Extreme Wave Loading on Offshore Wave Energy Devices using CFD

Jan Westphalen

University of Plymouth

A thesis submitted to the University of Plymouth in partial fulfilment of the requirements for the degree of
Doctor of Philosophy

June 20, 2011

Abstract

Two commercial Navier-Stokes solvers are applied to wave-wave and wave-structure interaction problems leading to the final application of simulating a single float of the wave energy converter (WEC) Manchester Bobber in extreme waves and a fixed section of the Pelamis in regular waves.

First the two software packages CFX and STAR CCM+ are validated against measured results from physical tank tests concerning the interaction of 3 non-linear focused wave groups of different steepness (Ning et al. 2007). The agreement for all of these cases is very good and could even be improved from first order to second order wave setup at the wavemaker. However, in preliminary regular wave tests, the damping of the waves is identified to be an issue, which is the reason for focusing the waves and placing the structures in the following experiments approximately one wavelength behind the wavemaker.

The interaction of fixed vertical and horizontal cylinders in regular waves are simulated concerning the forces on the structures (Kriebel 1998, Dixon et al. 1979). For the horizontal cylinder non-linear force oscillations of double the wave frequency could be modelled in good agreement with physical tank data, where linearised models failed. For the vertical cylinder the problem of the secondary load cycle due to a backward-breaking wave behind the cylinder is of special interest (Stansberg 1997, Chaplin et al. 1997). Here, the horizontal forces on a slender cylinder with a diameter approximately equal to the wave height are simulated successfully. Furthermore, the highly non-linear wave run-up in front of the cylinder is resolved well in the numerical approach.

The next set of simulations includes rigid body motion. Here, the forced oscillations of a cone shaped body near the still water surface is simulated. These results are compared with test data published by Drake et al. (2008). For these cases the non-linearity of the experiments is discussed by comparing the sum and differences of the force and surface elevation time histories for a set of simulations with opposite excursion of the cone. The hydrodynamic forces on the cone surface are resolved in very good agreement. The solution of the surface elevation close to the

cone surface is also resolved reasonably well.

After having validated the codes for fixed wave-structure interaction problems and forced motion, the CFD methods are finally applied to problems relevant to the survivability of WECs. First a single float in waves is modelled. This challenging case combines the extreme wave setup with a floating body problem in one and two degrees of freedom including the interaction of the float inertia with the inertia of a separate mass attached to it. The vertical translations of the float are compared with physical tank tests by Stallard et al. (2008). This case clearly demonstrates the capabilities and challenges in using CFD to simulate WECs. When representing the pulley and counterweight system by a simplified external body force rather than the full setup, the calculated translations of the float agreed better with the measured results from the physical tank test.

Furthermore the codes are used to simulate a single fixed section of the Pelamis device in regular waves. The surface elevations close to the device are discussed and the forces acting on different strips on the structure are presented.

Contents

List of Figures

Acknowledgements

On this page I would like to thank all those people, who were involved during the process of writing this thesis, running and setting up the simulations and interpreting the results.

First of all I would like to thank Deborah for her help and advice during the last 3 1/2 years, for reading through my "German" English over and over again and giving me the opportunity to attend all the different conferences. I genuinely enjoyed the last couple of years, first in Bath and then in Plymouth.

Alison and Chris I would like to thank for adding additional expertise and opinions on this subject, which often put the problems into another perspective. Also, the proofreading of my paper drafts and thesis chapters was incredible helpful.

The people involved in the project contributed to this work by asking a lot of questions and commenting on my presentations and paper drafts during our project meetings in Bath, Oxford and Manchester. A big "Thank you" to Paul Taylor, Derek Causon, Peter Stansby, Zheng Zheng Hu and Pourya Omidvar. Many thanks to Jun Zang, Kevin Drake and Tim Stallard for providing the experimental test data that we used for our comparisons with the CFD codes.

Andrew Sunderland from HPCx also put in an awful lot of work for getting the two packages to run on the cluster. Thanks to the ANSYS and CD-Adapco support teams, for answering my questions so quickly.

Finally, this work was funded by the Engineering and Physical Science Research Council under the project title "Extreme Wave Loading on Offshore Wave Energy Devices using CFD: a Hierarchical Team Approach". Thanks for that!

Authors declaration

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award.

Relevant scientific seminars and conferences were regularly attended at which work was often presented. One papers have been submitted for publication in Ocean Engineering.

Signed: _____

Date: _____

Publications :

Westphalen, J., Greaves, D.M., Williams, C.J.K., Hunt-Raby, A.C.& Zang, J. *Focused waves and wave-structure interaction in a numerical wave tank.* submitted for publication in Ocean Engineering. **2010**

Conference presentations :

2010:

The 20th International Offshore (Ocean) and Polar Engineering Conference & Exibition, *20–26 June, Beijing, China.* Oral presentation: **Numerical Simulation of Wave Energy Converters using Eulerian and Lagrangian CFD Methods** .

Marine & Offshore Renewable Energy, *21–23 April, London, UK.* Oral presentation: **Numerical simulation of a floating body in multiple degrees of freedom** .

2009:

European Wave and Tidal Energy Conference, *7–10 September, Uppsala, Sweden.* Oral presentation: **Extreme Wave Loading on Offshore Wave Energy Devices using CFD: a Hierarchical Team Approach** .

24th International Workshop on Water Waves and Floating Bodies, *19–22 April, St. Petersburg, Russia.* Oral presentation: **Numerical Simulation of an Oscillating Cone at the Water Surface using Computational Fluid Dynamics** .

2008:

The 18th International Offshore (Ocean) and Polar Engineering Conference & Exibition, *6–11 July, Vancouver, Canada.* Oral presentation: **Numerical Simulation of Extreme Free Surface Waves.**

2007:

Understanding Marine Systems, *17th December, Plymouth, UK*. Oral presentation: **Extreme Wave Loading on Offshore Wave Energy Devices using CFD.**

10th Numerical Towing Tank Symposium, *23–25 September, Hamburg, Germany.* Oral presentation: **Comparison of Free Surface Wave Simulations using STAR CCM+ and CFX** .

Word count for the main body of this thesis : 41,000

Chapter 1

Introduction

Modern western life style is heavily dependent on the growth of industrial production, mobility and telecommunication, all of which rely on the supply of energy or electricity. If either of these should break down, it would have a tremendous effect on our life. People might not be able to commute to work, food supply might come to a still stand or the telephone and internet networks would fail, which would effect all kinds of areas immediately. Thus it is essential to secure the supply of energy and meet the demand. On the other hand the customers need to be able to afford this electricity. For example private people need to heat and light their houses, or simply want to run their TV or kitchen appliances. These two requirements of the energy sector, i.e. security of supply and the affordability of the energy are opposed by the necessity of producing electricity in a environmentally friendly way. This includes the reduction of CO_2 emissions generated by burning fossil fuels, such as coal, oil and gas, and the reduction of radioactive waste from nuclear power plants. Both of these methods of generating electricity would be cheap and meet the demand. However, fossils are running out at some point in the future and nuclear waste pollutes the environment for a very long time. Facing this trilemma of meeting the demand at a low cost and in an environmentally friendly way is the challenge the energy suppliers face today, which is the reason for developing alternative technologies, such as wind energy, solar or hydro power, to substitute the traditional energy production

methods.

The hydro energy sector includes onshore energy production, such as river or storage power stations, and coastal and offshore energy production, which can be utilising tidal stream power, offshore wind and wave energy. As the onshore hydro power production is almost saturated, the offshore sector is being developed heavily. Here, the first option was to use established technologies, such as wind turbines, and put them offshore. This was followed by tidal energy devices, which were adapted from river power stations. Over the last 30 years the development of wave energy devices caught up and is a quickly growing market with a variety of designs for wave energy converters (WEC). However, this new technology is still expensive compared to the traditional methods of generating electricity, but does not pollute the environment and has the potential to meet the demand due to the shear size of the oceans.

To be competitive the production costs of electricity by WECs need to be reduced. This can be achieved by putting a large number of wave energy converters into place, make each of them more efficient during normal operation and reduce maintenance costs, which is a key factor in offshore engineering in general. Research is being done on the efficiency of devices using physical tank tests and linearised numerical models, in which the waves are relatively small. This work, however, looks into the latter design requirement by investigating the survivability of WECs. In normal sea states this is not an issue, but when storm seas arise, the WEC may be hit by very large waves, which might damage or even destroy the device. Traditionally, this phenomenon is investigated using scale model tank tests and linearised mathematical approaches, which might neglect rigid body motion or may not be able to simulate breaking waves. Relevant literature is discussed

in Chapter 2. Computational fluid dynamics (CFD) can provide an additional tool when investigating such problems, that can model fluid-structure interaction including wave-breaking, over topping and rigid body motion in a fully non-linear manner and in full scale.

For this work two commercial CFD packages are applied to wave-wave and wave-structure interaction problems relevant for the simulation of WECs. They both solve the Navier-Stokes equations and model both fluids, water and air. The governing equations leading to the Navier-Stokes equations, the descretisation schemes used for the solvers and the solution methodologies are described in Chapter 3.2.

First wave-only cases are simulated, which are described in Chapter 4. Here, the implementation of the boundary conditions for the numerical models are described. Results for regular and focused wave simulations are discussed and compared with physical tank tests.

After that, fixed vertical and horizontal cylinders are modelled and the results are compared with physical test data from the literature in Chapter 5. These simulations concern the forces on the structures due to the waves. Non-linear effects such as the ringing on slender vertical cylinder are identified and reproduced in the simulations. This section finishes with the case of a cone, that is forced to move near the water surface following a prescribed displacement. Here, the hydrodynamic forces and the surface elevation close to the structure are compared with physical tank tests.

The final Chapter 6 discusses the application of the CFD codes to WEC related problems, such as a floating body representing a single float of the Manchester Bobber. Here, the hydrodynamics of the float in extreme waves interacting with a counterweight mass, that is connected to the float by a pulley system is modelled in one and two degrees of freedom. Furthermore, the codes are used to calculate the

forces on a fixed single section of the Pelamis WEC in regular waves.

The work is concluded in Chapter 7.

Chapter 2

Literature Review

Introduction Extreme Wave Loading on Offshore Wave Energy Devices using Computational Fluid Dynamics (CFD) involves investigations in several fields. In this section an overview will be given referring to appropriate literature regarding wave energy and wave energy devices, the theory of waves, the mathematical methods to compute the wave loading on structures and finally an overview of Computational Fluid Dynamics and the approaches available.

The devices of interest here are floating on the free surface, driven by the motion of passing waves. The aim of this study is to investigate the behaviour of these devices in extreme waves. These extreme conditions might be ocean waves of extreme height. A famous example of such a wave is the New Year wave, which passed the Draupner Platform on 1 January 1995. The Draupner Platform, an oil rig off the coast of Norway was hit by a 30 m high wave, which caused severe damage on the structure. Extreme waves might also be defined as a sea state of several regular waves of a particular period, which force the energy device in extreme ways. For example the device could be excited to oscillate close to resonance, which might damage joints due to extreme displacements. Another problem might be high frequency oscillations, similar to the ringing of vertical cylinders (see Chapter 5.1.3), where fatigue failure might occur. Another important scenario is breaking of waves over the energy device. The loading due to green water effects might become sig-

nificant especially in combination with either an extreme high wave or an extreme regular wave. In the literature this is described as wave-structure interaction. First one can distinguish between fixed and moving bodies in general. As the wave energy devices are floating, a moving body description is of most relevance, but because of the simpler setup, experiments with fixed structures in the tank are considered first. In particular, experiments regarding the wave run-up on vertical and horizontal cylinders will be used as standard test cases to verify the model setup. The effects on the structure due to wave loading result in pressure differences and stresses, imparting forces on the body, whether fixed or floating. For the floating body the rigid body movement is constrained with different degrees of freedom. Whereas the cylinders of the Manchester Bobber, which will be described later in this chapter and Chapter 6.1, are constrained to move in the vertical direction only, the Pelamis device is able to move freely in all spatial directions and each section contrary to the next. Therefore rigid body movement with all degrees of freedom is of great interest.

The physical conditions driving the energy device are modelled numerically in this work using CFD. This is a fast growing field in offshore and coastal engineering, leaving the traditional approaches of physical experiments and much-specialised mathematical methods, to move to computational simulation techniques, which can be used for almost all physical problems to be solved numerically. Traditional offshore engineering problems are the flow around the hull of a vessel, the simulation of non-breaking wave structure interaction and simulations of tidal effects. For any of these examples one or more physical parameters are not important for the solution, hence are neglected to simplify the model. For example, for the flow around the hull, viscosity effects may not be important; for mathematical wave

simulations, as long as the waves do not break, they do not require the modelling of both air and water, additionally viscosity is often neglected. But when it comes to overturning waves interacting with structures producing green water on the structure, these physical parameters cannot be neglected. The two options are either to set up a physical experiment, which may be very expensive and provide small data sets, or a simulation using CFD.

Wave Energy Conversion and Devices The idea of converting the wave energy into usable forms of energy is more than 200 years old. The first British patent regarding a wave energy device is from 1855. The development of wave energy converters and national programs to exploit this energy in Europe until 2002 is described in Clement et al. (2002). The possible energy output for a nation depends on the length of the coastline and the exposure. Europe and especially Ireland and the UK are located in one of the most energetic sea areas of the world, the eastern Atlantic. Here the swell coming from the Atlantic plus the waves generated by the wind mainly blowing from the west provide a good basis for wave energy. In the European Union, Denmark, Ireland, Portugal, Sweden and the United Kingdom have considered wave power as energy resource since the 1970s. From that time onwards several million pounds have been invested in wave energy. One of the results is the European Marine Energy Centre Ltd., which is the world's first wave energy test ground near Orkney in the north of Scotland (EMEC 2010). The location was chosen, because of its reliable wave environment. The annual mean wave height of 3 m and maximum heights of 15m provide an excellent test environment.

Another test site for wave energy devices is under development. The Wave Hub project is based off the north coast of Cornwall in Great Britain and will provide a

grid connection point about 10 miles offshore. Here an area of 8 km^2 is reserved for the testing of up to four different energy devices (WaveHub 2010). Within this project the environmental impacts of wave farms are assessed. Millar et al. (2007) modelled the wave climate of that area and predicted some influence on the significant wave height and mean wave period in the lee of the Wave Hub. These are however so small that the impact on the shore and coastline can be neglected. The average loss in wave height over 11 month that were simulated is not more than 1 cm at a significant wave height between 2.7 and 3.4 m for the investigated area. The maximum value at a few locations is 4 cm.

Currently two wave power devices supply electricity into the grid in the UK. One is a 500-kilowatt shoreline oscillating water column device called Limpet at the Scottish island of Islay and the other is the Pelamis device producing 750 kilowatt, installed at the European Marine Energy Centre.

Limpet generates electricity from an oscillating water column (OWC). The change in water level inside a chamber induces airflow which drives a low pressure turbine. Limpet however is not an offshore floating device but built on solid ground at the shoreline. An example of a floating OWC is Sperboy (SPERBOY 2010). It employs the same principles of trapped air in a chamber that oscillates from the waterlevel changes inside it, but floats in the open sea.

Pelamis (PelamisWavePower 2010) is a floating device of the attenuator type. It works parallel to the wave direction, heading into the wave train. It consists of four slender semi-submerged cylinders, which are linked by hinged joints. The ocean waves move the adjacent sections relative to each other. From this motion the power take off modules sitting between them generate electricity. The design, simulation and testing of Pelamis and especially the power-take-off (PTO) is de-

scribed by Henderson (2006). During the development initial small scale tank tests were carried out, which were followed by a working 7^{th} scale model to investigate the hydrodynamics and non-linear dependencies in the motion of the sections. With this data the numerical simulation software was validated and extended. From the 7^{th} scale model a full scale test rig for the PTO was developed to estimate the losses in its hydraulic system. Other examples of this wave-profiling-type energy converter are described in Farley & Rainey (2006) with the "buckling raft" and a "bulge wave device". The latter device, also called Anaconda, comprises a water-filled rubber tube which floats just underneath the water surface. Due to the profile changes of the wave the diameter of the tube varies, depending on whether the section is in the trough or crest of a wave. Thus an internal fluid flow is obtained, which drives a PTO unit.

One of the first wave energy converters was Salter's Duck. This device is investigated theoretically and experimentally by Greenhow et al. (1982). In their work they address the survivability of the device in extreme waves using narrow channel tank tests and a non-linear potential flow code to calculate the hydrodynamics up to the point of wave breaking. Greenhow et al. identify the turbulent flow around the beak of the duck, the non-linearity in the buoyancy restoring force and the non-linear hydrodynamic effects resulting in 2nd harmonic wave generation to be the three main effects that influence the survivability for this particular device.

This device however never reached commercial status although it was very efficient. Recently the design has been altered towards a device for the desalination of sea water. This concept is investigated and described by Cruz & Salter (2006). Salter's Duck is a so-called point absorber. It can absorb energy from waves that may come from any direction. This makes it very efficient in a non-directional

wave climate, where for example attenuator-type WECs might be less efficient. The Manchester Bobber, based on a patent by Stansby & Jenkins (2006), is also a point absorber. It consists of an array of many cylinders. These float on the water and are allowed to move vertically. The displacement of the float coming from the waves drives a generator via a pulley system. The cylinders are attached to a superstructure which either floats stationary like an oil rig or sits on the sea bed. The Manchester Bobber and the Pelamis are investigated in this thesis.

Other concepts for WECs are the Wave Dragon, the Archimedes Wave Swing (AWS) and the Oyster. All of these follow another concept of harnessing the wave energy. The Wave Dragon is an overtopping device, which captures the waves in front of a ramp. Two reflectors on either side of the ramp focus the waves towards it, where they run up and feed a reservoir. The captured sea water runs through a number of low pressure hydro turbines, which produce electricity. Soerensen et al. (2003) describe the design process of this device beginning from a 50^{th} scale model to a working prototype. The Oyster, which is developed by Aquamarine Power, is a bottom mounted surge device. It does not pierce the free surface and is driven by the dynamics of the water particles underneath the wave crests and troughs. These drive the bottom hinged pendulum from where the electricity is taken from. The AWS also sits on the sea bed. This device is made of a fixed superstructure and a pressurised chamber that is allowed to move vertically due to the pressure differences underneath the wave profile.

Currently the Manchester Bobber is still under development. It has successfully passed the first and second design stages, which included 100^{th} scale physical tank tests of a full device, the successful scale-up to 10^{th} scale tank test of a single float and the development of computational models to predict average power output

and efficiency. In the third stage array tests were completed. These were done in 70^{th} scale and included the drive train and generator. A similar single float was tested beforehand in focused waves. These results are used for comparison with the numerical simulations described in this Chapter 6.1. The physical experiments including the array tests are presented in Stallard et al. (2008).

Description and Generation of Waves The two requirements that any of these WECs have to fulfil are: being able to convert wave energy efficiently in small to moderate seas and also being able to survive the harsh wave environment of a storm sea (Cruz 2008). To accomplish these requirements the device developers carry out a large test programme involving physical tank tests and numerical simulations.

At the very beginning of the design process however the description of the wave climate that might occur at the chosen location needs to be established; or vice versa: the sea state for which the device shall be optimised needs to be quantified. Deep water waves are random oscillations of the water surface. They are driven by shear stress that the wind generates, with swell coming from areas possibly far away from the location. These contributions make the sea state irregular and multi-directional. Wave buoys are used to record the surface elevation and wave direction over time. For most sites a main wave direction exists and the wave records may be taken as directional. Assuming that each wave is the superposition of several sinusoidal waves, the wave record can then be decomposed into a spectrum. This can be done by performing a Fast Fourier Transformation on the measured surface elevation time history. When no wave data is available, the sea climate may be described by using a design spectrum. Commonly used in offshore engineering are the JONSWAP and Pierson-Moscovitz (PM) spectra. They link the spectral den-

sity, i.e. the energy that is carried by the wave of a certain frequency, to the wind speed. Thus the wave climate can be reproduced by means of wind data. The JONSWAP spectrum is developed from the PM spectrum (Pierson & Moskowitz 1964), which describes a fully developed sea. Hasselman et al. (1973) carried out large field measurements in the North Sea to describe the wave environment. They discovered that the measured wave spectrum did not match with the earlier developed PM spectrum and modified the formula of the PM spectrum by adding empirical parameters to fit the measured results. The theory behind the spectral and statistical analysis is described in textbooks such as Rahman (1994) and Cruz (2008). The sea state can then be characterised by a number of spectral and statistical parameters. $H_{\max}$ and $T_{\max}$ are the largest wave height and period. Other important parameters are the significant waveheight $H_S = H_{1/3}$, which is the mean value of the largest 1/3 of waveheight from all waves in the record. Also interesting is $H_{1/10}$ which accordingly represents the mean wave height of the largest 10% of all waves from the measured time history.

For the economical operation of WECs smaller waves are important, which may be assumed linear or sinusoidal in their shape. The important information to ascertain are the predominant wave periods at the location where the device will be working. The developers then design their WECs to achieve optimum energy output for that range of wave periods. This makes it necessary to perform a comprehensive parametric study of different wave climates in short time to be commerially effective. In the case of Pelamis this is done by using linear simulation programs that operate either in the frequency domain, Pel_freq, or in the time domain, Pel_ltime (Retzler et al. 2003, Pizer et al. 2005). The calculations in the frequency-domain completely neglect the shape of the waves. The time-domain simulations assume

linear wave shapes and motion of the device and also do not take into account the effect of the WEC on the waves either. This saves simulation time, but cannot provide information regarding survivability or the non-linear behaviour of wave-body interaction and the interaction of the segments with one another.

For offshore survivability studies of fixed or floating structures, the wave profile is very important as this changes, with the wave height becoming larger and therefore more non-linear, or less sinusoidal. This can only be modelled in the time-domain. Describing wave dynamics accurately for all types of water waves has been subject of research for more than 300 years and an almost infinite number of publications exists. The origins of water wave theory are discussed by Craik (2004). The first attempt of describing the motion of linear, deep water waves in a theory was undertaken by Newton (1687). From then it took almost half a century until Bernoulli (1738) published his Hydrodynamica, where he gives a theory for shallow water waves. After that Laplace (1776), Euler (1757) and Lagrange (1781) worked on similar subjects, describing linear, shallow water waves. Airy (1841) published his results about non-linear, long, shallow water waves and tides in 1841 and Stokes (1847) describes non-linear waves in his work. Both theories assume irrotational flow for the waves, which means that the fluid particles move on closed orbits. Gerstner (1802) instead develops a higher order theory where he describes the flow underneath the waves to be rotational.

These early works are the basis for later studies and more accurate theories. Fenton (1985) develops a high order theory based on Stokes' work. Fenton's approach is valid for steady waves, shorter than 10 times the water depth, in constant water level and is as accurate as Stokes' fifth order theory. If higher or longer waves or higher accuracy is required the method developed by Rienecker & Fenton

(1981) may be used. The larger the waves get, the more non-linear they become, meaning that the waves differ from a perfect sinusoidal wave. The differences can be expressed in higher-order terms. Each of them describes a relatively small wave compared to the main wave. When reproducing waves in a wavetank, the waves will not reach the required height and shape, if these high-order parts are not included in the wave signal used at the wavemaker. This result is described by Schäffer (1996). He derives a second-order wavemaker signal from Stokes' theory and uses it in physical experiments, where he studies regular waves. Comparing with first-order accurate experiments he avoids the appearance of free waves, which result from the missing second-order component in the wave signal. Two types of free waves occur in the tank. They travel either with double the speed of the main wave or with half of its speed and interact with each other and the main wave in the wave tank. This results in spurious undulations on the measured surface elevation and also in changes of the wave shape. In his experiments Schäffer (1996) found that with the second order signal included, the generated wave shape stays stable when the waves travel along the tank and the free waves are eliminated.

Similar results can be seen in Taylor (1992). Rather than modelling regular waves this work concerns the numerical simulation of focused waves coming from NewWave theory. The aim of this approach is the superposition of several wave components taken from a wave spectrum, which interact with one another to produce one large extreme wave. To focus several waves of different periods and respectively wavelengths in a laboratory tank, the short waves have to be generated first. The long waves, which travel faster than the short waves, overtake them at a location in the tank and become superimposed. In contrast to traditional wave focusing as described by Rapp & Melville (1990), the NewWave theory is developed

statistically to reproduce the shape of a steep deep water wave. It is not intended for shallow water depths. In his study Taylor (1992) applies wave signals of first and second order accuracy and compares both results. The surface elevation of the focused wave generated by using the second-order accurate signal is significantly higher than that generated by the linear signal only. Walker et al. (2004) used the NewWave approach to reproduce the extreme wave event called the New Year wave at the Draupner oil platform numerically. The original NewWave theory with linear waves gives poor agreement of the wave height compared with the field data and the computational results. Hence it is extended using fifth order Stokes corrections terms, which improved the results significantly. The work carried out by Ning et al. (2007) and Zang et al. (2006) compares physical simulations of NewWave experiments with potential flow calculations. Ning et al. (2007) study the propagation of NewWave wave groups for 4 different heights and steepnesses. They are able to generate an almost breaking wave in a physical wave tank successfully and match this with a fully non-linear potential flow calculation. Zang et al. (2006) add a fixed ship shaped body which is passed by a NewWave wave group. Fully non-linear potential flow calculations are compared with experimental data, which give good agreement. In each case the test series do not involve breaking of the waves.

In an alternative approach to setting up an extreme wave as a design wave, which is also described by Tromans et al. (1991), one can simulate irregular sea states. This is done by Clauss et al. (2004), who combine a potential flow solver with a Reynolds-Averaged Navier-Stokes (RANS) solver. In their publication they describe the generation of extreme sea states, where they use a computationally less expensive potential flow solver until the point where wave breaking occurs. The result at this point is used as the initial condition for the RANS solver, which is more

expensive but able to simulate overturning waves. The wave generation in the physical experiments and in the numerical simulation is done by a wave paddle with the same setup. Johannessen & Swan (1997) published findings on non-linear transient water waves in which comparisons of the computational results are made with laboratory data. They apply the wave theory developed by Rienecker & Fenton (1981) to focus the waves in the tank. The study gives good agreement of the surface elevation and wave kinematics between the non-linear potential flow calculations and physical experiments.

The wave generation in a physical wavetank can either be done by a bottom hinged wave paddle or a piston wavemaker (Clauss et al. 2004, 2005). The fluid particle paths are circular in deepwater waves with decreasing radius with increasing water depth. For this reason deepwater waves are best reproduced by a bottom hinged wave paddle. When the water depth becomes shallower, the fluid particle orbitals of the waves get distorted and become shaped as ellipses. Hence, due to the translational motion, a piston wavemaker is better to generate the hydrodynamics of shallow water waves. Both of these methods can be applied to generate waves in CFD packages. Additionally, one can use a stationary boundary, define the surface elevation from any wave theory and flux the vertical and horizontal water velocity components into the domain. This method saves some computational effort, because the solver does not need to solve an extra equation to capture the motion of the wavemaker or boundary mesh.

Wave-Structure Interaction Traditionally the forces on offshore structures had been estimated by empirical formulae such as those given by Stokes (1851), Boussinesq (1885), Basset (1888) and Raleigh (1911). All of these equations describe the

force components of drag and the change in velocity of the surrounding fluid to estimate the total force on a cylinder (Keulegan & Carpenter 1958). Due to the exploitation of offshore oil reservoirs it became increasingly necessary to be able to estimate the forces on the structures used for that work, such as drilling platforms and oil rigs. Morison et al. (1950) systematically investigated the forces on cylindrical piles and published the equation

$$F = 1/2\,\rho C_D D u\,|u| + 1/4\,\frac{\rho\pi D^2 C_M du}{dt}. \tag{2.1}$$

Here, ρ is the fluid density, F is the horizontal force per unit length on the cylinder of diameter D and u is the horizontal component of the water particle velocity. C_D and C_M are the coefficients for drag and inertia respectively. Also having given a range of values for C_D and C_M, which are also described by Hogben et al. (1977) extensively, this became a convenient tool to estimate the forces per unit length for a pile. However, the equation was developed from tank tests driven with small sinusoidal waves. This makes it only applicable for certain cases as stated in Keulegan & Carpenter (1958). Their objective was to extend the Morison equation (2.1) by a supplementary function ΔR to represent the forces more accurately when C_D and C_M were considered to be constant throughout the whole wave cycle. Also they introduced a period parameter, which later became the Keulegan-Carpenter number N_{KC}, as

$$N_{KC} = \frac{AT}{D}, \tag{2.2}$$

with A being the amplitude of the oscillating fluid, T the period of the oscillation and D the diameter of the cylinder. N_{KC} describes the relationship between the

drag forces over the inertia force. For lower N_{KC} the inertia dominates the force contribution. Keulegan & Carpenter (1958) carried out physical tank tests with regular waves passing a fully submerged, horizontally mounted cylinder.

Numerical results for the horizontal cylinder cases described in Chapter 5.2 are compared with physical experiments published by Dixon et al. (1979). Their work was motivated by interesting results presented by the wave power research group from the University of Edinburgh. They carried out extensive measurements of wave forces on fully- and semi-submerged cylinders in two-dimensional regular waves. There, under certain circumstances, the vertical forces on the cylinder acted at twice the wave frequency and also were often negative for the entire wave cycle. This was believed to come from the interplay between buoyancy and inertial forces. Such a case was difficult to describe by any theoretical approach at that time. Hence Dixon et al. (1979) carried out similar experiments as presented by the wave power research group of the University of Edinburgh. The aim was to find an analytical expression, derived from the Morison formula (Morison et al. 1950), to describe the vertical forces due to waves on a horizontal cylinder and match it to the force measurements.

The Morison equation is only valid for body sizes that are small compared to the incident wave length, for which the flow is virtually uniform in the vicinity of the body (Sarpkaya & Issacson 1981). This regime is also referred to in the literature as the inertia regime. When the structure is large compared to the wavelength, which is generally considered at 1/5th of the wavelength, it alters the waves passing it. Rather than affecting only the wave shape close to the structure as done by a slender cylinder acting in the inertia regime, the large diameter cylinder diffracts and scatters the waves. For this case the Morison equation cannot be used and these

effects need to be taken into account. This so-called diffraction problem requires another approach, which was first addressed by MacCamy & Fuchs (1954). Generally the wave loads reduce when diffraction effects become important compared to the inertia regime. If a large cylinder was treated as a slender one the predicted forces would be overestimated. For structures in the diffraction regime, the effects of flow separation may also be neglected and the fluid assumed to be irrotational.

Calculation of Fluid Flow For the calculation of the hydrodynamics including the interaction with structures different theoretical approaches have been developed over the years. Furthermore, several mathematical tools, particularly numerical methods for their solution, exist. The most complete but also complex formulation of the motion of a fluid are the Navier-Stokes equations (NSE, see Chapter 3). The NSE are a set of partial differential equations, that here use a primitive variable formulation to describe the conservation of mass, momentum end energy. For a general flow property ϕ the NSE are given as

$$\frac{\partial \rho}{\partial t} + \operatorname{div}(\rho \boldsymbol{u}) = 0 \tag{2.3}$$

$$\frac{\partial (\rho \phi)}{\partial t} + \operatorname{div}(\rho \phi \boldsymbol{u}) = \operatorname{div}(\Gamma \operatorname{grad} \phi) + S_\phi. \tag{2.4}$$

ρ is the fluid density, t is time, $\boldsymbol{u}$ the velocity vector and Γ is the diffusion coefficient. The derivation of the NSE are described in Chapter 3.

For different flow regimes, the influence of one or more fluid properties may be considered to have a small effect and be neglected. Viscous effects for example are often not important in the far flow field or boundary layer effects might not

contribute to the overall force contribution and may be neglected. Taking this into account, the very complex and difficult to solve NSE can be simplified in some cases.

Considering the fluid to be inviscid, the NSE reduce to the Euler equations. Here the continuity equation is the same as in (2.3), but the momentum equations become

$$\frac{\partial(\rho u_i)}{\partial t} + \mathrm{div}(\rho u_i \boldsymbol{u}) = -\mathrm{div}(p\boldsymbol{i}_i) + S_\phi. \tag{2.5}$$

$\boldsymbol{i}_i$ is the Cartesian unit vector in the direction of the coordinate i, with i being x, y or z.

These are not necessarily easier to solve, but the boundary layer effects are neglected and no model needs to be applied to treat this region, which saves computational cost. Both, the NSE and the Euler equation can be used to calculate rotational and irrotational flow fields (Ferziger & Perić 2001).

When the flow field is considered irrotational and inviscid it can be expressed by a flow velocity potential. The continuity equation becomes the Laplace equation as

$$\mathrm{div}(\mathrm{grad}\Phi) = 0, \tag{2.6}$$

where Φ denotes the velocity potential.

The momentum equation reduces to the Bernoulli equation

$$p + \frac{1}{2}\rho \boldsymbol{u}_i^2 + \rho g z = \text{const.}, \tag{2.7}$$

which can be solved analytically once the velocity potential is known. Here, p

is pressure, g is gravity and z is the vertical Cartesian coordinate. From the velocity potential the corresponding stream function can be derived. The stream lines are orthogonal to the lines of constant potential. Both these equations form an orthogonal flow net.

If density variations are not large, density can be treated as constant in the unsteady and convective terms of the NSE but in the gravitational terms it is handled as a variable. This approximation is called the Boussinesq approximation.

Different mathematical methods can be applied to discretise and solve flow problems which are described by the theoretical techniques from above. From the empirical approaches described by Morison et al. (1950) and Keulegan & Carpenter (1958) panel techniques, also called Boundary Element Methods (BEM), have developed. BEM generally solve only the water fraction and assume the fluid to be irrotational and inviscid, hence employ potential flow theory.

Newman & Lee (2002) describe the use of BEM in offshore engineering and the way it is used to calculate wave loads and other hydrodynamic characteristics of the interaction of offshore structures with waves. Two recent developments are outlined which have evolved from low-order panel methods, where the surface of the structure is represented by a number of quadrilateral elements. For each of these elements the velocity potential is approximated by a constant value. A higher-order method uses B-splines to represent the submerged surface exactly. Also the velocity potential is calculated using B-splines. The advantage of this method is that the geometry is not restricted to being very simple. Also it is possible to combine several of these models to analyse multiple body interaction. Another enhancement of the low-order method is the pre-corrected Fast Fourier Transform method, which reduces the computational cost significantly. Although the higher-

order methods may converge more slowly for the tangential velocities near sharp corners or edges, which may result in problems for the pressure results, the usage of memory is reduced significantly and it is possible to compute large structures. Using this technique, Newman & Lee (2002) present results for a 1.5 km long semi-submerged Mobile Offshore Base, made up of five identical sections, for which the wave heights and drift forces were calculated.

In the attempt to reduce the computational time Bingham (2000) describes the use of a combination of established methods to simulate the wave induced motion of a restrained floating body in restricted water. Potential flow theory is used to calculate the waves near the floating structure and also for the wave-structure interaction. For the change of the wave shape when progressing from deep into shallow water a modified Boussinesq model is used. With this setup Bingham (2000) is able to predict the motion of the floating body in linear waves, even when the resulting motion is no longer linear. However, the model needs to be tuned via damping coefficients to be able to predict the response of the ship near resonance accurately.

A very similar approach to the one described by Bingham (2000) is used by v. d. Molen & Wenneker (2008). They also use a Boussinesq approximation to simulate the waves and feed the results in a Boundary Element code to compute the forces and translations from the waves on a container ship. The calculations for the waves and the forces are accurate to second order.

Another approach for the simulation of wave-body interaction by coupling different techniques is described by Wu & Eatock Taylor (2001). They assume the flow to be irrotational and incompressible so that a velocity potential exists. The flow is solved using a Finite Element (FE) discretisation for the far field simulations with a Boundary Element (BE) region close to the structure to simulate the interaction of

a fully submerged cylinder in waves.

Often more complex methods are mesh techniques such as the Finite Difference, Finite Element, Finite Volume or hybrid methods, which incorporate properties of either of them. The latest development are gridless Lagrangian methods, an example of which is Smoothed Particle Hydrodynamics (SPH) described by Monaghan (2005). Generally one can say that the computational effort increases from Potential Flow/BEM to CFD and gridless techniques. However, the output and the quality of the results, if applied correctly, also increases.

Computational Fluid Dynamics The field of computational fluid dynamics is extensive. Several methods can be used to describe the motion of a fluid. In any the fluid flow has to be described mathematically, using the conservation equations of mass and momentum. These equations are the basis for the Navier-Stokes equations, which become the Reynolds-Averaged-Navier-Stokes equations (RANSE) when turbulence is modelled as well. The three main discretisation methods used to calculate fluid flow, especially in engineering practice, are the Finite Element Method (FEM), Finite Difference Method (FDM) and Finite Volume Method (FVM). All three Methods are very well described by Ferziger & Perić (2001) and Versteeg & Malalasekera (2007).

In all approaches a grid is used to subdivide the domain. The FDM is the oldest of the three methods and mostly used for regular grids and simple geometries. The results are calculated at every grid point with an algebraic equation, which contains the variable value at the grid point and several of its neighbours as unknowns. If not given special attention, the FDM is not guaranteed to be fully conservative.

FVM and FEM are very similar to each other. Both can be applied to arbitrary

meshes and to complex geometries, which make them very interesting for engineering practice. The FVM uses a finite number of control volumes (CV) around the nodal points, which subdivide the domain. The physical values are calculated at these points. The FVM is conservative because of the nature of its construction, which makes it the first choice for CFD. Instead of using CVs, where the surface integrals represent the fluxes across these surfaces, the FEM uses weight or shape functions to interpolate the values between two nodal points. The FV and FE method can be combined in a control-volume-based finite element method (CV-FE), which uses the conservation approach of the FVM by applying control volumes around a central node but then calculates the fluxes through the CV by using element-wise shape functions. In STAR CCM+, a commercial CFD package produced by CD Adapco, the FV method is implemented, whereas in CFX, a commercial CFD solver by Ansys, the hybrid CV-FEM approach is used. The FVM approach used by STAR CCM+ and all appropriate models are described in the STAR CCM+ user manual (CD-Adapco 2009). The hybrid CV-FEM implementation used by Ansys in their software package CFX is described in the documentation (ANSYS 2006), both of which are also described in Chapter 3.

The Navier-Stokes equations describe mathematically the motion of a fluid. All velocity components appear in the momentum equations and also in the continuity equation; hence they are linked together. Additionally the pressure gradient appears in the momentum equation. Normally the pressure is not known beforehand and has to be solved during the calculation. For compressible flows the pressure can be calculated from the equation of state using the temperature and density correlation. In incompressible flows the density is constant, hence the pressure is not linked to the density and cannot be calculated in this way. An iterative solution strategy

developed by Patankar & Spalding (1972) called Semi-Implicit Method for Pressure Linked Equations (SIMPLE) can be used in this case (see Chapter 3.2). The strategy is to assume a pressure field, calculate the velocity field and update the pressure field until the continuity equation is satisfied. Starting from the SIMPLE method several prediction-correction algorithms have been developed. In Patankar (1980) the SIMPLER (SIMPLE Revised) algorithm is described. Van Doormaal & Raithby (1984) developed the SIMPLE-Consistent (SIMPLEC) algorithm and Issa (1986) describes the Pressure Implicit with Splitting of Operators (PISO) algorithm, which extends SIMPLE by adding another corrector step. Darwish & Moukalled (2006) implemented seven different pressure-correction schemes in a Navier-Stokes solver and studied their performance on incompressible multiphase flow phenomena. They found that SIMPLE and SIMPLEC are less computationally expensive than PISO for the same level of accuracy.

In Patankar (1980) an iterative solution procedure for a staggered variable arrangement using a FV discretisation is described. Patankar discusses different variable arrangements and extensions of the SIMPLE algorithm he proposed in Patankar & Spalding (1972). Baliga & Patankar (1983, 1980) describe a CV-FE method with a non-staggered grid, which is solved using a guess-and-correct procedure such as SIMPLE. Schneider & Raw (1987*a*,*b*) propose a fully coupled algorithm for the computation of fluid flow. They also use a CV-FE approach on a non-staggered grid, but fully couple the conservation equations to obtain a single set of linear equations, which can be solved iteratively using an algebraic multigrid procedure.

Simulations of wave energy devices in extreme waves require the calculation of the free surface. Here several approaches exist, which can be classified into two schemes: interface capturing and interface tracking schemes. The difference be-

tween these methods lies in the way the interface is reconstructed. One way to track the interface is by adding massless marker cells to the surface. These are attached to the surface and transported by the calculated velocities. Problems with this method occur when the simulation is three-dimensional, because of the increasing number of particles. Additionally it is necessary to either add or delete particles during the calculation depending on the surface curvature. As the particles have to stay in sequence, this means the particles have to be renumbered during the calculation. This gives two main disadvantages: the computational effort increases disproportionately to the level of surface curvature and dimension in the calculation and the use of merging or interrupting interfaces is very limited (Daly 1969). Another method of surface tracking is the use of a height function, as used by Hirt & Nichols (1981) to simulate waves in three dimensions. Here, points on the interface are related to points on a reference base, e.g. the sea bed in wave simulations, from where the distance or height is calculated. This method has the disadvantage that the reference point has to be single-valued, meaning only one point on the surface can be connected with one reference point. Hence it is not possible to simulate overturning waves with this method, although it is very efficient for simple flows. In single phase flows it is also possible to attach the surface to the mesh and reconstruct the whole mesh with every timestep. This technique reduces the storage requirement, e.g. of the marker particles, but is only valid for simple flows, as the surface cannot overturn. The limit of this method is the maximum deformation of the mesh and the ability to model one fluid only. However, the interface stays sharp This method is used in Demirdžić & Perić (1990), where it is applied to a co-located variable arrangement using a FV discretisation and the segregated iterative solution method SIMPLE. Zwart et al. (1998, 1999) use a cell-centred FV method for a single phase

simulation, where the grid is remeshed every timestep and follows the shape of the free surface. Zwart et al. call this approach Integrated Space Time (IST). They show the accuracy of their approach for a dam break simulation and for a breaking wave up to the breaking point. However, the reconnection of the surface as happens when waves break or fluid particles get separated cannot be computed.

The interface capturing methods can be divided into particle and volume fraction methods. Of the particle methods, the marker and cell (MAC) method by Harlow & Welch (1965) is the basis for several refinements and extensions. The idea is to add massless marker particles to one fluid, which are advected with it. If no marker particles are in a cell, this cell is by definition empty, or filled with another fluid. Cells containing marker particles and located beside "empty" cells, are considered to be partly filled, whereas cells surrounded by fully filled cells, are completely filled. This technique can deal with complex flow phenomena. However, the computational effort, especially for 3 dimensional simulations, is immense (Walker et al. 2004). When using a volume fraction approach an additional non-dimensional indicator function has to be solved for each cell reaching values from 0 to 1. If the cell is fully filled with the fluid, the value is one. If the cell is not filled, the value becomes 0. If the cell is partly filled, the volume fraction lies between 0 and 1. This leads to the problem with this approach, because the position of the interface separating the fluids is not known and has to be calculated. Different techniques may be used to find the interface and can be classified as line-techniques, which interpolate the position of the interface as a straight line parallel to one of the cell edges, donor-acceptor schemes and higher order differencing schemes. The donor-acceptor approach controls the advected amount of fluid over the cell faces to ensure that not more mass enters the cell than the volume of the cell can hold. One of the

first examples using the volume fraction equation is described by Hirt & Nichols (1981). They successfully apply the Volume of Fluid (VOF) technique to incompressible flow calculations with free surfaces, breaking wave problems and mixing of water and air in a container.

Commercial Software Packages For the calculations carried out in this project, two software packages CFX and STAR CCM+ are used. Both have the VOF method implemented to calculate the fluid volume fractions for each cell, which is described in Chapter 3.2.5. For the surface reconstruction both use different schemes. In CFX, the VOF method paired with the interface reconstruction scheme of Barth & Jesperson (1989) is used (ANSYS 2006). This scheme comes from the group of high order differencing schemes. The convection equation of the volume fraction is calculated depending on the maximum and minimum values of the surrounding cells. This treatment makes it independent of the chosen timestep. In STAR CCM+ the interface capturing scheme CICSAM (Compressive Interface Capturing Scheme for Arbitrary Meshes) developed by Ubbink (1997) is used (see CD-Adapco, 2009). Ubbink controls the flux over the cell faces depending on the Courant number. The Courant number links the flow velocity to the cell size and the timestep as described in Chapter 3.2.1. If the Courant number is high, either the cell is very small or the flow is fast.

The difference between high order differencing schemes and a high resolution scheme like CICSAM is described in Zwart et al. (2003) and Zwart (2005). They implement both schemes in their FV Navier-Stokes solver and compare the smearing of the interface for different test cases. The Barth & Jesperson (1989) scheme depends on the face values of the surrounding cells and not on the timestep or the

Courant number, so the timestep might be larger. Comparisons of other high resolution interface capturing schemes, CICSAM and surface reconstruction methods embedded in VOF are described by Darwish & Moukalled (2006), Muzaferija & Perić (1999) and Rhee et al. (2005).

Applications of CFD in wave-structure investigations Rainey & Chaplin (2003) investigate the hydrodynamics of a slender vertical cylinder with $kD = 0.29$, where k is the wavenumber and D the cylinder diameter, in steep waves with $kA = 0.42$. In physical experiments they observe small breaking waves on either side of the cylinder which propagate around the cylinder and collapse afterwards and generate a secondary loading. This phenomenon is associated with the ringing of offshore structures and described in Chapter 5.1.3. Rainey & Chaplin (2003) use linear theory to compute the water surface elevations around the cylinder; and also the forces on the cylinder are calculated. Although the model, which uses a particle-sheet definition, should resolve the breaking of the waves, it does not resolve the secondary loading cycle properly and also the wave run-up in front of the cylinder cannot be predicted.

The ringing phenomenon is also subject of investigation in Liu et al. (2001). They compare the approaches described by Faltinsen et al. (1995) and Malencia & Molin (1995) with their own mixed Eulerian-Lagrangian higher-order boundary element method and physical experiments. Differences are outlined between the frequency-domain perturbation methods proposed by Faltinsen et al. (1995) and Malencia & Molin (1995) and the physical experiments and the method used by Liu et al. (2001).

Huseby & Grue (2000) carried out physical tank tests for vertical cylinders of

two different radii in regular 1st order Stokes waves of four different parameters each. They generated waves up to a steepness of $kA = 0.24$ and discuss the higher-harmonic force contributions on the cylinder. These are also compared to the calculations described by Faltinsen et al. (1995) and Malencia & Molin (1995) and other measurements described by Stansberg (1997). Huseby & Grue (2000) conclude that the first-harmonic wave force is well predicted by the Morison equation or McCamy-Fuchs solution (MacCamy & Fuchs 1954) up to a wave steepness $kA = 0.2$. The third harmonic wave force, which is responsible for the ringing effect, is found to be in good agreement with the results by Malencia & Molin (1995) and Faltinsen et al. (1995) as long as the steepness is small. For steeper waves Huseby & Grue (2000) report significant differences between the measurements and the predictions by Faltinsen et al. (1995).

More experimental studies concerning the ringing of a vertical cylinder and comparisons with experiments from Chaplin et al. (1997) and also model tests of the Draugen and Heidrun platforms are presented by Grue & Huseby (2002). In their publication Grue & Huseby (2002) discuss different effects within the model tests, which might influence the quality of the results. These are scaling effects, if the secondary load cycle is produced by regular or focused waves and also the effect of tank width.

Luck & Benoit (2004) describe physical experiments in a wave tank for studying the wave load on fixed vertical cylinders. Additionally the measured forces on the structure and the water wave kinematics are compared with calculations. For these, Luck & Benoit (2004) use linear wave theory because of the simplicity. The results show clearly that this approach underestimates the measured values by up to a factor of 2.4. Retzler et al. (2000) study wave-structure interaction by moving a

vertical cylinder in still water and then stopping it suddenly. The wave field around the structure is photographed in the physical experiment and compared with linear potential flow calculations. The linear potential flow calculation overestimates the measured waves significantly with increasing Froude number, which may be because viscosity is not included in the potential flow model.

Inspired by the development of Salter's Duck, which is described by Greenhow et al. (1982), Newman & Lee (2002) developed a linearised theory for the prediction of the power absorption of an oscillating body in sinusoidal waves. For the Duck device an efficiency of 80 % was stated by Salter (1974). Newman & Lee prove theoretically that for a cylinder symmetric about the axis of oscillation, the maximum possible efficiency achievable is 50 %. This can be increased to up to 100 % for a body that is allowed to move in several degrees of freedom. However, the theory is limited to bodies with simple shapes and linear waves. For the calculation of asymmetric bodies like the Duck the added mass and damping coefficients need to be known beforehand.

Hong et al. (2004) investigate the drift force due to waves on a floating OWC using a panel technique, with the velocity potential being solved. Within the linear assumptions this model performs well and predicts the body motion in all degrees of freedom. Furthermore the added mass and damping coefficients are computed.

Agamloh et al. (2008) apply the software package COMET, a predecessor to STAR CCM+, to model the case of one and two cylindrical buoys, each with a single degree of freedom in regular waves. COMET is very similar to STAR CCM+ and solves the Navier-Stokes equations using a FV discretisation. The fluid interface is captured using a High Resolution Interface Capturing (HRIC) scheme similar to that described by Muzaferija & Perić (1999) and Ubbink (1997). The setup may

be used to simulate a point absorbing WEC. Therefore they also model the PTO represented by a linear damping coefficient. The results are not validated against physical tank tests but show possibilities for future applications regarding optimum spacing of arrays of point absorbers.

Widden et al. (2008) present an analytical method to model two different PTO arrangements for a WEC in pitch and surge in small sinusoidal waves. Assumptions are made for allowing small motions only and added mass coefficients.

CFD simulations for an OWC WEC using the commercial software package FLUENT are described by Paixão Conde & Gato (2008). Two steady state simulations are presented for the expected maximum and minimum air flow velocities inside the chambers. These simulations are used to identify flow conditions inside the chamber that might generate spray, which could be sucked in by the turbine and result in poor working performance of the device.

Kleefsman (2003) use a CFD package called COMFLOW to calculate the wave loading on a SPAR platform. The Navier-Stokes equations are solved using a FV approach. Both fluids, air and water, are simulated using a VOF formulation to take wave breaking and green water into account. They investigated the wave loading on the monopole of the platform, which was divided into three segments. For each section the horizontal forces were extracted. Furthermore the surface elevations were analysed. Kleefsman (2003) found a high mesh dependency for different wave scenarios. For longer waves the mesh could be relatively coarse compared to a short wave case, where the grid had to be much finer to resolve the wave correctly.

Also on the topic of water impacts on offshore structures Kleefsman (2005) published the results of her PhD thesis. Here green water effects are shown on fixed bodies and bodies able to move as a rigid body. The numerical studies are

calculated with a Navier-Stokes solver and the free surface is calculated using the Volume of Fluid approach. For the moving body a cut cell method is used. Waves are generated from linear theory and fifth order Stokes theory to get the loading of the structure in regular wave events.

Chapter 3

Mathematical Models

The simulation of extreme waves, breaking waves, wave loads on structures and body motion requires a mathematical model, which can handle all these scenarios. The Navier-Stokes equations (NSE) provide such a set based on the principles of mass conservation and force balance. In this chapter the governing equations of fluid flow for a Newtonian fluid will be derived and two solver approaches using different discretisation schemes and solution methodologies will be described.

3.1 Governing Equations

The governing equations describe the fluid flow mathematically making use of the conservation laws of physics. These state that the mass of a fluid has to be conserved, that the rate of change of momentum is equal to the sum of of forces that act on the fluid and, coming from the first law of thermodynamics, that the rate of energy change is equal to the rate of heat addition plus the rate of work done on the fluid particle.

For the derivation of the governing equations a fluid element of infinitesimal size is assumed. This is used to describe the behaviour of the fluid in properties such as velocity, pressure and density. The fluid itself is regarded as a continuum. This means that all fluid variables vary continuously through the fluid, which is valid for most fluids that have the mean free path of molecules much shorter than the characteristic length scale of the problem. Thus the fluid element is much larger

to be influenced by fluid molecules. Each side has the length of δi with i being x, y or z. Figure 3.1 shows the proposed fluid element with the centre at position (x, y, z).

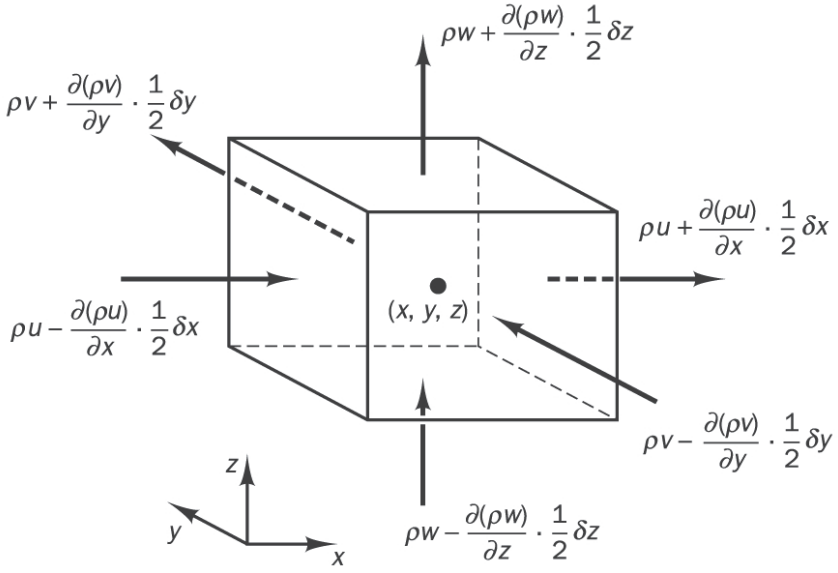


Figure 3.1: Fluid element

The first task is to describe the conservation of mass for the fluid element. The rate of increase of mass in the element is equal to the net rate of flow of mass into the fluid element. The change of mass over time is described by the product of density ρ and the volume of the fluid element over time, given by

$$\frac{\partial}{\partial t}\rho\,(\delta x \delta y \delta z). \tag{3.1}$$

The net rate of flow is given by the mass flow across the element faces. It can be described by the product of density, the face area and the flow velocity normal to the appropriate element face. For all six sides the net rate of flow sums up to

$$\left(\rho u - \frac{\partial(\rho u)}{\partial x}\frac{1}{2}\delta x\right)\delta y\delta z - \left(\rho u + \frac{\partial(\rho u)}{\partial x}\frac{1}{2}\delta x\right)\delta y\delta z$$
$$+\left(\rho v - \frac{\partial(\rho v)}{\partial y}\frac{1}{2}\delta y\right)\delta x\delta z - \left(\rho v + \frac{\partial(\rho v)}{\partial y}\frac{1}{2}\delta y\right)\delta x\delta z \tag{3.2}$$
$$+\left(\rho w - \frac{\partial(\rho w)}{\partial z}\frac{1}{2}\delta z\right)\delta x\delta y - \left(\rho w + \frac{\partial(\rho w)}{\partial z}\frac{1}{2}\delta z\right)\delta x\delta y.$$

Equations 3.1 and 3.1 lead to the continuity equation for compressible flow,

$$\frac{\partial\rho}{\partial t} + \frac{\partial\rho u}{\partial x} + \frac{\partial\rho v}{\partial y} + \frac{\partial\rho w}{\partial z}, \tag{3.3}$$

which reduces to

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \tag{3.4}$$

for incompressible flow, as the density is constant in this case. u, v and w are the Cartesian velocity components. To describe the fluid flow further the momentum equations need to be derived. The momentum equations are based on Newton's second law. That states that the rate of change of momentum of a fluid particle is equal to the sum of the forces acting on it. The forces can be divided into surface and body forces. Here, the pressure and viscous forces act on the surface and gravity, the centrifugal, Coriolis and electromagnetic forces are known as body forces. Only the surface forces will be included into the equations directly. The body forces will be included in the source term. According to the statement for the conservation of mass for a fluid element the equation for an arbitrary fluid property ϕ, which for the momentum equation would be replaced with u, v and w for the momentum in x, y

and z direction, is written as

$$\frac{\partial \phi \rho}{\partial t} + \frac{\partial \phi \rho u}{\partial x} + \frac{\partial \phi \rho v}{\partial y} + \frac{\partial \phi \rho w}{\partial z} \tag{3.5}$$

From the forces that act on the faces of the fluid element, i.e. those resulting from the pressure and stress components, the momentum equations can be derived for every coordinate direction separately. When multiplied by the face area they represent the force. For the whole fluid element as shown in Figure 3.1 the net force in one direction is the sum of all force components acting in that direction. For the x-direction the components are shown in the following equations. Therefore the force balance for forces in the x-direction is shown for each pair of opposing sides of the fluid element. The forces in the x-direction on the two element faces normal to the x-axis are

$$\begin{aligned} &\left[\left(p - \frac{\partial p}{\partial x}\frac{1}{2}\delta x\right) - \left(\tau_{xx} - \frac{\partial \tau_{xx}}{\partial x}\frac{1}{2}\partial x\right)\right]\delta y \delta z \\ &+\left[-\left(p + \frac{\partial p}{\partial x}\frac{1}{2}\delta x\right) + \left(\tau_{xx} + \frac{\partial \tau_{xx}}{\partial x}\frac{1}{2}\partial x\right)\right]\delta y \delta z \\ &= \left(-\frac{\partial p}{\partial x} + \frac{\partial \tau_{xx}}{\partial x}\right)\delta x \delta y \delta z. \end{aligned} \tag{3.6}$$

τ_{ij}, with i and j being the Cartesian coordinates x, y or z, are the viscous stress components acting in j-direction and normal to a surface in i-direction.

The x-forces on the faces normal to y-axis are

$$-\left(\tau_{yx} - \frac{\partial \tau_{yx}}{\partial y}\frac{1}{2}\delta y\right)\delta x \delta z + \left(\tau_{yx} + \frac{\partial \tau_{yx}}{\partial y}\frac{1}{2}\delta y\right)\delta x \delta z$$

$$= \frac{\partial \tau_{yx}}{\partial y} \delta x \delta y \delta z. \tag{3.7}$$

The x-forces on the faces normal to z-axis are

$$-\left(\tau_{zx} - \frac{\partial \tau_{zx}}{\partial z}\frac{1}{2}\delta z\right)\delta x \delta y + \left(\tau_{zx} + \frac{\partial \tau_{zx}}{\partial z}\frac{1}{2}\delta z\right)\delta x \delta y$$

$$= \frac{\partial \tau_{zx}}{\partial z} \delta x \delta y \delta z. \tag{3.8}$$

All components, i.e. equations (3.6), (3.7) and (3.8), summed up and divided by $\delta x \delta y \delta z$ give the total force per unit volume as

$$\frac{\partial (-p + \tau_{xx})}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z}. \tag{3.9}$$

To finalise the equation for the momentum conservation the body forces need to be added. These are included in the source term S_{Mx}. The same has to be done for the remaining coordinate directions y and z. Equations (3.10), (3.11) and (3.12) show the full set of momentum equations in non-conservative form.

$$\frac{\partial (-p + \tau_{xx})}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} + S_{Mx} \tag{3.10}$$

$$\frac{\partial \tau_{xy}}{\partial x} + \frac{\partial (-p + \tau_{yy})}{\partial y} + \frac{\partial \tau_{zy}}{\partial z} + S_{My} \tag{3.11}$$

$$\frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial (-p + \tau_{zz})}{\partial z} + S_{Mz} \tag{3.12}$$

For the calculation of 3-dimensional incompressible flow four equations have been described: the 3 momentum equations and the equation for the conservation of

mass. All are valid for Cartesian coordinates and developed from a fluid element with sides of finite lengths δx, δy and δz. The main unknown variables in these expressions are the three velocity components u, v and w and the pressure p. This set of four equations and four unknowns would be possible to solve. However, also included in the momentum terms are the nine expressions for the viscous stress components τ_{ij}. The definition for a Newtonian fluid says that the deformation of a fluid element is direct proportional to the viscous stresses that act on it. As all gases and most liquids, and certainly water, are isotropic, i.e. behave the same way in all coordinate directions, the viscous stresses can be linked to the dynamic viscosity of the fluid. The nine viscous stress components, of which six are independent, can then be expressed by

$$\tau_{xx} = 2\mu\frac{\partial u}{\partial x} - \frac{2}{3}\mu\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right),$$

$$\tau_{yy} = 2\mu\frac{\partial v}{\partial y} - \frac{2}{3}\mu\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right),$$

$$\tau_{zz} = 2\mu\frac{\partial w}{\partial z} - \frac{2}{3}\mu\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right),$$

$$\tau_{xy} = \tau_{yx} = \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right),$$

$$\tau_{xz} = \tau_{zx} = \mu\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right)$$

and

$$\tau_{yz} = \tau_{zy} = \mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right).$$

These being substituted into equations (3.10), (3.11) and (3.12) yield the Navier-Stokes equations for incompressible fluids

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}=0 \tag{3.13}$$

$$\frac{\partial u\rho}{\partial t}+\frac{\partial \rho u}{\partial x}+\frac{\partial \rho v}{\partial y}+\frac{\partial \rho w}{\partial z}=-\frac{\partial p}{\partial x}+\frac{\partial}{\partial x}\left(\mu\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right)+\frac{\partial}{\partial z}\left(\mu\frac{\partial u}{\partial z}\right)+S_{Mx}$$

$$\frac{\partial u\rho}{\partial t}+\mathrm{div}\left(\rho u\boldsymbol{u}\right)=-\frac{\partial p}{\partial x}+\mathrm{div}\left(\mu\,\mathrm{grad}\;u\right)+S_{Mx} \tag{3.14}$$

$$\frac{\partial v\rho}{\partial t}+\frac{\partial \rho u}{\partial x}+\frac{\partial \rho v}{\partial y}+\frac{\partial \rho w}{\partial z}=-\frac{\partial p}{\partial y}+\frac{\partial}{\partial x}\left(\mu\frac{\partial v}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial v}{\partial y}\right)+\frac{\partial}{\partial z}\left(\mu\frac{\partial v}{\partial z}\right)+S_{My}$$

$$\frac{\partial v\rho}{\partial t}+\mathrm{div}\left(\rho v\boldsymbol{u}\right)=-\frac{\partial p}{\partial y}+\mathrm{div}\left(\mu\,\mathrm{grad}\;v\right)+S_{My} \tag{3.15}$$

$$\frac{\partial w\rho}{\partial t}+\frac{\partial \rho u}{\partial x}+\frac{\partial \rho v}{\partial y}+\frac{\partial \rho w}{\partial z}=-\frac{\partial p}{\partial z}+\frac{\partial}{\partial x}\left(\mu\frac{\partial w}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial w}{\partial y}\right)+\frac{\partial}{\partial z}\left(\mu\frac{\partial w}{\partial z}\right)+S_{Mz}$$

$$\frac{\partial w\rho}{\partial t}+\mathrm{div}\left(\rho w\boldsymbol{u}\right)=-\frac{\partial p}{\partial z}+\mathrm{div}\left(\mu\,\mathrm{grad}\;w\right)+S_{Mz} \tag{3.16}$$

The Navier-Stokes equations ((3.13), (3.14),(3.15) and (3.16)) share a number of commonalities and can be re-written in the form of a general transport equation, as

$$\frac{\partial\left(\rho\phi\right)}{\partial t}+\mathrm{div}\left(\rho\phi\boldsymbol{u}\right)=\mathrm{div}\left(\Gamma\,\mathrm{grad}\;\phi\right)+S_{\phi}, \tag{3.17}$$

which states that the rate of increase for an arbitrary fluid property ϕ, such as a

velocity component, temperature or the concentration of a transported element, per fluid element (see Figure 3.1) plus the net rate of flow of the fluid element is equal to the rate of increase of ϕ due to diffusion plus the change of ϕ due to sources (or losses).

When equation (3.17) is integrated over a control volume (CV) it can be written as

$$\int_{CV} \frac{\partial(\rho\phi)}{\partial t} d\mathscr{V} + \int_{CV} \operatorname{div}(\rho\phi\boldsymbol{u})\, d\mathscr{V} = \int_{CV} \operatorname{div}(\Gamma \operatorname{grad}\phi)\, d\mathscr{V} + \int_{CV} S_\phi d\mathscr{V}. \quad (3.18)$$

Applying Gauss' divergence theorem[1] and integrating with respect to time for transient simulations results in the general integrated form of the transport equation:

$$\underbrace{\int_{\Delta t} \frac{\partial}{\partial t}\left(\int_{CV} \rho\phi d\mathscr{V}\right) dt}_{\text{transient term}} + \underbrace{\int_{\Delta t}\int_{\mathscr{A}} \boldsymbol{n}.(\rho\phi\boldsymbol{u})\, d\mathscr{A}\, dt}_{\text{convective term}} =$$

$$\underbrace{\int_{\Delta t}\int_{\mathscr{A}} \boldsymbol{n}.(\Gamma \operatorname{grad}\phi)\, d\mathscr{A}\, dt}_{\text{diffusive term}} + \underbrace{\int_{\Delta t}\int_{CV} S_\phi d\mathscr{V}\, dt}_{\text{source term}} \quad (3.19)$$

The first term on the left-hand side is called the transient term, followed by the convection term. On the right-hand side of the expression are the diffusion and source term.

This set of partial differential equations (PDE) defines the base for the computational methods used to calculate the unknown flow velocities u, v, w and the pressure p. For this work, two different solver techniques have been applied. Both use a mesh of finite elements, on which the calculation is performed . If the simu-

[1] Gauss' divergence theorem states for a vector a $\int_{CV} div(\boldsymbol{a})\, d\mathscr{V} = \int_{\mathscr{A}} \boldsymbol{n}.\boldsymbol{a} d\mathscr{A}$, with $\boldsymbol{n}$ being the vector normal to the surface $d\mathscr{A}$. For a fluid element or a CV respectively this is equal to the sum of the normal vectors times the appropriate face area building the surface of the control volume.

lation is 2-dimensional, it consists of quadliteral cells. For the 3-dimensional cases mostly hexahedral cells are used. Therefore the PDEs need to be discretised in time and space onto the mesh, i.e. the nodes, the element faces and volumes. The aim is to receive a set of linear equations with the same number of unknowns and equations, which can then be solved numerically.

3.2 Solution of the Governing Equations

The objective of this section is to outline the discretisation of the governing equations on a mesh of finite elements consisting of control volumes, element faces and nodes to obtain a linear set of equations containing the same number of variables and equations.

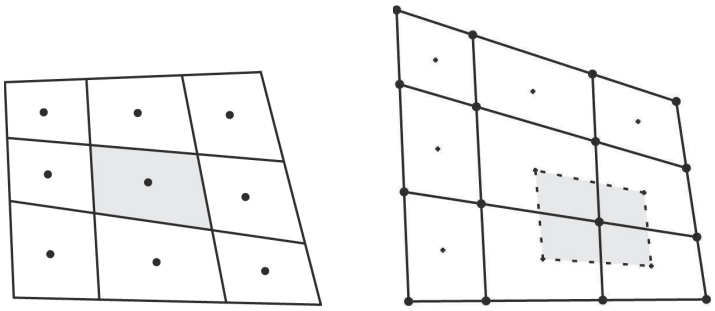
3.2.1 Discretisation of NSE

For unstructured meshes two ways exist to arrange the control volumes on the mesh. They can be constructed using either a cell-centred or vertex-centred approach. In a cell-centred arrangement the nodes are placed in the centre of the CV (Figure 3.2a), whereas in the vertex-centred method the nodes are placed on the vertices of the mesh. For this method the CVs are generated from the sub-volumes defined by the cell centroids and the mid points of the edges as it can be seen in Figure 3.2b. Both of these methods are used implemented in the solvers employed for this work. Due to a CV always having more vertices than centroids, the cell-centred method requires slightly less computational storage than the vertex-centred arrangement (Versteeg & Malalasekera 2007).

For the discretisation on any CV the transport equation (3.19) can be written as

$$\frac{\partial}{\partial t}\left(\int_{CV}\rho\phi d\mathscr{V}\right)+\sum_{\mathscr{A}_{CV}}\int_{\mathscr{A}_i}\boldsymbol{n}_i.\left(\rho\phi\boldsymbol{u}\right)d\mathscr{A}=\sum_{\mathscr{A}_{CV}}\int_{\mathscr{A}_i}\boldsymbol{n}_i.\left(\Gamma\text{grad }\phi\right)d\mathscr{A}+\int_{CV}S_\phi d\mathscr{V}. \tag{3.20}$$

$\boldsymbol{n}_i$ is the outward normal vector on the surface element, $\mathscr{A}_i$ of the CV as shown in Figure 3.3 for a 2D CV. Equation (3.20) sums up all fluxes of the fluid property ϕ



(a) cell-centred CV *(b)* vertex-centred CV

Figure 3.2: CV arrangement

through the CV faces of one CV and thereby connects it to a group of grid points that surround it. As only a few grid points participate in a given discrete equation, ϕ influences only the solution in the immediate neighbourhood. When the number of CVs in the domain becomes large, the discrete expressions are expected to converge to the exact solution for the whole domain. By assuring that the continuity and momentum equation are satisfied for every CV, conservation of mass and momentum for the whole domain is guaranteed. Assuming the face velocities of the CV are known the convective term of the discrete transport equation (3.20) can be written as

$$F_{CV} = \sum_{\mathscr{A}} \boldsymbol{n}_i \left(\rho \phi \boldsymbol{u}\right) d\mathscr{A} = \sum_{\mathscr{A}} F_i \phi_i \tag{3.21}$$

where ϕ_i is the transported fluid property at the centre of the CV surface element i and F_i is the convective flux parameter [kg/s]. Different methods are available to construct the convective flux at the CV face from the values of the surrounding nodes. Here, a second-order scheme is used. For the CV shown in Figure 3.4 the convective flux across the East face is calculated as

$$F_e\phi_e = \begin{cases} F_e\left(\phi_C + \Delta x_{Ce} \cdot \nabla\phi_C\right) & \text{for } F_e \geq 0 \\ F_e\left(\phi_E + \Delta x_{eE} \cdot \nabla\phi_E\right) & \text{for } F_e < 0. \end{cases} \tag{3.22}$$

$\nabla\phi_i$ are the reconstruction gradients in the cells either side of the CV face e. The computation of these will be described in Chapter 3.2.3. In Figure 3.4 N, S, E and W represent the North, South, East and West nodes of the centre node C. Small letters and subscripts stand for the CV faces and the flow in that direction. Δx is the distance in x-direction between te points indicated in the subscripts Ce and eE.

The same nomenclature is used for the vertex-centred CV arrangement. The centre node C is surrounded by the nodes N, S, E and W, which stands for North, South, East and West, as shown in Figure 3.5. The grey shaded area is the CV enclosed by a number of integration points ($\circ$) and constructed from the element face centres of the surrounding cells. For this case the convective flux is calculated as the sum of the fluxes at the integration points as

$$F_{CV} = F_C = \sum_{ip} \boldsymbol{n}_{ip}\left(\rho\phi\boldsymbol{u}_{ip}\right) = \sum_{ip} F_{ip}\phi_{ip} \tag{3.23}$$

where subscripts ip denote iteration point and n_{ip} is the area vector of the sub-face corresponding to ip as it can be seen from Figure 3.5. The convective flux needs to be approximated from the neighbouring vertex values and has to be bounded and accurate. Bounded means that for example the total flux across one cell from the surrounding cells must not be larger than the cell volume and must be limited if this occurs. This is achieved by blending between 1st and 2nd order differencing schemes. The implemented method is described by Barth & Jesperson (1989). The convective flux can be computed as

$$F_{ip} = F_{ip,up} + \beta \, \nabla F \cdot \boldsymbol{R} \tag{3.24}$$

with $F_{ip,up}$ being the upwind vertex value of the convective flux and $\boldsymbol{R}$ the vector from the upwind vertex to the integration point. ∇F is the best estimate of the solution gradient in the cell computed from the surrounding centroid data (see Chapter 3.2.3). The blending factor β needs to be calculated. For β = 0 the method is a first-order upwind scheme, which is bounded but diffusive. For β = 1 the scheme is a second order upwind scheme, but unbounded (Zwart et al. 2003, Zwart 2005). To make it a bounded high resolution scheme β is calculated as close to 1 as possible and reduced when necessary to prevent over- and under-shoots of the solution. Therefore β is calculated depending on the neighbouring vertex values. The first step is the computation of the maximum and minimum values of the neighbouring cells and the centre cell itself, i.e. β_C and β_{nb}. Then β_i is calculated for each cell vertex to get $\overline{\beta}_i$ given by

$$\overline{\beta}_i = \begin{cases} \min\left(1, \frac{F_C^{\max} - F_C}{F_{ip,i} - F_C}\right), & \text{if} \quad F_{ip,i} - F_C > 0 \\ \min\left(1, \frac{F_C^{\min} - F_C}{F_{ip,i} - F_C}\right), & \text{if} \quad F_{ip,i} - F_C < 0 \\ 1 & \text{if} \quad F_{ip,i} - F_C = 0 \end{cases} \tag{3.25}$$

β is then taken as the minimum of $\overline{\beta}_i$. For the centre cell C shown in Figure 3.5 this is given by $\beta = \min\left(\overline{\beta}_{nw}, \overline{\beta}_{ne}, ..., \overline{\beta}_{ws}, \overline{\beta}_{wn}\right)$ to find the value that insures the best level of accuracy and boundedness but also stability of the solution for this cell during the calculation of the convective flux.

The diffusive term of equation (3.20) is given as

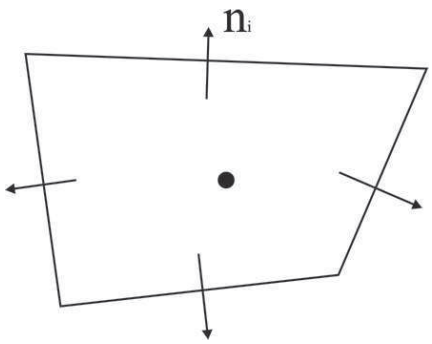


Figure 3.3: Surface element normal vector

$$D_{CV} = \sum_{\mathscr{A}_{CV}} \int_{\mathscr{A}_i} \boldsymbol{n}_i . (\Gamma \text{grad}\ \phi)\, d\mathscr{A}, \tag{3.26}$$

where $\boldsymbol{n}_i$ is the outward normal vector and $(\Gamma \text{grad}\ \phi)$ represents a diffusive flux vector for the CV element face $\mathscr{A}_i$. When the mesh is non-orthogonal corrections have to be applied, which are described by Mathur & Murthy (1997*a*,*b*) and Ferziger & Perić (2001). They introduce a cross-diffusion term, which is treated as a source term, when the discretised equations are solved. The derivation of this term is also described in Versteeg & Malalasekera (2007).

For the East face of the CV in Figure 3.4 the diffusive flux [kg/s] can be written as

$$\boldsymbol{n}_i . (\text{grad}\ \phi_e\ \Gamma_e) = \Gamma_e \left((\phi_E - \phi_C) \cdot \frac{\boldsymbol{n}_e}{\boldsymbol{n}_e \vec{CE}} \boldsymbol{n}_e + \left(\frac{\text{grad}\ \phi_E - \text{grad}\ \phi_C}{2} \right) \boldsymbol{n}_C \right) - \tag{3.27}$$

$$\Gamma_e \left(\frac{\text{grad}\ \phi_E - \text{grad}\ \phi_C}{2} \vec{CE} \frac{\boldsymbol{n}_e}{\boldsymbol{n}_e \vec{CE}} \boldsymbol{n}_e \right),$$

where Γ_e is the average of the cell values and (grad ϕ_i) are the gradients at the CV centroids. $\vec{CE}$ is the vector from C to E.

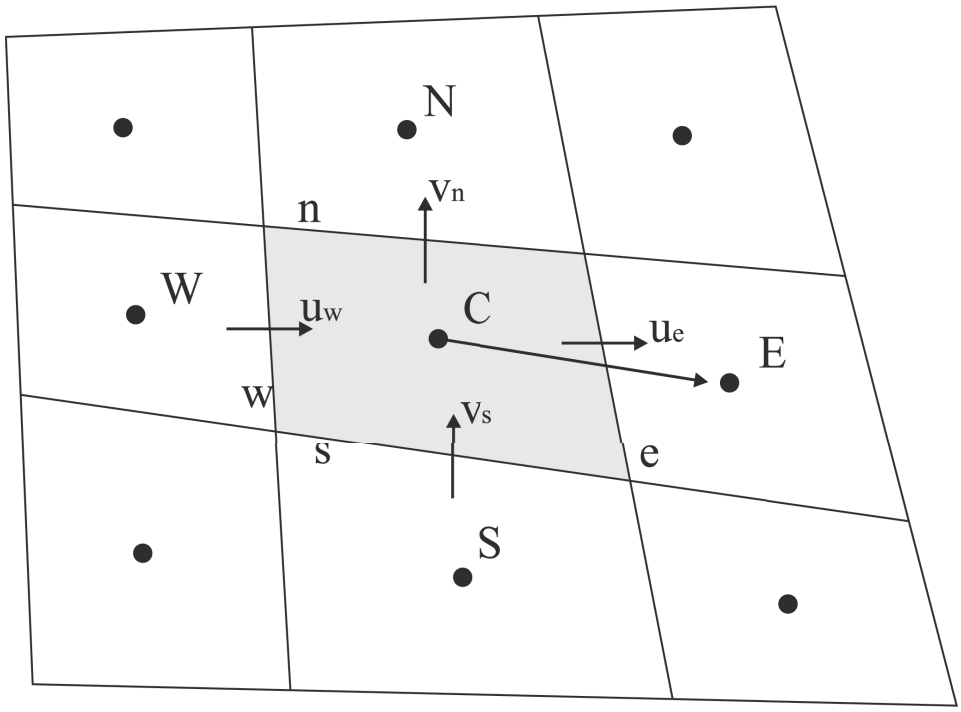


Figure 3.4: Cell-centred discretisation scheme
N,S,E,W are the North, South, East and West nodes of the centre node C, small letters and subscripts stand for the CV faces and flow in that direction

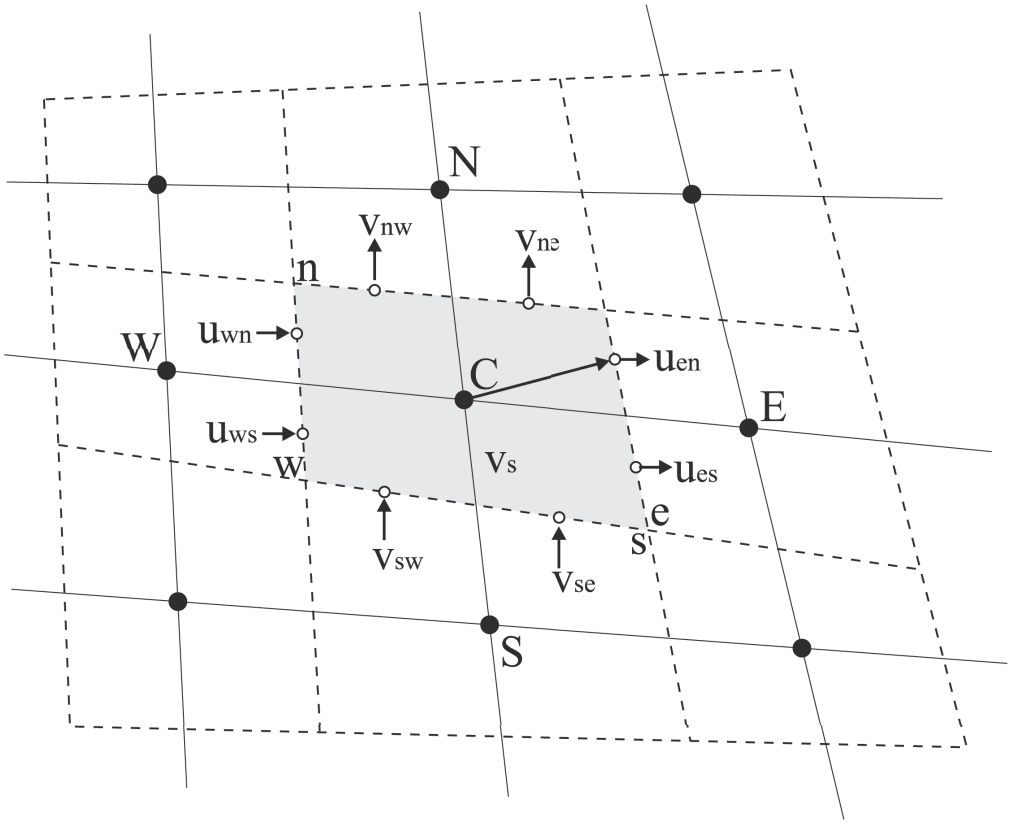


Figure 3.5: Vertex-centred discretisation
The solid lines represent the element boundaries, dashed lines divide the elements into sectors. The solution unknowns such as u, v, w and p are stored at the mesh nodes (•), surface fluxes are calculated at the integration points (○). A CV is constructed from the appropriate element sectors (grey shaded area).

3.2.2 Pressure-Velocity Coupling

For incompressible flow the pressure field cannot be calculated from equations of state that link the pressures to the density or temperature but have to be calculated from the relation to the velocities in the continuity and momentum equations. For the co-located variable arrangement, where the unknowns are solved and stored at the same locations, Patankar (1980) describes the so-called "checker-board" effect. This phenomenon occurs when the face values, which are used to calculate the node values, are interpolated linearly from values of the surrounding vertices only. When discretising these equations, the influence of the centre node cancels out for the calculation of the face fluxes and thereby the cell value is only dependent on the surrounding cells. For the upwind cell the same problem occurs, which might give a converged but non-physical solution of the pressure field. One remedy would be to store the variables, such as the velocities and pressures, at different locations. This is called a staggered variable arrangement. As this solution requires slightly more memory and is not implemented in the solvers used for the calculations in the following chapters it will not be discussed further. Patankar (1980) describes this method for an iterative solver.

To overcome this problem Rhie & Chow (1982) developed a higher-order pressure-velocity coupling method, which is widely used for co-located grid arrangements. For the cell-centred CV with centre node C as in Figure 3.4, the face velocities for the East face are interpolated as

$$u_e = \frac{u_P + u_E}{2} + \frac{1}{2}\left(\frac{\cancel{A}\Delta x}{a_C} + \frac{\cancel{A}\Delta x}{a_E}\right)\left(\frac{p_C - p_E}{\Delta x}\right) - \frac{1}{2}\left(\frac{\cancel{A}\Delta x}{a_C}(\nabla p_C) + \frac{\cancel{A}\Delta x}{a_E}(\nabla p_E)\right) \tag{3.28}$$

3.2.3 Gradient Computation

The computation of the gradients can be done differently. First, a method is described, which here is applied to the cell-centred FV approach. The approximation of the gradients, which are used to calculate the cell values for any fluid property ϕ involves three steps. First the unlimited reconstruction gradients are obtained. These are used to evaluate the cell gradients to reconstruct the cell face values for the computation of the diffusive and convective fluxes. To prevent the solution from over- and undershoots, the reconstruction gradients need to be limited (or bounded). Finally the cell gradients can be calculated from the limited reconstruction gradients.

The unlimited pressure reconstruction gradients $(\text{grad}\phi)_{unlimited}$ are computed using the weighted least squares method, which for the centre cell C in Figure 3.4 can be expressed as

$$(\text{grad}\phi)_{unlimited} = \left[\sum_f \frac{(x_{nb} - x_C) \otimes (x_{nb} - x_C)}{(x_{nb} - x_C) \cdot (x_{nb} - x_C)}\right]^{-1} \left[\sum_f \frac{(\phi_C - \phi_{nb})(x_{nb} - x_C)}{(x_{nb} - x_C) \cdot (x_{nb} - x_C)}\right] \tag{3.29}$$

with x_C and x_{nb} representing the centroids of cell C and the neighbour cell respectively. f stands for a control volume face and ϕ_C and ϕ_{nb} represent the data values for cell C and its neighbour, here the pressure. For all other variables the gradients

are approximated using Gauss' method as described by

$$(\text{grad}\phi)_{unlimited} = \frac{1}{\forall_C} \sum_f \phi_f \boldsymbol{a}_f \tag{3.30}$$

with ϕ_f being the arithmetic mean of the appropriate adjacent cell given as

$$\phi_f = \frac{\phi_C - \phi_{nb}}{2}. \tag{3.31}$$

The face value for any cell face centroid of the centre cell C can then be calculated using

$$\phi_{f,C} = \phi_C + \left(x_f - x_C\right) \cdot (\text{grad}\phi)_{unlimited} \tag{3.32}$$

When calculating the cell values from the unlimited reconstruction gradients, the cell values may exceed the face values. Thus the reconstruction gradients have to be limited to bound the face values and ensure physical behaviour of the solution. The limiting of the reconstruction gradients is described by Venkatakrishnan (1993). For the centre cell C the limited reconstruction gradient is given as

$$(\text{grad}\ \phi)_{limited} = \alpha\ (\text{grad}\ \phi)_{unlimited}, \tag{3.33}$$

where α is given by

$$\alpha = \min\left(\alpha_f\right) \quad \text{and} \quad \alpha_f = \frac{2r_f + 1}{r_f\left(2r_f + 1\right) + 1}. \tag{3.34}$$

The calculation of r_f depends on the maximum and minimum values of the neighbouring cells, which are used to limit the face values. r_f is given by

$$r_f = \begin{cases} \frac{\Delta_f}{\phi_C^{\max} - \phi_C} & \text{for} \quad \Delta_f > 0 \\ \frac{\Delta_f}{\phi_C^{\min} - \phi_C} & \text{for} \quad \Delta_f \leq 0, \end{cases} \tag{3.35}$$

where

$$\phi_C^{\max} = \max\left(\phi_C, \phi_{nb}\right), \tag{3.36}$$

$$\phi_C^{\min} = \min\left(\phi_C, \phi_{nb}\right) \tag{3.37}$$

and

$$\Delta_f = \phi_{f,C} - \phi_C = \left(x_f - x_C\right) \left(\text{grad } \phi\right)_{limited}. \tag{3.38}$$

These limited reconstruction gradients are then used for the calculation of cell gradients for all convected properties, such as u, v, w and p, which are needed in the discrete continuity and momentum equations and for the pressure-velocity coupling. Now the gradient for cell C of the fluid property ϕ can be calculated by

$$\text{grad}\phi = \frac{1}{\mathcal{V}_C} \sum_f \frac{\phi_{f,C} + \phi_{f,nb}}{2}, \tag{3.39}$$

where

$$\phi_{f,C} = \phi_C + \left(x_f - x_C\right) \left(\text{grad } \phi\right)_{C,limited} \tag{3.40}$$

and

$$\phi_{f,nb} = \phi_{nb} + \left(x_f - x_{nb}\right) \left(\text{grad } \phi\right)_{nb,limited}. \tag{3.41}$$

Rather than calculating the gradients using the procedure as described above, Schneider & Raw (1987*a*,*b*), Hutchinson & Raithby (1986), Baliga & Patankar (1983, 1980) adapt a procedure traditionally used in a Finite Element approach by using

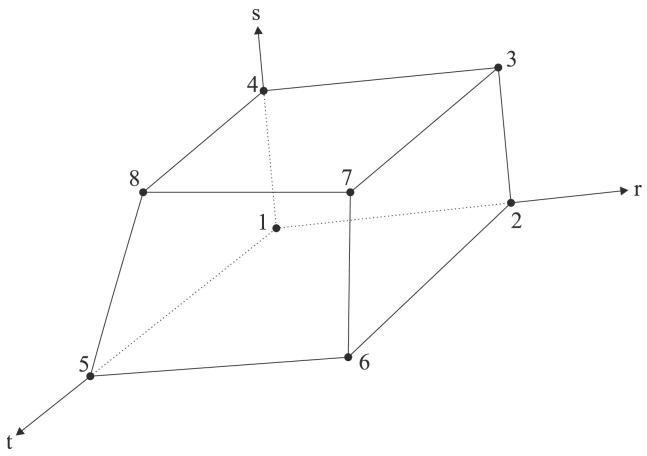


Figure 3.6: CV-FE shape function

shape functions to approximate the variation of a fluid property across the CV. This detail defines the name of a blended FE-FV method, which is also called a Control-Volume Finite Element (CV-FE) method. This method can be used for different solver strategies and grid arrangements. Baliga & Patankar use a segragative iterative solver such as SIMPLE with a staggered variable arrangement. Schneider & Raw, Hutchinson & Raithby describe a fully coupled solution procedure which is similar to the one outlined in Chapter 3.2.6 and co-located variables which are both used in CFX.

As the solution variables are saved in the mesh nodes, it is necessary to calculate gradients at the integration points. Here the CV-FE solver combines the FE with the FV method by making use of shape functions, which are typical for the FE approach. Across a mesh element a general flow variable ϕ varies as described by

$$\phi = \sum_{i=1}^{N_{Node}} N_i \phi_i, \tag{3.42}$$

with N being the shape function for node i and ϕ_i is the value of ϕ at node i. Fur-

thermore, the summation of the shape function over all element nodes gives unity and the shape functions are used to calculate geometric quantities, such as the coordinates of the integration points and surface area vectors $\boldsymbol{n}$. Figure 3.6 shows the geometry of the shape function for a hexahedral control volume and the local coordinate components r, s and t. N_i is given by

$$N_1(r,s,t) = (1-r)(1-s)(1-t), \tag{3.43}$$

$$N_2(r,s,t) = r(1-s)(1-t),$$

$$N_3(r,s,t) = rs(1-t),$$

$$N_4(r,s,t) = (1-r)s(1-t),$$

$$N_5(r,s,t) = (1-r)(1-s)t,$$

$$N_6(r,s,t) = r(1-s)t,$$

$$N_7(r,s,t) = rst$$

and

$$N_8(r,s,t) = (1-r)st.$$

As in a standard FE approach, the shape functions are used to evaluate the spatial derivatives of the diffusion terms and the pressure gradients.

3.2.4 Boundary Conditions

After assembling the discrete continuity and momentum equations for all interior faces conservation is ensured for the interior CVs. For the domain boundaries, however, the corresponding elements will have surfaces that are not yet included. For the cell-centred CV arrangement used for the FV solver the CV faces coincide with the boundary, whereas the vertex-centred CV is cut in half by the domain boundary. For all of these boundary elements the necessary data to solve the conservation equations needs to be specified by means of interior face and cell data and boundary values. As there are no further nodes outside the domain from which the data could be interpolated the values close the linear equations and not introduce further unknowns. By doing this, the fluid is confined and initialised for every timestep by the specifying face velocities and pressures at the CVs. For the simulations discussed in this work, a velocity inlet, an opening, wall boundaries and symmetry planes were used to define the problem.

At an inlet, the convective flux is known and specified in terms of the mass flow or flow velocities. At wall boundaries and symmetry planes the convective fluxes are zero and in the case of an outflow they are assumed to be independent of the coordinate normal to the boundary. Furthermore, at a no-slip wall the fluid velocities are equal to the wall velocity, which is zero in the case of a stationary wall. This is also called a Dirichlet boundary condition. Here the normal viscous stress is also zero. Both of these conditions need to be specified, as for the diffusive flux, or viscous flux, the derivative in the discretised flux equation would be non-zero. This is opposite to the symmetry plane, where the shear stress is zero but the normal stress has a non-zero value. For the co-located variable arrangement the

pressures need attention at the boundaries as all the solution data is stored in the same nodes. For the calculation of the pressure forces in the momentum equations the boundary values are needed. For small pressure gradients as they occur in flow parallel to a wall the values may be extrapolated from the interior nodes. For large pressure gradients from body forces, as they occur from buoyancy, this method is not accurate enough and large normal velocities in the vicinity of the boundary would be the result. By calculating only the normal velocities from the continuity equation or by refining the mesh locally this problem can be avoided (Ferziger & Perić 2001). This often results in prism layers near bodies in the flow.

For the vertex-centred CV the solution domain boundary coincides with the two subcontrol volumes of node C as it can be seen in Figure 3.7. The boundary is shown as the thick solid line across which the flow is advected into domain. At the appropriate boundary integration points $(bip, \square)$ the boundary flow of the property ϕ is included to the CV balance of node C. σ_n and σ_t are the normal and tangential stress components at the North and South boundary integration points *bipn* and *bips* respectively. Δx and Δy with subscripts n and s are the distances in vertcial and horizontal direction north- and southwards of C to the boundary of the CV.

For a scalar the convective and diffusive boundary conditions are obtained by interpolating from the surrounding node values and then integrating them over the CV as described for the interior faces. This procedure is straightforward (Schneider & Raw 1987*a*). The velocities and pressures, however, are treated as a set and not individually. Here, the boundary condition information is given as normal and tangential information. By doing this the boundary information can be defined by specifying

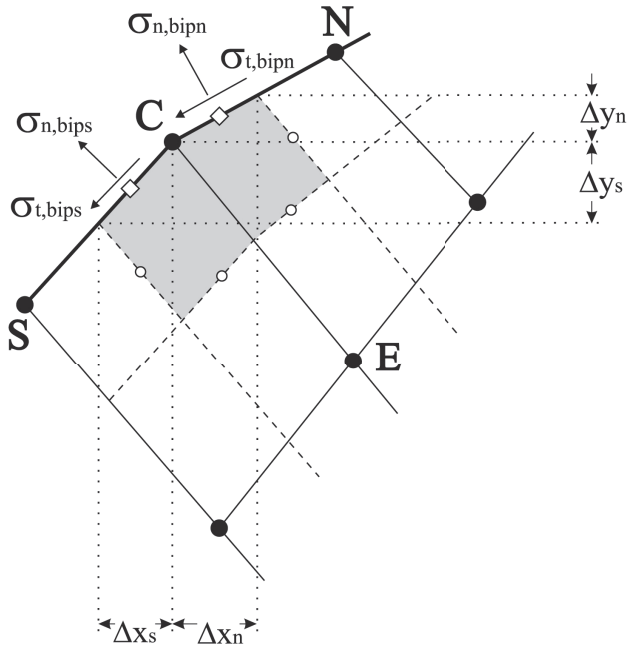


Figure 3.7: Boundary control volume for vertex-centred mesh arrangement
The boundary is represented by the thick solid line. For the grey shaded CV the boundary condition informations are specified at the boundary integration points (□), interior integration points are shown as (○)

- the velocities (Inlet)
- the pressure and velocity direction (Outlet)
- the zero normal stress with zero tangential velocity (Wall)
- zero tangential stress with zero normal velocity. (Symmetry)

At the boundaries the conservation equations for mass and momentum are assembled for all sides of the CV except for the boundary as a result of the discretisation procedure for all interior faces. In the case of the mass conservation the equations are closed by interpolation from the surrounding nodal values, as the mass flow is known. From the pressure-velocity coupling the pressures are obtained and

no more information regarding the pressures is needed. In case of the momentum equations, the convective, pressure and diffusive terms need special attention. The convective term of the momentum equation is assembled in a similar manner to the scalar and continuity equation, the pressures being interpolated from the nodal values (Raw 1985). The diffusive boundary momentum D_b including the pressure and velocity components (superscripts p, u and v) depends on both the normal and tangential viscous stress components, which in 2 dimensions are given as

$$D_b^{up} = \sigma_{n,bipn}\,\Delta y_n + \sigma_{n,bips}\,\Delta y_s - \sigma_{t,bipn}\,\Delta x_n - \sigma_{t,bips}\,\Delta x_s \tag{3.44}$$

$$D_b^{vp} = -\sigma_{n,bipn}\,\Delta y_n - \sigma_{n,bips}\,\Delta y_s + \sigma_{t,bipn}\,\Delta x_n + \sigma_{t,bips}\,\Delta x_s \tag{3.45}$$

These include the two unknown stress components in normal and tangential direction for each of the *bip*s. To resolve these, two boundary conditions are required, which are defined by

$$a\sigma_n + bu_n + cp = d \tag{3.46}$$

$$e\sigma_t + f\,(u_t + gu_n) = h \tag{3.47}$$

Here, the coefficients $a-h$ are user-supplied constants, which are used to define the boundary conditions as stated in the list above (Schneider & Raw 1987*a*).σ, u and p with subscripts n and t are the normal and tangential stress components, velocities and pressures. These are equivalent to velocity inlet, pressure outlet, opening and symmetry boundary conditions. When a stationary no-slip wall boundary is needed, the velocities and stresses are set to zero.

3.2.5 Interface Capturing

For the computation of multiphase flow and therefore the location of the fluid interface different methods can be used as discussed in Chapter 2. Here, two high resolution interface capturing (HRIC) schemes are described. Both rely on the Volume of Fluid (VOF) formulation, which originally was proposed by Hirt & Nichols (1981). Here an additional equation for a scalar c_i representing the volume fraction of fluid i is solved. As the form of this equation is similar to the transport equation it can be solved using the same methods as employed for the governing equations. For the computation the fluids are assumed to be immiscible. In a CV with several immiscible fluids present the pressure and velocity field is shared for all of them. Thus the fluid properties of a single fluid are calculated as from the fluid properties of its constituent phases and their volume fractions, such as

$$\rho = \sum_i \rho_i c_i \qquad \text{and} \qquad \mu = \sum_i \mu_i c_i, \tag{3.48}$$

with $c_i = \mathscr{V}_i/\mathscr{V}$. The transport equation for the volume fraction c_i is given as

$$\frac{\partial}{\partial t}\int_{\mathscr{V}} c_i d\mathscr{V} + \int_{\mathscr{A}} c_i \left(\boldsymbol{u} - \boldsymbol{u}_g\right) d\mathscr{A} = \int_{\mathscr{V}} S_{c_i} d\mathscr{V} \tag{3.49}$$

The capturing of the interface separating the fluids, here air and water, is done by a high-resolution compressive differencing scheme described by Ubbink (1997) and Ferziger & Perić (2001). It is also known under the acronym CICSAM, which stands for Compressive Interface Capturing Scheme for Arbitrary Meshes. It follows a donor-acceptor approach, where the boundedness is achieved using a Normalised Variable formulation as introduced by Leonard (1991). The Normalised

Variable Diagram (NVD) shown in Figure 3.8 forms the basis of the interface capturing scheme. It plots the normalised face value of the volume fraction c_f as a function of the normalised donor cell value c_C, where subscript C stands for the centre cell as used in Figure 3.9. Also in the digram are the boundedness criteria for central (CD), upwind (UD), 2nd order upwind (LUD) and downwind (DD) differencing. To achieve boundedness of the face value c_f, the influencing values, such as c_C and c_D are normalised to 0 and 1 and the face value has to lie within these bounds. The normalised face value of the centre cell C (Figure 3.9) is defined as

$$\widetilde{c}_f = \frac{c_C - c_U}{c_D - c_U}. \tag{3.50}$$

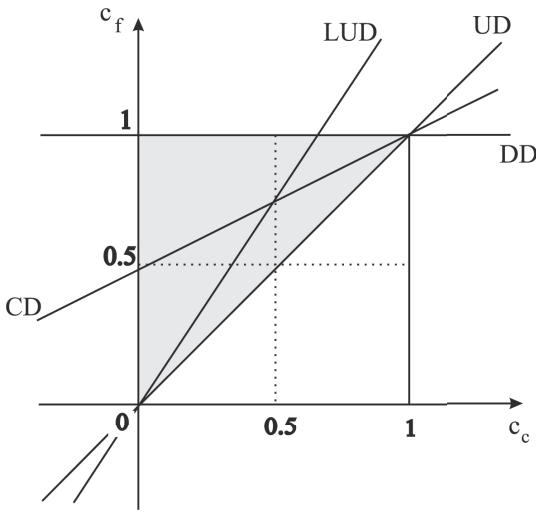


Figure 3.8: Normalised Variable Diagramm

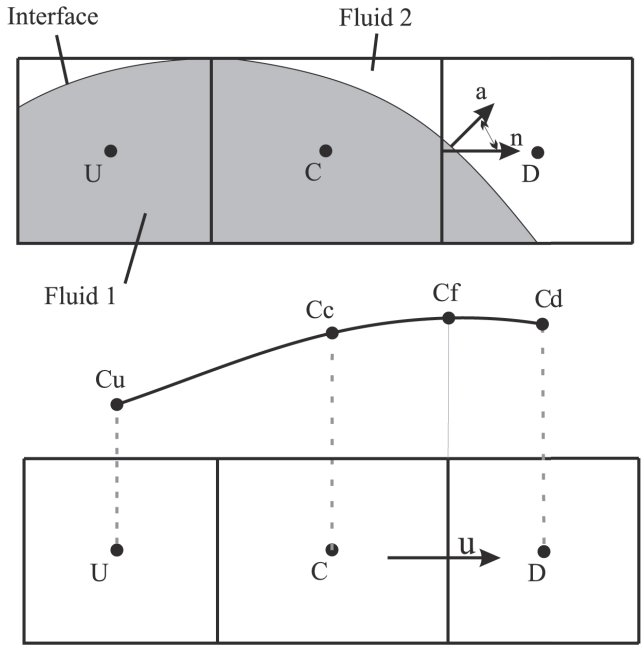


Figure 3.9: CICSAM scheme

The normalised face value $\widetilde{c}_f$ is calculated as

$$\widetilde{c}_f = \begin{cases} \widetilde{c}_C & \text{if} & & & \widetilde{c}_C & < 0 \\ 2\widetilde{c}_C & \text{if} & 0 & \leq & \widetilde{c}_C & < 0.5 \\ 1 & \text{if} & 0.5 & \leq & \widetilde{c}_C & < 1 \\ \widetilde{c}_C & \text{if} & 1 & \leq & \widetilde{c}_C & \end{cases} \tag{3.51}$$

To ensure that only such an amount of fluid leaves the donor cell as the acceptor cell can accommodate or vice versa the calculated value of $\widetilde{c}_f$ is corrected by using the local Courant number Co. The new face value $\widetilde{c}_f^*$ is calculated as

$$\widetilde{c}_f^* = \begin{cases} \widetilde{c}_f & \text{if} & & & Co & < Co_l \\ \widetilde{c}_C + \left(c_f - c_C\right) \frac{Co_u - Co}{Co_u - Co_l} & \text{if} & Co_l & \leq & Co & < Co_u, \\ \widetilde{c}_C & \text{if} & Co_u & \leq & Co & \end{cases} \tag{3.52}$$

with

$$Co = \frac{\boldsymbol{u}\ \boldsymbol{a}}{\forall_C}. \tag{3.53}$$

When the fluid interface as seen in Figure 3.9 is parallel to the flow direction the downwind scheme tends to wrinkle. Therefore the previous results are corrected once more by taking the orientation of the fluid interface into account. Here, θ, which is the angle between the normal of the fluid interface $\boldsymbol{a}$ and the cell-surface vector $\boldsymbol{n}$, is used as defined by

$$\widetilde{c}_f^{**} = \widetilde{c}_f^{*}\sqrt{\cos\theta} + \widetilde{c}_C\left(1 - \sqrt{\cos\theta}\right). \tag{3.54}$$

Finally the cell-face value is calculated as

$$c_f = \widetilde{c}_f^{**}\left(c_D - c_U\right) + c_U. \tag{3.55}$$

Subscripts U, C and D denote the upstream, cell-centre and downstream cell values. Co_l and Co_u are the lower and upper limits of the Courant number. For values $Co < Co_l$ HRIC is used, for $Co_l < Co < Co_u$ HRIC and UD are blended and for $Co_u < Co$ UD is used.

The compressive characteristics of controlled downwinding schemes such as CICSAM or other donor-acceptor methods (Ubbink 1997, Hirt & Nichols 1981) depends upon the timestep. Therefore they require a small timestep to capture the fluid interface sharply. Zwart (2005) and Zwart et al. (2003) describe another interface capturing approach, where the compressiveness is independent of the timestep length. The approach is similar to the advection scheme published by Barth & Jesperson (1989) and described in Chapter 3.2.1. However, when the method is applied

to the volume fraction equation it causes too much numerical diffusion. The solution to this problem was found by allowing $\beta > 1$ (see equation (3.24)), but still maintain the boundedness by reducing it as much as necessary to avoid under- and overshoots. For the treatment of the volume fraction equation β is allowed to go up to 2. As discussed earlier, the computation of the cell value is based on vertex values of the surrounding cells. Hence it is not timestep dependent.

3.2.6 Solution Methods

Segregated Solver

The solution of the flow equations can be done in a segregated-iterative manner. Here it is described for a co-located (non-staggered) variable arrangement combined with the Rhie-Chow type pressure-velocity coupling as outlined in Chapter 3.2.2 (Rhie & Chow 1982) and a SIMPLE algorithm (Semi-Implicit Method for Pressure-Linked Equations, Patankar (1980)).

The solution algorithm can be summarised starting from setting the boundary conditions. Then the reconstruction gradients are computed as described in Chapter 3.2.3 for the velocities and pressures in the flow field. From the reconstruction gradients the velocity and pressure gradients are obtained and the discretised momentum equation are solved to create an intermediate velocity field $u^\star$. This velocity field is used to compute the uncorrected mass fluxes or convective fluxes at the CV faces, $F^\star_{\mathscr{A}_i}$. After that the pressure correction equations are solved to produce cell values of the pressure correction p', which is the used to update the pressure field and the boundary pressure corrections. With the corrected pressure field the mass fluxes at the CV faces can be corrected as well as the cell velocities. Then the densities are updated due the pressure changes and the storage is freed.

Coupled Solver

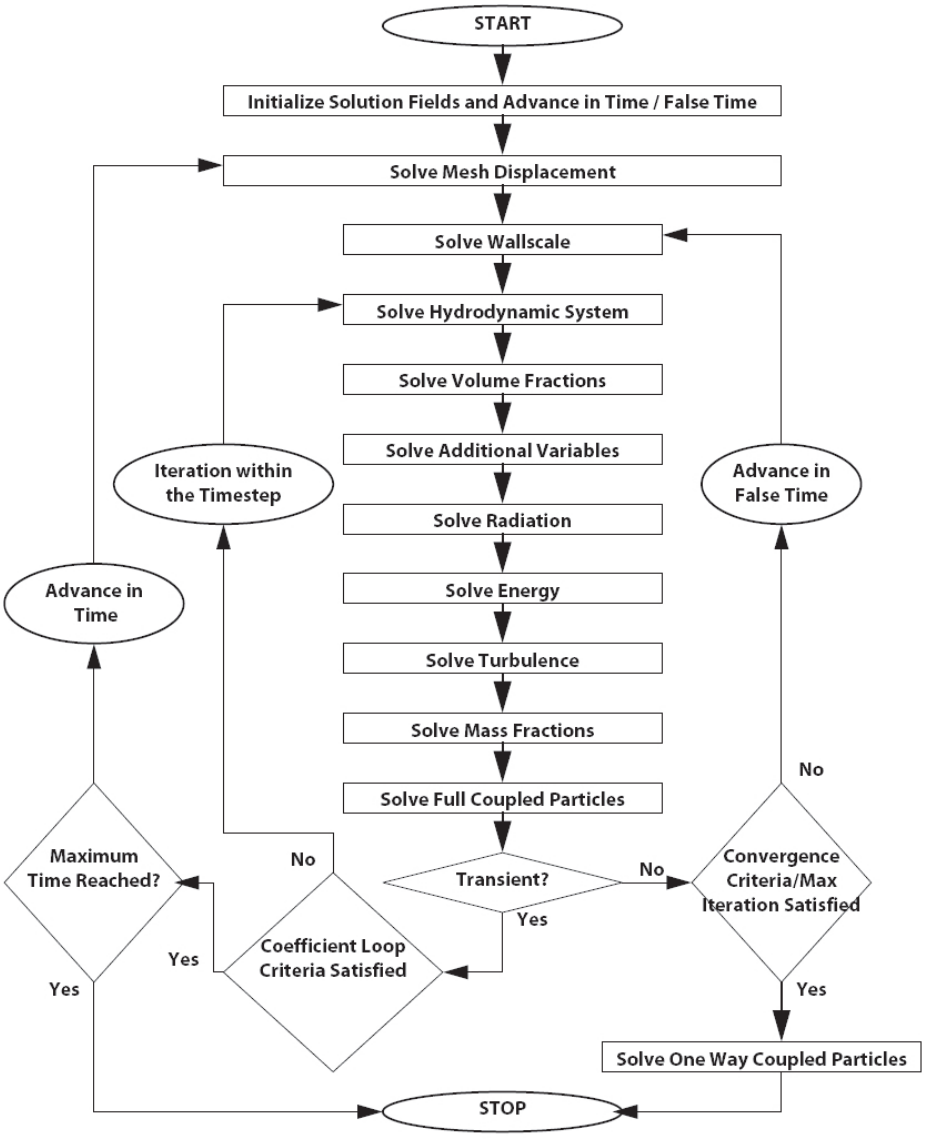


Figure 3.10: Solution strategy for the coupled solver (ANSYS 2006)

Rather than using a "guess and correct" approach as described for the segregated solver, which has the SIMPLE algorithm implemented, it is also possible to fully couple the discrete hydrodynamic equations for u, v, w and p and solve them as a

single system. When applying the conservation equations on the CVs of the mesh one receives a set of linear equations which can be written as

$$\sum_{nb_i} a_i^{nb} \phi_i^{nb} = b_i, \tag{3.56}$$

where ϕ is the solution, b the right hand side and a the coefficient of the equation. i denotes the CV or node in question and nb stands for "neighbour" but also includes the central coefficient multiplying the solution at the ith location as described by Schneider & Raw (1987*a*) and Raw (1996). For scalar equations such as the volume fraction the above variables are single valued; otherwise for the coupled mass and momentum equations, they are a 4x4 matrix and 4x1 vector given as

$$a_i^{nb} = \begin{bmatrix} a_{uu} & a_{uv} & a_{uw} & a_{up} \\ a_{vu} & a_{vv} & a_{vw} & a_{vp} \\ a_{wu} & a_{wv} & a_{ww} & a_{wp} \\ a_{pu} & a_{pv} & a_{pw} & a_{pp} \end{bmatrix}_i^{nb}, \tag{3.57}$$

$$\phi_i^{nb} = \begin{bmatrix} u \\ v \\ w \\ p \end{bmatrix}_i^{nb} \tag{3.58}$$

and

$$b_i = \begin{bmatrix} b_u \\ b_v \\ b_w \\ b_p \end{bmatrix}_i^{nb} . \tag{3.59}$$

The overall solution strategy for the CV-FE solver implemented in CFX is schematically shown in Figure 3.10. The definition file of the simulation contains all relevant data to start the simulation. The mesh values, such as the coordinates and numbers of the nodes, the connectivity and 2D and 3D regions, such as the boundaries and bodies, are defined here. At each of these nodes the CV, integration points are calculated and the velocities and pressures are initialized for the first timestep. If mesh motion is included, this is solved afterwards, followed by the hydrodynamics including the volume fractions of the fluids, additional variables, turbulence and mass fractions. Most likely the convergence criteria is not satisfied after the first iterations, meaning that the solution of the system of linear equations still differs too much from the maximum specified error allowed. If so, the hydrodynamics are iterated again within the timestep to improve the quality of the solution. For a well defined simulation using the CV-FE solver it usually takes 2-4 inner iterations before the solver advances in time.

Also included in Figure 3.10 are routines, which are not used in the process of this work, such as the modules that solve radiation, energy, mass fraction and full coupled particles. For the modelling of radiation different methods can be chosen in CFX. These depend on the type of radiation. If the medium is transparent to radiation at wavelengths in which the majority of heat transfer occurs, i.e. only the boundaries of the domain are heated or cooled by the radiation, the Monte Carlo

model should be used. The more optically dense the medium becomes, the Rosseland Model (Siegel & Howell 1981) or P1 Model (Raithby 1991) can be chosen instead.

For multiphase flow, such as modelled here using air and water, the volume fractions are solved after the hydrodynamic system. In the module that solves the mass fractions additional components such as smoke or pollutant can be solved, which do not influence the flow field of the fluid in which they are transported. This is valid for small to medium concentrations of such extra properties. When the concentration becomes larger one can also use particle tracking, which is solved at the end of the inner loop.

When turbulent flow is modelled, CFX solves the equations to close the Reynolds-Averaged Navier-stokes equations after the solution of energy equations. Here, the viscous stresses as described in Chapter 3.1 are calculated using for example:

- $k-\varepsilon$ or $k-\omega$ Model
- Large Eddy Simulation (LES)
- Detached Eddy Simulation (DES)
- Shear Stress Transport (SST) Model
- Baseline (BSL) Model

However, when the SST or BSL turbulence model are selected, the wall distance needs to be specified, which allows the solver to blend between the $k-\varepsilon$ and $k-\omega$ model. The wall scale region depending on these setting is solved at the beginning of the time loop. (ANSYS 2006)

Finally, when the stopping criteria is reached, the solution is terminated and the final results file is generated. Therefore the maximum number of timesteps or the total runtime can be specified. Transient result files can be saved after each timestep or at specified intervals. These contain all solution variables, such as the velocities for each fluid fraction u, v, w and pressures p for every cell of the domain.

Depending on the size of the domain, i.e. the number of cells and nodes, the disk space required to save one full transient simulation may be large and easily exceed several hundred gigabytes.

Chapter 4

Wave-Wave Interaction

4.1 Regular Waves

Prior to the focused wave cases, which are described in Chapter 4.2, regular waves were simulated in order to assess the suitability of the CFD methods for simulating extreme waves. The advantage of regular waves over irregular and focused waves is the simplicity in the setup. Regular waves should remain constant in shape and height throughout the tank, whereas focused and irregular waves interact with one another. This would make it difficult to judge, whether the results are correct or whether mesh effects or general problems such as numerical difficulties have occurred. For these reasons regular waves are used to investigate the performance of the numerical wave tank (NWT).

4.1.1 Computational Domain for Regular Waves

The regular wave tests are carried out in a 3-dimensional NWT. According to the description of the numerical methods in Chapter 3.2, the domain height is twice the initial water depth as both air and water are simulated. The height and width of the wavetank are both 3 m and the length is 35 m for the grid convergence study, and 100 m for the investigations of the damping in the NWT and the influence of the cell on the solution of the free surface.

The boundary arrangement is shown in Figure 4.1. A velocity inlet is used to

generate the waves at the left hand end. The side walls are defined as symmetry planes, the downstream end and the bottom surface are free-slip walls and at the top of the domain a pressure outlet condition is applied. At the pressure outlet only air is allowed to leave or enter the domain.

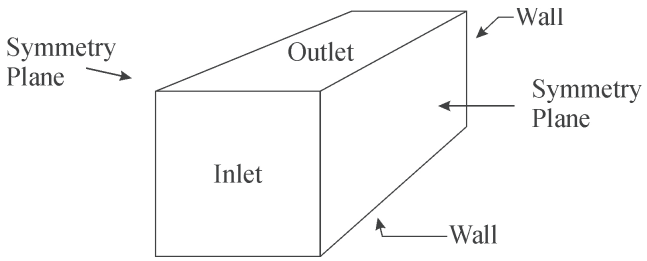


Figure 4.1: Regular wave domain

4.1.2 Generation of Regular Waves

The surface elevation η of the regular waves is defined from first order wave theory by

$$\eta = A\cos\left(kx - \omega t\right), \tag{4.1}$$

where A is the wave amplitude, k the wavenumber and x the Cartesian coordinate of the wavemaker (which is zero for all simulations, as the inlet sits at $x = 0$). ω is the angular frequency of the wave and t is time. η defines the vertical position of the water volume fraction. Below the level of the water surface the volume fraction of water is 1 and above it is 0. For the water fraction only, the vertical and horizontal wave velocity components from first order wave theory are fluxed across the inlet. These are given by

$$u = \frac{gAk\cosh\left(k\left(z+h\right)\right)\cos\left(kx-\omega t\right)}{\omega\cosh\left(kh\right)} \tag{4.2}$$

and

$$w = -\frac{gAk\sinh\left(k\left(z+h\right)\right)\sin\left(kx-\omega t\right)}{\omega\cosh\left(kh\right)}, \tag{4.3}$$

where u is the horizontal and w the vertical velocity component. z is the Cartesian coordinate in the positive upward vertical direction and h is the still water depth. The implementation of the inlet boundary conditions using different scripting approaches for the CFD codes is described in Chapter 4.2.3.

4.1.3 Grid Convergence Study

Figures 4.2 and 4.3 show short sections across the total domain height of the used meshes for the grid convergence study for both solvers. For the segregated iterative solver, which employs the FV approach and a cell-centred CV arrangement, five meshes are used. Initially the simulations were done on a pure tetrahedral mesh, then a pure polyhedral mesh and finally 3 hexahedral mesh arrangements were used. The properties of the meshes are described in Table 4.1. The tetrahedral and polyhedral mesh are refined in the region of the free surface, i.e. 0.2 m above and below the still water level. These showed a distorted solution of the captured fluid interface due to the shape of the mesh elements such that the face geometry could be seen on it. This finding led to the hexahedral meshes, where two degrees of refinement in the same region are tested. The coarse mesh (Figure 4.2c) has 8 cells per wave height with 5 cells above and below to resolve the remaining air and water fraction. For the fine mesh (Figure 4.2d), the number of cells per wave height is increased to

14. The last mesh, which is not shown in the picture, investigates the effect on the fluid interface solution due to refinement. Therefore the coarse mesh is used, where only a section in the tank is refined to the level of the fine mesh. The results for the hexahedral meshes for a regular wave with $A = 0.15$ m, $\lambda = 15.19$ m and $T = 4.2$ s at a waterdepth $h = 1.5$ m are shown in Figure 4.4a.

The meshes used for the coupled solver (CFX) are shown in Figure 4.3, with similar effects occurring for pure tetrahedral meshes, which led to the selection of the combined tetrahedral-prism grids. Here, the tetrahedral mesh representing the majority of the water and air fraction in the NWT is extruded towards the still water level. This results in a prism layer around this region, where the influence from the non-orthogonal vertices and faces of the pure tetrahedral mesh is reduced. Figure 4.4b shows the surface elevation results for the three meshes. The finer meshes give the same results, whereas the coarse mesh (Figure 4.3a) differs from the mesh independent solution. According to the findings in this section, the number of cells per wave height was set to 10 or more for all subsequent simulations.

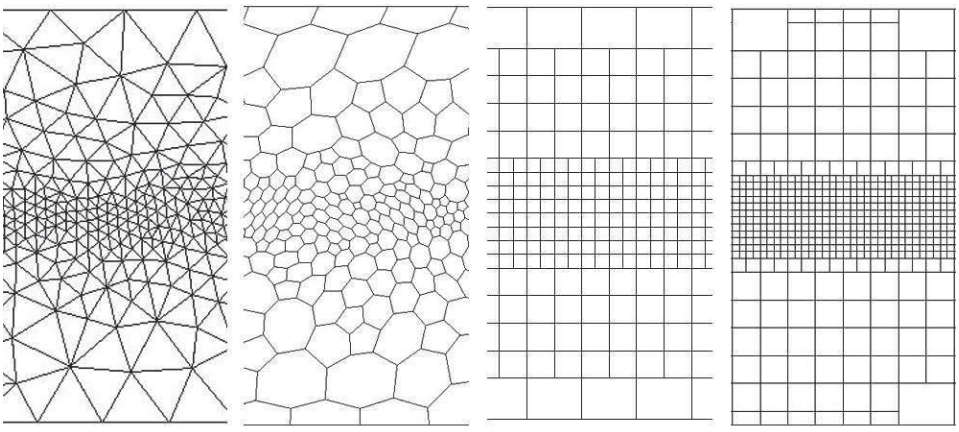


Figure 4.2: Mesh section of STAR CCM+ meshes

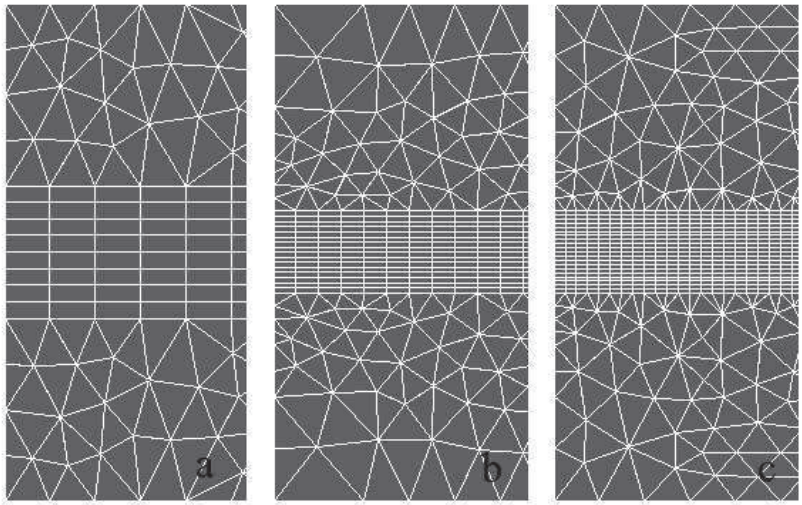


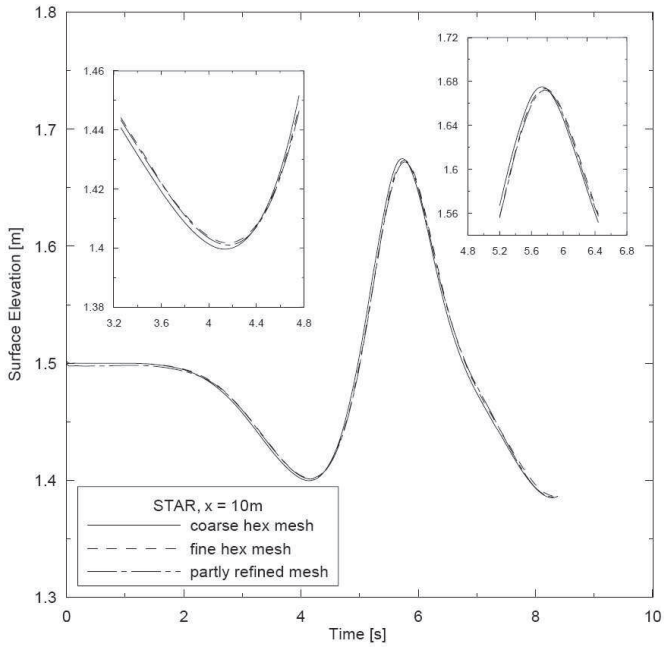
Figure 4.3: Mesh section of CFX 5.7 meshes

4.1.4 Damping of Waveheight in the Numerical Wave Tank

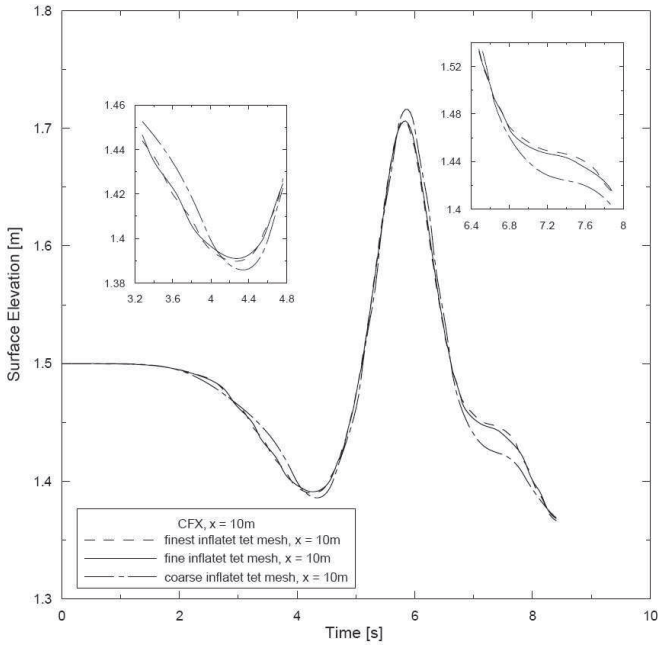
Another issue to resolve is the damping of the waves with distance from the wavemaker. For the wave-structure interaction cases it is important to know the shape and height of waves at a given position in the tank. Figure 4.5 shows the surface elevation in the tank for one timestep. The plot is shown for two different waves with kh/kA = 1.96/0.06 and 0.62/0.2. Although the waves damp out at different rates both waves have their defined height one wavelength behind the wavemaker. This is the preferred location in the tank to put structures, such as the vertical and horizontal cylinders, or the location to focus the extreme waves.

4.2 Focused Waves

In order to simulate an extreme wave of realistic profile for testing the survivability of WECs, NewWave focusing as described earlier in Chapter 2 (Taylor 1992, Taylor



(a) STAR CCM+



(b) CFX

Figure 4.4: Surface elevation at x = 10m

Table 4.1: Mesh properties

	Meshtype	$L/W/H$[m]	Cells	vert. no. of cells
STAR CCM+	Tetrahedral	35/3/3	596124	27
	Polyhedral	35/3/3	99087	16
	Coarse hexahedral	35/3/3	100663	10/8
	Fine hexahedral	35/3/3	543021	12/14
	Partly refined hexahedral	35/3/3	145280	12/14
	Coarse hexahedral	100/3/3	286804	10/8
CFX	Coarse tetrahedral-prism	35/3/3	150480	2x4 tet./8 prism.
	Fine tetrahedral-prism	35/3/3	636386	2x8 tet./16 prism
	Finest tetrahedral-prism	35/3/3	1534058	2x9 tet./20 prism
	Fine tetrahedral-prism	100/3/3	1818200	2x8 tet./16 prism

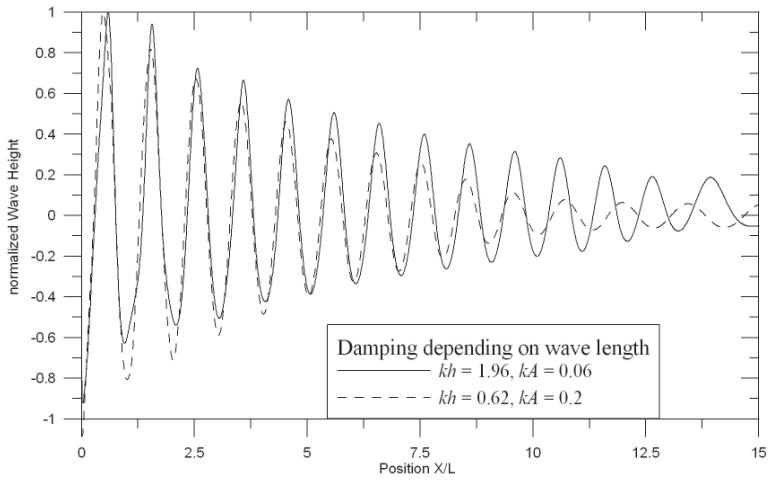


Figure 4.5: Damping in the NWT

& Williams 2002, 2004, Walker et al. 2004) is utilised.

4.2.1 Domain for Focused Waves

For the focused waves the wave tank geometry and set up used in the physical experiments described in Ning et al. (2007) was followed. They used a wave tank with plan dimensions of 69 m x 3 m and a water depth of 0.5 m. The waves were generated by a piston wavemaker and wave reflections were absorbed by a 4 m foam layer placed at the downstream end of the flume. Wave gauges (WG) were used to measure the surface elevation around the point of the maximum wave elevation. In the study by Ning et al. (2007), four NewWave cases are investigated with different input amplitudes; here we reproduce numerically cases 2, 3 and 4 with kA = 0.2, 0.3 and 0.405 respectively.

The NewWave theory describes the surface elevation and wave velocity components of a focused group of localised waves derived from a measured or theoretical spectrum, such as JONSWAP or Pierson-Moskowitz. The waves are superposed and brought into phase at one point in the tank at a specified time. This generates an extreme wave event, which represents the wave environment of the underlying spectrum. By increasing the input focused wave height, wave breaking at a defined location can be achieved. However, with increasing wave height the waves become more non-linear and the location where the waves focus or even break might vary more significantly from the specified focus location and time in the wavemaker signal. In the physical experiment the focus point was set to 11.4 m downstream of the wavemaker.

For both software packages similar domains are used as shown in Figure 4.6. To save computational resources the domain is shortened to 13 m, following the

approach taken by Ning et al. (2007) in their fully non-linear potential flow simulations. The entire domain is 1m high with a water depth of 0.5 m. Between x = 10 m and x = 13 m, a damping layer is installed which prevents reflections from the right-hand boundary. In this area the dynamic viscosity of the water fraction increases linearly from 8.94 x 10^{-4} Pa s up to 1600 Pa s.

The left-hand boundary is a velocity inlet, where the horizontal and vertical velocity components are applied together with the volume fraction of air and water. These are calculated following the derivations described by Dalzell (1999), which are extended to the number of wave components N required for this experiment. The velocities are applied for the water fraction only and the velocity of the air fraction at the inlet boundary is set to zero. The top boundary is a pressure outlet, allowing only air to leave or enter the domain. The remaining boundaries at the bottom and downstream end are walls. No turbulence model is applied.

The vertical coordinates where the water volume fraction is equal to 0.5 are taken to be the position of the free surface and are extracted at different horizontal positions in the tank for each time step. The results are given as surface elevation time history plots and discussed in Sections 4.2.4 - 4.2.4 .

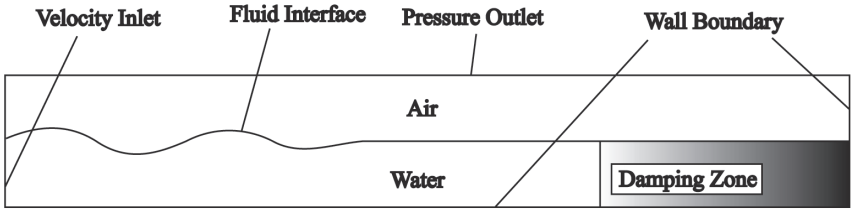


Figure 4.6: Computational domain

For the focused wave simulations, pure hexahedral meshes are used for the CFX simulations. These pure hexahedral meshes became available in the software during

the time of the project, but were not available for the early regular wave simulations. However, due to the restrictions of the mesh generation software it was necessary to use the same mesh resolution for the entire domain, which makes the simulations expensive in relation to a partly refined mesh such as those used for the segregated iterative solver. The simulations are done on a pseudo 2D mesh with one cell layer thickness. For all NewWave cases the same mesh is used. According to grid convergence studies, which are also published in Westphalen et al. (2007) and Westphalen et al. (2008), the number of cells needed in the vertical direction is set to 110 over the entire domain height. Thus for case $kA = 0.2$ the wave height is resolved by 14 cells in the vertical direction, this is sufficient to resolve the free surface accurately. For case $kA = 0.3$ and 0.405 with $A = 0.0875$ m and 0.1031 m this gives 18 and 22 cells respectively.

For the wave cases with $kA = 0.2$ and 0.3 the timestep is set to 0.01 s. These settings ensure a Courant number smaller than or close to 1. These settings resulted in instabilities for the last NewWave case at the inlet. Here, the vertical motion of the water surface per timestep due to the superposition of the N wave components was too large for the cell size. By reducing the timestep to 0.005 s the issue could be solved.

The calculations for the FV solver are carried out on a hexahedral mesh, which is refined around the free surface similar to the ones shown in Figure 4.2. Due to the increase in the expected wave height, the refined area is vertically extended according to the linear target amplitude. Additionally the cells at the inlet are refined for better definition of the velocity field and the rapidly moving free surface during the calculation. The refined area ends at $x = 6$ m. Beyond that point the mesh is uniform having the same cell size as the regions above and below the refined region.

The calculation for case kA = 0.2 is made with a timestep of 0.001s. The two other cases are started with the same timestep for the first 5s of the simulation reduces to 0.00025 s, for the same reasons as described for the coupled CV-FE solver (CFX).

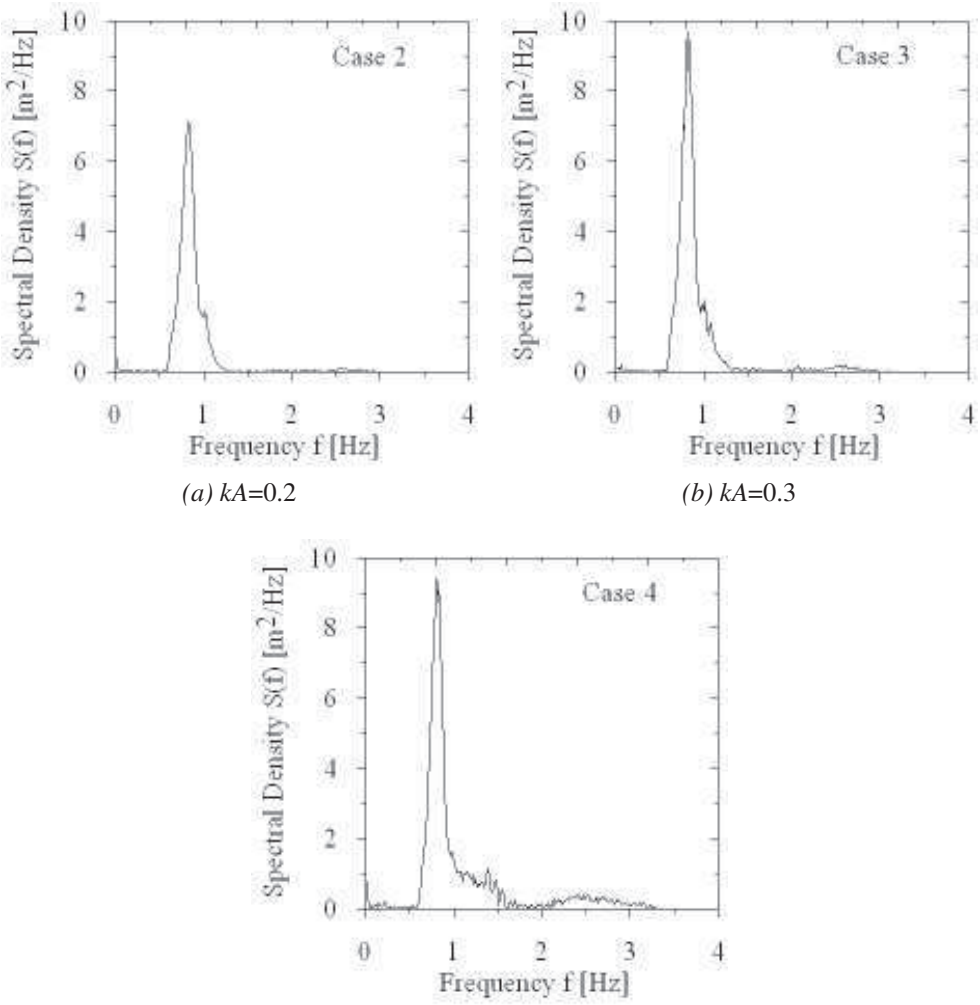
4.2.2 Generation of Focused Waves

The concept of the NewWave formulation is to generate a number of waves of relatively small amplitudes and range of periods (Taylor & Williams 2002, 2004). These waves interact with one another and thereby build up a localised extreme wave, larger than any individual wave created at the paddle, focused at a known time and location in the tank. In the numerical calculations the waves are generated from the spectra shown in Figure 4.7. These spectra are obtained by Fast Fourier Transformation of the free surface data measured in the experiments as described by Ning et al. (2007) and Zang et al. (2006).

For each wave component n the amplitude a_n is calculated as

$$a_n = A\frac{S_n(f)\Delta f}{\sum_N S_n(f)\Delta f} \tag{4.4}$$

where $S(f)$ is the spectral density, Δf is the frequency step depending on the number of wave components N and bandwidth. A is the target linear amplitude of the focused wave. Thus, the amplitude of every spectral component in the NewWave group scales as the power density within that frequency band in the assumed sea-state. Equivalently, NewWave is simply the scaled auto-correlation function corresponding to a specified frequency spectrum such as JONSWAP or Pierson-Moskowitz. The properties for each case considered are as shown in Table



(a) kA=0.2

(b) kA=0.3

(c) kA=0.4

Figure 4.7: Input spectra for generation of focused wave group

Table 4.2: Properties of simulated cases

kA	f_{range}[Hz]	N	A [m]	T [s]
0.2	0.6 - 1.3	16	0.0632	1
0.3	0.6 - 1.4	20	0.0875	1.25
0.405	0.5 - 1.8	16	0.1031	1.25

4.2.

The underlying equations, from which the signal of the physical wavemaker is derived, come from second order Stokes theory and are given by Ning et al. (2007) and Dalzell (1999). The input signal for the CFD runs is the sum of the first $^{(1)}$ and the second $^{(2)}$ order component for the horizontal and vertical water velocity component u and w and the surface elevation η:

$$\eta = \eta^{(1)} + \eta^{(2)}, \tag{4.5}$$

$$u = u^{(1)} + u^{(2)} \tag{4.6}$$

and

$$w = w^{(1)} + w^{(2)}. \tag{4.7}$$

The first order component for the surface elevation is

$$\eta^{(1)} = \sum_{i=1}^{N} a_i \cos \psi_i \tag{4.8}$$

where ψ_i is

$$\psi_i = k_i x - \omega_i t + \varepsilon_i. \tag{4.9}$$

k is the wavenumber, x the Cartesian coordinate, ω the angular frequency, t is time and ε the phase angle (which is zero for all simulations). The first order velocities $u^{(1)}$ and $w^{(1)}$ are

$$u^{(1)} = \sum_{i=1}^{N} \frac{a_i k_i g}{\omega_i} \frac{\cosh k_i (z+h)}{\cosh k_i h} \cos\left(k_i (x-x_0) - \omega_i (t-t_0) + \varepsilon_i\right) \tag{4.10}$$

and

$$w^{(1)} = \sum_{i=1}^{N} \frac{a_i k_i g}{\omega_i} \frac{\sinh k_i (z+h)}{\cosh k_i h} \sin\left(k_i (x-x_0) - \omega_i (t-t_0) + \varepsilon_i\right), \tag{4.11}$$

where x_0 and t_0 are the focus location and time and z is the vertical Cartesian coordinate at the wavemaker.

The second order parts for the surface elevation $\eta^{(2)}$ and the velocities $u^{(2)}$ and $w^{(2)}$ are given by

$$\eta^{(2)} = \sum_{i=1}^{N} \sum_{j=i+1} a_i a_j B^{+}_{i,j} \cos\left(\psi_i + \psi_j\right) + \sum_{i=1}^{N} \sum_{j=i+1} a_i a_j B^{-}_{i,j} \cos\left(\psi_i - \psi_j\right)$$

$$+ \sum_{i=1}^{N} \frac{a_j^2 k_i}{4 \tanh\left(k_i h\right)} \left[2 + \frac{3}{\sinh^2\left(k_i h\right)}\right] \cos\left(2\psi_i\right) - \sum_{i=1}^{N} \frac{a_j^2 k_i}{2 \sinh\left(2 k_i h\right)} \tag{4.12}$$

and

$$\Phi^{(2)} = \sum_{i=1}^{N} \sum_{j=i+1} a_i a_j A^{+}_{i,j} \frac{\cosh\left(\left(k_i + k_j\right)(z+h)\right)}{\cosh\left(\left(k_i + k_j\right) h\right)} \sin\left(\psi_i + \psi_j\right)$$

$$+ \sum_{i=1}^{N} \sum_{j=i+1} a_i a_j A^{-}_{i,j} \frac{\cosh\left(\left(k_i - k_j\right)(z+h)\right)}{\cosh\left(\left(k_i - k_j\right) h\right)} \sin\left(\psi_i - \psi_j\right)$$

$$+ \sum_{i=1}^{N} a_i^2 \frac{3\omega_i}{8} \frac{\cosh\left(2k_i + k_j h\right)}{\sinh^4\left(k_i h\right)} \sin\left(2\psi_i\right), \tag{4.13}$$

with components of the interaction kernels $A_{i,j}^{+/-}$, $B_{i,j}^{+/-}$ and $D_{i,j}^{+/-}$ defined as

$$A_{i,j}^{+} = -\frac{\omega_i\omega_j\left(\omega_i+\omega_j\right)}{D_{i,j}^{+}}\left[1-\frac{1}{\tanh\left(k_ih\right)\tanh\left(k_jh\right)}\right] + \frac{1}{2D_{i,j}^{+}}\left[\frac{\omega_i^3}{\sinh^2\left(k_ih\right)}+\frac{\omega_j^3}{\sinh^2\left(k_jh\right)}\right] \tag{4.14}$$

$$A_{i,j}^{-} = \frac{\omega_i\omega_j\left(\omega_i-\omega_j\right)}{D_{i,j}^{-}}\left[1+\frac{1}{\tanh\left(k_ih\right)\tanh\left(k_jh\right)}\right] + \frac{1}{2D_{i,j}^{-}}\left[\frac{\omega_i^3}{\sinh^2\left(k_ih\right)}-\frac{\omega_j^3}{\sinh^2\left(k_jh\right)}\right] \tag{4.15}$$

$$B_{i,j}^{+} = \frac{\left(\omega_i^2+\omega_j^2\right)}{2g}-\frac{\omega_i\omega_j}{2g}\left[1+\frac{1}{\tanh\left(k_ih\right)\tanh\left(k_jh\right)}\right] \cdot\left[\frac{\left(\omega_i+\omega_j\right)^2+g\left(k_i+k_j\right)\tanh\left(\left(k_i+k_j\right)h\right)}{D_{i,j}^{+}}\right] + \frac{\left(\omega_i+\omega_j\right)}{2gD_{i,j}^{+}}\left[\frac{\omega_i^3}{\sinh^2\left(k_ih\right)}+\frac{\omega_j^3}{\sinh^2\left(k_jh\right)}\right] \tag{4.16}$$

$$B_{i,j}^{-} = \frac{\left(\omega_i^2+\omega_j^2\right)}{2g}+\frac{\omega_i\omega_j}{2g}\left[1-\frac{1}{\tanh\left(k_ih\right)\tanh\left(k_jh\right)}\right] \cdot\left[\frac{\left(\omega_i-\omega_j\right)^2+g\left(k_i-k_j\right)\tanh\left(\left(k_i-k_j\right)h\right)}{D_{i,j}^{-}}\right] + \frac{\left(\omega_i-\omega_j\right)}{2gD_{i,j}^{-}}\left[\frac{\omega_i^3}{\sinh^2\left(k_ih\right)}-\frac{\omega_j^3}{\sinh^2\left(k_jh\right)}\right] \tag{4.17}$$

$$D^{+}_{i,j} = \left(\omega_i + \omega_j\right)^2 - g\left(k_i + k_j\right)\tanh\left(\left(k_i + k_j\right)h\right) \tag{4.18}$$

and

$$D^{-}_{i,j} = \left(\omega_i - \omega_j\right)^2 - g\left(k_i - k_j\right)\tanh\left(\left(k_i - k_j\right)h\right) \tag{4.19}$$

The second order velocity components are obtained by the relevant spatial derivatives of the second order potential $\Phi^{(2)}$ to be differentiated with respect to x and z in order to get the second order contribution to the inlet signal.

The x-coordinate is zero for the calculations, because the wavemaker sits at x = 0 m; x_0 is the focus location, which is set to 3 m for case kA = 0.2 and to 3.27 m for both other cases; t is the time; t_0 is the focus time, which is 9.2 s for case kA = 0.2 and 10s for the others. Also in the formulae are the wavenumber k_i, the frequency ω_i , the phase angle ε_i (which is set to 0 for the calculations), the water depth h, vertical position z and the number of wave components N.

For the simulations as presented here, the incoming wave entering the computational domain is fluxed in through a transparent boundary. This flux is defined in terms of either linear theory, or linear theory with second order corrections.

4.2.3 Implementation of Boundary Conditions

The implementation of the boundary condition is dependent on the user interface of the commercial software package. Both software packages use their own script languages to express simple mathematical equations and logical operations. Additionally, both also provide a user programming facility for more complex requirements. These offer more power and flexibility as they use traditional programming languages and thereby benefit from their libraries. In the case of CFX, the user can

programme a script in FORTRAN or PERL. STAR CCM+ uses Java and C++. As described earlier, for both packages a velocity inlet was chosen, where the horizontal and vertical velocity components need to be specified. Furthermore, the volume fraction, representing the surface elevation arising from the wave equations needs to be set. The wave velocities however, as the inlet reaches over the whole domain height including the air fraction, need to be restricted to the water fraction only. Otherwise one would experience a wind blowing in the domain as well. Thus there exist two requirements to the setup:

- Definition of the velocities using the internal script language or the user programming facility of the package
- Restriction of the velocities to the water fraction only

For simple mathematical equations and fairly short strings the internal script language is used. This approach is valid for regular waves, such as given by equations (4.1), (4.2) and (4.3) and 1st order focused waves as given by equations (4.8), (4.10) and (4.11). Here, u, v and η can be entered directly. Then these expressions are treated by a logical operation using the script language, which restricts the velocities to the water fraction defined by the position of the water volume fraction = 0.5, which is equivalent to the surface elevation η. For every cell at the inlet boundary, this operation can be expressed in words as follows:

- If the difference of water-fraction and air-fraction in the cell is positive, then multiply the velocity expression by 1 and apply at the boundary cell face, otherwise multiply by 0 and apply.

The reason for the use of the user programming facilities can either be that the string gets too long or difficult operations need to be carried out, such as iterations or other methods in order to receive the full boundary condition expression. Here, the velocity components and surface elevations including the second order contributions are calculated in a Java macro and a FORTRAN routine for STAR CCM+ and CFX respectively. The way both packages incorporate such user scripts is different and can be seen in Figure 4.8. The CFX solver is started and communicates several times during the timestep loop with the script and passes the variables, which are needed to calculate the wave equations to the script, i.e. the time and the z coordinate at the inlet. The FORTRAN routine returns a single number for the appropriate boundary cell which is then multiplied with the restriction expression.

The STAR CCM+ solver process on the other hand is started and controlled from the Java macro. The macro is called once before the timestep and updates the solver file with the expressions for the velocity components and surface elevations. These are still strings, but can be simplified as much as possible within the Java macro. As the time is known for every timestep beforehand, only the vertical variable is left unknown and passed on to the solver. Then the boundary values are calculated as described before.

For further interest, example scripts used for the generation of the waves in STAR CCM+ and CFX are given in the Appendix.

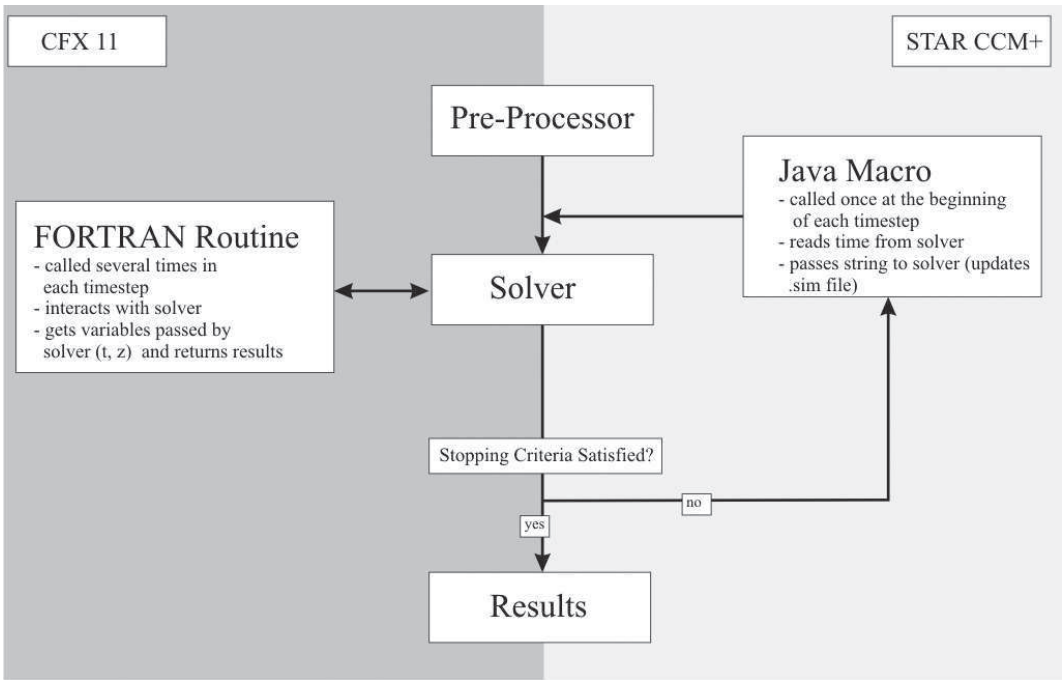


Figure 4.8: User scripting in STAR CCM+ and CFX

4.2.4 Focused wave results

Case 1, kA = 0.2

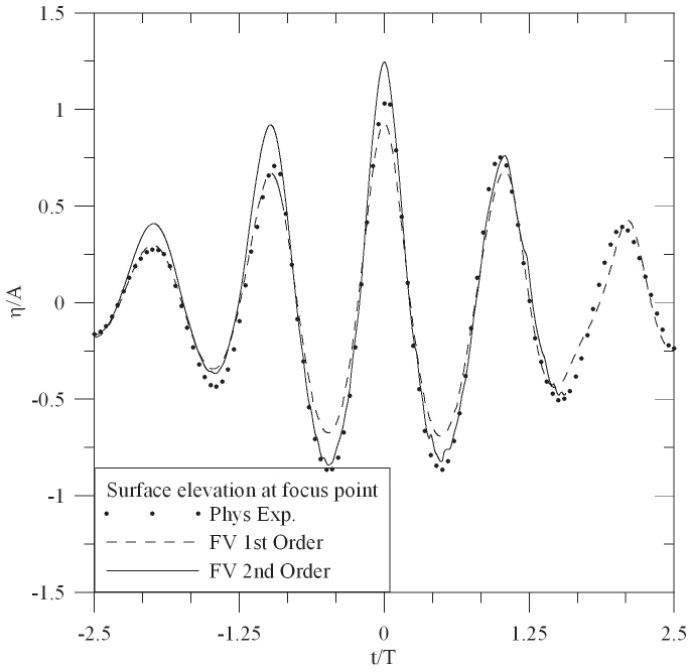
In total 16 simulations using the segregated iterative FV and coupled CV-FE solvers were performed in order to compare the surface elevations at the focus point with the experimental results by Ning et al. (2007). For both solvers, results are shown that correspond to input waves defined using linear and linear plus second order theory. The surface elevations are non-dimensionalised in terms of the linear amplitude of the target NewWave A, and time is defined in terms of the appropriate central wave period T, as shown in Table 4.2. Table 4.3 shows the maximum surface elevations for these cases. As the actual focus point and time differ for every simulation, the results are shifted in time to coincide with one another. They are adjusted at t/T =

Table 4.3: Maximum surface elevations of focused waves

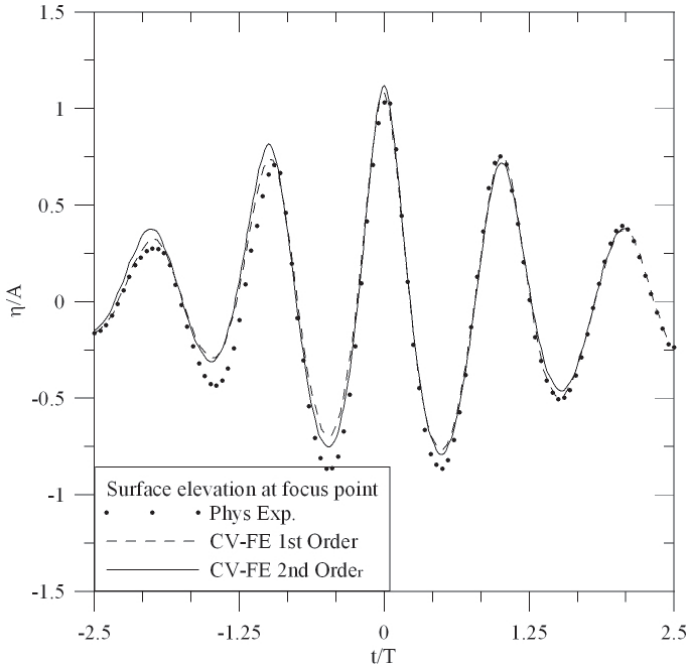
		CV-FE (CFX)		FV (STAR CCM+)		Phys. Exp.	
		Elev.	Dist.	Elev.	Dist.	Elev.	Dist.
Case 2	1st Ord.	1.08	3.0 m	0.93	2.61 m	NA	NA
	2nd Ord.	1.12	3.68 m	1.24	3.29 m	1.03	3.0 m
Case 3	1st Ord.	1.13	3.68 m	1.17	3.68 m	NA	NA
	2nd Ord.	1.17	4.08 m	1.23	4.08 m	1.10	3.29 m
Case 4	1st Ord.	1.07	3.68 m	1.26	3.68 m	NA	NA
	2nd Ord.	1.27	3.0 m	1.28	3.68 m	1.22	3.29 m

0, which is taken as the time when the maximum surface elevation occurs. Figures 4.9 - 4.11 show the water surface elevation from the two solvers compared with the physical experiment for all crest focused cases.

For a wave steepness of $kA = 0.2$, the case of the weakest non-linearity, the segregated iterative solver with first order wave input signal (Figure 4.9a) does not reach the required height. The simulation including the second order wave components, however, overestimates the crest elevations by approximately 24%, with a value of 0.015 m for an input amplitude of 0.0632 m. The trough elevations for the surrounding troughs improve from first order to first plus second order wave setup for the FV solver in terms of matching with the physical experiments. Figure 4.9b shows that the coupled solver predicts surface elevations for both wave signals slightly higher than those measured in the physical tank tests. Here the troughs that surround the central wave are higher and do not exactly coincide with the results of the experimental results, although one can see an improvement from first order to second order wave signal.



(a) FV (STAR CCM+)



(b) CV-FE (CFX)

Figure 4.9: Surface elevation at focus point for $kA = 0.2$ (Case 2)

Case 2, kA = 0.3

The numerical results for case kA = 0.3, presented in Figure 4.10, generally show the best agreement with the physical experiments; particularly the central crest elevations are predicted with only a slight difference for both input signals. Moving from first to second order wave input also improves the trough elevations.

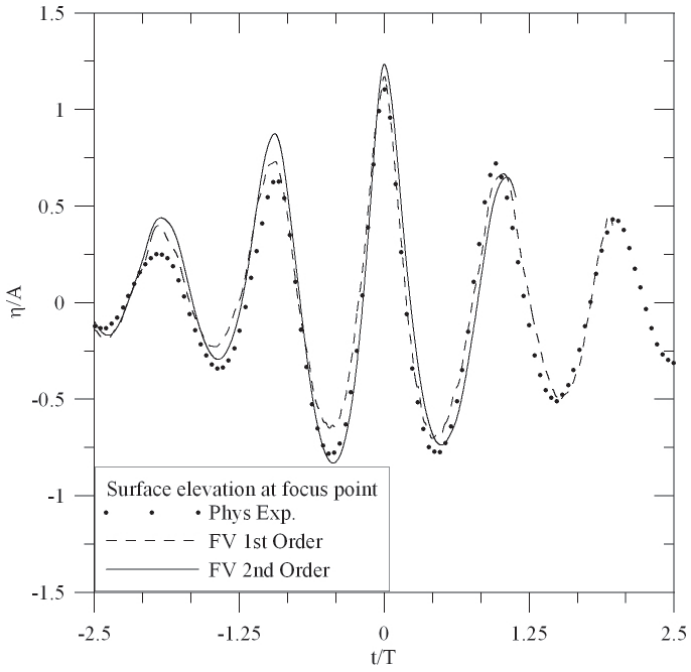
Case 3, kA = 0.405

This case is the test with the steepest wave, which almost broke in the physical experiment. The numerical results are very good for the maximum crest elevations. However, the surrounding wave train does not agree as well as the previous tests, though the trends are the same, as can be seen in Figure 4.11. The wave preceding the central wave is much larger than the measured values for both codes, but still smaller and not symmetric to the wave that follows the main wave, as can also be observed in Figure 4.11.

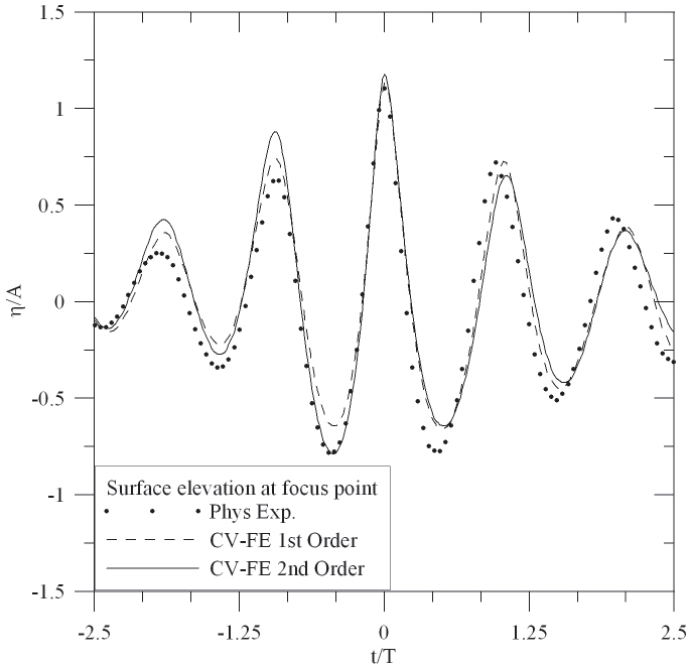
The actual focus point in the numerical calculations lies further downstream than specified, as can be seen from Table 4.3. Also, it is further away for the simulations with higher order wave signal.

4.2.5 Non-linearity of Focused Waves

A useful tool to asses the linearity of results is the comparison of the sum and difference plots. Therefore additional simulations have to be carried out, with trough focused instead of crest focused waves. By subtracting and summing the appropriate time histories (and dividing them by two) the plots for cases kA = 0.2 and 0.405, shown in Figure 4.12, are obtained. The difference represents the linear part of the

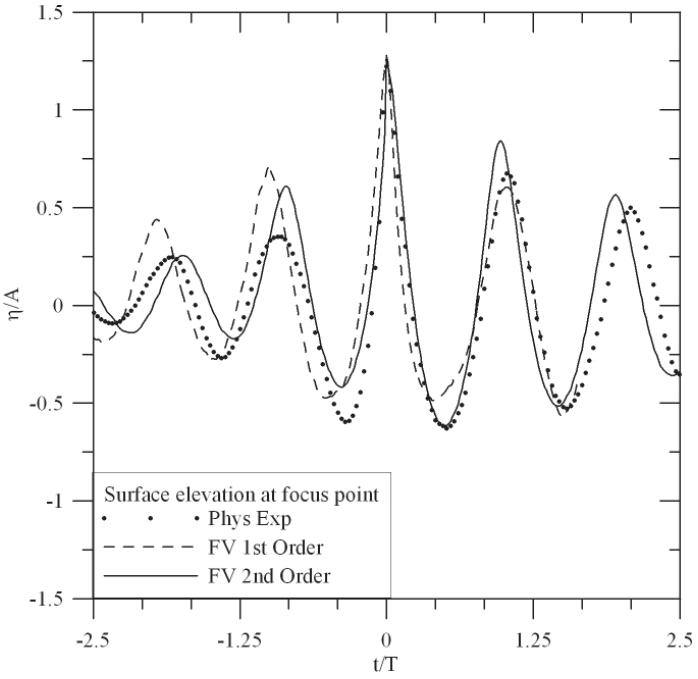


(a) FV STAR CCM+)

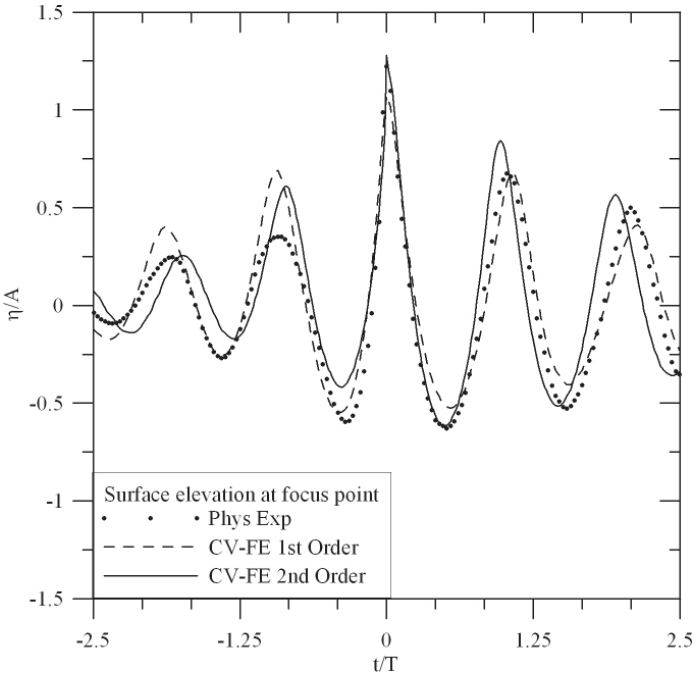


(b) CV-FE (CFX)

Figure 4.10: Surface elevation at focus point for $kA = 0.3$ (Case 3)

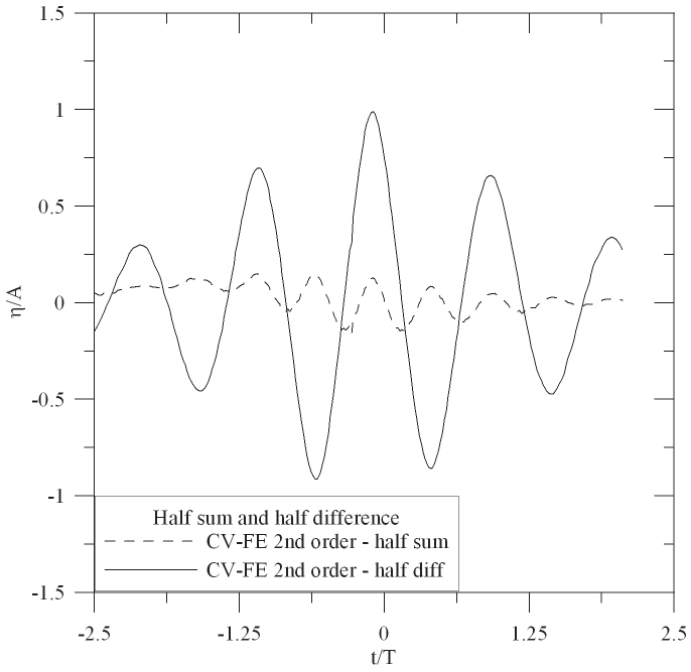


(a) FV (STAR CCM+)

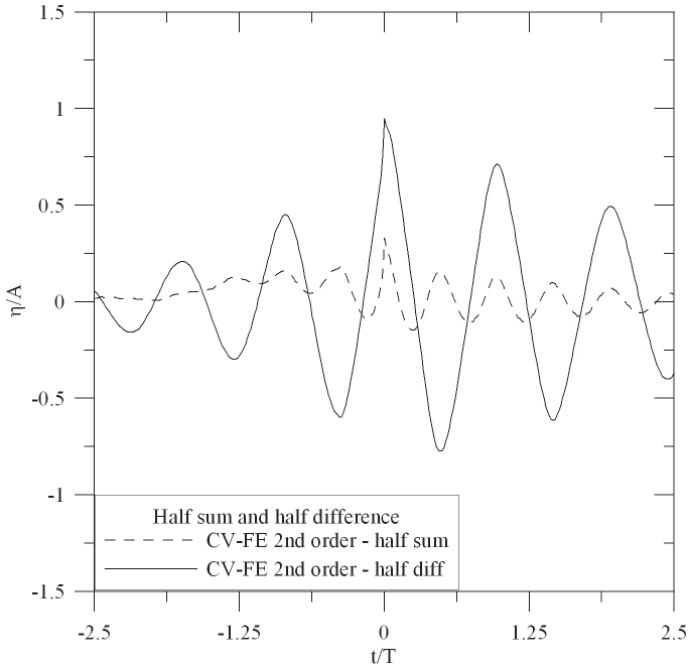


(b) CV-FE (CFX)

Figure 4.11: Surface elevation at focus point for $kA = 0.405$ (Case 4)



(a) Case 2



(b) Case 4

Figure 4.12: Sum and difference plots for the CV-FE solver (CFX)

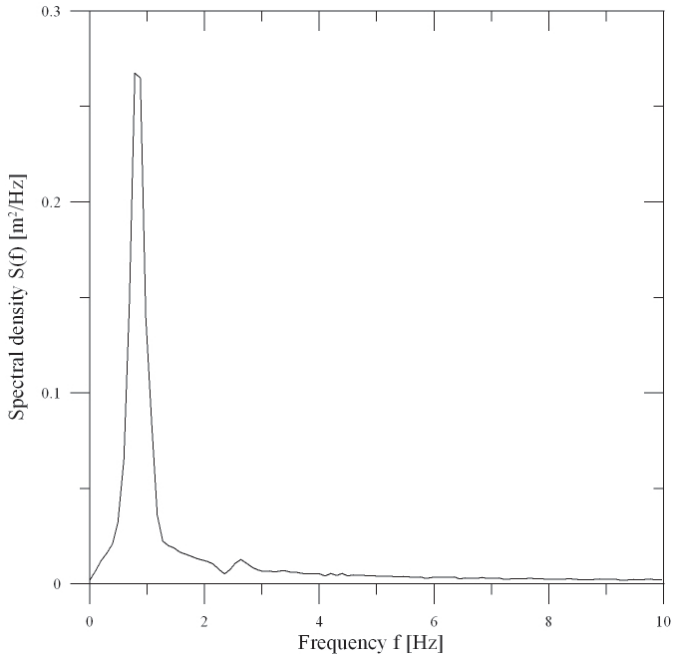
solution plus the odd harmonics and the sum is composed from the even harmonics of the wave time history. As expected, the non-linearities increase from $kA = 0.2$ to the steeper and thereby more non-linear case with $kA = 0.405$.

The sum and difference plots can then be analysed by performing a FFT. This gives the frequency spectrum of the even and odd components in the processed signal. Figure 4.13 shows the FFT results for the $kA = 0.2$ focused wave group. The wave frequencies contributing to the linear signal plus the odd harmonics can be identified through the peaks around 0.87 Hz and 2.7 Hz in Figure 4.13a. By applying a bandpass filter for these frequencies to the difference data from Figure 4.12, the linear signal, shown as the dashed line in Figure 4.14a and the 3rd order harmonics, seen as solid line in Figure 4.14a, can be separated from the difference signal. The same procedure is applied to the even harmonics, which peak at 0.09 and 1.66 Hz. The bandpass filtering of the sum data results in the half and double frequency second order components shown in Figure 4.14b. It should be noted that the vertical axis scale of both graphs is different. The linear component contributes almost 95 % to the overall wave elevation for the wave steepness $kA = 0.2$. The second harmonics contribute approximately 4 % each and the third about 1 % to the signal.

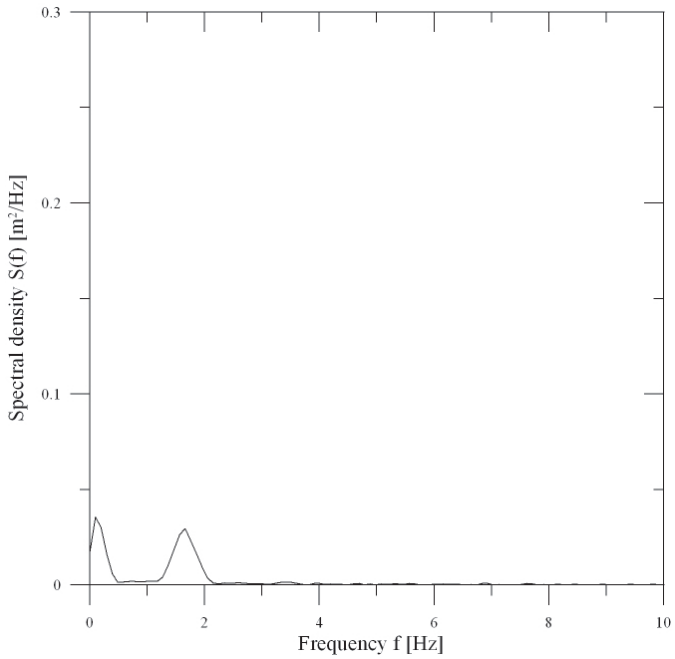
The same procedure has been applied to the appropriate data of the steepest NewWave case with $kA = 0.405$. Here, the spectra obtained through FFT from the sum and difference plots (Figure 4.12b) can be seen in Figure 4.15. For this case the linear part can be identified at 0.87 Hz, whereas the third harmonic is more difficult to spot in Figure 4.15a. It is taken to peak at 2.53 Hz. The 2nd harmonics can be seen in Figure 4.15b peaking at 0.097 and 1.75 Hz.

Using these values for the bandpass filter applied to the sum and difference data

results in the graphs shown in Figure 4.16. For this more non-linear case the linear part only adds 80 % to the total wave elevation time history. Both 2nd harmonics account for approximately 15 % each of the free surface elevation signal at the focus point.

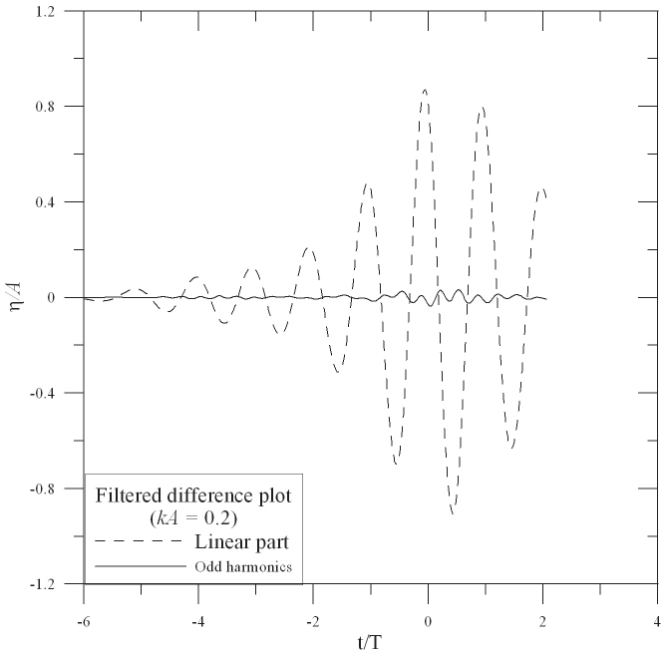


(a) Linear part and odd harmonics

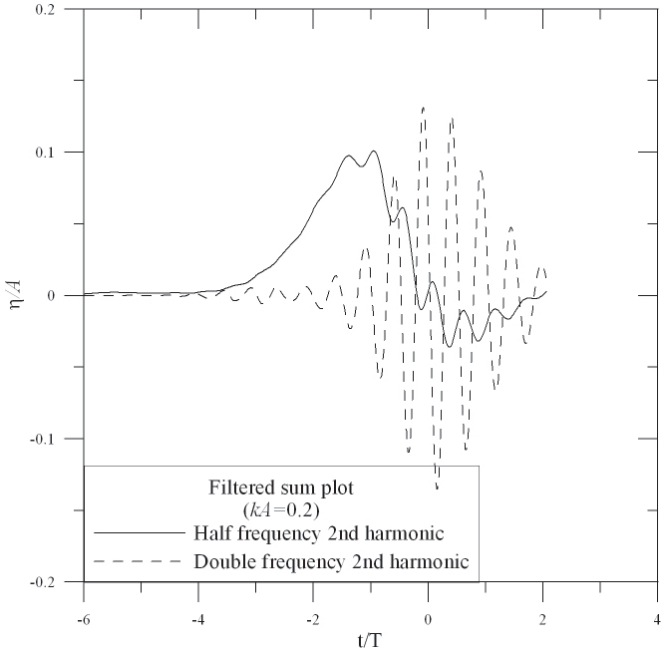


(b) Even harmonics

Figure 4.13: Spectra from difference (a) and sum plots (b) , kA = 0.2, Case 2

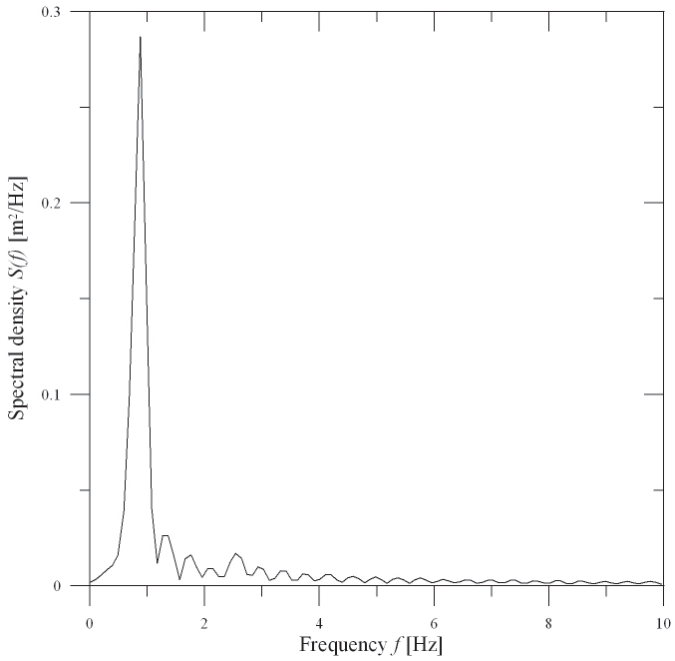


(a) Linear part and odd harmonics

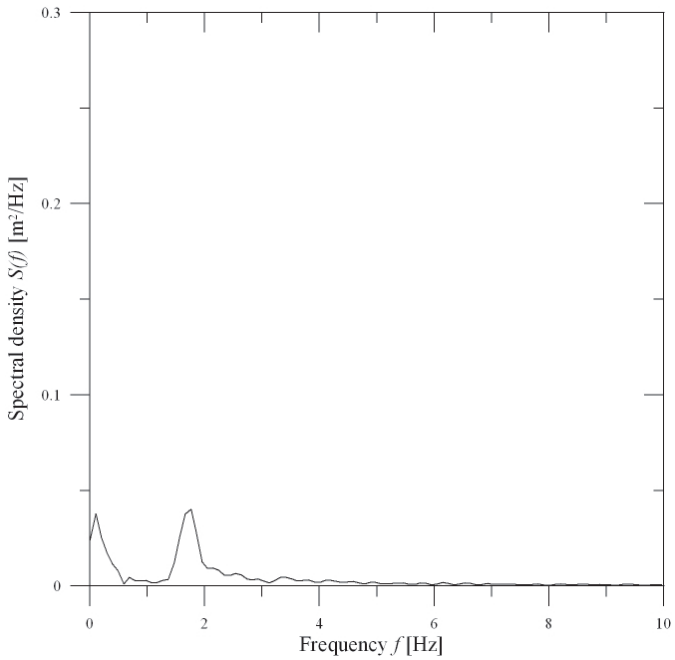


(b) Even harmonics

Figure 4.14: Filtered difference (a) and sum (b) results , $kA = 0.2$, Case 2

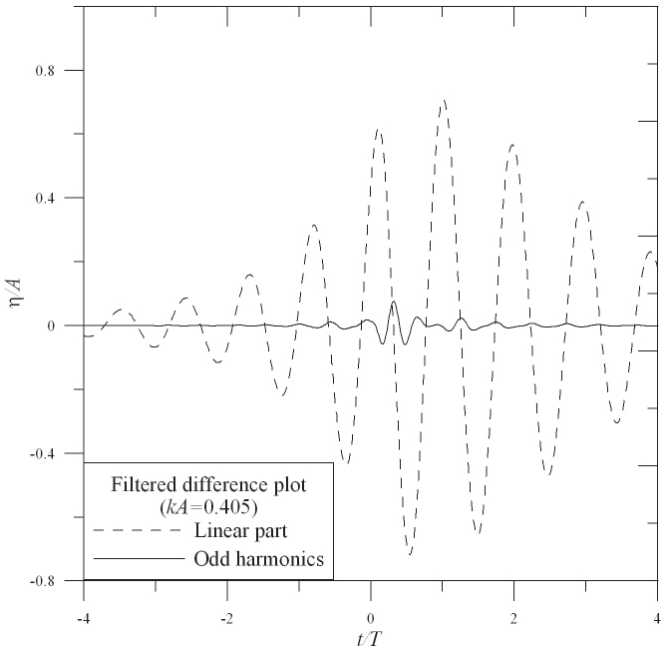


(a) Linear part and odd harmonics

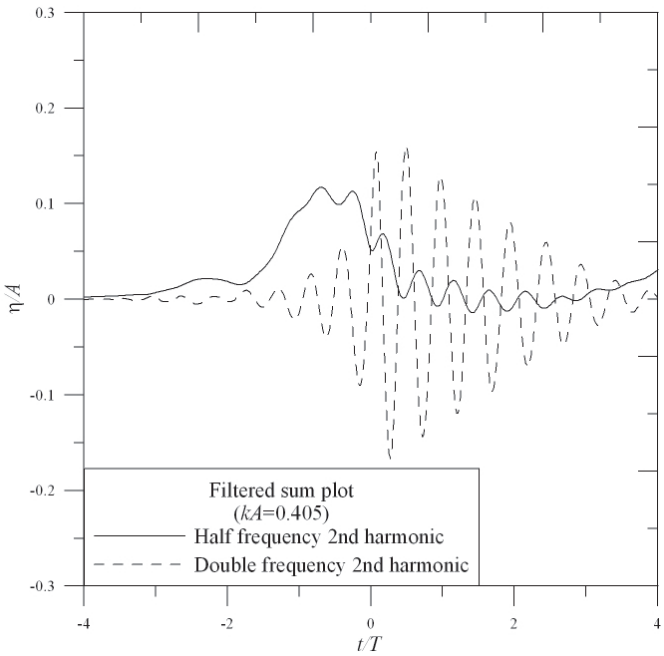


(b) Even harmonics

Figure 4.15: Spectra from difference (a) and sum plots (b) , kA = 0.405, Case 4



(a) Linear part and odd harmonics



(b) Even harmonics

Figure 4.16: Filtered difference (a) and sum (b) results , $kA = 0.405$, Case 4

Chapter 5

Wave-Structure Interaction

In this section the problem considered is the fluid structure interaction between regular waves and horizontal and vertical cylinders. The waves used for the cylinder computations are first order accurate as described in the publications by Dixon et al. (1979) and Kriebel (1998), with which they are compared. Here, the vertical and horizontal forces on the cylinder are of interest and the interplay of buoyancy and inertial force is discussed. A special case in the paragraph about the vertical cylinder is the section about excitation of the cylinder (Section 5.1.3). The so-called "ringing" effect will be shown in the results. Additionally the force and resulting surface elevations from the driven motion of an oscillating cone on the water surface are investigated. The motion of the cone is not influenced by the pressure and viscous forces from the water. The simulations involve mesh motion through deformation of the cells in the vertical direction as the cylinder oscillates vertically during the simulation.

5.1 Vertical Cylinder

This section describes the numerical simulations of a vertical cylinder in two regular wave environments. The first set of tests reproduces the measured results in physical experiments by Kriebel (1998). Kriebel compares the horizontal forces on a vertical circular cylinder in regular waves with computations using first and second order diffraction theory. From this publication two cases are chosen as described in Table

5.1. k is the wavenumber, A the wave amplitude, h the water depth and T the wave period. Furthermore, results of a slender vertical cylinder in steep waves of about the same diameter as wave height are discussed.

Table 5.1: Properties of Vertical Cylinder Simulations

	Simulation	
	1	2
kA	0.09	0.34
kh	1.61	1.61
N_{KC}	0.32	0.54
A [m]	0.053	0.088
T [s]	1.95	1.05

5.1.1 Computational Domain

The numerical wavetanks (NWT) for the CFX and STAR CCM+ are identical. The simulations are performed in a 3-dimensional domain with the dimensions x, y and z equal to 12, 1.65 and 0.9, with z being the vertical direction as shown in Figure 5.1. The still water level is 0.45 m. The diameter of the cylinder is 0.325 m as in the physical experiments. The centre of it is located 3.77 m downstream from the inlet, which is equal to 1 wavelength for the first wave setup and approximately 2 wavelengths for the second. This distance was shown to work best for different wave environments in Chapter 4.1.4.

As for the wave-wave interaction cases the waves are generated using the horizontal and vertical velocity components, which are applied underneath the appropriate surface elevations at the left hand side velocity inlet. The sides are symmetry planes and the top is a pressure outlet, with air being allowed to leave or enter the

domain. The remaining boundaries are modelled as walls, i.e. the cylinder, the bottom and the far end boundary.

The meshes, however, are different for each of the solvers and are shown in Figure 5.2. The grid that is used for the coupled solver contains 570,000 hexahedral cells. It is refined around the cylinder and the area where the free water surface is expected to travel, upstream of the cylinder. Downstream of the cylinder the mesh is relatively coarse to save computational resources. The cell size around the free surface upstream of the cylinder is 0.011 m. The cylinder is discretised by 3240 quadrilateral elements. The number of cells covering the perimeter is 72, which is the same over the entire cylinder height. The refinement of the cylinder in the area where the waves hit the structure is in the vertical direction only. The timestep for the CV-FE solver (CFX) is 0.005 s.

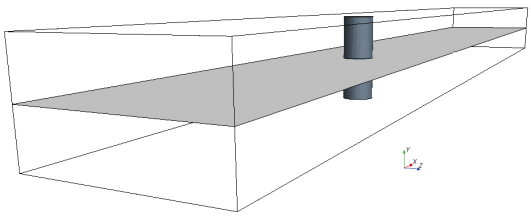
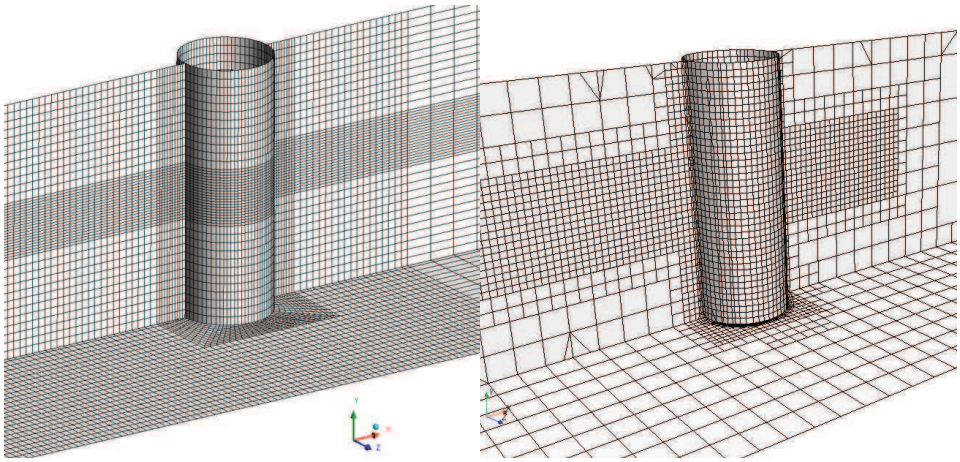


Figure 5.1: Numerical wavetank for vertical cylinder

The mesh used for the FV solver (STAR CCM+) contains 870,000 mostly hexahedral cells. The cells around the water surface area are isotropically refined to an edge length of 1.25 cm, which gives the necessary resolution of approximately 10 cells per wave height (Westphalen et al. 2008). The cylinder itself is modelled with 3005 faces, of which 37 are triangular, 2898 quadrilateral and 70 polygonal. The different cell types result from the meshing algorithm, which cuts the geometry out of the initial hexahedral mesh rather than using a body fitted grid as it is done for



(a) CFX *(b)* STAR CCM+

Figure 5.2: Mesh around large vertical cylinder

the CV-FE solver. At the top and the bottom the cylinder contains 52 faces around the perimeter. In the centre region of the cylinder the mesh is refined not only in the vertical direction, as it is done for the CV-FE solver, but also tangentially. Between the vertical positions of 0.35 m and 0.55 m 104 cells wrap the cylinder perimeter.

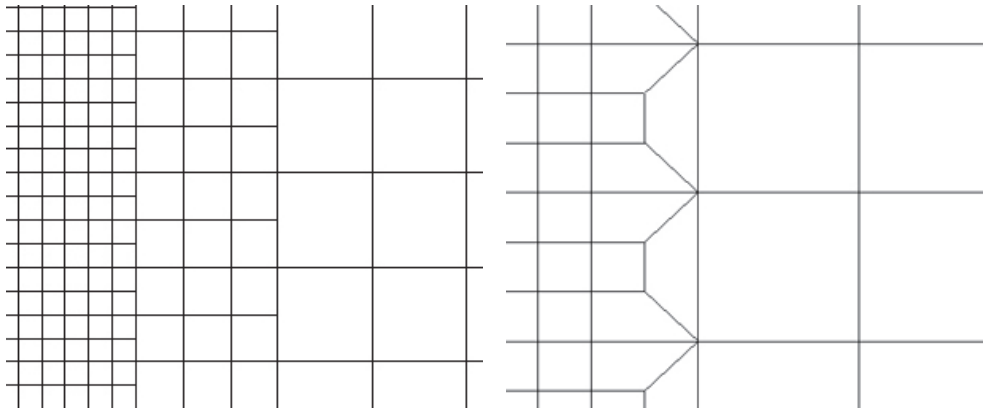
The mesh used for the CV-FE solver (CFX) is body fitted. Here, the hexahedral cells are deformed around the cylinder to coincide with the geometric requirements. This does not affect the orientation of the vertices in the vertical and longitudinal direction and therefore the solution of the free surface around the structure is unaffected.

The mesh used for the FV solver uses a cut-cell approach. The initial surface mesh is built of tetrahedral cell from which the initial volume mesh consisting entirely of hexahedral cells is generated. The structure, here the vertical cylinder, is cut out of the initial volume mesh which results in the polygonal, quadrilateral

and triangular faces representing the geometry. Figure 5.2b shows the mesh in the vicinity of the large cylinder for the segregated iterative FV solver. As the region of interest lies near the cylinder, the downstream part (right side of the cylinder in the figures) of the tank can be modelled using a relatively coarse mesh. This supports the damping layer at the end of the NWT by introducing additional numerical damping to prevent wave reflections from the downstream wall. The coarsening towards that region is shown in Figure 5.3. The mesh used for the segregated iterative solver (Figure 5.3a) employs a 1 cell - 2 cell/1 face - 2 face approach, whereas the mesh for the coupled solver requires the arrangement shown in Figure 5.3b. Here, one cell refines to three cells with every cell face coinciding with one neighbour cell face. In three dimensions the coupled solver still requires that each cell face is met by one neighbouring cell face. If the refinement does not meet these properties, the faces between non-matching cell regions are defined as internal interfaces, which might result in numerical inaccuracies when calculating the flow across the faces. The FV solver can deal with anisotropic refinement in 3 dimensions. One cell can sit next to 2 or 4 cells. The face area, however, must be the same for the coarse cell compared to the sum of the refining cells.

5.1.2 Results of vertical cylinder case

The validation of the codes is achieved by comparing the predicted horizontal forces on the cylinder due to the waves with the experimental results from Kriebel (1998). As described by Kriebel the forces are normalised by the analytic results from linear



(a) FV (STAR CCM+) *(b)* CV-FE (CFX)

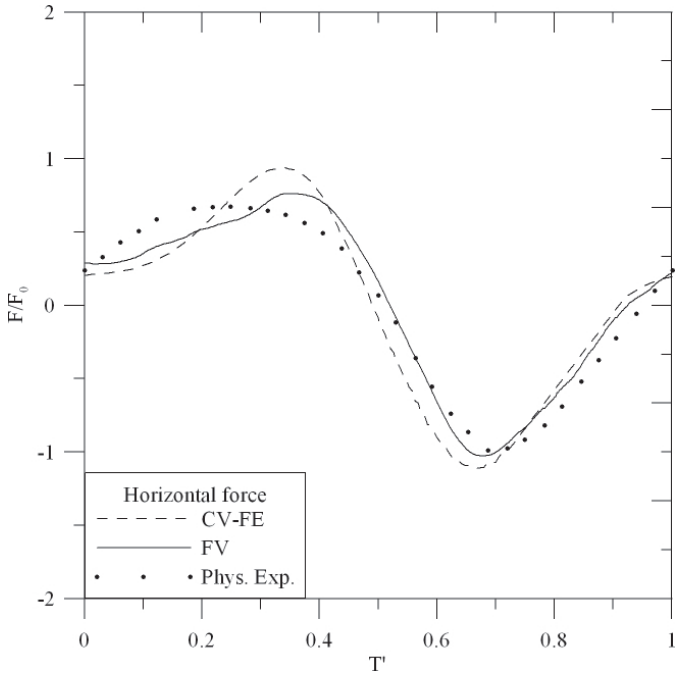
Figure 5.3: Mesh refinement for both codes

diffraction theory, resulting in

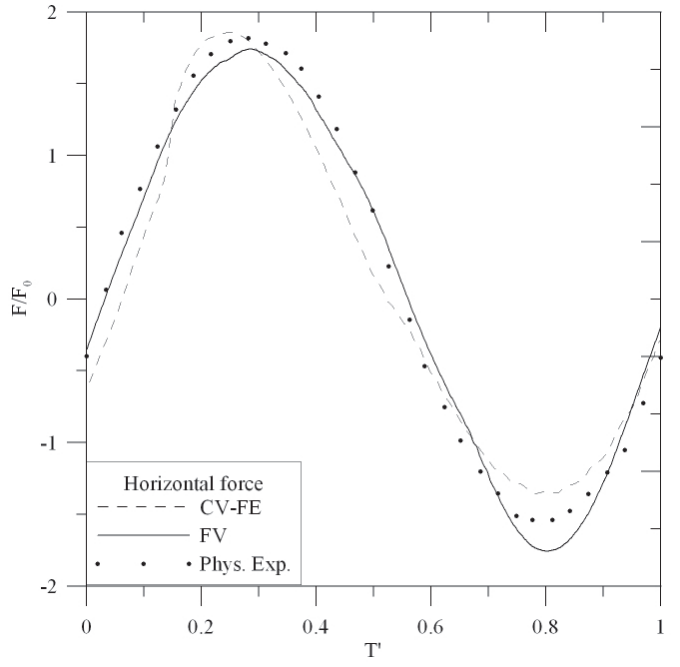
$$\frac{F}{F_0} = F\frac{kh}{\rho g a H h \tanh kd}, \tag{5.1}$$

where F is the measured or extracted horizontal force on the cylinder, F_0 is the force from diffraction theory, g is the acceleration due to gravity, ρ the water density, a the radius of the cylinder, H the waveheight, k the wavenumber and h the still water level (Sarpkaya & Issacson 1981).

Figure 5.4 shows the comparison between the experimental data and the two numerical simulations for both wave regimes investigated. Both numerical methods show good agreement for both cases, slightly better for the second one with kA = 0.34 and N_{KC} = 0.54 (Figure 5.4b). Small differences in the results are expected because the physical data is averaged over 10 wave periods, whereas the numerical results represent one wave cycle only. For the second case (Figure 5.4b), for which N_{KC} is larger, the drag forces are more important than for the first wave environment and as a result the physical experiments and the simulations reach a maximum value

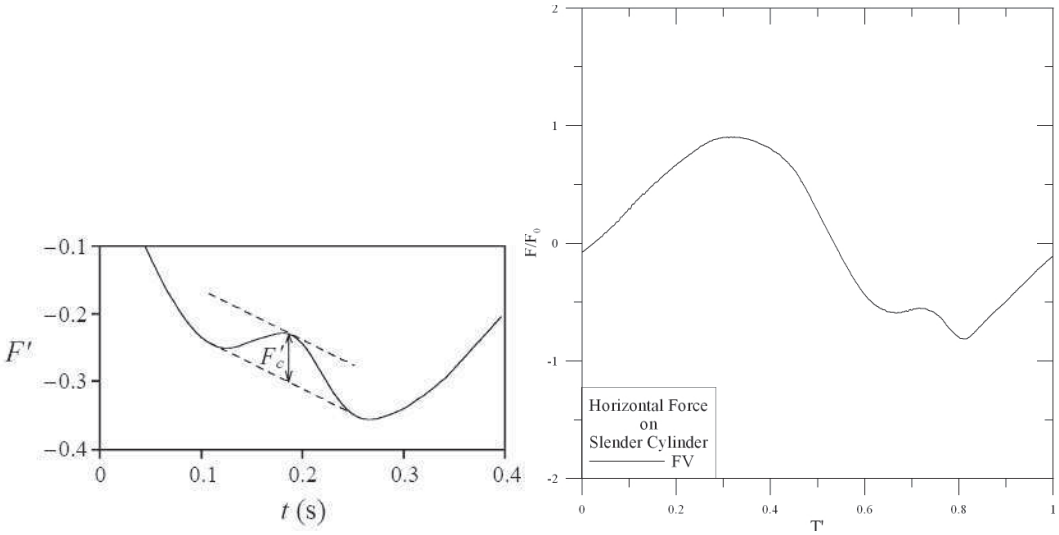


(a) $kA = 0.09$, $N_{KC} = 0.32$



(b) $kA = 0.34$, $N_{KC} = 0.54$

Figure 5.4: Horizontal forces on large vertical cylinder for both wave environments



(a) Presented by Chaplin et al. (1997) *(b)* Simulated by using the FV solver

Figure 5.5: Secondary load cycle

almost twice as much as predicted by linear diffraction theory. This shows the limits of linear methods in certain wave regimes.

5.1.3 Slender vertical cylinder and ringing of cylinder

A highly non-linear phenomenon is the so-called "ringing" of a vertical cylinder due to interactions with steep waves. This effect was discovered during the design of a large Norwegian oil rig in the 1990s (Stansberg 1997). Such tension leg or gravity based platforms, constructed from vertical cylinders, may be affected by waves with a period approximately 3-5 times larger than the resonance period of the structure. Usually resonance occurs at 3-5 s. Grue et al. (1994) describe physical tank tests of a vertical cylinder that is passed by transient wave groups of varying amplitudes. For the wave groups with large wave height at the cylinder they found secondary oscillations in the horizontal force measurements similar to the ones shown in Fig-

ure 5.5. The oscillations start when the wave is about one cylinder diameter behind the cylinder and last for 15 % of the wave period. The magnitude of the peak of the oscillation to the peak of the total force was found to be 11 %. Grue et al. also identified a low-pressure zone behind the cylinder, which they link to a suction force. Besides describing the problem in terms of the non-dimensional Keulegan-Carpenter and Reynolds numbers, Grue et al. state that the Froude number, given as

$$N_{Fr} = \frac{u}{\sqrt{gD}}, \tag{5.2}$$

with u being the particle velocity underneath the wave crest, g acceleration due to gravity and D the cylinder diameter, is important. As the time when the crest passes the structure is short, the problem is considered as a slowly varying horizontal current. Then u can be estimated as $u = H\omega$, where H is the local wave height and ω the local wave frequency. For N_{Fr} = 0.4 the effect of the secondary oscillations becomes pronounced.

The ringing itself only occurs when the cylinder is mounted elastically, which is not the case for the simulations presented here. However, Chaplin et al. (1997) have carried out tank tests to reproduce this effect. First they studied fixed vertical cylinders in steep focused waves and then cylinders mounted elastically in order to measure the excitation. Thus they were able to identify secondary loading within the measured force curve. Chaplin et al. (1997) present the definition of the secondary load cycle reproduced in Figure 5.5a, where the horizontal force on the cylinder is plotted against time. They state that this effect occurs when the wave height is equal to the cylinder diameter. Using the FV solver the horizontal force predicted

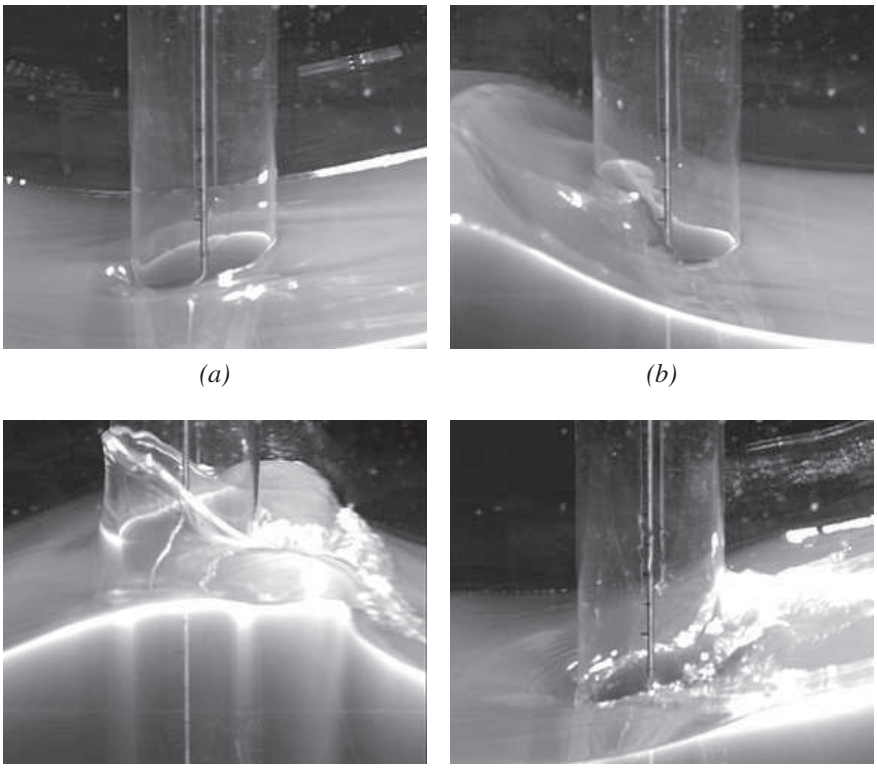


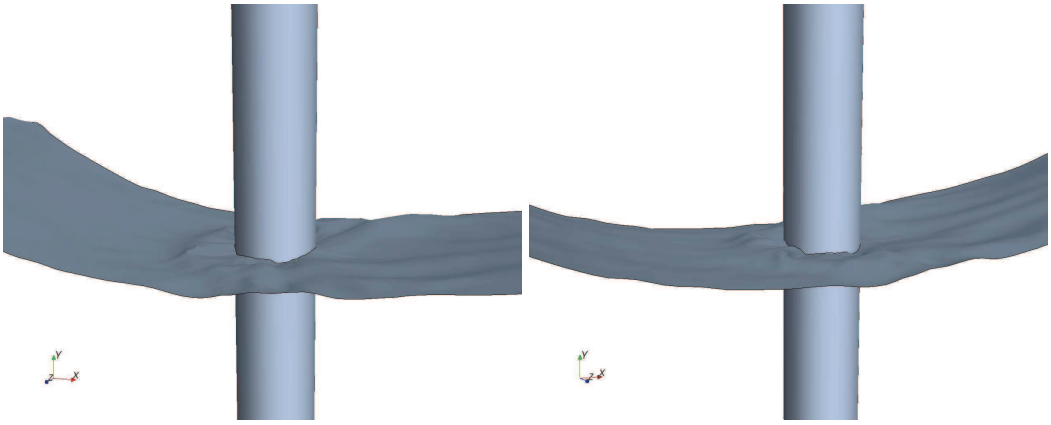
Figure 5.6: Surface elevation around slender cylinder (Rainey & Chaplin 2003)

for a slender cylinder of diameter equal to 0.1625 m in the same wave climate as described above can be seen in Figure 5.5b. The secondary load cycle is present for this wave, although it cannot be compared exactly with Figure 5.5a as the inlet wave condition is different. The secondary load cycle is initiated by the backward breaking wave behind the cylinder. Photographic footage from tank tests published by Rainey & Chaplin (2003) can be seen in Figure 5.6. In the first picture (a) the waterlevel is almost at still water position followed by the approaching wave in the next picture (b). Then the wave passing the cylinder axis is captured (c) and also the backward breaking wave is shown (d). Furthermore, Rainey & Chaplin provide

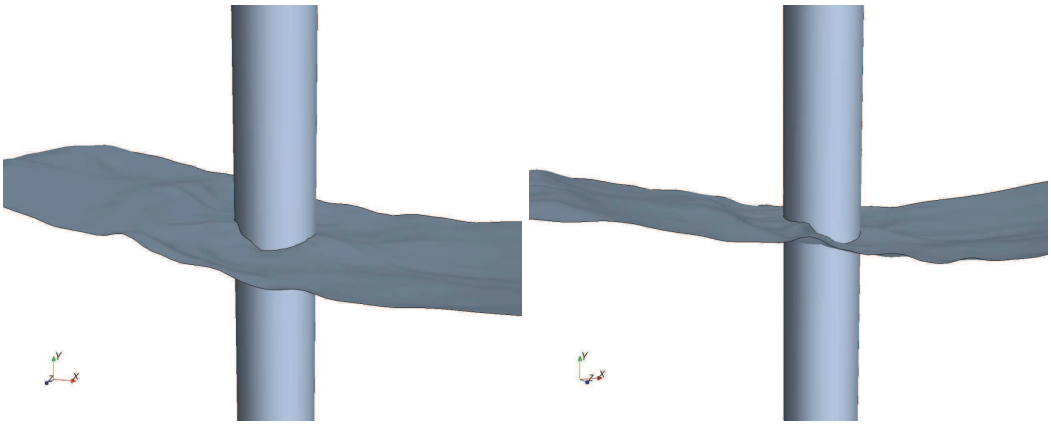
computational results of the free surface obtained from potential flow calculations. The surface is accurately resolved for the wave passing the cylinder. However, the highly non-linear effects of the wave breaking behind the vertical cylinder and also the wave run-up upstream of the structure cannot be captured. Figures 5.7 - 5.10 show the surface elevations calculated by the FV solver at the same instances in time from two perspectives: on the left-hand side the view is towards the negative x-direction (back of the cylinder) and on the right-hand side the cylinder is seen from an upstream position. Here, the wave run-up can be seen at t/T = 0.4-0.6, which is resolved well. Also the breaking wave responsible for the secondary load cycle can be seen at $t/T = 0.8$. Furthermore, highly non-linear processes appear at the cylinder at $t/T = 0.7$, when the run-up water collapses sideways.

5.2 Horizontal Cylinder

In the following section the numerical results for a fixed horizontal cylinder in regular waves and three levels of submergence are presented. They are compared with experimental results from Dixon et al. (1979). In the first simulation the horizontal cylinder is placed at the free surface being half-submerged. Then the structure is positioned deeper in the water with only 25% above still water level. At last the cylinder sits fully submerged but close to the water surface in the tank. The water depth h for all numerical simulations is 1 m, the wave period T is 1.646 s with wave amplitudes A being 0.125 m, 0.05 m and 0.075 m. The diameter of the cylinder D is 0.25 m. Further properties: the displacement of the cylinder from still water level d, kA, kh and the Keulegan-Carpenter numbers N_{KC} for each case are shown in Table

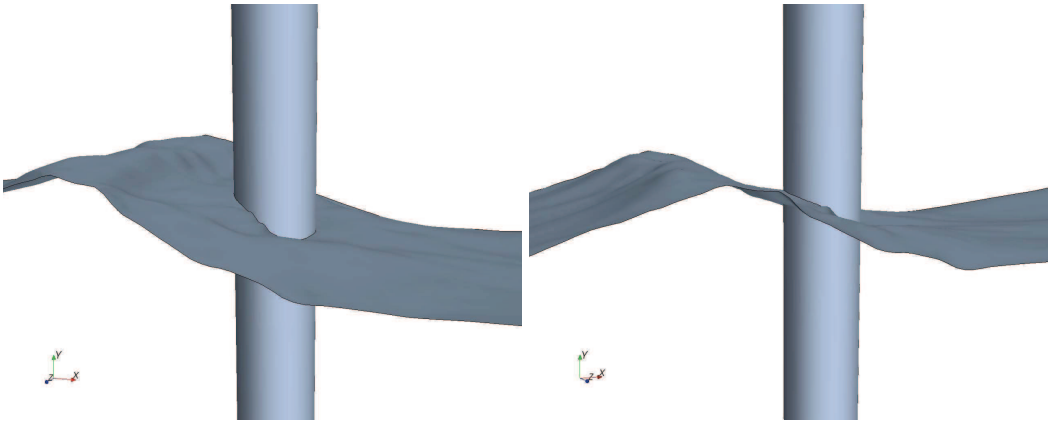


(a) $t/T = 0.0$

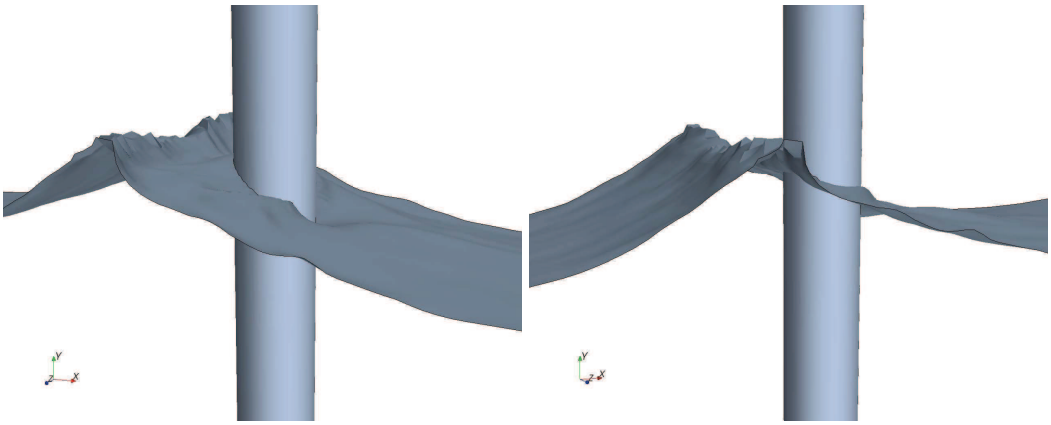


(b) $t/T = 0.2$

Figure 5.7: Surface elevation around slender cylinder

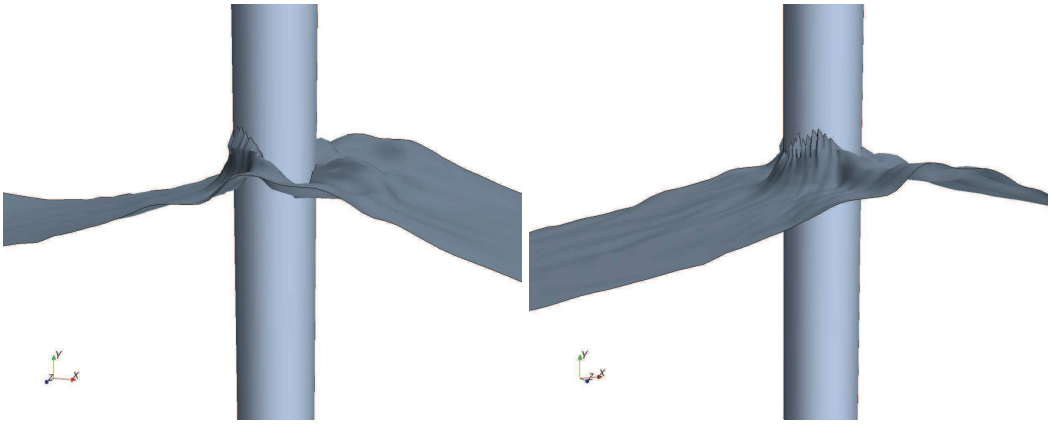


(a) $t/T = 0.3$

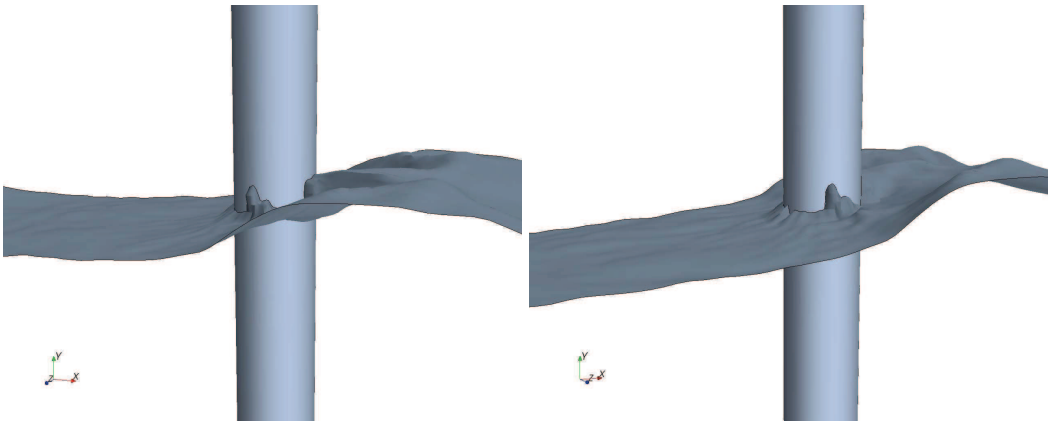


(b) $t/T = 0.4$

Figure 5.8: Surface elevation around slender cylinder

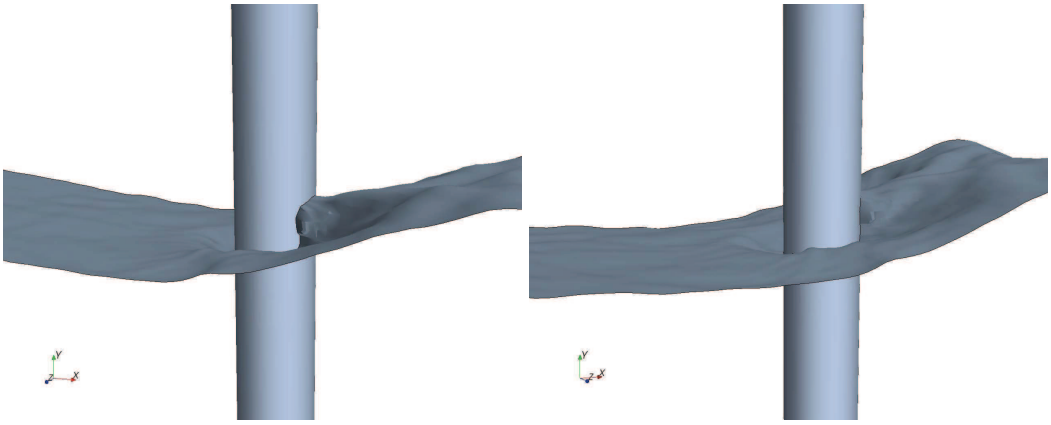


(a) $t/T = 0.6$

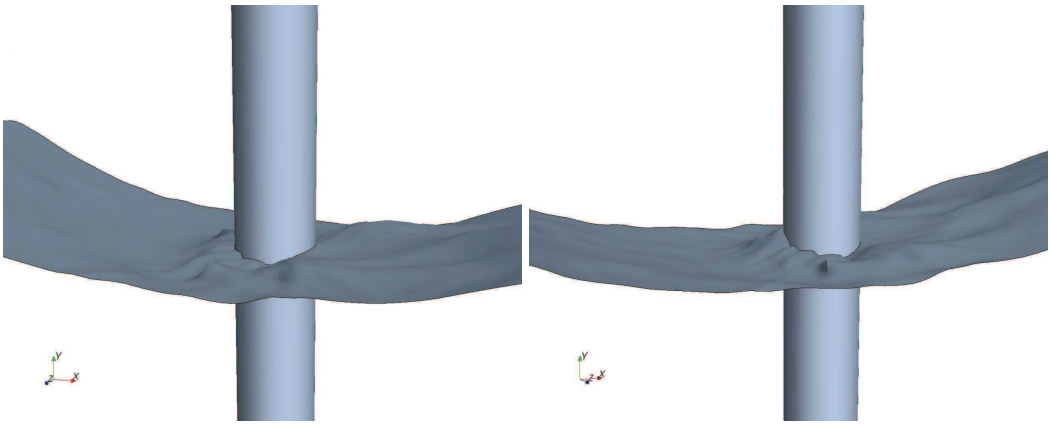


(b) $t/T = 0.7$

Figure 5.9: Surface elevation around slender cylinder



(a) $t/T = 0.8$



(b) $t/T = 1.0$

Figure 5.10: Surface elevation around slender cylinder

Table 5.2: Properties of Horizontal Cylinder Simulations

	Simulation		
	1	2	3
d [m]	0.0	-0.075	-0.15
kA	0.2	0.01	0.12
kh	1.61	1.61	1.61
N_{KC}	3.1	1.3	1.9
A [m]	0.125	0.05	0.075
T [s]	1.65	1.65	1.65

5.2.

5.2.1 Computational Domain and Meshes

For the numerical simulations 3-dimensional meshes containing mostly hexahedral cells are used. The domain as shown in Figure 5.11 is very thin, which makes the simulations essentially 2-dimensional. The same domain is used for both codes. It is 10 m long, 2 m high and has a width of 0.1 m. The cylinder sits one wavelength λ downstream of the inlet and is defined as a wall. In both codes the bottom and far side boundary are also walls. The sides are set up as symmetry boundaries. The waves are generated at the velocity inlet on the left-hand side. The top boundary is a pressure outlet with only air being allowed to leave or enter the domain.

Different meshes for both codes were used. For CFX the meshes contain only hexahedral cells. Due to the use of a blocking structure the cells around the cylinder may be deformed. This slightly influences the initial water surface, i.e. for the numerical experiments where the cylinder is 3/4 and fully submerged. As it takes some time until the wave arrives at the cylinder, this effect has been neglected,

because the water has enough time to settle and adjust itself to a physical behaviour. Figure 5.12a shows the mesh around the cylinder in detail. The meshes contain 79495, 69537 and 69537 cells and a timestep of 0.005 s was used.

The grids used for the segregated iterative solver were generated in the package itself and are mostly hexahedral. To map the geometry exactly the mesher cuts the cells around the structure. Additionally, around the cylinder a prism cell layer was applied to reduce the influence of deformed cells and optimize the computation of boundary flows. The mesh can be seen in Figure 5.12b. These meshes have 113856, 113606 and 114599 cells and are calculated with a timestep of 0.001 s.

5.2.2 Results of horizontal cylinder case

To compare the numerical results with those obtained by Dixon et al. (1979) the vertical forces F_z on the cylinder resulting from drag and pressure on the surface are exported. The forces F_z' shown in all figures are non-dimensionalised using the following expression

$$F_z' = \frac{F_z}{g\rho\left(1/4\,\pi D^2 l\right)} \tag{5.3}$$

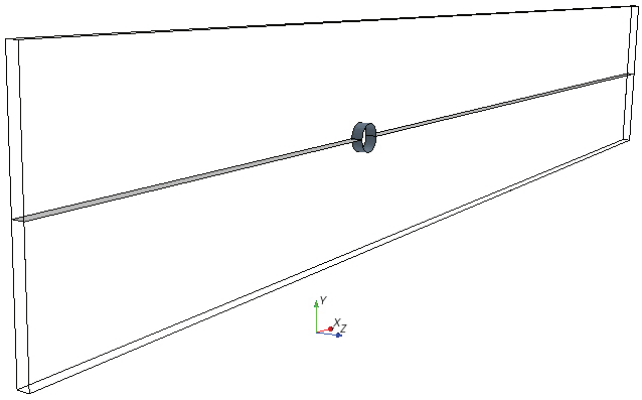
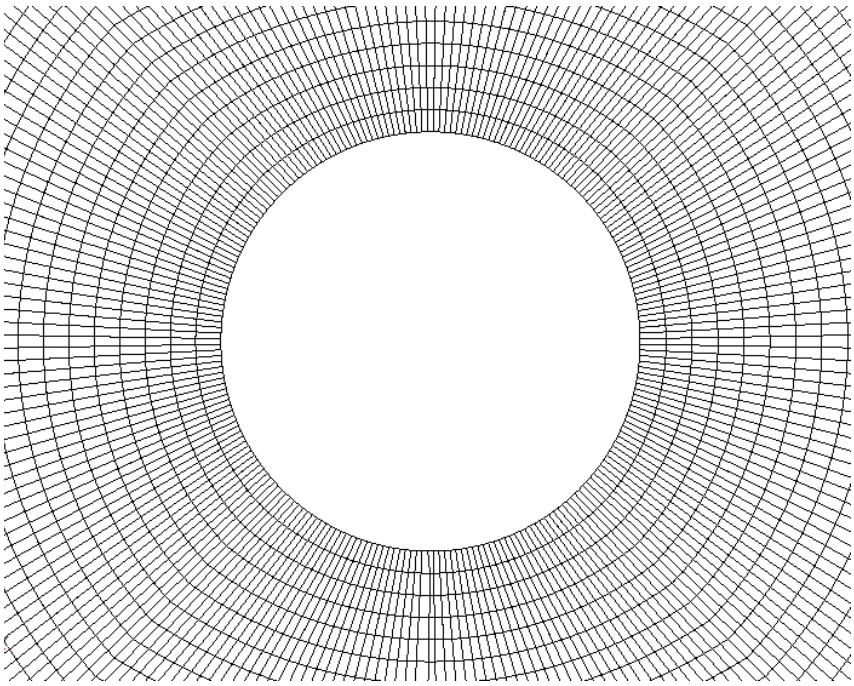
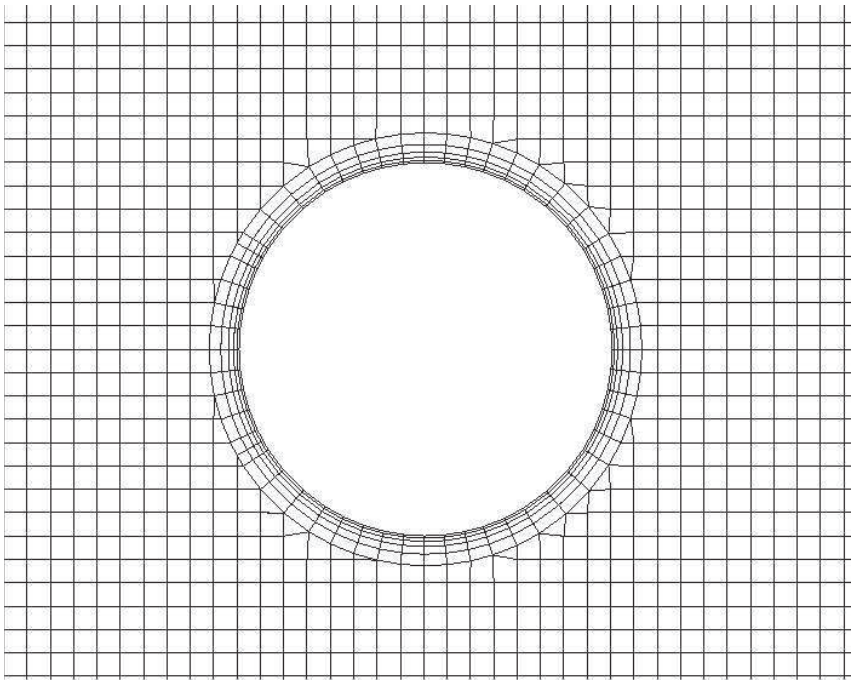


Figure 5.11: Domain of horizontal cylinder calculations



(a) CV-FE (CFX)



(b) FV (STAR CCM+)

Figure 5.12: Horizontal cylinder meshes

with F_z being the measured vertical force on the cylinder, g the acceleration due to gravity, ρ the density of water, D the cylinder diameter and l the length of the cylinder. Figures 5.13-5.15 show the relative vertical force for both numerical simulations compared with the physical experiments over the non-dimensionalised wave period t/T, where t is the time and T the wave period. For the case with a displacement of the cylinder d of 0.0 m the numerical predictions fit well with the measured ones (Figure 5.13). At the beginning of the wave cycle heave forces are dominant. Having passed the peak the forces reduce in a saddle point and become negative after half the relative wave period. When the wave passes further and the water level rises, the downwards force reduces. This can also be seen in Figure 5.16, which shows the surface elevation around the cylinder for 6 moments during the wave cycle.

Figure 5.14 shows the trend of the vertical forces for the horizontal cylinder, which is displaced by 0.075 m downwards from the still water level. Here the qualitative characteristics are the same as for the half-submerged cylinder but the actual values are much smaller. The heave hardly reaches 0.1 and the lift 0.2. The match between numerical and physical experiment is very good.

The last set of numerical results (Figure 5.15), however, does not agree with the physical experiments. For the cylinder displacement d = -0.15 m, which makes the structure fully submerged, the differences for both numerical methods are significant. The values are in the correct region and also the Navier-Stokes calculations coincide with each other, but not with those results received by Dixon et al. (1979).

In two of the three sets of results the CFD methods behave well for the problem

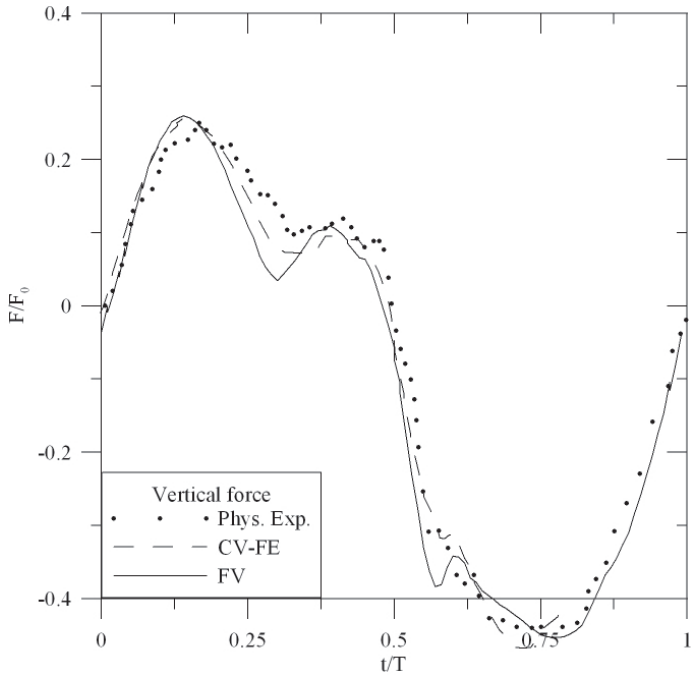


Figure 5.13: Vertical forces on horizontal cylinder (d = 0.0 m)

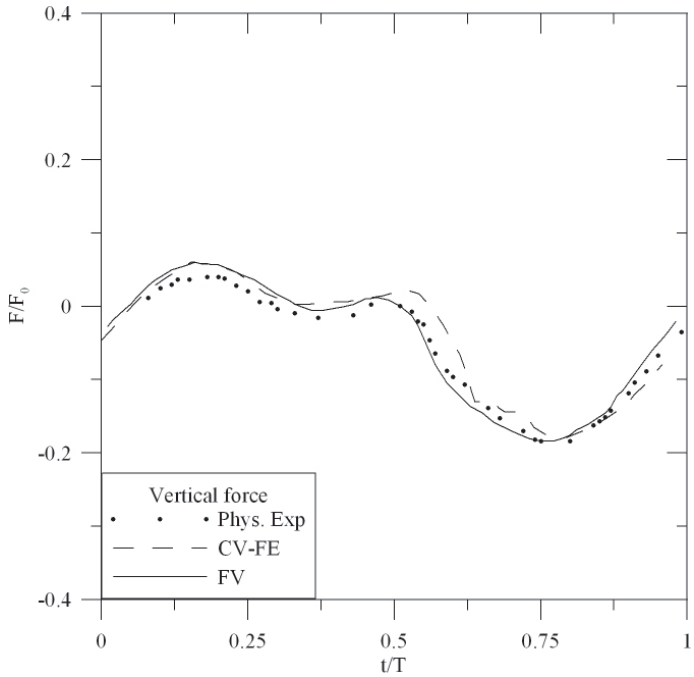


Figure 5.14: Vertical forces on horizontal cylinder (d = -0.075 m)

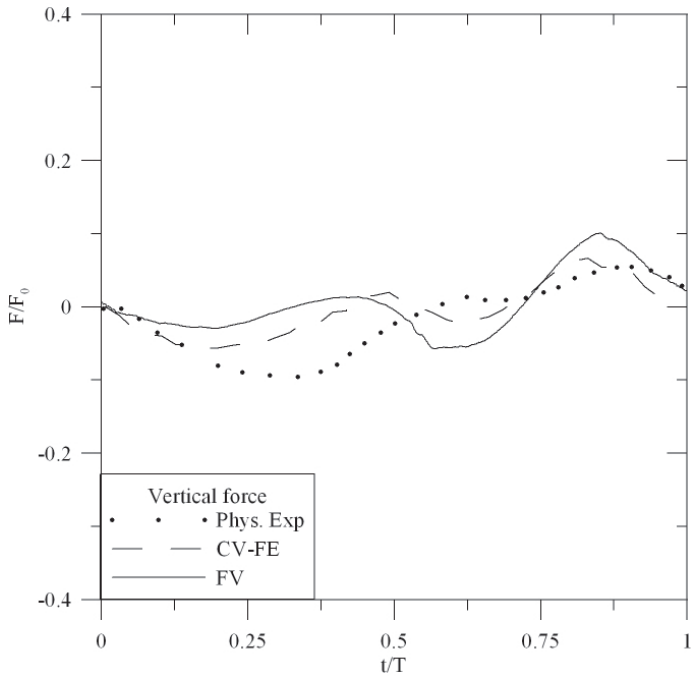


Figure 5.15: Vertical forces on horizontal cylinder (d = -0.15 m)

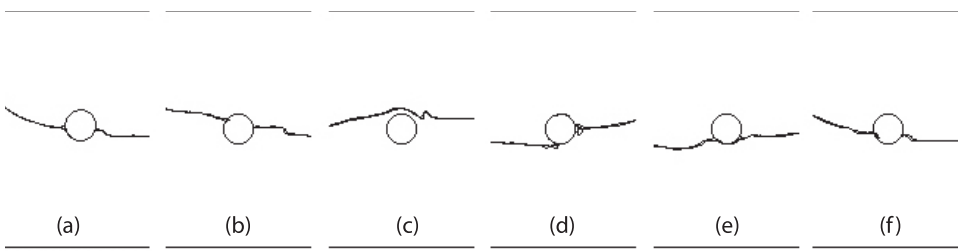


Figure 5.16: Surface elevation around cylinder (d = 0.0 m)
(a) T'= 0.0, (b) T'= 0.12, (c) T'= 0.36, (d) T'= 0.6, (e) T'= 0.73, (f) T'= 1.0.

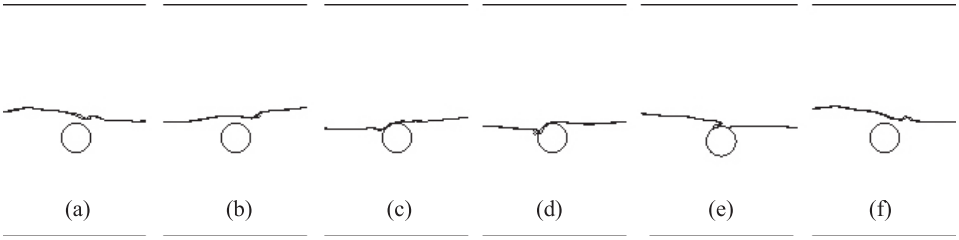


Figure 5.17: Surface elevation around cylinder (d = -0.15 m)
(a) T'= 0.0, (b) T'= 0.24, (c) T'= 0.48, (d) T'= 0.61, (e) T'= 0.85, (f) T'= 1.0.

they were applied to. They were able to calculate the forces on the cylinder within a margin of 4 %. For the case d = -0.15 m however, where the cylinder is fully submerged but close to the still water level, both codes had problems in generating physical results. The difficulty with this setup might result from the larger local non-linearity. The waves appear to be smaller than for d = 0.0 m, which implies less non-linearity for d = -0.15 m. However, the waves are in fact larger than in the case with vertical cylinder displacement d = -0.075 m, which also gave good results. Hence the case of the fully submerged cylinder must involve different non-linear effects. Important for this is the local waterdepth h_{local} above the cylinder as it can be seen in Figure 5.17. When the wave passes the structure the wave amplitude quickly grows to a significant fraction of the local water depth potentially causing it to break. Even when there is no wave breaking, the horizontal fluid velocity might be large ($> gh_{local}^{0.5}$, Longuet-Higgins 1977). Therefore, to solve this problem the local wave height and fluid velocity might force a refined mesh above the cylinder.

Note on 2-dimensional simulations In the development of this work genuine 2-dimensional simulations were also carried out using the FV solver (STAR CCM+). These meshes are the same as used in the previously discussed meshes used for the CV-FE solver (Figure 5.12a) but collapsed in the y-direction reducing the number of

cells and with this the run times by approximately 80 %. As above the vertical forces on the cylinder are extracted and non-dimensionalised to the cylinder diameter using Equation (5.3). Here the cylinder length is assumed to be 1 m. As it can be seen in Figure 5.18 the relative vertical forces do not match with the physical results for any case. Although the geometry appears to be 2-dimensional, the calculation itself is not. The physical effects on the cylinder, i.e. the wave run-up and run-down, seem to be influenced crosswise. The water needs to be replaced by air when the level falls again. In a 2-dimensional simulation this is only possible from the long sides. Hence the fluid exchange is performed more quickly in a 3-dimensional set up and the forces differently calculated.

5.3 Oscillating Cone

The case of a cone shaped rigid body at the free surface is chosen. The cone motion is not influenced by the fluid forces, but driven following the prescribed path of a Gaussian wave packet. This allows the investigation of the fluid effects on the structure without having to deal with a reaction of it, i.e. heave or pitch, and vice versa the investigation of the effects of the structure on the fluid. Drake et al. (2008) and Eatock Taylor, Taylor & Drake (2009) describe the physical tank tests of the oscillating cone. Their results are used for comparing the measured forces and water elevations at the cone with the numerical results achieved by the coupled CV-FE solver (CFX) described earlier in Chapter 3.2.

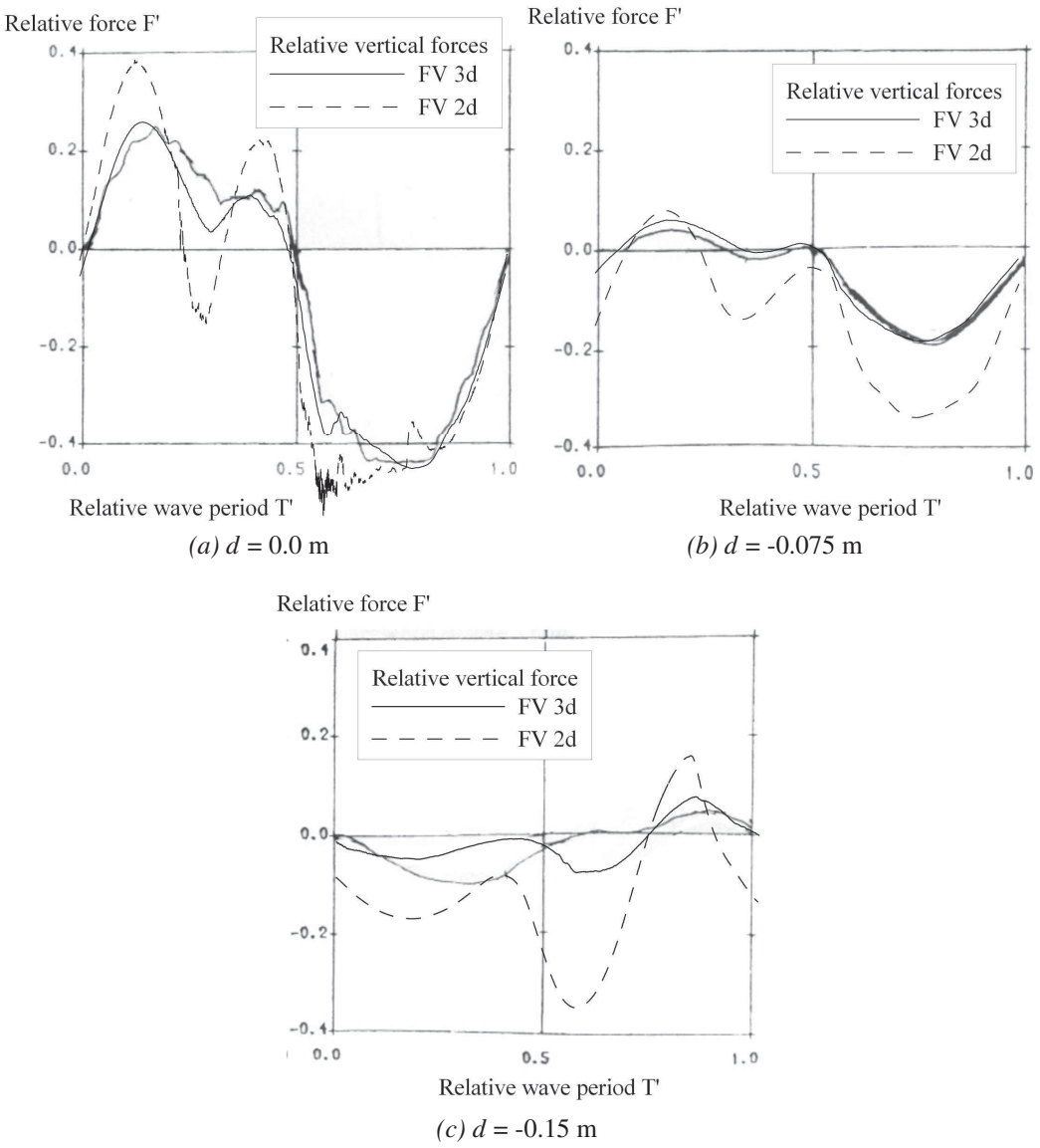


Figure 5.18: Relative vertical force on cylinder, 2d vs 3d

5.3.1 Motion of the cone

The motion of the cone is defined by the displacement $d(t)$ from the initial position at t = 0 s following the form of a Gaussian wave packet, which is described by

$$d(t) = A \sum_{n=1}^{N} Z(\omega_n) \cos\left[\omega_n (t - t_0) - \frac{h\pi}{2}\right] \Delta\omega_n, \tag{5.4}$$

where

$$Z(\omega_n) = \frac{1}{\frac{\omega_n}{2\pi}\sqrt{2\pi}} \exp\left[-\frac{(\omega_n - \omega_0)^2}{2\left(\frac{\omega_0}{2\pi}\right)^2}\right] \tag{5.5}$$

with h = 0 or 1. A denotes the largest excursion from the still water level. N is the number of frequency components and ω_n is the appropriate circular frequency. The central circular frequency ω_0 [rad/s] is defined by

$$\omega_0 = \frac{m\pi}{3} \tag{5.6}$$

with m being an integer between 1 and 12. The results presented in this section are calculated with h = 0, A = $\pm$ 50 mm and m = 3, 7 and 9. For a low m the central frequency of the Gauissian wave packet is low and the cone is moved slower compared to the high frequency case m = 9, where the cone follows the same displacement but in a shorter time. The frequency range, centred at t_0, is divided equally into N = 50 components.

5.3.2 Computational Domain and Meshes

The simulations are performed in a three-dimensional domain with a length and width of 2.5 m and a height of 2.0m. The cone is placed in the centre, as can be seen in Figure 5.19. It has a top diameter of 0.6 m and a deadrise angle of 45°. The

slope itself is 0.3 m high. The initial draught of the cone is 0.15 m and the water depth is 1.0 m. The cone is modelled as a cavity in the mesh. The outer boundaries,

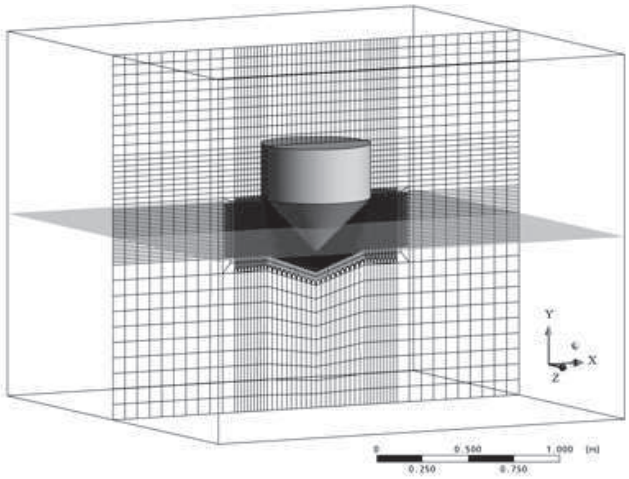


Figure 5.19: Computational domain for oscillating cone case

the bottom and the structure are modelled as free slip walls. The top boundary is defined as a pressure outlet with constant atmospheric pressure. The mesh consists of 820,000 hexahedral cells, where the regions around the water surface and the cone surface are highly refined to achieve cell edges of approximately 0.1 m. This value is based on the results of the regular wave tests in Chapter 4.1, where the wave height needed to be resolved by at least 10 cells. Here, the magnitude of the cone oscillation is 0.1 m. The outer regions are relatively coarse to save computational resources and encourage numerical damping, thus avoiding reflections from the walls. The simulations were carried out using high performance computing on 16 CPUs. The timestep is 0.0005s.

5.3.3 Oscillating Cone Results

Six simulations were carried out in pairs in order to consider the positive direction cone displacement for a maximum excursion of A = +0.05 m and the opposite negative displacement for A = -0.05 m. By analysing the sum and difference of the wave elevation and forces for the paired tests, this allows the non-linearity of the system to be considered. The same technique was applied and described for the focused wave simulations in Chapter 4.2.

The surface elevations for the numerical calculations are extracted at the intersection of the cone surface and the water volume fraction of 0.5, which is generally accepted as representative of the water surface. The time and the surface elevations are non-dimensionalised by dividing by the period of the central frequency and A, respectively. The forces in all plots are non-dimensionalised using the expression

$$F' = \frac{F}{\rho g \pi a^2 A}. \tag{5.7}$$

The measured force F is divided by the acceleration due to gravity g, the density of fresh water ρ, A and a, which is the radius of the cone at the waterline. Figures 5.20 to 5.22 show the vertical forces for cases m = 3, 7 and 9. The numerically predicted fluid forces are in good agreement with the experiments. A small difference can be observed in the crests and troughs in all plots, especially for the extreme values when the maximum or minimum displacement occurs.

Beside the forces the relative water surface elevations are of interest. Figures 5.23 to 5.25 show the plots for all cases. Generally the calculated results agree well with those of the physical experiments. However, a difference can be observed for

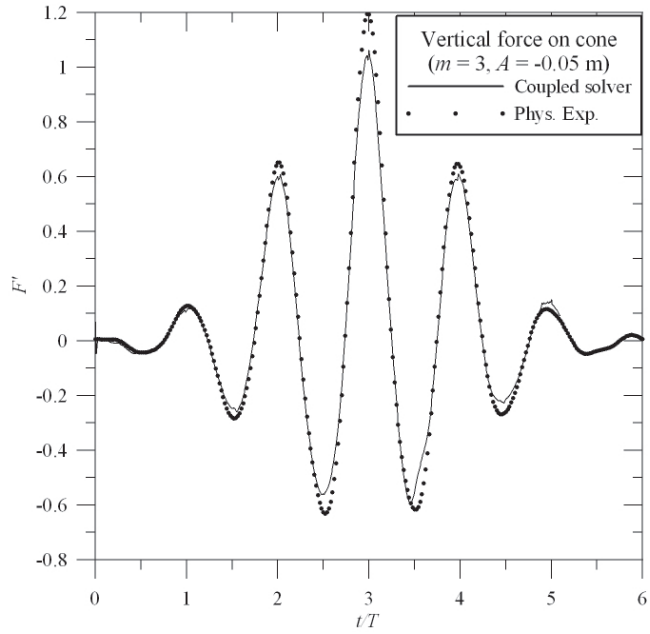
the maximum and minimum elevations, which become worse for the cases with higher central frequency of the Gaussian wave packet.

Although the cone does not react to the fluid forces, the interaction between inertia and drag forces and the driven cone motion appears to be very complex. Different from a typical water entry problem, where for example a wedge enters the water with a constant velocity or acceleration, the cone oscillates at the surface. This generates a phase decay of the maximum flow velocities, i.e. the time when the cone moves quickest does not coincide with the highest fluid velocities. For case m = 9 the maximum downward or upward cone velocity occurs at the turning point of the displacement curve. Due to the inertia of the water the maximum flow velocity however occurs later. The surface elevation shown in Figure 5.26 depends on the relative upward flow close to the cone surface, when the cone travels downwards. This jet-like effect happens to be the reason for the differences in the computational results for the surface elevation, from which the forces are also calculated, by averaging the wetted area of the cone. For the cases with lower central frequency, i.e. m = 3, this jet effect is less developed. Here the cone velocities are slower, which makes it a shear influenced regime, as indicated by the Keulegan-Carpenter number N_{KC} calculated from Equation (2.2).

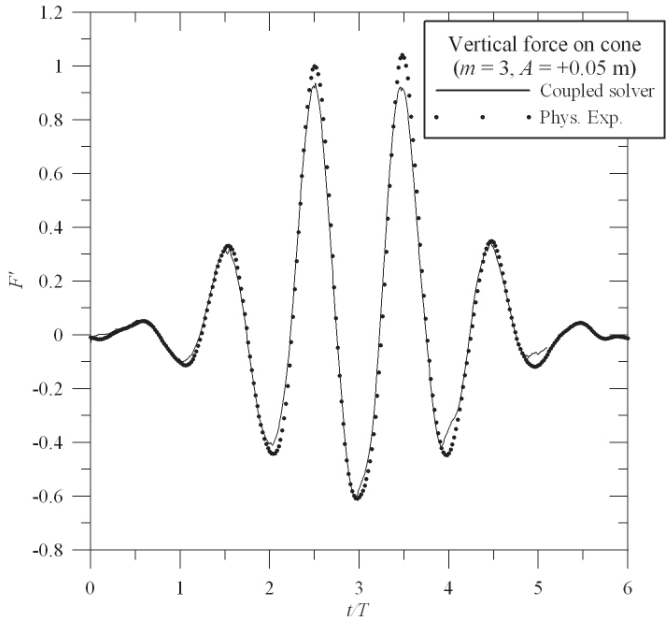
To analyse the non-linearity in the case, the time histories of the relative surface elevations for the paired tests have been subtracted and summed respectively and divided by 2. This enables results to be broken down into linear and higher order components and compared separately. The sum and difference elevations are plotted in Figures 5.27 to 5.29. For the central circular frequency corresponding to m = 9 the relative water surface elevation clearly contains a higher order component represented by the solid line. Unlike the linear part the higher order component is

not symmetric about the mean water line. It oscillates with double the frequency of the linear part around a slightly raised water level.

The total force may be decomposed into its hydrostatic and hydrodynamic parts. The hydrostatic part results from the buoyancy force, which is subtracted from the total force to obtain the hydrodynamic contribution. Figures 5.30 to 5.32 show the non-dimensionalised vertical forces for all cases, decomposed into dynamic and hydrostatic components. For the lower frequency case, $m = 3$, the hydrodynamic force is a much smaller component of the total force. The reason for this is, that due to the higher central circular frequency for larger m, the Keulegan-Carpenter number N_{KC} reduces. N_{KC} describes the relationship between the drag forces over the inertia. For lower N_{KC} the inertia dominates the force contribution. This can be seen in the results. For case $m = 9$, with $N_{KC} = 0.11$, the dynamic force component, which is related to the inertia of the cone is more developed than for case $m = 3$, with $N_{KC} = 0.33$.

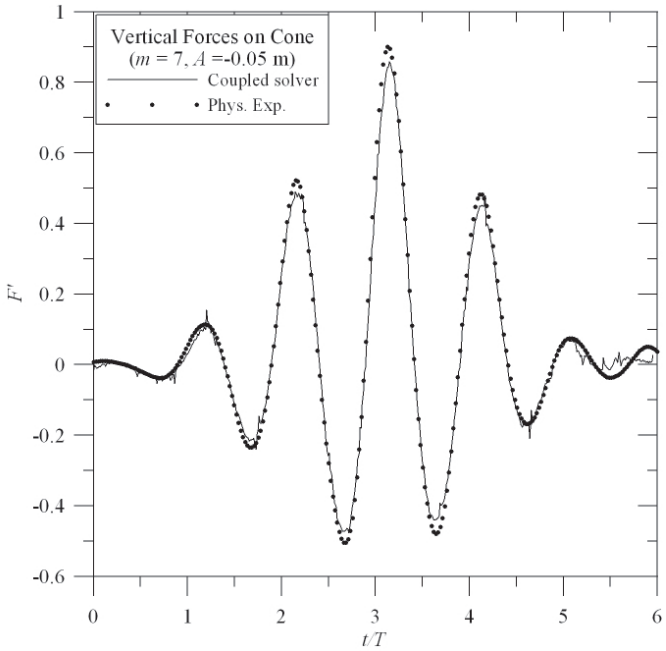


(a) A=-0.05 m; Vertical force on cone

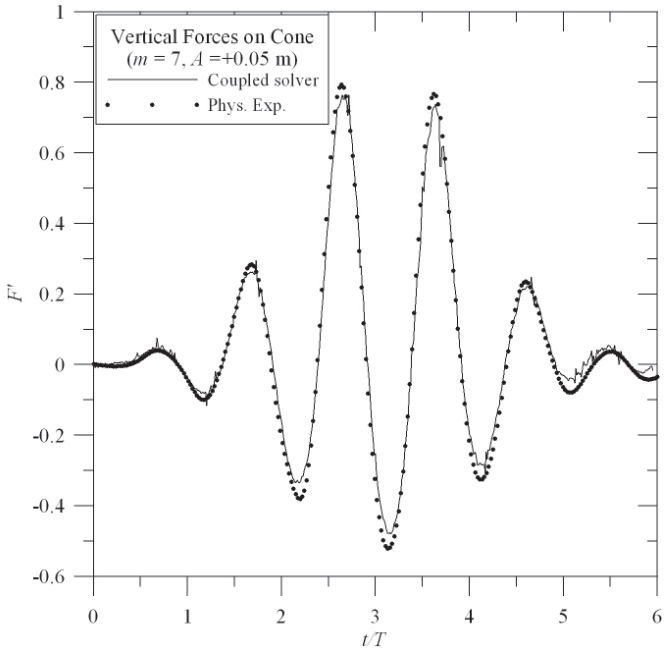


(b) A = +0.05 m; Vertical force on cone

Figure 5.20: Forces on cone for *m*=3

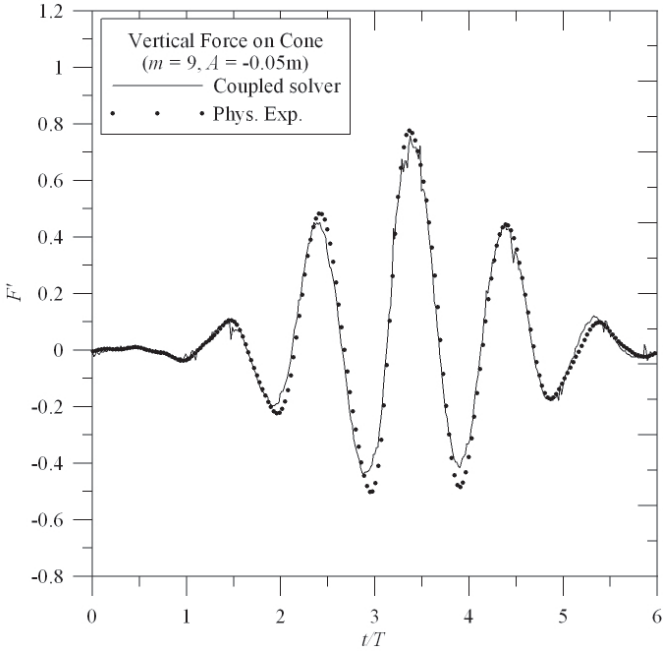


(a) A = -0.05 m; Vertical force on cone

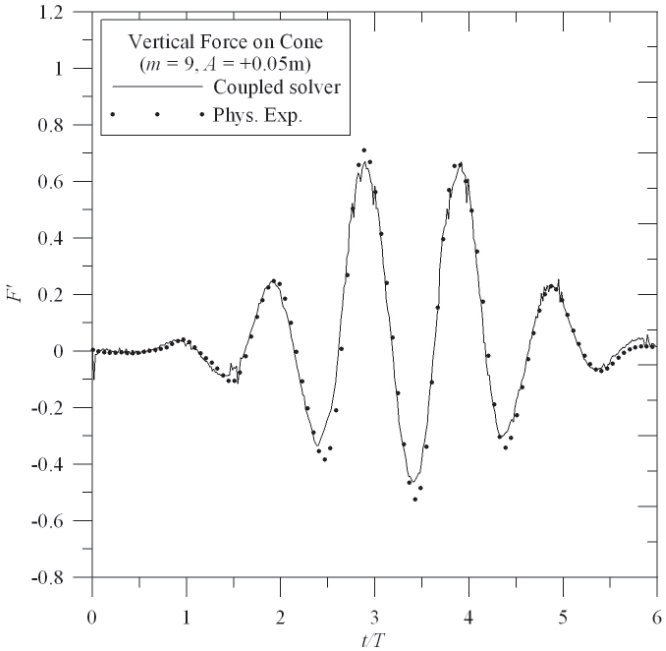


(b) A = +0.05 m; Vertical force on cone

Figure 5.21: Forces on cone for m=7

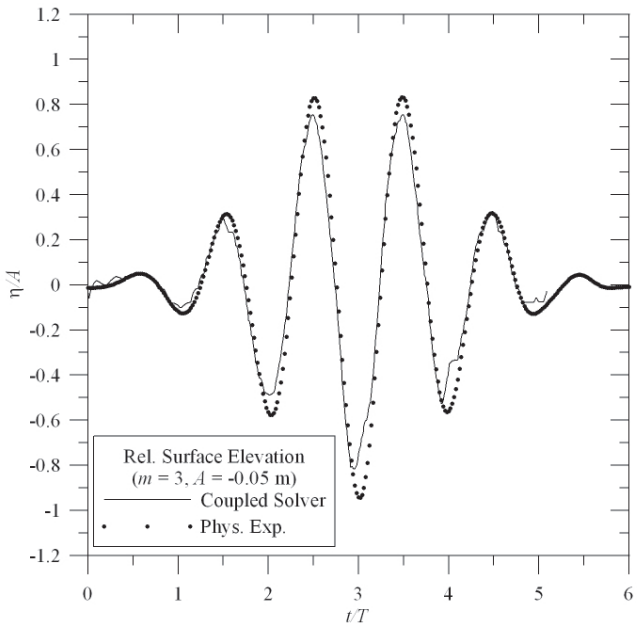


(a) A = -0.05 m; Vertical force on cone

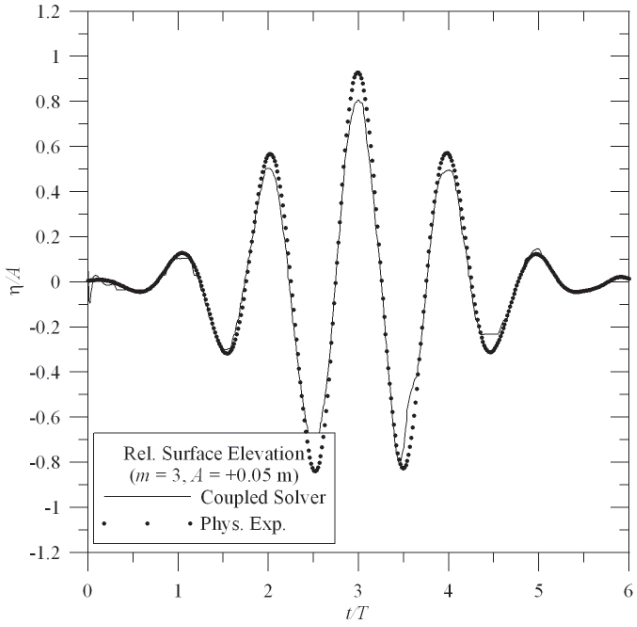


(b) A = +0.05 m; Vertical force on cone

Figure 5.22: Forces on cone for m=9

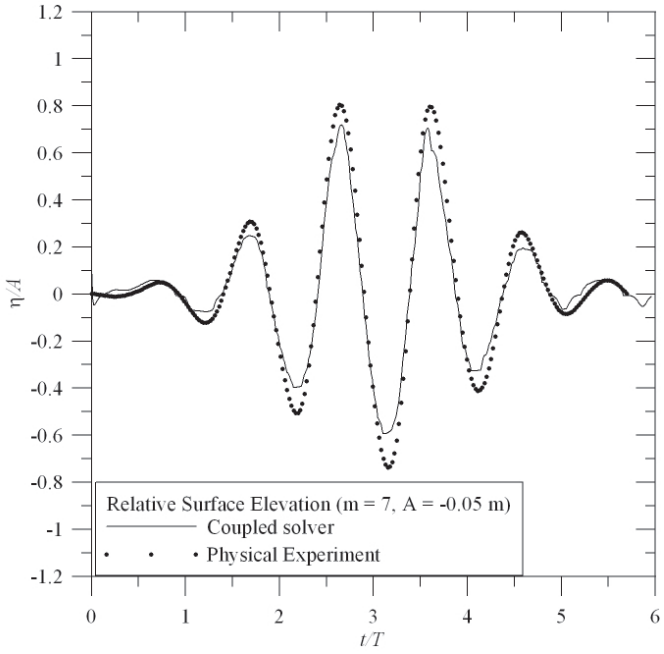


(a) A = -0.05 m; Rel. surface elevation

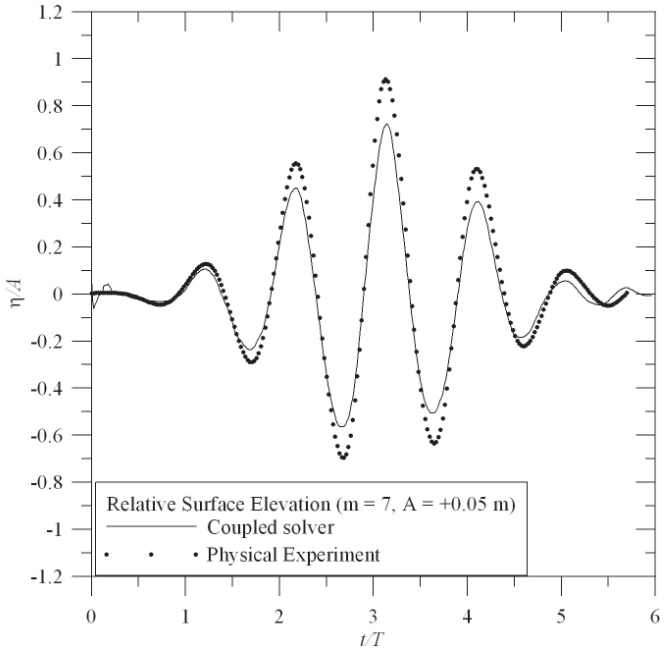


(b) A = +0.05 m; Rel. surface elevation

Figure 5.23: Surface elevation around cone (m=3)

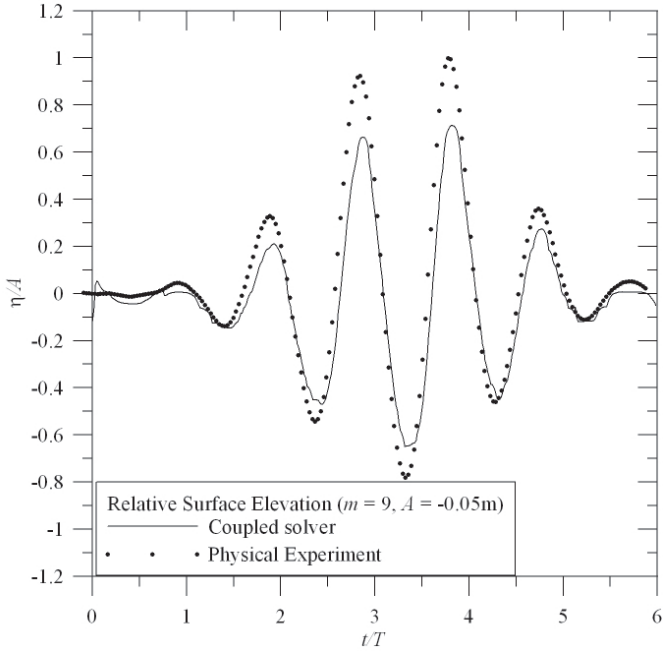


(a) A = -0.05 m; Rel. surface elevation

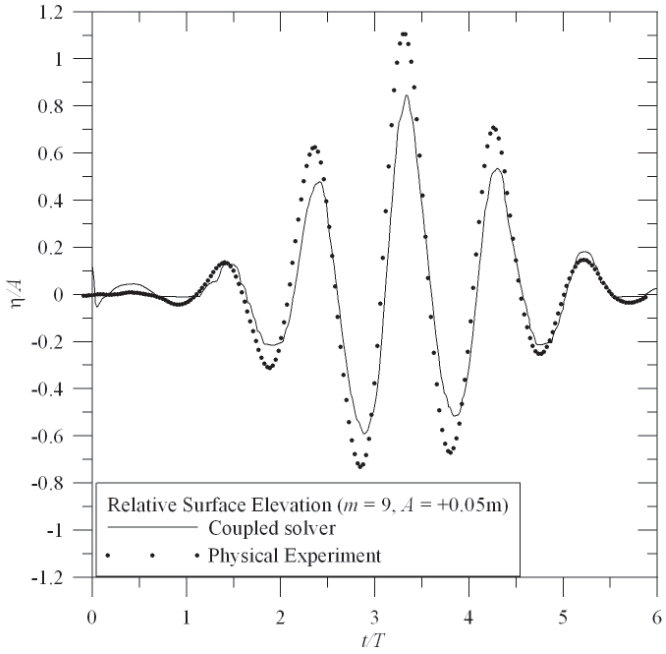


(b) A = +0.05 m; Rel. surface elevation

Figure 5.24: Surface elevation around cone (m=7)



(a) A = -0.05 m; Rel. surface elevation



(b) A = +0.05 m; Rel. surface elevation

Figure 5.25: Surface elevation around cone (m=9)

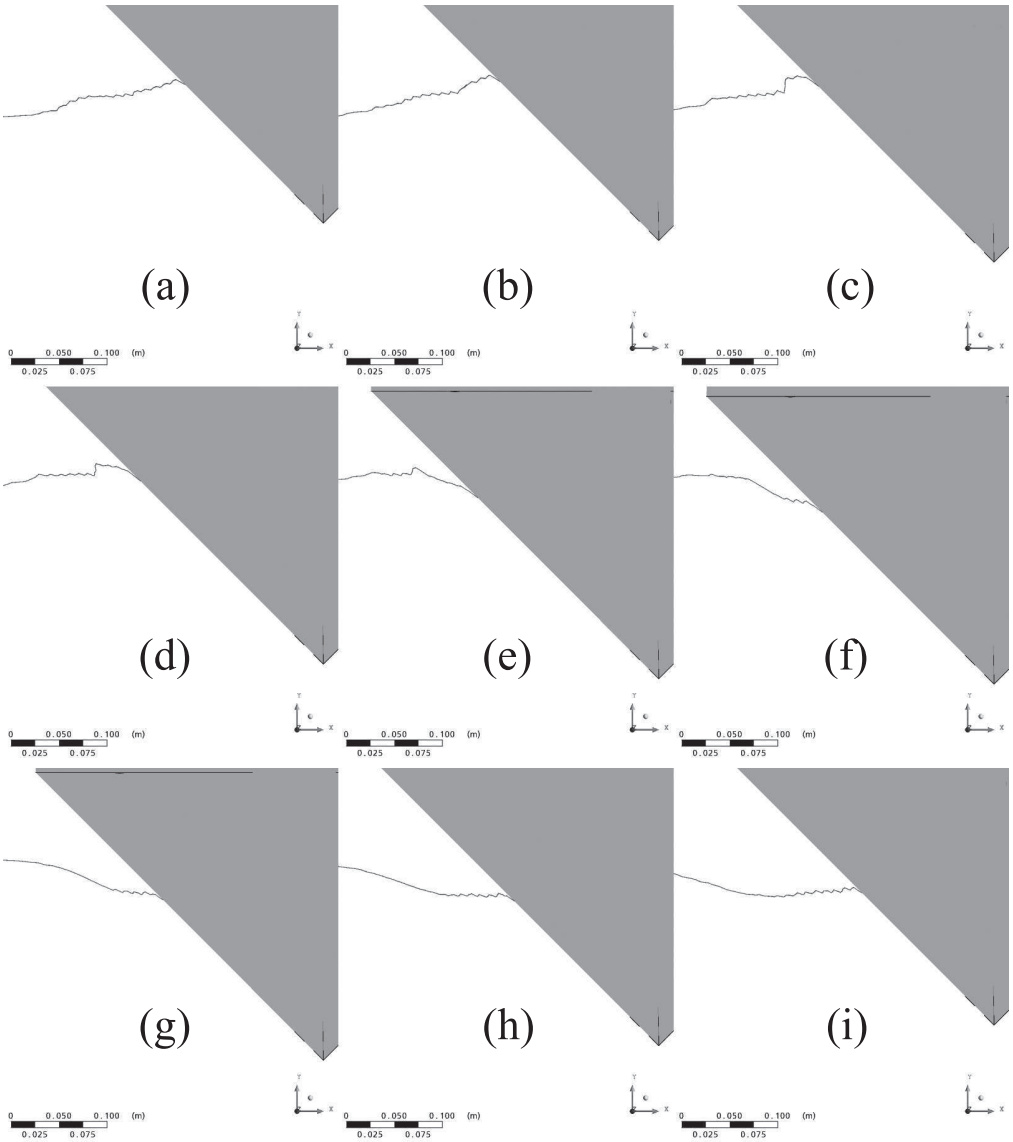


Figure 5.26: Jet formation around cone ($m = 9$)
(a) $t/T = 3.07$, (b) $t/T = 3.15$, (c) $t/T = 3.22$, (d) $t/T = 3.3$, (e) $t/T = 3.37$, (f) $t/T = 3.45$, (g) $t/T = 3.52$, (g) $t/T = 3.6$, (i) $t/T = 3.67$.

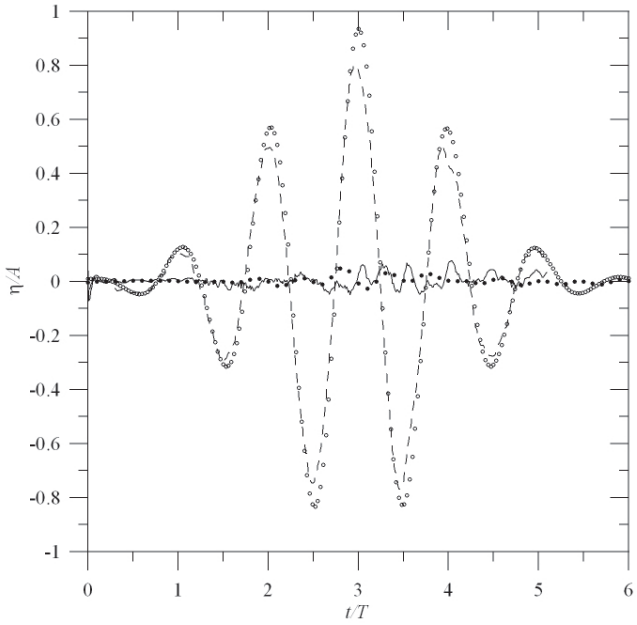


Figure 5.27: A = +/-0.05 m; Sum and difference of surface elevation (m = 3), (○) and (●) represent the difference and sum from the physical experiment. The dashed and solid lines are the appropriate numerical results

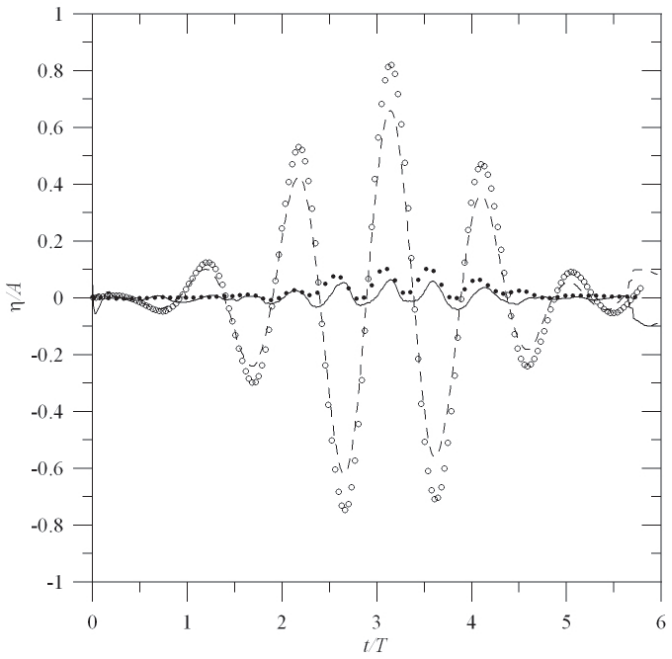


Figure 5.28: A = +/-0.05 m; Sum and difference of surface elevation (m = 7), (○) and (●) represent the difference and sum from the physical experiment. The dashed and solid lines are the appropriate numerical results

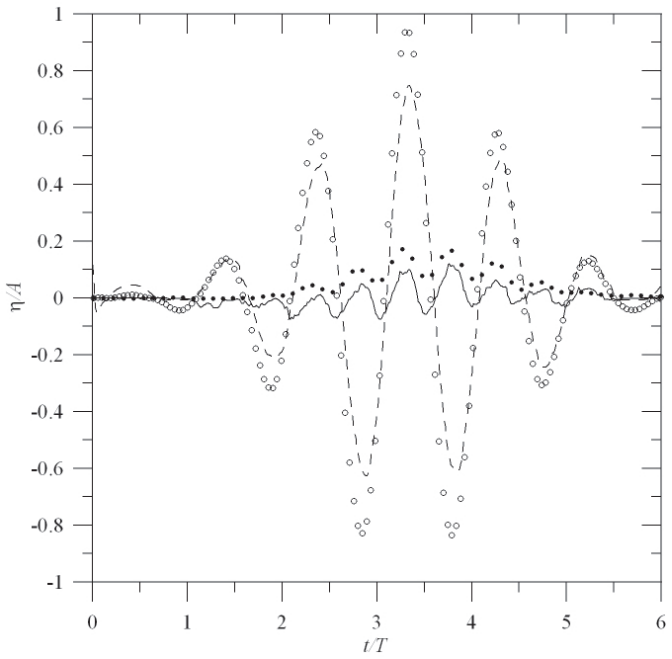
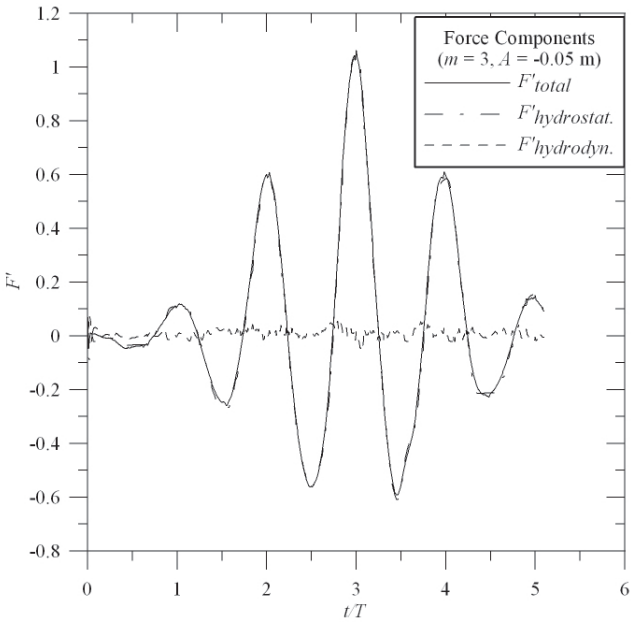
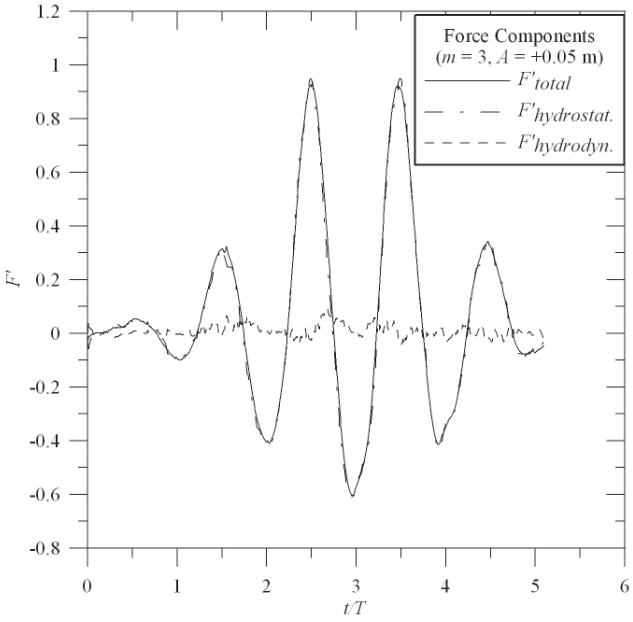


Figure 5.29: A = +/-0.05 m; Sum and difference of surface elevation (m = 9), (○) and (●) represent the difference and sum from the physical experiment. The dashed and solid lines are the appropriate numerical results

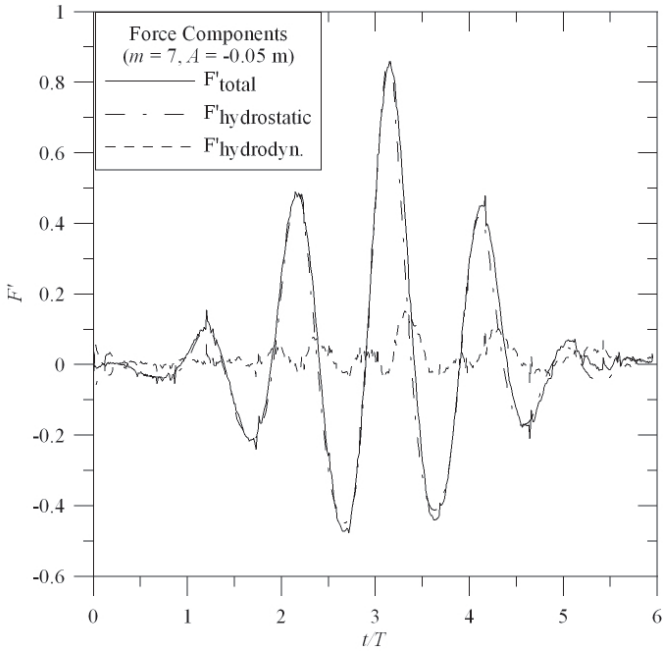


(a) A = -0.05 m; Force components

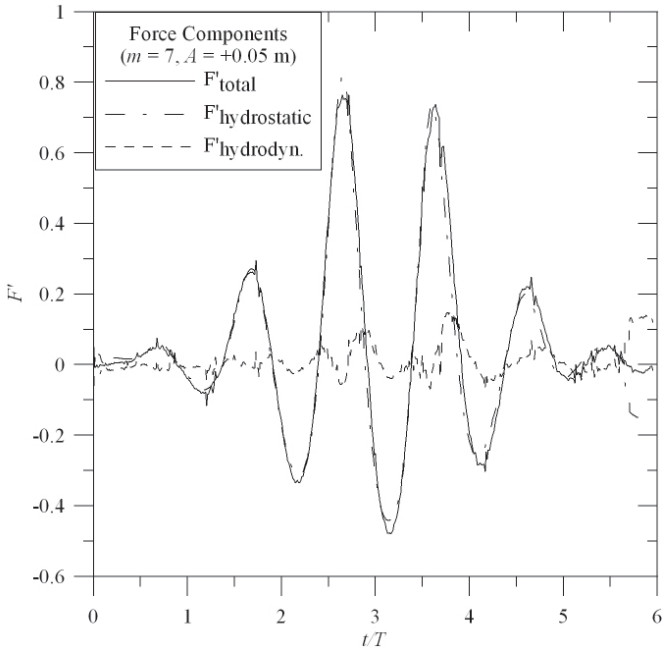


(b) A = +0.05 m; Force components

Figure 5.30: Force components for m=3

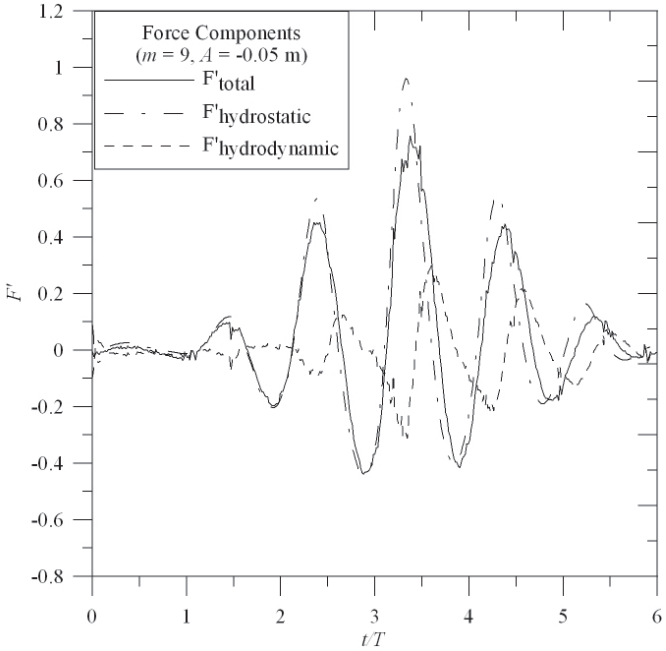


(a) A = -0.05 m; Force components

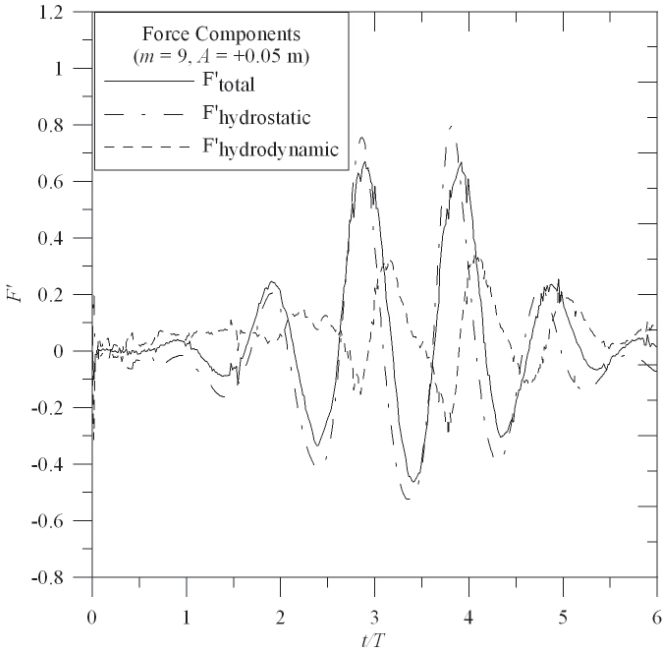


(b) A = +0.05 m; Force components

Figure 5.31: Force components for m=7



(a) A = -0.05 m; Force components



(b) A = +0.05 m; Force components

Figure 5.32: Force components for m=9

Chapter 6

Simulation of WECs

This first part of chapter deals with the final application of the mathematical methods on the testing of the Manchester Bobber. First the idea behind the Manchester Bobber and the different stages of its development are described. This is followed by a section about the physical experiments and finally the numerical simulations are considered.

The second part of this chapter deals with the application of CFD to substitute tank test data for a fixed single section of Pelamis in regular waves. The CFD results could be used to validate linearised numerical models.

6.1 Manchester Bobber

The Manchester Bobber is a point absorber type of wave energy converter. It converts the wave energy by extracting power from the vertical motion of floating cylinders. These cylinders are connected with a counterweight by a rope-pulley system. This runs over a drive shaft, which turns a generator and produces electricity. The device can be placed in water depths of 20 - 60 m and could accommodate 25 - 50 individual floats arranged in an array underneath its superstructure. Figure 6.1 shows an artist's impression of a Prototype device. Currently the Manchester Bobber is under development and single cylinder tests are planned. Up to now scale model tests have been carried out as described in Chapter 2. The tests concern the float design, the interaction of the floats when arranged in an array, the survivability

in extreme focused waves and the amount of damping of the system in still water conditions. For the latter drop and rise tests with different masses for the counterweights were performed (Stallard et al. 2009).

For this work physical tank tests similar to the ones described by Stallard et al. (2008) but with a single float arrangement rather than the full array of floats are simulated using the FV solver. Initially, the float is modelled without the influence of the counterweight by subtracting the mass of the counterweight from the mass of the float. For this case only vertical motion is allowed. Then the counterweight is taken into account by applying an upward force to the float. This time the mass of the float is similar to the mass of the float in the physical experiments with $m_f = 2.2$ kg. The float is allowed to move vertically only, which is equivalent to the tethered setup described later. Finally the simulations allow for vertical and horizontal motion of the float, which will be compared with the untethered setup in the tank tests. Here, an additional horizontal restoring force is applied to the float.

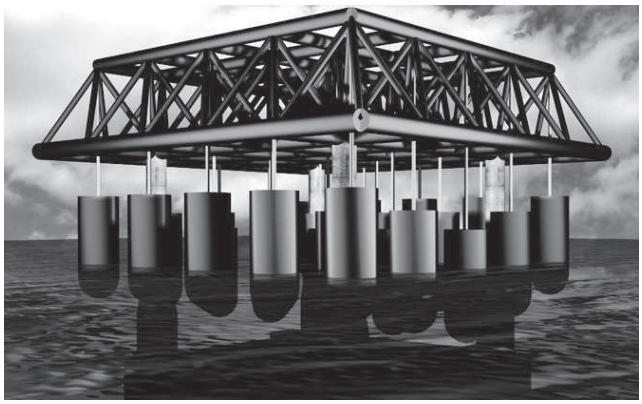


Figure 6.1: Artists impression of The Manchester Bobber (http://www.manchesterbobber.com)

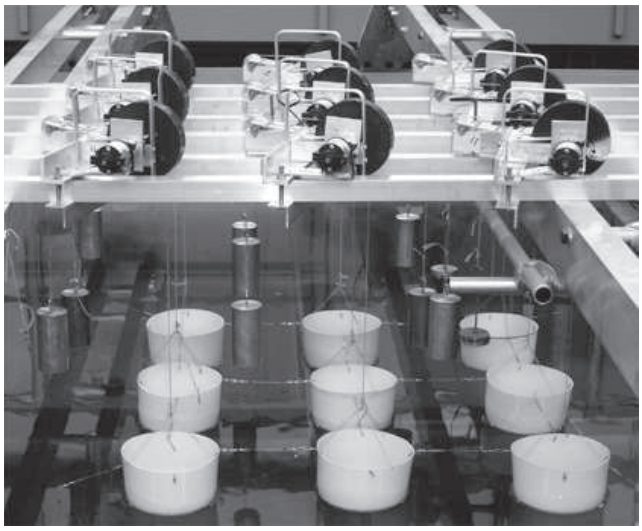


Figure 6.2: Array of Floats in 70th scale (http://www.machesterbobber.com)

6.1.1 Single float in focused waves

Physical Experiments

The physical tank tests were performed in the wavetank of the University of Manchester. It is 18.5 m long, 5 m wide and tests were done with a water depth of 0.5 m. The waves are generated using 8 No. piston type paddles operated using the Edinburgh Designs "OCEAN" interface. To minimise reflections from the far end wall, a curved surface piercing beach is installed. Figure 6.2 shows the test arrangement for an array of nine floats with the counterweights, pulley systems and power-take-off in the superstructure. Here, tests for a single float only are reproduced. The schematic float arrangement can be seen in Figure 6.3, where m_f and m_c are the masses of the float and the counterweight respectively. Figure 6.3a shows the arrangement for the tethered experiments. The horizontal displacement of the float is restricted due to the vertical cables. These are attached to the superstructure and held in tension by weights at their ends. In Figure 6.3b the untethered float can be

seen, which is free to move horizontally. The restoring force is equal to the horizontal components of the tension in the pulley cable that connects to the counterweight. In the physical experiments the displacements are measured by converting the angular displacement of the pulley ω_p (r = 17.5 mm). For the tethered case this is taken to be equal to the vertical displacement. As the untethered experiment involves considerable horizontal displacements, the angular displacement is effected by both directions of motion. During all tests no power was taken off the system and the friction in the pulley system is neglected. The cables are assumed to be stiff and constant in length.

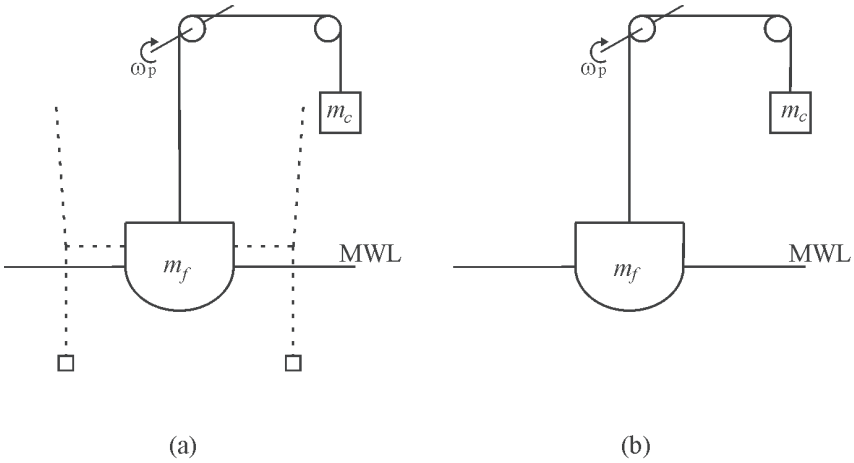


Figure 6.3: Geometry of a single float with counterweight and drive train

For the simulation of the mechanical system in CFD it is necessary to know the interrelation between the two accelerated bodies, i.e. the float and the counterweight. The reason for this is that the CFD code cannot model the pulley system and the counterweight directly, but they have to be approximated using additional body forces. Therefore the free body diagram as seen in Figure 6.4 for the vertical motion is used to find the unknown tension forces in the cable T_1 and T_2 and the

acceleration of the system $\ddot{z}$. For the left system representing the float, the force

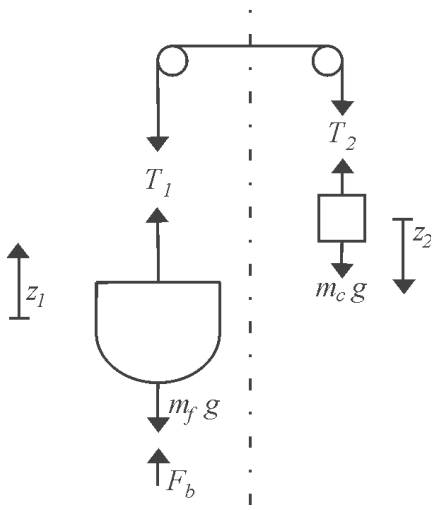


Figure 6.4: Free body diagramm of single float with counterweight

equilibrium is achieved when

$$m_f \ddot{z}_1 = -m_f g + T_1 + F_b, \tag{6.1}$$

with m_f being the mass of the float, $\ddot{z}_1$ the positive upward acceleration of the system, g is gravity, T_1 represents the tension force in the cable and F_b is the buoyancy force. For the travelled system the positive z-direction is downward, denoted with z_2. Then the force equilibrium can be written as

$$m_c \ddot{z}_2 = m_c g - T_2, \tag{6.2}$$

where m_c is the mass of the counterweight and T_2 the tension force in the cable. When moving the float vertically by a distance of z_1 the counterweight covers the

same distance, from which follows

$$z_1 = z_2 \qquad \Rightarrow \qquad \dot{z}_1 = \dot{z}_2 \qquad \Rightarrow \qquad \ddot{z}_1 = \ddot{z}_2, \tag{6.3}$$

where single dots stand for velocities and double dots for acceleration. Furthermore, the relation between the tension forces can be written as

$$T_1 = T_2. \tag{6.4}$$

The two unknowns of the system, such as $T = T_1 = T_2$ and $\ddot{z} = \ddot{z}_1 = \ddot{z}_2$ can then be rewritten as

$$\ddot{z} = \frac{\left(m_c - m_f\right) g + F_b}{m_f + m_c} \tag{6.5}$$

and

$$T = -m_c \frac{\left(m_c - m_f\right) g + F_b}{m_f + m_c} + m_c g. \tag{6.6}$$

In the computational approach F_b is calculated from the integrated pressures on the float surface and thereby known at any time.

6.1.2 Reproduction of wave signal from physical tank tests

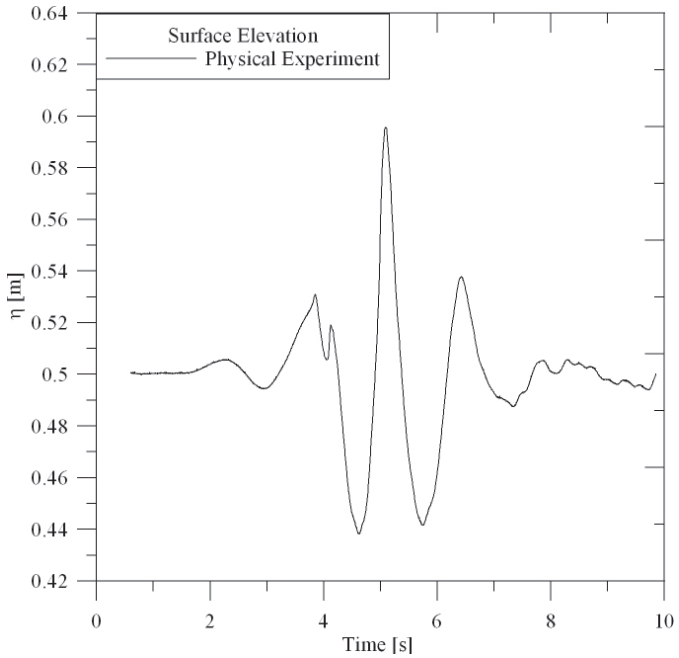
For the physical experiments a focused wave group was defined using the focus(*wv*, *t*, *x*, *y*) function of the Edinburgh Designs "OCEAN" software, which is used to control the wave paddle in the tank. To reproduce the surface elevation of the physical tank tests the measured surface elevation time history at the location of the float without the float being in place is considered. The time history shown in Figure 6.5a is decomposed into the frequency domain. Therefore a Fast Fourier Transformation (FFT) is performed, which gives the spectral density for each wave frequency in

Table 6.1: Input frequencies for generation of focused wave group

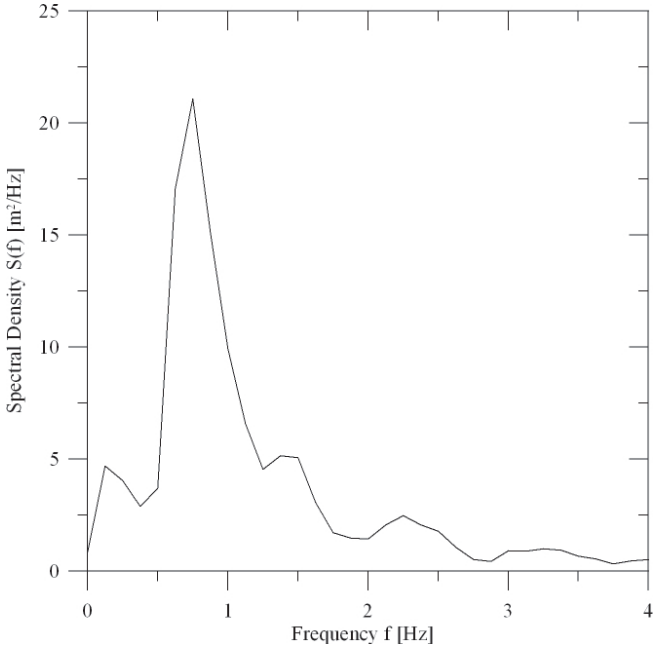
f [Hz]	S(f)[m^2/Hz]
0.125	4.68284732
0.25	4.03990015
0.375	2.87990199
0.5	3.69113908
0.625	17.0628873
0.75	21.0820567
0.875	15.1172159
1	9.93011908
1.125	6.58260372
1.25	4.54555462
1.375	5.14284693
1.5	5.06245887
1.625	3.06069072
1.75	1.70469081
1.875	1.47812389

the time history. The spectrum can be seen in Figure 6.5b. For the wave signal one has to limit the frequency range by cutting the spectrum at the ends. For these tests the frequencies go from 0.125 Hz to 1.875 Hz. The range is evenly split into 15 frequency components from which the waves are generated (see Table 6.1). From the measured maximum surface elevation the input amplitude A for the numerical simulations is identified and set to 0.11 m. The focus point is at the position of the float at x = 3.5 m behind the inlet and the focus time is 4.6 s. The waves are generated in the same way as described in Chapter 4.2.

To confirm the correctly derived wave settings, numerical simulations of the wave group using first and second order wave signals are carried out. For these simulations the domain has the same length and height as described for the full



(a) Experimental Surface Elevation Time History



(b) Spectrum from Surface Elevation

Figure 6.5: Input spectra for generation of focused wave group

system with the float included in the next section. But because the float is not modelled for this wave-only test the width is reduced to 0.1 m to save computational resources. The mesh resolution is the same as for the full 3 dimensional simulations that include the float. Figure 6.6 shows the calculated time histories at the focus location compared with the measured surface elevation without the float being in place.

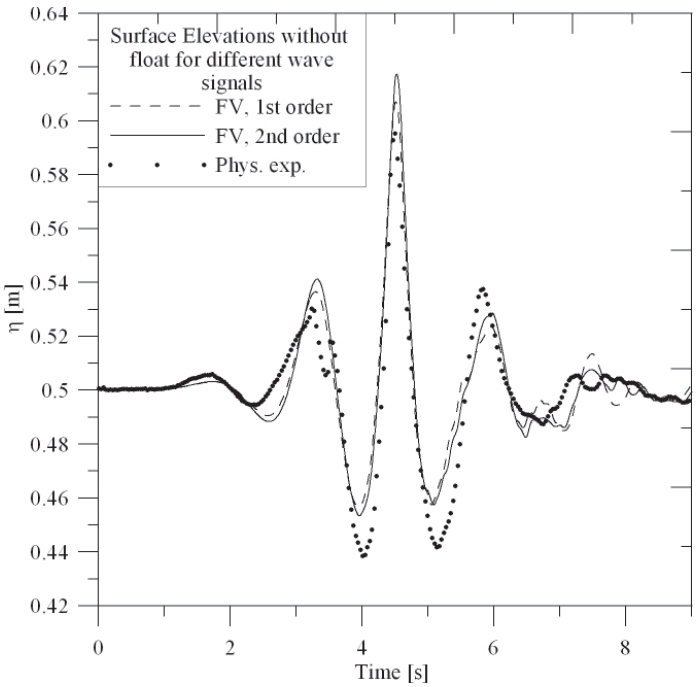


Figure 6.6: Calculated surface elevations at focus point without float and 1st and 2nd order wave signal (FV)

6.1.3 Computational Domain and Meshes

The geometry of the float is similar to the one described by Stallard et al. (2008). The radius of the cylinder section is 74 mm. The top end is flat and the bottom end is a half-ellipse with a horizontal axis of 74 mm and vertical axis of 60 mm as

seen in Figure 6.3. The float is modelled as a cavity in the mesh. Also shown in the picture is the pulley system with the counterweight and power take off. This assembly is not modelled in the numerical simulations.

The domain is 10 m long, 0.6 m wide and 1 m high and can be seen in Figure 6.7. The waterlevel is 0.5 m and the centre of the float sits at x = 3.5 m away from the inlet. The top is a pressure outlet, with only air being allowed to enter or leave the domain. The remaining boundaries are velocity inlets, where, apart from the far left one, the velocities are set to zero. The far left inlet acts as the wavemaker similar to the previous cases described. The mesh used for the FV solver consists of 530,000 cells. Figure 6.8 shows the refined mesh around the float. Also the area where the water surface is expected is refined. Here the cell size is about 0.5 cm. The rest of the mesh is coarse to save computational resources. As for the cylinder cases the mesh is mainly built from hexahedral cells. The float is discretised by 2153 faces, of which 13 are polygonal, 1775 quadrilateral and 365 triangular.

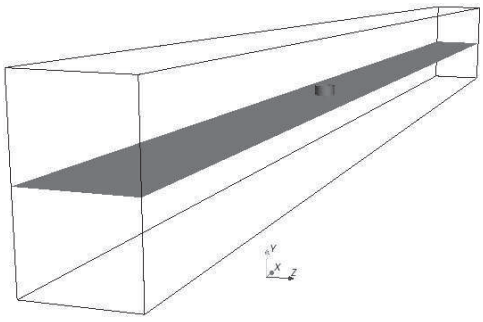


Figure 6.7: Domain for single float simulations

As the counterweight and the tethers cannot be modelled in the CFD approach, these have to be described in terms of additional body forces and restricted degrees of freedom (DOF). The computational setup for the six cases considered can be

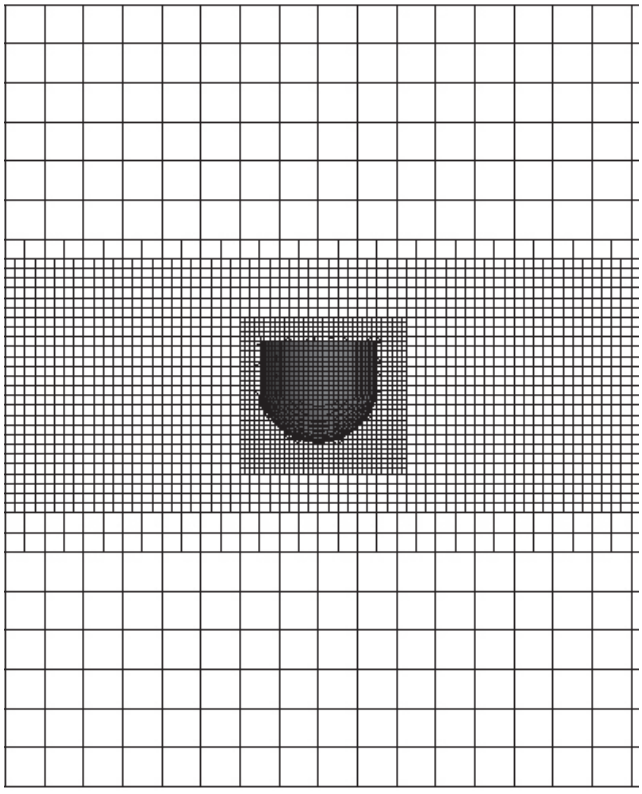


Figure 6.8: Mesh around float

seen in Figure 6.9 and Table 6.2. The first two cases include one degree of freedom, whereas in the third the float is allowed to move horizontally, too. For initial interest the float was modelled without the influence of the counterweight. Therefore the float mass in the computational setup is reduced to 0.5 kg, which is equal to the difference of the experimental float mass m_f = 2.2 kg and the experimental counterweight mass m_c of 1.7 kg (Case A). For case B the counterweight is modelled by including a constant upward body force T as

$$T = m_c \cdot g, \tag{6.7}$$

where g is gravity. Furthermore, the mass of the float in the CFD simulation is set to 2.2 kg. For cases B only the gravitational term is used, as this part is the larger one. These two cases are compared with the tethered arrangement shown in Figure 6.3a. When including the horizontal motion in case C (see Figure 6.9c), an additional restoring force is needed, which is equal to the horizontal component of the upward force, which is given by

$$T_h = \sin\alpha \cdot T \tag{6.8}$$

with

$$\alpha = \arctan\left(\frac{\Delta x}{h_p}\right), \tag{6.9}$$

where Δx is the horizontal displacement of the float and h_p the height of the pulley above still water level, which is 0.6 m. The vertical cable force reduces accordingly

to

$$T_v = \cos\alpha \cdot T \tag{6.10}$$

for the last case.

Cases D^1 and D^2 include the full term of the tension force, as given by

$$T = m_c g - m_c \ddot{z}, \tag{6.11}$$

where $\ddot{z}$ is the acceleration of the system, g is gravity and m_c the mass of the counterweight. The difference in the setup between cases D^1 and D^2 is the way $\ddot{z}$ is extracted and used for the calculation of the tension force T. For case D^1 the acceleration of the float is extracted directly from the running simulation and made available for the calculation of T. For case D^2 the acceleration is calculated according to equation (6.5). Here, the buoyancy force F_b is taken as the fluid forces acting on the float, which is extracted from the running simulation and used in the same timestep to compute T.

Case E uses a different approach. Here, the masses are rearranged to be calculated within the 6 degree of freedom solver (6DOF) used by the FV solver. The 6DOF solver sums up all the forces that act on the floating body as given by

$$M\ddot{z} = -F_g + F_b + F_{ext} = -(Mg) + F_b + F_{ext}. \tag{6.12}$$

F_g, F_b and F_{ext} are the gravitational, buoyancy and external forces. M stands for the total mass of the system contributing to the inertia. The inertia of the system is calculated from the masses assigned to the 6DOF body in the domain. For case A this is $\left(m_f - m_c\right)$ and for cases B, C and D it is m_f. The inertia of the counter-

weight is neglected in cases A, B and C, whereas in cases D[1,2] it is included in the expression for the tension force. This force however, is handled by the solver as an external force and the aim for the last setup (case E) is to include the inertia of the counterweight in the 6DOF computation of the inertia to be handled by the solver directly. Hence it must be assigned to the 6DOF body as an additional mass.

The rearrangement of equation (6.5) gives

$$\left(m_c + m_f\right)\ddot{z} = \left(m_c - m_f\right)g + F_b = -\left(m_f - m_c\right)g + F_b. \tag{6.13}$$

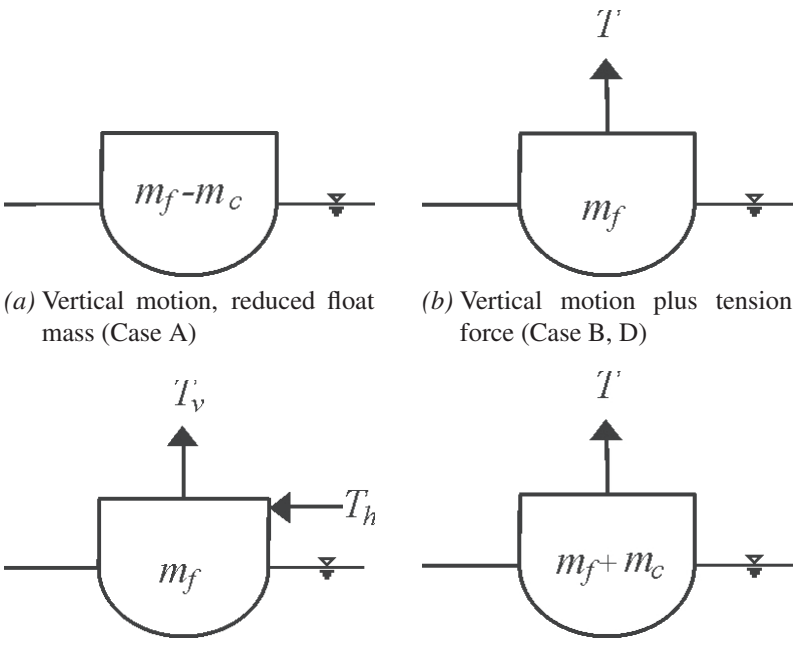
By including the mass of the counterweight on the left-hand side in the total mass of the system and thereby in the 6DOF body, the floating body becomes heavier and the initial floating position would be different to the physical experiment. Hence the external force needs to be increased, as expressed by

$$\left(m_c + m_f\right)\ddot{z} = -\left(m_f + m_c\right)g + F_b + 2m_c g, \tag{6.14}$$

which has the same form as equation (6.12). Thus case E has an increased body mass and a constant upward force of $2m_c g$ (Figure 6.9d, Table 6.2).

6.1.4 Hydrodynamics of bobbing float

Figure 6.10 shows the vertical translation for cases A and B compared with the physical experiment. The time history of this experiment can be divided into three phases. Phase one would represent the hydrodynamics of the float before the waves hit the structure, i.e. between t = 0 s and t = 2 s. In the second phase the float is



(a) Vertical motion, reduced float mass (Case A)

(b) Vertical motion plus tension force (Case B, D)

(c) Vertical and horizontal motion with components of tension force applied to float (Case C)

(d) Vertical motion plus tension force, increased float mass (Case E)

Figure 6.9: Representation of mechanical system in CFD package through body forces, values according to Table 6.2

Table 6.2: Properties of Single Float Simulations

	Simulation				
	A	B	C	D[1,2]	E
Mass of float	$m_f - m_c$	m_f	m_f	m_f	$m_f + m_c$
Vertical Motion	✓	✓	✓	✓	✓
Horizontal Motion	-	-	✓	-	-
Rotation	-	-	-	-	-
Vertical Force	-	$m_c g$	$m_c g \cdot \cos\alpha$	$m_c g + m_c \ddot{z}$[1,2]	$2m_c g$
Horizontal Force	-	-	$m_c g \cdot \sin\alpha$	-	-

m_f: Mass of float in physical experiment (2.2 kg); m_c: mass of counterweight in physical experiment (1.7 kg); ✓: included; -: excluded; 1: $\ddot{z}$ extracted from running simulation and fed into the calculation of $T = m_c g - m_c \ddot{z}$; 2: $\ddot{z}$ obtained as given by equation (6.5), the buoyancy force F_b is extracted and fed back into the calculation of the tension force T

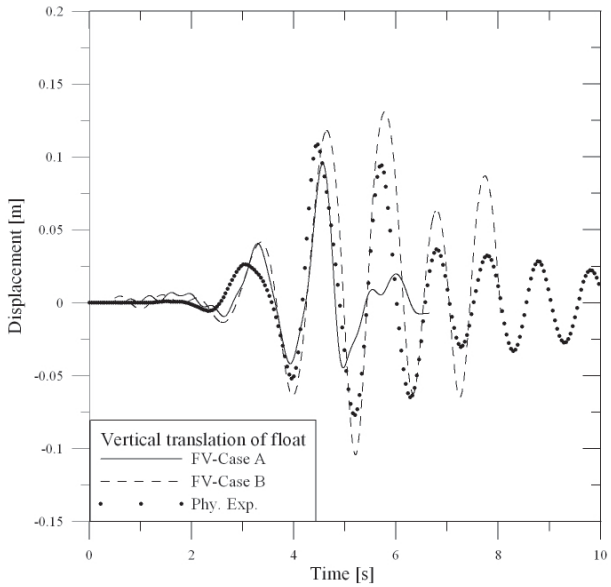


Figure 6.10: Vertical translation of float for tethered cases A and B

riding on the waves, which occurs between $t = 2$ s and $t = 5$ s. After $t = 5$ s the waves have passed the float. This is the free-oscillation phase in the physical experiment. In the first phase, differences between the numerical simulations and the physical tank test can be observed. In the physical experiment, the float sits stationary at the free surface. In contrast, the float in the numerical simulations oscillates slightly. The re-positioning of the float until all forces are in equilibrium, however, is the case for both setups, i.e. physical and numerical test. In the physical experiment this position was reached before the experiment was started and not quantified. It is known though, that small differences between the physical experiment and the numerical simulation regarding the initial floating position exist. This is due to small geometric discrepancies of the floats used in the experiments described in Stallard et al. (2008) and the CFD ones as seen in Figure 6.3.

Furthermore, during the first 0.2 s of the numerical simulation the float is held in place. The vertical fluid forces, which can be seen in Figure 6.11a for the two

tethered cases, are calculated for the stationary body to stabilise the simulation. After 0.02 s the forces are ramped up linearly in the motion solver of the code to reach 100 % after another 0.38 s. Within this time the motion model of the solver engages and computes the translation of the float. As the float is not held at its equilibrium position it adjusts itself in the numerical simulation in the first phase.

In the second phase, where the float rides the waves, slight differences between the numerical simulation and the tank tests are expected due to the differences in the wave signal (compare Figure 6.6). All results, however, follow the surface elevation up to t = 5 s, when they reach the subsequent trough of the main wave.

After this time, the float in the physical experiment submerges deeper, which generates restoring buoyancy forces that catapults the float out of the water to an altitude almost as high as the maximum translation of 0.11 m occurring due to the main wave (see Figure 6.10). The translation of the CFD case B is in phase with the wave and the physical experiment but larger in amplitude. In the wave trough following the main wave the numerically simulated float gets submerged even deeper than the float in the physical tank test and rises out of the water higher than the main wave vertical translation. The third and fourth oscillation are in phase with the physical experiments, but larger in the total amplitude. In particular the crest translation is less well resolved than the trough displacement. The vertical translation of case A damps out immediately after t = 5 s, at which point the bobber passes the second trough and follows the water surface.

The vertical pressure and shear forces on the float due to the waves can be seen in Figure 6.11a. The forces on the stationary float are the same for both cases. As the bobber is released the phase in which the forces act on the bodies is half the

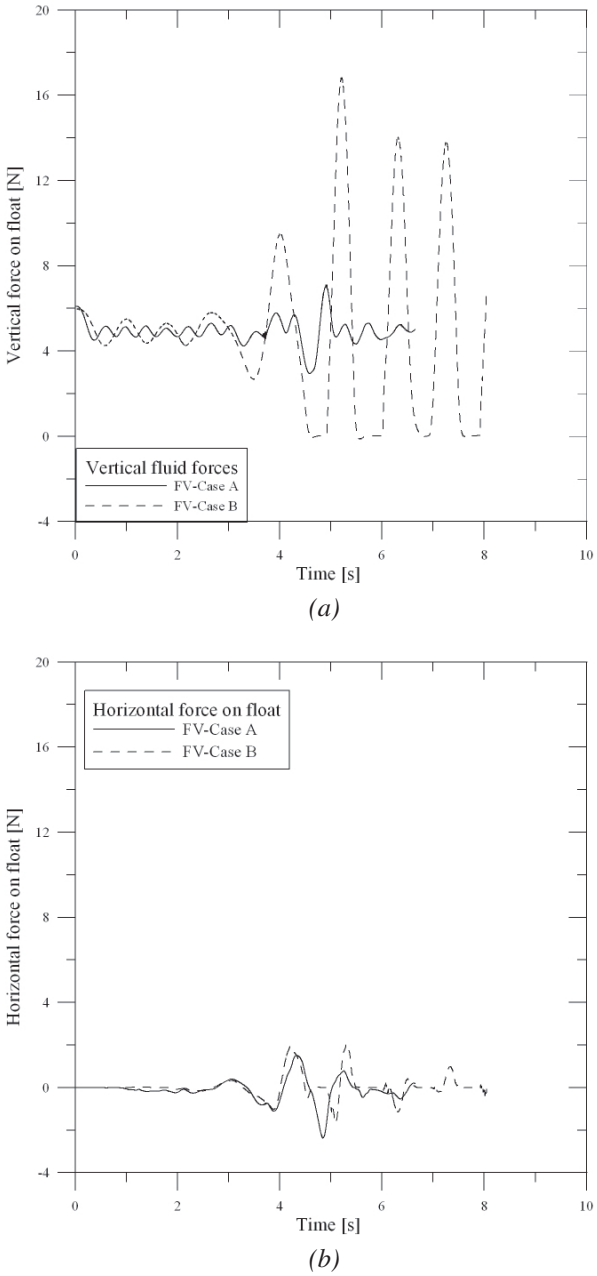


Figure 6.11: Vertical (a) and horizontal (b) forces on bobber float in extreme waves (Cases A and B)

frequency for the heavy float case compared to the reduced-weight case (Phase 1). During phase 2 the fluid forces increase slightly for case A, but significantly for case B. The float in case A never gets airborne, whereas in case B, the float leaves the water 4 times indicated when the forces acting on the float decrease to 0 N.

The horizontal forces are similar for phase 1 and 2 of the simulation for both cases as are the translations. When a wave approaches the float a positive horizontal force due to the positive fluid particle velocity acts on the structure. Accordingly, a negative current behind the wave is experienced by the float resulting in negative fluid forces. After 5 s of the simulation the horizontal forces differ from each other due to the different dynamics of the float. Whereas the horizontal forces for case A reduce to 0 N, the case with constant upward force experiences forces in the x-direction. The total horizontal force for both cases A and B range from -2 to +2 N, which is significantly less than the vertical forces on the structure, which range from 0 to 18 N as seen in Figure 6.11a. Knowledge of the horizontal forces is important for dimensioning the tether system to restrict the float motion in the x-direction. Also, for the design of the mooring system this data is important.

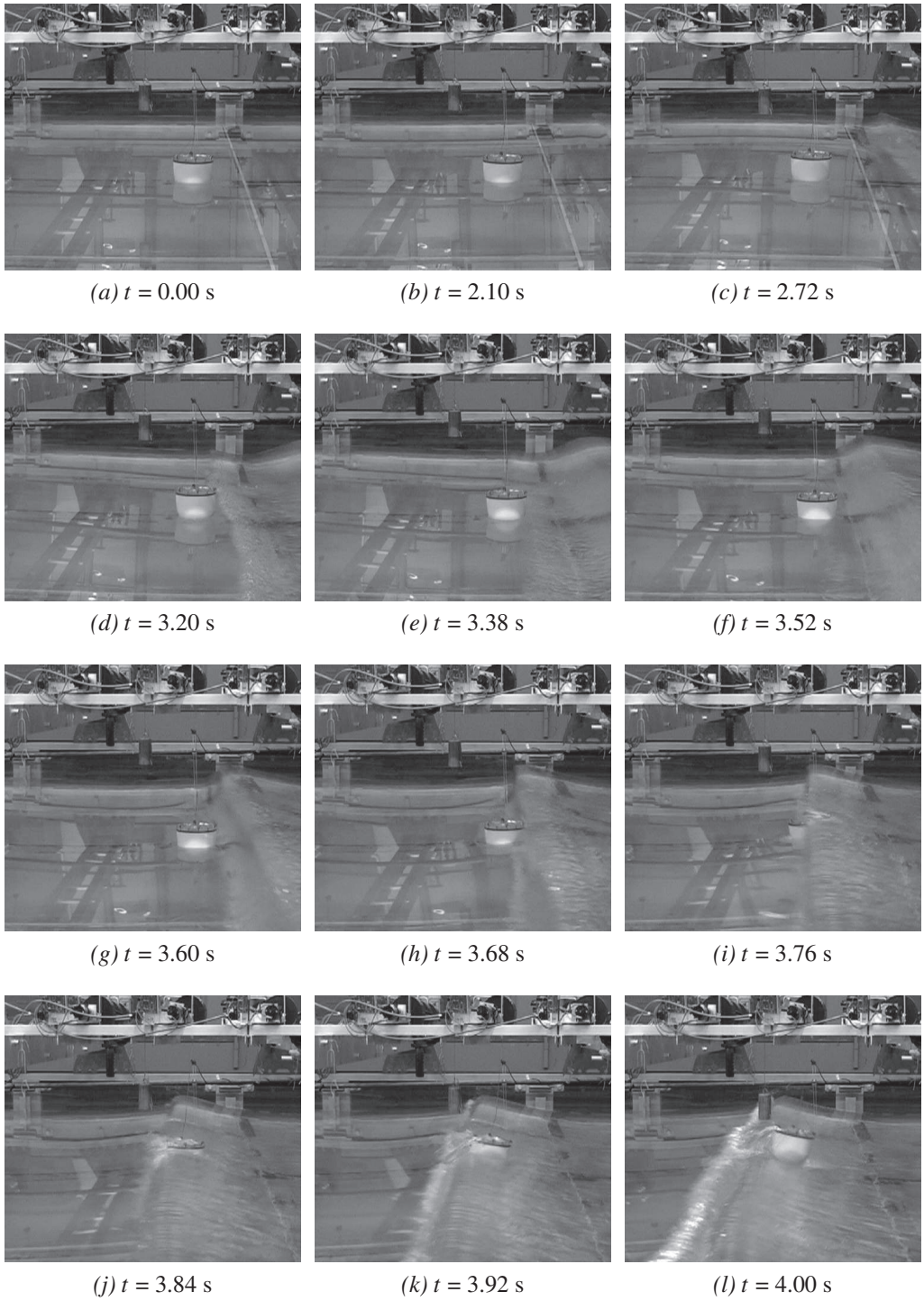
By including the motion in the horizontal direction in the numerical simulation the untethered physical experiment has to be used to compare results with. The mechanical and numerical setup is shown in Figures 6.3b and 6.9c. The numerical setup builds on case B, but the constant upward force T is split into its horizontal and vertical components T_h and T_v depending on the horizontal displacement relative to the pulley, which is taken as the pivoting point and 0.6 m above the float in its initial position.

The hydrodynamics of the float in the physical experiment can be seen in Figures 6.12 and 6.13. The pictures are taken from the video footage of the physical

experiments in the wavetank of the University of Manchester carried out and supplied by Tim Stallard. It must be noted that the times underneath the pictures are not synchronised with the results plots. From the initial position at $t = 0$ s the approaching small wave can be seen. As the larger wave is more important, Figure 6.12b shows the float after 2.1 s when the smaller first wave has already passed the bobber. The float rides this wave and hardly travels in the horizontal direction at all. The vertical displacement can be judged by the motion of the counterweight relative to the frame of the wavetank in the background of the pictures. When the large main wave approaches, first to be seen around $t = 3$ s, the float remains almost stationary in both horizontal and vertical directions. The counterweight moved significantly in Figure 6.12i, in which the float is submerged below the wave crest. Also at this time, the horizontal displacement increases. The float rises out of the back of the wave in Figure 6.12l. Here, the body is not only translated horizontally and vertically, but also rotated around its y-axis (which runs parallel to the wave crest). Figure 6.13a-l show the sequel of this event. After the main wave has passed the float, it acts as a pendulum. It travels airborne in the negative x-direction, hits the following wave front at $t = 4.8$ s, sinks in in a straight line, still being rotated. In the following Figures 6.13g-i the float rises again and simultaneously moves in the positive x direction and swings back again in Figures 6.13j-l.

The simulated results for the untethered case can be seen in Figure 6.14. The vertical and horizontal translations are combined using

$$\Delta_{xz} = \sqrt{\Delta_x^2 + \Delta_z^2} \cdot \frac{\Delta_z}{|\Delta_z|}, \tag{6.15}$$



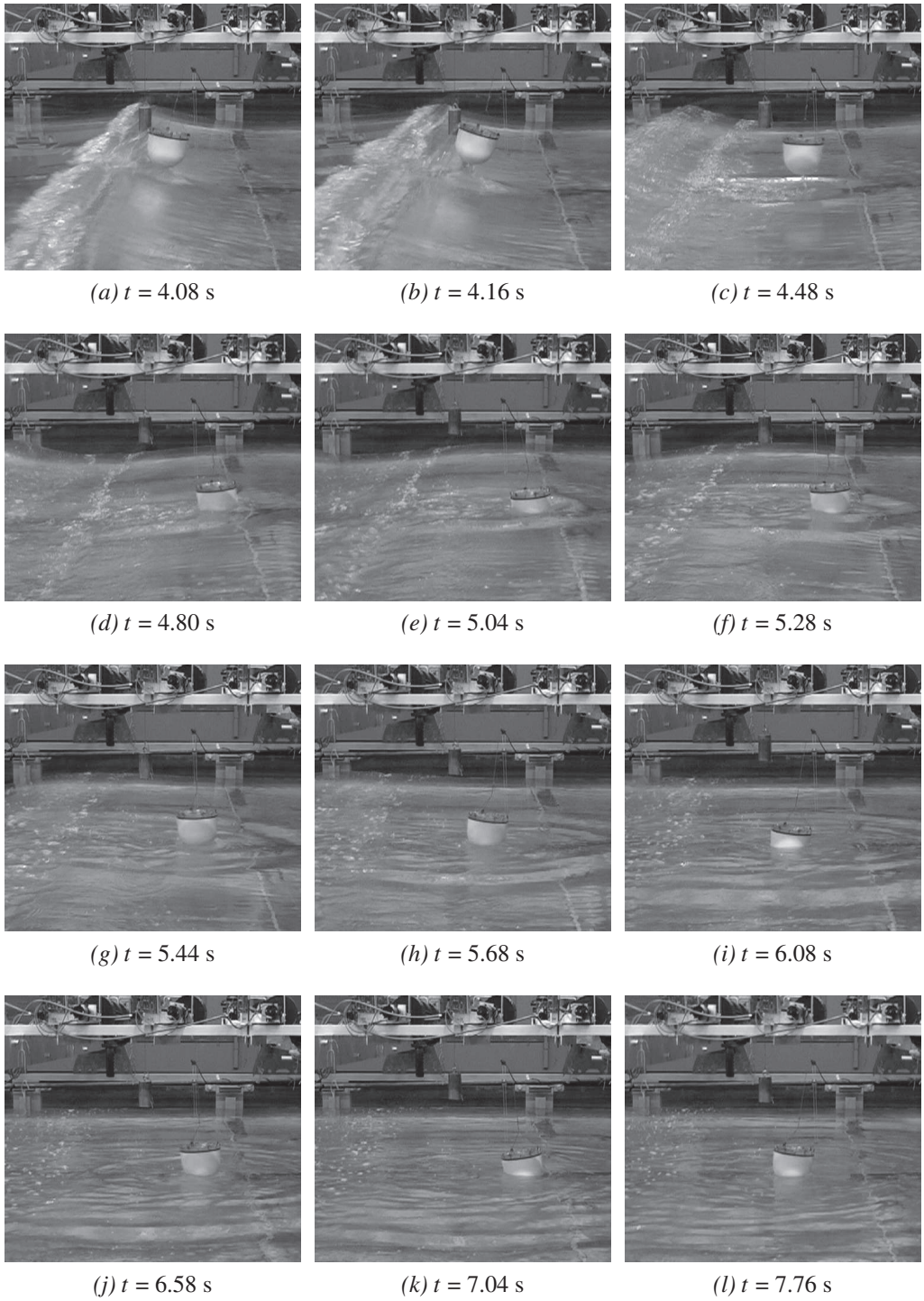
(a) $t = 0.00$ s *(b)* $t = 2.10$ s *(c)* $t = 2.72$ s

(d) $t = 3.20$ s *(e)* $t = 3.38$ s *(f)* $t = 3.52$ s

(g) $t = 3.60$ s *(h)* $t = 3.68$ s *(i)* $t = 3.76$ s

(j) $t = 3.84$ s *(k)* $t = 3.92$ s *(l)* $t = 4.00$ s

Figure 6.12: Single untethered float in physical experiment (Stallard 2010)



(a) $t = 4.08$ s *(b)* $t = 4.16$ s *(c)* $t = 4.48$ s

(d) $t = 4.80$ s *(e)* $t = 5.04$ s *(f)* $t = 5.28$ s

(g) $t = 5.44$ s *(h)* $t = 5.68$ s *(i)* $t = 6.08$ s

(j) $t = 6.58$ s *(k)* $t = 7.04$ s *(l)* $t = 7.76$ s

Figure 6.13: Single untethered float in physical experiment (Stallard 2010)

where Δ_i is the displacement in the i-direction. These are compared with the measured displacements from the physical tank test shown as the dotted line in Figure 6.14a. The numerical results are in phase with the physical experiments for the oscillations up to t = 6.5 s. After that, the oscillations in the physical experiments decay with a longer period than the numerical results. The magnitude of the oscillations differs significantly through the whole experiment and even increases in the numerical simulation after t = 7.2 s. From the force plots in Figure 6.14b can be seen that the vertical forces due to pressure and shear peak at a constant value, rather than being damped out. The ratio of vertical to horizontal forces that act on the float is similar to the tethered cases. The vertical force ranges from 0 to 17 N and the horizontal force due to the waves oscillates between -2 and 2 N.

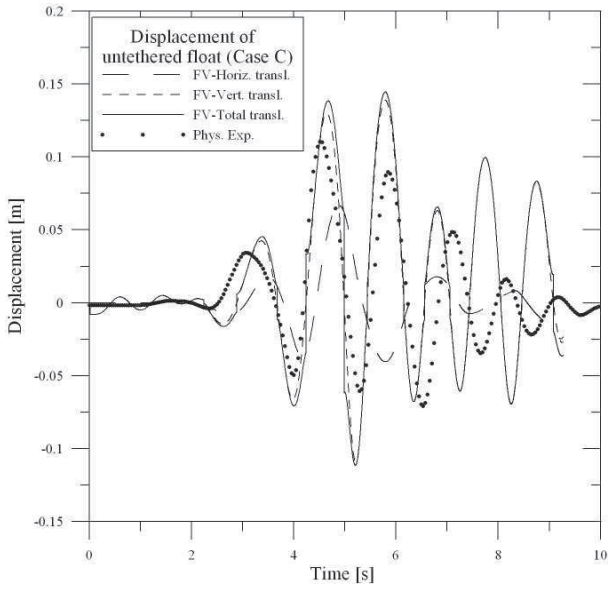
So far, the main differences in the numerical results compared to the physical data are the larger oscillations of the float in the third phase of the experiment, when the waves have passed the structure, which occurred for both the tethered and untethered cases with $T = m_c g$, and also the phase change of the oscillations for the untethered experiment. These can be linked to two differences in the numerical setup, which are the simplification of the tension force, rather than including the full term as given in equation (6.6). This is the cases A, B and C. Furthermore, the restriction of the float motion to vertical and horizontal displacements only. This does not allow for the rotational motion as it was observed in the physical experiment in Figures 6.12g-l and 6.13a-f. After being hit by the main wave, the float re-enters the water almost vertically as it does by definition in the numerical simulation. The differences in the phase are small up to this point of the simulations. After that however, the results differ in phase, which might be due to the float being rotated around its y axis (Figures 6.13g-l). This will have an effect on the restoring

fluid forces and thereby on the motion of the float.

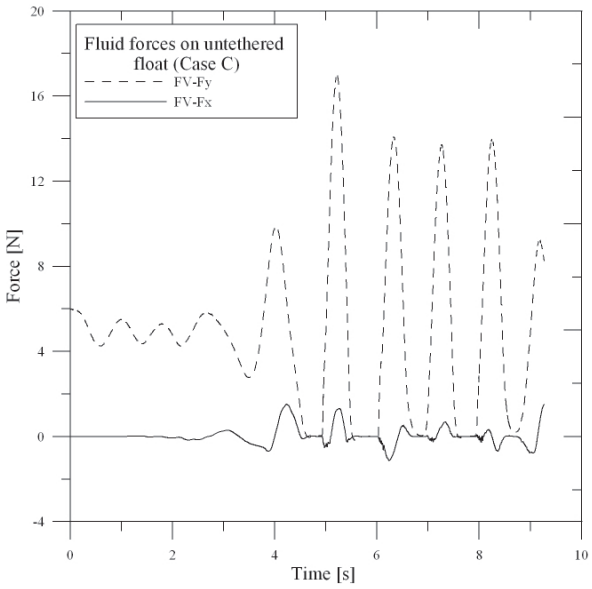
These two issues can be dealt with separately and the next step to improve the numerical results is to include the missing term of the upward force described in Equation (6.6) for the tethered setup to improve the magnitude of the oscillations. The phase shift and the rotational motion does not occur for this setup and may be improved once the magnitude of the oscillations in the numerical simulations matches the measured results. Depending on the acceleration of the float $\ddot{z}$ the total tension force is reduced by $m_c\ddot{z}$ in the setups used for cases B and C. As the acceleration of the float is calculated by default in the software package the improved setup for the tethered case can be seen in Figure 6.9b but with

$$T = m_c g - m_c \ddot{z}, \tag{6.16}$$

where m_c is the mass of the counterweight and $\ddot{z}$ the acceleration of the system calculated by the solver and g is gravity. The vertical translation for case D[1] can be seen in Figure 6.15a. These are significantly over estimated. However, the decay of the oscillations has improved qualitatively. Another option to include the acceleration of the mechanical system is to define this term as it is given in Equation (6.5). Therefore the buoyancy force F_b acting on the float needs to be extracted and made available for the computation of T. The buoyancy force is equal to the mass force of the displaced water volume by the float. This volume is not known during the computation as it changes with every timestep and cannot be calculated for the float as it is modelled as a cavity in the mesh, and so does not contain any CVs where the Navier-Stokes equations could be solved. Extracting the draught of the float during the simulation would be possible. However, as the waterlevel is not



(a) Translations



(b) Forces

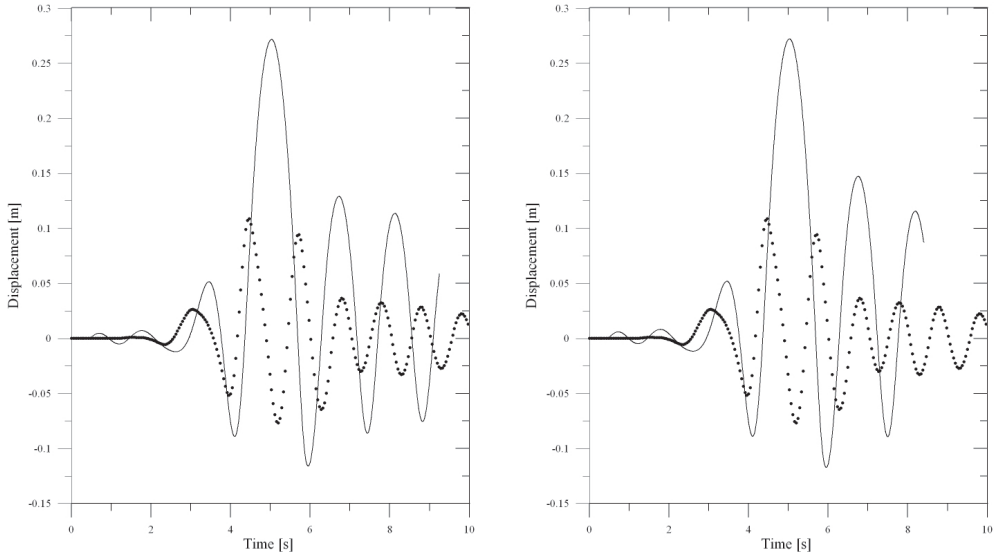
Figure 6.14: Hydrodynamics of untethered float (Case C)

level, meaning that the orientation and height changes due to the influence of the float on the waves, this would be very difficult for this method. The buoyancy force can also be calculated from the integrated stresses to receive the fluid force on the surface of the structure. Thus, for case D^2 the vertical fluid forces from pressure and shear are used as F_b for the computation of T. The vertical translations for this case can be seen in Figure 6.15b. Unfortunately this results in the same much overestimated displacements of the float as for the setup where $\ddot{z}$ is taken from the simulation directly.

For case E, where the inertia of the counterweight is calculated by the 6DOF solver directly by assigning it to the mass of the float, rather than including it in the expression for the tension force, the vertical translations can be seen in Figure 6.16 giving the same results as for cases $D^{1,2}$. Regardless of the method of including the inertia term in the calculation of the float displacements, the solver overestimates the vertical translation.

6.1.5 Numerical performance

Compared to the wave only cases discussed in Chapter 4 and the wave-structure interaction cases for the fixed horizontal and vertical cylinders (Chapter 5), the problem of the free floating body in waves is more complex due to the resulting motion of the float. This is reflected in the time the solver needs to solve the Navier-Stokes equations including the equations for motion, i.e. mesh motion. Figure 6.17 plots the elapsed time per timestep against the velocities of the float taken from the untethered bobber case. The simulation was done in parallel on 8 partitions, each assigned to one processor of a Duo Quad Core 2.5 GHz Windows XP workstation



(a) D^1 with $T = m_c g - m_c \ddot{z}$

(b) D^2 with $T = m_c g - m_c \frac{(m_c - m_f)g + F_b}{m_f + m_c}$

Figure 6.15: Vertical translation of tethered float for cases D^1 and D^2; (a) $\ddot{z}$ from running simulation used for calculation of T; (b) F_b from extracted pressure and shear forces on the float

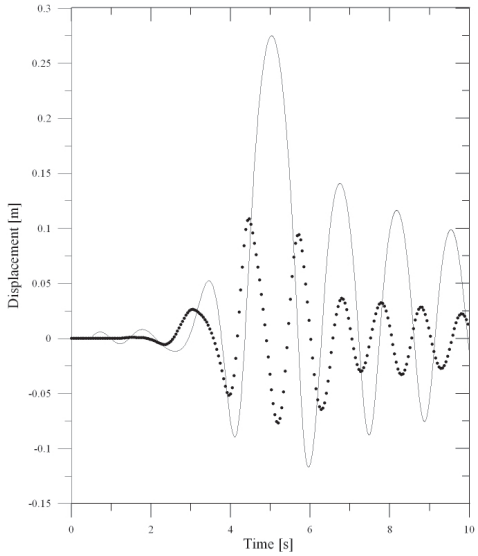


Figure 6.16: Vertical translation of float for case E

with 16 Gb RAM. It can be seen that the necessary time per timestep increases with the float velocity. As these simulations were done with a timestep of 0.0005 s, the runtime significantly increases with a motion model included. The total run time for the tethered case and 16400 timesteps was 691,200 s, so approximately 8 days. On average that is 42 s per timestep on 8 CPUs.

It should be noted that the run time per timestep is also influenced by other processes running on the system. These can be Windows specific processes such as the task manager or user started processes as e.g. anti virus software or when the computer is connected to the internet possible updates that run in the background. However, as the workstation was only used for numerical simulations, other processes are kept at a minimum and would not have influenced the run time significantly.

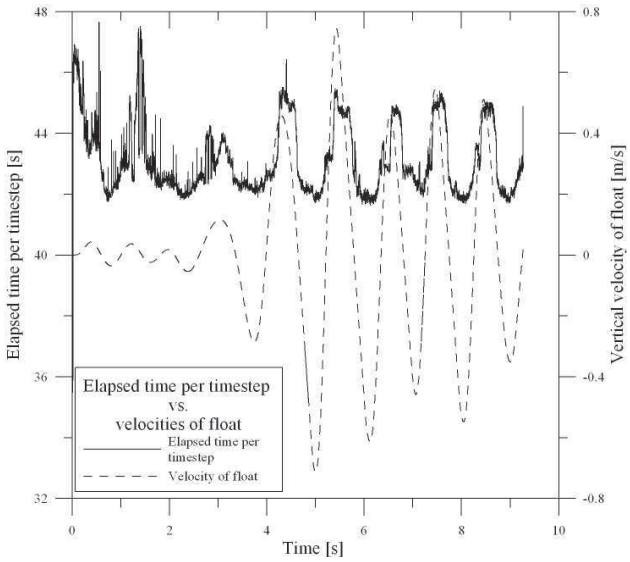


Figure 6.17: Elapsed time vs. float velocity

6.2 Pelamis

6.2.1 Description of the Device

This section discusses the application of CFD to a single section of the Pelamis WEC in regular waves. For these studies the verified solvers are used to generate data which was supposed to be measured in physical experiments. During the experiments technical problems arose, which corrupted the data. Thus the lost data is replaced by the previously validated CV-FE and FV solvers.

The Pelamis wave energy converter consists of 4 hinged sections, arranged end to end and which are displaced by the waves. It is moored to the sea bed, which makes the device free-floating. Each section individually interacts with the waves and the adjacent section with 6 degrees of freedom. This makes the hydrodynamics very complex. As described in Chapter 2, simplified numerical models exist, to assess the efficiency and survivability in different sea states. However, all of these models are aimed with different purposes. For example, when motion is calculated to assess the performance and efficiency in small and regular seas, the waves are linearised and not influenced by the device. Thus, higher order effects are neglected. To account for such higher order effects the simplified numerical models may be validated against physical tank tests or, as done here, against fully non-linear CFD results. To improve the outcome, the single section is modelled as 9 strips so that the forces acting on the different positions on the structure can be extracted separately. Due to the highly complex hydrodynamics it was not possible by the time this thesis was done to simulate the full device in such a non-linear manner with motion of the device included.

6.2.2 Simulation of a fixed single section in regular waves

For the simulations regular waves with an amplitude $A = 0.1$ m, a wave period $T = 1.77$ s and wavelength $\lambda = 4.922$ m are generated. The wavenumber k is 1.277 m^{-1} and the wave steepness kA is 0.1276. This makes the waves weakly non-linear if $kA < 0.1$ is considered to be linear.

The computational domain for both solvers is the same and can be seen in Figure 6.18. It is 30 m long, 2.8 m wide and 5.6 m high with the still water level being 2.8 m. The fixed single section representing one section of the Pelamis WEC is located 4.4 m behind the inlet and initially half submerged. Rather than modelling the actual Pelamis geometry, the shape is simplified being a cylinder with a radius of 0.095 m with spherical ends. The total length of the structure is 1.4 m. For the simulations the surface of the cylinder is subdivided into 9 strips, 2 of which are the spherical ends and the remaining 7 sections are evenly spaced on the cylindrical part of the device. Thus the forces on the structure can be investigated for each section separately. The first section is located at the bow of the cylinder, closest to the inlet.

The boundary arrangement is the same as described for the vertical and horizontal cylinder cases in Chapter 5. The waves are generated using a velocity inlet. Here, the horizontal and vertical wave velocity components are applied for the water fraction only. The vertical location of the surface elevation is defined as being where the water volume fraction equals 0.5 (see Chapters 4.1.2 and 4.2.3). The bottom and far end as shown in Figure 6.18 are wall boundaries, the sides are symmetry planes and the top is a pressure outlet, with only air being allowed to leave or enter the domain.

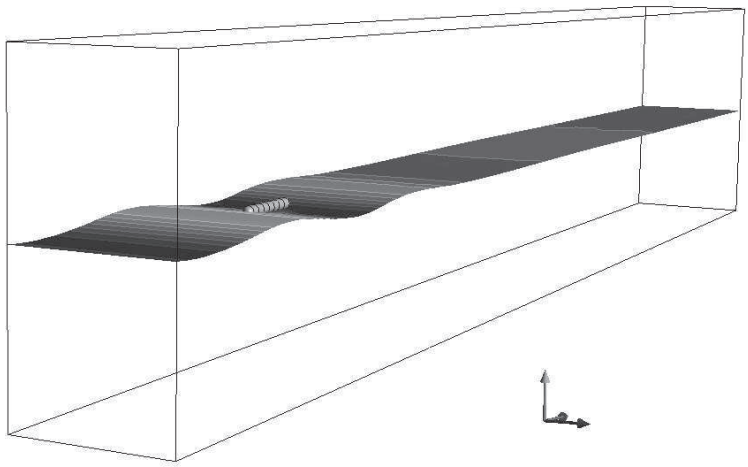


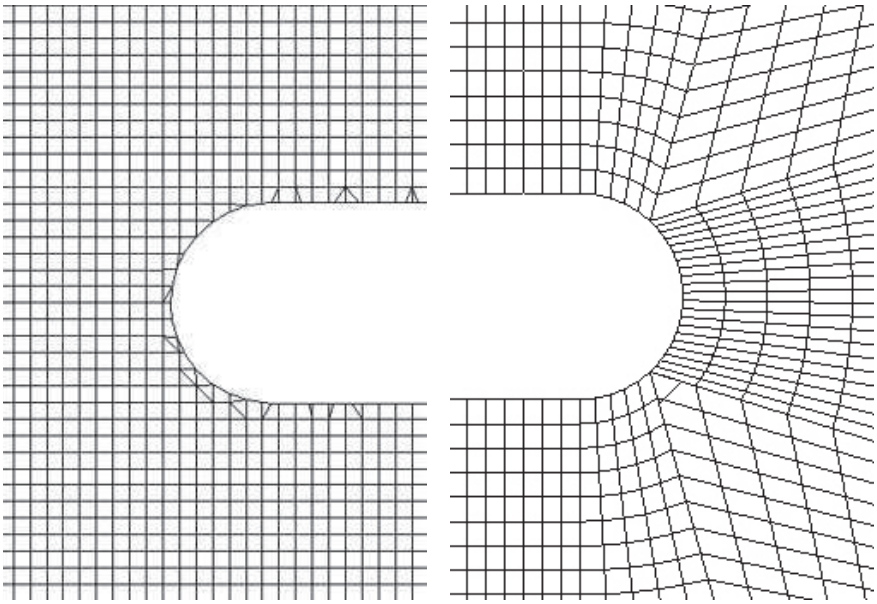
Figure 6.18: Domain for single Pelamis section simulations

The meshes for both codes are different from each other. For the FV solver the mesh consists of mainly hexahedral cells. Around the Pelamis section the cell are also tetrahedral and polyhedral shaped, as the structure is cut out of the initial pure hexahedral mesh by clipping the cells(Figure 6.19a). The structure is represented by a total of 4154 faces. These comprise triangular, quadliteral and polyhedral faces as presented in Table 6.3.

A vertical section of the mesh around the stern of the cylinder used for the CV-FE solver (CFX) can be seen in Figure 6.19b. Here, the mesh is body fitted and consists of 1,116,040 cells. The number of faces representing the structure are 458 cells each for the spherical parts and 580 each for the strips on the cylindrical part of the device. All of these 4976 faces are quadrilateral.

6.2.3 Wave Loads on the Section

One of the main interests when designing a WEC are the forces on the structure. All forces presented here are normalised by dividing through with the buoyancy force



(a) FV STAR CCM+) *(b)* CV-FE (CFX)

Figure 6.19: Mesh sections near structure

Table 6.3: Mesh properties of FV single Pelamis section

	No. of faces and type			
	Tri	Quad	Poly	Total
Sec 1 (Bow)	93	155	1	249
Sec 2	0	484	0	484
Sec 3	8	486	0	504
Sec 4	0	528	0	528
Sec 5	2	526	0	528
Sec 6	0	528	0	528
Sec 7	0	528	0	528
Sec 8	6	528	0	534
Sec 9 (Stern)	106	176	0	282
Total	215	3939	1	4154

of the fully submerged Pelamis section as given by

$$F_i' = \frac{F_i}{\rho g V}, \tag{6.17}$$

where F_i is the extracted force from CFD in the i-direction, ρ is the density of water, g the acceleration due to gravity and V the volume of the section given as

$$V = \frac{1}{4}\pi D^2 l + \frac{1}{6}\pi D^3 \tag{6.18}$$

where D is the diameter (0.19 m) and l the length of the cylindrical section, which is 1.21 m. The time is normalised to the wave period. Figure 6.20 shows the forces on the single Pelamis section in the x, y and z- direction for one wave cycle. Both the CV-FE (Figure 6.20a) and FV solver (Figure 6.20b) compute the same values for the forces. The main contribution comes from the heave forces in the z-direction. At the beginning of the wave cycle at $t/T = 0.0$ the total heave on the device is approximately half the buoyancy force. This implies that the device is half submerged, which is confirmed in Figures 6.21a and 6.22a for the CV-FE and FV solver respectively. Here, the surface elevation around the cylinder can be seen from the side. The wave approaches from the left. Furthermore, the 9 strips on the cylinder can be identified in these plots. The bow (Sec 1) is on the left. On average the Pelamis cylinder is approximately half submerged. Whereas the upstream part of it is underneath the wave crest, the stern is hardly in the water and contributes less to the heave force.

The total dimensionless heave peaks at 0.8 for $t/T \approx 0.25$. Here, the device is nearly fully submerged as it can be seen in Figures 6.21c and 6.22c. Pelamis is

designed to dive through steep waves, rather than ride on it. This should prevent the structure from being damaged due to the sections being moved too much relative to each other. The test case is not a very steep wave, but as the model is not allowed to move and thereby has to dive through the wave, this may represent the design state for more extreme wave cases. For this weakly non-linear wave the dimensionless vertical forces oscillate around 0.5 almost symmetrically. However, as the real device floats, it would not experience such low heave forces as can be seen around $t/T = 0.8$. Instead the device would follow the wave trough.

Besides the vertical forces the drift forces in the wave direction are interesting for the dimensioning of the mooring. These are small compared to the heave force acting on the device. The dimensionless force in the x-direction oscillates between -0.08 and 0.08. In front of the wave the device experiences positive drift floating away from the anchor point and behind the wave it moves in the opposite direction due to negative forces acting on the structure.

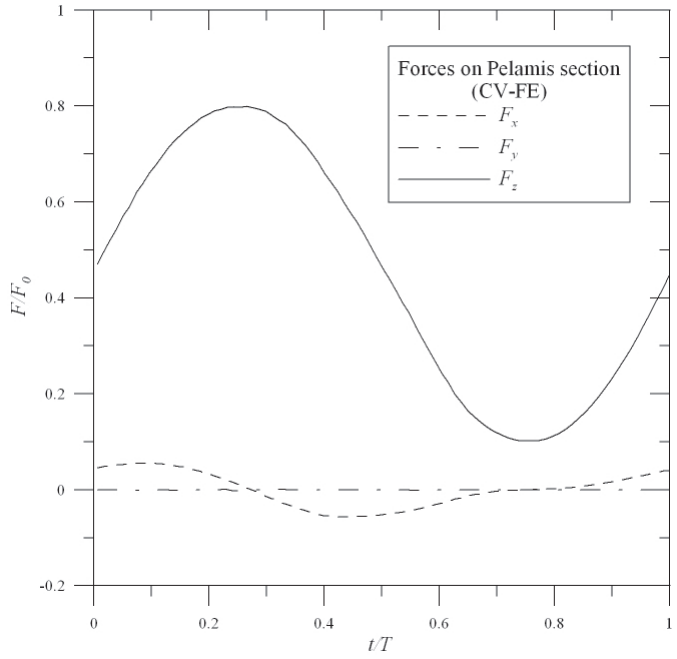
These drift forces are only dependent on the forces acting on the bow (Sec 1) and stern (Sec 9) of the device. This can be seen in Figures 6.23b and 6.24b, where the drift forces experienced by each section are plotted. Here, two wave cycles are shown. In both graphs the force acting on the bow is positive and negative on the stern. On the remaining sections zero horizontal forces are be measured.

Figures 6.23a and 6.24a show the vertical force calculated by both solvers for each of the 9 strips. These forces add up to the total heave force as seen in Figure 6.20. Due to the smaller surface area of the spherical end sections of the device the vertical force is smaller than for the remaining seven. Qualitatively all force curves show similar behaviour. After the maximum is reached, a saddle point is encountered after which the forces reduce due to the falling water level. From

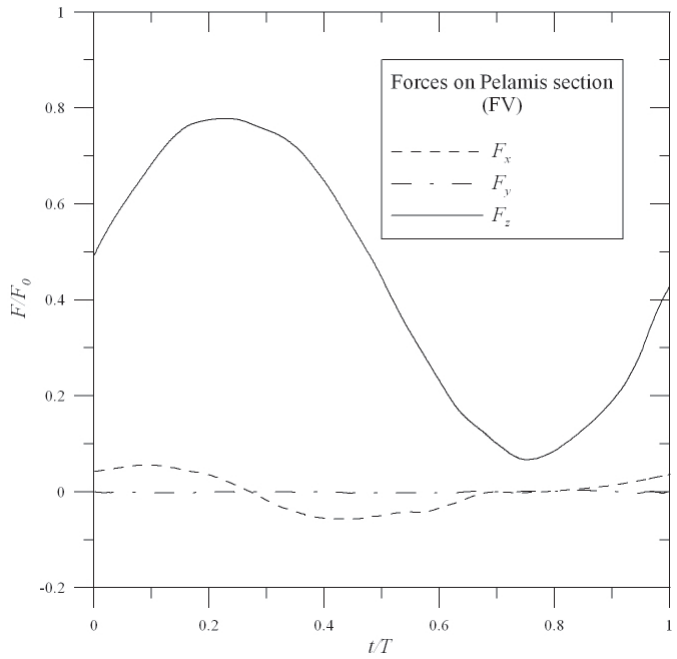
section 2 to section 8 the forces reduce slightly and also become more disturbed. The maximum dimensionless heave force on a strip is 0.11. Knowing this and the distribution during the wave cycle means that the device may be designed to be more cost efficient by saving material such as steel for the hull.

Figures 6.25 and 6.26 show the aerial view of the Pelamis section for the same instances in time as in Figures 6.21 and 6.22. Plotted is the isosurface of the water volume fraction of 0.5 representing the water surface around the device. On the isosurface the contours are plotted, which indicate the vertical position of the isosurface between 2.6 m (dark) and 3.0 m (light). The initial free surface was set to 2.8 m.

In Figure 6.25a the approaching wave can be seen left of the device. By this time of the simulation steady state is reached and 3 waves have already passed the device. The last can just be seen on the right side of the picture leaving the graphic. In front and behind the device the waves run parallel to the device hardly being disturbed. No influence from the boundaries can be identified, as the contours are parallel. Compared to physical tank tests, where wall friction is existent, the symmetry boundaries used for the simulation do not allow for this. Hence the NWT may not be as wide as a physical wave tank for the same problem. Blocking effects cannot be identified either. Close to the cylinder diffraction effects can be seen. The diffracted waves turn towards the stern of the device. In Figure 6.25c the crest has just approached the bow of the device. Here, the wave height reduces close to the device, whereas further away the wave height is maintained. The diffraction

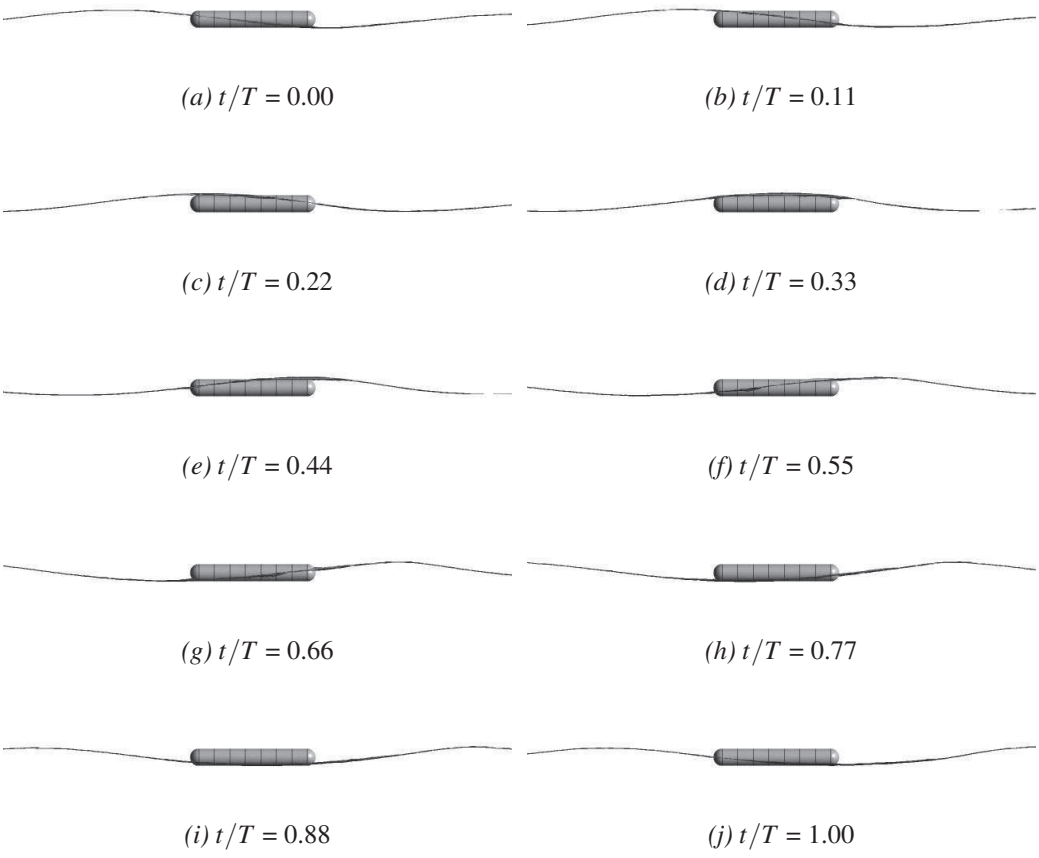


(a) CV-FE (CFX)



(b) FV (STAR CCM+)

Figure 6.20: Total force on single Pelamis section



(a) $t/T = 0.00$ *(b)* $t/T = 0.11$

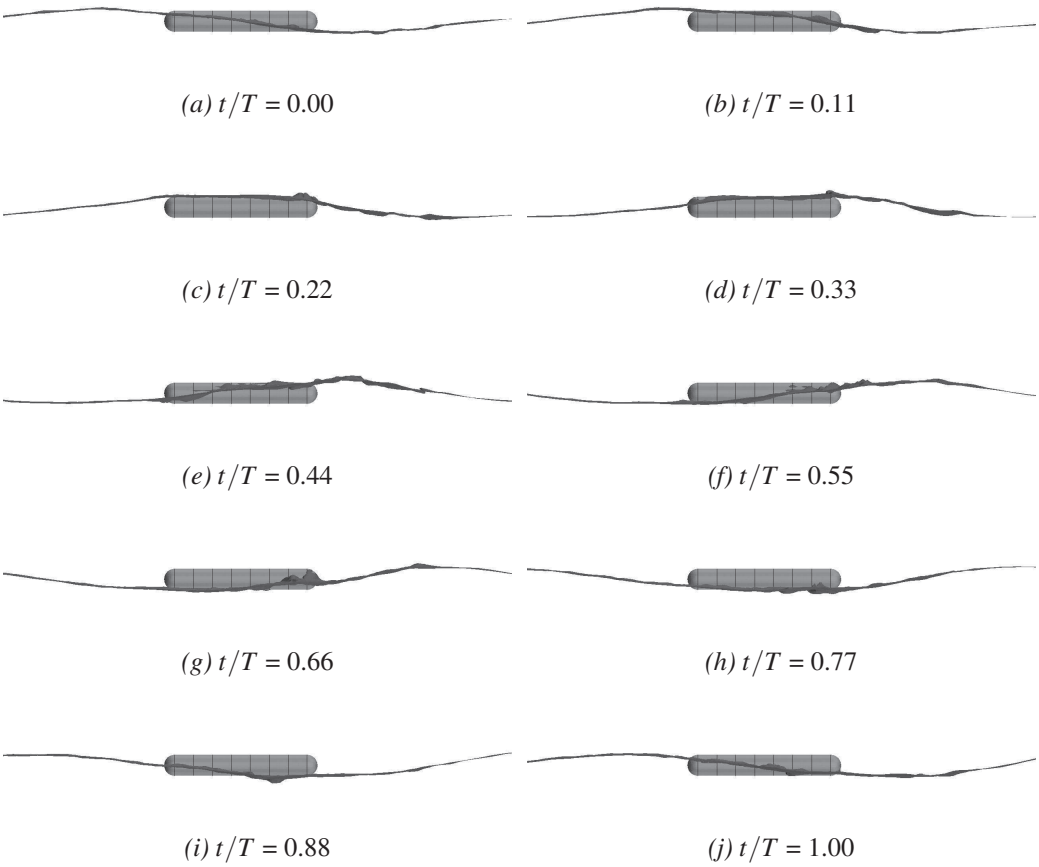
(c) $t/T = 0.22$ *(d)* $t/T = 0.33$

(e) $t/T = 0.44$ *(f)* $t/T = 0.55$

(g) $t/T = 0.66$ *(h)* $t/T = 0.77$

(i) $t/T = 0.88$ *(j)* $t/T = 1.00$

Figure 6.21: Side view on Pelamis section in regular waves (CFX)



(a) $t/T = 0.00$ *(b)* $t/T = 0.11$

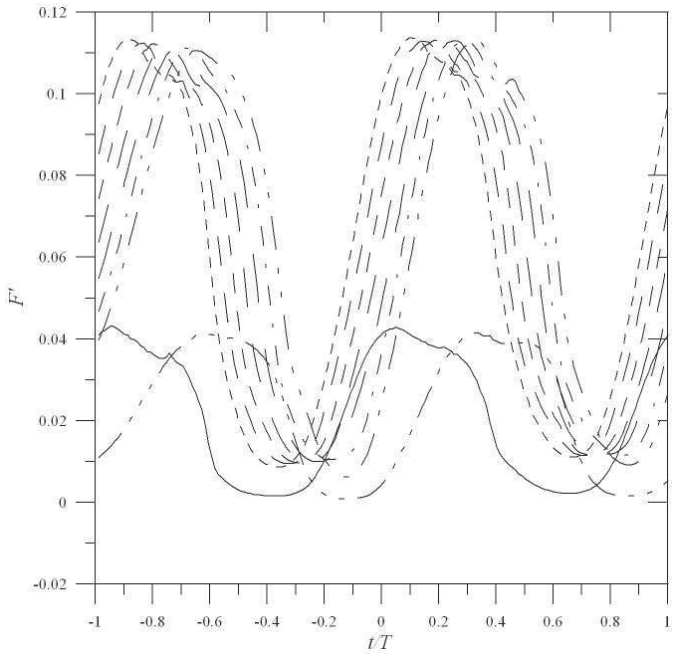
(c) $t/T = 0.22$ *(d)* $t/T = 0.33$

(e) $t/T = 0.44$ *(f)* $t/T = 0.55$

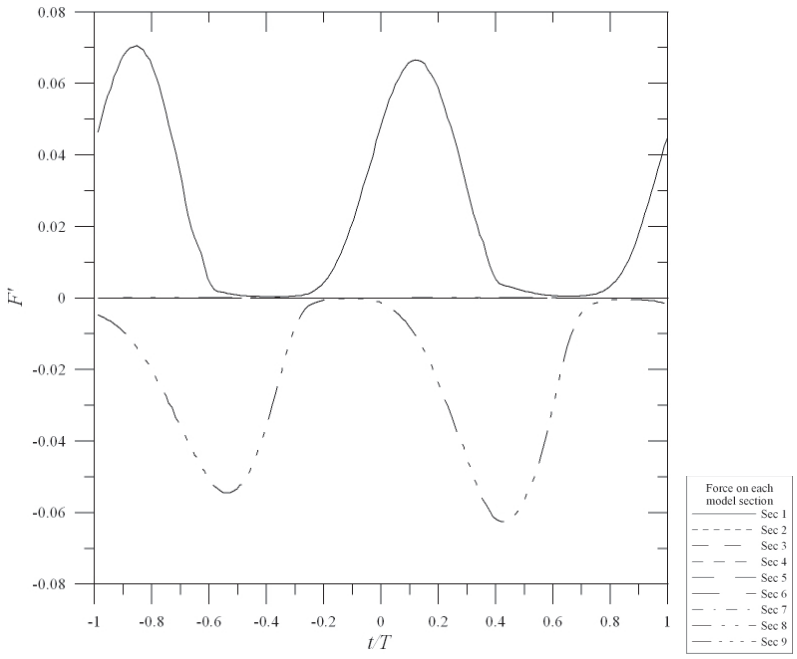
(g) $t/T = 0.66$ *(h)* $t/T = 0.77$

(i) $t/T = 0.88$ *(j)* $t/T = 1.00$

Figure 6.22: Side view on Pelamis section in regular waves (STAR CCM+)

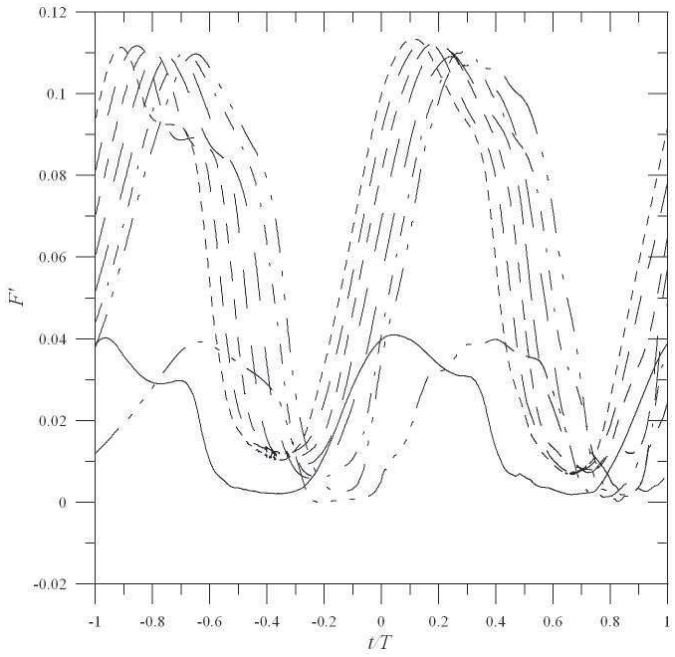


(a) Heave

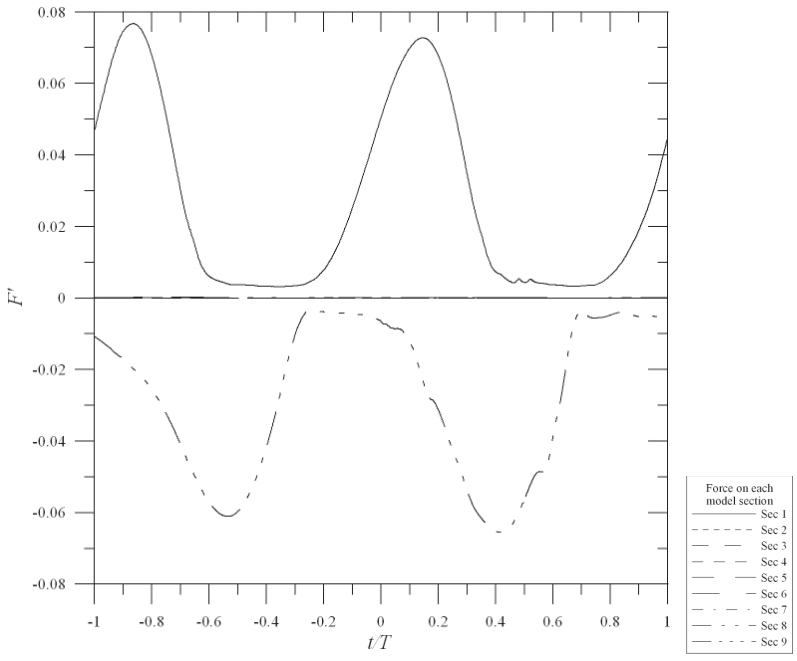


(b) Drift

Figure 6.23: Heave and drift forces calculated by CV-FE solver (CFX)



(a) Heave



(b) Drift

Figure 6.24: Heave and drift forces calculated by FV solver (STAR CCM+)

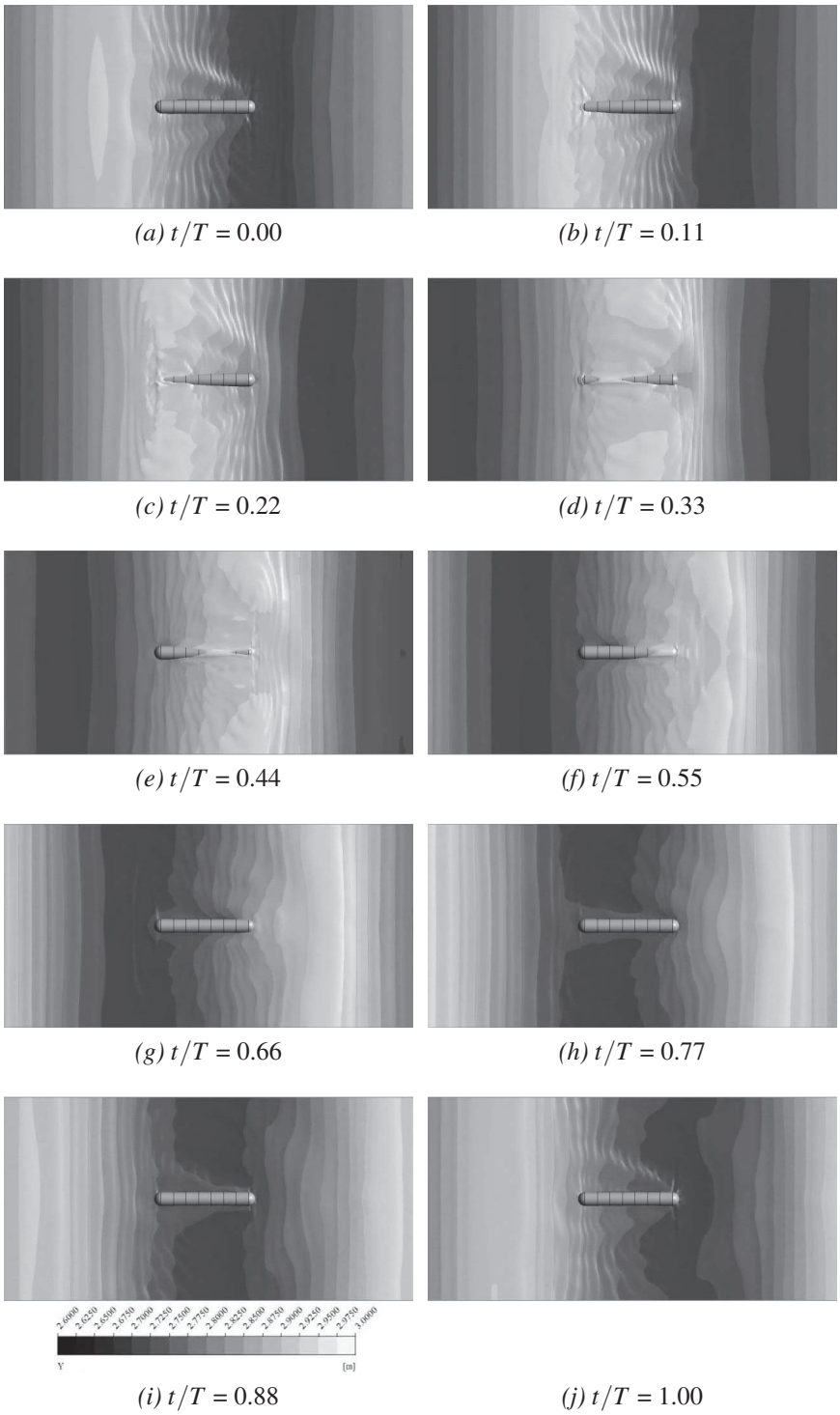
around the cylinder reduces and vanishes within the next 0.2 of a wave cycle (Figures 6.25d,e), when the crest almost submerges the device. In the vicinity of the device the wave height reduces. In Figures 6.25 f-h the wave has passed the section. Less influence on the water surface can be seen, which is due to less surface area of the device being in contact with the fluid (compare Figures 6.21f-h).

Although the solvers compute the same values for the forces on the structure, the surface elevation around the device looks more disturbed in the results calculated by the FV solver shown in Figure 6.26. Generally the same characteristics as for the CV-FE results can be identified in the plots. These are the diffraction of the waves around the Pelamis section towards the stern, the reduction of wave height, just before the wave crest hits the bow and less influence on the water surface, when the trough passes the cylinder. However, especially downstream of the device mesh effects can be observed. These do not have an influence on the calculation of the forces as both solvers give the same results within a margin of 1 %.

This effect can be illustrated by analysing the mesh around the structure, which is shown in Figure 6.27 for both solvers and the appropriate sections as shown previously in Figures 6.23 and 6.24. As described in Chapter 5.1 different meshing techniques are used for the two solvers. Both start from the same geometry file, which is generated in ANSYS ICEM CFD (ANSYS 2010). For the CV-FE solver an unstructured mesh is used. However, the procedure to generate the mesh uses a blocking approach, where the domain is subdivided into blocks. The surfaces of these blocks are mapped on the geometry. This distorts the cells around the structure, which might have slight influence on the solution of the free surface in this region. For each block the number of cells in each spatial direction has to be specified. However, as the solver requires that every cell face is neighbour to exactly

one cell face, the refinement from one block to the next has to be done in factors of three per spatial direction (see Figure 5.3). For a domain of this size the number of cells increases rapidly. For this reason the mesh is fairly uniform in cell size in the horizontal directions. The vertical resolution is higher, as the solution of the wave directly depends on this.

For the FV solver the meshing process is automatic. Here, the base cell size of the domain is specified. For the different regions, which need to be refined, finer cell sizes can be defined. This allows for very local refinement as is can be seen in Figure 6.27b, where the region around the structure and where the free surface is expected are refined. Such refinement has been investigated in the wave-wave interaction cases described in Chapter 4.1.3. In those cases however, the mesh is refined in the direction of wave propagation, which does not have an effect on the results. For this case the coarser region is in the direction of wave propagation, which influences the solution of the free surface as it can be seen in Figure 6.26. As this region is not of interest for the calculation of the forces and diffraction processes occur closer to the Pelamis section, this is acceptable.



(a) $t/T = 0.00$ *(b)* $t/T = 0.11$

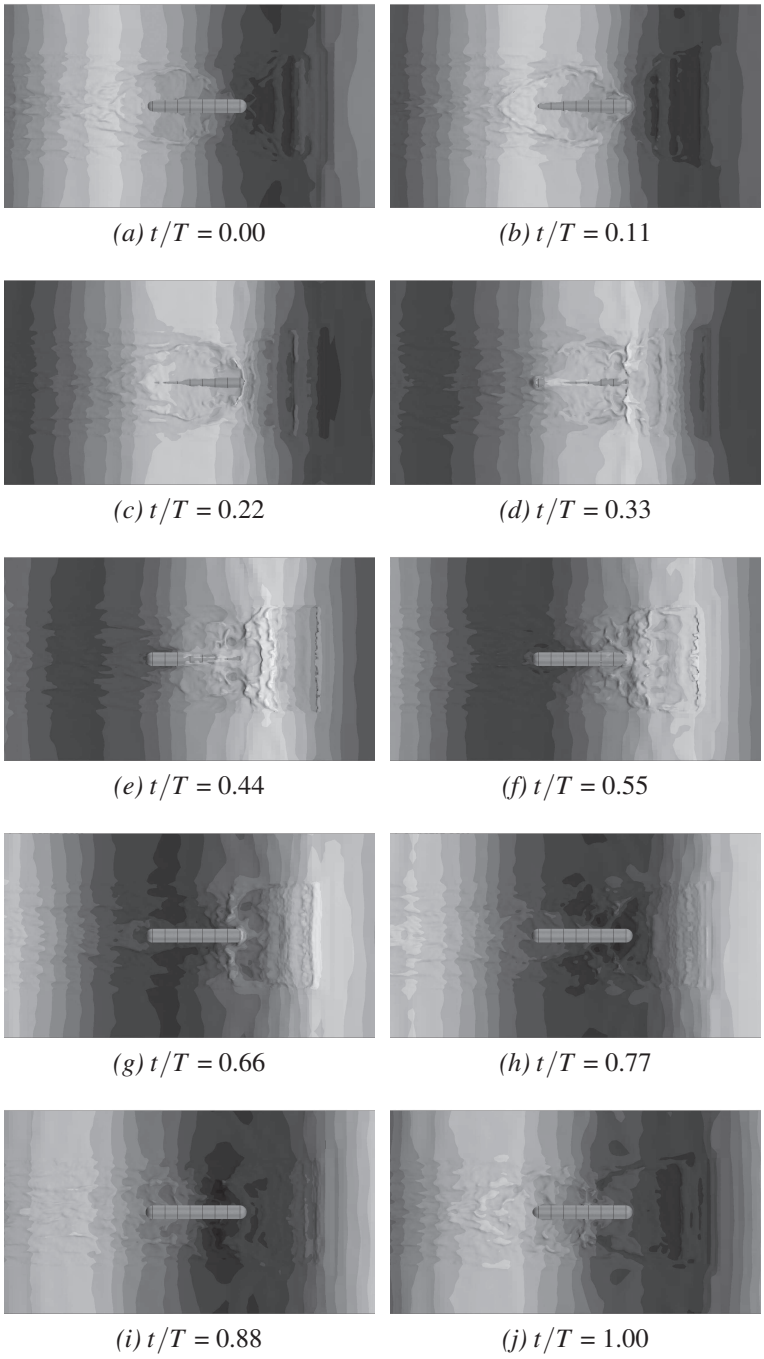
(c) $t/T = 0.22$ *(d)* $t/T = 0.33$

(e) $t/T = 0.44$ *(f)* $t/T = 0.55$

(g) $t/T = 0.66$ *(h)* $t/T = 0.77$

(i) $t/T = 0.88$ *(j)* $t/T = 1.00$

Figure 6.25: View from top on Pelamis section in regular waves (CFX)



(a) $t/T = 0.00$ (b) $t/T = 0.11$

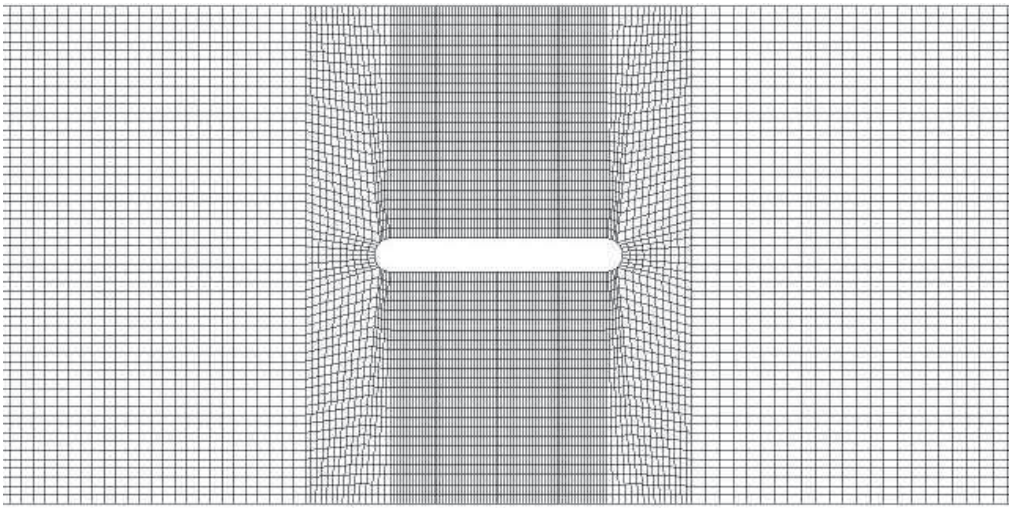
(c) $t/T = 0.22$ (d) $t/T = 0.33$

(e) $t/T = 0.44$ (f) $t/T = 0.55$

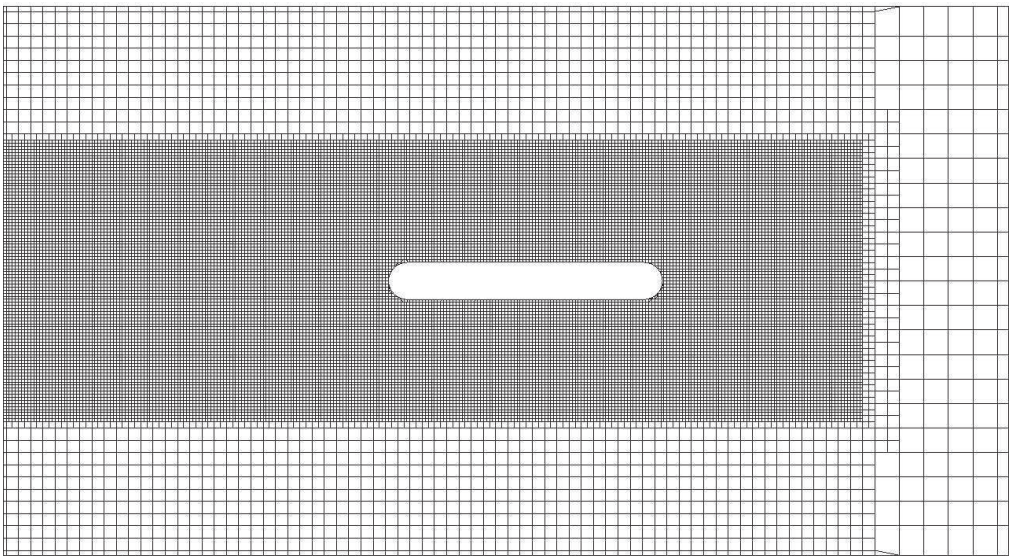
(g) $t/T = 0.66$ (h) $t/T = 0.77$

(i) $t/T = 0.88$ (j) $t/T = 1.00$

Figure 6.26: View from top on Pelamis section in regular waves (STAR CCM+)



(a) CV-FE (CFX)



(b) FV (STAR CCM+)

Figure 6.27: Mesh around single section (view from above)

Chapter 7

Conclusions

For this work wave-wave and wave-structure interaction cases leading to the simulation of two WEC related problems are discussed in the previous chapters. The aim was to investigate the use of CFD to such marine offshore problems. So, the main question is: "Is CFD capable of simulating waves and wave-structure interaction problems and especially WECs successfully?" This question will be answered within this final chapter concerning the generation wave in a NWT, its performance, the use of different types of meshes, necessary mesh resolutions and also general experiences with the two commercial software packages used for this work.

Waves in a NWT

The generation of both regular and focused waves was carried out by prescribing the surface elevation from 1st and 2nd order wave theory at the stationary velocity inlet of the domain and fluxing the appropriate horizontal and vertical velocity components of the wave into the domain. This approach can be easily extended to any higher order wave theory, for example 5th order Stokes wave, Gerstner's or Rienecker-Fenton's theory. The waves do not need to be derived for any kind of wave paddle as might be necessary for wavemakers used in physical tank testing. Neither is a transfer function needed to account for paddle motion. This would apply for a NWT, where the waves are generated using a moving boundary arrangement. The velocity inlet approach makes the setup used for this work very flexible

and also computationally less expensive than that including mesh motion, which would involve the solution of an additional equation for the boundary displacement.

By using damping layers at the end of the NWT the reflections from the far end wall boundary are minimised successfully. Here, a combination of linearly increased dynamic viscosity of the water fraction and numerical damping due to large increase in cell size was used.

It was found that the vertical mesh resolution per waveheight for a regular wave should be at least 10 cells. This value is the same for large and small waves, which makes linear waves of small amplitude relatively expensive to simulate. Here, the cells would be small and would thereby force a smaller timestep compared to the more non-linear wave simulations, which could be resolved with the same number but larger cells. When modelling wave groups, as was done for the focused wave cases, the simulation involves both rather small linear and extreme non-linear waves. Here, the number of cells needs to be related to the amplitude of the small waves, as these need to be resolved properly to superpose and form the extreme wave. The same argument would apply for the simulation of irregular waves.

Another aspect is the damping of the waves with increasing distance from the wavemaker. This was found to be different for waves of different wavelength and heights. However, at a distance behind the wavemaker of approximately 1-3 wavelengths, all waves behaved as defined and reached their prescribed height and shape. For this reason the focus point of the NewWave cases and the location of the structures was chosen to lie within this region.

Because of the

- increase in computational cost for linear waves relative to large waves

- damping of wave height and phase change with distance from the wavemaker

this CFD approach is better suited for extreme wave simulations or simulations where the region of interest is closer to the wavemaker.

Structures in a NWT

The forces on structures have been investigated by simulating fixed horizontal and vertical cylinders in regular waves. Results were compared with physical tank data published by Dixon et al. (1979), Kriebel (1998) and Chaplin et al. (1997). The hydrodynamics of the cases involved different kinds of non-linearities. Double frequency oscillations of the vertical forces on the horizontal cylinder occurred, which could not be modelled by the empirical approach presented by Dixon et al.. The vertical cylinder cases for the large cylinder were all within the diffraction regime, but could not be modelled by 1st-order diffraction theory due to the interaction of linear and higher order force components. Even 2nd order diffraction theory failed for some cases to predict the forces correctly. For a slender cylinder in steep waves however, no linear theory exists to predict the secondary load cycle, which might cause ringing of the cylinder and thereby severe damage to the structure.

The numerical results by both solvers for the forces on the fixed structures were all in good agreement regardless of the flow regime, i.e. the results were independent of the size of the structure and the waveheight and wavelength. The mesh resolution proportional to the one used for the wave-only cases worked well. The horizontal cylinder case which is fully submerged but close to the still water surface gave slightly less agreement especially for the qualitative force characteristics over one wave cycle. Here, the local waterdepth and waveheight above the cylinder is relevant for the mesh resolution. Even though, the quantitative forces were

predicted correctly and could be used for the design of such structures.

The CV-FE solver (CFX) was also applied to the cone case, where a cone shaped body was forced to move close to the free surface following a prescribed displacement. The numerical results were compared to those published by Drake et al. (2008). This was also the only case where the surface elevations close to the structure could be investigated, as in all previously described experiments either no structure was in the tank, or no measurements close to the structure were available. The forces on the structures in the CV-FE solver are the integrated pressure and shear stresses on the surface of the structure. This includes the stresses imprinted by both fluids, water and air, in which the water fraction contributes most, the stresses generated by the air fraction are negligible. Thus, the forces on the structure depend on the correct solution of the free surface at the structure. The cone results show a discrepancy in the agreement between the numerical and physical results. Whereas the surface elevations differ by a distance larger than one cell size, the forces are in good agreement. The results however, were worse for the more violent case, where the motion of cone was quicker due to the higher central frequency of the Gaussian wave packet, that was used to define the displacements of the structure.

For the improved solution of the surface elevation around the cone surface two effects are important. First, the jet-like formation for the high central frequency case and secondly the absolute difference of the displacement of the water surface and the displacement of the cone. For the case with $m = 3$ this jet-effect was not formed and the free surface was hardly influenced by the motion of the cone. Thus the relative displacement was equal to the maximum excursion of the cone and could be resolved by the number of cells. For the high frequency case, the formation of the jet close to the cone could be observed. Here, the point where the fluid separates from

the structure is very difficult to resolve, because the distance from water to solid becomes infinitely small. As the grid cells have a finite size and the surface needs to be captured within a cell, a difference in the results will occur, regardless of the mesh refinement. Furthermore, for the higher frequency case the inertia of the water acts opposite to the cone motion and thereby reduces the difference between the displacement of the cone and the water surface. So, rather than taking the absolute displacement of the cone as a reference to refine the mesh, the difference between the surface elevation and the cone displacement should be used.

The motions of a floating body, such as modelled for the single float in extreme waves using the FV solver (Chapter 6.1), are calculated from the fluid forces on the structure. This case was modelled for a tethered arrangement, where the float was allowed to move only in vertical direction, and an untethered arrangement, where also horizontal motion as allowed in the numerical setup. The computational method has been applied successfully to floating body problems (Xing et al. 2001). The difficulty for this case was the additional body force resulting from the interaction with the counterweight connected to the float by a rope-pulley system in the physical experiment. These had to be modelled in the FV solver by an external body force. This body force was simplified to the constant part of equation (6.6) by only including the gravitational part. For this setup the agreement was reasonable. Good agreement to the experimental data was achieved for the phase, where the float was riding the waves. However, after that the float in the physical experiment kept oscillating with decreasing amplitudes, whereas the numerical results showed no reduction in amplitude, even an increase at one point. For the untethered arrangement the lack of rotational motion in the numerical setup was identified to result in a phase difference for the oscillations compared to the physical experiment.

The magnitude of the oscillations, when the waves had passed the float in the CFD approach was similar to the tethered setup.

When modelling structures in a NWT,

- the simulations should be carried out in all three dimensions, even when it is a 2D problem, in which case a thin domain may be used.
- the structure should be placed within a range of 1-3 wavelengths behind the wavemaker, as further away the wave signal will be altered due to damping and closer to the wavemaker transient effects will occur, such as reflections from the structure which will alter the waves and make long steady wave simulations difficult.
- the correct solution of the surface elevations will result in good predictions of the forces on the structures.
- the displacement of the surface elevation relative to the structure (motion) should be used evaluate the mesh setting.
- the simulation of body motion due to waves is difficult and physical tank tests are necessary to validate the solvers. This is especially true for simulations where interaction with other masses is of interest, such as a counterweight.
- the effect of even small motions, such as rotational motion in the single float case, can have large impact, especially, when the relative influence of the driving force, such as the waves, vanish.

Meshes and Meshing

The quality of CFD results falls and rises with the quality of the meshes. For the simulations presented here, two different meshing procedures and arrangements were used, which were provided with the software packages. The geometry was done in the same package for both solvers and then used in the appropriate mesher to generate the computational grids consisting of hexahedral cells.

The FV solver applies a fully automated meshing procedure, which generates a grid, where the structure is cut out of the initially pure hexahedral mesh. This results in arbitrarily shaped cells close to the structure. If refinement is necessary, 3D regions in the domain are specified, where the refined mesh resolution is defined relative to the global cell size. The mesher then generates the refinement automatically. This automated meshing procedure can save a lot of time, when setting up a CFD simulation and is able to handle complex geometries, but might also be restrictive, as it cannot be influenced by the user.

The mesh used for the CF-FE solver was generated manually, by using a blocking procedure. Here, the domain is split into several blocks and their surfaces are mapped onto the geometry. The number of cells in all spacial directions needs to be specified for each of the blocks. This can make the meshing procedure very time consuming, but also very accurate, where a certain number of cells across a surface is needed.

For the numerical simulations, the flow field needs to be initialised. Especially the initial position of the interface between the water and the air fraction is of interest for marine applications. As described above, it is necessary, that the orientation of the cell vertices is normal and parallel to the water surface, which is the reason

for using hexahedral meshes. However, due to the use of the body-fitted meshes the initial surface can be affected by distorted cells around structures that are close or at the free surface. The same effects can occur when mesh motion is involved, where the domain is stationary and only the nodes and vertices in the domain are moved according to the rigid body motion.

The computational time can be reduced significantly by using well engineered meshes. Assuming the same results are computed, it might be less expensive to use a finer grid and smaller timesteps that converge quicker within a timestep, than using a large cell size with a larger timestep that needs more inner iterations to achieve convergence. The simulation time might also be reduced by making the "empty" domain smaller. That is the region, where only air is modelled that is far away from the free surface and thereby not needed to capture any non-linear effects such as splashing or mixture of the fluids.

Computational cost

The computational cost of CFD directly depends on the size of the problem. The larger the mesh and the smaller the timestep the longer the run times will be. Both software packages can be run in parallel rather than using a single processor for the whole problem. This is done by partitioning the mesh into smaller fractions equal to the the number of CPUs used for the run. By doing that, the load on each CPU is reduced and results are calculated quicker. For this work no direct comparison of the codes and the performance for certain problems and computing hardware was done. The reason for this is that over the time different computers were used. These were:

- Dell Optiplex 745, Intel Pentium Duo Core 2.4 GHz, 2 Gb RAM, Windows

XP 32 bit

- Viglen, Intel Xeon Duo Quad Core 2.66 GHz, 16 Gb RAM, Windows XP 64 bit

- HPCx Service, IBM Power5 RISC processor, 1.5 GHz, 16 Processors and 32 Gb RAM per frame/node, 160 nodes in total

The wave-only cases were mainly done on the Dell desktop. Here, the run time for a 12 s simulation on 2 CPUs was approximately 5 days. The cone case was entirely done on the HPCx cluster on 16 partitions. The run time for a 2 s piece of the simulation was approximately 28 hours. The single float in extreme waves were done on the Viglen workstation on 8 CPUs. The run time for such a simulation was approximately 1 week for 8 s.

The two solvers have different hardware requirements though. The segregated iterative FV solver (STAR CCM+) empties the memory within the iteration process as described for the SIMPLE algorithm in Chapter 3.2, whereas the coupled CV-FE solver (CFX) writes the full set of linear equations in the memory and solves it at once. The segregated iterative solver thereby performs better, on machines with higher computing frequency of the processor, whereas the coupled solver requires sufficient memory. Furthermore, when using the solvers in parallel, a speedy communication between the CPUs is more important for the segregated iterative solver than for the coupled one, as each of the numerical results need to be communicated more often for the solution process of the first one within a timestep than for the coupled solver. For this reason the segregated iterative solver performed better on the workstation than on the HPCx cluster on the same number of CPUs, as the frequency of the motherboard was higher here.

The meshing process for both solvers however is serial. For large meshes, i.e. meshes with many nodes and cells, a large memory AND a capable processing frequency of the computer is necessary, when generating a single mesh for a simulation. This problem could be overcome by combining several small meshes, which was not done for this work, as the number of cells was sufficient for the problems calculated here.

CFD for WEC applications

Finally, it can be concluded that CFD is capable of simulating marine problems very well. The results give confidence that fixed body problems can be simulated straight away, rigid body motion, however, should be validated against physical test data. Generally, CFD can be used to do long time simulations, linear wave studies, extreme waves simulations etc. The only issue one has to deal with is the computational cost. If there is enough computing power, CFD could even be used for long time irregular wave simulations. When modelling WECs the place of CFD within the range of numerical-mathematical techniques is clearly located where extreme wave conditions with non-linear waves and wave breaking are concerned, such as the survivability studies considered here.

Future Work

The open questions from this work concern the interaction between the floating body and the waves. First the phase shift identified in the displacement time-histories of the untethered single float in extreme waves should be investigated. This is likely to be resolved by including rotational motion in the numerical model.

The next issue to resolve is the overestimation of the translation of the single float in extreme waves when the full body force is included in the numerical model.

Generally this work could be extended by investigating the effects of compressible flow, which might effect slamming forces on the structure and also aeration of the water due to breaking waves around the WEC. The air-water mixture has another density and might affect the buoyancy of the device and thereby the hydrodynamics.

Furthermore the numerical models could be developed or extended. For example, moorings could be included in the CFD code. Currently that can be included by using body forces as it is done for the representation of the pulley system in the single float (Figure 6.3b and 6.9c) simulations. To model the mooring directly, the codes could be coupled with structural solvers to compute the stresses and deformation on the mooring line and also on the device itself. This might also be a method to simulate the highly complex motion of the Pelamis WEC, which is currently not possible with the methods used here.

To save computational cost, especially while the wave approaches the structure, the CFD models could be coupled with less expensive simpler flow models, such as Boundary Element Methods and potential flow solvers, up to the point where nonlinear effects, such as wave breaking, require the use of more complex methods.

Nomenclature

α	Angle between initial and displaced pulley cable
β	Blending factor
$\ddot{z}$	vertical acceleration[m/s^2]
$\dot{z}$	vertical velocity [m/s]
η	Surface elevation [m]
Γ	Diffusion coefficient
λ	Wavelength [m]
$\mathscr{A}$	*CV* face area
$\mathscr{V}$	*CV* volume
μ	Dynamic viscosity [Pa s]
ω	circular/wave frequency [1/s]
Φ	Velocity potential
ϕ	Fluid property
$\boldsymbol{a}$	Arbitrary vector and vector normal to fluid interface
$\boldsymbol{i}$	Cartesian unit vector
$\boldsymbol{n}$	Vector normal to surface
$\boldsymbol{u}$	Velocity vector $[u, v, w]$
ρ	Density [kg/m3]

τ_{ij}	Viscous stress component, acting in j-direction and normal to a surface in i-direction
θ	Angle between normal vector of fluid interface and $\boldsymbol{n}$
A	Wave amplitude [m]
bip	boundary integration point
c	Volume fraction
C_D	Coefficient of drag
c_f	Face value of volume fraction, with˜normalised, with * and ** corrected
C_M	Coefficient of inertia
Co	Courant number
CV	Control volume [m3]
D	Diameter of cylinder [m]
d	Displacement [m]
D_b	Diffusive boundary flux
f	Face
$F'_{x,y,z}$	Relative force in x, y or z direction
F_{CV}	convective flux across a CV
$F_{x,y,z}$	Force in x, y or z direction [N]
g	Acceleration due to gravity [m/s^2]
H	Wave height [m]
h	Water depth [m]
h_p	height of pulley above still water level

h_{local}	Local water depth [m]
ip	integration point
k	Wave number [1/m]
l	Cylinder length [m]
m_c	mass of counterweight [kg]
m_f	mass of float [kg]
N	Number of wave/frequency components
N_i	shape function
N_{Fr}	Froude number
N_{KC}	Keulegan-Carpenter number
nb	neighbour
p	Pressure [Pa]
R	Upwind vertex vector
r,s,t	Local Cartesian coordinates
S_ϕ	Source of fluid property ϕ
S_{M_i}	Source of fluid Momentum in i-direction
T	Wave period [s]
t	Time [s]
T'	Relative wave period
T_h, T_v	horizontal and vertical components of tension force T
u,v,w	Cartesian Velocity Components

x, y, z Cartesian coordinates

CD Central Differencing

CFD Computational Fluid Dynamics

CICSAM Compressive Interface Capturing Scheme for Arbitrary Meshes

CV Control volume

CV-FE Control-Volume Finite Element Method

DD Downwind Differencing

DOF Degrees of freedom

FD, FDM Finite Difference Method

FE, FEM Finite Element Method

FV, FVM Finite Volume Method

HRIC High Resolution Interface Capturing

LUD 2nd order upwind Differencing

NSE Navier-Stokes equations

NVD Normalised Variable Diagram

PISO Pressure Implicit with Splitting of Operators algorithm

PTO Power take off

RANS Reynolds-Averaged-Navier-Stokes equations

SIMPLE(R)(C) Semi-Implicit Method for Pressure-Linked Equations (Revised and Corrected)

SPH Smoothed Particle Hydrodynamics

UD Upwind Differencing

VOF Volume of Fluid

WEC Wave energy converter

Index

List of References.

Agamloh, E. B., Wallace, A. K. & von Jouanne, A. (2008), 'Application of fluid-structure interaction simulation of an ocean wave energy extraction device', *Renewable Energy* **33**(4), 748–757.

ANSYS (2006), *ANSYS CFX-Solver Theory Guide*, ANSYS CFX - Release 11.0, Canonsburg, USA.

ANSYS (2010), *ANSYS ICEM CFD User Manual*, ANSYS CFX - Release 12.0, Canonsburg, USA.

Baliga, B. R. & Patankar, S. V. (1980), 'A new finite-element formulation for convection-diffusion problems', *Numerical Heat Transfer, Part A* **3**, 393–409.

Baliga, B. R. & Patankar, S. V. (1983), 'A control volume finite-element method for two-dimensional fluid flow and heat transfer', *Numerical Heat Transfer, Part A* **6**, 245–261.

Barth, J. T. & Jesperson, D. C. (1989), The design and application of upwind schemes on unstructured meshes, *in* 'AIAA', Reno/Nevada.

Bingham, H. B. (2000), 'A hybrid Boussinesq-panel method for predicting the motion of a moored ship', *Coastal Engineering* **40**(1), 21–38.

CD-Adapco (2009), *STAR CCM+ Version 4.04.011*, User Guide, London, UK.

Chaplin, J. R., Rainey, R. C. T. & Yemm, R. W. (1997), 'Ringing of a vertical cylinder in waves', *Journal of Fluid Mechanics* **350**, 119–147.

Clauss, G. F., Schmitter, C. E. & Hennig, J. (2004), Modelling extreme wave sequences for the hydrodynamic analysis of ships and offshore structures.

Clauss, G. F., Schmitter, C. E. & Stück., R. (2005), Numerical wave tank simulation of extreme waves for the investigation of structural responses, *in* 'Proceedings of

OMAE 2005 24th International Conference on Offshore Mechanics and Arctic Engineering', Halkidiki, Greece,.

Clement, A., McCullen, P., Falcao, A., Gardner, A. F. F., Hammerlund, K., Lemonis, G., Lewis, T., Nielsen, K., Petroncini, S., Pontes, M.-T., Schild, P., Sjöström, B.-O., Sorensen, C. & Thorpe, T. (2002), 'Wave energy in europe; Current status and perspectives', *Renewable and Sustainable Energy Reviews* **6**, 405–431.

Craik, A. D. D. (2004), 'The origins of waterwave theory', *Annu. Rev. Fluid. Mech.* **36**, 1–28.

Cruz, J. (2008), *Ocean Wave Energy*, Springer, Berlin.

Cruz, J. M. B. P. & Salter, S. H. (2006), 'Numerical and experimental modelling of a modified version of the Edinburgh Duck wave energy device', *Proc. IMechE Part M: J. Engineering for the Maritime Environment* **220**, 129–147.

Daly, B. J. (1969), 'A technique for including surface tension effects in hydrodynamic calculations', *Journal of Computational Physics* **4**(1), 97–117.

Dalzell, J. F. (1999), 'A note on finite depth second-order wave-wave interactions', *Applied Ocean Research* **21**, 105–111.

Darwish, M. & Moukalled, F. (2006), 'Convective schemes for capturing interfaces of free-surface flows on unstructured grids', *Numerical Heat Transfer, Part B* **49**, 23.

Demirdžić, I. & Perić, M. (1990), 'Finite volume method for prediction of fluid flow in arbitrarily shaped domains with movin boundaries', *International Journal for Numerical Methods in Fluids* **10**, 771–790.

Dixon, A. G., Greated, C. A. & Salter, S. H. (1979), 'Wave forces on partially submerged cylinders', *Journal of the Waterway, Port, Coastal and Ocean Devision* **105**(4), 421–438.

Drake, K., Eatock Taylor, R., Taylor, P. H. & Bai, W. (2008), 'On the hydrodynamics of bobbing cones', *submitted for publication* .

EMEC (2010), 'http://www.emec.org.uk'.

Faltinsen, O. M., Newman, J. N. & Vinje, T. (1995), 'Nonlinear wave loads on slender cylinders', *Journal of Fluid Mechanics* **289**, 179–198.

Farley, F. J. M. & Rainey, R. C. T. (2006), Radical design options for wave-profiling wave energy converters, *in* '21st International Workshop on Water Waves and Floating Bodies, 2nd-5th April 2006', Loughborough, UK.

Fenton, J. D. (1985), 'A fifth-order Stokes theory for steady waves', *Journal of Waterway, Port, Coastal and Ocean Engineering* **111**(2).

Ferziger, J. H. & Perić, M. (2001), *Computational Methods for Fluid Dynamics*, 3rd edn, Springer, Heidelberg.

Greenhow, M., Vinje, T., Brevig, P. & Taylor, J. (1982), 'A theoretical and experimental study of the capsize of Salter's duck in extreme waves', *Journal of Fluid Mechanics* **118**, 221–239.

Grue, J., Bjørshol, G. & Strand, Ø. (1994), Nonlinear wave loads which may generate "ringing" responses of offshore structures, *in* '9th International Wokshop for Water Waves and Floating Bodies', 17-20 April, Kuju, Fukuoka, Japan.

Grue, J. & Huseby, M. (2002), 'Higher-harmonic wave forces and ringing of vertical cylinders', *Applied Ocean Research* **24**, 203–214.

Harlow, F. H. & Welch, J. E. (1965), 'Numerical calculation of time - dependent viscous incompressible flow of fluid with free surface', *The Physics of Fluids* **8**(12), 2182–2189.

Hasselman, K., Barnett, T. P., Bouws, E., Carlson, D. E. & Hasselmann, P. (1973), 'Measurements of wind-wave growth and swell decay during the joint north sea wave project (JONSWAP)', *Deutsche Hydrographische Zeitschrift* **8**(12).

Henderson, R. (2006), 'Design, simulation and testing of a novel hydraulic power take-off system for the Pelamis wave energy converter', *Renewable Energy* **31**, 271–283.

Hirt, C. W. & Nichols, B. D. (1981), 'Volume of Fluid (VoF) method for the dynamics of free boundaries', *Journal of Computational Physics* **39**, 201–225.

Hogben, N., Miller, B. L., Searle, J. W. & Ward, G. (1977), Estimation of fluid loading on offshore structures, *in* 'Institute of Civil Engineering, Part 2', Vol. 63, Institute of Civil Engineering.

Hong, D. C., Hong, S. Y. & Hong, S. W. (2004), 'Numerical study of the motions and drift force of a floating OWC device', *Ocean Engineering* **31**(2), 139–164.

Huseby, M. & Grue, J. (2000), 'An experimental investigation of higher-harmonic wave forces on a vertical cylinder', *Journal of Fluid Mechanics* **414**, 75–103.

Hutchinson, B. R. & Raithby, G. D. (1986), 'A multigrid method based on the additive correction strategy', *Numerical Heat Transfer* **9**, 511–537.

Issa, R. I. (1986), 'Solution of the implicitly discretised fluid flow equations by operator-splitting', *Journal of Computational Physics* **62**(1), 40–65.

Johannessen, T. B. & Swan, C. (1997), 'Nonlinear transient water waves - Part 1. a numerical method of computation with comparisons to 2-d laboratory data', *Applied Ocean Research* **19**, 15.

Keulegan, G. H. & Carpenter, L. H. (1958), 'Forces on cylinders and plates in an oscillating fluid', *Journal of Research of the National Bureau of Standards* **60**(5), 423–440.

Kleefsman, K. M. T. (2005), Water Impact Loading on Offshore Structures - A Numerical Study, Phd, Rijksuniversiteit Groningen.

Kleefsman, K. M. T.; Veldman, A. E. P. (2003), Numerical simulation of wave loading on a spar platform, *in* '18th International Workshop on Water Waves and Floating Bodies', 6.-9. April, Le Croisic/France.

Kriebel, D. L. (1998), 'Nonlinear 'wave interaction with a vertical circular cylinder: Wave forces', *Ocean Engineering* **25**(7), 597–605.

Leonard, B. P. (1991), 'The ULTIMATE conservative difference scheme applied to unsteady one-dimensional advection', *Computer Methodes in Applied Mechanics and Engineering* **88**, 17–74.

Liu, Y., Xue, M. & Yue, D. K. P. (2001), 'Computations of fully nonlinear three-dimensional wave-wave and wave-body interactions. part 2. nonlinear waves and forces on a body', *Journal of Fluid Mechanics* **438**, 41–66.

Longuet-Higgins, M. S. (1977), The mean forces exerted by waves on floating or submerged bodies with application to sand bars and wave power machines, *in* 'Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences', Vol. 352, pp. 463–480.

Luck, M. & Benoit, M. (2004), Wave loading on monopile foundation for offshore wind turbines in shallow-water areas, *in* 'ICCE'.

MacCamy, R. C. & Fuchs, R. A. (1954), 'Wave forces on piles: A diffraction theory', *U.S. Army Corps of Engineers, Beach Erosion Board* **Tech Memo No. 69**.

Malencia, S. & Molin, B. (1995), 'Third-harmonic wave diffraction by a vertical cylinder', *Journal of Fluid Mechanics* **302**, 203–229.

Mathur, S. R. & Murthy, J. Y. (1997*a*), 'Pressure-based method for unstructured meshes', *Numerical Heat Transfer, Part B* **31**(2), 195–214.

Mathur, S. R. & Murthy, J. Y. (1997*b*), 'Pressure boundary conditions for incompressible flow using unstructured meshes', *Numerical Heat Transfer, Part B* **32**(3), 283–298.

Millar, D. L., Smith, H. C. M. & Reeve, D. E. (2007), 'Modelling analysis of sensitivity of shoreline change to a wave farm', *Ocean Engineering* **34**, 884–901.

Monaghan, J. J. (2005), 'Smoothed particle hydrodynamics', *Reports on Progress in Physics* **68**(8), 1703–1759.

Morison, J. R., O'Brien, M. P., Johnson, J. W. & Shaaf, S. A. (1950), 'The force exerted by surface waves on piles', *Petroleum Transactions, American Institute of Mining and Petroleum Engineers* **189**, 149–154.

Muzaferija, S. & Perić, M. (1999), Computation of free-surface flows using interface-tracking and interface-capturing methods, *in* M. Mahrenholtz, O.; Markiewicz, ed., 'Nonlinear Water Wave Interaction', Vol. 24, WIT Press, Hamburg, p. 252.

Newman, J. N. & Lee, C.-H. (2002), 'Boundary-element methods in offshore structure analysis', *Journal of Offshore Mechanics and Arctic Engineering* **124**, 81–89.

Ning, D. Z., Zang, J., Eatock Taylor, R., Teng, B. & Taylor, P. H. (2007), 'An experimental and numerical study of nonlinear focused wave groups', p. 29.

Patankar, S. V. (1980), *Numerical Heat Transfer and Fluid Flow*, Series in Computational and Physical Processes in Mechanics and Thermal Sciences, 1st edn, Taylor&Francis.

Patankar, S. V. & Spalding, D. B. (1972), 'A calculation procedure for heat, mass and momentum tranfer in three-dimensional parabolic flows', *International Journal of Heat and Mass Transfer* **15**, 1787–1806.

PelamisWavePower (2010), 'http://www.pelamiswave.com/'.

Pierson, W. J. & Moskowitz, L. (1964), 'A proposed spectral form for fully developed wind seas based on the similarity theory of S. A. Kitaigorodskii', *J. Geophys. Res.* **69**.

Pizer, D. J., Retzler, C., Henderson, R. M., Cowieson, F. L., Shaw, M. G., Dickens, B. & Hart, R. (2005), Pelamis WEC - recent advances in the numerical and experimental modelling programme., *in* '6th European Wave and Tidal Energy Conference, August 29th - September 2nd 2005', Glasgow, UK.

Rahman, M. (1994), *Water Waves - Relating Modern Theory to Advanced Engineering Practice*, IMA Monograph Series, Oxford University Press Inc., Oxford.

Rainey, R. C. T. & Chaplin, J. R. (2003), Wave breaking and cavitation around a vertical cylinder: Experiments and linear theory, *in* '18th International Workshop on Water Waves and Floating Bodies', 6.-9. April, Le Croisic, France.

Raithby, G. D. (1991), 'Equations of motion for reacting, particle-laden flows', *Progress report, Thermal Science Ltd., provided to EMR* .

Rapp, R. J. & Melville, W. K. (1990), 'Laboratory measurements of deep-water breaking waves', *Phil. Trans. R. Soc. London A* **331**, 735–800.

Raw, M. (1985), A new control-volume-based element procedure for the numerical solution of the fluid flow and scalar transport equations, PhD thesis, University of Waterloo, Ontario, Canada.

Raw, M. (1996), Robustness of coupled algebraic multigrid for the navier-stokes equations, *in* '34th Aerospace Sciences Meeting and Exhibit', AIAA 96-0297, Reno, NV, USA, 15.-18. January.

Retzler, C. H., Chaplin, J. R. & Rainey, C. T. (2000), Transient motion of a vertical cylinder: Measurements and computations of the free surface, *in* '15th International Workshop on Water Waves and Floating Bodies', 27 Feb-1 March, Caesarea, Israel.

Retzler, C., Pizer, D., Henderson, R., Ahlqvist, J., Cowieson, F. & Shaw, M. (2003), Pelamis: Advances in the numerical and experimental modelling programme, *in* '5th European wave energy conference', Cork, Ireland.

Rhee, S. H., Makarov, B. P., Krishinanm, H. & Ivanov, V. (2005), 'Assessment of the Volume of Fluid method for free-surface wave flow', *Journal of Marine Science and Technology* **10**, 7.

Rhie, C. M. & Chow, W. L. (1982), A numerical study of the turbulent flow past an isolated airfoil with trailing edge separation, *in* 'AIAA/ASME 3rd Joint Thermophysics, Fluids, Plasma and Heat Transfer Conference', St. Louis, Missouri.

Rienecker, M. M. & Fenton, J. D. (1981), 'A Fourier approximation method for steady water waves', *Journal of Fluid Mechanics* **104**, 119–137.

Salter, S. H. (1974), 'Wave power', *Nature* **249**, 720–724.

Sarpkaya, T. & Issacson, M. (1981), *Mechanics of Wave Forces on Offshore Structures*, Van Nostrand Reinhold.

Schäffer, H. A. (1996), 'Second-order wavemaker theory for irregular waves', *Ocean Engineering* **23**(1), 47–88.

Schneider, G. E. & Raw, M. J. (1987*a*), 'Control volume finite-element method for heat fransfer and fluid flow using colocated variables - 1. Computational procedure', *Numerical Heat Transfer* **11**, 363–390.

Schneider, G. E. & Raw, M. J. (1987*b*), 'Control volume finite-element method for heat fransfer and fluid flow using colocated variables - 2. Application and validation', *Numerical Heat Transfer* **11**, 391–400.

Siegel, R. & Howell, J. R. (1981), *Thermal Radiation Heat Transfer*.

Soerensen, H. C., Friis-Madsen, E., Panhauser, W., Dunce, D., Nedkvintne, J., Frigaard, P., Kofoed, J. P., Knapp, W., Riemann, S., Holmén, E., Raulund, A., Praest, J., Hansen, L. K., Christensen, L., Nöhrlind, T., Bree, T. & McCullen, P. (2003), Development of wave dragon from scale 1:50 to prototype, *in* 'Fifth European Wave Energy Conference', Cork Ireland.

SPERBOY (2010), 'http://www.sperboy.com'.

Stallard, T. (2010), 'Personal communication'.

Stallard, T., Stansby, P. K. & Williamson, A. J. (2008), An experimental study of closely spaced point absorber arrays, *in* 'Proceedings of 18th (2008) International Offshore and Offshore and Polar Engineering Conference', Vancouver, Canada.

Stallard, T., Weller, S. D. & Stansby, P. K. (2009), 'Limiting heave response of a wave energy device by draft adjustment with upper surface immersion', *Applied Ocean Research* **Article in Press**.

Stansberg, C. T. (1997), Comparing ringing loads from experiments with cylinders of different diameters - an empirical study, *in* '8th Conference on the Behaviour of Offshore Structures (BOSS'97)', Vol. 2, Pergamon/Elsevier, pp. 95–109.

Stansby, W. & Jenkins (2006), 'Wave energy apparatus'.

Taylor, P. H. (1992), On the kinematics of large ocean waves, Technical report, Koninklijke/Shell Exploratie en Produktie Laboratorium.

Taylor, P. H. & Williams, B. A. (2002), Wave statistics for intermediate depth water - newwaves and symmetry, *in* '21st International Conference on Offshore Mechanics and Arctic Engineering', 23-28. June, Oslo, Norway.

Taylor, P. H. & Williams, B. A. (2004), 'Wave statistics for intermediate depth water-newwaves and symmetry', *Journal of Offshore Mechanics and Arctic Engineering* **126**, 6.

Tromans, P. S., Anaturk, A. & Hagemeijer, P. (1991), 'A new model for the kinematics of large ocean waves - Application as a design wave'.

Ubbink, O. (1997), Numerical prediction of two fluid systems with sharp interfaces, PhD thesis, Imperial College of Science, Technology & Medicine.
eatock Taylor et al.

Eatock Taylor, R., Taylor, P. H. & Drake, K. R. (2009), Tank wall reflections in transient testing, *in* 'Proceedings of 24. International Workshop on Water Waves and Floating Bodies', St. Petersburg, Russia.

Paixão Conde, J. M. & Gato, L. M. C. (2008), 'Numerical study of the airflow in an oscillating water column wave energy converter', *Renewable Energy* **33**(12), 2637–2644.

Van Doormaal, J. P. & Raithby, G. D. (1984), 'Enhancements of the SIMPLE method for predicting incompressible fluid flows', *Numerical Heat Transfer, Part B* **7**(2), 147 – 163.

v. d. Molen, W. & Wenneker, I. (2008), 'Time-domain calculation of moored ship motions in nonlinear waves', *Coastal Engineering* **55**, 409–422.

Venkatakrishnan, V. (1993), 'On the accuracy of limiters and convergence to steady state solutions', *AAIA* **93-0880**.

Versteeg, H. & Malalasekera, W. (2007), *An Introduction to Computational Fluid Dynamics: The Finite Volume Method*, 2nd edn, Prentice Hall.

Walker, D. A. G., Taylor, P. H. & Eatock Taylor, R. (2004), 'The shape of large surface waves on the open sea and the Draupner New Year wave', *Applied Ocean Research* **26**, 10.

WaveHub (2010), 'http://www.wavehub.co.uk'.

Westphalen, J., Greaves, D. M. & Williams, C. J. K. (2007), Comparison of free surface wave simulations using STAR CCM+ and CFX, *in* 'Numerical Towing Tank Symposium', Hamburg, Germany 23.-25. Sep.

Westphalen, J., Greaves, D. M., Williams, C. J. K., Zang, J. & Taylor, P. H. (2008), Numerical simulation of extreme free surface waves, *in* 'Proceeding of 18th International Offshore and Polar Engineering Conference', Vancouver, Canada.

Widden, M. B., French, M. J. & Aggidis, G. A. (2008), 'Analysis of a pitching-and-surging wave-energy converter that reacts against an internal mass, when operating in regular sinusoidal waves', *Proc. IMechE Part M: J. Engineering for the Maritime Environment* **222**, 153–161.

Wu, G. X. & Eatock Taylor, R. (2001), The coupled finite element and boundary element analysis of nonlinear interaction between waves and bodies, *in* '16th International Workshop on Water Waves and Floating Bodies', 22.-25. Apr. ,Hiroshima, Japan.

Xing, Y., Hadžić, I., Muzaferija, S. & Perić, M. (2001), Cfd simulation of flow-induced floating-body motions, *in* '16th International Workshop on Water Waves and Floating Bodies', 22.-25. April, Hiroshima, Japan.

Zang, J., Gibson, R., Taylor, P. H., Eatock Taylor, R. & Swan, C. (2006), 'Second order wave diffraction around a fixed ship - shaped body in unidirectional steep waves', *Journal of Offshore Mechanics and Arctic Engineering* **128**, 10.

Zwart, P. J. (2005), Numerical modelling of free surface and cavitating flows, Technical report, Ansys Canada Ltd.

Zwart, P. J., Raithby, G. D. & Raw, M. J. (1998), 'An integrated space-time finite-volume method for moving-boundary problems', *Numerical Heat Transfer, Part B* **34**, 257–270.

Zwart, P. J., Raithby, G. D. & Raw, M. J. (1999), 'The integrated space-time finite volume method and its application to moving boundary problems', *Journal of Computational Physics* **154**, 497–519.

Zwart, P. J., Scheuerer, M. & Bogner, M. (2003), Free surface modelling of an impinging jet, *in* 'ASTAR International Workshop on Advanced Numerical Methods for Multidimensional Simulation of Two-Phase Flow', Garching, Germany.

Appendix A

Java Macro used for the FV solver

```
1  // STAR-CCM+ macro
2    package macro;
3
4    import java.util.*;
5    import java.math.*;
6    import java.lang.*;
7    import java.io.*;    //  first and second order components, with surface elevation
8    import star.common.*;
9    import star.vis.*;
10   import star.base.neo.*;
11   import star.flow.*;
12
13
14 public class case2new extends StarMacro {
15
16   public void execute() {
17
18     //Set siulatin properties
19     Simulation simulation_0 = getActiveSimulation();
20   Solution solution_0 = simulation_0.getSolution();
21   solution_0.initializeSolution();
22
23     //Create fiel function for horizontal velocity
24     UserFieldFunction userFieldFunction_0 = simulation_0.getFieldFunctionManager().createFieldFunction();
25     userFieldFunction_0.setPresentationName("uVel");
26     userFieldFunction_0.setFunctionName("uVel");
27     userFieldFunction_0.setDimensionsVector(new IntVector(new int[] {0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 0}));
28
29   //Create fiel function for horizontal velocity
30     UserFieldFunction userFieldFunction_1 = simulation_0.getFieldFunctionManager().createFieldFunction();
31     userFieldFunction_1.setPresentationName("wVel");
32     userFieldFunction_1.setFunctionName("wVel");
33     userFieldFunction_1.setDimensionsVector(new IntVector(new int[] {0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 0}));
34
35   //Create fiel function for surface elevation
36     UserFieldFunction userFieldFunction_2 = simulation_0.getFieldFunctionManager().createFieldFunction();
37     userFieldFunction_2.setPresentationName("surf");
38     userFieldFunction_2.setFunctionName("surf");
39     userFieldFunction_2.setDimensionsVector(new IntVector(new int[] {0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
          0, 0, 0}));
40
41   //Create fiel function for surface elevation
42     UserFieldFunction userFieldFunction_3 = simulation_0.getFieldFunctionManager().createFieldFunction();
43     userFieldFunction_3.setPresentationName("tstar");
44     userFieldFunction_3.setFunctionName("tstar");
45     userFieldFunction_3.setDimensionsVector(new IntVector(new int[] {0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
          0, 0, 0}));
46
47     //Define stopping criteria (read from *.sim file)
48     StepStoppingCriterion stepStoppingCriterion_0 = ((StepStoppingCriterion) simulation_0.
          getSolverStoppingCriterionManager().getSolverStoppingCriterion("Maximum Steps"));
49
50     for (int p = 0; p <= stepStoppingCriterion_0.getMaximumNumberSteps(); p++)
51     {
```

```
52          //Step for one iteration
53          simulation_0.getSimulationIterator().step(1);
54      int ts = p;
55
56  //!!!!!!!!!!!!!Calculation of wave signal!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
57
58      final double  g = 9.81,      // acceleration due to gravity
59               pi  = Math.PI ;    // Pi = 3.14
60      String filename   = "export.txt";
61
62  //------------------------------------------------------ wave/tank parameters
63    int nn = 16;                          // Number of waves
64    double h = 0.5,                       // waterlevel [m]
65           k2,                            // only used for iteration of k
66           T,                             // Period [s]
67           L;                             // wavelength [m]
68    double[] k = new double[nn];
69    double alpha = 0.0632;
70    double del_om;                        // delta omega
71    double ommin  = 0.6 ,                 // omega minimum value
72      om_range  = 0.7 ;                   // omega range
73    double[] omega  = new double[nn];     // frequency
74    double[] s_om = new double[nn];       // spectral density
75    double x_foc  = 3.0;                  // focal point
76    double t_foc  = 9.2;                  // focal time
77    double timestep = 0.001;              // timestep
78    double t_star = timestep*(simulation_0.getSimulationIterator().getCurrentTimeLevel());
79    double x_star = 0.0;                  // position of wavemaker
80    String z_star = "$$Position[1]";      // vertical elevation ($$Position[1] for y pointing in the sky,
          $$Position[2] for z)
81    double h_star = 0.5;                  // waterlevel
82    double z_ausg = 0.5;                  // to move coordinate system to bottom of wavetank (z_ausgleich),
          equals h_star!!!
83
84  //-------------------------Definition of wave spectrum
85
86    omega[0] = 0.6*2*Math.PI;
87    omega[1] = 0.646666667*2*Math.PI;
88    omega[2] = 0.693333333*2*Math.PI;
89    omega[3] = 0.7400001*2*Math.PI;
90    omega[4] = 0.786666667*2*Math.PI;
91    omega[5] = 0.833333333*2*Math.PI;
92    omega[6] = 0.880000*2*Math.PI;
93    omega[7] = 0.926666667*2*Math.PI;
94    omega[8] = 0.973333333*2*Math.PI;
95    omega[9] = 1.020000*2*Math.PI;
96    omega[10] = 1.066666667*2*Math.PI;
97    omega[11] = 1.113333333*2*Math.PI;
98    omega[12] = 1.1600000001*2*Math.PI;
99    omega[13] = 1.206666667*2*Math.PI;
100   omega[14] = 1.253333333*2*Math.PI;
101   omega[15] = 1.30000*2*Math.PI;
102
103   s_om[0] = 0.3841;
104   s_om[1] = 1.134;
105   s_om[2] = 1.956;
106   s_om[3] = 3.5676;
107   s_om[4] = 5.8257;
108   s_om[5] = 7.1472;
109   s_om[6] = 5.0286;
110   s_om[7] = 2.4675;
111   s_om[8] = 1.6596;
112   s_om[9] = 1.7138;
113   s_om[10] = 0.83;
114   s_om[11] = 0.4604;
115   s_om[12] = 0.2302;
116   s_om[13] = 0.1375;
```

```
117    s_om[14] = 0.0696;
118    s_om[15] = 0.0054;
119
120  del_om = (omega[(nn-1)]-omega[0])/(nn-1);
121  om_range = omega[(nn-1)]-omega[0];
122  ommin = omega[0];
123  //------------------------- calculation of k-----------------------
124         double k1 = 1.0;
125  for (int j = 0; j < nn; j++ )
126  {
127    for (int q=1; q<2000; q++)
128     {
129       k2 = omega[j]*omega[j]/(g*Math.tanh(k1*h));
130       k1=k2;
131     }
132        k[j] = k1;
133  }
134  //------------calculation of denominator for 1st order wave signal---
135  double denom_sum = 0;
136    for (int i = 0; i < nn ; i++)
137    {
138      denom_sum += s_om[i]*del_om ;
139    }
140  //---------calculation of 2nd order interction kernels---------------
141  double Dplus[][]   = new double[nn][nn];
142  double Dminus[][]  = new double[nn][nn];
143  double Fplus[][]   = new double[nn][nn];
144  double Fminus[][]  = new double[nn][nn];
145  double Gplus[][]   = new double[nn][nn];
146  double Gminus[][]  = new double[nn][nn];
147  double Hplus[][]   = new double[nn][nn];
148  double Hminus[][]  = new double[nn][nn];
149
150    for (int m=0; m <nn ;m++)
151      {
152      for (int n=0;n<nn ;n++ )
153        {
154          Dplus[m][n]    =g*(k[m]+k[n])*Math.tanh((k[m]+k[n])*h_star)-(omega[m]+omega[n])*(omega[m]+omega[n
                   ]);
155          Dminus[m][n]   =g*(k[m]-k[n])*Math.tanh((k[m]-k[n])*h_star)-(omega[m]-omega[n])*(omega[m]-omega[n
                   ]);
156          Fplus[m][n]    =-(g*k[m]*k[n]*Math.cosh((k[m]-k[n])*h_star)*Math.cosh((k[m]-k[n])*h_star))/(2.0*
                   omega[m]*omega[n]*Math.cosh(k[m]*h_star)*Math.cosh(k[n]*h_star)   )+0.5*(k[m]*Math.tanh(k[m
                   ]*h_star)+k[n]*Math.tanh(k[n]*h_star));
157          Fminus[m][n]   =-(g*k[m]*k[n]*Math.cosh((k[m]+k[n])*h_star)*Math.cosh((k[m]+k[n])*h_star))/(2.0*
                   omega[m]*omega[n]*Math.cosh(k[m]*h_star)*Math.cosh(k[n]*h_star)   )+0.5*(k[m]*Math.tanh(k[m
                   ]*h_star)+k[n]*Math.tanh(k[n]*h_star));
158          Gplus[m][n]    =-g*g*(k[n]*k[m]*(omega[m]+omega[n])*(1-Math.tanh(k[m]*h_star)*Math.tanh(k[n]*
                   h_star))/(omega[m]*omega[n])+( (k[m]*k[m])/(2.0*omega[m]*Math.cosh(k[m]*h_star)*Math.cosh(k
                   [m]*h_star))+ (k[n]*k[n])/(2.0*omega[n]*Math.cosh(k[n]*h_star)*Math.cosh(k[n]*h_star)) ) );
159          Gminus[m][n]   =-g*g*(k[n]*k[m]*(omega[m]-omega[n])*(1+Math.tanh(k[m]*h_star)*Math.tanh(k[n]*
                   h_star))/(omega[m]*omega[n])+( (k[m]*k[m])/(2.0*omega[m]*Math.cosh(k[m]*h_star)*Math.cosh(k
                   [m]*h_star))-(k[n]*k[n])/(2.0*omega[n]*Math.cosh(k[n]*h_star)*Math.cosh(k[n]*h_star))   ) );
160          Hplus[m][n]    =(omega[m]+omega[n])/g*Gplus[m][n]/Dplus[m][n]+Fplus[m][n]   ;
161          Hminus[m][n]   =(omega[m]-omega[n])/g*Gminus[m][n]/Dminus[m][n]+Fminus[m][n];
162        }
163      }
164  //---------calculation of waveheight for each frequency component
165  double [] ai = new double[nn];
166    for (int o = 0; o<nn ; o++ )
167    {
168      ai[o] = alpha*s_om[o]*del_om/denom_sum ;
169    }
170  //------------definition/initialisation of variables for wave signal
171    String surf = "";
172    String uVel = "";
173    String wVel = "";
```

```
174
175   double surf1[] = new double[nn]; //surface elevation
176   double surf1sum = 0 ;
177   double surf21[][] = new double[nn][nn];
178   double surf21sum = 0 ;
179   double surf22[][] = new double[nn][nn];
180   double surf22sum = 0 ;
181
182   double double_u1 [] = new double[nn]; //u-velocity
183   String string_u1 [] = new String[nn];
184   String u1 = "";
185   double double_u21a[][] = new double[nn][nn];
186   String string_u21a[][] = new String[nn][nn];
187   double double_u21b[][] = new double[nn][nn];
188   String string_u21b[][] = new String[nn][nn];
189   String u21 = "";
190   double double_u22[][] = new double[nn][nn];
191   String string_u22[][] = new String[nn][nn];
192   String u22 = "";
193
194   double double_w1 [] = new double[nn]; //w-velocity
195   String string_w1 [] = new String[nn];
196   String w1 = "";
197   double double_w21a[][] = new double[nn][nn];
198   String string_w21a[][] = new String[nn][nn];
199   double double_w21b[][] = new double[nn][nn];
200   String string_w21b[][] = new String[nn][nn];
201   String w21 = "";
202   double double_w22[][] = new double[nn][nn];
203   String string_w22[][] = new String[nn][nn];
204   String w22 = "";
205
206   double eps = 0.0;
207   double eps1 = 0.0;
208   double eps2 = 0.0;
209
210 //----------double loop to calculate the wave signals
211
212 for (int r = 0;r<nn ; r++ )
213 {   //------------------------------ 1st order parts
214     surf1[r]     = (ai[r]*Math.cos(k[r]*(x_star-x_foc)-omega[r]*(t_star-t_foc)+eps));
215     surf1sum += surf1[r];
216
217     double_u1[r]  = ( g*ai[r]*k[r]*Math.cos(k[r]*(0-x_foc)-omega[r]*(t_star-t_foc)))/(omega[r]*Math.cosh(
          k[r]*h_star));
218     string_u1[r]  = ("*cosh("+z_star+"*"+k[r]+"+"+(k[r]*h_star-k[r]*z_ausg)+")+");
219     u1          += (double_u1[r]+string_u1[r]);
220
221     double_w1[r]  = ( g*ai[r]*k[r]*Math.sin(k[r]*(0-x_foc)-omega[r]*(t_star-t_foc)))/(omega[r]*Math.cosh(
          k[r]*h_star));
222     string_w1[r]  = ("*sinh("+z_star+"*"+k[r]+"+"+(k[r]*h_star-k[r]*z_ausg)+")+");
223     w1          += (double_w1[r]+string_w1[r]);
224
225       for (int s=r+1;s<nn ;s++ )
226       {
227         surf21[r][s] = (ai[r]*ai[s]*Hplus[r][s]*Math.cos((k[r]+k[s])*(x_star-x_foc)-(omega[r]+omega[s])*(
              t_star-t_foc)+(eps+eps))+ai[r]*ai[s]*Hminus[r][s]*Math.cos((k[r]-k[s])*(x_star-x_foc)-(
              omega[r]-omega[s])*(t_star-t_foc)+(eps-eps)));
228         surf21sum += surf21[r][s];
229
230         double_u21a[r][s] = (k[r]+k[s])*ai[r]*ai[s]*Gplus[r][s]/(Dplus[r][s]*Math.cosh((k[r]+k[s])*h_star
              ))*Math.cos((k[r]+k[s])*(x_star-x_foc)-(omega[r]+omega[s])*(t_star-t_foc)+(eps1+eps2));
231         string_u21a[r][s] = ("*cosh("+(k[r]+k[s])+"*("+z_star+"+"+(h_star-z_ausg)+"))+");
232         double_u21b[r][s] = (k[r]-k[s])*ai[r]*ai[s]*Gminus[r][s]/(Dminus[r][s]*Math.cosh((k[r]-k[s])*
              h_star))*Math.cos((k[r]-k[s])*(x_star-x_foc)-(omega[r]-omega[s])*(t_star-t_foc)+(eps1-eps2)
              );
233         string_u21b[r][s] = ("*cosh("+(k[r]-k[s])+"*("+z_star+"+"+(h_star-z_ausg)+"))+");
```

```
234         u21         += (double_u21a[r][s]+string_u21a[r][s]+double_u21b[r][s]+string_u21b[r][s]);
235
236         double_w21a[r][s] = (k[r]+k[s])*ai[r]*ai[s]*Gplus[r][s]/(Dplus[r][s]*Math.cosh((k[r]+k[s])*h_star
                ))*Math.sin((k[r]+k[s])*(x_star-x_foc)-(omega[r]+omega[s])*(t_star-t_foc)+(eps1+eps2));
237         string_w21a[r][s] = ("*sinh("+(k[r]+k[s])+"*("+z_star+"+"+(h_star-z_ausg)+"))+");
238         double_w21b[r][s] = (k[r]-k[s])*ai[r]*ai[s]*Gminus[r][s]/(Dminus[r][s]*Math.cosh((k[r]-k[s])*
                h_star))*Math.sin((k[r]-k[s])*(x_star-x_foc)-(omega[r]-omega[s])*(t_star-t_foc)+(eps1-eps2)
                );
239         string_w21b[r][s] = ("*sinh("+(k[r]-k[s])+"*("+z_star+"+"+(h_star-z_ausg)+"))+");
240         w21         += (double_w21a[r][s]+string_w21a[r][s]+double_w21b[r][s]+string_w21b[r][s]);
241       }
242
243     surf22[r][r] = ai[r]*ai[r]*Hplus[r][r]*Math.cos(2*(k[r]*(x_star-x_foc)-omega[r]*(t_star-t_foc)+eps));
244     surf22sum += surf22[r][r];
245
246     double_u22[r][r]  = 2.0*k[r]*ai[r]*ai[r]*Gplus[r][r]/(Dplus[r][r]*Math.cosh((k[r]+k[r])*h_star))*Math
            .cos(2*(k[r]*(x_star-x_foc)-omega[r]*(t_star-t_foc)+eps1));
247     string_u22[r][r]  = ("*cosh("+(k[r]+k[r])+"*("+z_star+"+"+(h_star-z_ausg)+"))+");
248     u22         += (double_u22[r][r]+string_u22[r][r]);
249
250     double_w22[r][r]  = 2.0*k[r]*ai[r]*ai[r]*Gplus[r][r]/(Dplus[r][r]*Math.cosh((k[r]+k[r])*h_star))*Math
            .sin(2*(k[r]*(x_star-x_foc)-omega[r]*(t_star-t_foc)+eps1));
251     string_w22[r][r]  = ("*sinh("+(k[r]+k[r])+"*("+z_star+"+"+(h_star-z_ausg)+"))+");
252     w22         += (double_w22[r][r]+string_w22[r][r]);
253 }
254
255   surf = (""+(h+surf1sum+surf21sum+surf22sum));
256   uVel = (u1+u21+u22+"0");
257   wVel = (w1+w21+w22+"0");
258   String tt_star = (""+t_star);
259
260 //!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!end of calculation of wave signal!!!!!!!!!!!!!!!!!!!!!!!!!!!
261
262     userFieldFunction_0.setDefinition(uVel);     //assign results from calculation above to
263     userFieldFunction_1.setDefinition(wVel);     //appropriate field functions, these will be
264     userFieldFunction_2.setDefinition(surf);     //used to update the simulation file
265     userFieldFunction_3.setDefinition(tt_star); //once before the timestep
266
267
268 //!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!Export of simulation results!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
269
270 // relies on the existence of an XyzInternalTable 1 to export
271     XyzInternalTable xyzInternalTable_0 =
272       ((XyzInternalTable) simulation_0.getTableManager().getTable("XyzInternalTable 1"));
273     xyzInternalTable_0.extract();
274
275       // tsn is name of file
276       String tsn1 = resolvePath( "./p1/p1_"+ts +"_exp.csv");
277
278     xyzInternalTable_0.export(tsn1, 1);
279
280    // relies on the existence of an XyzInternalTable 2 to export
281     XyzInternalTable xyzInternalTable_1 =
282       ((XyzInternalTable) simulation_0.getTableManager().getTable("XyzInternalTable 2"));
283     xyzInternalTable_1.extract();
284
285       // tsn is name of file
286       String tsn2 = resolvePath( "./p2/p2_"+ts +"_exp.csv");
287
288     xyzInternalTable_1.export(tsn2, 1);
289
290 //!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!END  EXPORT ROUTINE!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
291
292     } //end of loop that controls what has to be done during a timestep
293
294   }
295 }
```

Appendix B

Fortran Routine used for the CV-FE solver

Horizontal velocities

```
1  #include "cfx5ext.h"
2  dllexport(uvel)
3        SUBROUTINE UVEL (
4       &  NLOC, NRET, NARG, RET, ARGS, CRESLT, CZ,DZ,IZ,LZ,RZ )
5  CC
6  CD User routine: defines an inlet velocity profile using the 1/7 power law
7  CC
8  CC ----------------------
9  CC          Input
10 CC ----------------------
11 CC
12 CC  NLOC    - size of current locale
13 CC  NRET    - number of components in result
14 CC  NARG    - number of arguments in call
15 CC  ARGS()  - (NLOC,NARG) argument values
16 CC
17 CC ----------------------
18 CC        Modified
19 CC ----------------------
20 CC
21 CC  Stacks possibly.
22 CC
23 CC ----------------------
24 CC          Output
25 CC ----------------------
26 CC
27 CC  RET()   - (NLOC,NRET) return values
28 CC  CRESLT - 'GOOD' for success
29 CC
30 CC ----------------------
31 CC         Details
32 CC ----------------------
33 CC
34 CC  Sets the surface elevation at the inlet
35 CC
36 CC  This routine reads in the time (t) and the z value (with this setup the vertical position is y)
37 CC
38 CC     surfFunction(t)
39 CC
40 CC    the function returns the position of the surface depending on the time
41 CC
42 CC
43 CC=============================================================================
44 C
45 C ---------------------------------
46 C      Preprocessor includes
47 C ---------------------------------
48 C
49 C
50 C ---------------------------------
```

```
C         Global Parameters
C -------------------------------
C
C
C -------------------------------
C         Argument list
C -------------------------------
C
      INTEGER NLOC,NARG,NRET
C
      CHARACTER CRESLT*(*)
C
      REAL ARGS(NLOC,NARG), RET(NLOC,NRET)
C
      INTEGER IZ(*)
      CHARACTER CZ(*)*(1)
      DOUBLE PRECISION DZ(*)
      LOGICAL LZ(*)
      REAL RZ(*)
C
C -------------------------------
C         External routines
C -------------------------------
C
C
C -------------------------------
C         Local Parameters
C -------------------------------
C
C
C -------------------------------
C         Local Variables
C -------------------------------
C
      INTEGER ILOC
      double precision        K, K1, K2, OMEGA, SOM, G, H, DELOM, DENOMSUM
      double precision        ALPHA, AI
      double precision        XCFX, XFOC, TCFX, TFOC, EPS, ZCFX, ZAUSGL
C     VARIABLES FOR 1ST and 2nd ORDER TERMS
      double precision        SURFDAZELL1,SURFDAZELL21,SURFDAZELL22
      double precision        SURFDAZELL31,SURFDAZELL32
      double precision        U1, U21, U31, U32
      double precision        W1, W21, W31, W32
C     CONSTANTS for second order terms
      double precision        BPLUSDAZELL, BMINUSDAZELL, MU
      double precision        DPLUSDAZELL, DMINUSDAZELL
*
      Dimension APLUSDAZELL(16,16)
      Dimension AMINUSDAZELL(16,16)
      Dimension BPLUSDAZELL(16,16)
      Dimension BMINUSDAZELL(16,16)
      Dimension DPLUSDAZELL(16,16)
      Dimension DMINUSDAZELL(16,16)
*
      Dimension MU(16)

      Real          PI
      Parameter (G=9.81,H=0.5,ALPHA=0.0632,PI=3.14159265)
      Dimension OMEGA(16)
      Dimension K(16)
      Dimension SOM(16)
      Dimension AI(16)

      INTEGER NN
      ZAUSGL = H
C
C -------------------------------
```

```
118 C         Stack pointers
119 C ---------------------------
120 C
121 C=========================================================================
122 C
123 C ---------------------------
124 C     Executable Statements
125 C ---------------------------
126 C
127 C
128 C Initialise RET() to zero.
129       CALL SET_A_0( RET, NLOC*NRET )
130 C      EXPONENT = 1.0/7.0
131 C
132 C---- Compute the velocity profile. Include check for VALUE < 0.
133 C
134       DO ILOC = 1,NLOC
135 CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!beginning of skript
136 CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!beginning of skript
137 CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!beginning of skript
138
139 C!!!!!!!!!!!!!!TEMPORARY CONSTANTS!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
140       EPS = 0.0
141       XCFX = 0.0
142       XFOC = 3.0
143       TCFX = ARGS(ILOC,1)
144       TFOC = 9.2
145       ZCFX = ARGS(ILOC,2)
146 *
147 C ------------------------Definition of spectrum-------------------
148 *
149       NN = 16
150       K1 = 1.0
151 C---------------SPECTRUM FOR CASE 3
152 *
153        i = 1
154             do while (i < NN + 1)
155                   MU(i)= 1
156                    i= i+1
157       end do
158
159       OMEGA(1) = 0.6*2*PI
160       OMEGA(2) = 0.646666667*2*PI
161       OMEGA(3) = 0.693333333*2*PI
162       OMEGA(4) = 0.7400001*2*PI
163       OMEGA(5) = 0.786666667*2*PI
164       OMEGA(6) = 0.833333333*2*PI
165       OMEGA(7) = 0.880000*2*PI
166       OMEGA(8) = 0.926666667*2*PI
167       OMEGA(9) = 0.973333333*2*PI
168       OMEGA(10) = 1.020000*2*PI
169       OMEGA(11) = 1.066666667*2*PI
170       OMEGA(12) = 1.113333333*2*PI
171       OMEGA(13) = 1.1600000001*2*PI
172       OMEGA(14) = 1.206666667*2*PI
173       OMEGA(15) = 1.253333333*2*PI
174       OMEGA(16) = 1.30000*2*PI
175 *
176       SOM(1) = 0.3841
177       SOM(2) = 1.134
178       SOM(3) = 1.956
179       SOM(4) = 3.5676
180       SOM(5) = 5.8257
181       SOM(6) = 7.1472
182       SOM(7) = 5.0286
183       SOM(8) = 2.4675
184       SOM(9) = 1.6596
```

```
185         SOM(10) = 1.7138
186         SOM(11) = 0.83
187         SOM(12) = 0.4604
188         SOM(13) = 0.2302
189         SOM(14) = 0.1375
190         SOM(16) = 0.0054
191
192   *
193   C ------------------------CALCULATION OF WAVENUMBER-------------------
194         i = 1
195         do while (i < NN + 1)
196               j = 0
197               do while ( j < 2000)
198                     K2 = (OMEGA(i)**2)/(G*tanh(K1*H))
199                     K1 = K2
200                     j = j+1
201               end do
202               K(i) = K1
203               i= i+1
204         end do
205   *
206   C ------------Calculation of denominator sum (1st order)-------------
207   *
208         DELOM = (OMEGA(NN)-OMEGA(1))/(NN-1)
209   *
210         DENOMSUM = 0
211         i = 1
212         do while (i < NN + 1)
213               DENOMSUM = DENOMSUM+SOM(i)*DELOM
214               i = i + 1
215         end do
216   *
217   C-------------Calculation of linear amplitude AI
218         i = 1
219         do while (i < NN + 1)
220               AI(i) = ALPHA*SOM(i)*DELOM/DENOMSUM
221               i = i + 1
222         end do
223   *
224   C-------------Calculation of constants for 2nd order terms
225
226   *
227         m = 1
228         do while (m<NN+1)
229               n=m
230               do while (n<NN+1)
231   C Constants for shallow water dalzell wave
232
233               DPLUSDAZELL(m,n)=((OMEGA(m)+OMEGA(n))**2
234         1      -G*(K(m)+K(n)*tanh(H*(K(m)+K(n)))))
235   *
236         DMINUSDAZELL(m,n)=((OMEGA(m)-OMEGA(n))**2
237         1      -G*(K(m)-K(n)*tanh(H*(K(m)-K(n)))))
238
239         APLUSDAZELL(m,n)=
240         1      -((OMEGA(m)*OMEGA(n))*(OMEGA(m)+OMEGA(n))
241         2      /DPLUSDAZELL(m,n))
242         2      *(1-(cos(MU(m)-MU(m))))
243         3      /(tanh(K(m)*H)*tanh(K(n)*H)))
244         4      +1/(2*DPLUSDAZELL(m,n))
245         5      *(((OMEGA(m))**3)/((sinh(K(m)*H))**2)+
246         6      ((OMEGA(n))**3)/((sinh(K(n)*H))**2))
247
248         AMINUSDAZELL(m,n)=
249         1      ((OMEGA(m)*OMEGA(n))*(OMEGA(m)-OMEGA(n))
250         2      /DMINUSDAZELL(m,n))
251         2      *(1+(cos(MU(m)-MU(m))))
```

```
252         3          /(tanh(K(m)*H)*tanh(K(n)*H)))
253         4          +1/(2*DMINUSDAZELL(m,n))
254         5          *(((OMEGA(m))**3)/((sinh(K(m)*H))**2)-
255         6          ((OMEGA(n))**3)/((sinh(K(n)*H))**2))
256
257          BPLUSDAZELL(m,n)=
258         1          ((OMEGA(m)**2+OMEGA(n)**2)/(2*G))
259         2           -((OMEGA(m)*OMEGA(n))/(2*G))*
260         3          (1-cos(MU(m)-MU(n))/
261         4          (tanh(K(m)*H)*tanh(K(n)*H)))
262         5          *(((OMEGA(m)+OMEGA(n))**2
263         5          +(G*(K(m)+K(n))*tanh((K(m)+K(n))*H)))
264         6          /DPLUSDAZELL(m,n))
265         7          +(OMEGA(m)+OMEGA(n))/
266         8          (2*G*DPLUSDAZELL(m,n))
267         9          *((OMEGA(m)**3)/(sinh(K(m)*H)**2)
268         1          +(OMEGA(n)**3)/(sinh(K(n)*H)**2))
269  *
270          BMINUSDAZELL(m,n)=
271         1          ((OMEGA(m)**2+OMEGA(n)**2)/(2*G))
272         2           +((OMEGA(m)*OMEGA(n))/(2*G))*
273         3          (1+cos(MU(m)-MU(n))/
274         4          (tanh(K(m)*H)*tanh(K(n)*H)))
275         5          *(((OMEGA(m)-OMEGA(n))**2
276         5          +(G*(K(m)-K(n))*tanh((K(m)-K(n))*H)))
277         6          /DMINUSDAZELL(m,n))
278         7          +(OMEGA(m)-OMEGA(n))/
279         8          (2*G*DMINUSDAZELL(m,n))
280         9          *((OMEGA(m)**3)/(sinh(K(m)*H)**2)
281         1          -(OMEGA(n)**3)/(sinh(K(n)*H)**2))
282  *
283                    n=n+1
284                    end do
285          m=m+1
286          end do
287  C
288  C-------------Calculatiion 1st Order terms----------
289  *
290          SURFDAZELL1  = 0
291          SURFDAZELL21 = 0
292          SURFDAZELL22 = 0
293          SURFDAZELL31 = 0
294          SURFDAZELL32 = 0
295          U1   = 0
296          U21  = 0
297          U31  = 0
298          U32  = 0
299          W1   = 0
300          W21  = 0
301          W31  = 0
302          W32  = 0
303  *
304
305          i=1
306          do while (i<NN+1)
307  *
308  C       SURFACE Elevation 1st order
309
310          SURFDAZELL1 = SURFDAZELL1+
311         1          (AI(i)*dcos(K(i)*(XCFX-XFOC)
312         2         -OMEGA(i)*(TCFX-TFOC)+EPS))
313
314
315  C       U Velocity 1st order
316  *
317                    U1 = U1
318         1                +(AI(i)*G*K(i)*cosh(K(i)*(ZCFX+H-ZAUSGL))
```

```
319       2              /(OMEGA(i)*cosh(K(i)*H))
320       3              *(cos(K(i)*(XCFX-XFOC)-OMEGA(i)*(TCFX-TFOC)+EPS)))
321 *
322 C      W Velocity 1st order
323 *
324             W1 = W1
325       1              +(AI(i)*G*K(i)*sinh(K(i)*(ZCFX+H-ZAUSGL))
326       2              /(OMEGA(i)*cosh(K(i)*H))
327       3              *(sin(K(i)*(XCFX-XFOC)-OMEGA(i)*(TCFX-TFOC)+EPS)))
328 *
329 C-----------------second order components 1st parts-----------------------
330             j=i+1
331 C    j=1
332       do while (j < NN+1)
333
334
335 C                    SURFACE Elevation 2nd order
336
337
338                      SURFDAZELL31 =  SURFDAZELL31+
339       1                             (AI(i)*AI(j)*BPLUSDAZELL(i,j)*
340       2                             dcos((K(i)+K(j))*(XCFX-XFOC)-
341       3                             (OMEGA(i)+OMEGA(j))*(TCFX-TFOC)+
342       4                             (EPS+EPS))
343       5                             )
344
345                      SURFDAZELL32 =  SURFDAZELL32+
346       1                             (AI(i)*AI(j)*BMINUSDAZELL(i,j)*
347       2                             dcos((K(i)-K(j))*(XCFX-XFOC)-
348       3                             (OMEGA(i)-OMEGA(j))*(TCFX-TFOC)+
349       4                             (EPS-EPS))
350       5                             )
351
352
353 C                    U Velocity 2nd order
354                      U31 =    U31+(
355       1            AI(i)*AI(j)*(K(i)+K(j))*APLUSDAZELL(i,j)
356       2            *cosh((K(i)+K(j))*(ZCFX+H-ZAUSGL))
357       3            /cosh((K(i)+K(j))*H)
358       4            *cos((K(i)+K(j))*(XCFX-XFOC)-
359       5                               (OMEGA(i)+OMEGA(j))*(TCFX-TFOC)+
360       6                               (EPS+EPS))
361       7                              )
362 *
363                      U32 =    U32+(
364       1            AI(i)*AI(j)*(K(i)-K(j))*AMINUSDAZELL(i,j)
365       2            *cosh((K(i)-K(j))*(ZCFX+H-ZAUSGL))
366       3            /cosh((K(i)-K(j))*H)
367       4            *cos((K(i)-K(j))*(XCFX-XFOC)-
368       5                               (OMEGA(i)-OMEGA(j))*(TCFX-TFOC)+
369       6                               (EPS-EPS))
370       7                             )
371 *
372 C                    U Velocity 2nd order
373                      W31 =    W31+(
374       1            AI(i)*AI(j)*(K(i)+K(j))*APLUSDAZELL(i,j)
375       2            *sinh((K(i)+K(j))*(ZCFX+H-ZAUSGL))
376       3            /cosh((K(i)+K(j))*H)
377       4            *sin((K(i)+K(j))*(XCFX-XFOC)-
378       5                               (OMEGA(i)+OMEGA(j))*(TCFX-TFOC)+
379       6                               (EPS+EPS))
380       7                             )
381 *
382                      W32 =    W32+(
383       1            AI(i)*AI(j)*(K(i)-K(j))*AMINUSDAZELL(i,j)
384       2            *sinh((K(i)-K(j))*(ZCFX+H-ZAUSGL))
385       3            /cosh((K(i)-K(j))*H)
```

```
     4          *sin((K(i)-K(j))*(XCFX-XFOC)-
     5                              (OMEGA(i)-OMEGA(j))*(TCFX-TFOC)+
     6                              (EPS-EPS))
     7                            )
*
                      j=j+1
                end do
*
                      U21 =     U21+(
     1                              0.75*(AI(i)**2)*OMEGA(i)*K(i)
     2                *cosh(2*K(i)*(ZCFX+H-ZAUSGL))
     3                /(sinh(K(i)*H))**4
     4                *cos(2*(K(i)*(XCFX-XFOC)-OMEGA(i)
     5                *(TCFX-TFOC)+EPS))
     6                           )
*
                      W21 =     W21+(
     1                              0.75*(AI(i)**2)*OMEGA(i)*K(i)
     2                *sinh(2*K(i)*(ZCFX+H-ZAUSGL))
     3                /(sinh(K(i)*H))**4
     4                *sin(2*(K(i)*(XCFX-XFOC)-OMEGA(i)
     5                *(TCFX-TFOC)+EPS))
     6                           )
*
     SURFDAZELL21 = SURFDAZELL21+
     1                           AI(i)**2*K(i)/(4*tanh(K(i)*H))*
     2                           (2+3/((sinh(K(i)*H)**2)))*
     3                           cos(2*(K(i)*(XCFX-XFOC)-
     4                           OMEGA(i)*(TCFX-TFOC)+EPS))
*
                   SURFDAZELL22 = SURFDAZELL22+
     1                           AI(i)**2*K(i)/(2*sinh(K(i)*H))
*
C-----------------end of sec order comp.-------------------------
               i=i+1
        end do
*
CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!end of skript
CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!end of skript
CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!end of skript
          RET(ILOC,1) = U1+U21+U31+U32
        END DO
C
C Set success flag.
        CRESLT = 'GOOD'
C
C===============================================================================
        END
```

Vertical velocities

```
#include "cfx5ext.h"
dllexport(wvel)
      SUBROUTINE WVEL (
     &  NLOC, NRET, NARG, RET, ARGS, CRESLT, CZ,DZ,IZ,LZ,RZ )
CC
CD User routine: defines an inlet velocity profile using the 1/7 power law
CC
CC ------------------------
CC         Input
CC ------------------------
CC
CC  NLOC   - size of current locale
CC  NRET   - number of components in result
CC  NARG   - number of arguments in call
CC  ARGS() - (NLOC,NARG) argument values
```

```
16 CC
17 CC ---------------------
18 CC       Modified
19 CC ---------------------
20 CC
21 CC  Stacks possibly.
22 CC
23 CC ---------------------
24 CC         Output
25 CC ---------------------
26 CC
27 CC  RET() - (NLOC,NRET) return values
28 CC  CRESLT - 'GOOD' for success
29 CC
30 CC ---------------------
31 CC       Details
32 CC ---------------------
33 CC
34 CC  Sets the surface elevation at the inlet
35 CC
36 CC  This routine reads in the time (t) and the z value (with this setup the vertical position is y)
37 CC
38 CC    surfFunction(t)
39 CC
40 CC    the function returns the position of the surface depending on the time
41 CC
42 CC
43 CC=====================================================================
44 C
45 C ---------------------------------
46 C     Preprocessor includes
47 C ---------------------------------
48 C
49 C
50 C ---------------------------------
51 C         Global Parameters
52 C ---------------------------------
53 C
54 C
55 C ---------------------------------
56 C         Argument list
57 C ---------------------------------
58 C
59       INTEGER NLOC,NARG,NRET
60 C
61       CHARACTER CRESLT*(*)
62 C
63       REAL ARGS(NLOC,NARG), RET(NLOC,NRET)
64 C
65       INTEGER IZ(*)
66       CHARACTER CZ(*)*(1)
67       DOUBLE PRECISION DZ(*)
68       LOGICAL LZ(*)
69       REAL RZ(*)
70 C
71 C ---------------------------------
72 C         External routines
73 C ---------------------------------
74 C
75 C
76 C ---------------------------------
77 C         Local Parameters
78 C ---------------------------------
79 C
80 C
81 C ---------------------------------
82 C         Local Variables
```

```
83 C ---------------------------------
84 C
85       INTEGER ILOC
86       double precision         K, K1, K2, OMEGA, SOM, G, H, DELOM, DENOMSUM
87       double precision         ALPHA, AI
88       double precision         XCFX, XFOC, TCFX, TFOC, EPS, ZCFX, ZAUSGL
89 C     VARIABLES FOR 1ST and 2nd ORDER TERMS
90
91       double precision         SURFDAZELL1,SURFDAZELL21,SURFDAZELL22
92       double precision         SURFDAZELL31,SURFDAZELL32
93       double precision         U1, U21, U31, U32
94       double precision         W1, W21, W31, W32
95 C     CONSTANTS for second order terms
96       double precision         BPLUSDAZELL, BMINUSDAZELL, MU
97       double precision         DPLUSDAZELL, DMINUSDAZELL
98 *
99       Dimension APLUSDAZELL(16,16)
100      Dimension AMINUSDAZELL(16,16)
101      Dimension BPLUSDAZELL(16,16)
102      Dimension BMINUSDAZELL(16,16)
103      Dimension DPLUSDAZELL(16,16)
104      Dimension DMINUSDAZELL(16,16)
105 *
106      Dimension MU(16)
107
108      Real          PI
109      Parameter (G=9.81,H=0.5,ALPHA=0.0632,PI=3.14159265)
110      Dimension OMEGA(16)
111      Dimension K(16)
112      Dimension SOM(16)
113      Dimension AI(16)
114
115      INTEGER NN
116      ZAUSGL = H
117 C
118 C ---------------------------------
119 C        Stack pointers
120 C ---------------------------------
121 C
122 C=========================================================================
123 C
124 C ---------------------------------
125 C    Executable Statements
126 C ---------------------------------
127 C
128 C
129 C Initialise RET() to zero.
130      CALL SET_A_0( RET, NLOC*NRET )
131 C      EXPONENT = 1.0/7.0
132 C
133 C---- Compute the velocity profile. Include check for VALUE < 0.
134 C
135      DO ILOC = 1,NLOC
136 CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!beginning of skript
137 CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!beginning of skript
138 CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!beginning of skript
139
140 C!!!!!!!!!!!!!!TEMPORARY CONSTANTS!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
141      EPS = 0.0
142      XCFX = 0.0
143      XFOC = 3.0
144      TCFX = ARGS(ILOC,1)
145      TFOC = 9.2
146      ZCFX = ARGS(ILOC,2)
147 *
148 C ------------------------Definition of spectrum-------------------
149 *
```

```
      NN = 16
      K1 = 1.0
C-------------SPECTRUM FOR CASE 3
*
       i = 1
            do while (i < NN + 1)
                  MU(i)= 1
                  i= i+1
      end do

      OMEGA(1) = 0.6*2*PI
      OMEGA(2) = 0.646666667*2*PI
      OMEGA(3) = 0.693333333*2*PI
      OMEGA(4) = 0.7400001*2*PI
      OMEGA(5) = 0.786666667*2*PI
      OMEGA(6) = 0.833333333*2*PI
      OMEGA(7) = 0.880000*2*PI
      OMEGA(8) = 0.926666667*2*PI
      OMEGA(9) = 0.973333333*2*PI
      OMEGA(10) = 1.020000*2*PI
      OMEGA(11) = 1.066666667*2*PI
      OMEGA(12) = 1.113333333*2*PI
      OMEGA(13) = 1.1600000001*2*PI
      OMEGA(14) = 1.206666667*2*PI
      OMEGA(15) = 1.253333333*2*PI
      OMEGA(16) = 1.30000*2*PI
*
      SOM(1) = 0.3841
      SOM(2) = 1.134
      SOM(3) = 1.956
      SOM(4) = 3.5676
      SOM(5) = 5.8257
      SOM(6) = 7.1472
      SOM(7) = 5.0286
      SOM(8) = 2.4675
      SOM(9) = 1.6596
      SOM(10) = 1.7138
      SOM(11) = 0.83
      SOM(12) = 0.4604
      SOM(13) = 0.2302
      SOM(14) = 0.1375
      SOM(16) = 0.0054

*
C ------------------------CALCULATION OF WAVENUMBER-----------------
      i = 1
      do while (i < NN + 1)
            j = 0
            do while ( j < 2000)
                  K2 = (OMEGA(i)**2)/(G*tanh(K1*H))
                  K1 = K2
                  j = j+1
            end do
            K(i) = K1
            i= i+1
      end do
*
C ------------Calculation of denominator sum (1st order)----------
*
      DELOM = (OMEGA(NN)-OMEGA(1))/(NN-1)
*
      DENOMSUM = 0
      i = 1
      do while (i < NN + 1)
            DENOMSUM = DENOMSUM+SOM(i)*DELOM
            i = i + 1
      end do
```

```
217 *
218 C--------------Calculation of linear amplitude AI
219       i = 1
220       do while (i < NN + 1)
221             AI(i) = ALPHA*SOM(i)*DELOM/DENOMSUM
222             i = i + 1
223       end do
224 *
225 C----------------Calculation of constants for 2nd order terms
226
227 *
228       m = 1
229       do while (m<NN+1)
230             n=m
231             do while (n<NN+1)
232 C Constants for shallow water dalzell wave
233
234             DPLUSDAZELL(m,n)=((OMEGA(m)+OMEGA(n))**2
235      1       -G*(K(m)+K(n)*tanh(H*(K(m)+K(n)))))
236 *
237       DMINUSDAZELL(m,n)=((OMEGA(m)-OMEGA(n))**2
238      1       -G*(K(m)-K(n)*tanh(H*(K(m)-K(n)))))
239
240       APLUSDAZELL(m,n)=
241      1       -((OMEGA(m)*OMEGA(n))*(OMEGA(m)+OMEGA(n))
242      2       /DPLUSDAZELL(m,n))
243      2       *(1-(cos(MU(m)-MU(m))
244      3       /(tanh(K(m)*H)*tanh(K(n)*H)))
245      4       +1/(2*DPLUSDAZELL(m,n))
246      5       *(((OMEGA(m))**3)/((sinh(K(m)*H))**2)+
247      6       ((OMEGA(n))**3)/((sinh(K(n)*H))**2))
248
249       AMINUSDAZELL(m,n)=
250      1       ((OMEGA(m)*OMEGA(n))*(OMEGA(m)-OMEGA(n))
251      2       /DMINUSDAZELL(m,n))
252      2       *(1+(cos(MU(m)-MU(m))
253      3       /(tanh(K(m)*H)*tanh(K(n)*H)))
254      4       +1/(2*DMINUSDAZELL(m,n))
255      5       *(((OMEGA(m))**3)/((sinh(K(m)*H))**2)-
256      6       ((OMEGA(n))**3)/((sinh(K(n)*H))**2))
257
258       BPLUSDAZELL(m,n)=
259      1       ((OMEGA(m)**2+OMEGA(n)**2)/(2*G))
260      2       -((OMEGA(m)*OMEGA(n))/(2*G))*
261      3       (1-cos(MU(m)-MU(n))/
262      4       (tanh(K(m)*H)*tanh(K(n)*H)))
263      5       *(((OMEGA(m)+OMEGA(n))**2
264      5       +(G*(K(m)+K(n))*tanh((K(m)+K(n))*H)))
265      6       /DPLUSDAZELL(m,n))
266      7       +(OMEGA(m)+OMEGA(n))/
267      8       (2*G*DPLUSDAZELL(m,n))
268      9       *((OMEGA(m)**3)/(sinh(K(m)*H)**2)
269      1       +(OMEGA(n)**3)/(sinh(K(n)*H)**2))
270 *
271       BMINUSDAZELL(m,n)=
272      1       ((OMEGA(m)**2+OMEGA(n)**2)/(2*G))
273      2       +((OMEGA(m)*OMEGA(n))/(2*G))*
274      3       (1+cos(MU(m)-MU(n))/
275      4       (tanh(K(m)*H)*tanh(K(n)*H)))
276      5       *(((OMEGA(m)-OMEGA(n))**2
277      5       +(G*(K(m)-K(n))*tanh((K(m)-K(n))*H)))
278      6       /DMINUSDAZELL(m,n))
279      7       +(OMEGA(m)-OMEGA(n))/
280      8       (2*G*DMINUSDAZELL(m,n))
281      9       *((OMEGA(m)**3)/(sinh(K(m)*H)**2)
282      1       -(OMEGA(n)**3)/(sinh(K(n)*H)**2))
283 *
```

```
284               n=n+1
285               end do
286         m=m+1
287         end do
288 C
289 C-------------Calculatiion 1st Order terms----------
290 *
291         SURFDAZELL1  = 0
292         SURFDAZELL21 = 0
293         SURFDAZELL22 = 0
294         SURFDAZELL31 = 0
295         SURFDAZELL32 = 0
296         U1  = 0
297         U21 = 0
298         U31 = 0
299         U32 = 0
300         W1  = 0
301         W21 = 0
302         W31 = 0
303         W32 = 0
304 *
305
306         i=1
307         do while (i<NN+1)
308 *
309 C       SURFACE Elevation 1st order
310
311         SURFDAZELL1 = SURFDAZELL1+
312        1          (AI(i)*dcos(K(i)*(XCFX-XFOC)
313        2         -OMEGA(i)*(TCFX-TFOC)+EPS))
314
315
316 C       U Velocity 1st order
317 *
318               U1 = U1
319        1              +(AI(i)*G*K(i)*cosh(K(i)*(ZCFX+H-ZAUSGL))
320        2              /(OMEGA(i)*cosh(K(i)*H))
321        3              *(cos(K(i)*(XCFX-XFOC)-OMEGA(i)*(TCFX-TFOC)+EPS)))
322 *
323 C       W Velocity 1st order
324 *
325               W1 = W1
326        1              +(AI(i)*G*K(i)*sinh(K(i)*(ZCFX+H-ZAUSGL))
327        2              /(OMEGA(i)*cosh(K(i)*H))
328        3              *(sin(K(i)*(XCFX-XFOC)-OMEGA(i)*(TCFX-TFOC)+EPS)))
329 *
330 C---------------second order components 1st parts----------------------
331               j=i+1
332 C     j=1
333       do while (j < NN+1)
334
335
336 C                     SURFACE Elevation 2nd order
337
338
339                       SURFDAZELL31 =  SURFDAZELL31+
340         1                           (AI(i)*AI(j)*BPLUSDAZELL(i,j)*
341         2                           dcos((K(i)+K(j))*(XCFX-XFOC)-
342         3                           (OMEGA(i)+OMEGA(j))*(TCFX-TFOC)+
343         4                           (EPS+EPS))
344         5                           )
345
346                       SURFDAZELL32 =  SURFDAZELL32+
347         1                           (AI(i)*AI(j)*BMINUSDAZELL(i,j)*
348         2                           dcos((K(i)-K(j))*(XCFX-XFOC)-
349         3                           (OMEGA(i)-OMEGA(j))*(TCFX-TFOC)+
350         4                           (EPS-EPS))
```

```
      5                          )
C
C
C                 U Velocity 2nd order
                  U31 =     U31+(
     1          AI(i)*AI(j)*(K(i)+K(j))*APLUSDAZELL(i,j)
     2          *cosh((K(i)+K(j))*(ZCFX+H-ZAUSGL))
     3          /cosh((K(i)+K(j))*H)
     4          *cos((K(i)+K(j))*(XCFX-XFOC)-
     5                            (OMEGA(i)+OMEGA(j))*(TCFX-TFOC)+
     6                            (EPS+EPS))
     7                           )
*
                  U32 =     U32+(
     1          AI(i)*AI(j)*(K(i)-K(j))*AMINUSDAZELL(i,j)
     2          *cosh((K(i)-K(j))*(ZCFX+H-ZAUSGL))
     3          /cosh((K(i)-K(j))*H)
     4          *cos((K(i)-K(j))*(XCFX-XFOC)-
     5                            (OMEGA(i)-OMEGA(j))*(TCFX-TFOC)+
     6                            (EPS-EPS))
     7                           )
*
C                 U Velocity 2nd order
                  W31 =     W31+(
     1          AI(i)*AI(j)*(K(i)+K(j))*APLUSDAZELL(i,j)
     2          *sinh((K(i)+K(j))*(ZCFX+H-ZAUSGL))
     3          /cosh((K(i)+K(j))*H)
     4          *sin((K(i)+K(j))*(XCFX-XFOC)-
     5                            (OMEGA(i)+OMEGA(j))*(TCFX-TFOC)+
     6                            (EPS+EPS))
     7                           )
*
                  W32 =     W32+(
     1          AI(i)*AI(j)*(K(i)-K(j))*AMINUSDAZELL(i,j)
     2          *sinh((K(i)-K(j))*(ZCFX+H-ZAUSGL))
     3          /cosh((K(i)-K(j))*H)
     4          *sin((K(i)-K(j))*(XCFX-XFOC)-
     5                            (OMEGA(i)-OMEGA(j))*(TCFX-TFOC)+
     6                            (EPS-EPS))
     7                           )
*
C
                  j=j+1
            end do
*
                  U21 =     U21+(
     1                         0.75*(AI(i)**2)*OMEGA(i)*K(i)
     2            *cosh(2*K(i)*(ZCFX+H-ZAUSGL))
     3            /(sinh(K(i)*H))**4
     4            *cos(2*(K(i)*(XCFX-XFOC)-OMEGA(i)
     5            *(TCFX-TFOC)+EPS))
     6                        )
*
                  W21 =     W21+(
     1                         0.75*(AI(i)**2)*OMEGA(i)*K(i)
     2            *sinh(2*K(i)*(ZCFX+H-ZAUSGL))
     3            /(sinh(K(i)*H))**4
     4            *sin(2*(K(i)*(XCFX-XFOC)-OMEGA(i)
     5            *(TCFX-TFOC)+EPS))
     6                        )
*
      SURFDAZELL21 = SURFDAZELL21+
     1                        AI(i)**2*K(i)/(4*tanh(K(i)*H))*
     2                        (2+3/((sinh(K(i)*H)**2)))*
     3                        cos(2*(K(i)*(XCFX-XFOC)-
     4                        OMEGA(i)*(TCFX-TFOC)+EPS))
*
```

```
418                       SURFDAZELL22 = SURFDAZELL22+
419       1                           AI(i)**2*K(i)/(2*sinh(K(i)*H))
420  *
421  C-----------------end of sec order comp.-------------------------
422                 i=i+1
423           end do
424  *
425  CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!end of skript
426  CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!end of skript
427  CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!end of skript
428            RET(ILOC,1) = W1+W21+W31+W32
429           END DO
430  C
431  C Set success flag.
432           CRESLT = 'GOOD'
433  C
434  C=======================================================================
435           END
```

Surface elevation

```
1  #include "cfx5ext.h"
2  dllexport(surf)
3         SUBROUTINE SURF (
4        &  NLOC, NRET, NARG, RET, ARGS, CRESLT, CZ,DZ,IZ,LZ,RZ )
5  CC
6  CC
7  CC ---------------------
8  CC          Input
9  CC ---------------------
10 CC
11 CC  NLOC   - size of current locale
12 CC  NRET   - number of components in result
13 CC  NARG   - number of arguments in call
14 CC  ARGS() - (NLOC,NARG) argument values
15 CC
16 CC ---------------------
17 CC       Modified
18 CC ---------------------
19 CC
20 CC  Stacks possibly.
21 CC
22 CC ---------------------
23 CC          Output
24 CC ---------------------
25 CC
26 CC  RET()  - (NLOC,NRET) return values
27 CC  CRESLT - 'GOOD' for success
28 CC
29 CC ---------------------
30 CC        Details
31 CC ---------------------
32 CC
33 CC  Sets the surface elevation at the inlet
34 CC
35 CC  This routine reads in the time (t) and the z value (with this setup the vertical position is y)
36 CC
37 CC     surfFunction(t)
38 CC
39 CC    the function returns the position of the surface depending on the time
40 CC
41 CC
42 CC=======================================================================
43 C
44 C ---------------------------------
45 C      Preprocessor includes
46 C ---------------------------------
```

```
47 C
48 C
49 C ------------------------------
50 C       Global Parameters
51 C ------------------------------
52 C
53 C
54 C ------------------------------
55 C       Argument list
56 C ------------------------------
57 C
58       INTEGER NLOC,NARG,NRET
59 C
60       CHARACTER CRESLT*(*)
61 C
62       REAL ARGS(NLOC,NARG), RET(NLOC,NRET)
63 C
64       INTEGER IZ(*)
65       CHARACTER CZ(*)*(1)
66       DOUBLE PRECISION DZ(*)
67       LOGICAL LZ(*)
68       REAL RZ(*)
69 C
70 C ------------------------------
71 C       External routines
72 C ------------------------------
73 C
74 C
75 C ------------------------------
76 C       Local Parameters
77 C ------------------------------
78 C
79 C
80 C ------------------------------
81 C       Local Variables
82 C ------------------------------
83 C
84       INTEGER ILOC
85       double precision        K, K1, K2, OMEGA, SOM, G, H, DELOM, DENOMSUM
86       double precision        ALPHA, AI
87       double precision        XCFX, XFOC, TCFX, TFOC, EPS, ZCFX, ZAUSGL
88 C     VARIABLES FOR 1ST and 2nd ORDER TERMS
89
90       double precision        SURFDAZELL1,SURFDAZELL21,SURFDAZELL22
91       double precision        SURFDAZELL31,SURFDAZELL32
92       double precision        U1, U21, U31, U32
93       double precision        W1, W21, W31, W32
94 C     CONSTANTS for second order terms
95       double precision        BPLUSDAZELL, BMINUSDAZELL, MU
96       double precision        DPLUSDAZELL, DMINUSDAZELL
97 *
98       Dimension APLUSDAZELL(16,16)
99       Dimension AMINUSDAZELL(16,16)
100      Dimension BPLUSDAZELL(16,16)
101      Dimension BMINUSDAZELL(16,16)
102      Dimension DPLUSDAZELL(16,16)
103      Dimension DMINUSDAZELL(16,16)
104 *
105      Dimension MU(16)
106
107      Real         PI
108      Parameter (G=9.81,H=0.5,ALPHA=0.0632,PI=3.14159265)
109      Dimension OMEGA(16)
110      Dimension K(16)
111      Dimension SOM(16)
112      Dimension AI(16)
113
```

```
114         INTEGER NN
115         ZAUSGL = H
116
117   C
118   C ---------------------------------
119   C          Stack pointers
120   C ---------------------------------
121   C
122   C=====================================================================
123   C
124   C ---------------------------------
125   C     Executable Statements
126   C ---------------------------------
127   C
128   C
129   C Initialise RET() to zero.
130         CALL SET_A_0( RET, NLOC*NRET )
131   C       EXPONENT = 1.0/7.0
132   C
133   C---- Compute the velocity profile. Include check for VALUE < 0.
134   C
135         DO ILOC = 1,NLOC
136   CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!beginning of skript
137   CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!beginning of skript
138   CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!beginning of skript
139
140   C!!!!!!!!!!!!!!TEMPORARY CONSTANTS!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
141         EPS = 0.0
142         XCFX = 0.0
143         XFOC = 3.0
144         TCFX = ARGS(ILOC,1)
145         TFOC = 9.2
146         ZCFX = ARGS(ILOC,2)
147   *
148   C ------------------------Definition of spectrum-------------------
149   *
150         NN = 16
151         K1 = 1.0
152   C---------------SPECTRUM FOR CASE 3
153   *
154          i = 1
155               do while (i < NN + 1)
156                     MU(i)= 1
157                      i= i+1
158         end do
159
160         OMEGA(1) = 0.6*2*PI
161         OMEGA(2) = 0.646666667*2*PI
162         OMEGA(3) = 0.693333333*2*PI
163         OMEGA(4) = 0.7400001*2*PI
164         OMEGA(5) = 0.786666667*2*PI
165         OMEGA(6) = 0.833333333*2*PI
166         OMEGA(7) = 0.880000*2*PI
167         OMEGA(8) = 0.926666667*2*PI
168         OMEGA(9) = 0.973333333*2*PI
169         OMEGA(10) = 1.020000*2*PI
170         OMEGA(11) = 1.066666667*2*PI
171         OMEGA(12) = 1.113333333*2*PI
172         OMEGA(13) = 1.1600000001*2*PI
173         OMEGA(14) = 1.206666667*2*PI
174         OMEGA(15) = 1.253333333*2*PI
175         OMEGA(16) = 1.30000*2*PI
176   *
177         SOM(1) = 0.3841
178         SOM(2) = 1.134
179         SOM(3) = 1.956
180         SOM(4) = 3.5676
```

```
      SOM(5) = 5.8257
      SOM(6) = 7.1472
      SOM(7) = 5.0286
      SOM(8) = 2.4675
      SOM(9) = 1.6596
      SOM(10) = 1.7138
      SOM(11) = 0.83
      SOM(12) = 0.4604
      SOM(13) = 0.2302
      SOM(14) = 0.1375
      SOM(16) = 0.0054

*
C ------------------------CALCULATION OF WAVENUMBER-------------------
      i = 1
      do while (i < NN + 1)
            j = 0
            do while ( j < 2000)
                  K2 = (OMEGA(i)**2)/(G*tanh(K1*H))
                  K1 = K2
                  j = j+1
            end do
            K(i) = K1
            i= i+1
      end do
*
C ------------Calculation of denominator sum (1st order)-------------
*
      DELOM = (OMEGA(NN)-OMEGA(1))/(NN-1)
*
      DENOMSUM = 0
      i = 1
      do while (i < NN + 1)
            DENOMSUM = DENOMSUM+SOM(i)*DELOM
            i = i + 1
      end do
*
C-------------Calculation of linear amplitude AI
      i = 1
      do while (i < NN + 1)
            AI(i) = ALPHA*SOM(i)*DELOM/DENOMSUM
            i = i + 1
      end do
*
C-------------Calculation of constants for 2nd order terms

*
      m = 1
      do while (m<NN+1)
            n=m
            do while (n<NN+1)
C Constants for shallow water dalzell wave

            DPLUSDAZELL(m,n)=((OMEGA(m)+OMEGA(n))**2
     1       -G*(K(m)+K(n)*tanh(H*(K(m)+K(n)))))
*
      DMINUSDAZELL(m,n)=((OMEGA(m)-OMEGA(n))**2
     1       -G*(K(m)-K(n)*tanh(H*(K(m)-K(n)))))

      APLUSDAZELL(m,n)=
     1       -((OMEGA(m)*OMEGA(n))*(OMEGA(m)+OMEGA(n))
     2       /DPLUSDAZELL(m,n))
     2       *(1-(cos(MU(m)-MU(m))))
     3       /(tanh(K(m)*H)*tanh(K(n)*H)))
     4       +1/(2*DPLUSDAZELL(m,n))
     5       *(((OMEGA(m))**3)/((sinh(K(m)*H))**2)+
     6       ((OMEGA(n))**3)/((sinh(K(n)*H))**2))
```

```
248
249          AMINUSDAZELL(m,n)=
250         1          ((OMEGA(m)*OMEGA(n))*(OMEGA(m)-OMEGA(n))
251         2          /DMINUSDAZELL(m,n))
252         2          *(1+(cos(MU(m)-MU(m)))
253         3          /(tanh(K(m)*H)*tanh(K(n)*H)))
254         4          +1/(2*DMINUSDAZELL(m,n))
255         5          *((((OMEGA(m))**3)/((sinh(K(m)*H))**2)-
256         6          ((OMEGA(n))**3)/((sinh(K(n)*H))**2))
257
258          BPLUSDAZELL(m,n)=
259         1          ((OMEGA(m)**2+OMEGA(n)**2)/(2*G))
260         2            -((OMEGA(m)*OMEGA(n))/(2*G))*
261         3          (1-cos(MU(m)-MU(n))/
262         4          (tanh(K(m)*H)*tanh(K(n)*H)))
263         5          *(((OMEGA(m)+OMEGA(n))**2
264         5          +(G*(K(m)+K(n))*tanh((K(m)+K(n))*H)))
265         6          /DPLUSDAZELL(m,n))
266         7          +(OMEGA(m)+OMEGA(n))/
267         8          (2*G*DPLUSDAZELL(m,n))
268         9          *((OMEGA(m)**3)/(sinh(K(m)*H)**2)
269         1          +(OMEGA(n)**3)/(sinh(K(n)*H)**2))
270  *
271          BMINUSDAZELL(m,n)=
272         1          ((OMEGA(m)**2+OMEGA(n)**2)/(2*G))
273         2            +((OMEGA(m)*OMEGA(n))/(2*G))*
274         3          (1+cos(MU(m)-MU(n))/
275         4          (tanh(K(m)*H)*tanh(K(n)*H)))
276         5          *(((OMEGA(m)-OMEGA(n))**2
277         5          +(G*(K(m)-K(n))*tanh((K(m)-K(n))*H)))
278         6          /DMINUSDAZELL(m,n))
279         7          +(OMEGA(m)-OMEGA(n))/
280         8          (2*G*DMINUSDAZELL(m,n))
281         9          *((OMEGA(m)**3)/(sinh(K(m)*H)**2)
282         1          -(OMEGA(n)**3)/(sinh(K(n)*H)**2))
283  *
284                    n=n+1
285                    end do
286          m=m+1
287          end do
288  C
289  C-------------Calculatiion 1st Order terms-----------
290  *
291          SURFDAZELL1  = 0
292          SURFDAZELL21 = 0
293          SURFDAZELL22 = 0
294          SURFDAZELL31 = 0
295          SURFDAZELL32 = 0
296          U1   = 0
297          U21  = 0
298          U31  = 0
299          U32  = 0
300          W1   = 0
301          W21  = 0
302          W31  = 0
303          W32  = 0
304  *
305
306          i=1
307          do while (i<NN+1)
308  *
309  C       SURFACE Elevation 1st order
310
311          SURFDAZELL1 = SURFDAZELL1+
312         1          (AI(i)*dcos(K(i)*(XCFX-XFOC)
313         2          -OMEGA(i)*(TCFX-TFOC)+EPS))
314
```

```
315
316 C     U Velocity 1st order
317 *
318               U1 = U1
319       1             +(AI(i)*G*K(i)*cosh(K(i)*(ZCFX+H-ZAUSGL))
320       2             /(OMEGA(i)*cosh(K(i)*H))
321       3             *(cos(K(i)*(XCFX-XFOC)-OMEGA(i)*(TCFX-TFOC)+EPS)))
322 *
323 C     W Velocity 1st order
324 *
325               W1 = W1
326       1             +(AI(i)*G*K(i)*sinh(K(i)*(ZCFX+H-ZAUSGL))
327       2             /(OMEGA(i)*cosh(K(i)*H))
328       3             *(sin(K(i)*(XCFX-XFOC)-OMEGA(i)*(TCFX-TFOC)+EPS)))
329 *
330 C-----------------second order components 1st parts-----------------------
331               j=i+1
332 C    j=1
333      do while (j < NN+1)
334
335
336 C                   SURFACE Elevation 2nd order
337
338
339                     SURFDAZELL31 =  SURFDAZELL31+
340       1                         (AI(i)*AI(j)*BPLUSDAZELL(i,j)*
341       2                         dcos((K(i)+K(j))*(XCFX-XFOC)-
342       3                         (OMEGA(i)+OMEGA(j))*(TCFX-TFOC)+
343       4                         (EPS+EPS))
344       5                         )
345
346                     SURFDAZELL32 =  SURFDAZELL32+
347       1                         (AI(i)*AI(j)*BMINUSDAZELL(i,j)*
348       2                         dcos((K(i)-K(j))*(XCFX-XFOC)-
349       3                         (OMEGA(i)-OMEGA(j))*(TCFX-TFOC)+
350       4                         (EPS-EPS))
351       5                         )
352
353
354 C                   U Velocity 2nd order
355                     U31 =    U31+(
356       1         AI(i)*AI(j)*(K(i)+K(j))*APLUSDAZELL(i,j)
357       2         *cosh((K(i)+K(j))*(ZCFX+H-ZAUSGL))
358       3         /cosh((K(i)+K(j))*H)
359       4         *cos((K(i)+K(j))*(XCFX-XFOC)-
360       5                           (OMEGA(i)+OMEGA(j))*(TCFX-TFOC)+
361       6                           (EPS+EPS))
362       7                          )
363 *
364                     U32 =    U32+(
365       1         AI(i)*AI(j)*(K(i)-K(j))*AMINUSDAZELL(i,j)
366       2         *cosh((K(i)-K(j))*(ZCFX+H-ZAUSGL))
367       3         /cosh((K(i)-K(j))*H)
368       4         *cos((K(i)-K(j))*(XCFX-XFOC)-
369       5                           (OMEGA(i)-OMEGA(j))*(TCFX-TFOC)+
370       6                           (EPS-EPS))
371       7                         )
372 *
373 C                   U Velocity 2nd order
374                     W31 =    W31+(
375       1         AI(i)*AI(j)*(K(i)+K(j))*APLUSDAZELL(i,j)
376       2         *sinh((K(i)+K(j))*(ZCFX+H-ZAUSGL))
377       3         /cosh((K(i)+K(j))*H)
378       4         *sin((K(i)+K(j))*(XCFX-XFOC)-
379       5                           (OMEGA(i)+OMEGA(j))*(TCFX-TFOC)+
380       6                           (EPS+EPS))
381       7                         )
```

```
*
                  W32 =     W32+(
     1         AI(i)*AI(j)*(K(i)-K(j))*AMINUSDAZELL(i,j)
     2         *sinh((K(i)-K(j))*(ZCFX+H-ZAUSGL))
     3         /cosh((K(i)-K(j))*H)
     4         *sin((K(i)-K(j))*(XCFX-XFOC)-
     5                          (OMEGA(i)-OMEGA(j))*(TCFX-TFOC)+
     6                          (EPS-EPS))
     7                        )
*

                  j=j+1
             end do
*
                  U21 =     U21+(
     1                        0.75*(AI(i)**2)*OMEGA(i)*K(i)
     2            *cosh(2*K(i)*(ZCFX+H-ZAUSGL))
     3            /(sinh(K(i)*H))**4
     4            *cos(2*(K(i)*(XCFX-XFOC)-OMEGA(i)
     5            *(TCFX-TFOC)+EPS))
     6                      )
*
                  W21 =     W21+(
     1                        0.75*(AI(i)**2)*OMEGA(i)*K(i)
     2            *sinh(2*K(i)*(ZCFX+H-ZAUSGL))
     3            /(sinh(K(i)*H))**4
     4            *sin(2*(K(i)*(XCFX-XFOC)-OMEGA(i)
     5            *(TCFX-TFOC)+EPS))
     6                      )
*
      SURFDAZELL21 = SURFDAZELL21+
     1                      AI(i)**2*K(i)/(4*tanh(K(i)*H))*
     2                      (2+3/((sinh(K(i)*H)**2)))*
     3                      cos(2*(K(i)*(XCFX-XFOC)-
     4                      OMEGA(i)*(TCFX-TFOC)+EPS))
*
                 SURFDAZELL22 = SURFDAZELL22+
     1                      AI(i)**2*K(i)/(2*sinh(K(i)*H))
*
C---------------end of sec order comp.-----------------------
             i=i+1
         end do
*
CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!end of skript
CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!end of skript
CC!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!end of skript
            RET(ILOC,1) = H+SURFDAZELL1+SURFDAZELL21-SURFDAZELL22
     1          +SURFDAZELL31+SURFDAZELL32
         END DO
C
C Set success flag.
         CRESLT = 'GOOD'
C
C======================================================================
         END
```