major problem, and OpenFOAM's laminar DNS solver should be sufficient. Some turbulent flow will exist and some of these effects will be solved. The trouble is that not all scales are captured by the mesh, which had the finest mesh had 1.5 million elements. According Moin and Mahesh

(1998), the full Kolmogorov scales is not needed to accurately describe and physical phenomena using DNS, and give numerical values of about 5-10 times larger for specified cases. If this is taken to account, maybe only 400'000 elements needed, which is fulfilled at some of the simulations.

5.2.3. Courant number

There are numerical limitations on how far fluid is allowed to travel during a single time step. This limit is given by the Courant–Friedrichs–Lewy condition. It is a necessary, however not a sufficient[1], condition for convergence while solving explicit time-marching schemes for numerical solutions of partial differential equations. As a consequence, the time step Δt has to be adjusted to comply with the condition; otherwise the simulation will produce wildly incorrect results. The condition in a two dimensional case is usually given as (Skamarock, 2005)

$$\left|\frac{\mathrm{U_x}\Delta t}{\Delta L_x}\right|, \left|\frac{\mathrm{U_y}\Delta t}{\Delta L_y}\right| \leq C \qquad 5.6$$

With an limited linear first order method to solve the diffusive term, the Courant number could be $C \leq 1$. In OpenFOAM, the variation of Courant number can be met by variation of Δt at runtime as the velocities and mesh change. Using a VOF also has an impact on the chosen Courant number, because it is directly related to the diffusion of the interface. Experience has shown that the stability of the numerical solution procedure is ensured if the Courant number is kept below 0.5 (Rusche, 2002). Subsequently, a limit on the Courant number was set to 0.4, to be on the safe side.
[Implisit vs explicit][explicit unstable at C>1, implicit relaxes the steps, thus underdamps the problem]

5.3. Preprocessing

Mesh was created with a text based script coupled with OpenFOAM's regular mesh builder *blockMesh*. blockMesh only needs nodal points, connectivity of nodes and wanted amount of elements between notes to create a mesh. This makes it perfect for simple geometries, as it reduces overhead. The in-house mesh generation drawing tool of MARINTEK, *mega*, was used for as well. However, with no possibility to mass produce grids quickly or a good way to refine a grid by a factor globally, the text-based *blockMesh* was preferred. In all moonpool cases the domain was so simple that no visualization was needed, and simply provided extra work. Also, *mega* was only available on a server, so the graphics on the local computer was bottlenecked by the network transfer rate. Even with high-speed university network the larger meshes made unendurable lags when paning, scrolling or

[1] On the topic of different types of stability, see for example (Aarseth, 2009) pages 193-205, 213-237

rotating. It an impossible task to effectively create the mesh, as just locating a node was not trivial when the size got big. Also, mega had some overflow issues, making it randomly exit when more then 40'000 2D nodes where used. There were similar limitations in number of elements along a line of about 50. Text based pre-processing removes all these problem, but adds the problem of less control. To deal with this, mesh were instead checked for consistency by *checkMesh* tool in OpenFOAM. CheckMesh survey the grid for overlapping, too skew or non-orthogonal elements. To verify that elements was placed as intended and the initial values were correct, *ParaView* was used.

5.4. Postprocessing

Postprocessing was done in two ways. The first was using the texted based sampling tool of OpenFOAM to pick out data points for further processing into plots. The second was 3D visualization tools for checking the flow for unphysical behavior and general monitoring of the result. For 3D visualization *parallelView* was preferred as it connects seamlessly to FOAM-cases, while more professional commercial tools like *GLview* needed conversion. The files visualized were of considerable size, typically 5-10 GB and made conversion to *GLview* impossible for the big cases as the converter ran out of memory. Even though *GLview* is a more complete tool, *parallelView* delivers the basic routines like cut-plane, isosurface and can turn interpolation of result on and off. When viewing the result, the default option was to view interpolated data. This option does not show the user the quality of the underlying mesh, and should in general be used with care. When viewing for mesh convergence, it is essential to not use this option, as great jumps between elements is a good indication as to where to add more elements. For better understanding of the physics, interpolating the data might still be a good choice, as this makes better use of the obtained data.

5.5. Possibility of parallel processing

Since the transport equations boil down to a set of $\boldsymbol{A\boldsymbol{x} = \boldsymbol{b}}$, there is possibility to use parallel processing. To get the full potential out of multiple processors, the data passed between cores should be as little as possible, ideally no communication . Given several dependencies between solution variables, it might seem the possibility of harnessing full effect of a parallel processing is limited. However, in general, iterative problems have great capabilities of being solved in parallel, as the iteration can complete on each sub-domain before needing to communicate. Also the dependencies are between pressure and velocities of neighboring nodes, which are placed in sub domain by proximity in space.

Another important issue of the use of supercomputer in this field is the large amount of data. Memory is not really a problem, as a conjugate gradient method memory demand scale linearly $O(n)$ with problem size, In terms of storage requirements, the memory peak occurs during the pressure-velocity solution using PISO where it is necessary to simultaneously store the momentum and pressure matrices (Jasek & Tukovi, 2004). In OpenFOAM a run of 40000 elements requires 70Mb and 150000 requires 270MB, in agreement with the assumption of

linear memory requirement scaling. So, a prediction for a 4’000’000 element mesh required to capture all spatial and temporal scales would need about 7 GB of memory, superseding the capabilities of a common computer today. A similar problem arises with post-processing of such large solutions. With parallel processing, the memory can be split between processors and the native post-processor of OpenFOAM, *paraView* can view only part of a solution if they are too big to load into the memory or process the vast amount of data on separate processors and send them only to a single screen. *OpenMPI* is a library for parallel processing, and is included out of the box in the standard distribution of OpenFOAM. Spatial domain decomposition methods for distributing the sub-domains evenly between processors only needs configuration in *decomposeParDict* specifying the number of processors needed.

During this study, the candidate tried to get access to a supercomputer, for greater turnaround of the result, simulate for higher quality grid and study the parallel performance of the CFD solvers in OpenFOAM with respect to speed up. Unfortunately it was not permitted to use the supercomputer at SINTEF, while the supercomputer at NTNU which he got access to lacked support for OpenFOAM.

6. Comparison with experiment

The experimental which the numerical data will be compared to is presented. First the setup is given in section 6.1, then observations from the experiment is given in section 6.2.

6.1. Experiment setup

As part of the information regarding the experiment was property of SINTEF, the setup from an almost identical setup will be used as referenced: The experiment from Faltinsen et al. (2006) was conducted in the same facility with the same equipment. This experiment was an extension of that project, and for that reason is identical except from a wave damping is changed with a beach.

The model tests used in the present study were conducted by PhD graduate Trygve Kristiansen during January-February 2010. The tests were performed in the narrow-wave flume of the Department of Marine Technology, NTNU, in Trondheim. This wave flume is $0.6\,m$ wide, with water depth $1.03\,m$ and has a total length of $13.5\,m$. The flume was equipped with convex curved beaches at each ends to damp out waves at the end zones. A two-hulled ship section, representing a two dimensional longitudinal cut of the moonpool, was placed in the middle of the tank, see *Figure 6-1*. The moonpool covered the whole width of the flume apart from a few centimeters to of allow frictionless vertical heave movement. The hull had constant dimensions and was made of plywood, apart from replaceable corners made of steel. The width of each hull was $B = 0.36m$, moonpool gap of $b = 0.18m$, draft to undisturbed water surface $D = 0.18m$ and water depth of $1.03m$. All dimensions are shown on *Figure 6-1,* note that the flume was $0.60m$ wide, while the hull only $L = 0.595m$, so some three dimensional flow effects were permitted. The bottom corners facing into the moonpool where replaceable, so that different moonpool configurations could be tested out. More than seven different design configurations were tested in the experiment, including some curved corners and some with perforated plates. However, only four configurations were tested numerically, and will be covered here. These were *(i)* without any appendage, *(ii)* with a medium appendage covering 20% of the moonpool opening, *(iii)* a large appendage covering 40% of the opening[2] and *(iv)* 'skirts' attached vertically at the corners to increase the amount of sheltered water inside the moonpool. All mentioned concepts are illustrated in *Figure 6-2*. A regular oscillatory heave motion was forced at amplitude $5\,mm$. The amplitude was reached after a linear ramp-up to smooth out motions, lasting 10 seconds. The forcing was done with a servo motor connected to a ball screw and rail system. A Labview computer application generated the steering input to the servo-motor. The placement of the servo unit is visible in *Figure 6-1a*. After about 140 seconds, or 100 oscillations, the system was ramped-down in new 10 seconds.

[2] A configuration with 30% was done in the experiment, but these data was confidential, so 40% were done instead as it would give a larger variation.

All configurations examined in the present study had sharp corners, such that vertex shedding is forced at correct stagnation point. A total of 12 wave gauges was used, the locations are found in *Figure 6-1*. The internal sampling frequency was 19.2 kHz which is by far good enough to capture all relevant waves, and a Butterworth low-pass 1 kHz filter was applied to rule out "folding back" (Aarsnes & Steen, 2008). Finally, the data was stored at 400 Hz. The last 10 oscillations before ramp down were used for sampling.

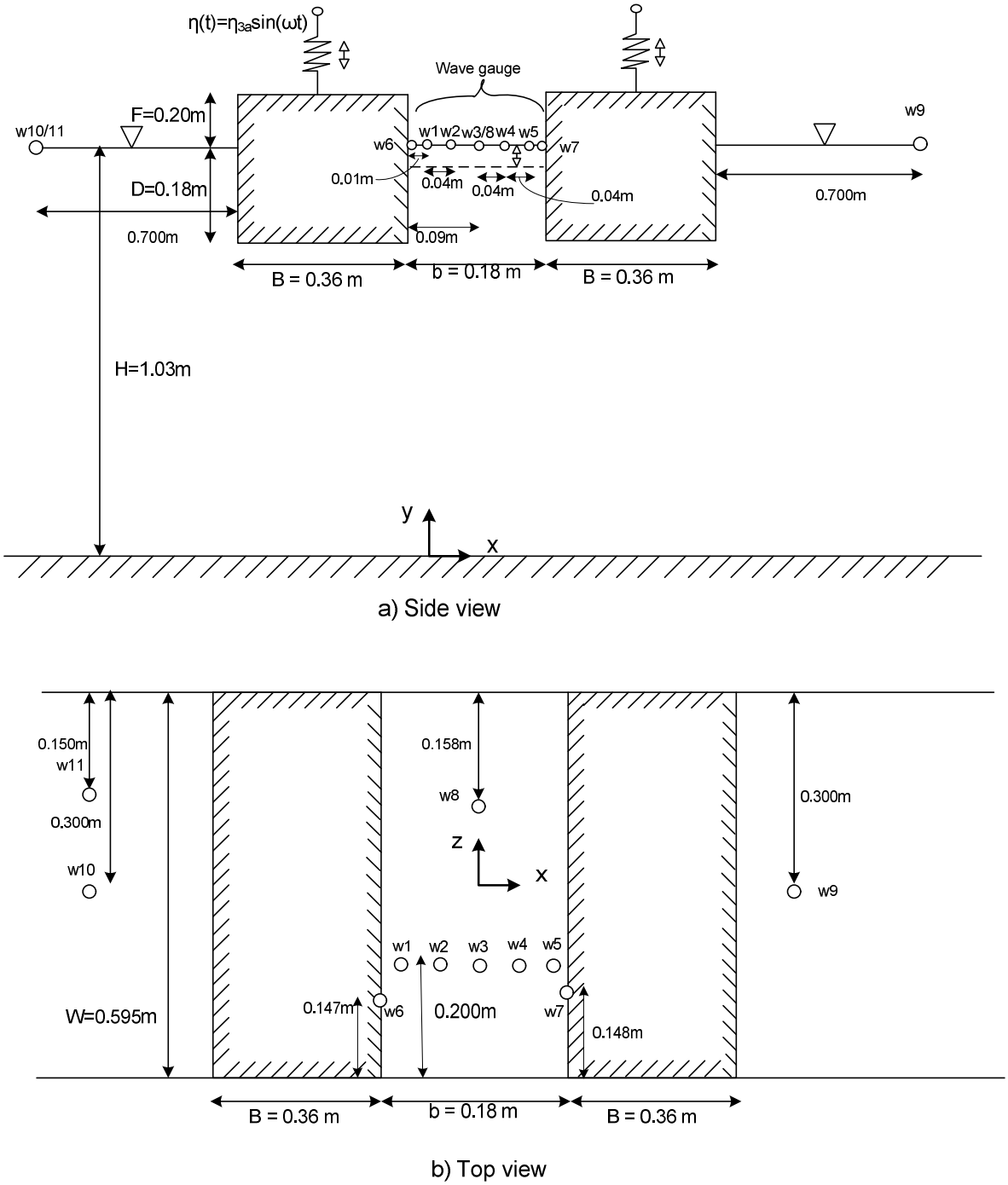
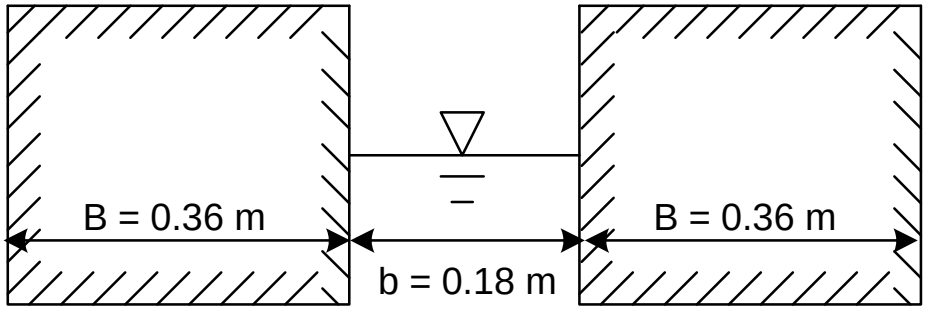
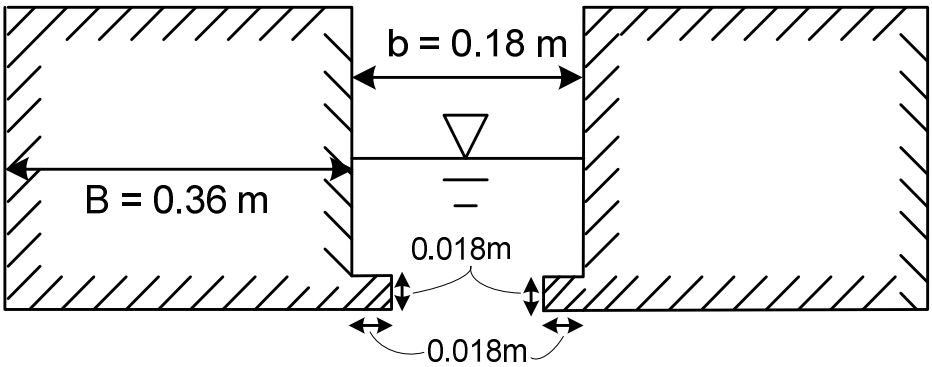


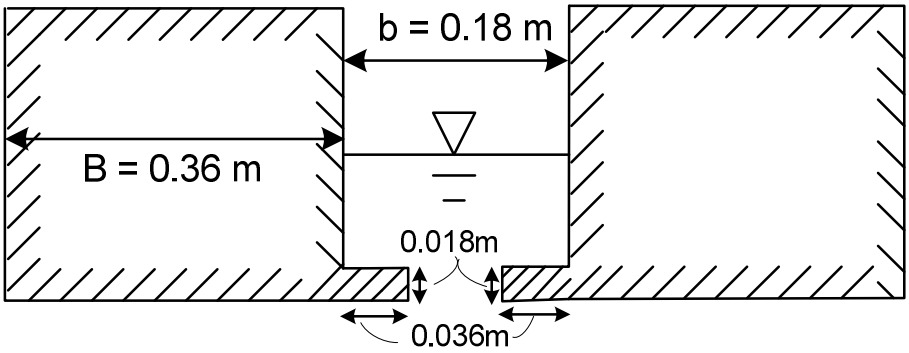
Figure 6-1: a) Side view and b) top view of the experimental setup, including dimensions of moonpool and placements of wave gauges w1-w11



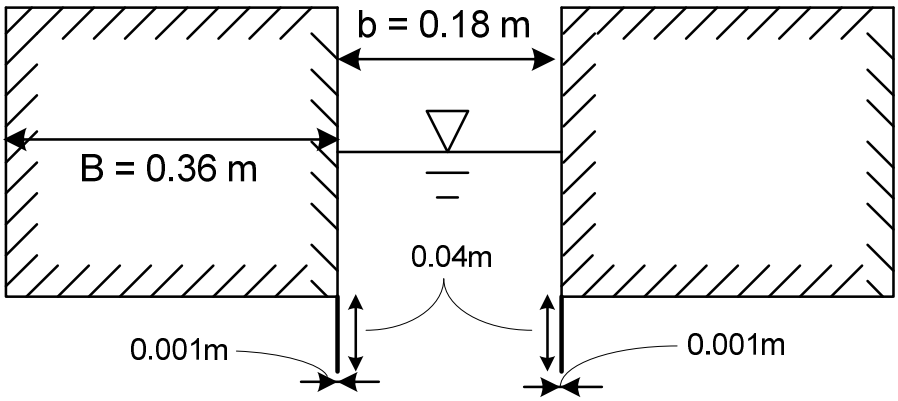
(i)



(ii)



(iii)



(iv)

Figure 6-2: Different moonpool appendages tested during experiments. (i) No appendage (ii) Medium appendage (iii) Large appendage (only tested numerically) (iv) Skirts

6.2. Observations and remarks about the experiment

After the ramp up was complete, the experiments demonstrated piston-like $2\pi/T$-periodic wave elevations in the moonpool, where T is the forcing period of the heave motion. Outside the moonpool smaller waves was observed for cases with large heave inside, and waves with decreasing amplitude travelled away from the hull, symmetric about the center of the moonpool. Viscous effects were visible around the corners of the rectangular ship hulls, where tiny particles in the water visualized vortex shedding. This vorticity was especially evident when the free-surface-motion amplitude in the moonpool was large. The separation point could be traced back to the corner/bilge.

Cross-flow and vertical flow in the 2.5 mm gap between the moonpool and the wall was also observable, but only at the frequencies where oscillation was large inside the moonpool and at much less scales than water movement elsewhere. This 3-D effect should ideally have been avoided, and cannot easily be reproduced in numerical 2-d simulations.

The height of the wave gauges drifted during some tests, so that it recorded higher wave elevation after ramp down when there was no visible waves then before ramp up. This could be as a result of water meniscus remains on the steel wire, creating a short circuit or because the wave gauge is flawed with a non-linearity. As two wave gauges in particular was drifting, this was attributed to non-linearity. Anyway, there were enough redundant wave gauges, so it could be ignored.

The wave beach never damp out the wave perfectly, and will generally work poorly with long small amplitude wave as produced in this experiment, as the purpose of the beach is to make the energy dissipate in wave breaking. This has not been accounted for, and will introduce some energy into the system. Fortunately, the numerical method exhibit a similar weakness, so this error might to some degree cancel out, although no quantitative analysis was preformed.

7. Results

Before assessing the moonpool flow, the performance of the two-phase free-surface VOF algorithm in OpenFOAM will be evaluated on a validation case in section 7.1. Because of its relative simplicity, the experimental studies of wavy flow generated by a NACA0012 hydrofoil moving beneath the free surface (Duncan, 1983). This hydrofoil is often used as benchmark of numerical methods developed to compute free surface flows for instance see (Vogt & Larsson, 1999), (Rhee & Stern, 2002). The solutions obtained with on the NACA foil are compared with available experimental data (Duncan, 1983) and numerical solutions with one-phase LS algorithm (Mascio, Broglia, & Muscari, 2007).

Three different moonpool configurations will be analyzed in section 7.2; without, medium and large appendage and moonpool with skirt in section 7.2.1-4 respectively. Verification has been carried out in order to assess the reliability of the computed results for the first case. To this aim, each numerical solution has been computed at least on three grid levels, each obtained by removing every other point from the previous finer one, i.e. with a global grid refinement ratio of two. Validation is carried out by using the methodology proposed by the (ITTC, 2001). The results from the different moonpool cases are compared to experiments presented in chapter 6, as well as with linear theory study of Kristiansen et.al (2008).

7.1. Submerged NACA-0012 hydrofoil

The hydrofoil has been chosen as a validation test case for the proposed algorithm in this work, because it is easily adjustable between breaking and non-breaking wave. The case is stationary, which lightens the computational burden as previous solutions may be used as initial values for the finer mesh. However the main reason is that it is well documented in literature, and high quality experimental data is available.

This test case will be limited in that it does not use a moving mesh, which will be used with the moonpool: the *interFoam* solver will be used instead of *interDyMFoam*. In addition the flow here will be turbulent, not laminar, so the verification is more of a test study to master the solver than it is help in the moonpool case. At first, the goal of the verification was to get a rough estimate on how many elements would be needed for convergence of the free-surface. Instead, the verification was used just to see how the solver preformed on this case separately and generally just used to get know OpenFOAM.

7.1.1. NACA test setup

In the experimental set-up, a NACA-0012 hydrofoil, whose chord length is $c = 0.2033\,m$ with an angle of attack $\alpha = 5°$, is towed in a tank with towing speed $U = 0.8\,m/s$ (see **Error! Reference source not found.**). The distance between the tank bottom and the profile was fixed as the towing height, $H = 0.175m$ to the midpoint of the chord. Depth of submergence s is varied by varying the water level, and this produces breaking wave when s

is sufficiently low. The relevant non-dimensional parameters based on the chord length and the free-stream velocity are $Fr = U/\sqrt{gL} = 0.567$ and $Re = UD/\nu = 1.6 \cdot 10^5$. All lengths and velocities are made non-dimensional by the chord length. Simulations are done for the non-breaking case as the moonpool exhibit little or none visible breaking behavior during the experiment. The non-dimensional depth of submergence was selected equal to s/c =1.286 (Duncan, 1983).

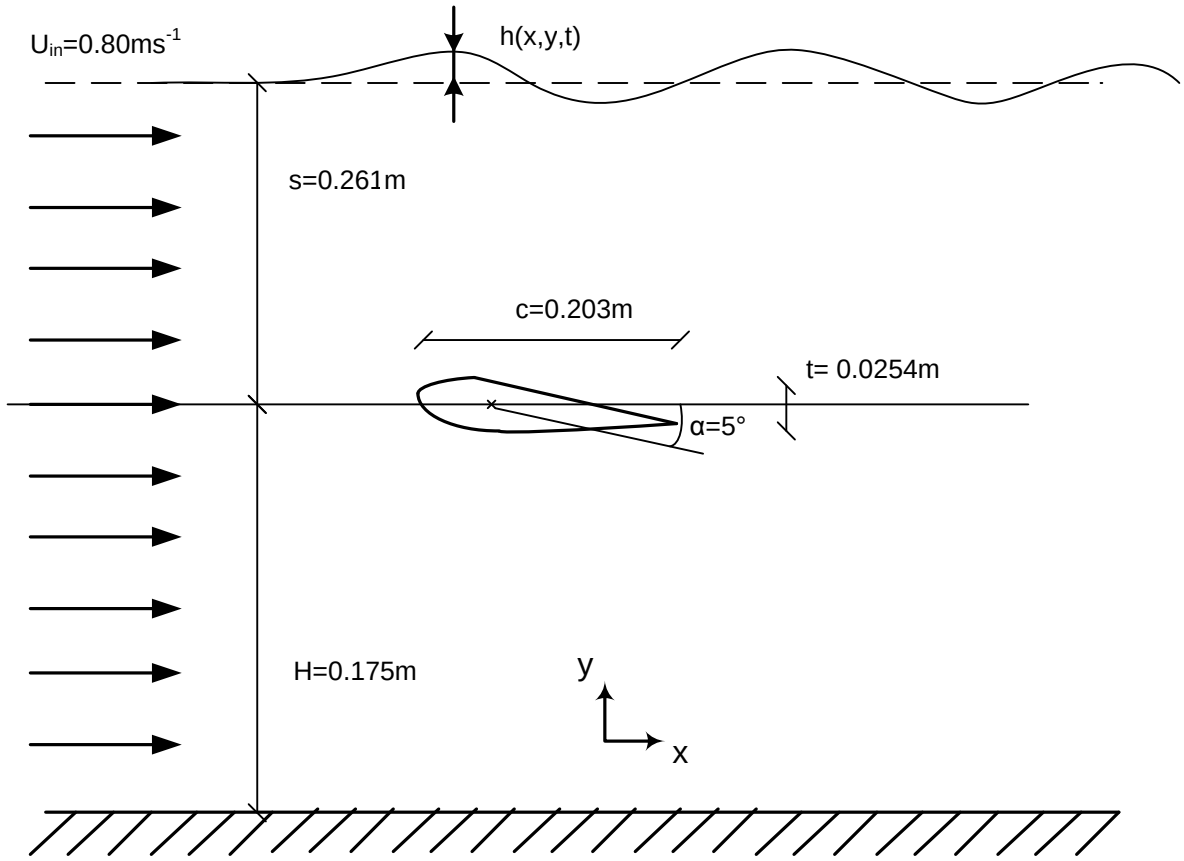


Figure 7-1: Experimental setup of NACA foil. For details on the setup, the reader is referred to Duncan (1983).

An existing mesh for a NACA foil was available from Holm. This mesh was $\lambda = 4.97$ larger than in the Duncan's experiment. An attempt to scale down the mesh failed, as the geometry of the foil was lost in the scaling process in *mega*, so the original NACA mesh dimension was kept. Instead, similarity between forces was achieved by Froude-scaling, to preserve the free surface forces. As a result however, the Reynolds number was scaled by a factor $\sqrt{\lambda^3}$, and the case is consequently turbulent, $Re = 1.8 \cdot 10^6$. As this flow already was turbulent, the increase doesn't really matter. However, as a verification case for a laminar flow case, involving turbulence it is not optimal.

The computational domain was set up, starting with a no-slip boundary condition of the bottom. Since the experiment was conducted by towing, the bottom was set to move with velocity $U = 0.80ms^{-1}$, which is the relative velocity as seen from the foil standing still. An

atmosphere was clamped at $z/c = 1.5$ where zero relative pressure was forced. Inflow velocity was forced 4.5 chord lengths before the nose of the chord, and 14 chord lengths of included as the outflow, to ensure enough far-field damping. The outflow boundary was set to zero-velocity gradient, meaning wave should die out before point. The free-surface was set initially flat at $\frac{s}{c} = 1.286$ and pressure was set to buoyant pressure throughout the domain.

Using an approximation for boundary layer of a flat plate (White, 2003, p. 452)

$$\frac{\delta}{x} \sim \frac{0.16}{\sqrt[7]{Re}} \qquad 7.1$$

the boundary layer is approximately $\delta/x \sim 20\,mm$. The coarsest mesh has element size per chord length $\Delta/c \sim \frac{1}{200}$ in the direction perpendicular to the surface on the foil. This means 20 elements within the boundary at the trailing end of the foil ($x = 1.009$). Due to the curvature involved in the foil, the text-based approach for gridding was unavailable, and global refinement in *mega* proved difficult. More specifically, the program crashed when trying to extrude the mesh with more than around 40000 elements in 2D. Therefore, it was very hard to follow the ITTC standard for verification (2001), as refinement was only partly possible, and using less elements then 13000 would seriously misrepresent the physics. A solution to this problem could have been looking into a different preprocessing program to do the job, but since *blockMesh* worked perfectly for the moonpool case this was not an option. It was attempted to use OpenFOAM's *refineMesh*, among other tools, to improve the grid, but mega and Foam coupled badly as long as the grid from mega remained 2D. The solution was to ignore the poor convergence study here, and do them directly on the moonpool instead, with the *blockMesh* tool.

Mesh #	# volume elements	Largest elements size at boundary	#element inside BL	# elements in one wave height
1	~13000	5mm	20	6
2	~25000	3mm	30	8
3	~40000	2 mm	40	12

Table 7-1: Mesh used to simulate NACA-0012

Initial conditions were given as the Froude scaled inflow velocity, uniform over the whole domain, including the air phase. The code is run till a steady-state is achieved. The solution from the coarser mesh was used as initial conditions using the *mapFields* tool of OpenFOAM.

7.1.2. NACA results

The result of the three meshes compared to experimental data is given in Figure 7-3. In general, the agreements are good. The phase is extremely well captured, for all three meshes. The amplitude is a bit off, about 10% for the finest mesh and 25% for the coarse. A similar

discrepancy, of 5% – 15% was found in Mascio et al. (2007), so with the limited mesh size used, this is acceptable. What is not satisfactory is that increasing the number of elements doesn't monotonically improve the result; at some points mesh 1 is closer to the analytic solution than mesh 3. This of course has to do with free surface not being a good way of validating. Also, it is noted that elements was rather randomly added in areas near the foil and near the surface, and not consistently dividing every cell in half as recommended by ITTC for convergence study. In areas with a dense mesh close to sparse, the added elements in the accuracy in total may be cancelled out, since the sparse area will pollute the finer meshed areas. Therefore, global refinement will be done with the moonpool case. Another cause of error was initialization of the VOF variable in the sloped cells, which was seen to cause numerical difficulties. Using this case as an indication how many elements is needed to capture was a surface was a bit problematic. However, mesh 3 had about 50 elements one wave length, and 12 elements over the wave height, so less then this is not recommended.

Figure 7-2: Simulation of the surface in paraView. Result obtained from mesh3, with interFoam solver. Red represent water, blue air, with a thin interface inbetween.

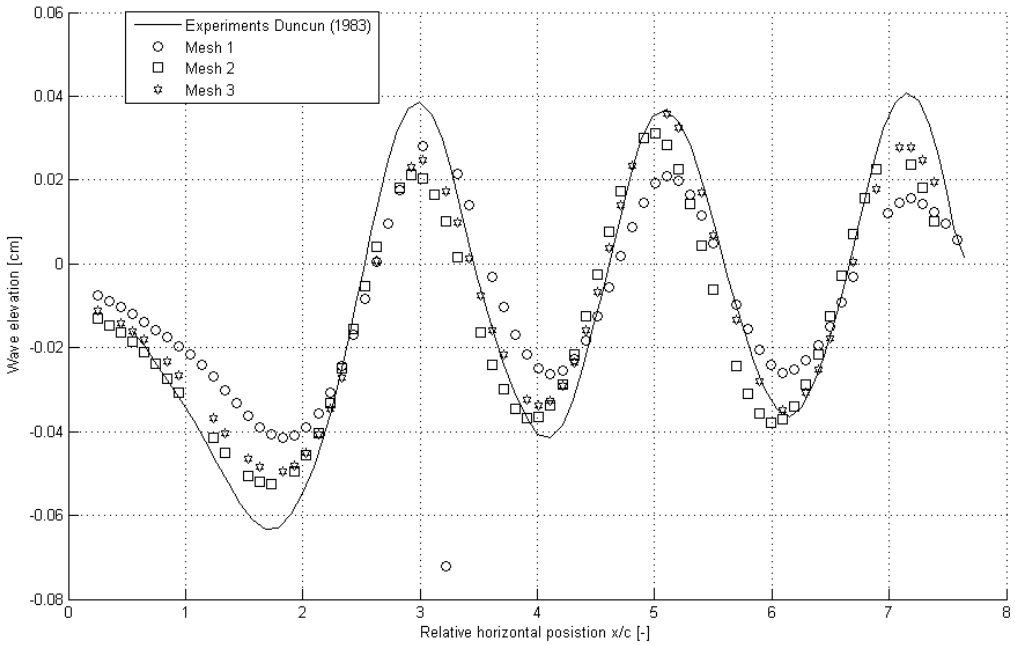
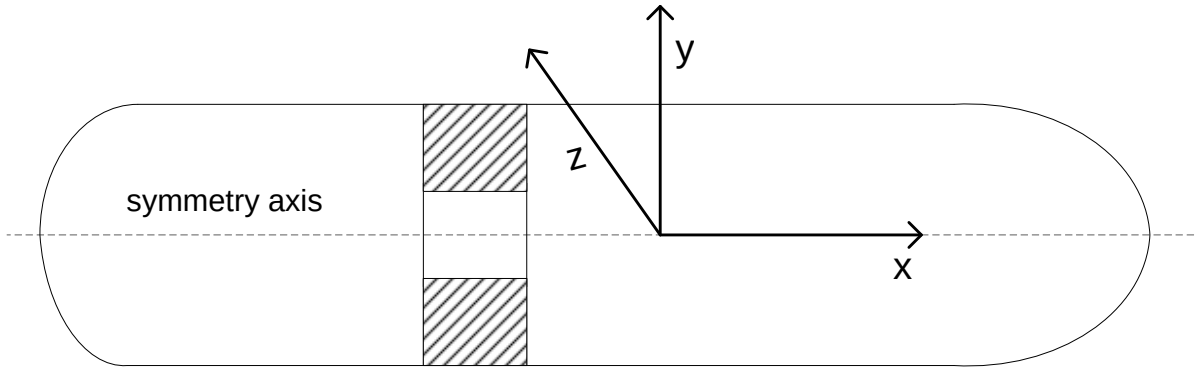


Figure 7-3: Comparison of mesh resolutions with experiments of Duncan (1983).

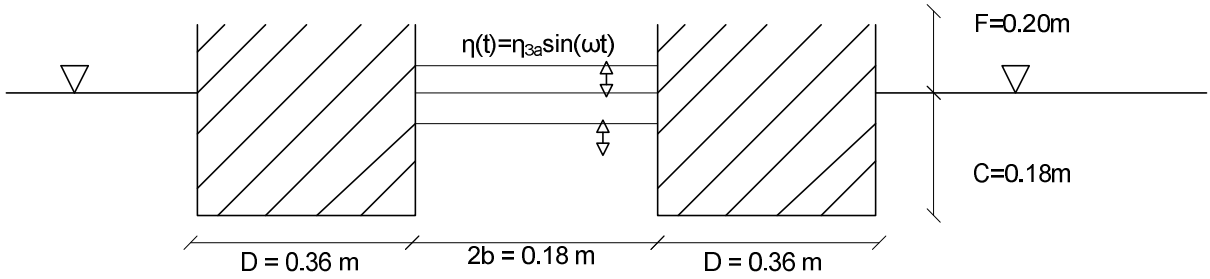
7.2. Moonpool setup

Symmetry is assumed about the x-axis (see *Figure 7-4*(*i*)), so only half the ship is modeled. Therefore a symmetry plane indicated the boundary condition that cut the moonpool in half. Also, only a strip is modeled in the x-direction. This is equivalent to assuming the moonpool is infinitely long, which again is assuming end effects/3D-effects are neglectible. As mentioned in chapter 6, the model is also based on this assumption, so comparison should be unproblematic.

Tank bottom and ship hull is modeled by a no-slip boundary condition, the top of the tank by atmospheric pressure and the outlet with zero velocity and ambient pressure e.g. buoyant $p(y) = \rho g y$. Initial values were given as zero velocity and static pressure throughout the domain. The VOF-variable α was initialized at $h = 1.03$.



i)



ii)

Figure 7-4: Schematic i) topview and ii) sideview of the moonpool.

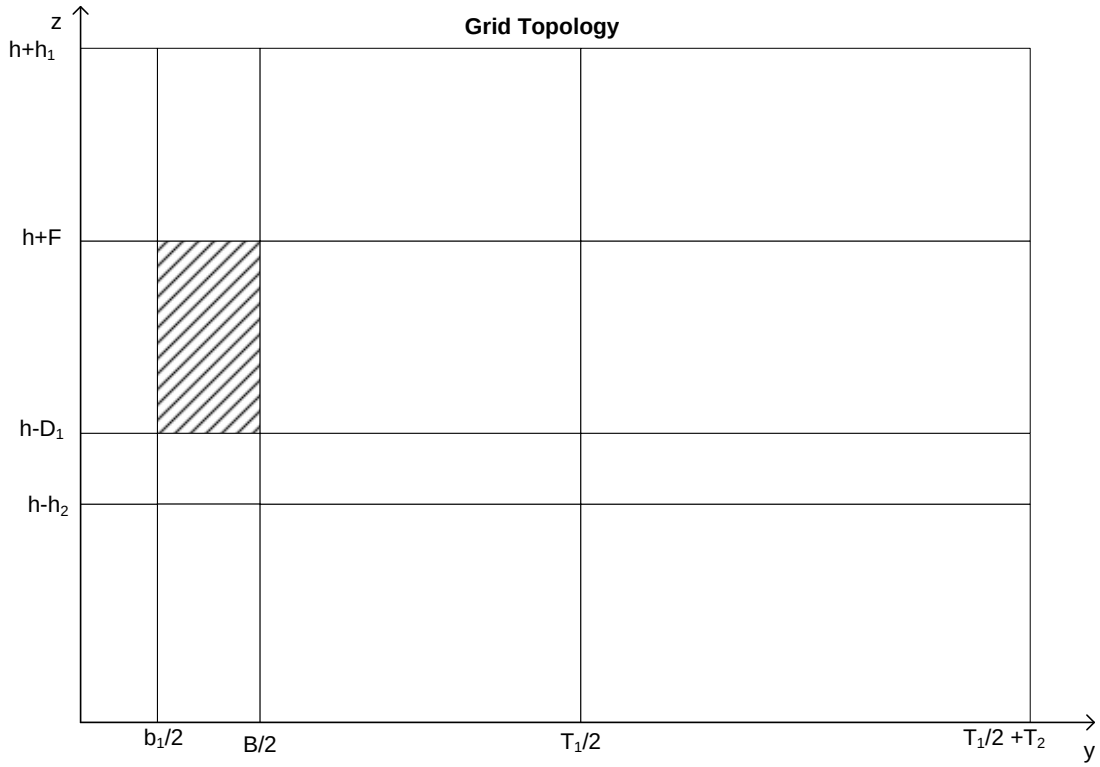


Figure 7-5: Grid domain with patches. The filled box represent the moonpool. The distances on the figure do not represent actual distance.

The case with the regular moonpool was most thoroughly analyzed, and this case was used to check the limits of the code. The results of these tests were taken directly to the other cases, without testing. Tests include the effect of moving the atmosphere, decreasing the far field, increasing grid expansion ratio in a patch.

The sensitivity to the result was minimal when changing the atmosphere level. However, since no findings in literature confirmed that the air flow was insignificant, it was thought best to leave it identical to the wave tank, and reduce the number of elements in this area instead. Although air probably has very little viscous damping involved, its effect on the surface when at high velocities were unknown, and could have the some of the same effects as forcing a too small outflow-field.

As for the outflow, it was much more sensitive, and 30 moonpool length of outflow was chosen as far-field after having tried also 10 and 20. This could arguably have been increased more. However, adding far field was very expensive on the chosen mesh structure, as a lot of elements were placed in the y-direction in the free surface region, see Figure 7-6. To further increase the outflow, a different meshing structure is recommended.

A broad spectrum of frequencies was simulated. Based on these simulations, the resonance was located. More simulation was then done in this the vicinity of the resonance, and refinement of the mesh was done in an area.

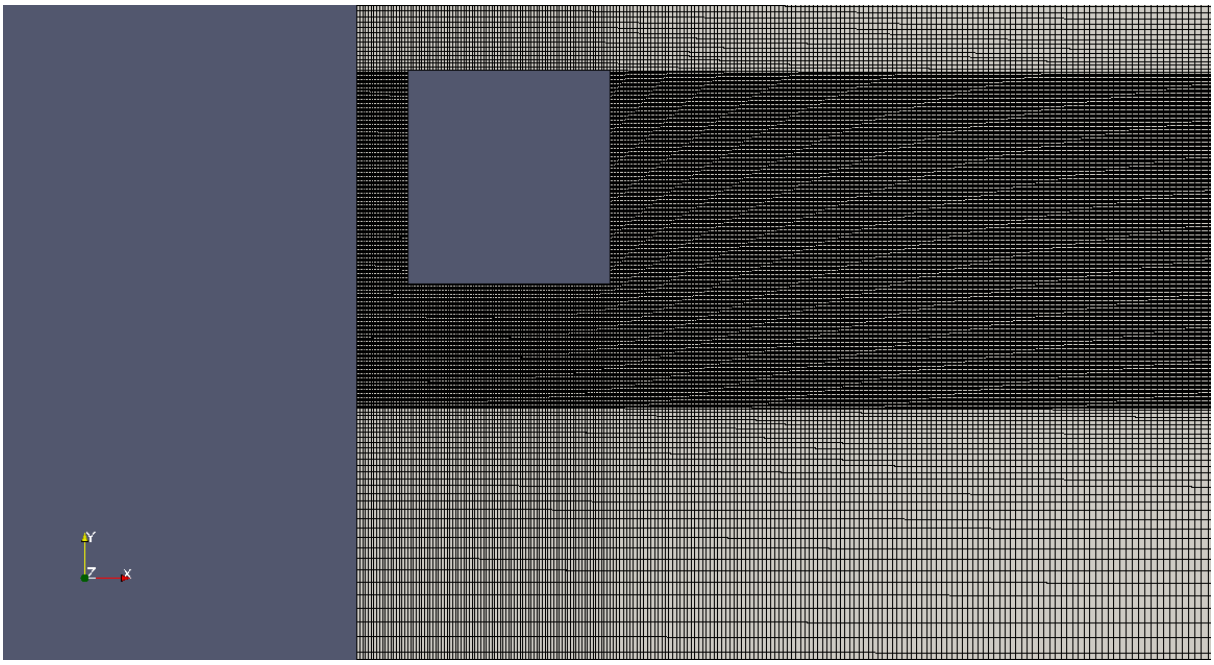


Figure 7-6: Mesh 3. A refinement around the bilges and around the surface is noticed.

7.2.1. Sampling

Three numerical 'wave probes' is placed inside the moonpool, spread equidistant over the width, to mimic w1-w5 in the experimental setup. A single wave probe was set up 0.70 m away from the moonpool hull, to produce the results of wave probes w9-w11. The numerical probes sampled the VOF parameter α from tank bottom to the atmosphere at 1000 points, using linear interpolation between elements. As $\alpha = 1$ equals water, and $\alpha = 0$ air, the proper expression to find the location of the free surface is through the integral,

$$\eta(x) = \int_0^{\infty} \alpha(x, y)dy - h \qquad 7.2$$

where η is the position of the free surface relative to the mean water level, h is the $1.03m$ to the tank bottom, and α is the VOF indicator variable. The surface elevation is in equation 7.3 a function of x, but as this study is of piston mode only, the surface elevation should ideally be uniform across the moonpool. By taking this integral in three different probing locations, and averaging over the width, the variation with x, if any, can be filtered out. Note that this coupling between η and x is equivalent to sloshing modes. As Figure 7-7 show, the sloshing is minimal. Even in the cases with appendage that induced transverse motion, the motion of the surface is dominated of the vertical motion only, see Figure 7-8.

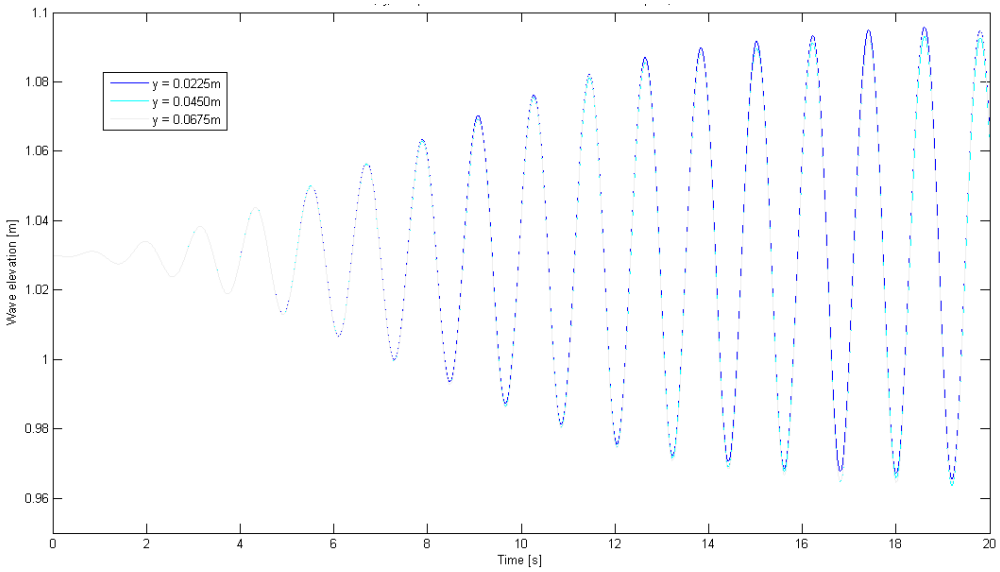


Figure 7-7: Wave elevation inside the moonpool at resonance, sampled at three locations spread over the whole width. Simulation with no appendage, excitation period $T = 1.19s$*, mesh 3. Note that the three graphs are totally overlapping.*

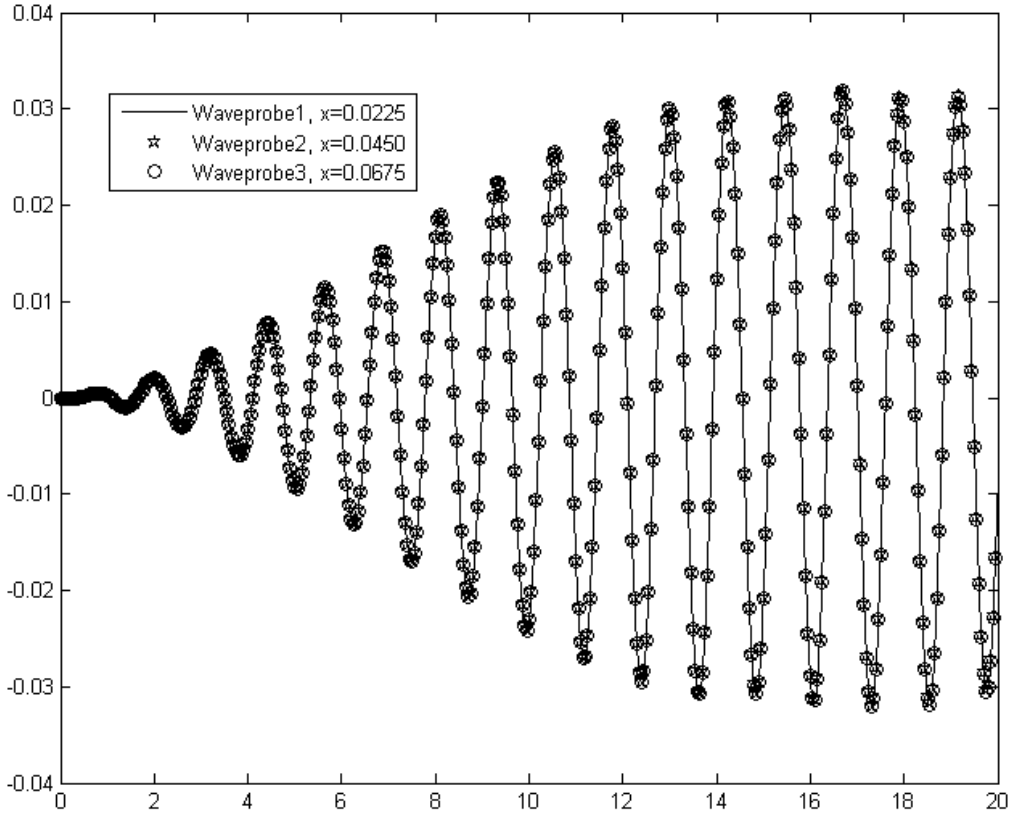


Figure 7-8: Wave elevations at different probes with medium appendage. Taken with mesh3 and excitation period T=1.05. The largest difference in the measured η *was* $\sim 10^{-4}$

To compare the sample series to that of experiment, it is wanted to extract one characteristic wave height of the stable flow. In the experiment, this was done around 120 seconds too be sure, but it was seen in that wave heights stabilized after around 20 second. In an experiment 20 and 120 seconds is not very different. Numerically however, it could easily mean difference in a day of computational time. So, as short run as possible is wanted, while being sure the flow has stabilized. At first, the wave height was simply picked as the single largest wave in a 20 second run. Of course, waves behave stochastically, and outcast waves will occur. Therefore, some mean of ensuring statistically correct wave height must be implemented, and the simplest is to take an average over a series of crests and troughs. Obviously, taking an average needs a set of data, so the run was increased to 30 seconds. Finally, the maxima between 17 second and 30 seconds were picked out, and gave the averaged wave height, see Figure 7-9.

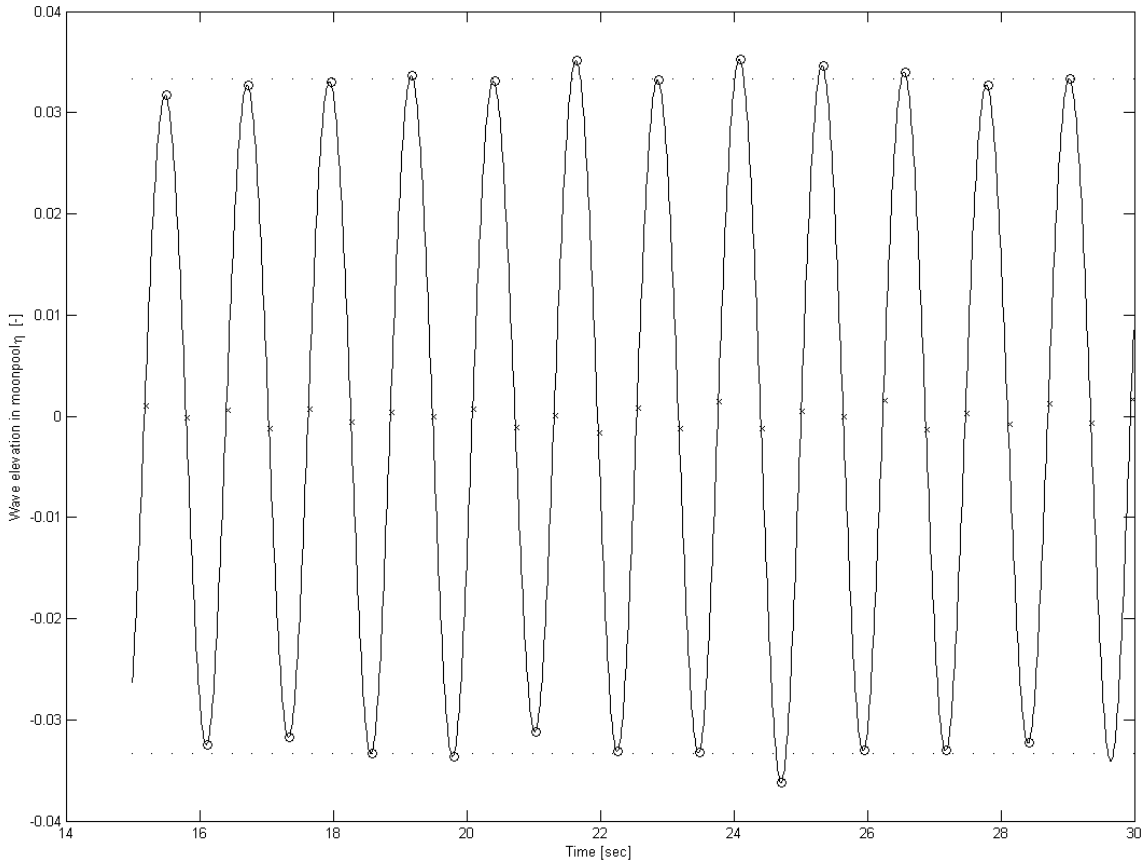


Figure 7-9: Averaging of a time series. The maxima are located with circles, the dotted lines are the given average as it is calculated. The crosses represent the zero-upcrossing, used to locate a extreme value between.

7.2.2. Boundary layer

An accurate estimate for the boundary layer thickness is difficult to obtain. Even after the simulation is run, the accuracy of the solution is dependent on the chosen resolution of boundary layer, which might have been to coarse. An approximation is given for an oscillatory plane (Kristiansen T. , 2009), as stated in section 5.2.2,

$$\delta \sim 4.6\sqrt{\frac{2\nu}{\omega}} \sim 3-6\ mm \qquad 7.3$$

or a flat plate with laminar incompressible flow, (Emanuel, 1994, p. 231),

$$\frac{\delta}{x} = \frac{4.910}{\sqrt{Re}} \sim 0-10\ mm \qquad 7.4$$

The boundary layer thickness is dependent on the position on the hull and the frequency of oscillation. Sing that 7.3is a sub domain of 7.4, a characteristic thickness of $\delta \sim 3-6mm$ will be used in the following.

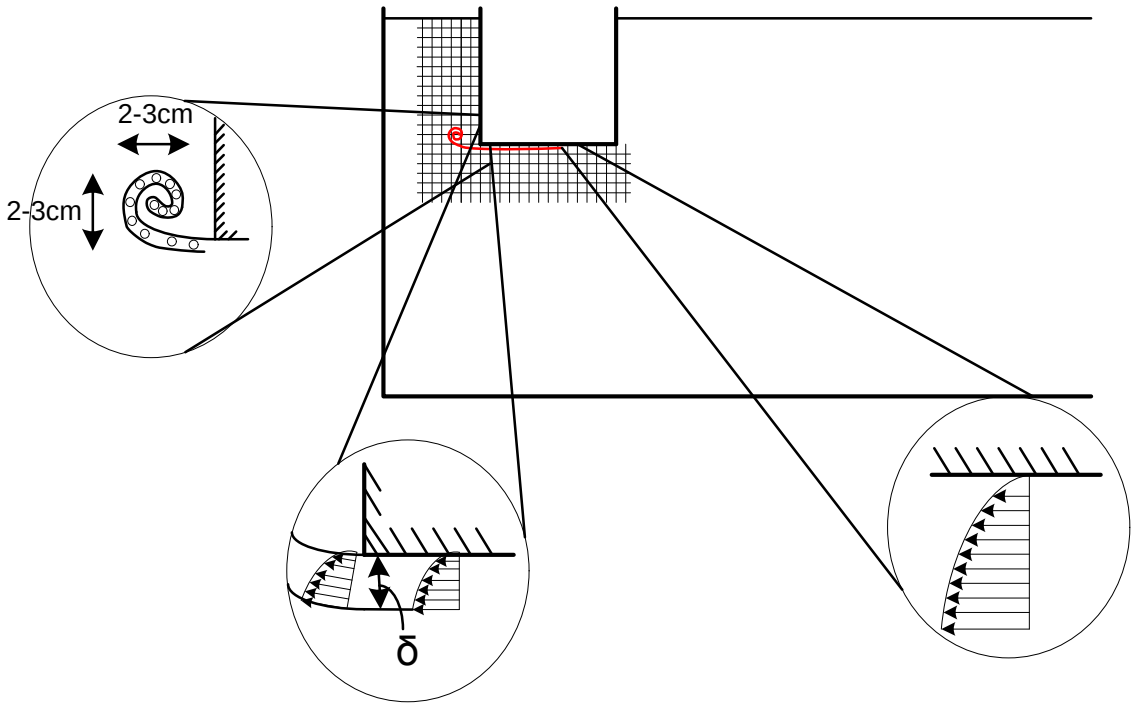


Figure 7-10: Boundary layer, separation point and vortex shedding in the moonpool. Illustration of what qualities are required of the mesh to capture the different phenomena.

As a rule of thumb, a boundary layer should be represented by 10 elements in the direction normal to the surface with close to orthogonal elements to properly capture the parabolic shape of the velocity field (see Figure 7-10). In addition to, to capture the free surface of moving waves, a general recommendation of 60 elements per wave length and 20 element per wave height.

7.3. Plain square moonpool

In table 7.2 the different meshes ran for the square, no appendage case is gathered. As it can be seen, none of the used meshes are sufficient to adequately represent the demand on 10 elements inside the boundary layer. A boundary conforming mesh was tested, with 11 elements within 2mm. The result was amazing, about 8% improved relative accuracy and 0.5% absolute error, on a mesh with 40000 elements for this boundary conforming mesh. However, refining the mesh further gave unphysical results and an unnaturally high wave elevation. It was unknown exactly what the cause was, but it was assumed to be skew mesh added to the moving boundary, which made ill-shaped elements causing spurious flow. As a result, the regular grid worked was used in the continuing as it produced reasonably well accuracy. The malfunction was not looked into further, and is a task for further work. The reason was that the author was told this boundary conforming mesh had known compatibility issues, so it was tried to avoid skew elements at all cost, meaning little cell spacing close to the boundary. A first attempt was made at a very dense mesh, and got spurious result with poor accuracy, so the 'suspicion' was confirmed. On a second attempt months later however, with a coarser mesh, it suddenly worked a whole lot better. However, at that point it was too little time left to do a full study with the mesh. Experimenting with this, and unveiling the limit for unphysical flow could vastly improve the accuracy. This could actually even be done without refining further, as an accuracy of 0.5% at resonance is more than enough. Similar results was gained on a couple of other forcing periods, on the same mesh with errors ~0-2%.

Mesh #	# volume elements	Largest elements size at boundary	#element inside BL	#element in vertex shedding	# elements in one wave height	Runtime [in CPU-hours]
1	~10000	10 mm	~0.5-1	2	5	~1
2	37104	5 mm	~1	4	10	~5
3	149235	3 mm	~1-2	6	18	~24
4	380709	2.5 mm	~2	8	20	~168
5	1523642	1.25mm	~4	16	40	~1000*

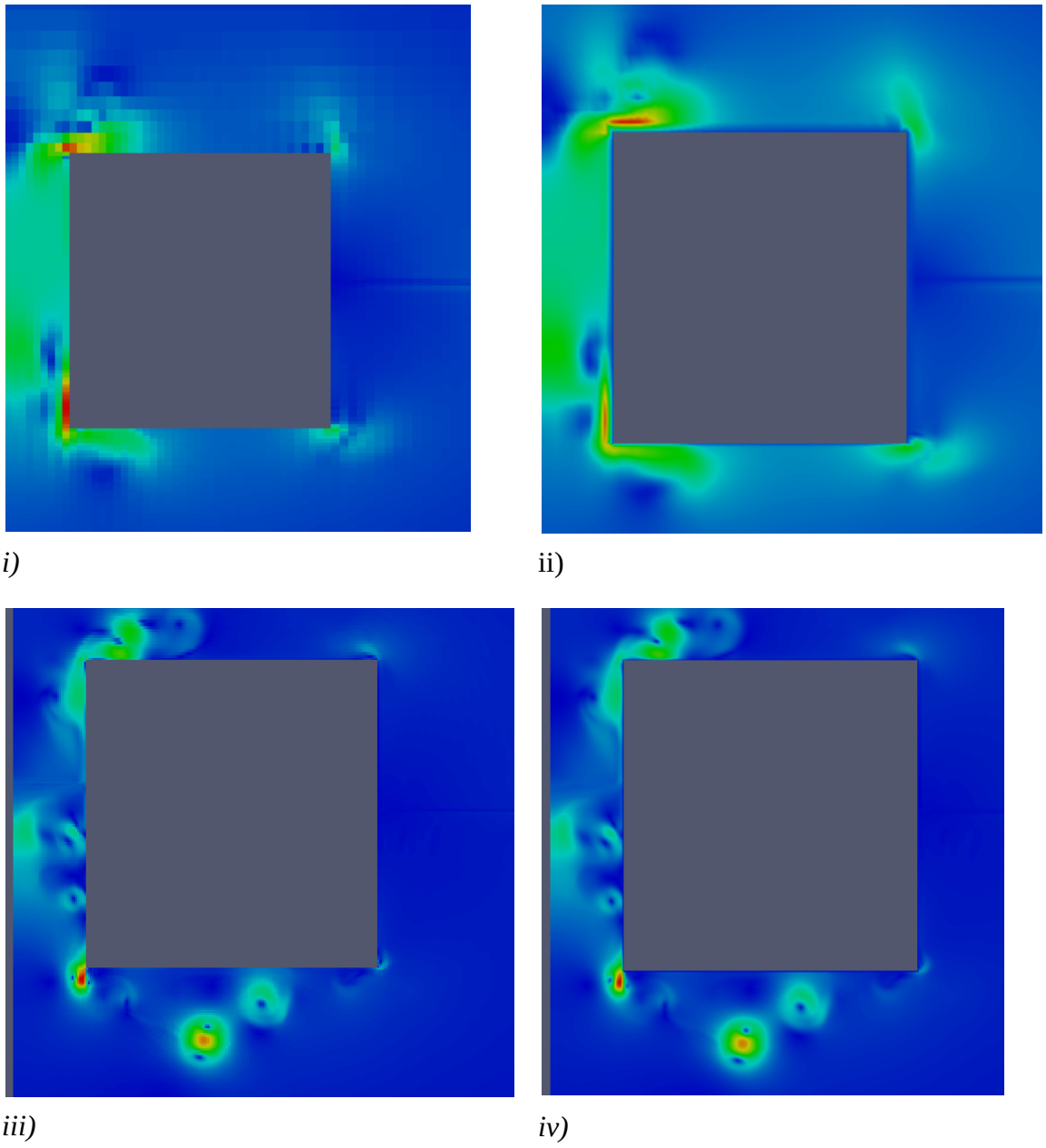
Table 7-2: Different mesh size and run time for the square moonpool case. Case 5 never finished, and used more than 2 weeks.

From the rule of thumb, it should not be feasible to solve with so few elements inside the boundary. However, quite good results are obtained anyway. A possible explanation could be that the boundary layer itself is not dominating the damping, meaning viscous damping on the wall is not the main contribution to damping in this case. Kristiansen (2009) investigated the matter, and came to the conclusion that viscous damping on the hull was neglectable compared to the vortex shedding. Therefore, it might be that accurately capturing the vertices and wave damping alone is enough to get a good numerical representation of the damping.

To capture the vertices correct, the correct location of the spawning of the vertex plus the whole path of the vertex during its entire lifetime should be captured. As the separation point is forced by the sharp corner, the viscous flow will be in the vicinity of the corner, in and around the moonpool corner, and also around the bilge facing away from the moonpool. Expansion ratio grids function poorly, since the vortices travel over a large area. No single place is proper to space against as opposed to boundary layer; hence a uniform fine grid is necessary. No recommended resolution was known to exist in literature, but 20 elements in each direction of a vertex seem reasonable, based on similar recommendation of the boundary layer and wave height/length. The critical size to capture is the abrupt velocity change for maximum flow inside the vortex and nearly still water outside, so enough elements to differentiate the border of the vertex is the requirement stated in words. Note that the number 20 is just a guess, and secondly it could not met by any of the meshes here. This means, even if a boundary conforming mesh was used, it would lack the ability to reproduce the vertex, hence more elements is needed in both cases anyway.

Figure 7-11: Overview of the generated waves, and the free surface. Red is water, blue is air, and between is the interface.

In Figure 7-12 the absolute magnitude of the velocity is shown, giving a good indication for vortex shedding. For the coarser mesh 2, vortices don't develop till this stage ($t{\sim}15sec$), while for the fine grid 4, there are several visible. Another important detail is that while ii) looks very smooth, it is not as accurate as it looks. On the other hand, iii) and iv) are almost identical, meaning that the mesh is so fine that interpolation does not add anything.



i) ii)

iii) *iv)*

Figure 7-12: Velocities magnitudes at the same time instant (T=15.04sec) for different mesh and visualization options. i) mesh 2, without interpolation of results, ii)mesh 2, with interpolation, iii) mesh 4 without interpolation, iv) mesh 4 with interpolation

In Table 7-2, the runtimes is given for different mesh sizes. Note that these times are just approximations, taken as the time when the final time-step-folder was created minus the first time-step-folder. Also note that runtime scale linearly with problem size up to mesh3. At mesh 4, the runtime increases heavily. This is thought to be due to memory deficiency, e.g. the calculated matrix in the conjugate gradient method can no longer be stored in cache, and communication with the RAM is needed. Mesh 3 uses about 70 MB memory, Mesh 4 about 270MB, so it might be a vector operation that just fitted in L3 cache now exceeds it, and must be stored in the regular memory. Equally, at mesh 5 there is even further increase in expense for time per element. Here, over 2 GB memory is used, and could cause overflow of the RAM, such that the hard drive must be used to temporally hold data. The further away from the CPU memory is stored, the more expensive it is per floating point operation (Douglas, Haase, & Hu, 2000). Again, supercomputing could deal with this problem, and allow to solve larger meshes, not to mention solving in a fraction of the time as more cpu-hours are available, as discussed in section 5.4.

In Figure 7-13 the comparison between experiments and numerical data for the square moonpool case is shown. Note that mesh 4 and mesh 5 was to computational expensive to produce a whole series, so is not included. First of all, qualitative agreements are quite good, but quantitatively a clear tendency of under predicting the damping is observed numerically. It can be several reasons for this; the most obvious is that not enough elements are used. The problem with this explanation is that mesh 3 should be superior to mesh 2, but this is only true for frequencies below resonance. It is instead thought that the finer mesh generates more spurious currents. Looking at Figure 7-14, it is clear that the wave energy grows the closer it gets to the end zone, thus spurious energy is introduced to the system. The reason why the errors grow with mesh refinement is not fully understood, especially since the mesh with boundary conforming grid gave such great result, adding elements in the boundary layer should get closer to this results. One proposal might be lack of wave damping when refining the mesh: In the far field, the horizontal direction is refined, just as much as the rest of the grid to ensure numerical dissipation of energy even at fine grids. However, this will introduce higher expansion ratios between far and near field, which also cause spurious flow. Another possible explanation might be that the finer mesh smears the interface, hence increasing numerical diffusion in the convection term of α numerical dissipation of waves fail with mesh refinement. In Figure 7-15 spurious current is shown near the hull. The current seem small, but yet significant. Yet another explanation could also be that if the same amount of spurious current is generated per cell, doubling the cell amount would double the error. Either way, according to Harvie et al. (2006) a series of numerical experiments performed using a VOF algorithm have shown that the magnitude of generated spurious currents can increase with increasing mesh refinement.

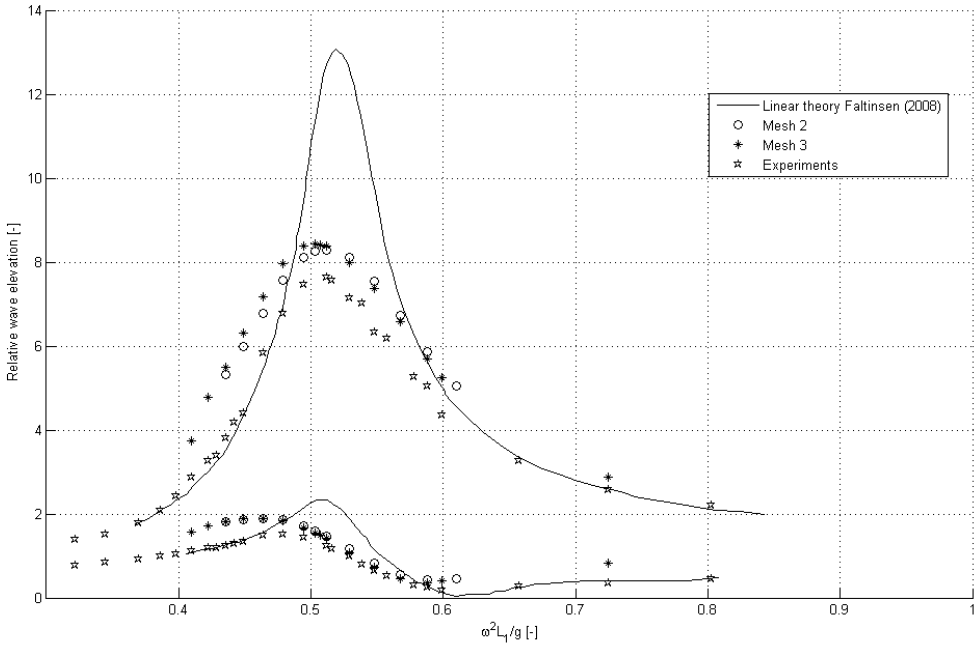
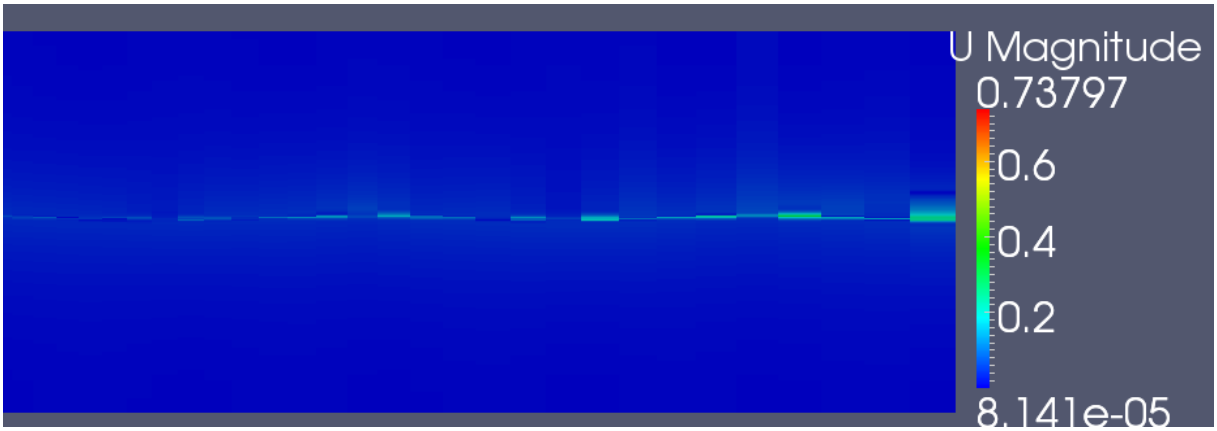


Figure 7-13: Comparison of mesh resolutions with experimental data and linear analysis, for the square moonpool case. The top graph is elevation inside the moonpool, the lower graph is the wave elevation 0.70m away



i)

ii)

Figure 7-14:Spurious reflected waves from mesh 3 outflow i) regular ii) interpolated

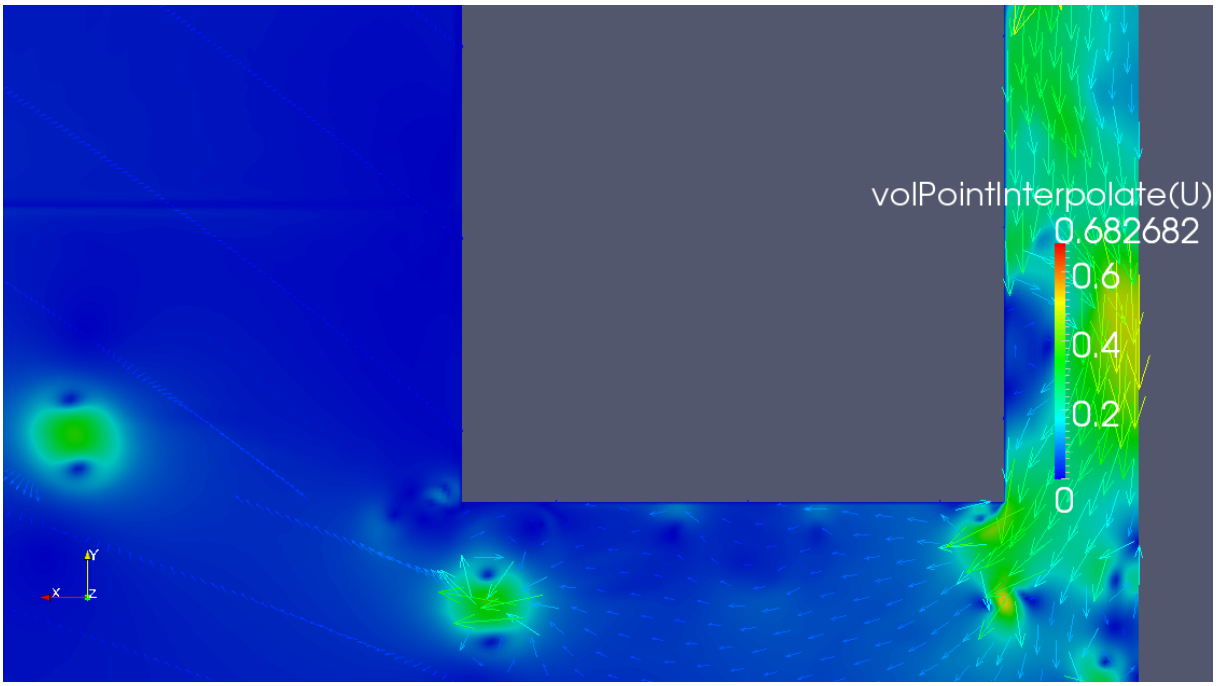


Figure 7-15: Spurious current near the moonpool, see the diagonal collection of velocities. Taken from mesh 4

Regardless of the reason for the divergence with mesh refinement, the spurious current is probably not responsible for all the deviation. The deviation about 10% at resonance is nearly constant independently of frequency. This indicates a bias error. Reasons for the error might be inadequate modeling of the boundary layer. 3 different cases was run with the same boundary conforming grid, all resulting in absolute errors of $\sim 0-2\%$. Clearly, this is the main cause for the discrepancy. Since all modeling is done with inadequate boundary layer, this can be said to be a bias error. However, it cannot explain why adding more elements, thus resolving the boundary layer more between mesh 2 and mesh 3 gave worse results. This can only be accounted to spurious currents and wave energy reflected. The magnitudes of these errors remain unknown, but from literature are generally small compared to other governing flow. Notice that the growth of the error between mesh 2 and mesh 3 is minimal compared to the error between either mesh. Another possible cause for the discrepancy is that the assumption of symmetry is not accurate, but this can hardly be thought to have that big effect. Other effects not accounted for was 3D flow effect in a small proportions between the model and the tank wall in experiment. This accounts for some extra damping in the experimental case. The magnitude of this damping is however entirely unknown.

7.4. Medium appendage

Only mesh 3 was run as a series for this case. It had the same specification as the square case, see Table 7-2 for details. Visually, it solutions looked physical and a variety of small and large vertex was formed, see Figure 7-16. When compared to experimental data the results were again qualitatively very good, see Figure 7-17. Qualitatively it is just ok, with an

overestimate of the wave elevation by about 20 % at resonance, and even more for the longer periods, as it did without appendage. This time it seems even clearer that there is a bias error between the measurements, as it remains a close to constant discrepancy for all frequencies. Again, the major error is believed to be lack of boundary elements, and the growth of the error between meshes is some form of numerical artifact; unknown which. Again, it could also be accounted by 3D effects and asymmetric effects.

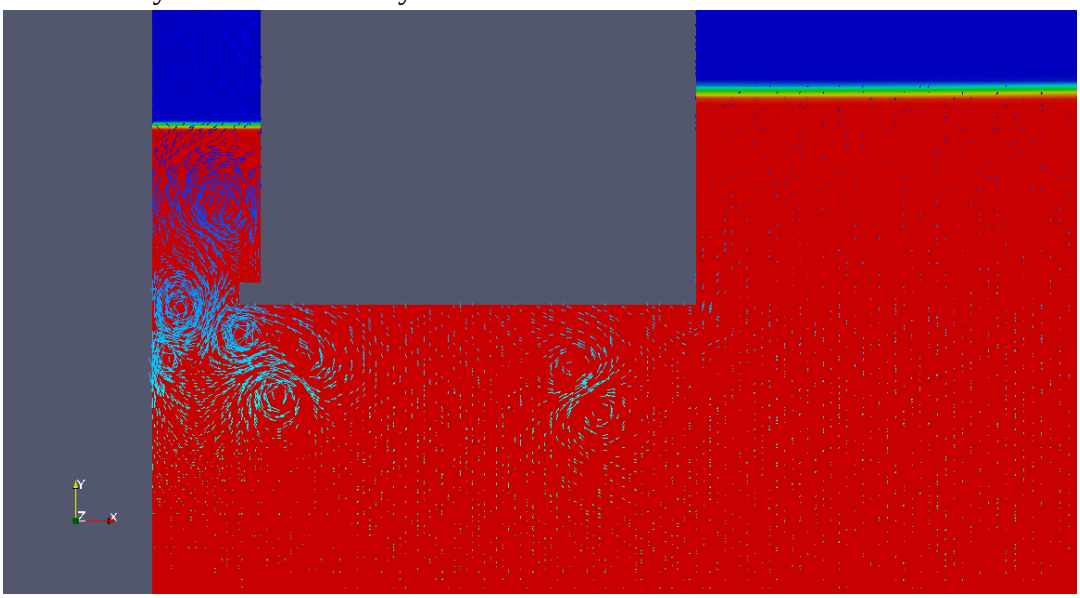


Figure 7-16: Illustrative snapshot($t = 20.0\text{sec}$) of flow around medium appendage. The background coloring is the free surface indicator α, with velocity vectors on top. Notice four clearly visible vortices. Taken at resonance, forced oscillation T=1.1225sec.

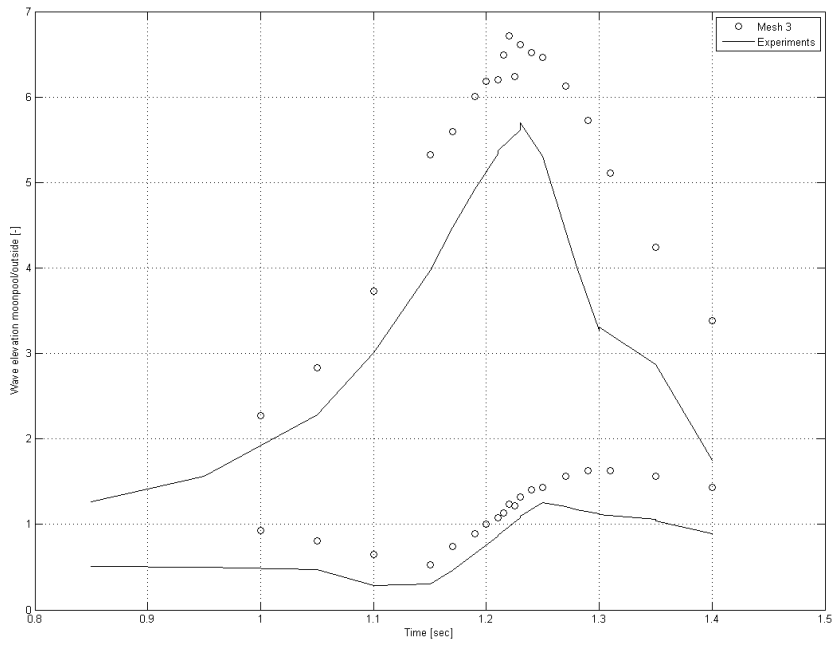


Figure 7-17: Comparison of experients and mesh 3. The top graph is wave elevation inside the moonpool, the bottom is wave elevation 0.70m outside.

7.5. Large appendage

With the large appendage, no experimental data was available. This mesh was run only on mesh 2, see specification in Table 7-2. Due to the appendix sticking out far, the wave probing was done at $x = 0.300$ instead. Still, sloshing was not recognized at the surface level. Figure 7-18 show typical velocity flow field around the appendage.

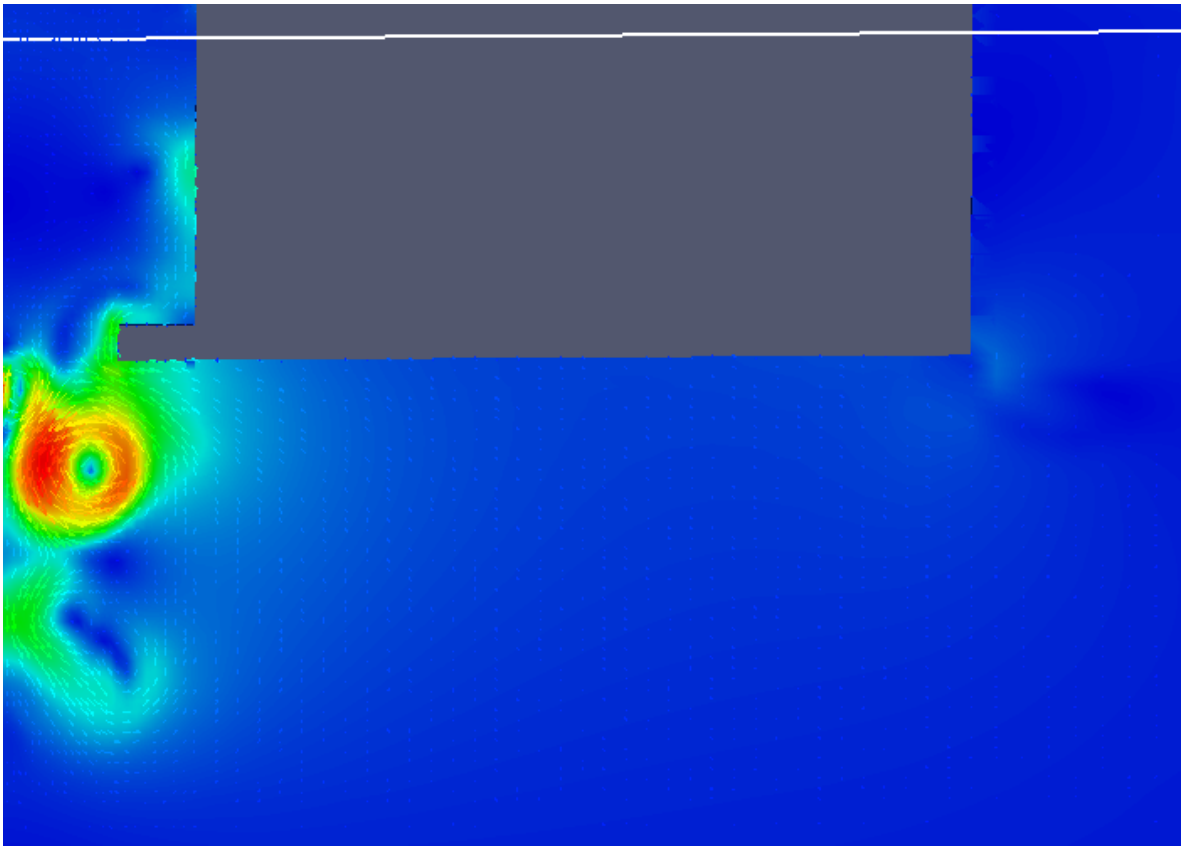


Figure 7-18: Vertex shedding with large appendage. Contour of velocity, where red is max, blue zero. Glyphs represent direction and magnitude of velocity.

The response of the moonpool with large appendage is by far the configuration with best sheltering abilities, see Figure 7-19. The maximum response is only 4.5 of the external load, but this comes at the cost of having a broader response spectrum. Considering this CFD method has over predicted the wave elevation inside the moonpool by $10\% - 20\%$, this should also be expected in this case, giving even more favorable properties.

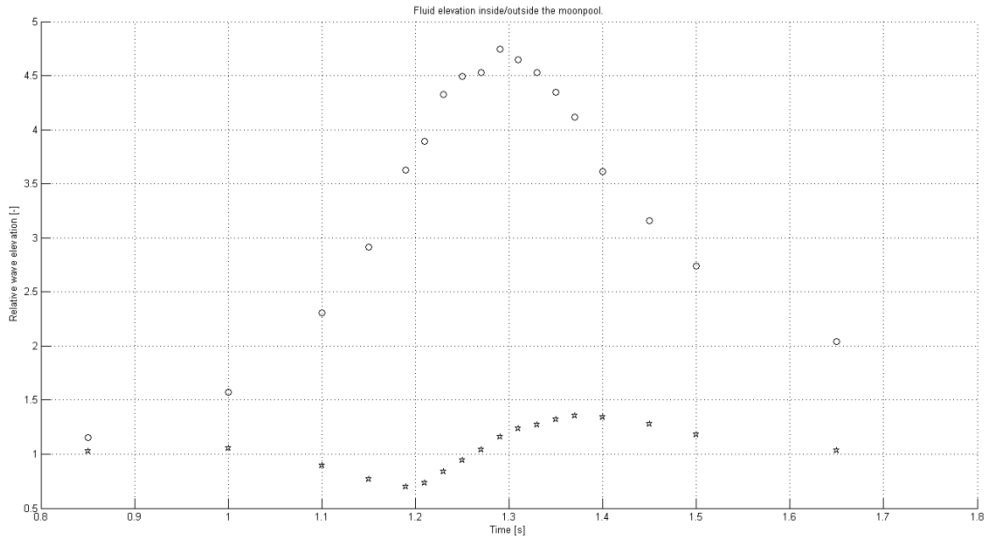


Figure 7-19: Overview for wave elevation versus forcing of time for large appendage.

7.6. Moonpool with 'skirt'

This case is more of a test case than a practical example of a moonpool design. If a ship was to have a set of skirts, it would immensely increase drag in transit, without a set of adjustable flaps. In that case, the flaps could be used to change the eigenperiod of the moonpool to adjust to a another sea state. The idea is to push up the resonance frequency by trapping more water e.g. mass into the piston-mode. Adding the skirts is equal to changing the draft. It is seen in Figure 7-21, that the effect is very similar to moving the resonance from $T = 1.19$ in the no appendage case, to $T = 1.25$; little variation in wave amplitude and damping is seen. This can be seen as an indicator that the form of the moonpool has a bigger impact on the maximum wave elevation inside the moonpool, than the width, length, freeboard and draft alone. In Figure 7-20 the time series of resonance is shown, and it is seen not to be stationary yet after 30 seconds. It could have been wise to run the simulations longer, to see if the elevations stabilized more, but since no experimental data was available to verify against, this was neglected.

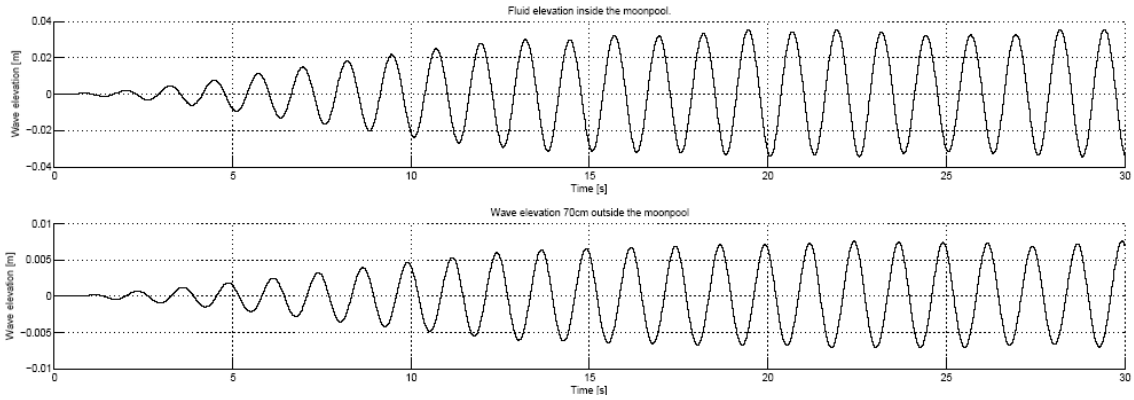


Figure 7-20: Timeseries of 'skirt' configuration at resonance $T = 1.25sec$

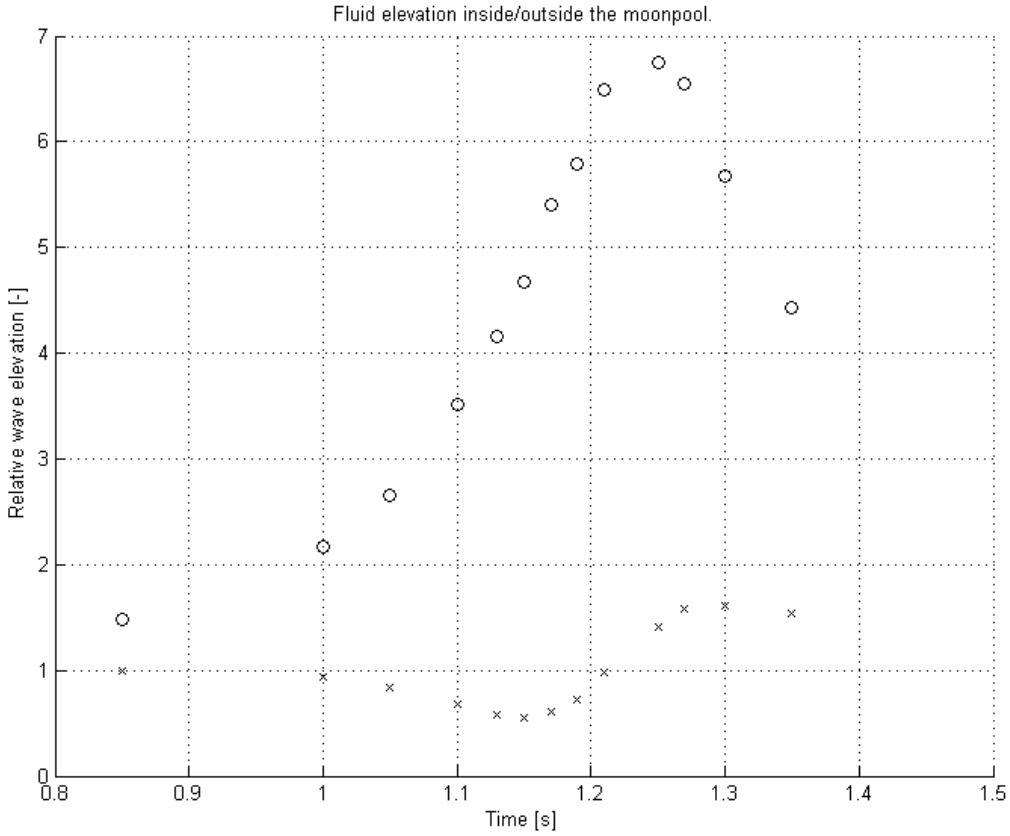


Figure 7-21: Overview of moonpool with 'skirt'

8. Closure

In this paragraph, any loose threads will be sewn together. First a summary of findings will be presented in section 8.1. Then work that has not been looked into and other ways of attacking the same problems is looked at in section 8.2, before the closure is completed where the introduction started in section 8.3; fit this piece into the full domain of marine technology. The future for designing is briefly discussed.

8.1. Summery of findings

Prediction of resonance phenomena in a moonpool is fairly good with OpenFOAM using the VOF method. Qualitatively, wave elevation is captured well with respect to frequency of oscillation. However, the quality is not as good, with a about $10\% - 20\%$ from experiment data, varying between cases. This error was almost constant with respect to absolute difference (about 2mm), which is a strong indication that the same error repeats itself in every experiment. Such an error is known as bias error. Given the uncertainties in the experiment and that of discretization and numerical solution, 10%-20% difference is acceptable. What is not acceptable is that increasing the mesh density did not monotonically increase accuracy.

According to ITTC definition, the verification was diverging. An increase in mesh density provided in some cases worse results. The results got about a relative percentage of 1%, which is not much, but still this is a bad characteristic for a method. The reason for divergence was not properly located, but clues point to numerical diffusion through the free surface, numerical currents introduced by the far field, and too few elements to capture the boundary accurately. The two last are thought to be the larger portion of the bias error.

Since all cases overpredict the wave elevation, the errors origin from poor capturing of damping. As increasing the mesh did not improve results, vortex capturing is thought to be off satisfactory degree. However, boundary conforming grid showed the possibility to improve the result dramatically in the few cases it worked, to almost exactly experimental result ($0 - 2\%$ error). Using the boundary mesh introduced unphysical effects when refined to much, and was not looked into due to time issues, and that the regular mesh worked presentably. Wave beaches were implemented in a crude way, by relieving energy through numerical diffraction, which showed to introduce energy back into the system, especially for finer meshes.

Despite the apparent problem with damping, the added mass seems fairly accurately predicted, and the resonance period is well predicted. As the predictions of the simulations are conservative, it is still potentially useful as a supplement to experiments.

It should be noted, as this method stands there exist potential solvers with added logic to capture vertices such as the non-linear Boundary element method coupled with a vortex tracking method presented by Kristainsen and Faltinsen (2008). This method currently predicts wave elevation more accurately than the presented method, and in a fraction of time. However, with better preprocessing tools to allow a non-skew boundary conforming grid

result improve beyond that of Kristiansen. A better implementation of wave beaches is also important to let refinement of the grid to decrease the accuracy instead of increasing it. VOF methods with RANS has shown excellent results, relatively fast (Krüger, Manzke, Rung, & Vorhölter, 2009; Lu, Cheng, Teng, & Sun, 2010).

8.2. Further work and alternative approaches

Two major things need improvement from the following CFD approach to get accurate results. First thing is to look into is how to implement a numerical beach in a RANS or DNS solver that couples with VOF-method and is not influenced by grid refinement. Energy reflected from the outflow boundary is thought to be a major contributor to the corruption of the solution at grid refinement.

Equally, or even more important, is a clever way to implement 10 or more elements within the boundary without causing to skew mesh or to big ratio between elements that the solution is corrupted is wanted. Added to the difficulty here is moving mesh, so a preprocessor that let you define the grid movement with time should be of great aid to getting a quantitative agreement between calculations and experiments. Unstructured mesh is a powerful tool to get accurate grids. Unstructured meshes generally work very well on finite volume method compared to other discretization techniques. Unstructured meshing could also overcome the problem were structured mesh either gives to rapid expansion or too skew elements in transition between areas of dense and coarse mesh. As no preprocessor that provides unstructured grids for OpenFOAM is freely distributed and known to the author, this was not attempted. Unstructured grids could bring very good accuracy-per-element ratios in this setting, as elements could more easily be placed were they are needed, including more in the boundary layer. Also, possibilities for automatic mesh generation open with unstructured grids.

A natural extension of this work is to implement the code in 3D. In this study all 3D effects has been neglected, which could possibly account for much of the bias error. The experiment had visible 3D-effect flow in the tiny 2.5 cm gap on each side of the model. Although limited amount of fluid can flow here, it is enough to induce vortices in the cross direction, which add damping in the experiment. This damping is completely ignored in the current 2D numerical simulation.

A similar extension would be to examine the assumption of symmetry. This may seem less necessary; however it is unlikely that vertex shedding would act in a perfectly symmetric manner across the moonpool. As matter a fact it would be reasonable to believe that some form of coupling between vertices shed on the right corner and the left, thus providing synergy, and increasing the damping. In the current numerical code, this is completely neglected.

Several examples in the literature have shown good description using RANS and the Level set method to accurately capture free surface flow including turbulence. To this date, this is not possible out-of-the bow with OpenFOAM. Advancement has been made, and subsequent versions of OpenFOAM might get these tools, or it could be implemented. This is however

just moving the problem from one to another, as the level set method has similar deficiencies as VOF.

Another possibility to avoid the meshing problems, mesh-free numerical techniques exist, such as Smooth Particle Hydrodynamics. Again this just moving the problem, but seeing as meshing was a cause to error in this work, it might be better solved with a mesh-free technique.

8.3. Concluding remarks

Numerical simulation using CFD provides great possibility to run series of test a short time, and varying design can be done quickly; more so than possible by conventional methods today. However there are numerous limitations, a few has been covered in the current work. Mesh generation, turbulence modeling and free surface are all complex topic, that add an extra layer of difficulty. For instance VOF method has a problem with especially thin interface on fine grids, causing an abrupt jump in α, which again causing spurious currents by numerical diffusion. Nonetheless, the simulations proved to be in good qualitative agreements and also fairly good quantitative agreement with experiments.

If this study has proved anything, it is that the direct numerical solver of OpenFOAM is not a brute force way of getting the correct result, but is a nice addition to experiments and needs careful tuning to produce accurate, physical correct results. Knowing what to look for is the key ingredient you finding it. Further, numerical simulations could in this study fairly well predict the performance of the hull configurations, although the accuracy was off by more than 10%. Thus CFD has good potential in being used early in ranking designs in marine hydrodynamic settings where damping is important. Presently, little full-automatic tools of this sort exist, as there are well established design techniques that can do design at a bit higher cost and time, but at greater accuracy. As CFD develops further, it will guaranteed be more robust, more accurate and can be applied on more fields in commercial settings, including problems related to free surface and turbulence.

Bibliography

Baker, G., Meiron, D., & Orzag, S. (1981). *Application of a Generalized Vortex Method to non-linear Free Surface flows.* Paris: 3rd Int. Conf. Numerical Ship Hydrodynamics.

Chen, X. (2005). *Hydrodynamic analysis for offshore LNG terminals* . Rio de Janeiro: Proceedings of the 2nd offshore hydrodynamics symposium.

Chung, T. J. (2002). *Computational Fluid Dynamics.* Cambridge University Press.

Clement, A. (1996). Coupling of two absorbing boudary conditions for 2D time-domain simulations of free surface gravity waves. *J. Comp Phys vol 126* , 139-151.

Cramer, H., & Krüger, S. (1999). *CFD within the ship design process.* Flensburger Schiffbau Gesellschaft.

Douglas, C., Haase, G., & Hu, e. a. (2000). *Portable Memory Hierarchy Techniques For PDE Solvers: Part I.* Munchen: SIAM News.

Drobyshevski, Y. (2004). Hydrodynamic coefficients of a two-dimensional, truncated rectangular floating structure in shallow water. *Ocean Engng 31* , 305–341.

Duncan, J. (1983). The breaking and non-breaking wave resistance of a wave two-dimensional hydrofoil. *J Fluid Mech 126* , 507–527.

Emanuel, G. (1994). *Analytical Fluid Mechanics.* CRC Press: Boca Raton.

Faltinsen, O. M. (1990). *Sea Loads On Ships and Offshore Structures.* Trondheim: Cambridge University.

Faltinsen, O. M., Rognebakke, O. F., & Timokha, A. N. (2006). Two-dimensional resonant piston-like sloshing. *J. Fluid Mech.* , vol. 575, pp. 359–397.

Flow Science. (n.d.). *Free-Surface Modeling Methods*. Retrieved April 2010, from FLOW3D: http://www.flow3d.com/cfd-101/cfd-101-free-surface-flows.html

Gaillarde, G., & Cotteleer, A. (2005). *Water motion in moonpools Empirical and Theoretical approach.* Maritime Research Institute Netherlands.

Gorski, J. J. (2002). *Present State of Numerical Ship Hydrodynamics and Validation.* Götenborg: ASME.

Graham, J. M., & Kendon, T. E. (2008). Viscous Damping of Large Floating Structures. In E. Kreuzer, *IUTAM Symposium on Fluid-Structure Interaction in Ocean* (pp. 69-78). Hamburg: Springer Science.

Harlow, F., & Welch, J. (1965). Numerical calculation of time-dependent viscous incompressible flow of fluids with free surface. *Phys Fluids vol 8* , 2182–9.

Harvie, D., Davidson, M., & Rudman, M. (2006). An analysis of parasitic current generation in Volume of Fluid simulations. *Applied Mathematical Modelling Vol 30* , 1056-1066.

Hirt, C. W., & Nichols, B. D. (1981). Volume-of-fluid (VOF) method for dynamics of free boundaries. *J Comput Phys* , vol 39:201–21.

ITTC. (2001). *ITTC Quality Manual.* Resistance Committee of 23th, ITTC.

ITTC. (1996). Report of the Resistance and Flow Committee. *Proceedings of 21st Conferance* .

Jasak, H. (1996). *Error analysis and estimation in the Finite Volume method with applications to fluid flows.* Imperial College, University of London.

Jasak, H. (2006). *Numerical solution algorithms for compressible flows: Lecture Notes.* Croatia: Tech. rep. University of Zagreb.

Jasek, H., & Tukovi, Z. (2004). *Automatic Mesh Motion for the Unstructured Finite Volume Method.* London: Nabla Ltd/University of Zagreb.

Kreyszig, E. (1999). Advanced engineering mathematics 8th ed. Singapore: John Wiley & Sons.

Kristiansen, T. (2009). *Two-Dimensional Numerical and Experimental Studies of Piston-mode Resonance.* Trondheim: NTNU.

Kristiansen, T., & Faltinsen, O. (2008). Application of a vortex tracking method to the piston-like behaviour in a semi-entrained vertical gap. *Applied Ocean Research vol 30* , 1-16.

Krüger, S., Manzke, M., Rung, T., & Vorhölter, H. (2009). *Introduction of RANS-CFD into the Initial Design Process.* Hamburg: Hamburg University of Technology.

Kuznetsov, N., Maz'ya, V., & Vainberg, B. (2002). Linear Water Waves. A Mathematical approach. *Cambridge University Press* .

Kuznetsov, N., McIver, P., & Linton, C. M. (2001). On uniqueness and trapped modes in the water-wave problem for vertical barriers. . *Wave Motion 33* , 283–307.

Larsson, L., Bertram, V., & Stern, F. (2000). Proceedings of Gothenburg. *A workshop on CFD in ship hydrodynamics* .

LeVeque, R. (2002). *Finite Volume Methods for Hyperbolic Problems.* London: Cambridge University Press.

Lu, L., Cheng, L., Teng, B., & Sun, L. (2010). *Comparison of Potential Flow and Viscous Fluid Models in Gap Resonance.* China: International Workshop on Water Waves and Floating Bodies.

Maisondieu, C., & Clement, A. (1993). *A realizable force feedback-forward control loop for a poston wave-absorber.* St. John's Newfoundland: 8th Int. Workshop on Water Waves and Floating Bodies.

Maisondieu, C., Molin, B., Kimmoun, O., & Gentaz, L. (2001). Simulation bidimensionalle des écouleements dans une baie de forage: étude des mode de résonance et des amortissements. *Acte des Huitémes Journées de l'Hydrodynamique* , 251-264.

Mascio, A., Broglia, R., & Muscari, R. (2007). On the application of the single-phase level set method. *Computers & Fluids 36* , 868–886.

McIver, M. (1996). An example of non-uniqueness in the two-dimensional linear water wave problem. *J. Fluid Mech vol 315* , 257–266.

McIver, P. (2005). Complex resonances in the water-wave problem for a floating structure. *J. Fluid Mech vol 536* , 423-443.

Medina, P. R. (2008). *Study and Numerical Simulation of Sediment Transport in Free-Surface Flow.* Spain: Doctor Europaeus of University of Malaga.

Moin, P., & Mahesh, K. (1998). Direct Numerical Simulation: A Tool in Turbulence Research. *Annu. Rev. Fluid Mech. vol 30* , 539–78.

Molin, B. (2001). On the piston and sloshing modes in moonpools. *J. Fluid Mechanicsvol* , vol.430, pp. 27-50.

Molin, B. (2000). *On the piston mode in moonpools.* Marseille: ESIM.

Muscari, R., & Mascio, A. D. (2004). Modeling breaking waves. *J Mar Sci Tech 9* , 158–170.

Muscari, R., & Mascio, A. D. (2004). Modeling breaking waves. *J Mar Sci Technol vol 9* , 158–170.

Newman, J. (1977). *Marine Hydrodynamics.* MIT Press.

Newman, J. N. (2008). *Analysis of wave generators and absorbers in basins.* MIT.

Newman, J. N. (2008). *Analysis of wave generators and absorbers in basins.* Massachusetts: MIT.

Nichols, B., & Hirt, C. (1975). Methods for Calculating Multi-Dimensional, Transient Free Surface Flows Past Bodies. *Proc. First Intern. Conf. Num. Ship Hydrodynamics* , 20-23.

Noh, W., & Woodward, P. (1976). SLIC (Simple Line Interface Calculation). *Proceedings of Fifth International Conference on Fluid Dynamics vol 59* , 330-340.

OpenFOAM. (2004-2010). *Breaking of a dam*. Henta May 2010 frå OpenFOAM: The open source CFD toolbox: http://www.openfoam.com/docs/user/damBreak.php

Patankar, S. V. (1980). *Numerical Heat Transfer and Fluid Flow.* London: Hemisphere.

Pauw, W. H., Huijsmans, R. H., & Voogt, A. (2007). *Advances in the hydrodynamics of side-by-side moored vessels.* San Diego: Proceedings of the 26th International Conference on Offshore Mechanics and Arctic Engineering.

Pope, S. B. (2000). *Turbulent flows.* Cambridge: Cambridge University Press.

Rhee, S., & Stern, F. (2002). RANS model for spilling breaking waves. *J Fluid Engineering vol124* , 1-9.

Rider, W., & Kothe, D. (1998). Reconstructing volume tracking, . *Journal of Computational Physics. vol 141* , 112–152.

Romate, J. E. (1992). Absorbing Boundary Conditions for Free Surface Waves. *Journal of Computational Physics* .

Rudman, M. (1997). 1997. Volume-tracking methods for interfacial flow calculations. *Int. J. Numer. Methods Fluids. vol 24* , 671-691.

Rusche, H. (2002). *Computational Fluid Dynamics of Dispersed Two-Phase Flows at High Phase Fractions.* London: Imperial College of Science.

Sadiq, S., Zhuang, K., & Liang, Y. (2006). Resonance in Moonpool Due to Flow Induced Oscillations near Free Surface. *ISOPE Pacific/Asia Offshore Mechanics Symposium 7* , 190.

Salvesen, N., Tuck, E., & Faltinsen, O. (1970). Ship motions and sea loads. *SNAME 78* .

Sauder, T., Kristiansen, T., & Östman, A. (2009). *Validation of a numerical method for the study of piston-like oscillations between a ship and a terminal.* Trondheim: MARINTEK.

Scardovelli, R., & Zaleski, S. (1999). Direct Numerical Simulation Of Free-surface and interfacial flow. *Annual Review of Fluid Mechanics 31* , pp 567–603.

Senocak, I., & Iaccarino, G. (2005). *Progress towards RANS simulation of free-surface flow around modern ships.* Stanford University, CA: Center of Turbulence Research.

Skamarock, W. C. (2005). Positive-Definite and Monotonic Limiters for Unrestricted-Time-Step. *American Meteorological Society* , 2241-2250.

Sussman, M., Smekerda, P., & Osher, S. J. (1994). A level set approach for computing solutions to incompressible two-phase flow. *J Comput Phys* , vol 114:146–59.

Taylor, D. W. (1943). *The Speed and Power of Ships.* Washington: U.S. Government Printing Office.Original Volume 1910. Second Revision.

Tuck, E., & Newman, J. (2002). Longitudinal sloshing in slender moonpools. *Proc. 17th Intl.*

Ubbink, U. (1997). *Numerical prediction of two fluid systems with sharp interfaces.* London: Imperial College of Science.

Unverdi, S. O., & Tryggvason, G. (1992). A front-tracking method for viscous,incompressible, multi-fluid flows. *J Comput Phys vol 100* , 25–37.

Van de Velde, E. F. (1994). *Concurrent scientific computing.* California: Springer.

van Leer, B. (1979). Towards the Ultimate Conservative Difference Scheme, V. A Second Order Sequel to Godunov's Method . *J. Comput. Phys. vol 32* , 101–136.

Vogt, M., & Larsson, L. (1999). Level set methods for predicting viscous free surface flows. *Proceedings of the 7th International Conference on numerical ship hydrodynamics* , 14-19.

Wang, Y. (1994). A study on periodic wave breaking by absorbed numerical wave channel. *Proc. 4th Intl. Offshore and Polar Engrg. Conf., Japan, Vol. III* , 32–36.

Weller, H. G. (2002). *Derivation, modelling and solution of the conditionally averaged two-phased flow equations.* London: Technical Report Naba Ltd.

White, F. (2003). *Fluid Mechanics fifth ed.* New York: MCGraw-Hill.

Wilson, R., Paterson, E., & Stern, F. (2000). Verification and validation for RANS simulation of a naval combatant. *Procedings of Gothenburg* .

Aarseth, J. B. (2009). *Numeriske Beregningsmetoder.* Trondheim: NTNU.

Aarsnes, J., & Steen, S. (2008). *Experimental Methods in Marine Hydrodynamics.* Trondheim: Institute of Marine Technology.