Fluid-Structure Interaction Simulation of Complex Floating Structures and Waves

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Antoni Calderer Elias

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Professors Fotis Sotiropoulos and Lian Shen

November, 2015

Acknowledgements

First and foremost, I would like to express my sincere thanks to my principal supervisor, Professor Fotis Sotiropoulos, for giving me the opportunity to study at the Saint Anthony Falls Laboratory and for the invaluable encouragement, guidance, and ongoing support during this research. I extend my sincere gratitude to my co-advisor, Professor Lian Shen, for his valuable contribution in this research and his practical support. I also owe thanks to the other members of my dissertation committee, Professors Michele Guala and Vaughan Voller, who gave me new perspective and helped me to enrich the value of this work.

I would like to recognize the contribution of all members of my research group who provided me with valuable help and many enlightening discussions, Dr. Xiaolei Yang, Dr. Anvar Gilmanov, Dr. Dennis Angelidis, Dr. Ali Khosronejad, Dr. Trung Le., Dr. Aaron Boomsma, Mohammad Hajit, Saurabh Chawdhary, Danny Foti, Ming Li, Professor Seokkoo Kang, and Professor Imman Borazjani. As well, I appreciate the invaluable support and assistance that I received from Dr. Xin Guo. Thanks to Christ Feist for providing the experimental data, to Leonardo Chamorro for introducing me to the wind energy topic, and Arno Mayrhofer from who I learned many new computer tools that enhanced research productivity.

I owe thanks to all collaborators in Sandia National Laboratories, Kelley Ruehl for hosting and instructing me during my visit to the Sandia base in Albuquerque and for providing the low order models simulation results, and Dr. Todd Griffith, Dr. Ann Dallman , and Dr. Tommy Herges, for their support and advice throughout the preparation of material for releasing the computational code.

I deeply appreciate the financial support for this work from the US Department of Energy (DE-EE 0005482), the US National Science Foundation (CBET-1341062), the

Office of the Naval Research (00043722), and the University of Minnesota Initiative for Renewable Energy and the Environment. The computational resources for obtaining the numerical results presented in this dissertation were provided by the Minnesota Supercomputing Institute and Sandia National Laboratories.

Finally, my heartfelt thanks to my beloved partner, Laura, for the love, caring, and unconditional support that made this five years wonderful. I am also grateful for the support of my parents Valenti and Pepita, my uncle Doug, my aunts Carme and Queralt, my brother Ramon and his wife Gemma, and my cousin Clara.

Dedication

To Laura

Abstract

A novel computational framework for simulating the coupled interaction of complex floating structures with large-scale ocean waves and atmospheric turbulent winds has been developed. This framework is based on a domain decomposition approach coupling a large-scale far-field domain, where realistic wind and wave conditions representative from offshore environments are developed, with a near-field domain, where wind-wave-body interactions can be investigated. The method applied in the near-field domain is based on a partitioned fluid-structure interaction (FSI) approach combining a sharp interface curvilinear immersed boundary (CURVIB) method with a two-phase flow level set formulation and is capable of solving free surface flows interacting non-linearly with complex real life floating structures. An aspect that was found critical in FSI applications when coupling the structural domain with the two-fluid domain is the approach used to calculate the force that the fluid exerts to the body. A new force calculation approach, based on projecting the pressure on the surface of the body using the momentum equation along the local normal to the body direction, was proposed. The new approach was shown, through extensive numerical tests, to greatly improve the ability of the method to correctly predict the dynamics of the floating structure motion. For the far-field domain, a large-scale wave and wind model based on the two-fluid approach of Yang and Shen [1, 2], which integrates a viscous Navier-Stokes solver with undulatory boundaries for the motion of the air and an efficient potential-flow based wave solver, was employed. For coupling the far-field and near-field domains, a wave generation method for incorporating complex wave fields into Navier-Stokes solvers has been proposed. The wave generation method was validated for a variety of wave cases including a broadband spectrum. The computational framework has been further validated for wave-body interactions by replicating an experiment of floating wind turbine model subject to different sinusoidal wave forces. The simulation results, which agree well with the experimental data, have been compared with other numerical results computed with available numerical codes based on lower order assumptions. Despite the higher computational cost of our method, it yields to results that are in overall better accuracy and it can capture many additional flow features neglected by lower order models. Finally, the

full capabilities of the framework have been demonstrated by carrying out large eddy simulation (LES) of a floating wind turbine interacting with realistic ocean wind and wave conditions.

Contents

List of Tables

List of Figures

List of Abbreviations

2D two-dimensional. 13, 15, 21, 36, 41, 47, 55, 56, 67, 74, 86, 87, 93, 94

3D three-dimensional. 6, 8, 13, 14, 21, 26, 42, 49, 51, 67, 71, 74, 78, 80, 83, 84, 86, 91, 92, 94, 95, 101, 102, 153, 157

ALE Arbitrary Lagrangian-Eulerian. 9, 10, 14

BCs boundary conditions. 43–45

BEM boundary element method. 5, 7, 16

CFD Computational Fluid Dynamics. 129, 159, 160

CG center of gravity. 71, 110, 111, 129, 131

CURVIB Curvilinear Immersed Boundary. 11, 14, 18, 19, 31, 32, 35, 38, 39, 55, 65, 66, 96, 113, 148, 155, 156

DNS direct numerical simulation. 5, 45

DoF degrees of freedom. 15, 22, 26, 27, 111, 112, 120, 131, 136, 142, 144, 149, 154, 158–160

EoM equations of motion. 114, 129, 144

FSI fluid-structure interaction. 1, 3, 9, 11, 15, 18, 21, 22, 39–43, 46, 47, 53, 55, 65–75, 77–82, 84, 85, 108, 127–132, 155–157, 159, 160

GMRES generalized minimal residual. 29

HCIB hybrid-Cartesian/immersed boundary. 10, 11, 35

HOS high-order spectral. 43–46, 155

IB immersed boundary. 3, 10–12, 14, 15, 31–36, 38, 40, 150

JONSWAP Joint North Sea Wave Observation Project. 149

JPD joint probability distribution. 111, 143

LC loose coupling. 9, 40, 43, 67, 74

LES large-eddy simulation. 6, 15–20, 45, 46, 67, 74, 133, 135, 136

MSL mean sea level. 110, 111, 143–145

NDBC National Data Buoy Center. 111

NREL National Renewable Energy Laboratory. 1, 17, 18

PFEM particle finite element method. 10, 15

PNW Pacific Northwest. 111, 133, 143

PPBC pressure correction boundary condition. 33, 35–39, 65, 66, 69, 70, 77, 79, 80, 83, 156

RANS Reynolds-Averaged Navier-Stokes. 5–8, 14, 17, 18

RAO response-amplitude operator. 108, 112, 124–128, 130, 131, 137, 159

RK2 second-order Runge-Kutta. 29

RK4 fourth-order Runge-Kutta. 45

SAFL Saint Anthony Falls Laboratory. 109, 110, 113, 126, 134, 160

SC strong coupling. 9, 40–43, 82

SGS sub-grid scale. 24, 28, 45

SNL Sandia National Laboratories. 108, 110

SPH smoothed-particle hydrodynamics. 10

TSR tip-speed ratio. 136

VOF volume of fluid. 13–15, 161

WEC wave energy converter. 1–3

WEC-Sim Wave Energy Converter Simulator. 108, 128–131, 159

WENO weighted essentially non-oscillatory. 28, 29

Chapter 1

Introduction

1.1 Motivation

The study of FSI problems involving waves and complex floating structures is currently a subject of intense research by the scientific community. Examples of application include floating oil platforms, ships, wave energy converter (WEC) devices, and offshore wind turbines. The fact that motivated the recent increased attention for studying such type of problems, however, has been the enormous potential seen on the ocean as a supply of unlimited and clean energy.

Many assessment studies have quantified the offshore energy potential, either by extracting kinetic energy from the wave motion with WECs, or by employing floating turbines to capture power from the wind (see a map distribution in figure 1.1). According to Gunn and Stock-Williams [3] the gross generating potential of the ocean waves is of approximately 2.11 TW, which would be nearly sufficient to satisfy the global electricity demand. The potential for the offshore winds is even more prominent as reflected by an offshore wind energy resource assessment published by the National Renewable Energy Laboratory (NREL). The study determined that the gross generating capacity of the U.S. offshore wind could produce four times the present overall electric capacity of the country [4].

Despite the potential of floating devices to capture part of the offshore resource, the state of the art for these technologies is in a very first stage of development and the current deployment is still limited. As of 2015 only two full scale floating turbines

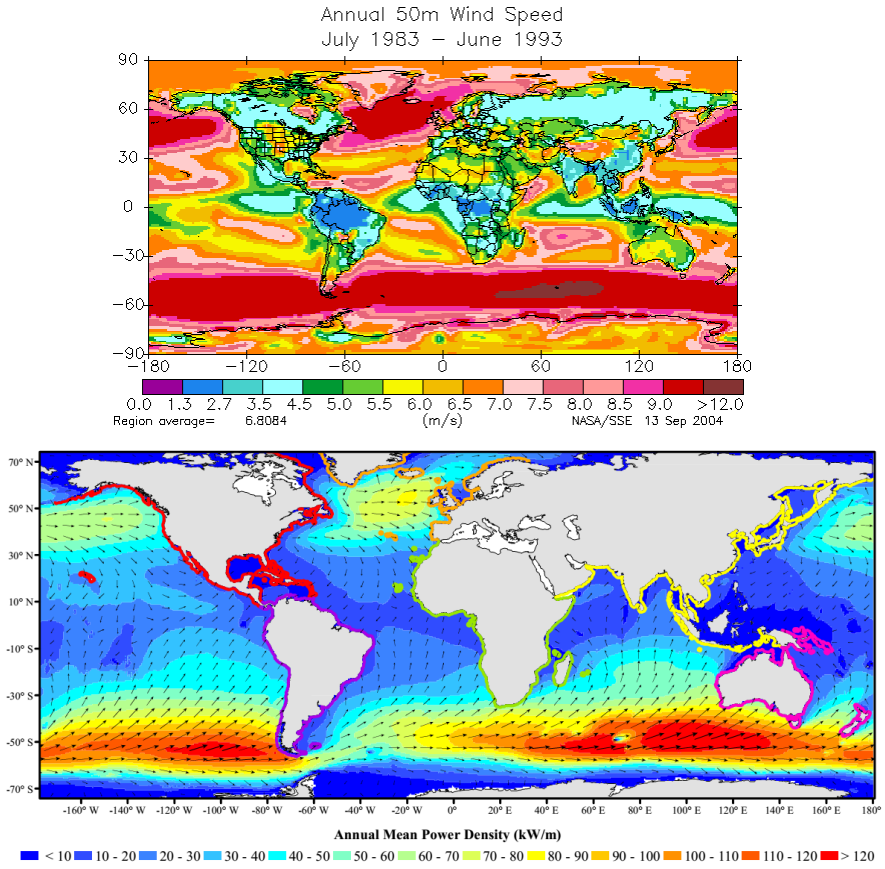


Figure 1.1: Distribution of annually averaged wind speeds by NASA (top), and anual mean wave energy density by [3] (bottom).

are operational, Hywind in Norway, and WindFloat in Portugal. Similarly, the global installed capacity for WEC devices is minimal and does not exceed 10MW [5].

Major technological advances are required for improving the performance and reliability of the offshore floating systems before the production cost can be reduced to competitive levels. There is an urgent need for new assessment tools such as numerical models, experiments and prototypes. In particular, high-fidelity numerical simulations can play an important role in the development of novel offshore technologies, and may be the only feasible way to tackle site-specific optimization of the designs.

A major challenge in the development of numerical methods for simulating floating devices is the need to resolve the flow around geometrically complex floating structures.

Moreover, the body undergoes large deformations resulting from the dynamic interaction of the flow with the 6 degrees of freedom response of the floating structure. This involves solving the linear and angular equations of motion in a coupled manner with the flow. When coupling the flow with the free body motions, problems in the stability of the solution algorithm may arise due to the so called added mass effect [6]. This is common when the density of the structures is similar or lower to that from the fluid. In such cases the problem can only be circumvented by using strong coupling techniques, which in turn, significantly increases the computational cost of the simulation.

Another major difficulty confronting the coupling of the fluid and structural domains is in the implementation of the FSI interface boundary conditions. IB methods are particularly suited for dealing with arbitrary complex structures subject to large deformations [7]. However, the body mesh is superposed on the fluid domain without matching with the underlying fluid mesh. In such methods (for example that developed by [6]), problems arise in the approach used to calculate the force imparted by the two fluids on the submerged structure. A numerical method has been proposed in this work to circumvent these limitations and simulate FSI problems with free surface flows.

Numerical modeling of floating structures is further complicated by the inherent two-phase nature of this type of flows and the non-linear effects of the air-water free surface interface. Complex non-linear phenomena such as wave breaking, water overtopping, or air-water entrainments, arises on the free surface. Only few available methods are capable for tracking such complex topological patterns, e.g. the volume of fluid [8], or the level set method [9]. In addition, flow around moving floating structure is mainly dominated by turbulence and non-linear viscous effects with areas of large separation and strong vorticity [10]. A common practice seen on the literature and widely adopted by the industry to simulate floating devices is to make the potential flow theory assumptions and consider the flow to be irrotational and inviscid (see for example [11, 12]). Such assumptions may only be reasonable for modeling scenarios with small amplitude motions of the floating body and low to moderate sea states with waves of low steepness. Floating devices, however, not always operate in regimes of low motion suitable for linear potential theory [13]. For example, WECs are commonly designed to work close to the resonance frequency to enhance its motion and increase the energy capture. The only approach that can accurately model most of these non-linear effects is through

the solution of the Navier-Stokes equations.

Another critical difficulty particularly challenging for studying floating devices located offshore stands from the large disparity in the spatial scales of the flow. While the scales of the structure are in the order of meters and require very fine meshes to resolve relevant features of the flow, the scales required for the site-specific meteorological and ocean wave effects are in the order of several kilometers. Splitting the problem in two coupled domains is a feasible solution, but requires development of algorithms for transferring the flow conditions from the two domains. An advantage of this approach is that a lower order model can be more efficiently applied in the large-scale domain. Transferring the flow velocity and free surface elevation between two computational models, which may be of different nature, requires complex numerical treatments (see for example [14]).

1.2 Literature Review

All these challenges can explain in part the lack of advanced numerical frameworks able to accurately simulate offshore real life floating structures under realistic ocean environmental conditions. In the rest of this section we provide a thorough review of the different modeling elements needed for the simulation of such type of problems. In addition, we also present the modeling techniques for the particular application of floating offshore wind turbines.

1.2.1 Numerical modelling of offshore environments

The action of the wind on the free surface of the ocean is responsible for the formation of complex waves fields composed of a broad range of frequencies and directions. Any element of the ocean, whether a ship, an oil platform, a floating turbine or the sand bed near the coast is subject to the effect of waves. Therefore, the study of waves is a fundamental topic that has been on the core of many studies in the last decades. We present in this section a review of the numerical methods available for simulating water waves.

Water waves modelling

Initially, modelling approaches for water waves were based on either the shallow water equations [15] or the Boussinesq equations [16]. Both methods are derived from depth-integrating the Navier-Stokes equations with the assumption that vertical velocities are small when the horizontal length scales are significantly larger than the vertical length scale. Under this assumption the dimension of the problem can be reduced, which makes the method computationally very efficient. Applicability of these methods is limited to specific cases with long wavelengths, such as tidal waves [16] and tsunami waves [17], or short dispersive waves in shallow waters. In addition, breaking waves phenomena, or air/water entrainment cannot be captured.

Alternatively, water waves can be modeled using potential flow theory which involves solving a Laplace equation. Although finite difference methods or finite element methods are both solution methods suitable for solving the potential flow governing equations, the boundary element method (BEM) excels for its high efficiency. In the BEM, Green's function is applied to obtain the boundary integral equations. The first work employing the BEM method was presented in 1976 by Longuet-Higgins and Cokelet [18]. Their method could successfully simulate non-linear overturning waves in deep water environments. Since then, it has been used in multiple studies [19, 20, 21, 22]. The main limitations intrinsic of potential flow methods, is that the flow is considered inviscid and irrotational, and therefore, cannot be used in cases of strong vorticity or in highly turbulent flows.

The most complete approach to model water waves that can include vertical components and non-linearities of the flow, and viscous and turbulent stresses is by solving the complete set of the Navier-Stokes equations. Turbulence in waves can be simulated accurately by employing direct numerical simulation (DNS). For example Lin et al. [23] developed an air-water Navier-Stokes DNS solver to simulate the process of wave generation and growth driven by wind. The study focused on low wind speeds and is limited to low wave slopes due to the linearization of the interfacial boundary condition.

Since DNS can be computationally prohibitive due to the high Reynolds number flow of most applications of engineering interest, Reynolds-Averaged Navier-Stokes (RANS) models have become very popular. Liu et al. [24] solved the RANS equations with the k-ϵ model to simulate wave interactions with porous structures. Repalle et al. [25]

used a RANS type approach to model the wave run-up around a spar-cylinder in a rectangular channel.

Alternatively, LES can be used resulting in better fidelity than RANS. In particular, the dynamic Smagorinsky subgrid-scale model has been applied in a wide range of wave modeling applications. Hieu et al. [26] applied it to simulated wave breaking on sloping bottoms. Ying et al. [27] simulated the ocean mixed layer with wave breaking and Langmuir circulation. Lubin et al. [28] studied 3D plunging breaking. Or the work of Yang and Stern [29] who proposed a method for carrying out LES of breaking waves induced by immersed bodies.

Wave makers for Navier-Stokes solvers

As discussed above, solving the full 3D Navier-Stokes equations is the only viable approach to overcome the limitations that most methods have in modeling non-linear wave phenomena such as turbulence, dispersion or wave overturning and breaking. One of the main challenges in applying Navier-Stokes solvers for modeling water wave is to deal with phenomena occurring at a disparate range of scales. The study of the interaction of a floating structure with swells is an example of this problem. While swells interact with the wind flow and evolve for long distances requiring a large computational domain in the scale of several kilometers, the floating structure highly depends on phenomena at a much smaller scales and requires the use of very fine meshes. An efficient approach to address this problem is to decompose the domain and use the most appropriate method at each of the regions (see [14, 30]). Considering that solving the Navier-Stokes equations is computationally very demanding, such method can be appropriate to study in high resolution the region where local phenomena are important, such as a floating structure, wave breaking dynamics, turbulence, etc. At the region with large-scale phenomena one can take advantage of the computational expedience of a lower order models (see figure 1.2 for a schematic representation).

A key aspect for developing such multi-scale methods is to choose the appropriate technique for transferring the wave and wind fields from the two domains. In particular, a major challenge in this regard is the approach for prescribing a specific large-scale wave environment as input into the 3D Navier-Stokes flow solver.

The simplest and most obvious way to incorporate waves into the computational

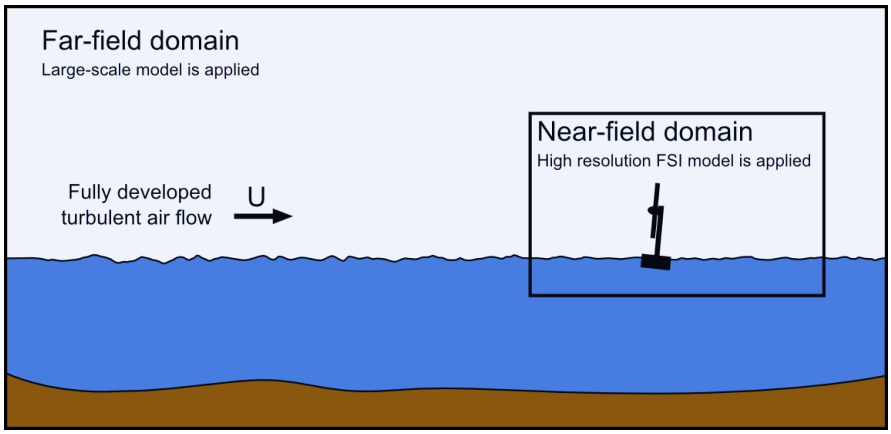


Figure 1.2: Schematic description of the far-field/near-field approach adopted for simulating offshore applications with large disparity of scales.

domain governed by Navier-Stokes is by directly specifying at the inlet boundary the velocity profile and surface elevation. For example, in the work of Colicchio et al. [14] an algorithm for coupling a potential flow based BEM solver with a Navier-Stokes levelset solver is presented. In Repalle et al. [25] theoretical wave velocities are fed into a RANS model to simulate the wave run-up on a spar cylinder. In Christensen [31], waves from a Boussinesq based model are incorporated to a Navier-Stokes solver.

Generation of waves by specifying inlet boundary conditions can be problematic when strong reflected waves reach the inlet boundary. For example Wei and Kirby [32] demonstrated that in long computations, even if a generating-absorbing boundary condition is employed, large errors accumulate and lead to inaccurate solutions.

An approach that can avoid the aforementioned difficulties when dealing with wave reflections is to employ an internal wave maker in combination with the use of sponge layers at the boundaries. The basic idea of internal wave generators is to apply an oscillatory force within an internal region of the domain known as the source region. The force is introduced by adding a source/sink term either in the continuity or momentum equations.

The internal wave generation method based on a mass source/sink was proposed in 1999 by Lin and Liu [33]. A general method for designing source function expressions was derived based on the assumption that all the increase/decrease of mass at the source region contributes to the target wave generation. Given a submerged rectangular source

region, the proposed method was used to obtain expressions for the following wave cases: linear monochromatic waves, irregular waves, Stokes waves, cnoidal waves, and solitary waves. The accuracy of the method was demonstrated by comparing the results to analytical solutions. It was also shown that the source term wave generator was not affected by the presence of reflected waves. Although this method has been widely used by many authors [34, 35, 36] it has yet to be extended to the generation of 3D directional waves.

A step further in the development of internal wave makers is to implement the source terms, not in the continuity equation but in momentum equations. It is not obvious, however, how to derive a forcing term expression that can generate a free surface wave pattern with the specific target amplitude, since the free surface elevation is not a variable in the momentum equation. Such relation, however, can be directly established in exact form in the depth-integrated Boussinesq equations as shown by Wei et al. [37]. In this work, source terms for the momentum Boussinesq equations were proposed for generating regular and irregular waves. The idea of Wei et al. was later implemented by Choi and Yoon [38] in a RANS turbulent model. In particular, the capabilities of the method to generate directional waves in a 3D basin were successfully demonstrated.

The above internal wave makers employ as source region a fixed rectangular domain located under the free surface. An alternative momentum source method is that proposed by of Guo and Shen [39] in which the source region is not fixed but follow the pattern of the free surface. This is equivalent of applying a surface pressure on the free surface which is similar to the physical process of wave generation by wind.

Wave reflections at the lateral boundaries are prevented by using a sponge layer method. It consists of adding a dissipation term in the momentum equation at the regions where waves are desired to be absorbed (sponge region). The classic sponge layer method was proposed in 1981 by Israeli and Orszag [40] and was based on the use of a viscous loss term known as Darcy term. An extension by Choi and Yoon [38] incorporated an additional term to account for inertial losses which may be useful for decreasing fluctuations cause by wave breaking.

1.2.2 FSI simulation of two-phase flows with floating structures

Numerical algorithms for handling two-phase flow FSI problems require integrating numerical techniques capable of: coupling the fluid and structural domains in a robust and efficient manner; simulating arbitrarily complex bodies undergoing motion with arbitrarily large amplitude; and handling non-linear free surface effects, such as wave breaking and structure overtopping.

FSI coupling algorithms can be broadly classified in two categories, monolithic and partitioned. In the monolithic approach the equations for the fluid and for the structural domains are both discretized and solved in a single system of equations. Although this approach is more robust than its partitioned counterpart, the leading system of equations is larger and ill-conditioned, and hence, computationally more demanding. Examples of applications are given in the following references [41, 42, 43, 44]. In contrast, in the partitioned approach each domain is treated separately, resulting in two independent problems with smaller and better conditioned systems of equations, which are coupled by imposing boundary conditions at its interfaces. Depending on the nature of the time integration technique, explicit or implicit, the algorithm is known as loose coupling (LC)-FSI or strong coupling (SC)-FSI, respectively. In the SC-FSI algorithm several sub-iterations are performed in each time step ensuring that a full convergence of the two domains is achieved, which makes it more expensive, but it is required in some situations where the LC-FSI algorithm leads to stability issues. For recent extensive reviews of various FSI algorithms, algorithmic advances and applications to a range of complex problems see Sotiropoulos and Yang [45], Yu, Baek and Karniadakis [46], Yang, Preidikman and Balaras [47], Borazjani and Sotiropoulos [6], etc.

There are three broad categories of numerical methods for handling flows with moving boundaries and/or interfaces: (1) moving grid methods; (2) mesh-free methods; and (3) fixed grid methods. In the moving grid methods, also known as the classic Arbitrary Lagrangian-Eulerian (ALE) approach, the mesh conforms to the moving boundary/interface at all time, and consequently boundary conditions on the body can be satisfied precisely. Examples of application of ALE methods can be found in [48, 49, 24, 50, 51]. In situations where the boundary and/or interface is arbitrarily complex, and/or undergoes large deformations, ALE methods become impractical. In

the particular case that the interface is the free surface, such as in [52, 53], the applicability of the ALE approach is limited to problems for which the free surface remains smooth and continuous. Therefore, this method is not able to handle complex phenomena such as splashing or breaking. Mesh free methods, on the other hand, were designed to circumvent the limitations of the body fitted methodologies in such complex situations. Two widely used methods in this category are the smoothed-particle hydrodynamics (SPH) applied for example in [54, 55] or the particle finite element method (PFEM) in [56, 57]. In spite of significant recent advances, however, mesh-free methods are still computationally very demanding. The last category of methods are IB methods which were first introduced by Peskin [58] to study the blood flow in the heart. The key idea of this class of methods is to account for the effect of the boundary/interface by introducing a fictitious force field in the Navier-Stokes equations that are solved on the background, non-boundary-conforming grid. Peskin's initial approach can be classified as a diffused IB method as it uses a discrete delta function to smear the body force and thus the effect of the immersed body over several grid nodes. Diffused IB methods generally require a higher mesh resolution near the boundary/interface in order to produce accurate solutions. More recently, sharp-interface immersed boundary methods have emerged from the need to circumvent this limitation. Such methods include the cut-cell method, the immersed interface method, and the hybrid-Cartesian/immersed boundary (HCIB) method. The cut-cell method proposed by Clarke [59] is based on the idea of modifying the cells near the boundary/interface so that they conform to its geometry. The solution resulting from cut-cell methods is highly accurate since the mass is appropriately conserved near the interface. However, application of the method to three-dimensional problems is not always straightforward an often require special treatments, in particular when dealing with geometrically complex three-dimensional boundaries/interfaces (see for example [60, 61, 62]). The immersed interface method was first introduced by Leveque and Li [63] and consists of applying a singular force that introduces a jump conditions across the interface. Examples of applications of the immersed interface method for moving boundaries are given in [64, 65]. In the HCIB method, proposed by Mohd-Yusof [66], the fictitious force added to account for the effect of the interface is introduced implicitly by imposing velocity boundary conditions at the grid nodes located at the vicinity of the interface (referred to as IB nodes). This

approach has been used successfully to simulate a number of complex problems in engineering and biology [67, 68, 69, 70, 71, 72, 73, 74]. The HCIB approach has also been extended to generalized curvilinear grids by Ge and Sotiropoulos [7] who proposed the CURVIB in which the governing equations may be solved on a background curvilinear grid. The CURVIB method has the ability to fit the fluid mesh to relatively simple boundaries of the computational domain while still maintaining the ability to introduce immersed boundaries. This is particularly interesting for solving biological flows [70, 67] or in environmental applications [75, 76]. More recently the CURVIB approach has also been extended to incorporate domain decomposition with overset, Chimera grids [77]. For an extensive overview of IB methods we refer the reader to the recent review papers of Sotiropoulos and Yang [45] and Mittal and Iaccarino [78].

A major difficulty of IB methods that is particularly critical in FSI applications arise from the approach used to impose the boundary conditions between the fluid domain, represented by the Eulerian mesh, and the solid body which is typically tracked with a Lagrangian mesh. In the classical IB approach of Peskin the force field is calculated from constitutive laws and imposed on the background grid by employing a discrete delta function. The displacements of the structure follow the motion of the surrounding fluid. In other variations of the IB method such as the HCIB method proposed by Mohd-Yusof [66], the fictitious force added to account for the effect of the interface is not computed explicitly but is introduced implicitly by imposing velocity boundary conditions at the grid nodes located in the vicinity of the interface (referred to as IB nodes). To then solve the equations of motion of the moving body, the forces and moments that the fluid exerts to the structure need to be computed by integrating the pressure and shear stresses either directly at the background mesh or at the body surface via extrapolation. In the cut-cell method proposed by Clarke [59] the cells near the boundary are modified so that they conform to its geometry and the velocity boundary condition can be imposed with high accuracy. However, similar to the HCIB, the forces of the fluid acting on the body need to be calculated with special treatments.

The most straightforward technique for computing the forces and moments acting on the structure is by projecting the pressure and shear stresses to all the elements of the Lagrangian mesh of the body and perform a subsequent integration along the surface. Such an approach has been widely applied in single-phase flow problems [79, 80, 81]

as well as in FSI applications with floating structures [82]. In [81] the force at each material point of the structural mesh was calculated by applying inversed distance weight coefficients to the stresses of the nearest IB nodes. Alternatively, Haeri and Shrimpton [83] proposed a method in which two equally spaced auxiliary points located in the wall normal line centered on each material element are defined. In these auxiliary points the stress tensor is reconstructed by interpolating from eight surrounding fluid cells. Finally, the auxiliary points are used to extrapolate with second order accuracy the values of the stress tensor at the center of each Lagrangian element.

The computational cost of the aforementioned projection techniques is considerable and implementation of such algorithms in parallel computing is not straightforward. Several authors have proposed alternative methods to simplify the calculation of the forces and torques. For example, Balaras [84] proposed a method applicable to moving bodies based on the idea of Lai and Peskin [85] in which the stresses are integrated directly at the underlying fluid mesh along a rectangular bounding box. With such an approach there is no need for performing any projection step and the parallelization of the algorithm is straightforward. However, the force calculation expression contains a term that involves integration within the domain inside the bounding box and exterior to the body. This term becomes difficult to compute for geometrically complex bodies and/or moving bodies. Shen et al. [86] extended the method of Balaras to ease the implementation difficulties in complex and moving bodies by splitting the force in two parts, one representing the flow external to the bounding box, and a second representing the virtual flow inside the domain of the body. In the work of Sanders et al. [87] the solid domain is filled with a fictitious fluid that is forced to move at the same velocity as the solid body. The forcing terms to be applied in the equation of motion are obtained by integration within the interior of the solid domain. Borazjani et al. [6] proposed an approach that is also based on integrating the stresses directly at the Eulerian fluid mesh (details of the algorithm are given in [88]). In this case, however, instead of using the surface of a bounding box, the integration is performed along the surface that is constructed by the fluid cells located in the immediate vicinity of the body.

When a fixed grid approach is adopted for dealing with the presence of a either a body or the free surface it is important to consider a tracking method relating the location of the moving interface to the underling Eulerian mesh. The following two methods

are available: (1) the front tracking method; and (2) the front capturing method. In the front tracking method the description of the interface is established in a Lagrangian manner through an additional grid superposed on the fluid mesh. The advantage of this approach over the front capturing method is its higher accuracy as it treats the interface as a sharp discontinuity. Examples of application of the method are given in [79] for tracking immersed bodies and in [89, 90] for simple configurations of the interface of a two immiscible fluids. When this approach is applied to complex three-dimensional two-phase flows with complex free surface motions, however, it becomes computationally demanding and impractical to use. In such situations, a front capturing approach is better suited. The front capturing method tracks the motion of the interface via a scalar marker function defined on the background Eulerian mesh. Although it is not as accurate as the front tracking method it is the most popular method for dealing with two-phase flows as it can handle with ease complex situations such as breaking waves or air/water entrainments. Two popular front capturing methods are: (1) the volume of fluid (VOF) method of Hirt and Nichols [8]; and (2) the level set (LS) method of Osher and Sethian [9]. The VOF method uses as a marker function the volume fraction of water contained in each computational cell, and as a result it is conservative in nature allowing the interface to be advected without any loss of mass in any of the two fluids. The limitation of such approach is that the marker function does not define a clear interface, rather it has to be reconstructed through an special treatment. A review of methods for interface reconstruction are given in [91]. In the LS method the scalar marker function or distance function is defined in each computational cell as the signed distance to the closest point contained in the interface. It is obvious from the definition that the exact location of the interface is explicitly known, as it always coincides with the zero value of the market function, and therefore there is no need for reconstructing the interface as in the VOF method. We note that the reconstruction step can be quite challenging in three-dimensional applications, and may be prohibitively complex in a generalized curvilinear framework. Iafrati [92] applied the level set method to simulate the free surface dynamics of the 2D wave breaking phenomena. Yue et al. [93] simulated a 3D dam breaking problem, demonstrating the capability of the level set method to capture complex free surface non-linear phenomena such as splashing of the surge front

and air entrainment in the water. Liu et al. [94] employed the level set method to simulate droplets impacting the surface of a geometrically complex body. Carrica et al. [95] presented a single phase level set approach and applied it to simulate diffracted waves induced by a ship. Kang and Sotiropoulos [96] developed a level set approach capable of simulating 3D two-phase turbulent free surface flows using the unsteady RANS in the CURVIB context, and were able to predict the water surface elevations over complex hydraulic structures. For a comprehensive overview of level set approaches the reader is referred to the review paper of Sethian and Smereka [97]. A technique that takes advantage of both the VOF mass conserving properties and the LS accuracy to locate the interface is to couple these two methods (CLSVOF) such as in [98, 99]. Nevertheless, this approach does not eliminate the need to reconstruct the interface at every time step.

The numerical simulation of rigid bodies interacting with a free surface is a challenging task and for the most part previous studies have not considered all the physical aspects required for studying such a complicated problem. Some of these studies treat the problem as a single-phase flow, i.e. instead of solving for the two phases, only the water side is computed while the effect of the air is introduced by satisfying the kinematic and dynamic boundary conditions on the free surface. An example is the work of Kleefsman et al. [82] which employs a finite volume approach to solve the incompressible Navier-Stokes equations with Cartesian fixed grids, and the cut cell method for dealing with the moving bodies. The free surface interface is tracked using the VOF method with a local height function. The method was validated for a damp break problem, and bodies impinging the free surface such as a free falling wedge. Tanaka et al. [100] uses a single phase approach with an ALE finite element method with the introduction of an additional background mesh that helps in the process of re-generating the body fitted mesh. With this technique the method allows large motions of the body and the free surface. The method is validated for a floating cylinder and a falling wedge. The work of Paik [101] employs the CFDShip-Iowa flow solver of [95], which is based on RANS models. The free surface is modeled with a single-phase level set approach, and the large amplitude motions of the structure are treated with a dynamic overset grid. The method is applied to simulate a container ship in regular waves, as well as a flexible bar in a sloshing tank.

A two-phase flow formulation can be found in the work of Sanders et al. [87] who employed a fixed grid finite difference IB approach combined with the level set method for tracking the free surface interface. The method is validated by simulating a heave decay test of a cylinder and a roll decay test of a rectangular barge, and then applied to simulate a floating buoy. The method has yet to be extended to three-dimensional applications. Similarly, Shen and Chan [102] presented a two-phase flow solver combining the IB method and the VOF. The method was applied to two-dimensional problems such as the propagation of waves over a submerged body, or waves induced by a moving bed. Walhorn et al. [103] developed a monolithic FSI approach to handle two-phase flows with flexible bodies using a finite element discretization. The bodies where handled with moving mesh and the free surface by using the level set method. The proposed model was applied to the 2D case of a damn break interacting with a flexible wall. Ryzhakov et al. [104] proposed another monolithic FSI approach for two-phase flow and flexible bodies combining the PFEM with the idea of quasi-incompressible fluids. Both the fluid and the structure are tracked in a Lagrangian manner, and the method is applied to simulate the case of damn break next to a flexible wall, and an elastic plate deforming by the effect of water pressure.

The work that is most similar to the numerical method presented in this thesis is that of Yang et al. [10] who proposed a sharp interface immersed boundary method for carrying out LES of two-phase turbulent flows with a level set free surface implementation. The method was applied to simulate the interaction of the free surface interface with bodies undergoing prescribed motion, such as a water entry/exit of a body, landslide induced waves, or the hydrodynamics of a ship. This work however has not been extended to fluid-structure interaction problems.

1.2.3 Simulating offshore floating wind turbines

From a numerical viewpoint, simulating a land-based wind turbine is already a challenging task as it involves solving highly turbulent atmospheric wind flow around a geometrically complex moving body. Placing turbines offshore on a floating platform not only has the added complexity of facing stronger winds that are affected by the wave patterns, but are also subject to 6 DoF FSI motions resulting from the wave-turbine and wind-turbine interactions. Furthermore, the two-phase nature of such flows

is characterized by complex non-linear effects such as wave breaking.

Due to all these complexities, most numerical models in the literature for simulating floating wind turbines are based on oversimplified assumptions. A first group of models which are generally applied in preliminary floating turbine feasibility studies, evaluate the floating system response to wind and waves by solving the rigid body equations of motion in the frequency-domain [105, 106]. These models consume very low computational resources but cannot capture key features of the problem such as transients or nonlinearities as well as wave radiation or diffraction.

A second group of models that can capture transient effects, and thus deliver an overall better agreement, solve the equation of motion in the time domain [107, 108, 109]. Wave loading however is introduce to the solver by the use of the semi-empirical Morison's equation. In addition the flow solver assumes inviscid and irrotational flow.

Gueydon et al. [13] simulated a laboratory scale floating turbine by using a potential flow solver. Wave loads were calculated by integrating the pressure distribution at the frequency domain, and wind loads were taken from force coefficients. The numerical model comparison with the experiment agreed fairly well in predicting the wave responses in a design sea state, but failed to reproduce the response in operational sea state condition.

Sebastian and Lackner [11] using a BEM theory simulation model demonstrated that the flow in offshore wind turbines is far more complex than in land-based or fixed offshore wind turbines. It also showed that low order models such as BEM or dynamic-inflow-based models are not intended for solving such complex problem. As a consequence, high fidelity models are a required tool for accurately modeling floating wind turbines.

Yang et al. [110, 111] recently simulated the effect of swells on infinitely long offshore wind farm by proposing a method based on that from [1, 112] that combines a LES solver for the air flow and a potential flow based wave solver for the water field. The turbines were assumed to be fixed and the rotors were represented as actuators disks. The flow statistics and power output of different wave and wind cases as well as different turbine spacing were investigated.

1.2.4 Turbine rotor modelling

There are basically two main approaches that are suitable for simulating the rotor of a wind turbine, (1) the rotor resolving model, and (2) the rotor parameterization model. The former class of models employ a mesh that is sufficiently fine to resolve the geometric configuration of the turbine as well as the boundary layer attached to the walls of the blades. In the latter, the effect of the turbine is introduced in the simulation, not by solving the actual geometry of the blades, but by extracting from the flow the equivalent amount of energy. The rotor resolving model is the approach that delivers the best accuracy, as the amount of modeling is minimal, and therefore it can provide the flow physics necessary for turbine specific optimization. Its high computational cost, however, limits the applicability to only single turbine studies. The computational expedience of rotor parameterization makes it the optimum approach for farm-scale studies allowing to simulate several arrays of turbines.

Rotor resolving models

The first available study attempting to fully resolve a wind turbine rotor is the work of Sorensen and Hansen [113] in 1998. RANS equations solved in a rotating frame of reference with a shear-stress transport (SST) k-w eddy viscosity model were employed to simulate a series of isolated rotor problems in operational wind conditions. The method was shown to be accurate in predicting the power production at low wind speeds (below 10m/s), however, due to the grid resolution and turbulence model limitation, the method was not accurate when applied to stronger winds. In 2002, Sorensen et al. [114] published a similar study applying a RANS model to simulate another isolated turbine rotor. The reported data focused on analyzing the dependence of the incoming wind velocity on the following quantities: shaft torque, flap and edge moments, and aerodynamic coefficients.

Johansen et al. [115] implemented a Detached-Eddy Simulation model (DES) to study the flow around the NREL Phase-VI turbine blade under no rotation. The basic idea of DES is to use a RANS model to solve the boundary layer and LES for the rest of the computational domain. No significant improvement over classical RANS model was observed in the computation of the blade characteristics. The same turbine rotor was

studied in 2006 by Sezer-Uzol et al. [116]. In this study the full rotor was considered under rotation by attaching the unstructured grid to the blade motion. The method employed a three-dimensional, time accurate, finite volume inviscid flow solver. This flow solver was later extended to viscous simulations by incorporating a LES turbulent model with wall function in [117]. Comparison of the inviscid and turbulent solutions were presented with experimental data.

In 2009, Zahle et al. [118] employed a finite volume RANS model with overset grids to simulate not just the rotor, of the aforementioned NREL Phase VI wind turbine, but the tower and tunnel floor as well. The method was validated with experimental data and was shown to captures blade-tower wake interactions.

The finite element FSI approach of Bazilevs et al. [119, 120] can account for the aeroelastic effects of a full scale turbine rotor. The air field was simulated with a residual-based variational multiscale model and was couple with a rotation-free Kirchhoff-Love based shell model for the rotor blades. The primary limitations of the model is that it cannot incorporate the full turbine configuration, and cannot account for ambient atmospheric turbulence.

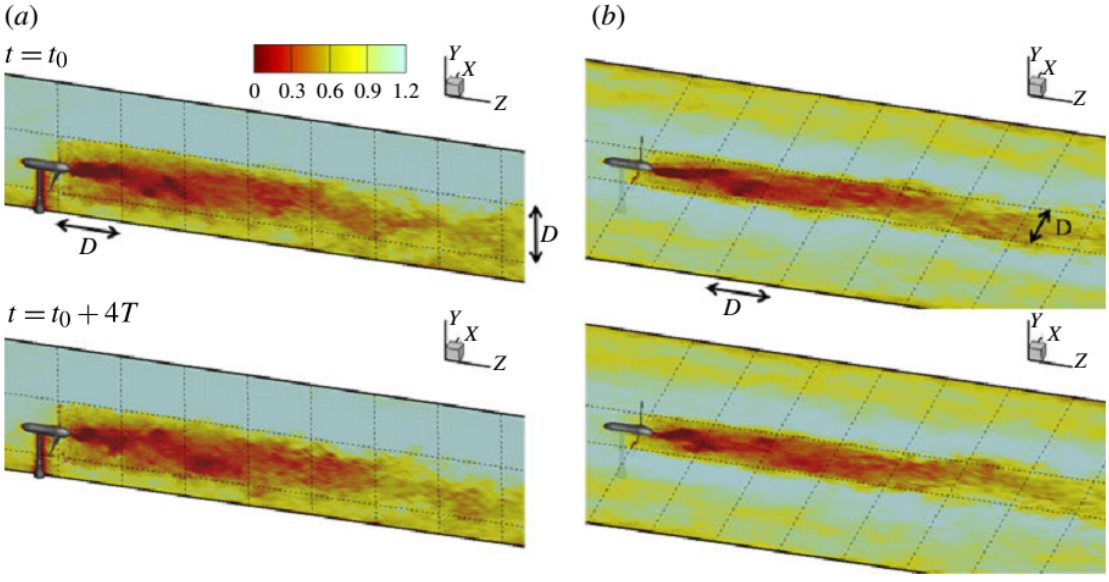


Figure 1.3: Contour plots at two different times of the instantaneous streamwise velocity computed with the turbine resolving model based on the CURVIB method. T represents the period of a full rotor rotation. Reproduced from [121].

Kang et al. [96] applied the CURVIB method [7, 122] to carry out high-resolution LES of a real marine hydrokinetic turbine mounted on the bed of a rectangular channel. A wall modeling strategy was used to account for the boundary layer effects near the turbine geometry. A very fine simulation of 185 million grid nodes, of the complete turbine geometry, including the rotor, nacelle and pylon, allowed to observe for the first time the three-dimensional wake structure downstream of the turbine. In a later work Kang et al. [121], presented a further study of the same hydrokinetic turbine elucidating in more detail the turbulent structure of the wake and explaining the mechanisms responsible for wake meandering (see figure 1.3).

Although the studies of [96, 121] involved a hydrokinetic turbine, it is noteworthy top mention that CURVIB framework can be directly applied without any further development to simulate the fully resolved case of a complete wind turbine configuration.

Rotor parametrizations: actuator disk and actuator line

The rotor parametrization approach approximates the effect that the turbine rotor has to the flow by introducing a sink term S_{AD} in the momentum equation. Depending on how the sink term is spatially distributed in the flow the parametrization can be classified as actuator disk model, or actuator line. In the actuator disk the sink is applied in the form of a permeable disk, and its force is usually considered uniformly distributed adopting the following form

$$S_{AD} = \frac{1}{2}\rho U_0^2 C_T, \tag{1.1}$$

where U_0 is the wind incoming velocity at the disk, a is the induction factor of the rotor, and C_T is the thrust coefficient computed as follows

$$C_T = 4a(1-a). \tag{1.2}$$

One of the first works employing the actuator disk to simulate the wake of a turbulent flow around a turbine is that of Sorensen et al. [123]. The above constant loading actuator disk distribution was implemented in combination with a Navier-Stokes flow solver expressed in vorticity-velocity variables.

Alternatively, the force in the actuator disk can be distributed in a non-uniform

manner. In the work of Masson et al. [124] for example, the force in the disk depends on the distance to the hub center, and its variation is computed using blade element theory.

Calaf et al. [125] applied the actuator disk model to carry out LES of a large wind farm in a fully developed wind-turbine array boundary layer. Several turbine spatial arrangements, turbine loading factors and surface roughness were investigated to estimate vertical momentum and kinetic energy transfer. A similar study by Yang et al. [126] presented an LES model with parametrization of the turbine rotor with actuator disks. The study analyzed how the stream-wise and span-wise spacing of a wind farm arrays can influence the overall power output. Another LES study by Wu and Port-Agel [127] showed that including rotation in the actuator disk model can improve the computed results at the near-wake region of a wind turbine.

On the other hand, in the actuator line model each of the rotor blades is parametrized with a sink distributed along a line that rotates following the rotor motion. The actuator line can be seen as a particular case of the actuator disk with non-uniform force distribution. The method was proposed by Sorensen et al. [128] in 2002. The force distributions at the lines representing the blades were computed iteratively with the combination of airfoil data and the blade-element technique. The method which was successfully validated with experimental power production data, showed some interesting features of the flow field behind the turbine such as tip vortices.

The study of Kang et al. [121] assessed the accuracy of the two parametrization models, actuator disk and actuator line, by comparing its results with those from the fully resolving model and experimental data. Although the actuator line was shown to be in overall better agreement with the experiments than the actuator disk, significant differences were still observed.

For a further review of the different approaches for simulating wind turbine rotors and wakes the reader is referred to the work of Sanders et al. [129]

1.3 Thesis objectives and outlines

The literature review presented above for modeling offshore floating structures points to the following assessment of the state-of-the-art:

- Most studies previously applied to simulate floating offshore structures, such as floating turbines, are based on oversimplified assumptions. Generally, these studies assume inviscid and irrotational flow and account for the wave forcing by using the semi-empirical Morison equation.

- High-resolution and high fidelity numerical methods based on Navier-Stokes for simulating some aspects of the offshore floating structure problem have been developed. However, to the best of our knowledge, a numerical methodology that is powerful enough to simulate the coupled dynamics of the floating structure with wind and waves do not exist. Methods that account for far field wave phenomena on near-field floating structure FSI also do not exist.

The objective of this thesis is to develop a powerful numerical framework capable of simulating in high fidelity the coupled interactions of a complex floating structure with an offshore wave/atmospheric-turbulence environment taking into account far-field wave effects on near-field structure dynamics. This work also seeks to demonstrate the benefits of such computational method in comparison to other computational packages based on lower order assumptions that are typically used by the offshore industry. Such computational framework can be applied to many offshore and nearshore applications providing relevant insights for designing optimized and more robust technologies. It can also be applied to develop and test physics-based, low-dimensional dynamic models of floating structures. The specific goals of this work are as follows:

1. Develop a FSI model for simulating arbitrarily complex floating rigid bodies interacting with nonlinear free-surface flows.

2. Validate and demonstrate the predictive capabilities of the FSI method and its ability to simulate non-linear free-surface phenomena, such as breaking waves, and apply it to simulate various cases involving 2D/3D free surface-rigid body interactions.

3. Implement and validate a wave maker method for generating multiple sets of three-dimensional directional waves.

4. Couple the FSI model with a large-scale wave model to efficiently incorporate realistic ocean wave and wind conditions.

5. Validate the coupled model with experimental results from a laboratory scale floating turbine and compare the results with other lower order models.

6. Apply the computational framework to simulate the structural dynamics and flow physics of a full scale floating turbine operating in site-specific ocean conditions.

The thesis is organized as follows

- Chapter 2 details the numerical methods of the near-field flow solver for simulating complex floating structures, the far-field flow solver for developing wind and wave offshore conditions, and the near-field/far-field coupling algorithms.

- Chapter 3 presents various test cases for validating the capabilities of the near-field FSI model to study complex structures interacting with two-phase free surface flows.

- Chapter 4 reports results validating the pressure forcing method for various wave cases including far-field/near-field coupling.

- Chapter 5 discusses the application of the model to study wave-body interactions of an offshore floating wind turbine model.

- Chapter 6 addressees the case of an offshore wind turbine model with prescribed oscillatory motion in the pitch DoF.

- Chapter 7 report simulation results of a larges-scale offshore floating wind turbine responding in 6 DoF as a results of site-specific offshore wind and wave conditions.

- Chapter 8 provides a summary and the main conclusions of this work and discusses future areas of research.

- Appendix A presents a manual for using the computational framework, which has been publicly released.

Chapter 2

Governing equations and numerical methods

The numerical framework couples an efficient large-scale model, which is referred in this work to as the far-field flow solver, and is suitable for simulating realistic ocean wave and wind conditions, with a high resolution near-field model capable of solving complex free-surface flows interacting non-linearly with arbitrarily complex real life floating structures.

2.1 The near-field flow solver

2.1.1 The two-phase Navier-Stokes equations

The near-field model solves the spatially-filtered Navier-Stokes equations governing incompressible flows of two immiscible fluids in non-orthogonal generalized curvilinear coordinates. We adopt the two-fluid, level set formulation of Kang and Sotiropoulos [96], where a single equation is used in all the computational domain taking the corresponding fluid properties values in each fluid phase.

Using compact Newton notation, where repeated indices imply summation, the equations read as follows ($i, j, k, l = 1, 2, 3$):

$$J\frac{\partial U^j}{\partial \xi^j} = 0, \tag{2.1}$$

$$\frac{1}{J}\frac{\partial U^i}{\partial t} = \frac{\xi_l^i}{J}\left(-\frac{\partial}{\partial \xi^j}\left(U^j u_l\right) + \frac{1}{\rho\left(\phi\right) Re}\frac{\partial}{\partial \xi^j}\left(\mu\left(\phi\right)\frac{\xi_l^j \xi_l^k}{J}\frac{\partial u_l}{\partial \xi^k}\right) - \frac{1}{\rho\left(\phi\right)}\frac{\partial}{\partial \xi^j}\left(\frac{\xi_l^j p}{J}\right) - \right.$$
$$\left. - \frac{1}{\rho\left(\phi\right)}\frac{\partial \tau_{lj}}{\partial \xi^j} - \frac{\kappa}{\rho(\phi) We^2}\frac{\partial h(\phi)}{\partial \xi^j} + \frac{\delta_{i2}}{Fr^2} + S_l^w + S_l^s + S_l^{AL}\right), \tag{2.2}$$

In the above equations: ϕ is the level set function (see below for details), ξ^k are the curvilinear components, ξ_l^i are the transformation metrics, J is the Jacobian of the transformation, U^i are the contravariant volume fluxes, u_i are the Cartesian velocity components, ρ is the density, μ is the dynamic viscosity, p is the pressure, τ_{lj} is the sub-grid scale (SGS) tensor, κ is the curvature of the interface, δ_{ij} is the Kronecker delta, h is the smoothed Heaviside function, S_i^w is the source term for wave generation, S_i^s is the source term for wave dissipation, S_l^{AL} is the actuator line body force, and Re, Fr, and We are the dimensionless Reynolds, Froude, and Weber numbers, respectively, defined as follows:

$$Re = \frac{UL\rho_{water}}{\mu_{water}}, Fr = \frac{U}{\sqrt{gL}}, We = U\sqrt{\frac{\rho_{water} L}{\sigma}} \tag{2.3}$$

where, U and L are the characteristic velocity and linear dimension, ρ_{water} and μ_{water}, are the density and dynamic viscosity of the water phase, g is the gravity, and σ is the surface tension. The subgrid scale stress tensor τ_{lj} in Eq. (2.2) is modeled in the present work as described in [122] using the dynamic Smagorinsky eddy viscosity model of [130].

In the level set method, the interface is tracked using the signed distance function $\phi(x,t)$, also known as the level set function, which is an scalar function defined in the whole computational domain, measuring the minimum distance from any point x in the fluid to the closest point of the free surface interface. The interface is located at the level $\phi = 0$, and the sign is positive in the liquid phase, and negative in the gas phase.

The jump condition of the density and viscosity fields at the interface in a level set approach is taken to be continuous, and is smeared over a thin layer of thickness 2ϵ to prevent the formation of numerical instabilities. It can be expressed as follows:

$$\rho\left(\phi\right) = \rho_{air} + \left(\rho_{water} - \rho_{air}\right) h\left(\phi\right), \tag{2.4}$$

$$\mu(\phi) = \mu_{air} + (\mu_{water} - \mu_{air})\, h(\phi), \tag{2.5}$$

where the smoothed Heaviside function [131] $h(\phi)$ is

$$h(\phi) = \begin{cases} 0 & \phi < -\epsilon, \\ \frac{1}{2} + \frac{\phi}{2\epsilon} + \frac{1}{2\pi}\sin(\frac{\pi\phi}{\epsilon}) & -\epsilon \leq \phi \leq \epsilon, \\ 1 & \epsilon < \phi, \end{cases} \tag{2.6}$$

Typical values for ϵ are between one and three times the length of the smallest grid cell. Our experience shows, however, that for problems in which the interface undergoes rapid deformations due to complex phenomena such us air/water entrainments or wave breaking, it is necessary to employ larger values of ϵ that can be up to the length of six or eight grid cells for the most extreme scenarios. The need for an increased number of grid cells within the transition region has been investigated and discussed in [132, 92]. The study of Iafrati and Campana[132] shows that spurious velocity effects are introduced in the interior of the transition layer caused by the smearing of the interface. If the thickness ϵ of the layer is sufficiently large the spurious effects remain confined within the transition layer without altering the exterior flow. It was also shown that larger interface thickness is required for increased Reynolds number flows. The kinematic and dynamic boundary conditions ensuring continuity of the velocity and the normal and tangential stresses at the interface, are intrinsic in the current formulation and are satisfied in a smooth manner.

2.1.2 The Level set equations

The motion of the free surface interface can be modeled by the level set method proposed by Osher and Sethian [9]. The spatially filtered advection equations in generalized curvilinear grids will assume the form:

$$\frac{1}{J}\frac{\partial \phi}{\partial t} + U^j \frac{\partial \phi}{\partial \xi^j} = -\tau_{ij}^L, \tag{2.7}$$

where τ_{ij}^L is the sub-grid scale stress tensor responsible of the effect of the unresolved subgrid scales on the level set field. In the present model the effect of τ_{ij}^L is neglected

assuming that the residual field of ϕ is small and its overall contribution to the energy containing scales is negligible.

As Eqn. (2.7) is integrated in time to determine ϕ, there is no guarantee that the resulting solution will satisfy the required, for a distance function, unit gradient condition $|\nabla\phi| = 1$. Such incosistent solution will in turn lead to poor conservation of mass between the two fluids. This problem is circumvented by solving the following mass conserving re-initialization equation proposed by Sussman and Fatemi [133]:

$$\frac{\partial\phi}{\partial\tau} + S(\phi_0)(|\nabla\phi| - 1) = \lambda\tilde{\delta}(\phi)\,|\nabla\phi|\,, \tag{2.8}$$

where τ denotes a pseudo-time, ϕ_0 the distance function at the initial step of the pseudo-time iteration procedure, $S(\phi_0)$ is the smoothed sign function defined as

$$S(\phi_0) = \begin{cases} 1 & \phi_0 \geq \epsilon, \\ -1 & \phi_0 \leq -\epsilon, \\ \frac{\phi_0}{\epsilon} - \frac{1}{\pi}\sin\left(\frac{\pi\phi_0}{\epsilon}\right) & \text{otherwise}, \end{cases} \tag{2.9}$$

$\tilde{\delta}(\phi)$ is the smoothed delta function defined as $\tilde{\delta}(\phi)$ is the smoothed delta function given as

$$\tilde{\delta}(\phi) = \begin{cases} \frac{1}{2\epsilon}\left(1 + \cos\left(\frac{\pi\phi}{\epsilon}\right)\right) & |\phi| \leq \epsilon, \\ 0 & \text{otherwise}, \end{cases} \tag{2.10}$$

and,

$$\lambda = -\frac{\int_\Omega \tilde{\delta}(\phi)S(\phi_0)(1 - |\nabla\phi|)d\Omega}{\int_\Omega \tilde{\delta}^2(\phi)\,|\nabla\phi|\,d\Omega}, \tag{2.11}$$

being Ω the volume of a grid cell. A detailed description of the method in the context of curvilinear coordinates can be found in Kang and Sotiropoulos [96].

2.1.3 Equations of motion for rigid bodies

For computing the general 6 DoF motion of 3D rigid bodies we use the Lagrangian form of Newton's second law, i.e. the linear and angular momentum equations. With no loss of generality we write the equations for a single body, although the formulation can be extended to multiple bodies, expressed along the principle axes. Under the above

assumptions, the general form of the equations for a body mounted on an elastic and damped system can be written in the inertial frame of reference in the following form (i=1,2,3):

$$M\frac{\partial^2 X^i}{\partial t^2} + b_{t,i}\frac{\partial X^i}{\partial t} + k_{t,i}X^i = F_{fluid,i} + F_{ext,i} \tag{2.12}$$

$$I_i\frac{\partial^2 \Theta^i}{\partial t^2} + b_{r,i}\frac{\partial \Theta^i}{\partial t} + k_{r,i}\Theta^i = M_{fluid,i} + M_{ext,i} \tag{2.13}$$

where Eq. (2.12) represents the pure translation motion, and X^i are the components of the position vector, M is the mass of the structure, $b_{t,i}$ the damping coefficients, $k_{t,i}$ the spring stiffness coefficients, $F_{fluid,i}$ the components of the force exerted by the fluid, and $F_{ext,i}$ the external forces. The case of pure rotation is represented by Eq. (2.13), and Θ^i denote the components of the relative angle of rotation vector, I_i the moment of inertia, $b_{r,i}$ the damping coefficients, $k_{r,i}$ the spring stiffness coefficients, and $M_{fluid,i}$ and $M_{ext,i}$ the moments in respect to the axis of rotation exerted respectively by the fluid, and external forces. For 6 DoF motions the two sets of the equations, linear and angular, have to be solved concurrently.

The solid and fluid domains are coupled together via the Dirichlet boundary condition that the fluid velocity field must satisfy on the surface Γ of the body, as follows:

$$u_i = \frac{\partial X^i}{\partial t} + \varepsilon_{ijk} r_j \frac{\partial \Theta^k}{\partial t} \tag{2.14}$$

The system of second order ordinary differential equations (2.12) and (2.13) that govern the motion of the structure is first transformed into following system of first order ordinary differential equations

$$\frac{\partial X^i}{\partial t} = Y^i \tag{2.15}$$

$$\frac{\partial \Theta^i}{\partial t} = \Psi^i \tag{2.16}$$

$$\frac{\partial Y^i}{\partial t} + \frac{b_{t,i}}{M}Y^i + \frac{k_{t,i}}{M}X^i = \frac{F_{fluid,i} + F_{ext,i}}{M} \tag{2.17}$$

$$\frac{\partial \Psi^i}{\partial t} + \frac{b_{r,i}}{I_i}\Psi^i + \frac{k_{r,i}}{I_i}\Theta^i = \frac{M_{fluid,i} + M_{ext,i}}{I_i} \tag{2.18}$$

Finally, it is integrated in time (see [6] for details).

2.1.4 Solution of Navier-Stokes equations

We employ the fractional step method developed by Ge and Sotiropoulos [7], which is briefly described herein. The momentum equations (2.2) are discretized in space and time with a second-order central differencing scheme for the pressure gradient and viscous and SGS stresses, second-order central differencing or a third-order weighted essentially non-oscillatory (WENO) scheme [134] for the advective terms, and the second-order Crank-Nicholson method for time advancement as follows:

$$\frac{1}{J}\frac{\mathbf{U}^* - \mathbf{U}^\mathbf{n}}{\triangle t} = \mathbf{P}(p^n, \phi^n) + \frac{1}{2}(\mathbf{F}(\mathbf{U}^*, \mathbf{u}^*, \phi^{n+1}) + (\mathbf{F}(\mathbf{U}^\mathbf{n}, \mathbf{u}^\mathbf{n}, \phi^n)), \tag{2.19}$$

where n indicates the previous time step, $\triangle t$ the time step size, $\mathbf{F}$ is the discrete right hand side of Eq.(2.2) excluding the pressure term, and $\mathbf{P}$ the discrete pressure gradient term. The continuity condition is discretized with three-point central differencing and is enforced in the second stage of the fractional step method with the following pressure Poisson equation, which is discretized with a second-order central differencing scheme:

$$-J\frac{\partial}{\partial \xi^i}\left(\frac{1}{\rho(\phi)}\frac{\xi_l^i}{J}\frac{\partial}{\partial \xi^j}\left(\frac{\xi_l^j \Pi}{J}\right)\right) = \frac{1}{\triangle t} J \frac{\partial U^{j,*}}{\partial \xi^j}, \tag{2.20}$$

where Π denotes the pressure correction. The pressure and the velocity fields resulting from the first step of the method, can then be updated as follows:

$$p^{n+1} = p^n + \Pi, \tag{2.21}$$

$$U^{i,n+1} = U^{i,*} - J\Delta t \frac{1}{\rho(\phi)}\frac{\xi_l^i}{J}\frac{\partial}{\partial \xi^j}\left(\frac{\xi_l^j \Pi}{J}\right), \tag{2.22}$$

The momentum equations are solved using an efficient matrix-free Newton-Krylov solver, and the Poisson equation with a generalized minimal residual (GMRES) method preconditioned with multi-grid. For details about the two solvers and their discrete implementation in the hybrid staggered/non-staggered formulation the reader is referred to [122, 96].

2.1.5 Solution of level set equations

As detailed in [96] the level set equation is discretized with a third-order WENO scheme [134] in space, and an explicit second-order Runge-Kutta (RK2) scheme in time.

$$\left.\frac{\partial \phi}{\partial \xi^1}\right|_{i,j,k} = \phi_{i+1/2,j,k} - \phi_{i-1/2,j,k} \tag{2.23}$$

where the value of $\phi_{i\pm1/2,j,k}$ at the cell faces is computed with the WENO scheme. For example, $\phi_{i+1/2,j,k}$ reads as follow

$$\phi_{i+1/2,j,k} = \frac{\beta_1}{\beta_1+\beta_2}\left(\frac{\phi_C}{2}+\frac{\phi_R}{2}\right) + \frac{\beta_2}{\beta_1+\beta_2}\left(-\frac{\phi_L}{2}+\frac{3\phi_C}{2}\right) \tag{2.24}$$

where

$$\beta_1 = \frac{2}{3}\frac{1}{\left((\phi_C-\phi_R)^2+\epsilon_0\right)^2} \tag{2.25}$$

$$\beta_2 = \frac{2}{3}\frac{1}{\left((\phi_L-\phi_C)^2+\epsilon_0\right)^2} \tag{2.26}$$

and

$$(\phi_L,\phi_C,\phi_R)_{i+1/2,j,k} = \begin{cases} (\phi_{i-1,j,k},\phi_{i,j,k},\phi_{i+1,j,k}) & U^1 > 0, \\ (\phi_{i+2,j,k},\phi_{i+1,j,k},\phi_{i,j,k}) & U^1 \leq 0, \end{cases} \tag{2.27}$$

The constant ϵ_0 was taken to be 10^{-6}, and the cell face values $\phi_{i-1/2,j\pm1/2,k\pm1/2}$ are computed in a similar manner.

For solving the mass conserving re-initialization equation (2.8), the invariant $|\nabla\phi|$

in generalized curvilinear coordinates reads as follow

$$|\nabla\phi| = \sqrt{\left(\xi_1^i \frac{\partial\phi}{\partial\xi^i}\right)^2 + \left(\xi_2^i \frac{\partial\phi}{\partial\xi^i}\right)^2 + \left(\xi_3^i \frac{\partial\phi}{\partial\xi^i}\right)^2} \tag{2.28}$$

where the spatial derivatives of the distance function are discretized with the second-order ENO scheme of [135] extended by [96] to generalized curvilinear grids,

$$\frac{\partial\phi}{\partial\xi^1} = \begin{cases} \frac{\partial\phi}{\partial\xi^1}^+ & \text{sgn}(\phi_0)(\phi_{i+1,j,k} - \phi_{i,j,k}) < 0 \quad \text{and} \\ & \text{sgn}(\phi_0)(\phi_{i,j,k} - \phi_{i-1,j,k}) < -\text{sgn}(\phi_0)(\phi_{i+1,j,k} - \phi_{i,j,k}), \\ \frac{\partial\phi}{\partial\xi^1}^- & \text{sgn}(\phi_0)(\phi_{i,j,k} - \phi_{i-1,j,k}) > 0 \quad \text{and} \\ & \text{sgn}(\phi_0)(\phi_{i+1,j,k} - \phi_{i,j,k}) > -\text{sgn}(\phi_0)(\phi_{i,j,k} - \phi_{i-1,j,k}), \\ \frac{1}{2}\left(\frac{\partial\phi}{\partial\xi^1}^+ + \frac{\partial\phi}{\partial\xi^1}^-\right) & \text{otherwise,} \end{cases} \tag{2.29}$$

where the function sgn denotes the standard sign function and $\frac{\partial\phi}{\partial\xi^1}^+$ and $\frac{\partial\phi}{\partial\xi^1}^-$ can be written as

$$\frac{\partial\phi}{\partial\xi^1}^+ = (\phi_{i+1,j,k} - \phi_{i,j,k}) - \frac{1}{2}\text{min}(\phi_{i+1,j,k} - 2\phi_{i,j,k} + \phi_{i-1,j,k}, \phi_{i+2,j,k} - 2\phi_{i+1,j,k} + \phi_{i,j,k}), \tag{2.30}$$

$$\frac{\partial\phi}{\partial\xi^1}^- = (\phi_{i,j,k} - \phi_{i-1,j,k}) + \frac{1}{2}\text{min}(\phi_{i+1,j,k} - 2\phi_{i,j,k} + \phi_{i-1,j,k}, \phi_{i,j,k} - 2\phi_{i-1,j,k} + \phi_{i-2,j,k}), \tag{2.31}$$

The reinitialization of the level set function is a key aspect of the method to ensure the conservation of mass, and consequently the overall accuracy of the method. As it was proven in [133] accuracy is maximized when the complete area covered by the thickness ϵ of the interface strip is fully re-distanced, and this is achieved when the total re-distancing time τ is equal to ϵ. The reinitialization time is computed in the following manner:

$$\tau = n_\tau \cdot \Delta\tau, \tag{2.32}$$

where $\Delta\tau$ is the time step size, and n_τ is the number of time steps. We have observed that when $\Delta\tau$ is large ($\Delta\tau > 0.1\Delta x$), numerical instabilities arise preventing the method from converging. This is particularly important for problems with complex free surface

flow phenomena and will be revisited again in the results section of this paper.

2.1.6 Solution of the equations of motion for rigid bodies

Computation of the body forces

The forces $F_{fluid,i}$ and moments $M_{fluid,i}$ that the fluid exerts to the rigid body and appear in the right hand side of the structural equations of motion (eqns. 2.12 and 2.13) are computed by integrating the pressure and the viscous stresses along the surface Γ of the body as follows:

$$F_{fluid,i} = \overbrace{\int_\Gamma -pn_i d\Gamma}^{F^i_{f,p}} + \overbrace{\int_\Gamma \tau_{ij} n_j d\Gamma}^{F^i_{f,s}} \tag{2.33}$$

$$M_{fluid,i} = \overbrace{\int_\Gamma -\varepsilon_{ijk} r_j p n_k d\Gamma}^{M^i_{f,p}} + \overbrace{\int_\Gamma \varepsilon_{ijk} r_j \tau_{kl} n_l d\Gamma}^{M^i_{f,s}} \tag{2.34}$$

where p denotes the pressure, τ the viscous stress, r the position vector, and n the normal vector. The subscripts p and s in the terms in right hand side of the above equations identify contributions from the pressure and shear forces or moments.

For IB methods in general the computation of the forces and moments in equations (2.33) and (2.34) cannot be directly performed. The mesh of the fluid domain does not conform with the structure, hence, the pressure and velocity gradients are not known on the surface of the body. Particularly, in the CURVIB method the nodes of the background curvilinear grid are classified in three categories depending on their position with respect to the immersed body: structural nodes, IB nodes, and fluid nodes (see figure 2.1). The structural nodes are located within the interior of the immersed body and are excluded from the computational domain. The rest of the nodes are either IB nodes, if they are located in the immediate vicinity of the body surface, or otherwise fluid nodes. Since in the IB nodes the velocity boundary condition is reconstructed, the only nodes for which the equations are solved and thus the pressure is available are the fluid nodes.

A technique that was used in [6] to remedy this situation is to employ an integration

surface Γ_1 (see figures 2.1 and 2.2) that encloses the body and is defined by the fluid nodes immediately adjacent to the IB nodes. At the nodes defining this surface both the pressure and velocity gradients can be calculated and thus the forces $F^i_{f,1}$ and moments $M^i_{f,1}$ can be obtained. This technique introduces an error inherent to immersed boundary methods, but, as shown in [88], the error approaches zero as the grid resolution increases.

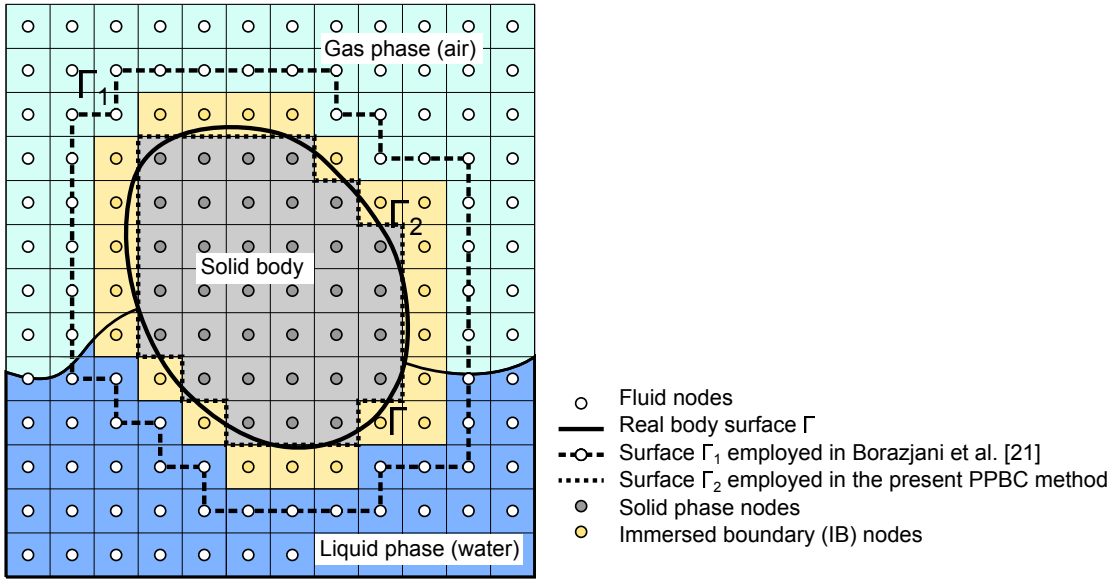


Figure 2.1: Schematic representation of the node classification in the CURVIB method. The surfaces Γ, Γ_1, and Γ_2 are the actual surface of the body and various approximate surfaces on which the pressure field can be integrated to calculate the pressure force acting on the body.

Let Γ be the actual surface of the body and Γ_1 the aforementioned approximate surface surrounding the solid body. Let m_{IB} be the mass of the fluid delimited by Γ and Γ_1. While in previous studies for single phase problems this mass was neglected, it can be important when the equations are formulated for two-phase flow and include a gravitational force. For instance, let us assume that we have a body submerged in stagnant water. From Archimedes's principle we can easily see that if we integrate the forces along a surface larger than the actual body surface, such as Γ_1, the resulting buoyant force will be higher than the real force felt by the body. Hence, if the method from [6] is applied directly for applications involving gravity, the vertical force will be

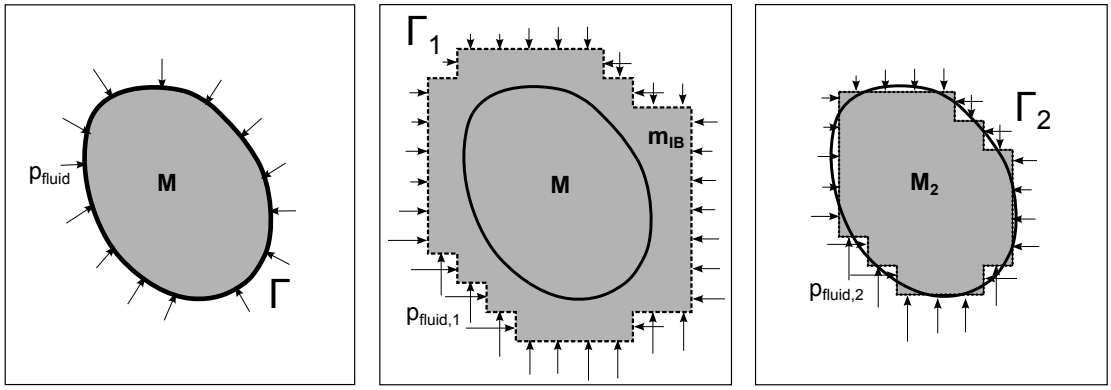


Figure 2.2: Schematic representation of the various approaches for calculating the pressure force by integrating the pressure field on: 1) the actual body surface Γ; 2) the surface Γ_1 outlining the volume defined by the IB cells, which was employed in Borazjani et al. [6]; and 3) the approximate surface Γ_2 employed in the proposed PPBC approach. M is the exact or approximate (M_2) mass of the body for each case.

over-predicted. To overcome this situation we propose an alternative approach referred to as the PPBC method in which the resultant force and moment due to pressure is computed by integrating the pressure distribution on a surface Γ_2. This new approach is presented in the following subsection.

The pressure projection boundary condition method

In the proposed PPBC approach, the part of the force F_{fluid} that is due to pressure is computed by directly integrating the pressure on the surface Γ_2 (see figures 2.1 and 2). To enable such integration, however, the pressure on Γ_2 has to be appropriately projected from the fluid nodes of the background grid where it is known. We propose a two-step approach for performing this pressure projection. First the pressure is projected to the center of the IB cells which are adjacent to the body. In a second step, the pressure at the center of a given IB cell is projected to its lateral faces belonging in Γ_2 as illustrated in the schematic shown in figure 2.3 and described in detail as follows.

To obtain the pressure at the IB nodes in the first step, the momentum equation (2.2) is projected along the direction of the wall normal as done in [79] and applied on the surface of the body. Neglecting viscous and subgrid-scale stresses, the normal

momentum equation applied on the body reads as follows:

$$-\frac{dp}{dn} = \rho\left(\phi\right) n^i \left(\frac{Du_i}{Dt} - \frac{\delta_{i3}}{Fr^2}\right) \quad \text{on } \Gamma \tag{2.35}$$

where n^i denotes the unit vector normal to the body surface and u_i the velocity components of the body computed with Eq. (2.14). With reference to figure 2.3, and since the value of the pressure p_c can be readily obtained by interpolating the pressure values between neighboring fluid cells [79], we can obtain the pressure at the IB node b as follows:

$$p_b = p_c - d_{cb}\rho_a n^i \left(\frac{u_i^n - u_i^{n-1}}{\Delta t} - \frac{\delta_{i3}}{Fr^2}\right) \tag{2.36}$$

where d_{cb} is the distance from points c to b, and the superscripts n and $n-1$ denote the current, and previous time steps. The density value ρ_a on Γ is unknown. However, it can be set to be equal to the density ρ_b as the Neumann boundary condition is applied for the distance function ϕ normal to the wall. The above equation (2.36) has been obtained by combining the following two expressions, which are approximations of equation (2.35) applied on the surface of the body:

$$-\left(\frac{p_a - p_c}{d_{ca}}\right) = \rho_a n_a^i \left(\frac{u_i^n - u_i^{n-1}}{\Delta t} - \frac{\delta_{i3}}{Fr^2}\right) \tag{2.37}$$

$$-\left(\frac{p_a - p_b}{d_{ba}}\right) = \rho_a n_a^i \left(\frac{u_i^n - u_i^{n-1}}{\Delta t} - \frac{\delta_{i3}}{Fr^2}\right) \tag{2.38}$$

where d_{ca} is the distance from points c to a and d_{ba} the distance from points b to a. In the second step, the so-computed pressure p_b at a cell center is projected to its cell faces a' and a'' on Γ_2 in the following manner:

$$-\left(\frac{p_{a'} - p_b}{d_{ba'}}\right) = \rho_b n_{a'}^i \left(\frac{u_i^n - u_i^{n-1}}{\Delta t} - \frac{\delta_{i3}}{Fr^2}\right) \tag{2.39}$$

where $n_{a'}$ is taken as the unit normal to the corresponding cell face. A similar expression is used to obtain the pressure at a''. Once the pressure has been computed at the center of all the faces forming Γ_2 the forces $F_{f,p,2}^i$ and moments $M_{f,p,2}^i$ can be computed as

follows:

$$F^i_{f,p,2} = \sum_{j=1}^{N_{faces}} p^j_{a'} S^j_{a'} n^j_{a',i} \tag{2.40}$$

$$M^i_{f,p,2} = \sum_{l=1}^{N_{faces}} \varepsilon_{ijk} r^l_j p^l_{a'} n^l_{a',k} S^l_{a'} \tag{2.41}$$

where N_{faces} is the total number of cell faces forming Γ_2, $S^j_{a'}$ is the area of the j-th cell face a'.

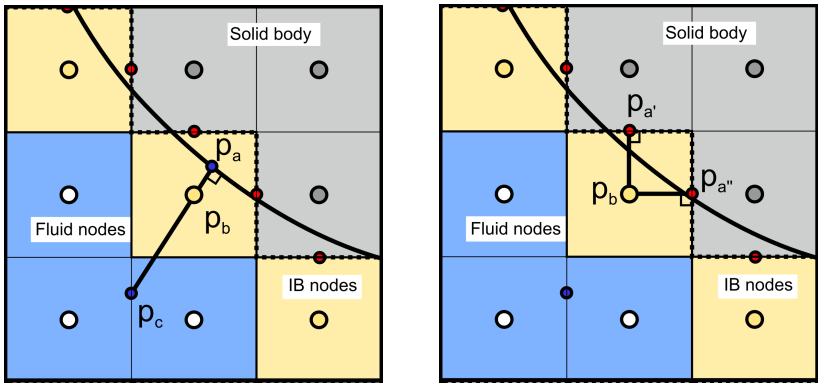


Figure 2.3: Schematic description of the successive pressure projections used to calculated the pressure on Γ_2 in the proposed PPBC method: step 1 (left) and step 2 (right).

The implementation of the PPBC approach is straightforward in methods such as the HCIB method or the CURVIB method in which a similar algorithm is used for reconstructing the velocity boundary condition at the IB nodes. Also, since the integration of the stresses is carried out in the background mesh, the parallel implementation of the algorithm is significantly simpler than in methods based on projecting the stresses to the unstructured Lagrangian mesh of the body. The simplicity and expedience of both algorithms, i.e. the PPBC and that from [6], when compared to methods based on projecting directly on the body surface, is in expense of an additional error caused by the approximation of the surface area. In the following we discuss the relative accuracy of the two methods.

Truncation error analysis

Whether we apply the method of [6] or the proposed PPBC method, the accuracy of the calculated forces and moments on the body depends on the accuracy of the pressure field and the accuracy of the approach used to approximate the surface of the body. As far as the pressure is concerned, both methods employ the same fractional-step approach to calculate the pressure field. It is well known that fractional step algorithms yield first-order accurate pressure field [136] and as such both the method of [6] and the PPBC employ similar accuracy pressure fields. We note, however, that in problems with floating or fully submerged bodies at equilibrium, when the force on the body is dominated by the hydrostatic pressure, a first order approximation of the pressure can accurately represent the linear variation of the pressure field. Therefore, for such problems the accuracy of the two approaches will be determined by the relative accuracy of the approach each method employs to approximate the surface of the body.

For the method of Borazjani et al. [6], it can be readily shown that the error in approximating the surface area of the body is exactly first order. This is due to the fact that this method approximates the volume of the body V_Ω by a larger volume V_{Ω_1}. Assuming that the layer of IB nodes surrounding the body has thickness of order Δh, an approximate expression for V_{Ω_1} can be written as follows:

$$V_{\Omega_1} = V_\Omega + S_\Gamma \Delta h + H.O.T, \tag{2.42}$$

where S_Γ is the surface area of the body and H.O.T indicates terms that are higher order in Δh. Since Δh is of order Δx, equation (2.42) suggests that the method of [6] for calculating the force is first order accurate in space.

Calculating the accuracy of the body volume approximation in the PPBC method turns out to be somewhat more involved than for the method of [6]. This is because the computation of the volume in the PPBC as the sum of quadrilateral cells that cover the interior of the body is equivalent to the well known problem of calculating the area within a closed convex curve C using lattice points [137, 138]. To demonstrate the underlying concepts and for the sake of simplicity we consider the 2D equivalent of the problem, namely the calculation of the area S of a closed curve as a summation of Cartesian grid cells forming a lattice. In such a case the surface S can be written as

follows:

$$S = N\Delta h^2 - E \tag{2.43}$$

where N is the number of lattice points located in the interior of the curve, Δh is the spacing of the lattice points (equal to the grid spacing), and E is the error of the approximation. Several authors have demonstrated the existence of bounds that limit the error E (see for example [137]). These bounds, which depend on the curvature of the boundary, can be expressed in the following form:

$$E \leq \frac{C}{\Delta h^{K-2}} \left[\log\left(\frac{1}{\Delta h}\right)\right]^M \tag{2.44}$$

The study of Van der Corput [139] demonstrated that for piece-wise curves of class C^2 (a curve is of class C^r when its radius of curvature is nonzero and $r-2$ times continuously differentiable with respect to the curve tangent direction) an upper limit of the error bound is (2.44) with values of K and M equals to 2/3 and 0, respectively. In such case the error in the computation of the surface is of order $O(\Delta h^{4/3})$. For curves of class C^3 Huxley [137] obtained $K = 131/208$ and $M = 131/8320$. The classical example of application of this type of lattice point methods is the so-called Gauss's circle problem, which is schematically represented in figure 2.4. The error bound of [137] for C^3 curves also applies for this problem. For an extensive review of lattice point methods applicable to other type of boundaries, such as boundaries with zero curvature points, or higher dimension boundaries, the reader is referred to the work of Ivic et al. [138].

Therefore, and based on the above analysis, we can estimate the order of accuracy of the volume calculation method used in the PPBC to be better than first order. Consequently we also anticipate that the PPBC will yield more accurate results in the calculation of the buoyancy force than the method of [6]. We will revisit this issue in section 4.2 where we will present a numerical accuracy test that confirms the theoretical arguments presented in this section.

Implementation for arbitrarily complex bodies

The PPBC method can, in principle, be applied to study one or more bodies of any general shape. However, in particular cases, such as in concave shaped bodies, or when

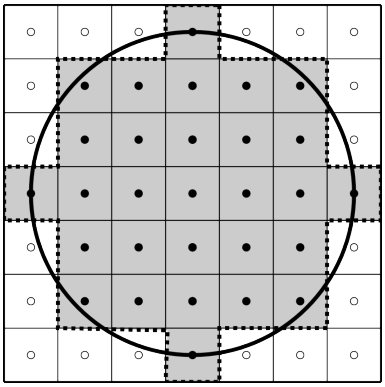
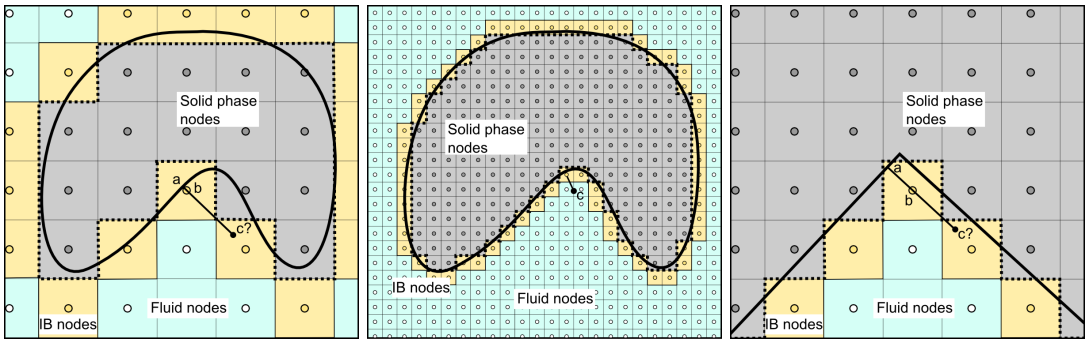


Figure 2.4: Schematic description of the Gauss's circle problem for the case of a unit radius circle and $\Delta h = 1/3$.

two or more surfaces are located in the proximity of each other, numerical difficulties may arise requiring special attention. In fact, the same difficulties are also encountered in the reconstruction step of the CURVIB method (see section 2.1.7 below for a brief description of the CURVIB method). As sketched in figure 2.5, this problem can be observed when the wall normal vector passing through the IB node b remains within the layer of IB nodes and does not extend into the area of fluid nodes from where flow variables can be extracted to reconstruct the pressure and/or velocity at the IB node b. Note that in both procedures, i.e. the first step of the proposed PPBC method and the velocity reconstruction in CURVIB, we employ the exact same wall normal vector as well as the same auxiliary point c. In such cases, neither the velocity nor the pressure can be obtained at point c, and thus, the method cannot proceed. Such difficulties, however, are encountered only when the background grid is too coarse to represent the shape of the body. As illustrated in figure 2.5b, with sufficient grid refinement this situation is alleviated. Therefore, preventing a situation such as that discussed above from occurring requires using a grid that is sufficiently fine to accurately represent regions of concave curvature of the body under consideration. Naturally, the coarsest grid spacing required to satisfy this constraint is a function of the specific body shape under consideration.

Another difficulty in implementing the PPBC (as well as in performing the velocity reconstruction for the CURVIB method) arises when two or more adjacent surfaces intersect to form sharp angles, as illustrated in figure 2.5c. In such a case, the surface



(a) Continuous concave surface, coarse grid

(b) Continuous concave surface, finer grid

(c) Discontinuous surface

Figure 2.5: Schematic description of problems arising in the implementation of the PPBC and CURVIB methods to handel concave surfaces. a, b) A continuous concave surface for which difficulties can be alleviated by grid refinement; c) A discontinuous surface for which a special treatment is required.

discontinuity cannot be eliminated even with grid refinements as there will always be at least one point (located near the intersection of the two surfaces) for which the velocity/pressure reconstruction will not be with the general approach. The only practical approach to handle such singular points is to implement a special treatment, based on interpolating the information from the closest fluid cells.

2.1.7 The FSI-CURVIB method

The method for tracking the motion of geometrically complex bodies is the sharp interface CURVIB method of Ge and Sotiropoulos [7], which has been thoroughly validated for simulating deformable bodies with large motions in various applications, including FSI problems [6, 140]. For the sake of completeness only a brief description of the method is presented herein. In the CURVIB approach, the body is represented by an unstructured triangular mesh which is embedded in the background curvilinear or Cartesian grid. An efficient searching algorithm is used to classify all nodes of the computational domain depending in their location with respect to the position of the body as already discussed in section 2.1.6. The linking between the background and structure grids is done through a sharp interface approach by reconstructing the boundary conditions for the velocity field and the distance function ϕ at the IB nodes (see Ge and Sotiropoulos

[7] and Kang and Sotiropoulos [96] for details). The velocity is reconstructed in the wall normal direction with either linear or quadratic interpolation in the case of low Reynolds number flows when the IB nodes are located in the viscous sub-layer, or using the wall models described by [141, 142, 143] in high Reynolds number flows when the grid resolution is not sufficient to accurately resolve the viscous sub-layer. The distance function ϕ is reconstructed by setting its gradient to be zero at the cell faces that are located between the fluid and IB nodes. This is equivalent to applying a zero Neumann boundary condition along the grid line corresponding to the aforementioned cell faces. A further description of how to reconstruct the distance function in the IB nodes is given in [96].

After calculating the force acting on the body by the method presented above, the system of first order ordinary differential equations (2.15-2.18) are coupled with the fluid domain equations through a partitioned FSI approach. The time integration can be done explicitly with loose coupling LC-FSI, or implicitly with strong coupling SC-FSI. To illustrate the difference between the two algorithms, we formulate the system of equations (2.17-2.18) in semi-discrete form as follows:

$$\begin{aligned}\frac{\partial Y^i}{\partial t} + \frac{b_{t,i}}{M} Y^i + \frac{k_{t,i}}{M} X^i = \quad & \tfrac{\alpha}{M} F_{fluid,i}(\mathbf{X}^{n+1}, \boldsymbol{\Theta}^{n+1}, \mathbf{U}^{n+1}, p^{n+1}) \\ & + \tfrac{(1-\alpha)}{M} F_{fluid_i}(\mathbf{X}^n, \boldsymbol{\Theta}^n, \mathbf{U}^n, p^n) + \tfrac{1}{M} F_{ext,i} \end{aligned} \tag{2.45}$$

$$\begin{aligned}\frac{\partial \Psi^i}{\partial t} + \frac{b_{r,i}}{I_i} \Psi^i + \frac{k_{r,i}}{I_i} \Theta^i = \quad & \tfrac{\alpha}{I_i} M_{fluid,i}(\mathbf{X}^{n+1}, \boldsymbol{\Theta}^{n+1}, \mathbf{U}^{n+1}, p^{n+1}) \\ & + \tfrac{(1-\alpha)}{I_i} F_{fluid,i}(\mathbf{X}^n, \boldsymbol{\Theta}^n, \mathbf{U}^n, p^n) + \tfrac{1}{I_i} M_{ext_i} \end{aligned} \tag{2.46}$$

where the parameter α can be adjusted in the range between 0 and 1 to determine the time accuracy of the algorithm. When α is 0 the above equation takes the form of the LC-FSI algorithm, for other values of α the equation is expressed in the form of the SC-FSI algorithm. The latter algorithm requires sub-iterations every time.

The Aitken acceleration technique of [144] is able to significantly reduce the number of sub-iterations when the SC-FSI algorithm is used by choosing the optimum value of α. A detailed description of both time-integration algorithms is given in [6].

2.1.8 The actuator line model

The method that we use for parameterizing a turbine rotor is the actuator line model of Yang et al. [145]. The basic idea behind the method is to subtract from the flow field an equivalent amount of kinetic energy to that from a turbine rotor without the need to resolve the flow around its actual geometry. This effect is implemented by introducing a sink term in the right hand side of the momentum equation acting on those grid nodes that are located at the vicinity of the turbine rotor.

In the present actuator line method, the location of the turbine rotor is tracked by discretizing each of the blades in a Lagrangian manner with straight lines composed of several elements aligned with the radial direction. In each of the elements, the lift (L) and drag (D) forces are computed using the following expressions:

$$L = \frac{1}{2}\rho C_L C V_{rel}^2, \tag{2.47}$$

$$D = \frac{1}{2}\rho C_D C V_{rel}^2, \tag{2.48}$$

where C_L, C_D are the lift and drag coefficients, respectively, taken from tabulated 2D airfoil profile data, C is the chord length, and V_{ref} is the incoming reference velocity computed as

$$V_{rel} = (u_z, u_\theta - \Omega r) \tag{2.49}$$

where u_z and $u_\theta - \Omega r$ are the components of the velocity in the axial and azimuthal directions, respectively, Ω the angular velocity of the rotor, and r the distance to the center of the rotor.

The reference velocity at the line elements ($u(X)$) can be calculated by using interpolation from the surrounding fluid nodes where the velocity is known. This is necessary as the nodes from the fluid mesh and line segments do not necessarily coincide. If we consider X to be the coordinates of the actuator line nodes and x the coordinates of the fluid mesh nodes, we can perform the interpolation using a discrete delta function in the following manner

$$u(X) = \sum_{N_D} u(x)\delta_h(x - X)V(x), \tag{2.50}$$

where δ_h is a 3D discrete delta function, $V(x)$ is the volume of the corresponding fluid cell, and N_D is the number of fluid cells involved in the interpolation.

The lift and drag forces, which have been computed at each of the line elements, can be transferred in a diffused manner into the flow domain using the following equation:

$$S^{AL}(x) = \sum_{N_L} F(X)\delta_h(x - X)A(x). \tag{2.51}$$

where N_L is the number of segments composing one of the actuator lines, $A(x)$ is the length of each segment, and $F(X)$ is the projection of L and D into the Cartesian coordinates.

2.1.9 Solution Procedure

The procedure for solving the overall problem using the SC-FSI approach is summarized as follows.

1. Starting with an initial condition or previous time step values for the fluid domain (ϕ^n,$\mathbf{U}^n$,p^n) and the structural domain ($\mathbf{X}^n$,$\mathbf{\Theta}^n$) variables, advance the solution as follows.

2. For $k = 1$, set $\widetilde{\mathbf{U}}^k = \mathbf{U}^n$, $\widetilde{p}^k = p^n$, $\widetilde{\mathbf{X}}^k = \mathbf{X}^n$, and $\widetilde{\mathbf{\Theta}}^k = \mathbf{\Theta}^n$ and start a sub iteration loop.

3. Calculate the forces and moments using Eqs. (2.33) and (2.34) $\widetilde{\mathbf{F}}^{k+1}_{Fluid} = \widetilde{\mathbf{F}}^{k+1}_{Fluid}(\phi^n, \widetilde{\mathbf{U}}^k, \widetilde{p}^k, \widetilde{\mathbf{X}}^k, \widetilde{\mathbf{\Theta}}^k)$ and $\widetilde{\mathbf{M}}^{k+1}_{Fluid} = \widetilde{\mathbf{M}}^{k+1}_{Fluid}(\phi^n, \widetilde{\mathbf{U}}^k, \widetilde{p}^k, \widetilde{\mathbf{X}}^k, \widetilde{\mathbf{\Theta}}^k)$.

4. Solve the equations of motion (2.12-2.13) to obtain the new position of the structure $\widetilde{\mathbf{X}}^{k+1} = \widetilde{\mathbf{X}}^{k+1}(\widetilde{\mathbf{X}}^k, \widetilde{\mathbf{\Theta}}^k, \widetilde{\mathbf{F}}^{k+1}_{Fluid})$ and $\widetilde{\mathbf{\Theta}}^{k+1} = \widetilde{\mathbf{\Theta}}^{k+1}(\widetilde{\mathbf{X}}^k, \widetilde{\mathbf{\Theta}}^k, \widetilde{\mathbf{M}}^{k+1}_{Fluid})$.

5. If $k = 1$, solve the Level set and reinitialization equations (2.7) and (2.8) to obtain $\phi^{n+1} = \phi^{n+1}(\phi^n, \mathbf{U}^n, \mathbf{X}^{k+1}, \mathbf{\Theta}^{k+1})$.

6. Solve the momentum and Poisson equations (2.2) and (2.20) to obtain new values for the flow variables $\widetilde{\mathbf{U}}^{k+1} = \widetilde{\mathbf{U}}^{k+1}(\phi^{n+1}, \widetilde{\mathbf{U}}^k, \widetilde{\mathbf{X}}^{k+1}, \widetilde{\mathbf{\Theta}}^{k+1})$ and $\widetilde{p}^{k+1} = \widetilde{p}^{k+1}(\phi^{n+1}, \widetilde{\mathbf{U}}^k, \widetilde{\mathbf{X}}^{k+1}, \widetilde{\mathbf{\Theta}}^{k+1})$.

7. Check the convergence of the structural position: $\left\|\widetilde{\mathbf{X}}^{k+1} - \widetilde{\mathbf{X}}^{k}\right\| < \epsilon_0$ and $\left\|\widetilde{\mathbf{\Theta}}^{k+1} - \widetilde{\mathbf{\Theta}}^{k}\right\| < \epsilon_0$.

8. If convergence has been achieved, advance to step 1, otherwise increment k by one and return to step 3.

The LC-FSI algorithm is derived by the above algorithm by setting k=n and performing only one FSI iteration per time stepping, essentially stopping the algorithm at step 6 above and advancing to the next time step. Note that in the above SC-FSI algorithm the level set equation is solved only once per time step. Numerical tests showed that incorporating this equation in the SC sub-iteration loop increases the computational time without any quantifiable benefit in the accuracy of the computed solution.

2.2 The far-field flow solver

The large-scale wave-wind model is based on the two-fluid coupled approach of Yang and Shen [1, 2], which employs a potential based wave solver with a HOS method for the water motion and a viscous solver with undulatory boundaries for the air motion. A brief description of the governing equations and numerical methods as well as the coupling algorithm is provided in this section.

2.2.1 The high order spectral method for simulating water waves

To model the far-field free surface elevation of the water field in a non-linear manner, we solve the potential flow wave problem formulated in the form of Zakharov [146] by applying the HOS method of Dommermuth and Yue [147]. The kinematic and dynamic boundary conditions (BCs) can be written as functions of the free surface elevation η and the velocity potential Φ as follows

$$\nabla^2\Phi = \frac{\partial^2\Phi}{\partial x_i \partial x_i} = 0, \tag{2.52}$$

$$\frac{\partial\eta}{\partial t} + \frac{\partial\eta}{\partial x_\alpha}\frac{\partial\Phi^s}{\partial x_\alpha} - \left(1 + \frac{\partial\eta}{\partial x_\alpha}\frac{\partial\eta}{\partial x_\alpha}\right)\frac{\partial\Phi}{\partial x_3}\bigg|_{x_3=\eta} = 0, \tag{2.53}$$

$$\frac{\partial \Phi^s}{\partial t} + \frac{\eta}{Fr^2} + \frac{1}{2}\frac{\partial \Phi^s}{\partial x_\alpha}\frac{\partial \Phi^s}{\partial x_\alpha} - \frac{1}{2}\left(1 + \frac{\partial \eta}{\partial x_\alpha}\frac{\partial \eta}{\partial x_\alpha}\right)\left(\frac{\partial \Phi}{\partial x_3}\right)^2\Bigg|_{x_3=\eta} = -P_a, \tag{2.54}$$

where $i = 1, 2, 3$, $\alpha = 1, 2$, and $\Phi^s = \Phi|_{x_3=\eta}$ and P_a are the velocity potential and air pressure at the water surface, respectively. Note that x_1 and x_2 correspond to the coordinate components along the horizontal directions and x_3 along the vertical direction.

The following perturbations series of Φ which are expressed with respect to the wave steepness to order M and the Taylor series expansion to the same order about the mean water level $x_3 = 0$ are used,

$$\Phi\left(x_1, x_2, x_3, t\right) = \sum_{m=1}^{M} \Phi^{(m)}(x_1, x_2, x_3), \tag{2.55}$$

$$\Phi^s\left(x_1, x_2, t\right) = \sum_{m=1}^{M}\sum_{l=0}^{M-m} \frac{\eta^l}{l!}\frac{\partial^l \Phi^{(m)}}{\partial x_3^l}\Bigg|_{x_3^l=0} \tag{2.56}$$

and each $\Phi^{(m)}$ can be express with N modes using an eigenfunction expansion,

$$\Phi^{(m)}\left(x_1, x_2, x_3, t\right) = \sum_{n=1}^{N} \Phi_n^{(m)}(t)\Psi_n(x_1, x_2, x_3),\ z_3 \leq 0, \tag{2.57}$$

For deep water waves Ψ_n can be taken as

$$\Psi_n(x_1, x_2) = \exp\left(\sqrt{k_{1n}^2 + k_{2n}^2}\cdot x_3 + \imath(k_{1n}\cdot x_1 + k_{2n}\cdot x_2)\right), \tag{2.58}$$

where $\imath = \sqrt{-1}$, and k_{1n} and k_{2n} are the components of the wavenumber vector k in the x_1 and x_2 directions, respectively. Later in this work, we use k_x to refer to k_1 and k_y to refer to k_2.

Periodic BCs are imposed on the horizontal directions, which allows the HOS method to use an efficient spacial discretization scheme based on a pseudo-spectral method. The equations (2.53) and (2.54) are advanced in time with the fourth-order Runge-Kutta (RK4) scheme. An extensive description of this approach with validations and

applications was presented in [147] and in chapter 15 of [148].

2.2.2 The LES method for the air field

To simulate the air flow over water waves in the far-field model, we employ the method of Yang, Meneveau, and Shen [149] which is an extension to LES of the initial DNS approach of Yang and Shen [1]. We solve the following filtered incompressible Navier-Stokes equations governing the flow of a single fluid ($i, j = 1, 2, 3$)

$$\frac{\partial \widetilde{u}_i}{\partial t} + \frac{\partial \left(\widetilde{u}_i \widetilde{u}_j\right)}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \widetilde{P}}{\partial x_i} - \frac{\partial \tau_{ij}}{\partial x_j}, \tag{2.59}$$

$$\frac{\partial \widetilde{u}_i}{\partial x_i} = 0. \tag{2.60}$$

where $\widetilde{u}_i$ are the filtered velocity components, $\widetilde{P}$ is the filtered dynamic pressure, and τ_{ij} is the SGS stress tensor modeled by the scale-dependent Lagrangian dynamic model [150]. Note that the viscous term in equation (2.59) has been neglected due to the consideration of high Reynolds number and the negligible effect of viscosity at the resolved scale. A wall model is used to account for the viscous effects at the bottom free surface boundary. A boundary fitted grid is employed and adapts to the motion of the free surface which is seen by the air domain as a undulatory boundary (see [1] for details of the coordinate transformation mapping). The geometry and the velocity of the bottom boundary are prescribed from the HOS simulation. The air flow can be driven either by applying a constant pressure gradient in the stream-wise direction or by applying a shear stress at the top boundary and periodic BCs are considered in the horizontal directions.

For the spacial discretization in the horizontal directions, a Fourier-series-based pseudo-spectral method is used taking advantage of the fact that the BCs are periodic. In the vertical direction, a second-order finite difference scheme is used. A semi-implicit fractional-step method is applied to advance the governing equations (2.59) and (2.60).

2.2.3 The LES-HOS coupling algorithm

The LES and HOS models are dynamically coupled by following an iterative procedure. First the HOS method is used to advance the wave to the next time step $(n+1)$ under the forcing of air pressure P_a^n on the wave surface, i.e., the free surface elevation η^{n+1} and surface velocity u_s^{n+1} which can then be imposed as Dirichlet BC at the bottom boundary of the LES model. The air flow can then be advanced to time step $(n+1)$ by solving the described LES model, and a new value of surface pressure P_a^{n+1} is computed to continue the simulation.

2.3 Far-field/Near-field coupling

We propose two different far-field/near-field coupling approaches to take full advantage of the numerical expedience of the far-field method for large-scale wind-wave simulations and the advanced capabilities of the near-field solver for simulating wind-wave-body interactions.

In approach 1, the two solvers are loosely coupled in time. The wind flow from the far-field can be incorporated directly into the near-field solver by prescribing at each time step the instantaneous air velocity at the inlet boundary. The process for incorporating the wave field involves the following two steps: (1) extract the energy and phases of surface waves from the far-field model by performing a Fourier analysis, and (2) incorporate the resulting far-field waves into the near-field domain by applying the surface forcing method of Guo and Shen [39] which has been appropriately adapted to the level set method. This approach is aimed to simulate a single or multiple floating structures.

In approach 2, the far-field simulation at a particular instant in time for which the flow is fully developed is used as initial condition for the near-field method, i.e., the far-field simulation is restarted using the FSI-level set method with the addition of the floating structures. The wind field is directly prescribed in the whole air domain of the FSI-level set method using interpolation and the wave field is initialized with the pressure forcing method of Guo and Shen [39]. In this approach periodic boundary conditions are used in the near-field domain. This approach is suited for simulating infinite arrays of floating structures.

The major difference in approach 2 compared to approach 1 is the way that the pressure forcing method is applied both in time and in space. While in approach 1 the distributed pressure is continuously applied in time during all the simulation, in approach 2 the forcing method is only applied at very initial time. The area of application of the pressure, which is referred to as the source region also differs from the two approaches. While in approach 1 the source region corresponds to a rectangular band centered on the origin as illustrated in figures 2.6 and 2.7, in approach 2 it spans the whole free surface.

The forcing method of [39] relies on the linearized Cauchy-Poisson problem which allows to relate a given free surface pressure field to its leading free surface elevation response (details of the Cauchy-Poisson problem can be found in [148]). For simplicity but without lost of generality we write the pressure-elevation relationship in 2D form which writes as follows

$$\eta(x,t) = -\frac{1}{2\pi}\int_{-\infty}^{\infty} dk_x \exp[ik_x x]\int_0^t P_a(k_x,\tau)Fr^2\omega\sin[\omega(t-\tau)]d\tau, \tag{2.61}$$

where ω is the wave angular frequency, and P_a is the free surface pressure field.

2.4 Approach 1: Inlet-outlet FSI simulation

A schematic description of the overall procedure in approach 1 is depicted in figures 2.6 and 2.7. After the wind and wave fields from the far-field simulation are fully developed, the far-field solution can be fed to the near-field domain as described in this section.

2.4.1 2D wave generation

Using equation (2.61) one can obtain a progressive monochromatic wave of amplitude A and frequency ω adopting the following form

$$\eta(x,t) = A\cos(k_x x - \omega t), \tag{2.62}$$

and surface velocity

$$\eta_t(x,t) = A\omega\sin(k_x x - \omega t), \tag{2.63}$$

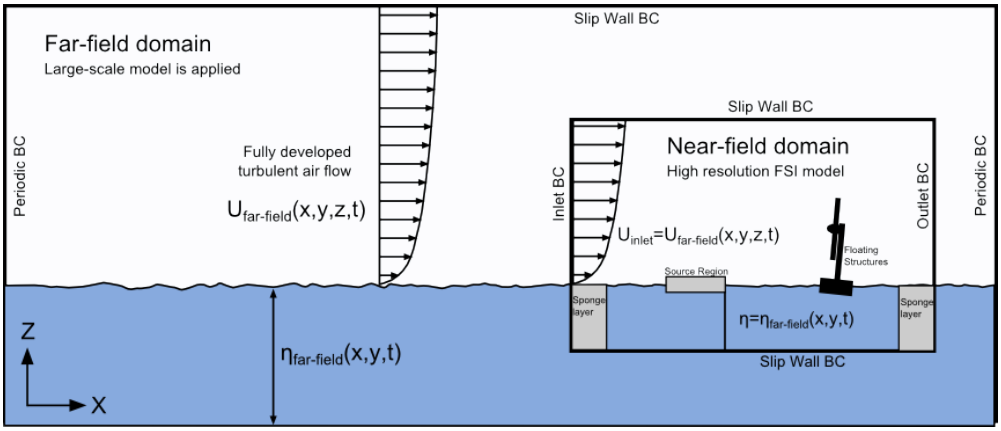


Figure 2.6: Schematic description of the near-field/far-field coupling approach 1. The inlet velocity at the near-field domain is prescribed from the far-field velocity field and the wave field is imposed through applying the pressure forcing method at the source region. Wave-wind-body interactions can be studied by placing the body between the source region and the outlet wall sponge layer.

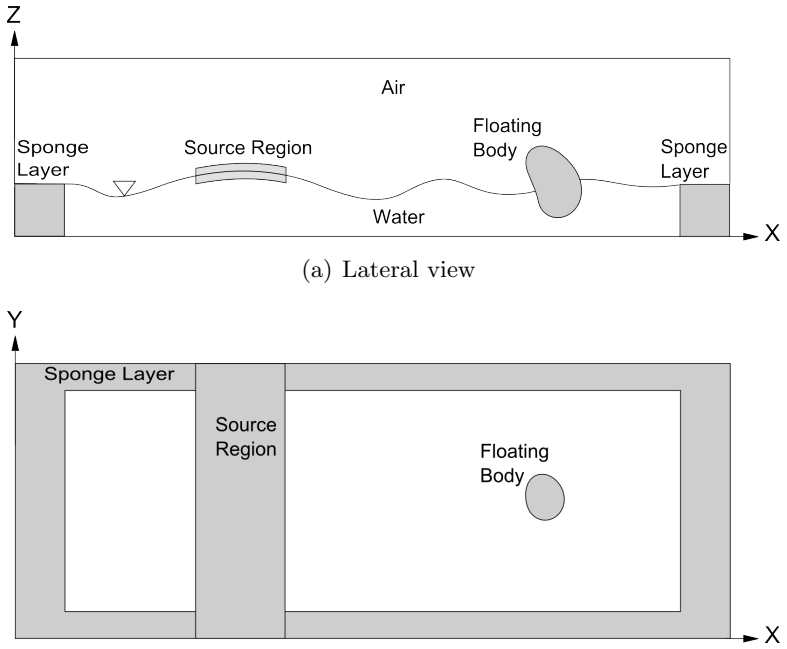
by applying the following free surface force corresponding to a oscillatory nodal force applied at $X = 0$

$$P_a(t) = P_0 \sin(\omega t), \tag{2.64}$$

where P_0, described below, is a coefficient that depends on the wave and fluid characteristics.

The application of a nodal force $P_a(t)$ generates progressive waves which propagate symmetrically with respect to the point of application $X = 0$. The numerical implementation of the nodal force on the free surface is not straightforward. It is important to consider that the interface is moving in time and the point of application of the force does not necessarily coincide with a grid coordinate point of the fixed Eulerian fluid mesh. Also, in the context of the present level set free surface tracking method the interface is diffused which adds additional numerical difficulties when adding the nodal force. For these reasons, we propose to diffuse the nodal force both, a distance ϵ_x along the x direction and a distance ϵ_ϕ along the free surface normal direction as follows

$$Ps_a(x, \phi, t) = P_0 \delta(x, \epsilon_x) \delta(\phi, \epsilon_\phi) sin(\omega t), \tag{2.65}$$



(a) Lateral view

(b) Top view

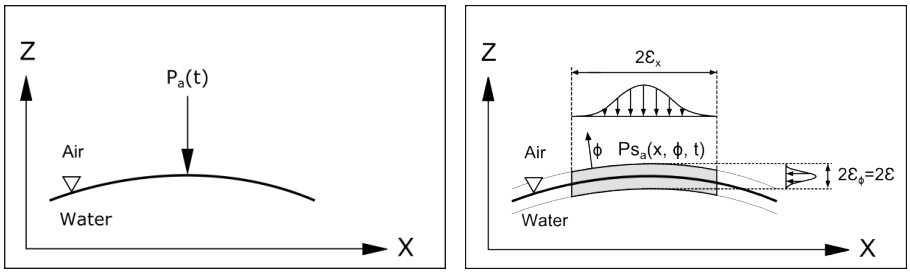
Figure 2.7: Schematic description of the near-field domain when using the far-field/near-field coupling approach 1. The source region for wave generation, which is applied on the free surface along a span-wise rectangular band centered on $X = 0$, generates 3D directional waves propagating symmetrically with respected to $X = 0$. When the waves reach the side walls the wave energy is dissipated using the sponge layer method to prevent wave reflections. The floating structure subject to wave interactions is located between the source region and the outlet sponge layer.

where δ is a distribution function defined as

$$\delta(\alpha,\beta) = \begin{cases} \frac{1}{2\beta}\left[1+\cos\left(\frac{\pi\alpha}{\beta}\right)\right] & \text{if } -\beta < t < \beta \\ 0 & \text{otherwise.} \end{cases} \tag{2.66}$$

We choose the value of ϵ_ϕ to be equal to the smoothing thickness of the free surface interface ϵ. The smoothing distance ϵ_x can adopt any value comprised in the interval $(0, L/2)$ being L the wavelength, although we observed better accuracy using $\epsilon_x \approx L/2$.

In order to determine the value of P_0 in equation (2.64), we use the wave energy flux concept. The energy flux Q_{Ps_a} induced by the distributed pressure Ps_a is set to be equal to the theoretical wave energy flux Q_{theory}. The theoretical wave energy flux



(a) Nodal force (b) Distributed pressure

Figure 2.8: Schematic description of the force applied on the free surface for wave generation when using the far-field/near-field coupling approach 1. The nodal force shown in (a) is implemented in the code in the distributed manner shown in (b).

is well known from linear wave theory and can be written as follows

$$Q_{theory} = Ec_g, \tag{2.67}$$

were E is the mean wave energy density defined as $E = \frac{1}{2}\rho g A^2$ and c_g is the wave group velocity which using the dispersion relation can be written as $c_g = \frac{1}{2}\sqrt{\frac{g}{k}}$.

The energy flux induced by the smoothed pressure force is computed as follows

$$Q_{Ps_a} = \frac{1}{T}\int_0^T \int_{-\epsilon_x}^{\epsilon_x} \int_{-\epsilon_\phi}^{\epsilon_\phi} Ps_a(x,\phi,t)\rho(\phi)\eta_t(x,t)d\phi dx dt. \tag{2.68}$$

Substituting (2.63) and (2.65) into (2.68) and considering the density smoothing across the interface as $\rho(\phi) = \frac{\rho_a+\rho_w}{2} + \frac{\rho_a-\rho_w}{2}\sin(\frac{\pi\phi}{2\epsilon_{phi}})$, the wave energy density induced by the force becomes

$$Q_{Ps_a} = -\frac{P_0 A\omega}{4\epsilon_x}\frac{\pi^2}{k(\pi^2-\epsilon_x^2k^2)}sin(k\epsilon_x)\frac{\rho_a+\rho_w}{2}, \tag{2.69}$$

and using the theoretical energy flux in equation (2.67),

$$P_0 = A\frac{g^2}{\omega^2}\frac{2\rho_w}{\rho_a+\rho_w}\frac{\epsilon_x}{f(\epsilon_x,k_x)}, \tag{2.70}$$

where $f(\epsilon_x, k_x)$ is

$$f(\epsilon_x, k_x) = \frac{\pi^2}{k_x\left(\pi^2 - \epsilon_x^2 k_x^2\right)} \sin(k_x \epsilon_x). \tag{2.71}$$

The distributed force in the source region is introduced in the code in form of a source term in the filtered momentum equations as follows

$$S_i^w(x,t) = n_i(\phi) P_0 \delta(x, \epsilon_x) \delta(\phi, \epsilon_\phi) sin(\omega t), \tag{2.72}$$

where n_i denotes the normal direction of the free surface. The source region always follow the vertical motion of the free surface adapting to its topological shape.

Typical values adopted in this work for ϵ_x are in the order of half wavelength, and for ϵ_ϕ between 3 and 6 grid sizes. By applying superposition principles, the above method can be applied to generate complex wave fields with multiple wave frequencies as it is demonstrated in the result section.

2.4.2 3D directional wave generation

The wave generation method can be extended to the generation of directional waves defined by the following free surface elevation

$$\eta(x,y,t) = A\cos(k_x x + k_y y - \omega t + \theta), \tag{2.73}$$

where k_x and k_y are the components of the wavenumber vector and θ is the wave phase. In this case the so constructed forcing term reads as follow

$$S_i^w(x,y,t) = n_i(\phi) P_0 \delta(x, \epsilon_x) \delta(\phi, \epsilon_\phi) sin(\omega t - k_y y - \theta), \tag{2.74}$$

where P_0 is a coefficient that depends on the wave and fluid characteristics,

$$P_0 = A\frac{g^2}{\omega^2} \frac{\epsilon}{f(\epsilon_x, k_x)} \frac{2\rho_w}{\rho_a + \rho_w} \frac{k_x}{\left(k_x^2 + k_y^2\right)^{1/2}}, \tag{2.75}$$

δ is the distribution function as defined in (2.66), and $f(\epsilon_x, k_x)$ is given in (2.71).

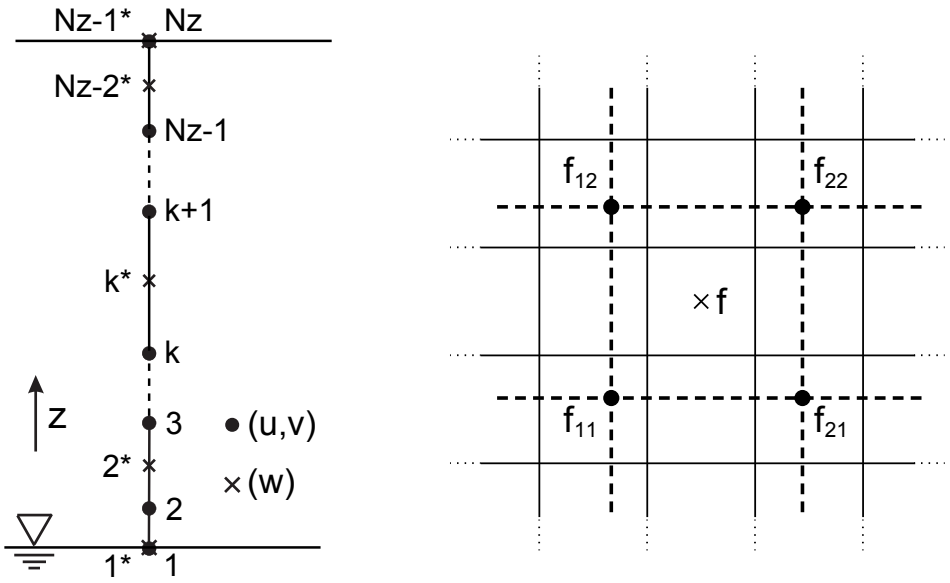
2.4.3 Sponge layer

The method used in this work for dissipating the energy of the waves near the boundaries of the computational domain is the sponge layer method expressed in the form proposed by [38] as follows

$$S_i^s(x,y,t) = -\left[\mu C_0 u_i + \rho C_1 u_i \left|u_i\right|\right] \frac{\exp\left[\left(\frac{x_s - x}{x_s}\right)^{n_s}\right] - 1}{\exp(1) - 1} \text{ for } (x_0 - x_s) \leq x \leq x_0, \quad (2.76)$$

where x_0 denotes the starting coordinate of the source region, x_s is the length of the source region, and C_0, C_1, and n_s are coefficients to be determined empirically.

2.4.4 Air flow coupling



(a) Far-field vertical grid storage

(b) Near-field interpolation

Figure 2.9: Schematic description of the far-field and near-field grid data storage. (a) Location of the points where the three velocity components are stored at the far-field mesh in the vertical direction. (b) Superposition of the far-field mesh, in dash line, over the near-field mesh, indicated in continuous line.

The air flow from the far-field solver is directly prescribed in the inlet plane of the

near-field solver at every time step. This process requires special considerations as the far-field method uses a boundary-fitted approach for which the mesh continuously deforms adapting to the free surface elevation. In contrast, the fluid mesh in the near-field method is purely Eulerian and its grid points are kept fixed during all the simulation. Since the grid nodes of the far-field source plane and the near-field inlet plane do not necessarily coincide, we perform a bi-linear interpolation.

The storage of the velocity components at the far-field domain is particular as it depends on the direction. While along the horizontal direction all three velocity components, u, v, and w are stored at the grid points, along the vertical direction, v and u are stored at the grid points and w at the grid midpoint as illustrated in figure 2.9a.

The far-field plane corresponding to the last cross section of the computational domain is exported to a data file which is later read by the near-field solver. Once the far-field velocity for a particular time step is available at the near-field solver, it is prescribed at the inlet plane of the near-field domain using bi-linear interpolation. The velocity components in the inlet-plane of the near-field domain are stored in Cartesian form at the cell centers. The bi-linear interpolation for obtaining the approximated velocity components f (see figure 2.9) is as follows:

$$\begin{aligned} f(x,y) \quad \approx \quad & \frac{1}{(x_2-x_1)(y_2-y_1)}\left[(x_2-x)(y_2-y)f_{11}+(x-x_1)(y_2-y)f_{21}+\right. \\ & \left.+(x_2-x)(y-y_1)f_{12}+(x-x_1)(y-y_1)f_{22}\right], \end{aligned} \tag{2.77}$$

where f_{ij} indicate the far-field velocity components.

2.5 Approach 2: Periodic FSI simulation

In this approach the wave generation method is a direct extension of the δ-Function method, described in [39], to the level set method. Equation (2.61) can be used to relate a surface pressure with the following free surface elevation, corresponding to a progressive wave, expressed as a sum of two standing waves,

$$\eta(x,t) = A\cos(kx-\omega t+\theta) = A\sin(\omega t-\theta)\cos(kx) + A\cos(\omega t-\theta)\sin(kx). \tag{2.78}$$

The above free surface elevation can be accomplished in a sharp free surface interface

method by applying an instantaneous pressure field on the complete undisturbed free surface. As proposed in [39], the pressure field is smeared over a period of time equal to 2Δ to facilitate its numerical implementation by using a smoothed delta function as follows

$$P_a(x,t) = -\frac{A}{Fr^2\omega}\delta(t,\Delta)\cos(kx) + \frac{A}{Fr^2\omega}\delta\left(t-\frac{\pi}{2\omega},\Delta\right)\sin(kx), \tag{2.79}$$

with the smoothed delta function given in equation (2.66). As indicated by [39], Δ adopts a value comprised between $4dt$ and 10% of the wave period.

To extend the pressure force to a smoothed interface approach, we can distribute $P_a(x,t)$ as follows:

$$P_\phi(x,t,\phi) = B\delta(\phi)P_a(x,t), \tag{2.80}$$

where B is a coefficient to ensure that the energy flux P_{P_ϕ} induced by $P_\phi(x,t,\phi)$ is equivalent to the energy flux P_{P_a} induced by $P_a(x,t)$. The energy fluxes are defined as follows

$$P_{P_a} = \frac{1}{2T\Delta}\int_0^L\int_{-\Delta}^{\Delta} P_a(x,t)\rho_w\eta_t(x,t)dtdx, \tag{2.81}$$

$$P_{P_\phi} = \frac{1}{2T\Delta}\int_{-\epsilon}^{\epsilon}\int_0^L\int_{-\Delta}^{\Delta} P_a(x,t)\rho(\phi)\eta_t(x,t)dtdxd\phi, \tag{2.82}$$

By setting (2.81) equals to (2.82), B becomes

$$B = \frac{2\rho_w}{\rho_w+\rho_a} \tag{2.83}$$

Chapter 3

Validation of the Near-Field solver for simulating floating structures

In this section, we apply the coupled CURVIB, level set, FSI algorithm to simulate a number of test cases of gradually increasing complexity. We seek to demonstrate the accuracy of the method by comparing its results with previous numerical studies and experimental measurements, probe via numerical experiments various aspects of the proposed computational algorithm, and demonstrate the ability of the method to simulate complex FSI problems involving geometrically complex structures interacting with the free surface.

3.1 Prescribed motion test case: Water entry/exit of a horizontal circular cylinder

The first test case was selected to validate the CURVIB level set method and involves a body moving across the air/water interface with prescribed motion. Therefore, the FSI algorithm is not employed in this test. We consider the two 2D cases of an infinitely long cylinder of radius R=1 moving with constant speed in an infinitely wide domain downward (upward) crossing the free surface from the air (water) phase. We shall refer

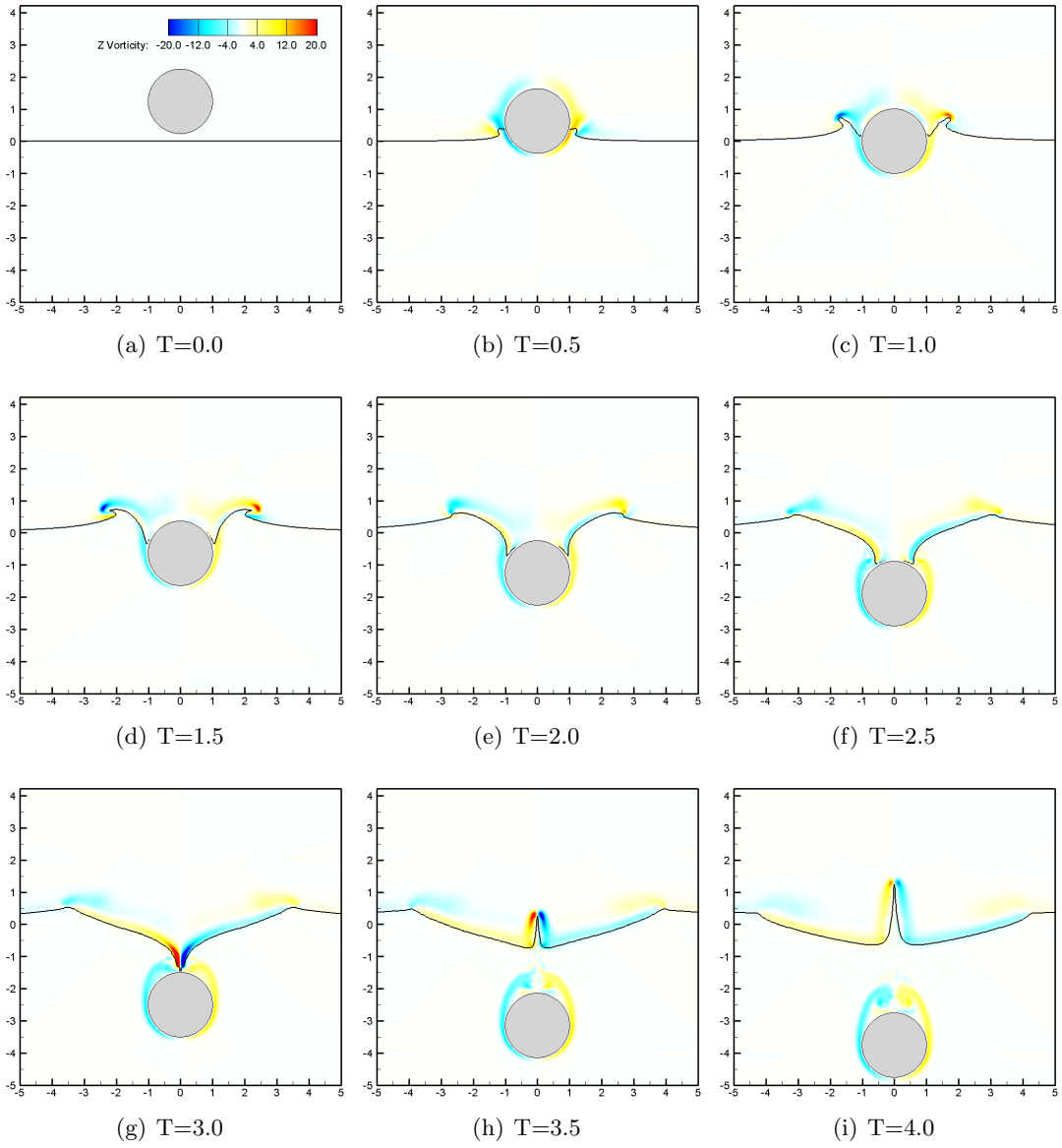
Table 3.1: Description of the four grids employed in the water entry/exit case of a horizontal circular cylinder.

Grid	Grid size (horizontal, vertical)	Near cylinder spacing
1	170×125	0.2R
2	220×160	0.1R
3	360×255	0.05R
4	640×450	0.0025R

to the first case as the entry (from air to the water) problem and to the second case as the exit (from the water to the air) problem. In the entry case, the configuration we study herein is identical to that simulated by [151, 10], where an identical cylinder, initially positioned above the free surface at a distance $h = 1.25$, moves downwards with constant velocity of $u = -1$. The water density is $\rho_{water} = 1$ and the dynamic viscosity $\mu_{water} = 1 \cdot 10^{-3}$ while the corresponding values for air are $\rho_{air} = 1 \cdot 10^{-3}$ and $\mu_{air} = 1.8 \cdot 10^{-5}$. The gravity is set to $g = -1$ and the time is normalized as $T = ut/h$.

The 2D computational domain is $40R \times 24R$ in the horizontal and vertical directions, respectively, as in [10]. Four non-uniform, successively finer meshes were employed to systematically investigate the grid sensitivity of the computed solutions. Each grid consists of an inner region, centered on the cylinder, within which the mesh is uniform and an outer region where the grid is gradually stretched. For all grids the inner region is the rectangular domain defined by [-5, 5] and [-4, 2.6] in the horizontal and vertical directions, respectively. Within this inner domain uniform grid spacing is employed along both directions, which is equal to $0.2R$ for grid 1, $0.1R$ for grid 2, $0.05R$ for grid 3, and $0.025R$ for grid 4. Outside of this inner domain the mesh is stretched gradually away from the cylinder using the hyperbolic stretching function with a stretching ratio kept below 1.05. The number of nodes for each mesh is: 170×125, 220×160, 360×255, and 640×450 for grids 1 to 4, respectively. A summary of the grids we employ is given in table 3.1.

The free surface and flow patterns calculated on grid 3 with a time step of 0.01 are presented at different instances in time in figure 3.1, which shows the calculated free surface position, cylinder position and induced vorticity field. It is evident from this figure that as the cylinder impinges onto the free surface two inclined jets of water form on each side of the cylinder where the free surface steepens and ultimately breaks (see



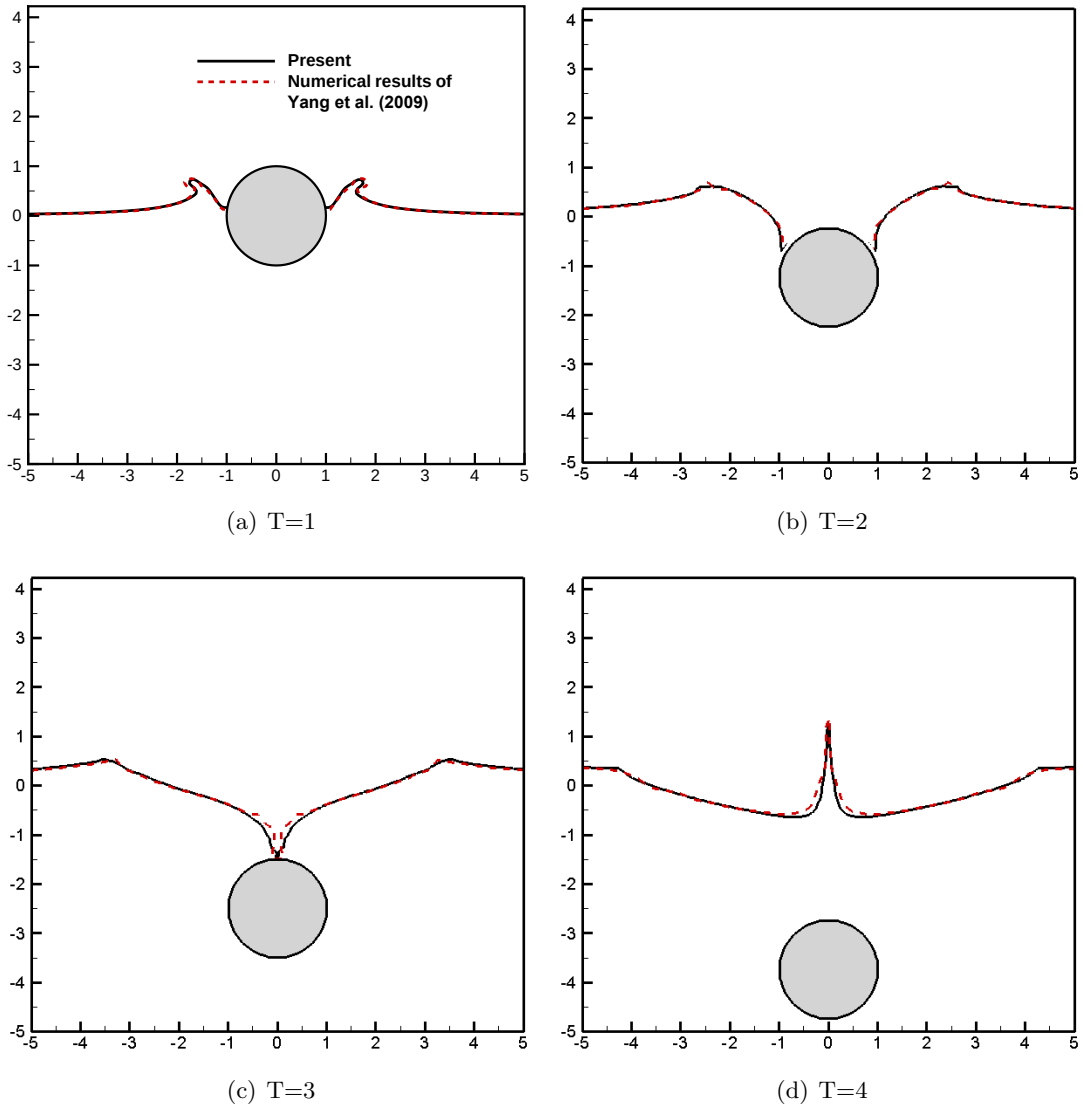
(a) T=0.0 (b) T=0.5 (c) T=1.0

(d) T=1.5 (e) T=2.0 (f) T=2.5

(g) T=3.0 (h) T=3.5 (i) T=4.0

Figure 3.1: Water entry of a horizontal circular cylinder moving with prescribed velocity. Simulated free surface position at different times along with vorticity contours. The results have been obtained on grid 3 with a time step of 0.01.

figure 3.1d). As the cylinder continues its motion through the surface it creates a system of waves that propagate outward away from the body. Ultimately the cylinder enters fully the water phase and its downward motion is seen to create a steep, vertical cusp



(a) T=1 (b) T=2

(c) T=3 (d) T=4

Figure 3.2: Water entry of a horizontal circular cylinder moving with prescribed velocity. Free surface position at different non-dimensional times T calculated by the present method and the method of Yang et al. [10]. The results have been obtained on grid 3 with a time step of 0.01.

in the free surface marking the trailing jet of the water in the cylinder wake (see figures 3.1h and i). At all instances in time vorticity is generated in the vicinity of the cylinder but also across the air-water interface in regions where curvature develops in the free

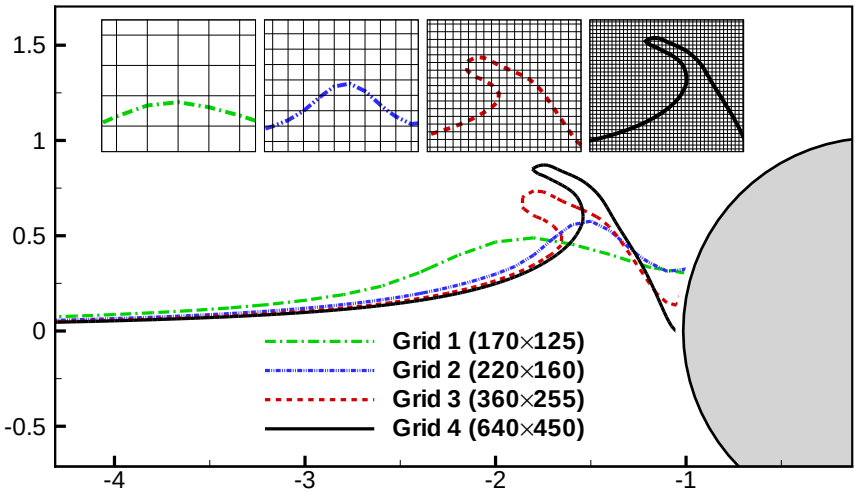
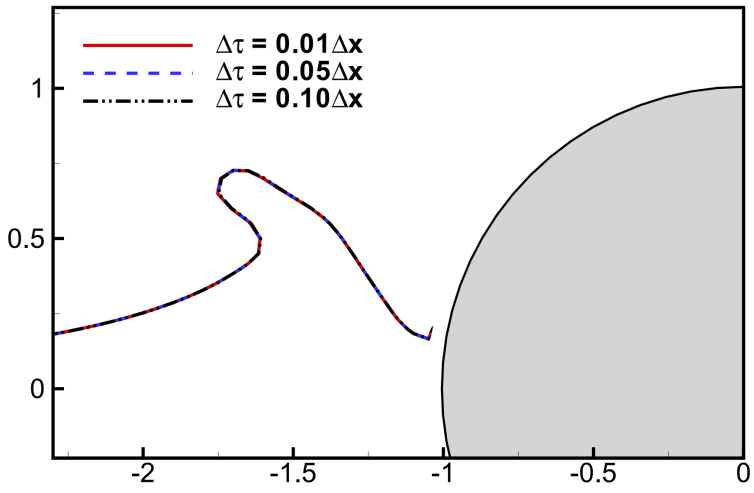


Figure 3.3: Water entry of a horizontal circular cylinder moving with prescribed velocity. Influence of the grid refinement on the free surface position at time $T = 1.0$ computed with a time step of 0.005.

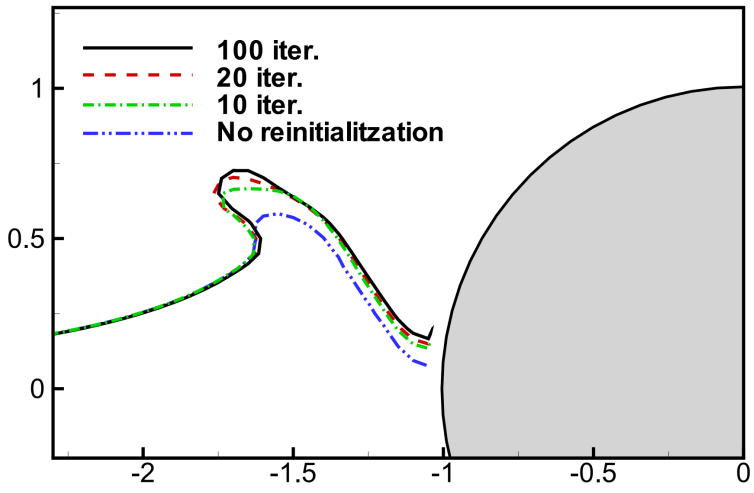
surface.

The free surface patterns captured by our simulation on grid 3 are compared with those reported by [10] who employed a grid of similar resolution (figure 3.2). More specifically [10] employed a non-uniform two-region grid of size $300x240$ with uniform inner rectangular domain centered on the cylinder with the same grid spacing of 0.05R. It is evident from this figure that our results are in good agreement with those obtained by [10]. However, minor discrepancies are observed at times $T = 3$ and $T = 4$ which can be explained by the fact that [10] employed a sharp interface level set method and a contact angle boundary condition while the present method uses a diffused interface method and no contact angle boundary condition. Treating the air/water interface with a sharp method results in overall higher accuracy as the actual pressure jump condition is taken directly in consideration. On the other hand, diffused methods are in general simpler to implement and the introduction of a diffusion thickness allows the method to prevent the formation of disturbances which may originate with the presence of an abrupt jump as discussed in [89].

To demonstrate the sensitivity of the computed results to grid refinement we show



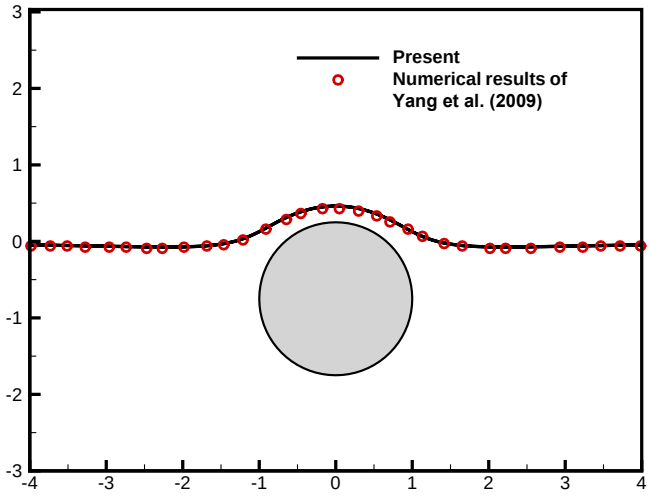
(a)



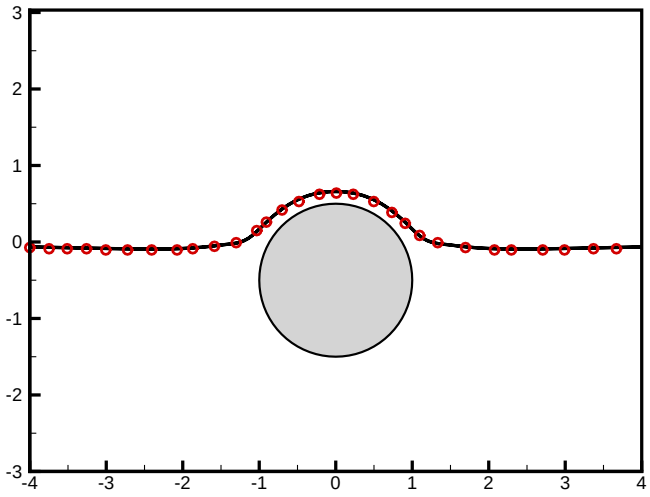
(b)

Figure 3.4: Water entry of a horizontal circular cylinder moving with prescribed velocity. Influence of the reinitialization time step $\Delta\tau$ (a) and the pseudo-time interval τ (b) on the accuracy of the calculated free surface position at time $T = 1.0$ computed on grid 3. Figure (a) shows the free surface when the full interface thickness is reinitialitzed using different time step sizes $\Delta\tau$. (b) shows different levels of reinitialization for a constant step size of $\Delta\tau = 0.01\Delta x$. For both cases the interface thickness is $\epsilon = \Delta x$, therefore when $\Delta\tau = 0.01\Delta x$ the interface is fully reinitialized with 100 iterations.

in figure 3.3 the predicted free surface shape on all four grids at an instant in time when the water jet off the side of the cylinder steepens just before the wave breaks. The same



(a) T=0.4



(b) T=0.6

Figure 3.5: Water exit of the horizontal circular cylinder moving with prescribed velocity. Free surface position at different non-dimensional times T calculated by the present method and the method of Yang et al. [10]. The results have been obtained on grid 3 with a time step of 0.005.

time step size of 0.005 and an interface thickness ϵ of 0.04 was employed for all grids. A number of important conclusions are obtained from this figure. First, it is evident that only the two finest meshes, grids 3 and 4, are able to capture the steepening and folding

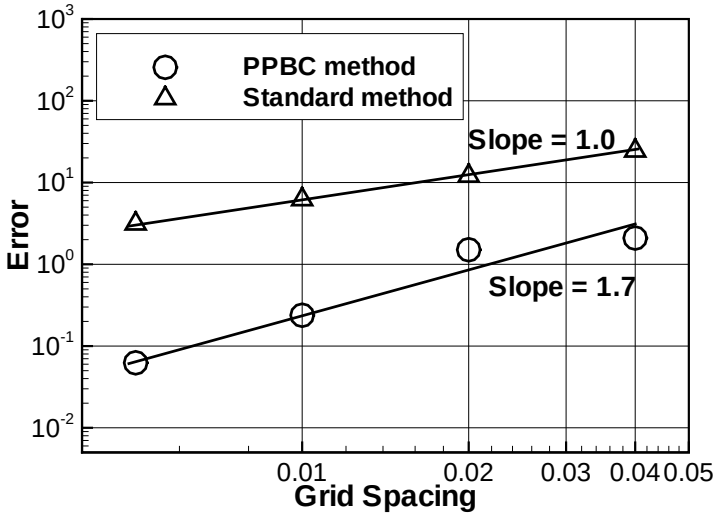
of the free surface that ultimately leads to wave breaking. The two coarser meshes fail completely to resolve this feature of the flow. Second, within the region of steep surface curvature the solution is clearly not grid insensitive since there is considerable difference between the calculated free surface shapes on grids 3 and 4. More specifically, as the grid is refined the surface becomes steeper, reaches higher elevation, and a much sharper cusp develops at the wave front. It is expected, however, that as the surface steepens and begins to fold phenomena at continuously finer scales will emerge, ultimately leading to the formation of a very thin film, and a much finer mesh will be required to resolve them. This finding notwithstanding, however, it is also evident from figure 3.3 that away from the near-cylinder region where the interface steepens and folds grids 3 and 4 yield similar results. A common feature observed in all four grid cases is the formation of a wave that grows and steepens up to a certain degree depending on the grid resolution, although in the case of grids 1 and 2 the resolution is not sufficiently fine to capture the wave folding and breaking. Looking at the insets in the same figure 3.3 showing the wave pattern along with the mesh, it is clear that the wave stops steepening when the crest is resolved by approximately four grid cells. If we consider as the characteristic length of the phenomena half of the wave length (approximately equal to the size of the inset window size in figure 3.3), we note from figure 3.3 that this length is discretized with 6, 12, 24 and 48 grid nodes in grids 1 to 4 respectively. It is evident that grids 1 and 2 fail to capture the wave steepening, which only arises for the first time on grid 3. We can thus conclude that resolving with the present method wave steeping and folding phenomena requires at least 24 grid nodes per one half of the wavelength.

The influence of the level set reinitialization pseudo-time step size $\Delta\tau$ and the total re-initialization time τ in Eqn. (2.8) is illustrated in figure 3.4, which shows the calculated free surface patterns on grid 3. When the thickness of the interface layer is fully reinitialized, i.e. $\tau = \epsilon$ (see discussion in section 3.2 above), and different re-initialization time step sizes are used to solve Eqn. (2.8), $\Delta\tau$=$0.01\Delta x$, $0.05\Delta x$ and $0.10\Delta x$, it is evident from figure 3.4a that the resulting free surface shapes are identical. Note here that since ϵ is equivalent to one grid cell spacing, to fully reinitialize the above cases it was necessary to perform 100, 20, and 10 reinitialization iterations, respectively. Clearly, therefore, the computed solutions are not sensitive to the size of the re-initialization time step as long as the condition $\tau = \epsilon$ is satisfied. As shown in

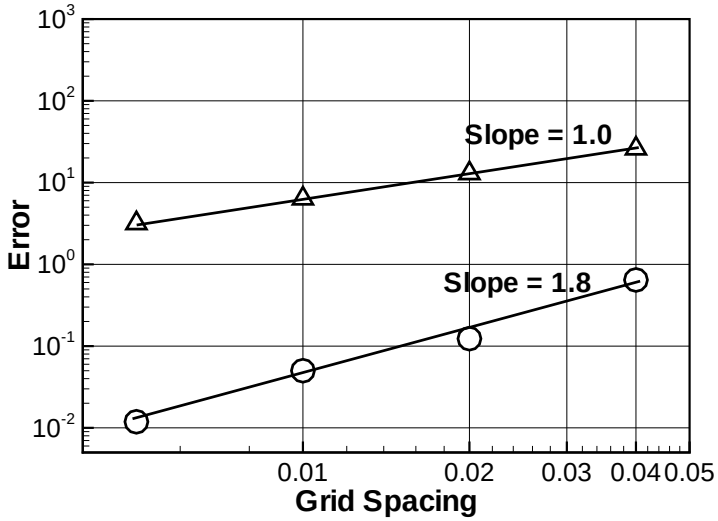
the figure 3.4b, however, the computed free surface pattern is very sensitive to the total reinitialization time. This figure compares the free surface shape at the same instant in time using the same re-initialization time step $\Delta\tau = 0.01$ but progressively longer re-initialization times: from no re-initialization ($n_\tau = 0$) to full re-initialization ($n_\tau = 100$). It is evident from this figure that the result without re-initialization is not accurate but as the re-initialization time increases the computed solution converges monotonically toward the fully re-initialized solution. Moreover, it is also evident that using between 10 to 20 re-initialization iterations, which is equivalent to a reinitialization covering $10 - 20\%$ of the interface thickness, is sufficient to obtain a solution that is very close to the fully converged solution. The non-reinitialized case shows large discrepancies when compared to the fully one. This is an important result to consider when simulating complex free surface interface phenomena, which require the use of large thickness ϵ of the strip in conjunction with small step size $\Delta\tau$. For such cases, full re-initialization would be very expensive but based on the results reported in figure 3.4b re-initializing the solution within a region covering $10 - 20\%$ of the interface thickness should provide a good compromise between accuracy and computational efficiency. We also have to consider that using a large interface thickness ϵ introduces and additional source of error. To understand the sensitivity of the interface thickness to the flow solution the reader is referred to the work of Iafrati and Campana [132]. It was shown that the use of a large interface thickness does not prevent the model from obtaining accurate dynamics of the free surface and the vorticity field of the flow.

Finally, the water exit case for the cylinder problem we simulate herein corresponds to the same configuration as in [152] or [10]. Compared to the previously discussed case, the difference is that the cylinder is initially positioned to be fully submerged underwater at a distance $h = -1.25$ below the free surface and its prescribed velocity is upward and equal to $u = 0.39$. This case has been simulated on grid 3 with a time step size of 0.005 and a free surface interface thickness of 0.03. Figure 3.5 shows comparisons between our simulations and the results of [10] for the free surface position at times $T = 0.4$ and $T = 0.6$. It is evident from this figure that agreement between the two simulations is excellent.

3.2 Force calculation test case: Static buoyant cylinder case



(a) Partially submerged cylinder



(b) Fully submerged cylinder

Figure 3.6: Static buoyant cylinder. Grid convergence of the error of the computed buoyant force due to pressure, for a partly (a) and fully (b) submerged static cylinder.

We employ herein two simple test cases to validate and demonstrate the accuracy of the PPBC method we proposed in section 3.3 for calculating the force acting on a floating structure. For case 1 a circular cylinder is partially submerged by half of its diameter while for case 2 the cylinder is completely submerged under water. The exact solution of both problems is known through the Archimedes' principle, namely the resultant buoyant force is equal to the weight of the displaced fluid. We employ the hydrostatic pressure field to calculate the buoyancy force using the standard method of Borazjani et al. [6] and the proposed PPBC method. Although this test case only evaluates the accuracy of the method we use to calculate the force due to pressure, it is highly relevant to our work since the method for calculating the pressure force is responsible for the numerical difficulties encountered when attempting to extend the FSI-CURVIB method of [6] to two-phase flow simulations. In addition, the pressure force is the main contributor to the total force acting on floating structures in offshore applications and as such the method for calculating it is of critical importance for developing an accurate FSI algorithm.

With reference to figure 3.6, the buoyancy force calculated with the standard method (see [6]), $F = F_S$, is equal to the net force due to pressure computed by integrating along the surface Γ_1. When using the PPBC method, on the other hand, F is the resultant projected pressure force as described in section 2.1.6 in the surface Γ_2. For both cases we consider herein, the exact force is known and, therefore, the error between the two methods for calculating the force and the exact value can be defined as follows: $E = |F - F_{exact}| / F_{exact}$. In figure 3.6 we show in a log-log plot the convergence of this error calculated in a series of successively refined uniform meshes with spacing: $0.04D$, $0.02D$, $0.01D$, and $0.005D$. It is evident from this figure that for both cases and all grids the standard method for calculating the force grossly over predicts the buoyancy force on the cylinder. Furthermore, while this method converges with grid refinement, the rate is very slow and the method is at best first-order accurate as expected from the analysis we presented in section 2.1.6. The proposed PPBC method, on the other hand, yields on all grids errors lower by nearly one order of magnitude than the standard method and its convergence rate is clearly better than first order for both cases –in fact the results in figure 3.6 suggest a nearly second-order convergence for the PPBC method. Therefore, the results in figure 3.6 confirm the theoretical arguments we presented in

section 2.1.6 regarding the relative accuracy of the PPBC and the method of [6] for calculating the buoyancy force. The superior performance of the PPBC method will be further demonstrated in subsequent test cases where it will be used to solve complex FSI problems involving bodies interacting with a free surface.

3.3 FSI case 1: Free heave decay test of a circular cylinder

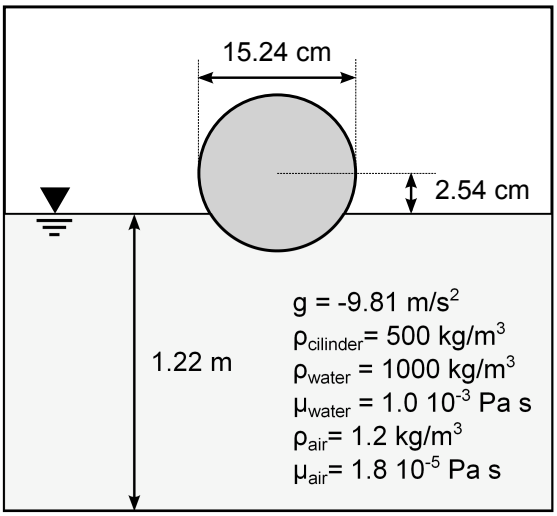


Figure 3.7: Free heave decay test of a circular cylinder. Schematic description of the cylinder configuration studied experimentally by Ito [153].

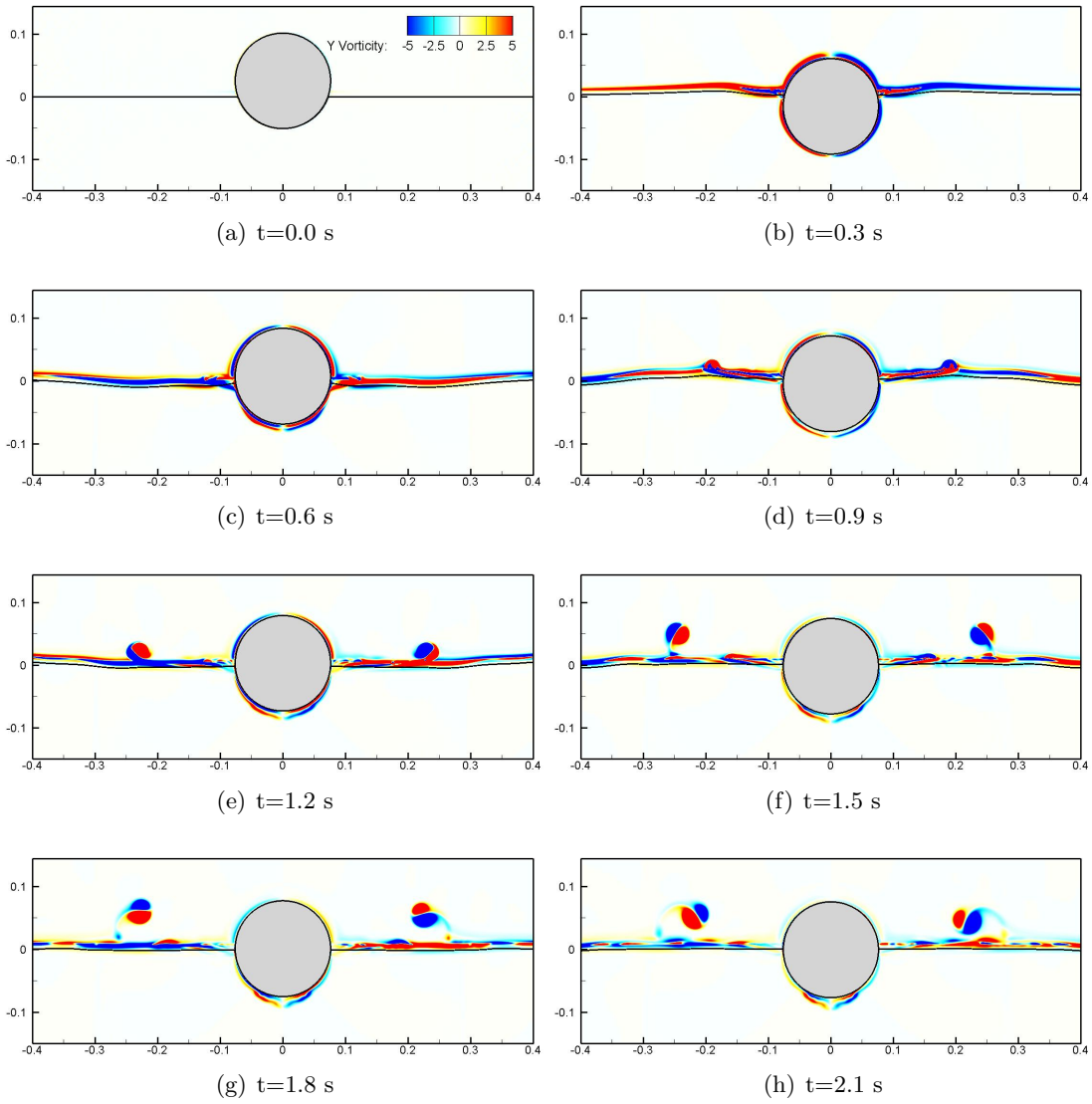
With this and the subsequent presented test cases we seek to demonstrate the predictive capabilities of the coupled FSI, CURVIB, level set formulation for bodies of increasing complexity interacting with a free surface. We consider herein the same case as that studied experimentally by Ito [153], which is suited for validating the FSI algorithm in a translational single degree of freedom. A horizontal circular cylinder of diameter $D = 0.1524m$ and density $\rho = 500kg/m^3$ is partially submerged with its center positioned 0.0254m above the free surface of a rectangular channel (see figure 3.7 for a schematic representation). The computational domain is a 27.4m long channel, 2.59m wide, with a depth of 1.22m and the water in it is initially stagnant. The cylinder movement is restricted to the vertical degree of freedom, and it is allowed to oscillate freely. The Reynolds number as defined in equation (2.3) based on the cylinder diameter

Table 3.2: Description of the four grids employed in the free heave decay test of a circular cylinder and the corresponding interface thickness ϵ employed.

Grid	**Grid size**	**Near cylinder spacing**	**Interface thickness ϵ**
1	$320 \times 220 \times 8$	0.040D	0.131D
2	$440 \times 260 \times 8$	0.020D	0.065D
3	$680 \times 420 \times 8$	0.010D	0.039D
4	$1130 \times 720 \times 8$	0.005D	0.039D

and its maximum velocity is equal to $Re = 30,000$. We assume that the cylinder spans the entire width of the channel and that the side walls are slip walls. We carry out 3D LES but using a coarse grid in the spanwise direction, that is, essentially assuming that the flow is 2D. Tests with finer grid in the spanwise direction did not yield any appreciable differences in the simulated response of the cylinder, which is the only quantity recorded experimentally. More specifically, we employ four non-uniform meshes: grid 1 with $320 \times 160 \times 8$; grid 2 with $440 \times 260 \times 8$; grid 3 $680 \times 420 \times 8$; and grid 4 with $1130 \times 720 \times 8$ nodes in the horizontal, vertical, and spanwise directions, respectively. The grids are uniform in a rectangular region centered around and containing the body defined by [-0.3,0.3] in the horizontal direction and [-0.2,0.2] in the vertical direction, and have a grid spacing equal to $0.04D$, $0.02D$, $0.01D$, and $0.005D$ for grids 1 to 4, respectively. In the domain outside this uniform grid region, the grids are stretched using a hyperbolic function and the ratio never exceeds 1.05. The grids description as well as the interface thickness ϵ used in each case are summarized in table 3.2. All simulations for this case were carried out using the LC-FSI algorithm and a time step size of $0.0005s$.

Several snapshots on grid 4 of the calculated position of the cylinder and the free surface along with vorticity contours are presented in figure 3.8. It is seen in this figure that as the cylinder impinges on and begins to deform the free surface thin layers of positive and negative vorticity form along the interface. A system of waves is generated and starts propagating away from the cylinder in the horizontal direction. The so induced curvature of the free surface and the associated pressure gradients that develop in the flow give rise to a complex vorticity field as evident in figure 3.8d-h. A distinct characteristic of the flow on the air side is the formation of dipole vortical structures that initially rise away from the surface but then alter their propagation direction to



(a) t=0.0 s (b) t=0.3 s

(c) t=0.6 s (d) t=0.9 s

(e) t=1.2 s (f) t=1.5 s

(g) t=1.8 s (h) t=2.1 s

Figure 3.8: FSI simulation of heave decay of a circular cylinder. Several snapshots of the calculated position of the cylinder, the free surface, and corresponding out-of-plane vorticity contours are shown in these figures. The solution has been obtained on grid 4, which has near-cylinder spacing equal to $\Delta x = \Delta y = 0.005D$, and a time step size of $0.0005s$ has been used.

curve downward toward the surface. This complex flow patterns in the air flow, which could only be predicted in grid 4, are the result of flow separation off the crest of the

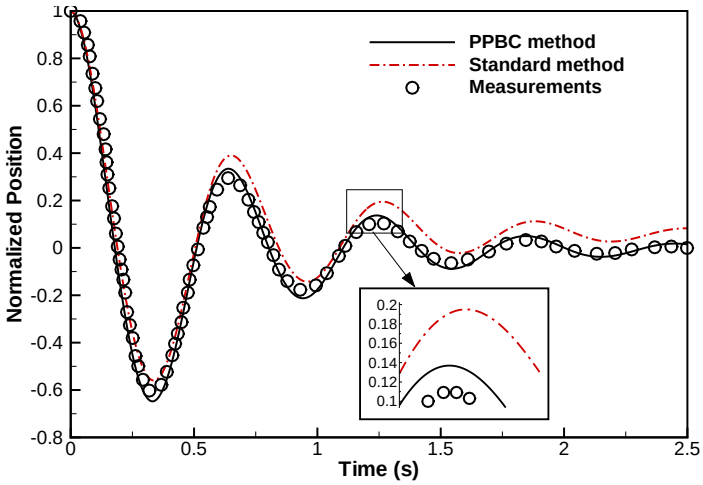


Figure 3.9: FSI simulation of heave decay of a circular cylinder. Normalized position of the cylinder computed on grid 3 with the PPBC method, the standard method of [6], and the experimental data of Ito [153]. The time step size employed is $0.0005s$ and the interface thickness ϵ is $0.006m$.

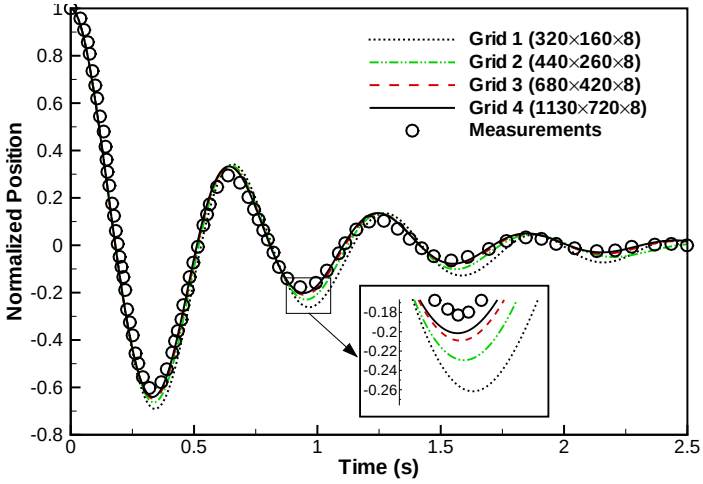


Figure 3.10: FSI simulation of heave decay of a circular cylinder. Normalized position of the cylinder for several grid refinements (grids 1 to 4) and the experimental data of Ito [153]. The time step size employed is $0.0005s$.

waves induced by the motion of the cylinder (see figure 3.8d). Even though experimental measurements to allow us to assess the accuracy of these findings are not available for this case, it is important to note that similar vortex dipole structures have also been

observed in a recent numerical study by Iafrati et al. [154] who simulated the modulation instability process of water waves.

To analyze the accuracy of the FSI algorithm and demonstrate the accuracy of the proposed PPBC method for calculating the force we compare in figures 3.9 and 3.10 the measured ([153]) and computed normalized position of the cylinder as function of time. In figure 3.9 we compare the accuracy of the PPBC method with that of the standard method for calculating the force on the body by carrying out FSI simulations with both methods on grid 3. It is seen in this figure that the PPBC method is able to accurately predict the frequency as well as the amplitude of oscillation. On the other hand, when the standard method is implemented the resulting position of the cylinder is shifted upwards and does not agree well with the measurements. This finding is consistent with the previously discussed (see section 3.3) limitation of the standard method, which over predicts the net force acting on the structure. To quantitatively compare the accuracy of the two methods, the temporally averaged error of the position of the cylinder is calculated using the following expression:

$$E = \frac{1}{n}\sqrt{\sum_{i=1}^{n}\left(z(t_i) - z_{exp}(t_i)\right)^2} \tag{3.1}$$

where z is the vertical position of the center of the cylinder with respect to the calm free surface, the subindex exp indicates the experimental results ([153]), and n is the total number of time steps. The error for the PPBC method is $E = 6.13 \cdot 10^{-4}$ while for the standard method nearly one order of magnitude higher $E = 1.28 \cdot 10^{-3}$. Therefore, this test case as well as the convergence studies we reported in the previous section of this paper establish clearly the superiority of the PPBC method for calculating the pressure force acting on floating structures. For that this method is used in all subsequently reported test cases.

Finally, in figure 3.10 we report the results of a grid sensitivity study using the PPBC method in grids 1-4. It is evident that as the grid is refined the solution converges monotonically toward the experimental measurements, thus, establishing the accuracy of our coupled FSI formulation. One small discrepancy between simulations and experiments is that the latter appear to exhibit a slightly higher damping factor, which may be

explained by the friction present in the experimental apparatus which is not taken into account in the simulations. Note that the damping coefficient b_t in equations (2.12) is set to zero. Overall, however, the computed results are in excellent agreement with the experiments.

3.4 FSI case 2: Free roll decay of a rectangular barge

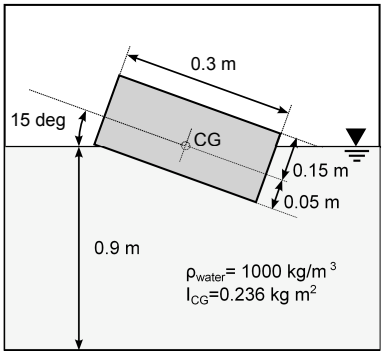


Figure 3.11: FSI simulation of roll decay of a rectangular barge. Schematic description of the barge configuration studied experimentally by Jung et al. [155].

In this case we seek to validate the FSI algorithm under a single degree of freedom in the rotational direction by applying it to simulate a rectangular barge like structure identical to that studied experimentally by Jung et al. [155]. The barge is $0.9m$ long, $0.3m$ wide and $0.2m$ tall, and it is positioned in a $35m$ long rectangular channel with a $0.9m$ wide section and water depth of $0.9m$. The barge is allowed to rotate freely with respect to its center of gravity (CG), which coincides with the calm free surface water level, and its rotational inertia is $I = 0.236kgm^2$. It is initially inclined at an angle of 15 degree with respect to the free surface level and its draft is $0.05m$. The configuration is illustrated in figure 3.11.

Similar to the heave cylinder case presented in the previous section, this is a 3D simulation where the barge width coincides with the width of the channel. Free-slip conditions are applied at the side walls of the channel and the Reynolds number based on the barge width, and the maximum value of velocity achieved by its corner is equal to $Re = 45,000$. A two-region grid structure is employed for this case as for previous

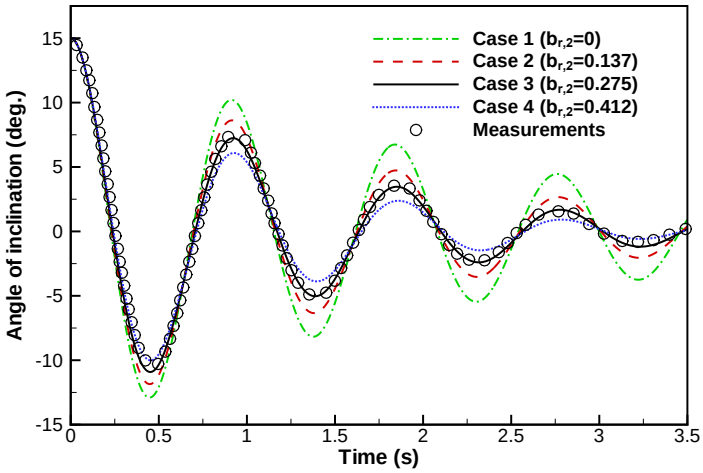


Figure 3.12: FSI simulation of roll decay of a rectangular barge. Angle of inclination of the barge for the several cases of artificial damping, and the experimental data of Jung et al. [155]. The results have been obtained on a grid with uniform near-body spacing of 0.001m and a time step of 0.0005s.

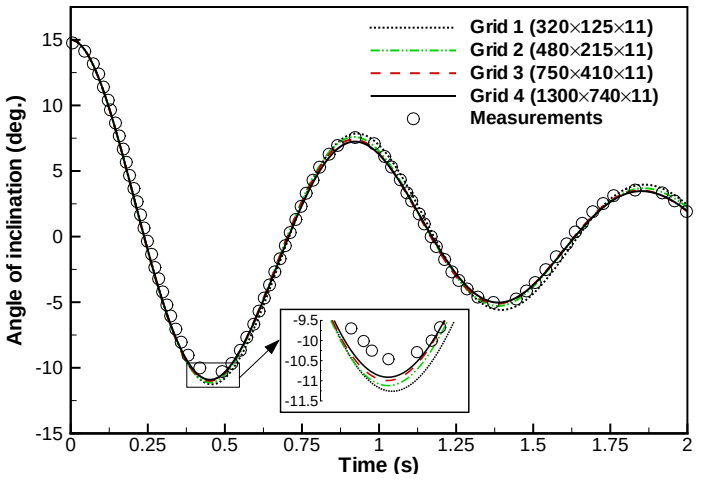
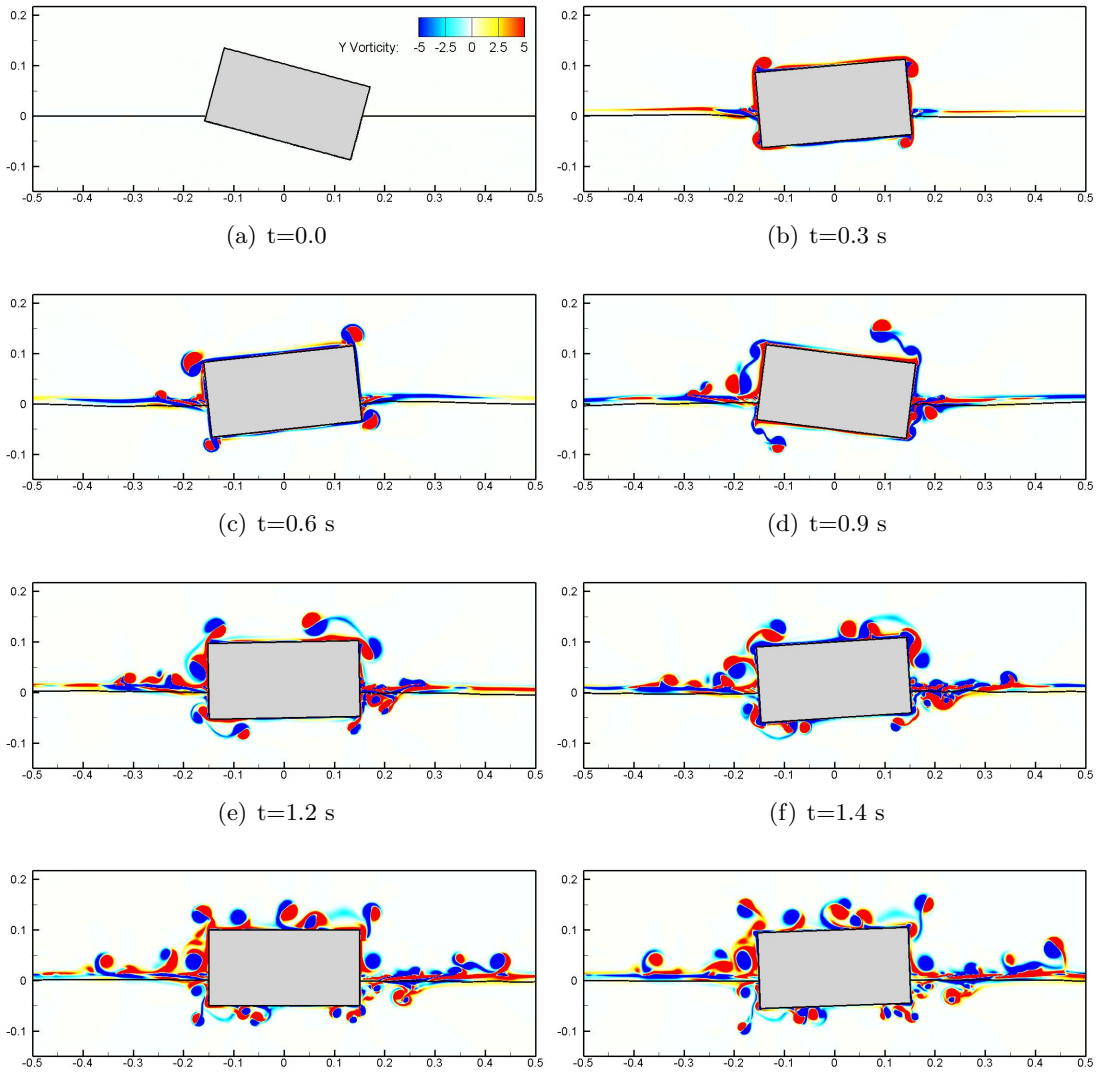


Figure 3.13: FSI simulation of roll decay of a rectangular barge. Computed angle of inclination of the barge on a series of refined grids (grids 1 to 4) and for fixed value of $b_{r,2} = 0.275$, and the experimental data of Jung et al. [155]. The grid spacing near the barge for each of the four grids is $0.008m$, $0.004m$, $0.002m$ and $0.001m$, respectively. The time step size employed is $0.0005s$ and the interface thickness ϵ is $0.02m$

cases. In the inner region centered around the structure ($[-0.8, 0.8]$ in the horizontal direction and $[-0.3, 0.3]$ in the vertical direction) the grid is uniform with grid spacing



(a) t=0.0 (b) t=0.3 s

(c) t=0.6 s (d) t=0.9 s

(e) t=1.2 s (f) t=1.4 s

(g) t=2.1 s (h) t=2.4 s

Figure 3.14: FSI simulation of roll decay of a rectangular barge. Several snapshots of the calculated position of the barge, the free surface, and corresponding out-of-plane vorticity contours are shown in these figures. The solution has been obtained on a grid with near-body spacing equal to $0.001m$, the time step size is $0.0005s$, the interface thickness ϵ is $0.02m$, and the damping coefficient $b_{r,2}$ is 0.275.

both in the horizontal and vertical directions equal to $0.001m$. Outside of this inner region the grid is gradually coarsened away from the structure using the hyperbolic

stretching function. The overall grid size we employ herein consist of approximately 10.5 million, with $1300 \times 740 \times 11$ nodes in the horizontal, vertical, and spanwise directions, respectively. As for the previous case, we carry out 3D LES but without resolving the spanwise direction, essentially assuming that the underlying flow phenomena that govern the dynamics of the structure response are 2D. The time step for the simulation is $0.0005s$ and the LC-FSI algorithm is used.

The damping of the roll motion of the barge can in general be due to the combined action of viscous dissipation of the energy transferred from the barge to the flow and the friction of the experimental apparatus [155]. Our method can in principle resolve the former effects since it can capture the complex vortical field that emerges as the barge oscillates, including vortex-barge and vortex-vortex interactions (see figure 3.14), which are responsible for dissipating the energy transferred from the barge to the flow. However, the apparatus-specific frictional damping is not know a priori and needs to be incorporated in the simulation by specifying the damping coefficient $b_{r,2}$ in the equation of motion (2.13). While in the previous case we found that friction effects were small and a value of $b_{r,2} = 0$ was adequate for obtaining accurate solutions, in the present case the computed amplitude of oscillations is found to be sensitive to the value of $b_{r,2}$. To investigate this sensitivity, we consider four test cases with increasing value of damping coefficient: $b_{r,2} = 0$, $b_{r,2} = 0.1375$, $b_{r,2} = 0.275$, and $b_{r,2} = 0.425$ for cases 1 to 4, respectively. The angular position of the barge as a function of time calculated for each of the four cases using the same computational grid for all cases is compared with the experimental results of [155] in figure 3.12. It is evident from this figure that the numerical model can resolve the experimentally documented natural frequency of the structure with excellent accuracy regardless of the value of $b_{r,2}$. The value of friction coefficient, however, does affect the amplitude of the oscillation. For example, for the $b_{r,2} = 0$ case the amplitude is over-predicted and as $b_{r,2}$ is increased the amplitude, as one would anticipate, is systematically dampened. The best fit with the experimental data is clearly obtained for Case 3. This test case underscores the difficulties involved in using such a case to validate a coupled FSI code since the friction of the experimental apparatus is shown to affect the accuracy of the results. Nevertheless, the results in figure 3.12 clearly show that the frequency of the oscillation does not depend on $b_{r,2}$ and is resolved with good accuracy for all four cases. Therefore, the results presented

herein can be best viewed as a calibration of the friction coefficient in our coupled FSI model for this case under the assumption that our solver is able to accurately resolve viscous effects due to the flow. To explore the validity of this assumption, and since the viscous damping depends on the ability of the computational grid to resolve the underlying flow structures, we carry out another sensitivity study using the value of $b_{r,2}$ that yielded the best agreement with the experimental data in figure 3.12 ($b_{r,2} = 0.275$) but systematically refining the computational grid. The results of this study are summarized in figure 3.13, which compares the computed amplitude on the four successively finer grids with the same value of $b_{r,2}$ and the same computational time step and for the first 2 secs of the decay process. It is evident from this figure that the calculated dynamic response of the barge is essentially the same on all four grids. Only minor differences are observed between the coarser grids 1 and 2 and the two finer grids 3 and 4. The solution, however, converges monotonically as the grid is refined.

The flow patterns that develop in the air and water phases during the various phases of the barge oscillation are illustrated for $b_{r,2} = 0.275$ in figure 3.14, which shows instantaneous contours of spanwise vorticity along with the corresponding shape of the free surface. As seen in this figure, the flow at the early stages of the oscillation is dominated by vortex dipoles shed from the four corners of the barge align with thin layers of vortices generated along the undulating free surface. The mechanism for generating free surface vorticity appears similar to that we discussed in the previous cylinder case and is associated with the unsteady system of wave induced by the motion of the structure. As the oscillation advances, the corner-shed dipoles are seen to, depending on the specific corner of the barge off which they were shed, impinge either on the free surface or the flat surface of the barge giving rise to a very complex vortical flow. Another mechanism for enhancing the complexity of the flow is observed at the later stages of the oscillation ($t > 0.9s$) when the barge-induced waves propagate away from the structure and give rise to vortex dipoles that are shed off the free surface and rise into the air phase. This is clearly seen in figure 3.14(h), which shows a series of three consecutive dipoles propagating away from the structure and rising into the air.

In summary, the results we have presented herein demonstrate the ability of the coupled FSI, level set formulation to simulate rotational degrees of freedom and resolve the very complex flow phenomena that arise as a result of the coupled interaction

between the floating structure and the free surface. Our simulations along with the results we presented above for the cylinder case and recent numerical simulations of Iaffratti et al. [154] further show that the formation of vortex dipoles in the air-phase induced by waves propagating along the free surface appears to be a ubiquitous feature of such flows. Our results suggest that such phenomena occur regardless of how the waves have been generated and clearly point to the need for further studies to probe their underlying physics.

3.5 FSI case 3: Free falling wedge

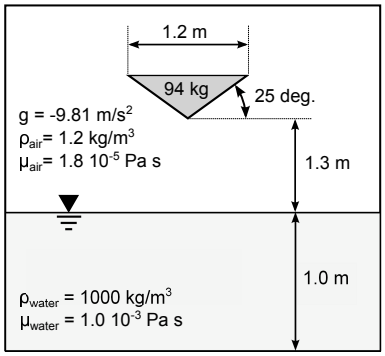


Figure 3.15: Schematic description of the free falling wedge configuration studied experimentally by Yettou et al. [156].

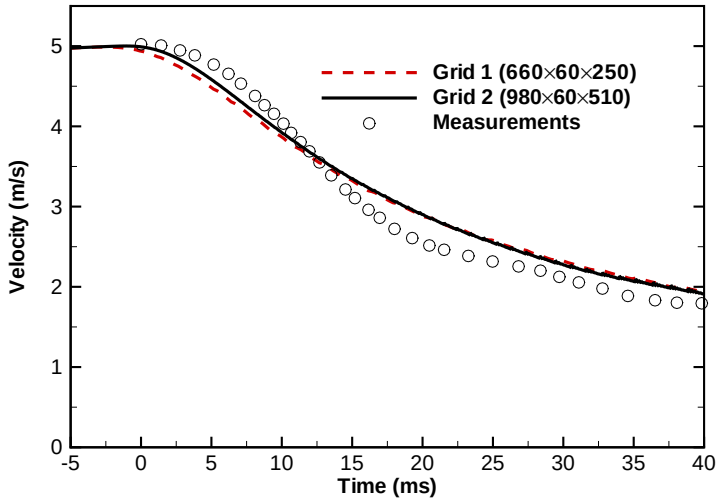
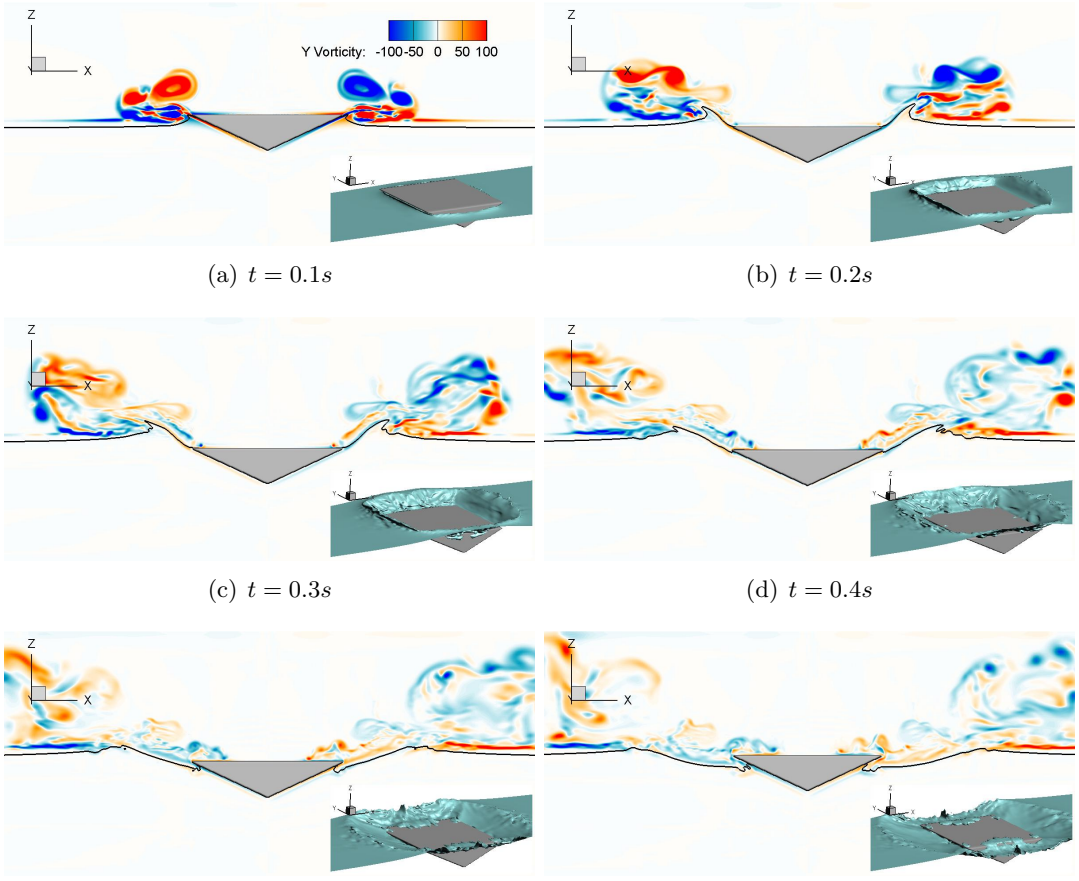


Figure 3.16: FSI simulation of a free falling wedge. Velocity of the wedge at the initial stage of the impact computed with the PPBC method on grids 1 and 2, and the experimental data of Yettou et al. [156].



(a) $t = 0.1s$ (b) $t = 0.2s$

(c) $t = 0.3s$ (d) $t = 0.4s$

(e) $t = 0.5s$ (f) $t = 0.6s$

Figure 3.17: FSI simulation of a free falling wedge. Several snapshots of the calculated position of the wedge, the free surface, and corresponding out-of-plane vorticity contours are shown in these figures at the cross middle plane ($Y = 0$). A small 3D view of the wedge is superposed. The solution has been obtained on grid 1, which has near-body spacing equal to $\Delta x = \Delta z = 0.005L$, and a time step of $0.00025s$ has been used.

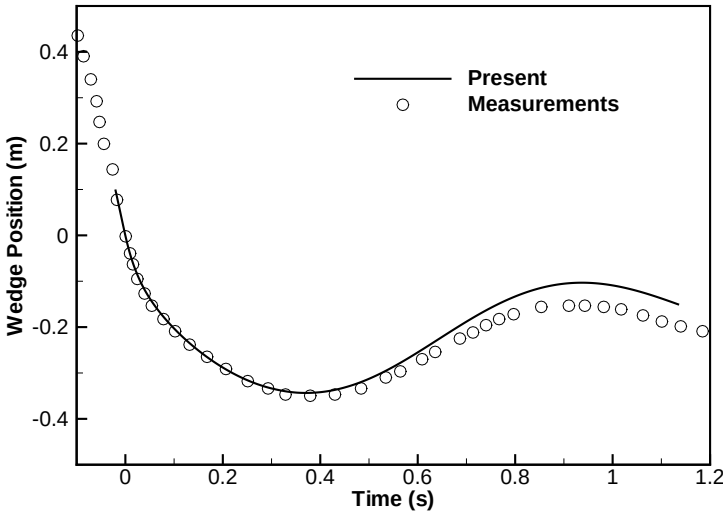


Figure 3.18: FSI simulation of a free falling wedge. Vertical position of the wedge computed with the PPBC method on grid 1, and the experimental data of Yettou et al. [156].

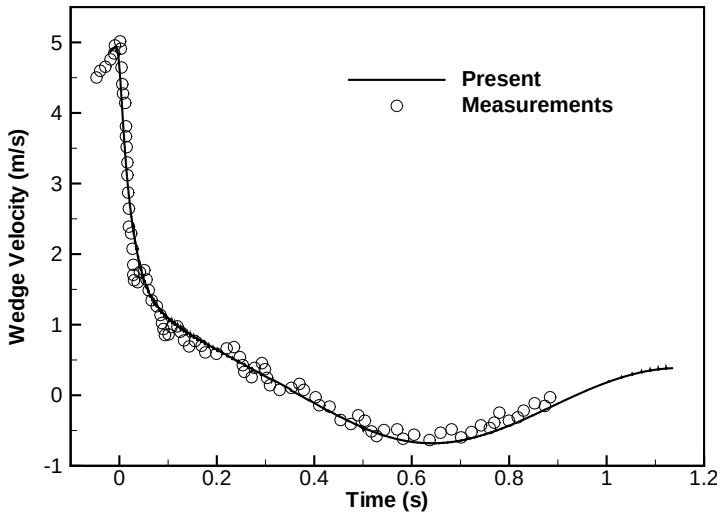


Figure 3.19: FSI simulation of a free falling wedge. Vertical velocity of the wedge computed with the PPBC method on grid 1, and the experimental data of Yettou et al. [156].

The last test case we consider herein involves the simulation of a geometrically complex body, a 3D wedge, falling freely and impinging with its pointed keel into the free surface. The geometrical complexity of the structure along with the large pressure and velocity gradients that develop as the structure's keel impinges on the surface make this the most challenging of all the cases we have studied in this paper. The specific test case we simulate corresponds to that studied experimentally by Yettou et al. [156], who reported detailed data sets of the wedge velocity and position as function of time. In this work we focus on both the dynamics of the wedge motion and the analysis of the coupled air and water complex flow patterns, which have not been addressed in previous studies. For the most part, previous works on wedge impact problems have used laboratory experiments [157, 158, 156, 159] and computational models [160, 161, 162, 157, 158, 163, 82, 100] to investigate the dynamics of the wedge motion but reported only limited information about the free surface and flow patterns.

The simulated 3D body has a symmetric wedge-shaped section with a 25 degree dead-rise angle. Its mass is $94Kg$, which is equivalent to a structural density of $\rho = 466.6Kg/m^3$. Initially, the keel of the wedge is placed 1.3 m above the free surface and starts moving downwards under the action of gravity towards the free surface.

The geometrical configuration of the simulation is illustrated in figure 3.15. The flow Reynolds number based on the wedge length and its maximum velocity is of the order of $Re = 5 \cdot 10^5$.

In the experiment of [156], the length of the wedge in the spanwise direction is $L = 1.2m$, while the channel width is $2m$, i.e. there is a gap of $0.4m$ between the wedge and the lateral walls of the channel. In our simulation these gaps are taken into account and resolved by the spanwise grid, which consists of 60 nodes with 12 nodes resolving each gap. The lateral boundaries of the computational domain as well as the wedge surface are treated as no-slip walls.

The computational domain is a $30m$ long channel, $2m$ wide with and a water depth of $1m$, and the wedge is located at the center section of the channel. Two grids of different resolution have been used, both of which exhibit a similar inner/outer region structure we employed in the previous simulations. In grid 1 the constant grid spacing near the structure is of $0.005L$ both in the longitudinal and vertical directions while in grid 2 the corresponding grid spacing is of $0.0025L$. Both meshes have 60 grid nodes in the span-wise direction and the overall grid size is approximately 10 and 30 million, respectively.

The dimensions in the horinzontal (x), spanwise (y) and vertical (z) directions for grid 1 are $660 \times 60 \times 250$ and for grid 2 $980 \times 60 \times 510$. The inner rectangular region of constant grid spacing that encloses the wedge is of size [-0.6, 0.6] in the horizontal direction, and [-0.6, 0.4] in the vertical direction. In the outer region, the stretching ratio in the x direction has been kept reduced (1.001) and constant along the following interval ($0.6 < |x| < 2.5$) which coincides with the area of high vorticity generated by the wedge. Away from this interval the stretching ratio is increased progressively up to a limit of 1.05.

Due to the large forces that develop on the structure during impact, this test case was proven especially challenging to simulate for the FSI algorithm. Using loose coupling FSI did not yield converged solutions on any of the computational grids and regardless of how small time step we selected. During these initial tests with the loose coupling algorithm, the FSI scheme was consistently found to become unstable approximately between $0.1s$ and $0.2s$ from the start of the simulation. Converged solutions were obtained only when the strong coupling FSI algorithm was employed and with reduced

time steps equal to $\Delta t = 0.00025s$ and $\Delta t = 0.00005s$ for grids 1 and 2, respectively. Moreover the use of the Aitken acceleration technique was found essential for increasing the efficiency of the string-coupling FS algorithm, reducing the number of SC-FSI sub-iterations required for achieving convergence by a factor of 2-3: from 10-12 without Aitken to only 4-5 with Aitken.

Another important issue for enhancing the robustness and accuracy of the algorithm in this case was the need to use an interface thickness ϵ equivalent to 8 grid cell spacings, and a reduced reinitialization time step of $\Delta\tau = 0.02\Delta x$. The overall re-initialization time τ was taken to be approximately 10% of the time required for a full reinitialization. As shown before, this value is adequate for obtaining accurate solutions.

The most challenging, from the computational standpoint, stage of this problem is the first $40ms$ of the process during which the wedge impinges on the free surface and decelerates rapidly from $5m/s$ to $2m/s$. During this initial stage we carry out simulations on both the coarse and fine meshes and the calculated temporal variation of the the wedge velocity is compared with the experiments of [156] in figure 3.16. It is evident from this figure that the computed results on both grids are in good agreement with each other and in reasonable overall agreement with the experimental measurements.

To demonstrate the complex flow patterns that develop as the wedge impinges on the free surface, we show in figure 3.17 a series of snapshots of the simulated free surface patterns along with contours of the Y component of the vorticity on the $Y = 0$ plane. As seen in this figure, as the wedge impinges on the surface it generates a wave on each of its sides. These waves steepen as they propagate away from the wedge and ultimately break leading to massive separation off the the wave cusp and production of large-scale vortical structures of opposite sign in the air phase. This breaking wave generated vorticity is entirely confined in the air phase and ultimately breaks up into smaller scales giving rise to a highly turbulent flow state that rises for several wedge heights above the free surface. These results are entirely consistent with the already discussed findings of Iafrati et al. [154] who showed that in breaking wave phenomena most of the energy is transferred from the wave to the air phase. [154] also showed that vortical structured generated in the air phase near the free surface during wave breaking are able to penetrate into the air phase at significant heights above the surface,

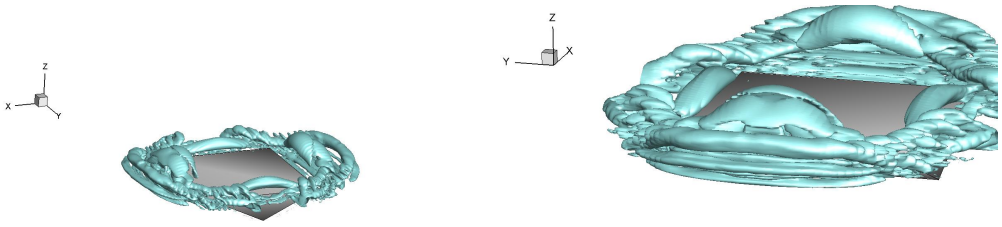
a phenomenon which is also observed in our simulations. We note, however, that due to the impact of the wedge on the surface, the condition we have simulated herein is more severe than previously studied cases with enormous amounts of energy transferred from the wedge, to the breaking waves and ultimately to the air phase.

In figures 3.18 and 3.19, we compare the simulated temporal variation of the wedge position and velocity over a longer time interval–namely, the first $1.2s$ of the process– with the experimental measurements of [156]. The simulation results shown in these figures have been obtained on grid 1 since as shown in figure 3.16 this grid is adequate for accurate predictions of the structure response. It is readily seen from figures 3.18 and 3.19 that the computed results are in very good agreement with the measurements, capturing both the frequency and amplitude of the wedge oscillation with good accuracy. These results further reinforce the accuracy of the PPBC method we developed herein for calculating the pressure force acting on floating structures. However, as the simulation advances in time, it is observed an increasing discrepancy in the prediction of the wedge oscillation amplitude (between $0.7s$ and $1.1s$). A reason can be extracted by looking at the insets in figure 3.17 that show the 3D structure of the zero level set to illustrate the complex phenomena that are simulated during wedge impingement and wave breaking. As seen in these snapshots, when the wedge approaches maximum depth water over topping is observed. During this process a significant amount of water is trapped on top of the structure as it starts moving upward. As the layer of trapped water spreads laterally to occupy a larger area its thickness is reduced. The level set approach we employ herein is an inherently diffused interface method and as such free surface phenomena can be captured up to scales for which the resolution of the grid is sufficiently high to resolve the air-water interface. When the water thickness becomes unresolvable, consecutive interfaces tend to merge meaning that the water mass balance is not fully conserved. Ultimately this under-resolved layer of over-topped water will spuriously disappear reducing the gravity force acting on the wedge. This issue could, at least in part, be responsible for the slight over-prediction of the vibration amplitude observed in figure 3.18 at times greater than $0.8s$ when the wedge is moving upward. The only practical approach to correctly resolve this problem is to employ adaptive mesh refinement, which will enable local refinement of the mesh at resolution sufficiently fine for resolving the interface thickness of over-topped water. This is beyond the scope

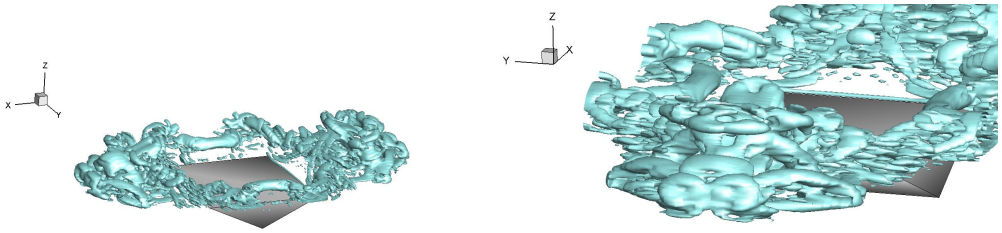
of the present work but will be pursued in future work. Also, another challenge for accurately resolving over-topping phenomena stems from the need to take into account the surface tension force, which will undoubtedly become important in determining the dynamics of thin pockets of water trapped in the air phase on top of a solid structure. Our level set approach can incorporate the surface tension force (see eqn 2) but this needs to be done in conjunction with sufficient grid resolution to resolve local variations of the interface at scales for which surface tension forces become important. This is also beyond the scope of our work. Nevertheless, this resolution limitation notwithstanding, the proposed method is able to simulate the dynamic response of the structure with reasonable accuracy.

Finally, in figure 3.20 we present several snapshots of the simulated 3D coherent structures visualized with iso-surface of q-criterion [164] to illustrate the richness of the ensuing air and water dynamics as the waves off the sides of the wedge steepen and break. As seen, the massive separation zone induced by the breaking waves is dominated by a series vortex loops, arch vortices, and hairpin vortices. It is also evident from sub-figures 3.17 (a) and (b), corresponding to times $t = 0.1s$ and $t = 0.2s$, that the 3D complexity of the simulated flow grows rapidly as the waves steepen and begin to break and the flow transitions from laminar to turbulent.

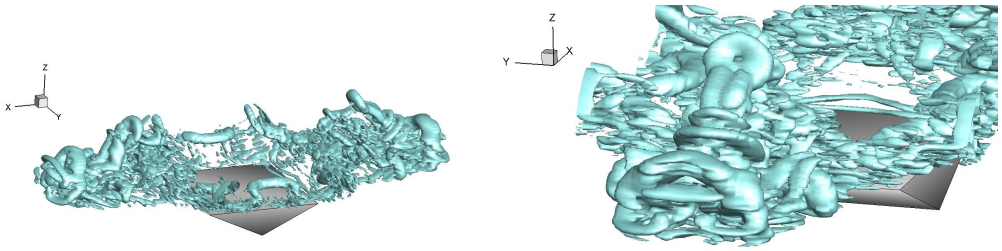
In summary, the results we presented herein show that the proposed coupled FSI level set approach is able to accurately simulate a very challenging case involving large forces on the impinging structure, wave-breaking, and overtopping. The ability of the method to also resolve the complex dynamics of the flow that emerges in the air phase during wave breaking was also illustrated.



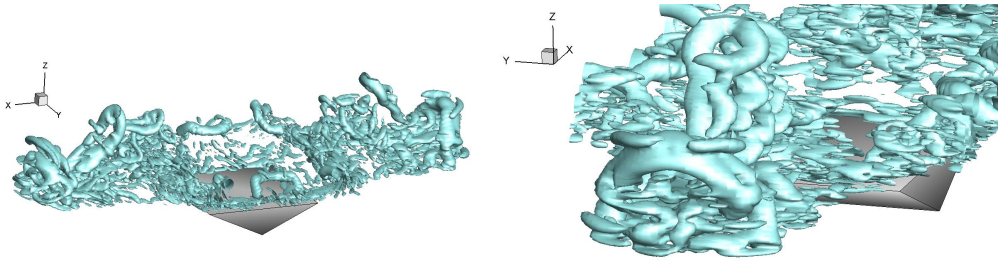
(a) $t = 0.1s$



(b) $t = 0.2s$



(c) $t = 0.3s$



(d) $t = 0.4s$

Figure 3.20: FSI simulation of a free falling wedge. Snapshots showing coherent structures in the flow visualized by the Q criterion (Q=-200). The right figures show a closer view of the coherent structures near the area where wave breaks. The solution has been obtained on grid 1, which has near-body spacing equal to $\Delta x = \Delta z = 0.005L$, and a time step of 0.00025s has been used.

Chapter 4

Validation of the wave generation method: far-field/near-field coupling

In this section, we employ the proposed far-field/near-field coupling algorithm to simulate a number of water wave cases for which the analytic solution is known. We first test the ability of the pressure forcing method to generate monochromatic waves in a 2D rectangular channel and a directional waves in a 3D basin. Subsequently, we evaluate the approach 1 far-field/near-field coupling algorithm by incorporating various 3D directional wave cases, including a broadband wave spectrum, initially originated at the far-field domain, to the near-field domain. Finally, we validate the approach 2 coupling algorithm by simulating a wave case consisting of a broadband spectrum.

4.1 Forcing method validation case: monochromatic waves

We employ a simple test case of linear monochromatic waves in a 2D rectangular channel of constant depth to validate and demonstrate the accuracy of the the wave forcing method described in section 2.4. To analyze the sensitivity of the method when generating waves with different wavelengths and wave slopes we consider six different wave cases, summarized in table 4.1. In the three first cases, case 1, case 2, and case 3 the

Table 4.1: Description of the parameters used in each of the monochromatic wave cases

Wave Case	$L[m]$	$A[m]$	Ak_x	ϵ_x
1	0.6	0.005	0.05	0.3
2	1.2	0.01	0.05	0.6
3	2.4	0.02	0.05	1.2
4	1.2	0.0019	0.01	0.6
5	1.2	0.0095	0.05	0.6
6	1.2	0.0191	0.10	0.6

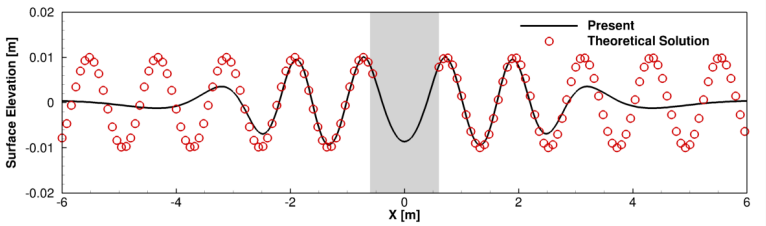
Table 4.2: Description of the five grids employed in the monochromatic wave cases

Grid	**Grid size**	**Streamwise spacing [m]**	**Minimum vertical spacing [m]**
1	301×180	$L/7.5$	0.005
2	601×180	$L/15$	0.005
3	1200×180	$L/30$	0.005
4	601×140	$L/15$	0.01
5	601×100	$L/15$	0.02

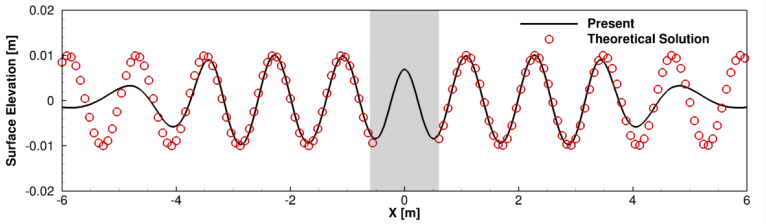
wavelength is, respectively, $L = 0.6m$, $L = 1.2m$, and $L = 2.4m$ while keeping a constant wave slope of $Ak_x = 0.05$. In the remaining three wave cases, case 4, case 5, and case 6, the value that is maintained constant is the wavelength $L = 1.2m$ and the slope varies as follows, $Ak_x = 0.01$ for case 4, $Ak_x = 0.05$ for case 5, and $Ak_x = 0.1$ for case 6. Note that the wave slope, also known as wave steepness, can be defined as the wave amplitude A times the wavenumber k. For a given linear wave of amplitude A and length L the analytical solution is known through the linear wave theory provided in equation (2.62).

For all wave cases, we consider herein a 2D domain of length equal to $40L$, water depth of $2m$, air column above the water of $1m$, and a gravitational acceleration of $g = 9.81m/s^2$. Such large domain length was taken with the purpose of preventing, or at least minimizing, possible wave reflections at the side walls which are also treated with the sponge layer method.

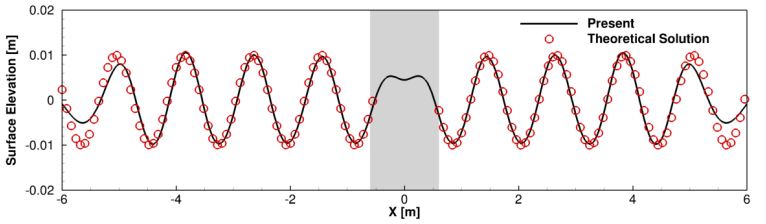
We employ five different non-uniform meshes: grid 1 with 301×180; grid 2 with 601×180; grid 3 1201×180; grid 4 with 601×140; and grid 5 with 601×100 nodes in the horizontal and vertical directions, respectively. While the horizontal grid spacing is constant all along the domain, the vertical spacing is only constant along a rectangular



(a) Time $t = 4s$



(b) Time $t = 6s$



(c) Time $t = 8s$

Figure 4.1: Generation of monochromatic waves. Free surface elevation at three instances in time for wave case 2. The time step used is $0.002s$. The grey shaded area represents the source region where the solution cannot be representat by the anaylitical solution.

region centered on the undisturbed free surface defined by $Z = [-0.1m, 0.1m]$. Within this region the vertical grid spacing is $0.005m$ for grid 1, grid 2, and grid 3, $0.01m$ for grid 4, and $0.02m$ for grid 5. Outside of this region the vertical grid spacing increases progressively with a stretching ratio limited to 1.05. The horizontal grid spacing is $L/7.5$ for grid 1, $L/15$ for grid 2, grid 4, and grid 5, and $L/30$ for grid 3. The description of the five meshes is summarized in table 4.2.

The time step of the simulation for all cases is $0.002s$, the thickness ϵ of the interface is four times the vertical grid spacing. The source region is centered on the origin and its length ϵ_x is half the wavelength $L/2$ and its thickness ϵ_ϕ is equal to the interface

thickness ϵ. The sponge layer method with length equal to L is applied at the two ends of the computational domain. The initial velocity field is zero, the initial pressure at the air phase is zero, and the initial pressure at the water phase is hydrostatic. The density and dynamic viscosity are respectively set to $1000kg/m^3$ and $1.0 \times 10^{-3}Pas$, for water and $1.2kg/m^3$ and $1.8 \times 10^{-5}Pas$ for air. The free-slip boundary condition is considered at all the four boundaries of the domain.

The free surface elevation for wave case 2 under grid 2 is presented at different instances in time in Fig. 4.1, which shows the formation of the wave fields propagating symmetrically with respect to the origin $X = 0$. As it is observed in the figure the resulting surface elevations are nearly identical to the theoretical solution, except in the source region and in the wave front region where the simulated results are not expected to follow the analytical free surface pattern. For the same wave case (case 2) computed on grid 2, Fig. 4.2 presents the velocity profiles at several streamwise locations confirming the accuracy of the free surface forcing method to generate monochromatic waves. The analytical velocity profiles that have been used in the figure for comparison are the following

$$u_{lin}(x,z,t) = A\omega \frac{\cosh(k_x h + k_x z)}{\sinh(k_x h)} \cos(k_x x - \omega t), \tag{4.1}$$

$$v_{lin}(x,z,t) = A\omega \frac{\sinh(k_x h + k_x z)}{\sinh(k_x h)} \sin(k_x x - \omega t). \tag{4.2}$$

To investigate the grid sensitivity effects in the application of the forcing method, we present in figure (4.3) the free surface elevation of wave case 2 using the five aforementioned grids (see table 4.2). In particular, grid 1, grid 2, and grid 3 are successively refined only in the streamwise direction, while grid 5, grid 4, and grid 2 are refined only in the vertical direction. The objective of refining separately along the streamwise direction and along the vertical directions is to better analyze: (1) the number of grid points required along a wavelength and (2) the vertical grid spacing requirement to properly resolve the diffused interface. The same time step of $0.002s$ and interface thickness of $\epsilon = 4\Delta Y_{min}$ has been used for all grids. As one would expect, the surface elevation converges monotonically to the analytical solution, either when refining in the streamwise direction (figure 4.3) or when refining in the vertical direction (4.3b). With reference

Table 4.3: Description of the parameters used in the directional wave cases.

Wave case	**Direction [deg]**	k_x **[rad/m]**	k_y **[rad/m]**	ω **[rad/s]**
1	15	5.0576	1.3552	7.1669
2	30	4.5345	2.6180	7.1669

to figure 4.3a, grid 2 and grid 3, with respectively 15 and 30 cells across the wavelength, can accurately capture the wave frequency and wave amplitude. In contrast, grid 1, with only 7.5 cells across the wavelength, slightly over-predicts the frequency and under-predicts the amplitude. A first conclusion that can be extracted from these results is that, in order to obtain accurate wave fields, the mesh needs to be constructed with a number of nodes per wavelength larger than 7.5 and ideally approaching 15. Figure 4.3b shows that the vertical grid spacing at the vicinity of the interface also has an important role for obtaining accurate results. While grid 2 and grid 4, with a near interface vertical spacing of $0.005m$ and $0.01m$, respectively, accurately predict the free surface, grid 5, with spacing of $0.02m$, fails considerably.

To analyze the performance of the wave forcing method for different wavelengths while maintaining a fixed wave slope, we compare in figure 4.4 the computed surface elevation resulting from case 1, case 2, and case 3, employing grid 2 in all the three cases. As shown in the figure the accuracy of the proposed approach is insensitive to the wavelength as the free surface agrees very well for the three cases.

Finally, we present in figure 4.5 the surface elevation of case 4, case 5, and case 6, all computed on grid 2, which analyses the ability of the proposed wave maker to generate wave of different slope. In these three cases, the wavelength has been kept constant to $L = 1.2m$, while the wave amplitude has been varied such that the wave slope considered were $Ak = 0.01$ for case 4, $Ak = 0.05$ for case 5, and $Ak = 0.10$ for case 6. As it is shown in the figure the computed waves compare very well with the linear theoretical solution. Note however that for the high slope case (case 6) the wave troughs are slightly raised compared to the linear solution. This may be due to the fact that the minor non-linear effects start taking part in the simulation.

4.2 Forcing method validation case: directional waves

In this case we validate the forcing method for wave generation by simulating, in a 3D basin of constant depth, two wave cases involving a linear directional wave field. The wave we consider in case 1 has an amplitude of $A = 0.01m$, a wavelength of $L = 1.2m$, and a direction of $\beta = 15^o$, being the direction β defined as the angle between the wave propagation direction and the X axis. The wave in case 2 has the same amplitude and wavelength as in case 1, but a direction of $\beta = 30^o$.

The domain length for case 1 is $24m$ ($X = [-12m, 12m]$) in the longitudinal direction and $13.91m$ ($Y = [-6.96m, 6.96m]$) in the span-wise direction, and the depth of water and air is $2m$ and $1m$, respectively. For case 2, the only difference in the domain size with respect to case 1 is the span-wise dimension which is taken as $12m$ ($Y = [-6m, 6m]$). That difference in the span-wise length between the two cases is due to the use of periodic boundary conditions, which require sufficient space to accommodate an integer number of wavelengths.

Three non-uniform Cartesian grids with successively refined resolution are defined in order to perform a mesh convergence study. The three meshes follow a common pattern consisting of an inner rectangular region with uniform grid spacing and an outer region within which the mesh is gradually stretched towards the boundaries using a hyperbolic function and limiting the stretching ratio to 1.05. The inner region, which contains the source region and part of the propagated waves, is the same for the three meshes and spans: $X = [-6m, 6m]$ in the stream-wise direction, the whole length in the span-wise direction, and $Z = [-0.1m, 0.1m]$ in the vertical direction. The grid spacing in the inner region for grid 1, which is the finest of the three meshes, is $\Delta_x = \Delta_y = 0.06m$ and $\Delta_z = 0.005m$, for grid 2 is $\Delta_x = \Delta_y = 0.1m$ and $\Delta_z = 0.01m$ and for grid 3 is $\Delta_x = \Delta_y = 0.2m$ and $\Delta_z = 0.02m$. The total number of nodes per each mesh is: $301 \times 201(233) \times 199$, $201 \times 121(141) \times 139$, $109 \times 61 \times 92$, for grids 1 to 3, respectively. The number that is in parenthesis in the span-wise direction refers to the number of grid nodes for case 1, which slightly differs from the value in case 2 as mentioned before.

The source region is centered on $x = 0$ with horizontal thickness of $\epsilon_x = L/2 = 0.6m$. Its vertical thickness is equal to the thickness of the smoothed free surface interface $\epsilon_\phi = \epsilon$, which is taken as four times the vertical grid spacing. That is $\epsilon = 0.02m$ for

grid 1, $\epsilon = 0.04m$ for grid 2, $\epsilon = 0.08m$ for grid 3. The time step used is $0.001s$ and the gravitational acceleration is $g = 9.81m/s^2$. While the sponge layer method with length equal to $1.2m$ is applied at the stream-wise boundaries, periodic boundary conditions is applied in the span-wise direction. The free-slip boundary condition is considered at the top and bottom wall as well as in the stream-wise boundaries. The simulation is started with an undisturbed free surface and a zero velocity field, pressure field of zero at the air phase, and the hydrostatic pressure at the water phase. The density and dynamic viscosity are the same as in the previous wave cases, $1000kg/m^3$ and $1.0 \times 10^{-3}Pas$, respectively, for water and $1.2kg/m^3$ and $1.8 \times 10^{-5}Pas$ for air.

The results of the free surface elevation with the grid sensitivity study and the comparison with the analytical solution is presented in figure 4.6 for case 1 and in figure 4.7 for case 2. The two figures show that the proposed wave generation method can successfully simulate 3D directional waves of different propagation angle β with results converging monotonically to the expected theoretical solution as the grid resolution is refined. Analyzing the effect of the grid resolution on the accuracy of the computed waves, we observe that while grids 1 and 2 can represent the wave with an accurate amplitude and frequency, specifically grid 1 provides excellent accuracy, grid 3 fails to provide reasonable results. To see the number of grid nodes per wavelength in a 3D directional wave case, we first need to compute the projected wavelength to the X and Y axis, which is done with the following expressions: $L_x = 2 * \pi/k_x$ and $L_y = 2 * \pi/k_y$. In case 1 $L_x = 1.25m$ and $L_y = 4.6m$ which respectively corresponds to 25 and 92 grid nodes for grid 1 and 12.4 and 46 grid nodes for grid 2. In case 2 $L_x = 1.39m$ and $L_y = 2.4m$ and the number of grid nodes per wavelength is respectively 28 and 48 for grid 1, 14 and 24 for grid 2, and 7 and 12 for grid 3. Looking at the X direction values, which are the critical ones, the minimum number of grid nodes per wavelength to get accurate free surface results is approximately 12, corresponding to grid 2. The 7 nodes used for case 2, grid 3 are clearly insufficient. This grid resolution requirements are in line with those discussed in the previous section for monochromatic waves.

Table 4.4: Description of the three wave frequencies in the Simulation of 2D superposed monochromatic waves case

Wave Component	L **[m]**	k_x **[rad/m]**	A **[m]**	Ω **[rad/s]**
1	12.57	0.5	0.040	2.24
2	6.28	1.0	0.010	3.16
3	3.14	2.0	0.005	4.47

4.3 Far-field/near-field coupling validation cases

In this section we present three wave cases aimed to validate the far-field/near-field wave coupling algorithm described in section 2.3. We also want to demonstrate the ability of the forcing method to generate complex wave fields composed of various superposed frequencies.

In all of the three cases we present, the simulation is started at the HOS domain of the far-field code, by setting the initial velocity potential and free surface elevation to that of the given wave case at time zero. As soon as the far-field simulation is started a Fast Fourier Transform of the free surface elevation is applied at every time step to extract the wave frequencies and amplitudes, which are then incorporated to the near-field solver with the proposed surface forcing method. In the present cases we do not consider any airflow in order to minimize the complexity of the problem.

4.3.1 Simulation of 2D superposed monochromatic waves

In this test case, a set of three superposed monochromatic waves are simulated in a 2D rectangular channel. The three wave components have an amplitude of $A_1 = 0.040m$, $A_2 = 0.009m$ and $A_3 = 0.005m$, and wavenumber of $k_{x1} = 0.5$, $k_{x2} = 1.0$ and $k_{x3} = 2.0$ (as summarized in table 4.4).

The computational domain is a 2D rectangular channel of length $120m$ $X = [-30m, 90m]$, the water depth is $10m$, and the air column height is $1m$. A grid of size 300×200 is constructed following the same dual inner/outer region pattern described in section 4.1 for the monochromatic wave case. The region of uniform spacing is defined respectively in the stream-wise and vertical directions as $X = [-7m, 60m]$ and $Z = [-0.1m, 0.1m]$, and the corresponding grid spacing is $0.3m$, and $0.005m$. The time step is $0.00125s$ and the gravity is $g = 10m/s^2$. Free-slip condition is considered at all the four boundaries

Table 4.5: Description of the three wave frequencies in the simulation of 2D superposed monochromatic waves case

Wave comp.	L **[m]**	k_x **[rad/m]**	k_y **[rad/m]**	L_x **[m]**	L_y **[m]**	A **[m]**	ω **[rad/s]**
1	12.57	0.5	0.0	12.57	0	0.040	2.24
2	4.44	1.0	1.0	6.28	6.28	0.014	3.76
3	2.51	2.0	1.5	3.14	4.20	0.008	5.00

in combination with a sponge layer of length $9m$ applied at the side boundaries.

The computed free surface elevation at the near-field solver resulting from the far-field/near-field coupling process is presented in figure 4.8 which is shown to agree very well with the computed free surface at the far-field domain. As already discussed, the solution at the source region $X = [-6.2m, 6.2m]$ does not generally match the expected solution which in that case is the far-field solution.

With the stream-wise spacing of $0.3m$ that we used in the simulation the first wave component ($L = 12.57m$) is resolved with 42 grid cells, the second component ($L = 4.44m$) with 15 grid cells, and the third component ($L = 2.51m$) with 8.4 grid cells. Using the conclusions extracted from the grid sensitivity analysis in section 4.1, we can say that the number of cells per wavelength used for wave components 1 and 2 is sufficient for resolving them with high accuracy. However, for wave component 3 the number of cells per wavelenth of 8.4 is rather low, which may explain the very minor discrepancies that can be seen in figure 4.8.

4.3.2 Simulation of three 3D superposed directional waves

In this test case, a wave field composed of three directional wave components are simulated in a 3D rectangular basin. As already mentioned, the wave field is initialized at the far-field domain and transferred with the pressure forcing method to the near-field solver by following the coupling algorithm described in section 2.4.

The three wave components of the present case have an amplitude of $A_1 = 0.040m$, $A_2 = 0.014m$, and $A_3 = 0.008m$, a wave-number in the x direction of $k_{x1} = 0.5$, $k_{x2} = 1.0$ and $k_{x3} = 2.0$, and a wave-number in the y direction of $k_{y1} = 0.0$, $k_{y2} = 1.0$ and $k_{y3} = 1.5$. A summary of the three wave component parameters including wavelength and wave frequency is presented in table 4.5.

The computational domain is a rectangular basin of length $93.5m$ ($X = [-31.16m, 62.34]$) in the stream-wise direction, 31.16m ($Y = [0m, 31.16m]$) in the span-wise direction, water depth of $10m$, and air column depth of $1m$. The grid is non-uniform with the same structure used in the directional wave cases in section 4.2. The region with uniform spacing is the following: $X = [-7.5m, 30m]$, $Y = [0m, 31.16m]$, and $Z = [-0.2, 0.2]$, and the grid spacing within this area is , respectively, $0.4m$, $0.25m$, and $0.02m$. The time step is $0.0025s$ and the gravity $g = 10m/s^2$.

The results of the surface elevation resulting from both the far-field simulation and the near-field simulation are presented in figure 4.9. As shown in the figure, the computed near-field solution agrees well with the far-field results. In this case, the wave component that is resolved with least number of nodes per wavelength is the third with 12 nodes per L_x and 10.5 nodes per L_y.

4.3.3 Simulation of a broadband wave spectrum

The final case for validating the far-field/near-field coupling algorithm and the ability to simulate a wave field with multiple frequency components is the simulation of a broadband wave spectrum of peak approximately equal to $L \approx 8m$. The wavenumber distribution of the wave field that is extracted from the far-field domain at time $t = 30s$ is shown in figure 4.10.

For this test case the domain size, the mesh dimensions, and all the case parameters such as the fluid properties, the gravity, the boundary conditions, or the time-step size are the same as in the simulation of three 3D superposed directional waves case in the previous subsection 4.3.2.

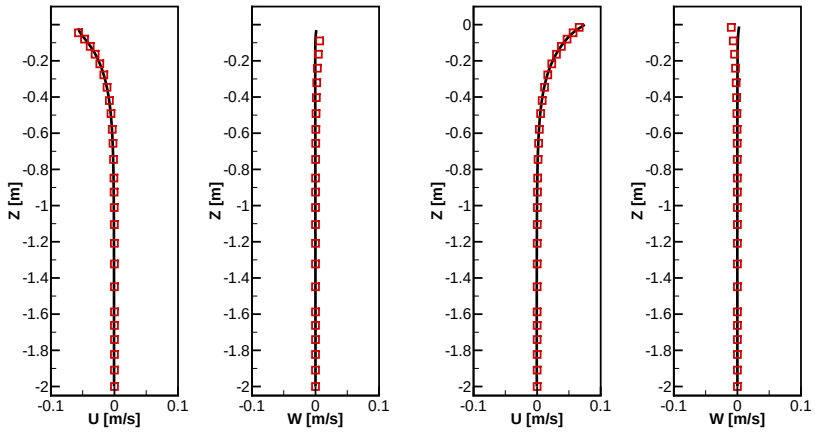
The free surface elevation results computed at the near-field domain and at the far-field domain are presented in figures 4.11 and 4.12. In particular, figure 4.11 shows elevation contours at different times, $t_1 = 26s$ and $t_2 = 30s$, and figure 4.12 several elevation profiles at different Y planes at time $t_2 = 30s$. As demonstrated in the two figures, the resulting wave field in the near-field solver, which is generated in the pressure forcing method, agree well with that simulated in the far-field domain. Obviously, there are some discrepancies which are explained due to the fact that not all wave frequency components can be resolved with the optimum number of grid nodes per wavelength. These discrepancies can be minimized with increased grid resolution.

4.4 Approach 2 Validation case: generation of a broadband wave spectrum

This test case is aimed to validate the far-field/near-field coupling algorithm in approach 2. In approach 2 the wave and wind conditions are run at the LES-HOS code until fully developed conditions are achieved. Then the wind and wave data is transferred to the near-field code as initial condition. In this test case we validate the pressure forcing method used to initialize, in the near-field solver, the wave field given from the precursor simulation. We use the same far-field precursor simulation data as that from the floating platform case in section 7 which consists in the broadband wave spectrum shown in figure 7.4.

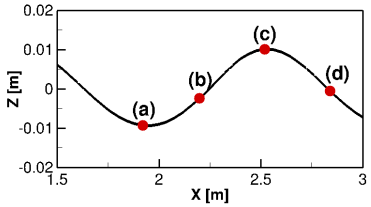
The computational domain is a $6283m$ long and $3141m$ wide basin with $280m$ of water depth and $500m$ of air column. The mesh is uniform in the horizontal directions with 211 and 106 grid points in the stream-wise and span-wise directions, respectively. In the vertical direction the grid is uniform from $Z = -12m$ to $Z = 12m$ with a spacing $\Delta z = 2m$, and then the spacing progressively increases towards the top and bottom walls. The fluid properties and the gravity are the same as in the turbine case. Slip-wall boundary conditions are considered at the top and bottom boundaries, and periodic boundary conditions at the lateral boundaries. The time step is set to $0.02s$ and the interface thickness ϵ to $4m$. The time smoothing parameter $Delta$ has been chosen to be fixed for all wave frequencies an equal to $1s$.

Figures 4.13 and 4.14 show the computed free surface elevation right after initialization at time $t = 7s$ and compares it with the expected solution taken from the far-field domain. The excellent agreement seen in the figure demonstrate the ability of the pressure forcing method to initialize a complex wave fields in a diffused based Navier-Stokes two-phase flow solver such the presented Level set-CURVIB method.



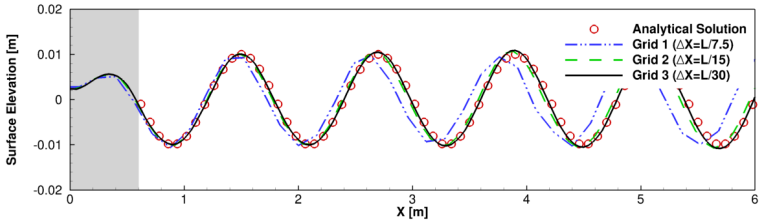
(a) $X = 1.92m$ (b) $X = 2.20m$

(c) $X = 2.52m$ (d) $X = 2.84m$

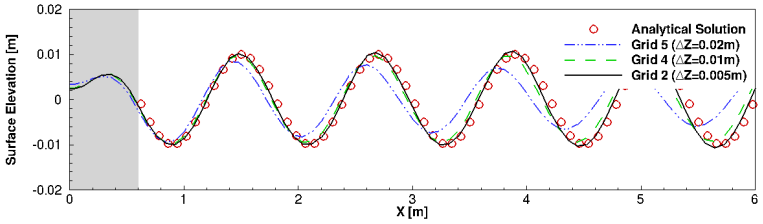


(e) Position of the wave profiles

Figure 4.2: Generation of monochromatic waves. Vertical profiles of horizontal velocity u and vertical velocity v at different streamwise locations for wave case 2 ($L = 1.2m$). Continous lines represent the present computed solution and circles the analytical solition from linear wave theory. The profiles correspond to time $t = 14s$ and the time step used is $0.002s$.

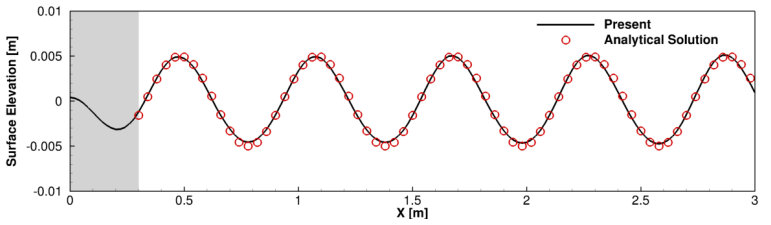


(a) Grid refinement in the stream-wise direction

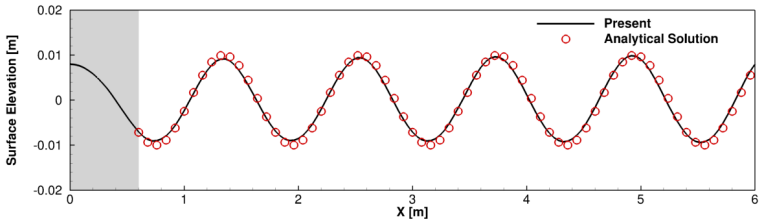


(b) Grid refinement in the vertical direction

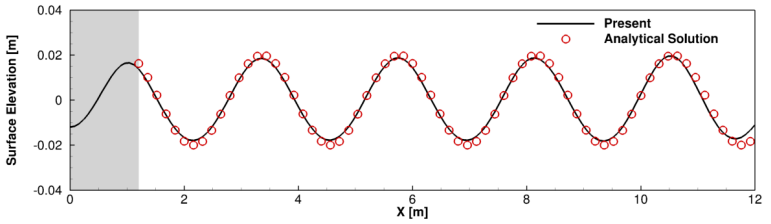
Figure 4.3: Generation of monochromatic waves. Free surface elevation for wave case 2 using grid 2 at three instances in time. Continous lines represent the computed solution and circles the analytical solution from linear wave theory. The results correspond to time $t = 16s$ and the time step used is $0.002s$. The grey shaded area represents the source region where the solution cannot be representat by the anaylitical solution.



(a) Wave case 1: $L = 0.6m$

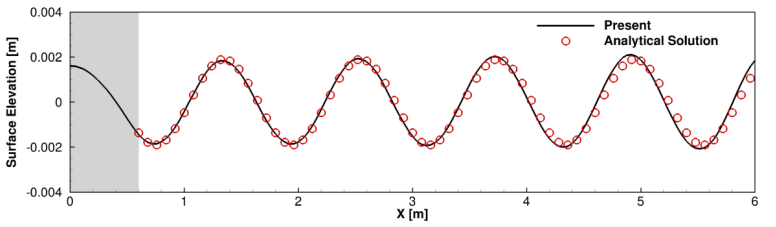


(b) Wave case 2: $L = 1.2m$

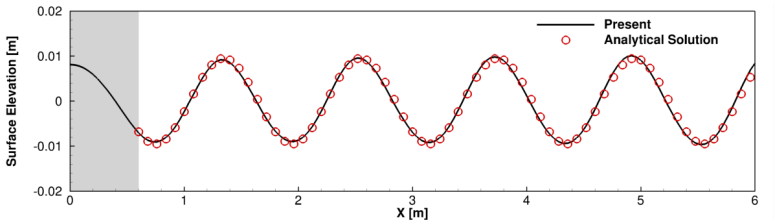


(c) Wave case 3: $L = 2.4m$

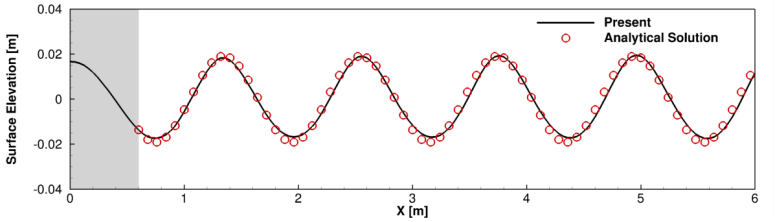
Figure 4.4: Generation of monochromatic waves. Computed and analytical free surface elevation of several monochromatic wave cases with different wavenumbers but maintaining a fixed wave slope ($Ak = 0.01$). The results correspond to time $t = 14s$ and the time step used is $0.002s$.



(a) Wave case 4: $Ak = 0.01m$

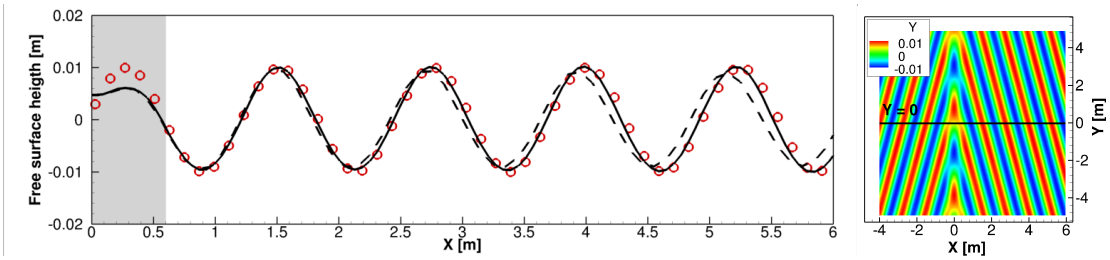


(b) Wave case 5: $Ak = 0.05m$

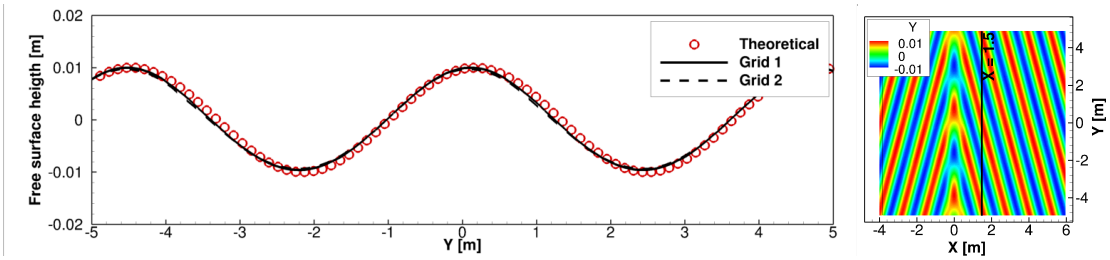


(c) Wave case 6: $Ak = 0.10m$

Figure 4.5: Generation of monochromatic waves. Computed and analytical free surface elevation of several monochromatic wave cases with fixed wavelength ($L = 1.2m$) but different slope. The results correspond to time $t = 14s$ and the time step used is $0.002s$.

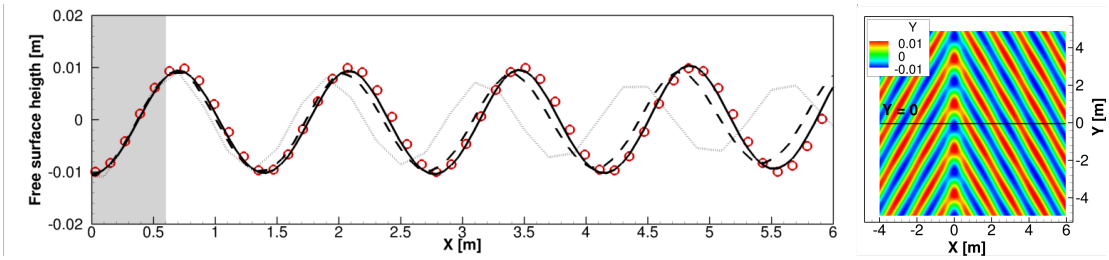


(a) Free surface profile at plane $Y = 0m$

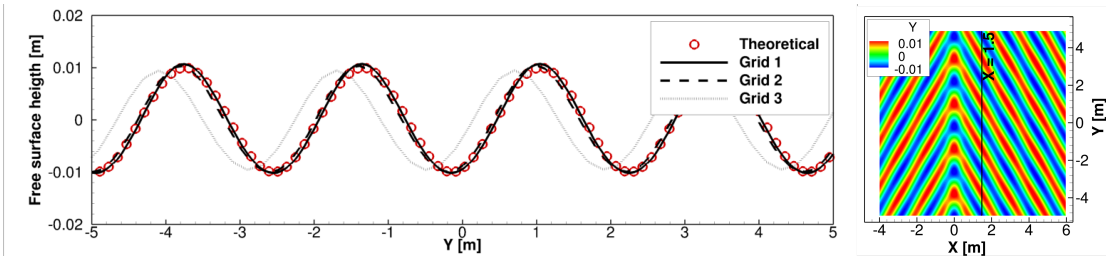


(b) Free surface profile at plane $X = 1.5m$

Figure 4.6: Generation of 3D directional waves. The left figure shows the free surface elevation profiles computed on grid 1 and grid 2 and the theoretical solution from linear wave theory. The right figure show the surface elevation contours (in meters) with a horizontal line in (a) and a vertical line in (b) to indicate the position of the plane shown in the left figure. The results correspond to time $t = 14s$ and the time step used is the simulation is $0.001s$. The grey shaded area represents the source region where the computed solution cannot always be represented by the analytical solution.



(a) Free surface profile at plane $Y = 0m$



(b) Free surface profile at plane $X = 1.5m$

Figure 4.7: Generation of 3D directional waves. The left figure shows the free surface elevation profiles computed on grid 1, grid 2, and grid 3 and the theoretical solution from linear wave theory. The right figure show the surface elevation contours (in meters) with a horizontal line in (a) and a vertical line in (b) to indicate the position of the plane shown in the left figure. The results correspond to time $t = 16s$ and the time step used is the simulation is $0.001s$. The grey shaded area represents the source region where the computed solution cannot always be represented by the analytical solution.

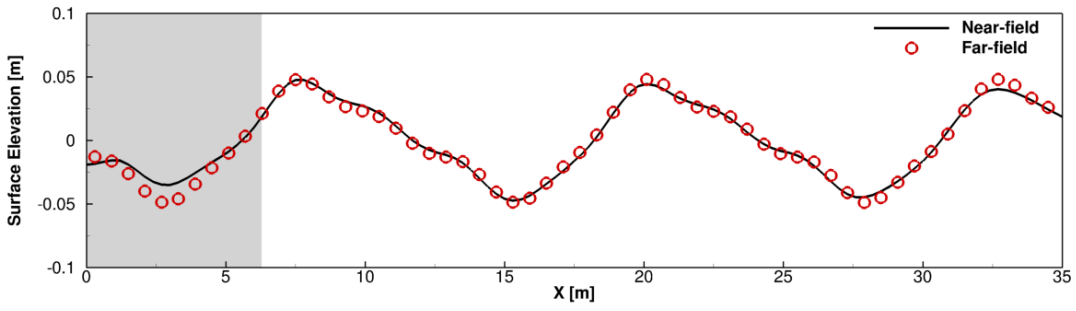
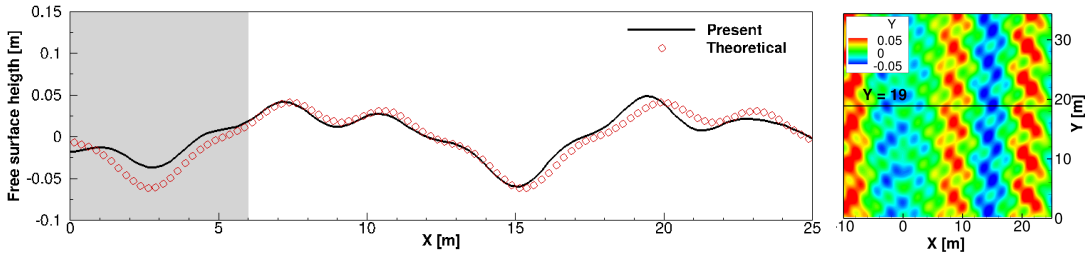
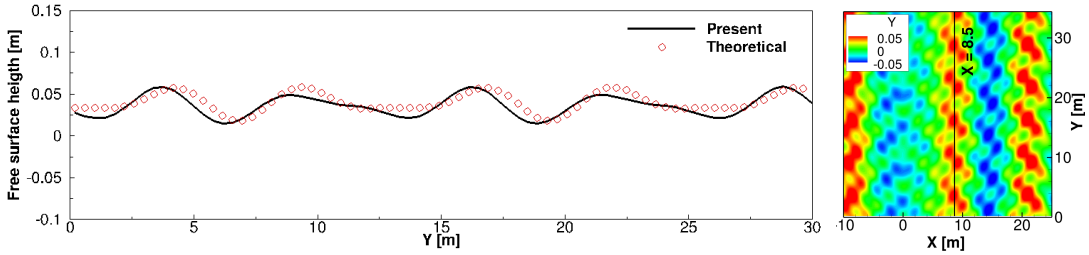


Figure 4.8: Far-field/Near-field coupling case 2: simulation of superposed monochromatic waves. Computed and analytical free surface elevation of a set of 3 superposed waves that have been incorporated from the far-field to the near-field solvers.



(a) Section $Y = 2$



(b) Section $X = 8$

Figure 4.9: Far-field/Near-field coupling case 2: simulation of three superposed directional waves. The left figure shows computed free surface elevation profiles from both the far-field and the near-field domains. The right figure show the surface elevation contours (in meters) with a horizontal line in (a) and a vertical line in (b) to indicate the position of the plane shown in the left figure. The results correspond to time $t = 30s$ and the time step used is the simulation is $0.0025s$. The grey shaded area represents the source region where the computed solution cannot always be represented by the analytical solution.

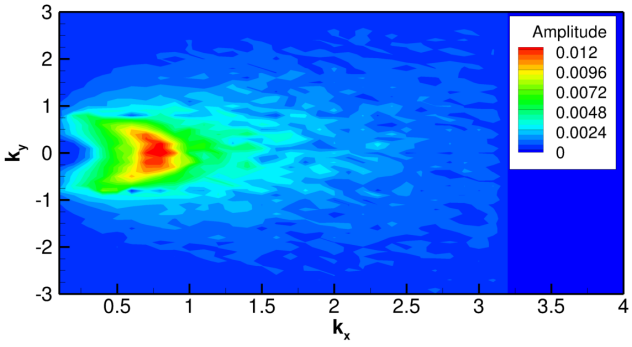
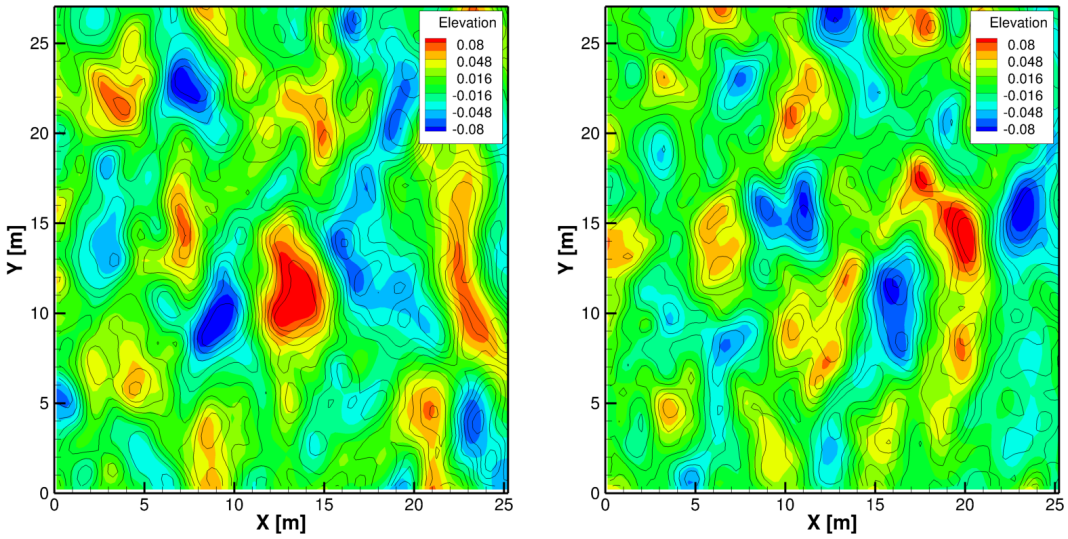


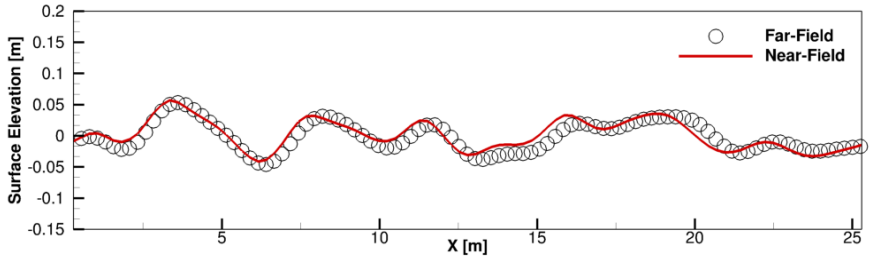
Figure 4.10: Far-field/Near-field coupling case 3: simulation of a broadband wave spectrum. Definition of the wave field with contours of amplitude as a function of the wavenumber computed at the far-field domain at time $t = 30s$.



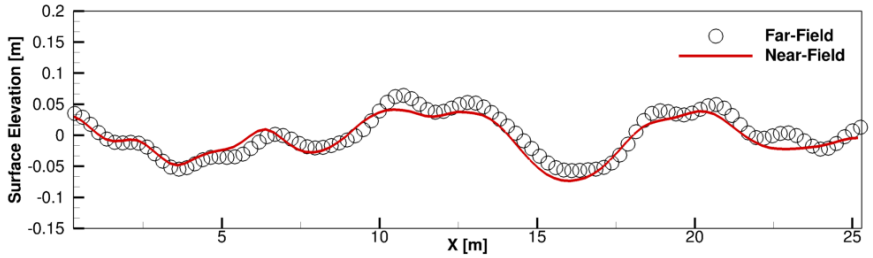
(a) Time $t = 26s$

(b) Time $t = 30s$

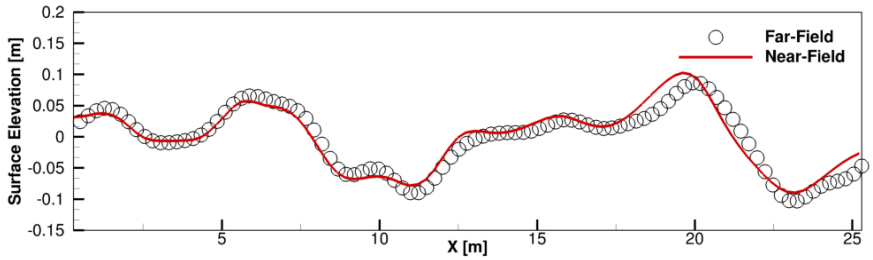
Figure 4.11: Far-field/Near-field coupling case 3: simulation of a broadband wave spectrum. Computed free surface elevation in meters at the far-field domain (contour lines) and at the near-field domain (colored contours).



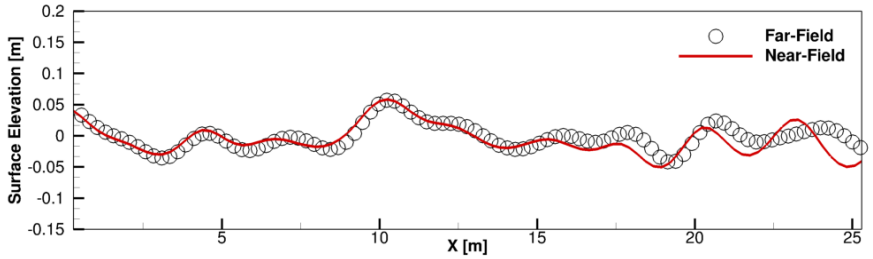
(a) $Y = 5.0m$



(b) $Y = 10.0m$



(c) $Y = 15.0m$



(d) $Y = 20.0m$

Figure 4.12: Far-field/Near-field coupling case 3: simulation of a broadband wave spectrum. Computed free-surface elevation profiles from both the far-field and the near-field domains at different Y planes. The results correspond to time $t = 30s$ and the time step used is the simulation is $0.0025s$.

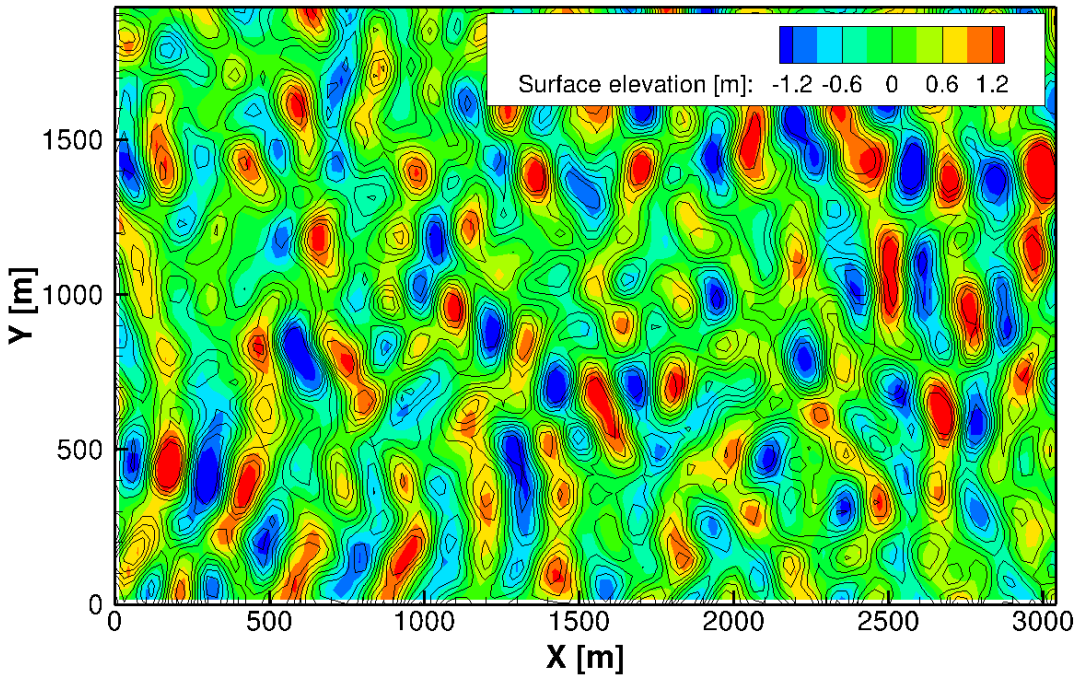
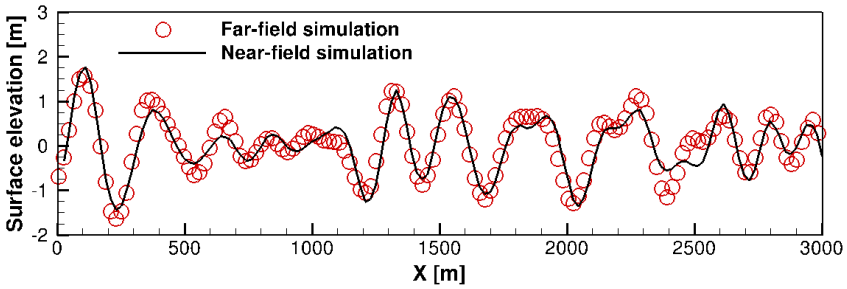
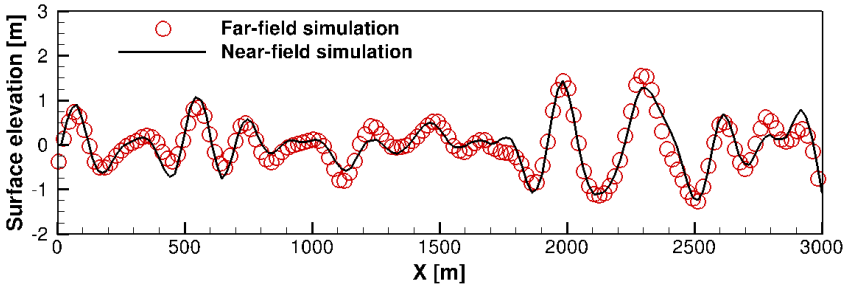


Figure 4.13: Approach 2 validation case. Free surface elevation from the HOS far-field domain (contour lines) and at the level set near-field domain (colored contours). Results at time $t = 7s$ with a time step size of $0.02s$.



(a) Time $Y = 500m$



(b) Time $Y = 1500m$

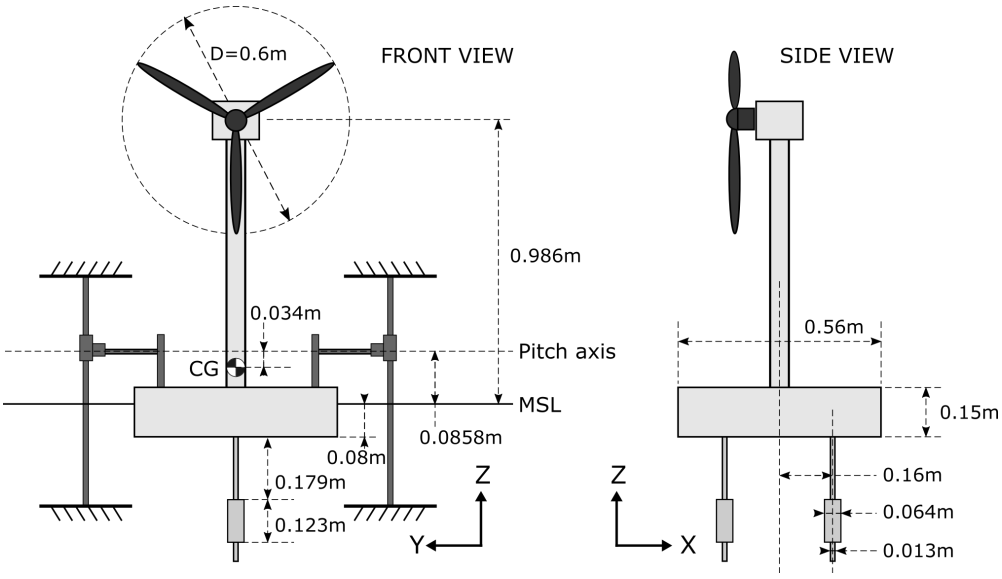
Figure 4.14: Approach 2 validation case. Free surface elevation at two vertical planes. The results correspond to time $t = 7s$ and have been computed with a time step size of $0.02s$.

Chapter 5

Application and validation of the framework for simulating wave-body interactions

In this chapter we further validate the near-field FSI computational model for simulating the interaction of two-phase free surface flows with complex floating structures. Specifically, we aim to demonstrate its capabilities to simulate wave-body interactions which has not been tested. The method is thus applied to replicate the experiments of Feist et al. [165] consisting of a scaled-down offshore floating wind turbine model placed on a wave basin and subject to different wave conditions. Another goal of this chapter is to demonstrate the benefits of using a Navier-Stokes based FSI model compared to the more generally used approaches based on the computationally less expensive potential flow theory. To present this comparison of the FSI with lower order models, we also computed the RAO of the experimental turbine model of [165] using the two following packages: Wave Energy Converter Simulator (WEC-Sim) and ANSYS-AQWA. This lower order model simulations were carried out by Kelley Ruehl, a collaborator of the project from Sandia National Laboratories (SNL).

(a) SAFL wave basin with the floating turbine model



(b) Schematic description of the turbine model

Figure 5.1: View of the SAFL wave basin with the offshore floating turbine model of Feist et al. [165] and schematic description of the turbine mounting system and turbine geometry.

Table 5.1: Description of the floating turbine model properties.

Parameter	**Value**
Platform diameter	$0.56m$
Platform height	$0.15m$
Draft	$0.08m$
Hub height with respect to the mean sea level (MSL)	$0.986m$
Mass	$19.704kg$
Rotor diameter	$0.60m$
Center of rotation with respect to the MSL	$0.0858m$
CG with respect to the MSL	$0.0518m$
Center of buoyancy with respect to the MSL	$-0.04m$
Mass moment of inertia with respect to the CG	$3.82kgm^2$
Mass moment of inertia with respect to the pitch rotation axis	$3.81kgm^2$

5.1 Description of the experimental setting

We present in this section a brief summary of the experiments performed by Feist et al. [165], which we use in this work for validating the computational model. Their experimental work is composed of two quasi-coupled phases. In the first phase, an offshore floating turbine model is placed in the SAFL wave basin to study the structural motion when the turbine platform is subject to a wave forcing. In this set of experiments the wind flow is not considered, treating the floating turbine as a generic floating structure. In the second phase, the floating turbine model is placed in the SAFL atmospheric boundary layer wind tunnel with the objective to investigate the effect of the structural motions in the turbine wake. The turbine is installed on an actuator system that allows to prescribe the motion recorded at the first phase of the experiments.

Since the present work focuses on the first phase of the experiments, we summarize the configuration and main parameters for that phase. The SAFL wave basin has a length of $84m$, a width of $2.743m$, and a water depth of $1.37m$. A hinged-paddle type wave maker is used at one end of the basin to generate the linear monochromatic waves, which are then dissipated with an artificial beach at the other end.

The turbine model from first phase is a 1/100 Froude scaled version of the 13.2MW floating offshore wind turbine prototype designed by SNL (see [166] for details of the prototype design). The prototype has a rotor diameter of $200m$ and a hub height, measured from the MSL, of $119.5m$, and is installed on a barge-style platform, consisting

in a vertical circular cylinder of $56m$ diameter, $15m$ height, and $8m$ draft. The experimental model with $\lambda = 1/100$ scale has a $0.56m$ diameter platform of $0.15m$ height, and $0.08m$ draft. The hub height, when the system is at rest, is located $0.986m$ above the MSL.

With the goal to reduce the complexity of the problem and minimize the level of uncertainty, the experimental turbine is installed on a system of radial and linear roller bearings that restrict the motion to two DoF: heave and pitch. Heave refers to translational motions in the vertical direction (z-axis) and pitch refers to rotations about the y-axis. As illustrated in Figure 5.1(b), the pitch axis is located $0.0858m$ above the MSL and the CG of the system at a $0.0518m$ height above the MSL. The mass of the floating system is $19.704kg$, and the mass moment of inertia is $3.81kg \cdot m^2$, (calculated with respect to the pitch axis), or $3.82kg \cdot m^2$, (calculated with respect to the CG). In Figure 5.1, we present an image of the experimental floating turbine model and a schematic description of the key elements of the mounting system and the geometry dimensions. The aforementioned turbine parameters are summarized in Table 5.1.

The parameters characterizing the waves considered in the experiments were defined based on data measurements available from Buoy 46041 of the National Data Buoy Center (NDBC), which is representative of the wave conditions of a typical deployment site in the U.S. Pacific Northwest (PNW) coast. In particular, using a joint probability distribution (JPD) of the monthly average wave period and significant wave height, the most common wave conditions of the site could be characterized. Using Froude scaling, the wave conditions were adapted to the model scale, resulting in a wave period T ranging from $1.0s$ to $1.29s$, and a wave amplitude A between $0.0075m$ and $0.012m$.

The data measured during the experiments was the surface elevation at 4 given points and the motion of the floating structure. The surface elevation was measured using 4 Massa ultrasonic M-300/150 sensors with an accuracy of $0.25mm$. Two of the sensors were located near the generation boundary to capture the period, amplitude, and celerity of the incident wave and the other two were located near the structure to investigate the effects of the floating platform. The turbine motions were captured using two Keyence laser displacement sensors. Assuming rigid body motions and that the structure is restricted to move in two DoF, measuring the displacement at two points of the structure is sufficient to characterize its motion.

Table 5.2: Description of the three grids employed in the free decay tests

Grid	**Grid size**	**Near body spacing [m]**	**Interface thickness ϵ [m]**
G1	$179 \times 103 \times 130$ (2.4M)	$\Delta X = \Delta Z = 0.020$, $\Delta Y = 0.0100$	0.02
G2	$259 \times 169 \times 217$ (9.5M)	$\Delta X = \Delta Z = 0.010$, $\Delta Y = 0.0050$	0.01
G3	$369 \times 259 \times 350$ (33.4M)	$\Delta X = \Delta Z = 0.005$, $\Delta Y = 0.0025$	0.005

5.2 Simulation results

In this section, we present the simulation results of the floating turbine experiments of [165], described in the previous section. First of all, we perform a series of tests of the floating turbine without waves to characterize the dynamic parameters of the system in order to determine the natural frequencies and the damping ratios in the heave and in the pitch motions. This can be achieved by simulating free decay tests of the platform, first in a single DoF (in heave and in pitch), and then in the two DoF (combining heave and pitch). Then, we simulate wave-turbine interactions by performing a RAO with specific effort on the frequencies close to the fundamental oscillatory modes.

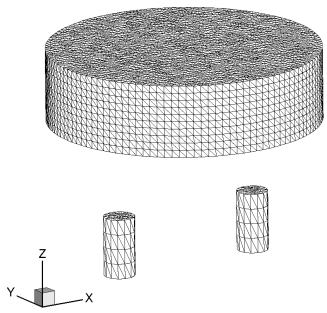


Figure 5.2: Unstructured triangular mesh used to discretize the floating structure.

5.2.1 Turbine dynamics without waves: free decay tests

For all the free decay tests, we consider, as computational domain, a $10m$ long and $2.743m$ wide rectangular basin with a water depth of $1.37m$ and an air column of $0.5m$.

Table 5.3: Definition of the free decay test cases

Case	**DoF**	$h_0[m]$	$\theta_0[deg.]$
C1H	Heave	0.017	0.0
C2H	Heave	0.010	0.0
C3P	Pitch	0.0	5.5
C4P	Pitch	0.0	4.2
C5P	Pitch	0.0	3.2
C6HP	Heave-pitch	0.026	4.94

Since we apply the sponge layer method of thickness $\epsilon_x = 2m$ at the X_{min} and X_{max} boundaries, preventing wave reflections, we do not need to simulate the complete length ($84m$) of the SAFL wave basin. With the exception of the basin length, the other dimensions are equal to that from the experiments.

To be able to perform grid sensitivity studies, three non-uniform Cartesian grids with a dual inner-outer region structure are defined. The inner region, which encloses the floating structure, has uniform grid spacing. In the outer region, the grids are stretched using a hyperbolic function with a maximum ratio limited to 1.045. The three grids, G1, G2, and G3, of sizes 2.4M nodes ($179 \times 103 \times 130$), 9.5M ($259 \times 169 \times 217$), and 33.4M nodes ($369 \times 259 \times 350$), respectively, are successively refined by dividing the inner region grid cells by 2. The inner region of constant grid spacing is defined by the following intervals: $X = [-0.4, 0.4]m$ in the stream-wise direction, $Y = [1.0716, 1.6716]m$ in the span-wise direction, and $Z = [-0.4, 0.1]m$ in the vertical direction. The spacing within this region is $\Delta X = \Delta Y = 0.02m$ and $\Delta Z = 0.01m$ for grid G1, $\Delta X = \Delta Y = 0.01m$ and $\Delta Z = 0.005m$ for grid G2, and $\Delta X = \Delta Y = 0.005m$ and $\Delta Z = 0.0025m$ for grid G3. The three grids are summarized in Table 5.2.

The unstructured triangular mesh used for tracking the floating structure in the CURVIB method is shown in Figure 5.2. With the assumption that the motion of a floating structure is mostly dominated by the forces induced at the water domain, since there is no air flow, the turbine rotor is not rotating, and the air density is much lower than the water density, we simplified the geometry of the structure by neglecting the turbine tower, the motor, and the rotor. Also, we did not include the two rods supporting the stabilizing masses located under the platform, which have a relative small dimension (diameter of $0.012m$) compared to the masses (diameter of $0.064m$).

Even the finest of the meshes we employ in this work, G3, with $33.4M$ grid nodes, would not be sufficiently fine to resolve the flow around these two rods. Note that all these parts that are not considered in the geometry are still accounted for in the mass and the inertia of the system. The fact of simplifying the geometry may have an small noticeable effect when the turbine is in motion. Neglecting this elements in the geometry means that the drag force considered in the simulation is somewhat smaller.

For all the simulations in this paper, the gravitational acceleration is taken as $g = 9.81m/s^2$ and the fluid properties adopt the values corresponding to room temperature. The density and dynamic viscosity of water are, respectively, $1000kg/m^3$ and $10^{-3}kg/(ms)$, and those of air are $1.2kg/m^3$ and $1.8 \times 10^{-5}kg/(ms)$.

The time step size for all the decay tests is taken as $\Delta t = 0.005s$, and the interface thickness as $\epsilon = 2\Delta Z_{min}$, which corresponds to $0.02m$ for grid G1, $0.01m$ for grid G2, and $0.005m$ for grid G3.

As already discussed in previous works (see for example [155]), the damping of a floating platform motion is the combined action of the wave making damping, eddy making damping, and the friction damping. We assume that our computational method can, in principle, account for the damping contributions due to the eddy shedding and the wave damping, but, in contrast, it cannot predict the apparatus frictional damping. Particularly, in the present simulations, the code may also neglect a small fraction of the eddy making damping, since we used a simplified version of the floating turbine geometry by not including some of the secondary elements, such as, the two rods supporting the stabilizing masses, the tower, and the motor, rotor, and nacelle.

To incorporate the fraction of the damping that is not intrinsically accounted in the simulation, we consider the additional damping term given in the rigid body equations of motion (EoM) in equations (2.12) and (2.13). Similar to Calderer et al. [167], the coefficients $b_{t,3}$ and $b_{r,2}$ in the additional damping term are tuned in order to add only the part of the damping effect that is unaccounted.

Heave free decay tests

In the heave free decay tests, the floating structure is initially positioned a distance h_0 above the equilibrium state, which after releasing it, starts a vertical oscillatory motion. We define two heave decay cases with different initial displacements of $0.017m$ for case

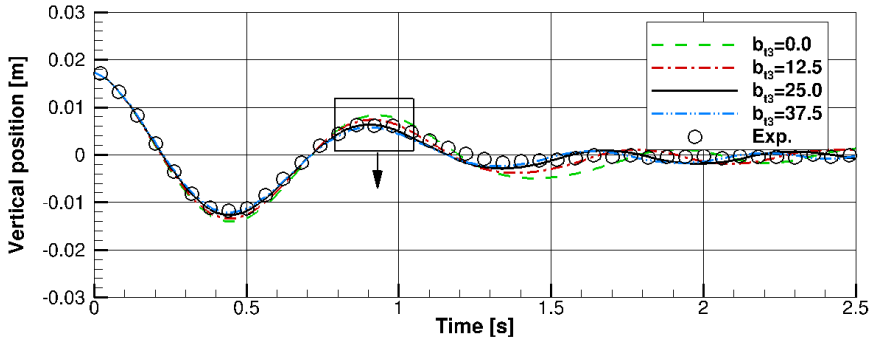


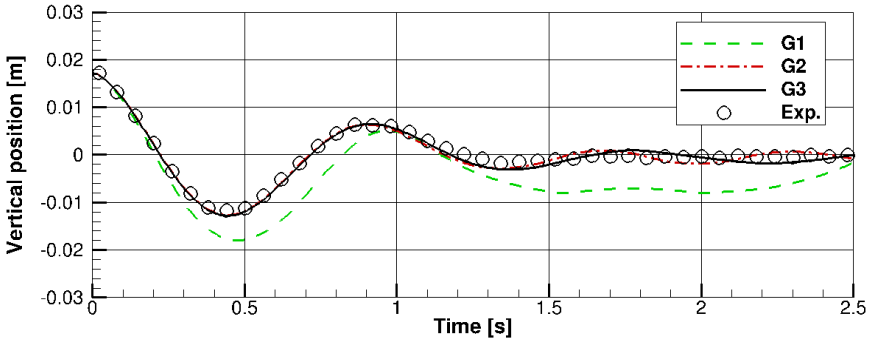
Figure 5.3: Heave free decay test of the floating structure. Vertical position of the structure for case C1H using different values of artifiacial damping, and the experimental data of Feist et al. [165]. The results have been computed on grid G2 using a time step of $0.005s$.

C1H, and $0.010m$ for case C2H (see Table 5.3).

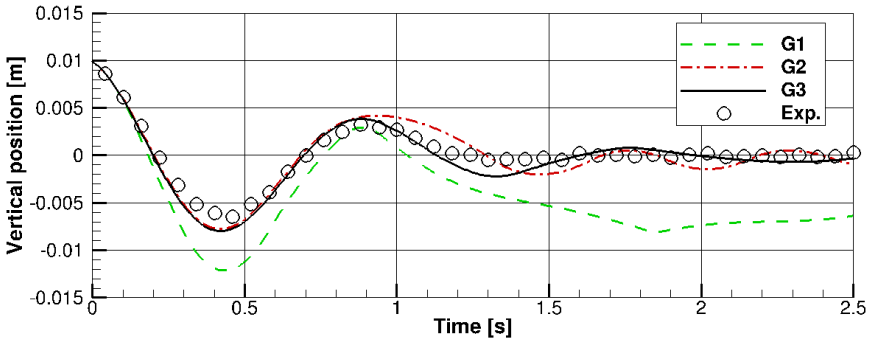
To understand the sensitivity of the coefficient $b_{t,3}$ in the structural response and to obtain the optimum value of additional damping, we simulated test case C1H on grid G2 using several values of $b_{t,3}$ (0.0, 12.5, 25.0, and 37.5). We chose G2 as it is an intermediate resolution grid that should provide sufficiently accurate results for this purpose while still not be computationally too expensive. As shown in Figure 5.3, the computed results agree best with the experimental data when using $b_{t,3} = 25.0$. However, when no additional damping is considered ($b_{t,3} = 0$), the code already captures most part of the overall damping, which means that the frictional damping of the apparatus is small and that the eddy making damping is mostly captured, despite using a simplified geometry.

In Figure 5.4, we present a grid sensitivity study, on grids G1, G2, and G3, of the structural response for cases C1H and C2H using the optimum value of additional damping $b_{t,3} = 25.0$. From the figure, we can first conclude that when the grid is refined the computed results converge to the experimental data. While grid G1 is not sufficiently fine to resolve the platform geometry and is unable to predict the structural motion, G2 and, particularly G3, are able to accurately capture the heave natural period. We also noted that the results agree slightly better in the case with larger motions C1H.

Finally, we show in Figure 5.5(b) and in Figure 5.5(c) the free surface elevation at two points, B_1 and B_2, located a distance D before and after the turbine platform



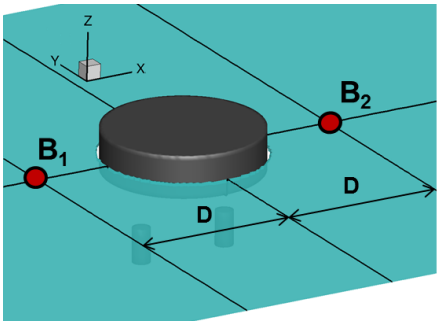
(a) Case C1H: $h_0 = 0.02m$



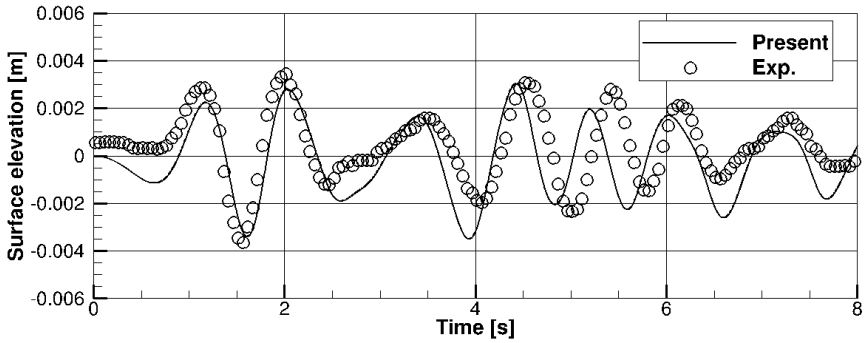
(b) Case C2H: $h_0 = 0.01m$

Figure 5.4: Heave free decay test of the floating structure. Vertical position of the structure computed on grids G1, G2 and G3, and experimental data of Feist et al. [165]. A time step of $0.005s$ and a damping factor of $b_{t,3} = 25$ have been used.

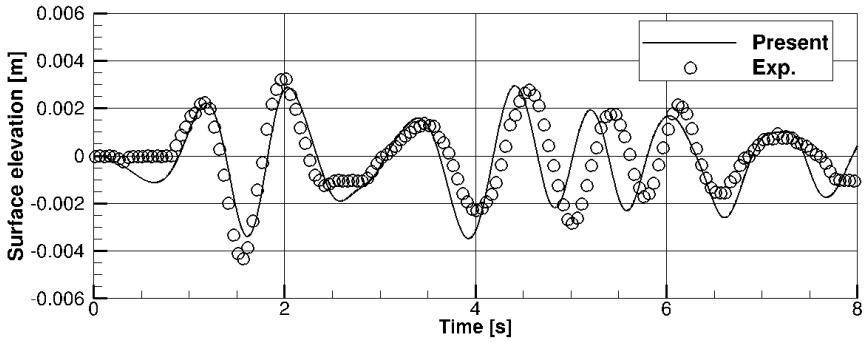
center (see Figure 5.5(b) for an schematic description of the location of the two points). Considering that the free surface motion is in the order of 2 to $4mm$, we can say that the computed results and experimental data agree very well. We noticed that disturbances of around $1mm$ can be observed in the basin even when the water is at the rest condition.



(a) Schemtic description of points B_1 and B_2



(b) Surface elevation at point B_1 ($1D$ before the turbine)



(c) Surface elevation at point B_2 ($1D$ after the turbine)

Figure 5.5: Free surface elevation for the heave free decay test C1H at points B_1 and B_2 located, respectively, at $Z = -1D$ and at $Z = 1D$. The results have been computed on grid G3 and a time step of $0.005s$ and a damping factor of $b_{t,3} = 25$ have been used.

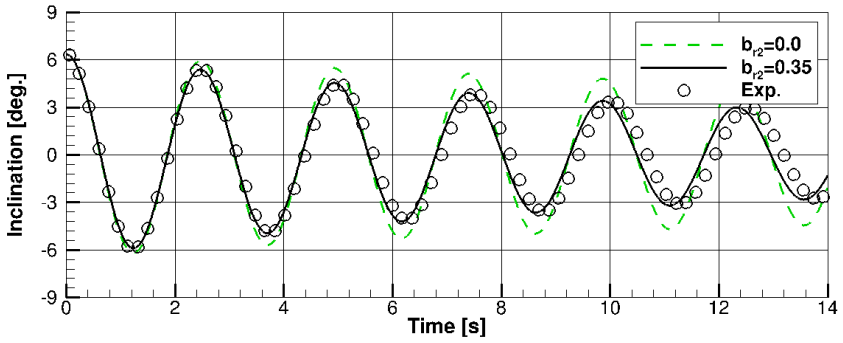


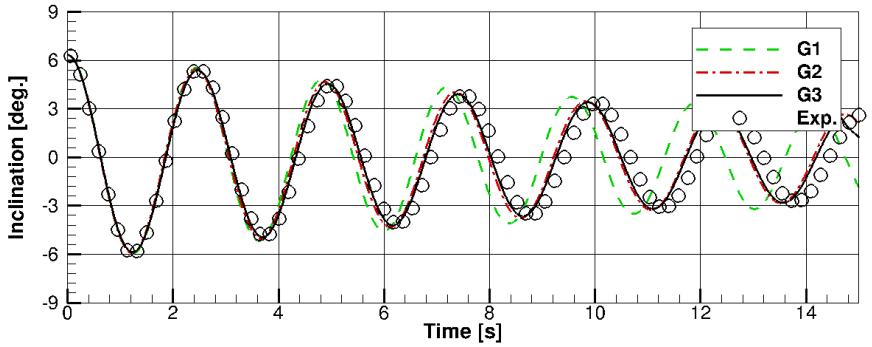
Figure 5.6: Pitch free decay test of the floating structure. Pitch response of the structure for case C3P using different values of artifiacial damping, and the experimental data of Feist et al. [165]. The results have been computed on grid G3 using a time step of $0.005s$.

Pitch free decay tests

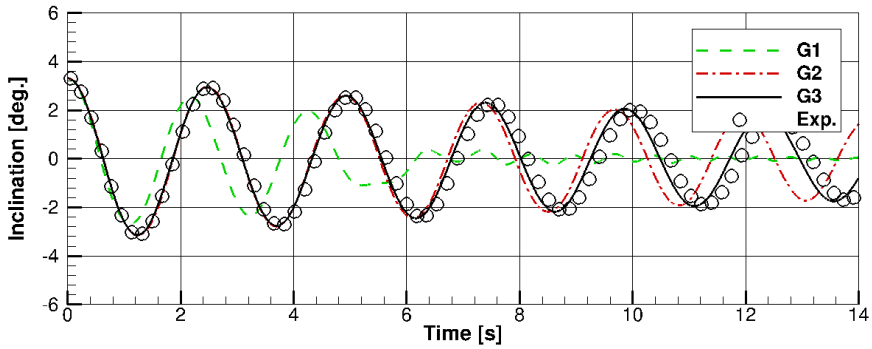
In the pitch free decay tests, the structure is given an initial inclination of angle θ_0 with respect to the pitch rotational axis, which is aligned with the Y axis. We define several cases with different initial angle: case C3P with large initial angle of $5.5deg$, case C4P with medium initial angle of $4.2deg$, and C5P with small initial angle of $3.2deg$, as summarized in Table 5.3.

Similar to the heave decay tests, we first evaluated the optimum value of additional damping. We present in Figure 5.6 the structural response of case C3P on grid G3 using both, no damping, $b_{r,2} = 0.0$, and the optimal value of additional damping which was observed to be $b_{r,2} = 0.35$.

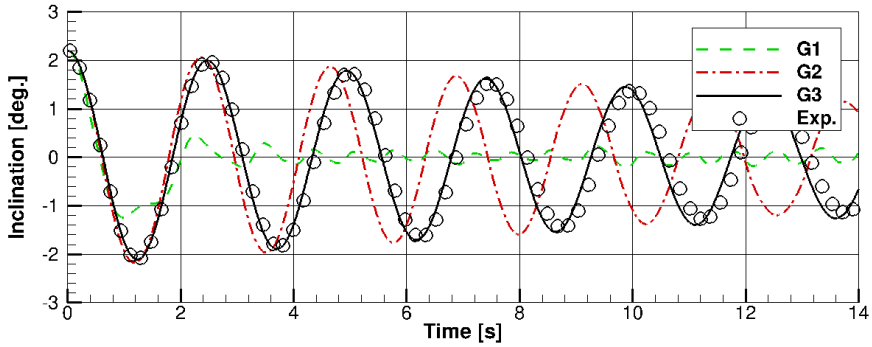
In Figure 5.7, we present the results of a grid sensitivity study for the three pitch decay cases, using grids G1, G2, and G3 and the optimum value of additional damping ($b_{r,2} = 0.35$). For the three cases, the computed solution converges to a solution in which the period is $2.46s$. In contrast, the experiments show a slightly larger natural period of about $2.50s$. This small difference of about 2% can be explained due to the fact of considering a simplified version of the floating structure. Since we are neglecting some geometrical elements, such as the two rods supporting the stabilizing masses as well as the turbine elements above the platform, it means that we are considering a case in which the drag force is slightly less than that from the actual turbine. Note that



(a) Case C3P $\theta_0 = 5.5deg.$



(b) Case C4P $\theta_0 = 4.2deg.$



(c) Case C5P $\theta_0 = 3.2deg.$

Figure 5.7: Pitch free decay test of the floating structure. Pitch response of the structure computed on grids G1, G2 and G3, and experimental data of Feist et al. [165]. A time step of $0.005s$ and a damping factor of $b_{r,2} = 0.35$ have been used.

when the drag force decreases, it also decrease the period.

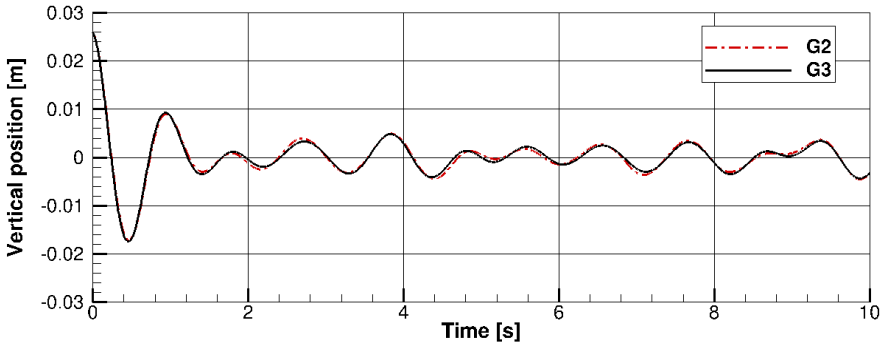
On the other hand, looking at Figure 5.7(a) showing case C3P, all three grids can capture with high accuracy the first $6s$ of the simulations in which the amplitude of the motion is larger than $4deg$. After $t = 6s$, in which the amplitude reduces below $4deg$, grid G1 starts under-predicting the period of the motion. In Figures 5.7(b) and 5.7(c) showing case C4P and C5P, respectively, a similar phenomena occurs with grid G2. When the amplitude motion reduces below $2deg$., after about $9s$ for case C4P and $4s$ for case C5P, grid G2 starts under-predicting the period. From the same figures, grid G1 is shown to become completely insensible to motions under $2deg$. As a result, for each grid resolution there is a certain angle for which smaller amplitude motions are inaccurately captured, and this angle reduces with increased grid resolution. This shows that the code can accurately capture large amplitude motions with relatively coarser grids than that required for small amplitude motions. Also, for capturing the motions of the smallest amplitudes, the resolution of the grids needs to be sufficiently fine.

Combined heave and pitch decay test

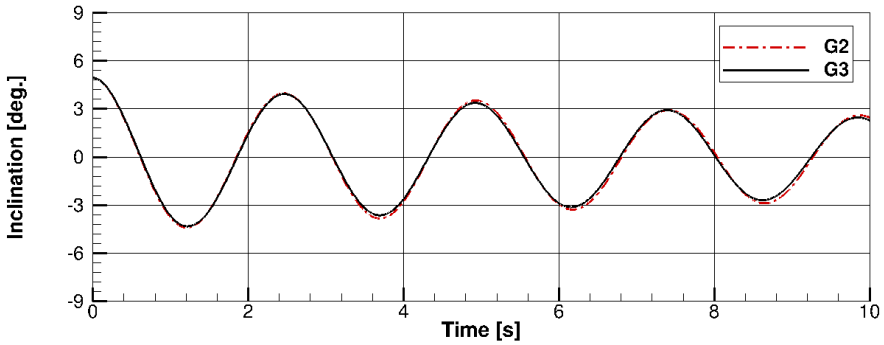
After demonstrating the capability of the computational method to accurately simulate separately heave decay tests and pitch decay tests and obtaining the optimum additional damping for each DoF, we present simulation results of a two DoF decay test combining an initial angle θ_0 with an initial displacement h_0. Although there is no experimental data for this particular test case, we considered that is a relevant case that might be useful for comparing the results of other numerical codes.

In this test case, named C6HP, the initial angle is set to $4.94deg$. and the initial displacement to $0.026m$, as summarized in Table 5.3. The optimum damping factors of $b_{r,2} = 25.0$, for heave, and $b_{r,2} = 0.35$, for pitch, obtained in the single DoF tests are adopted.

Figure 5.8 shows the heave and pitch motion of the structure computed on grids G2 and G3. As seen in the figure, there is minimal difference between the solutions on grids G2 and G3 both in heave and pitch. Looking at the heave motion, in Figure 5.8(a), it is observed that after two seconds of the simulation, the motion does not completely dampen, instead, an oscillation of approximate 2 to $3mm$ remains. The reason of this motion can be explained by looking at Figure 5.10, showing free surface



(a) Heave DoF

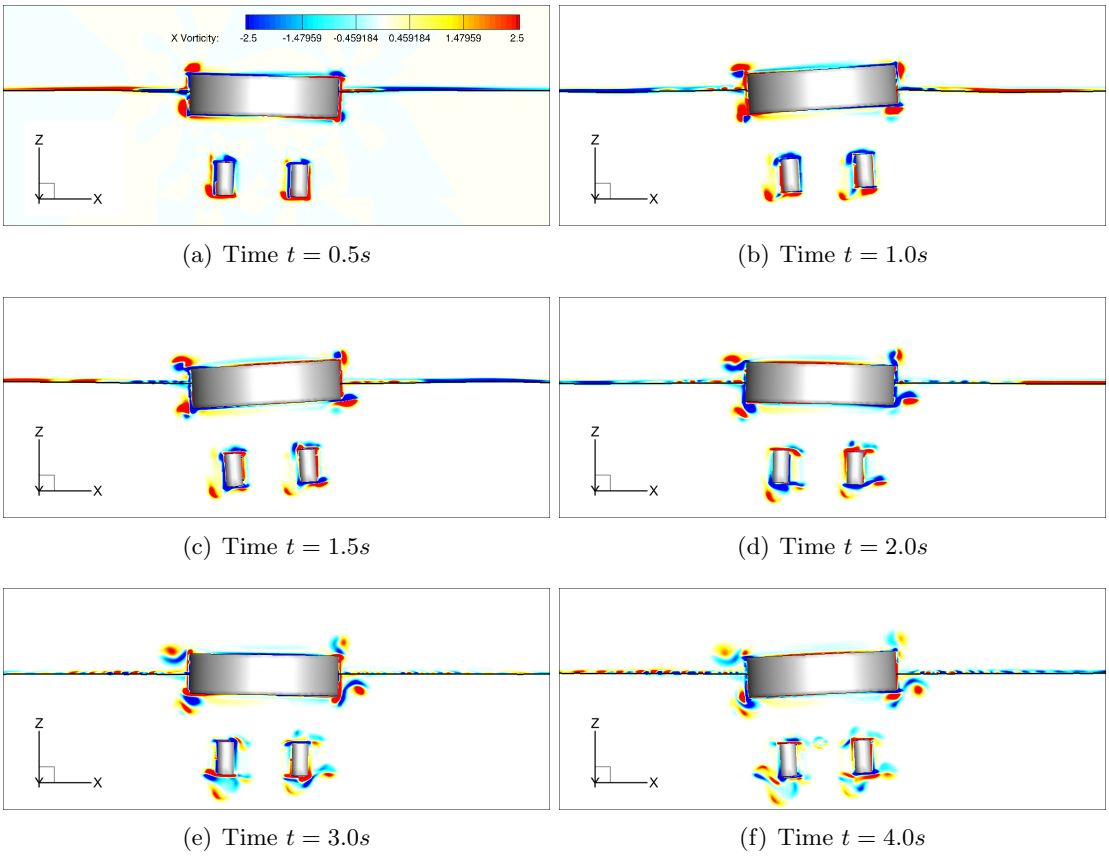


(b) Pitch DoF

Figure 5.8: Combined heave and pitch decay test of the floating structure with $h_0 = 0.026m$ and $\theta_0 = 4.94deg$. Vertical position in (a) and inlcination in (b)of the structure computed on grids G2 and G3. A time step of $0.005s$ and a damping factors of $b_{t,3} = 25$ for the heave and $b_{r,2} = 0.35$ for the pitch have been used.

elevation contours at different time steps. The figure clearly shows how the radiated waves induced by the floating platform reflect at the lateral boundaries Y_{min} and Y_{max}, creating a complex wave field with progressive and standing waves, that eventually interacts with the structure.

In Figure 5.9, we show, at a vertical plane of constant Y, the out of plane vorticity contours at different instances in times. The results in this figure were computed on grid G3. As it is shown from the figure, a series of counter-rotating vortices formed at the structure edges are captured. Also, strong vorticity is observed attached to the free surface propagating away from the structure. In later times, after the system has



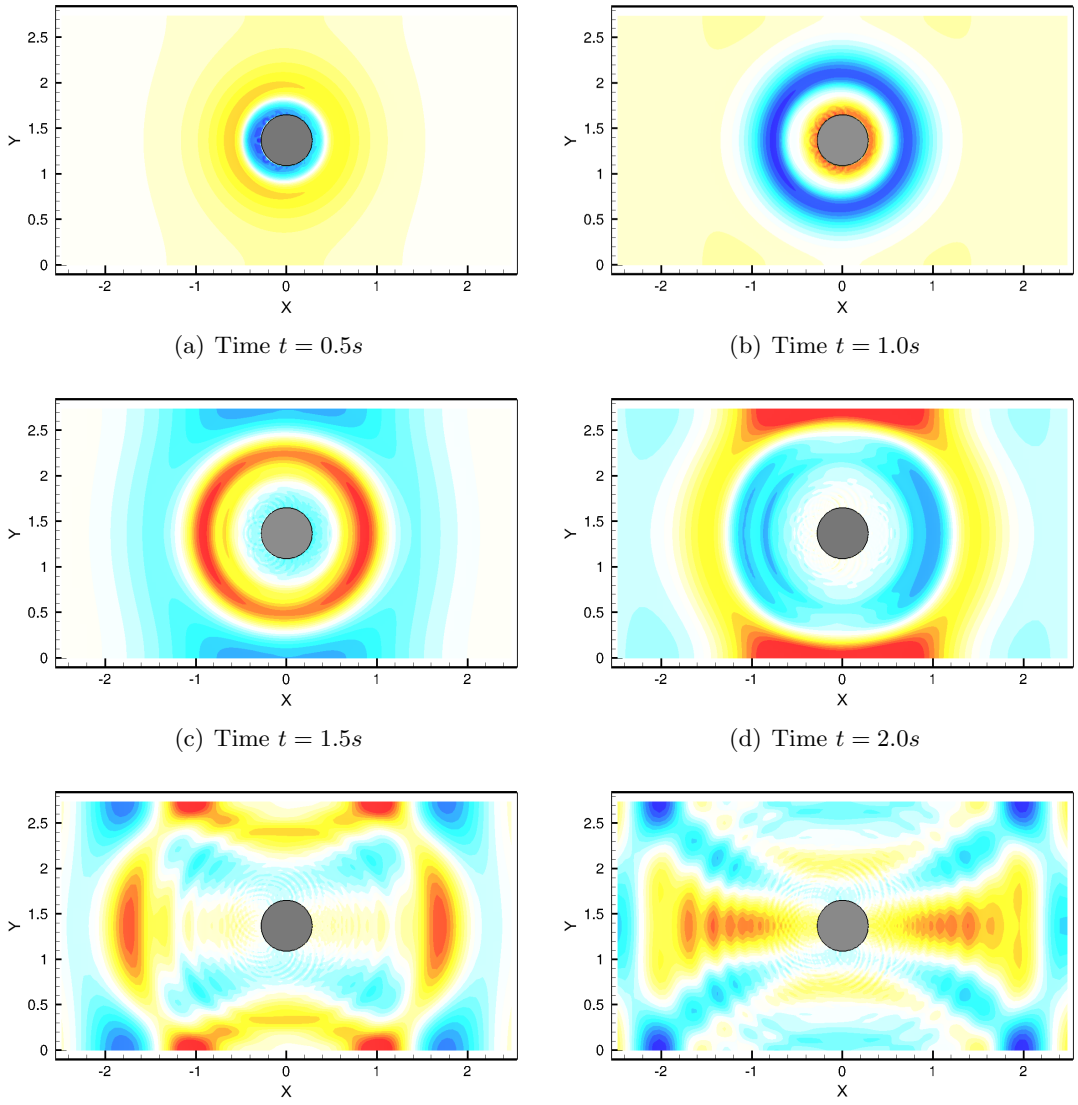
(a) Time $t = 0.5s$ (b) Time $t = 1.0s$

(c) Time $t = 1.5s$ (d) Time $t = 2.0s$

(e) Time $t = 3.0s$ (f) Time $t = 4.0s$

Figure 5.9: Combined heave and pitch decay test of the floating structure with $h_0 = 0.026m$ and $\theta_0 = 4.94deg$. Vertical position in (a) and inlcination in (b)of the structure computed on grids G2 and G3. A time step of $0.005s$ and a damping factors of $b_{t,3} = 25$ for the heave and $b_{r,2} = 0.35$ for the pitch have been used.

undergone through several oscillations, the out of plane vorticity formed at the vicinity of the free surface, is shown to alternate the sign several times. Overall, the figure shows the ability of the methods to capture the flow features formed around the structure.



(a) Time $t = 0.5s$

(b) Time $t = 1.0s$

(c) Time $t = 1.5s$

(d) Time $t = 2.0s$

(e) Time $t = 3.0s$

(f) Time $t = 4.0s$

Figure 5.10: Combined heave and pitch decay test of the floating structure with $h_0 = 0.026m$ and $\theta_0 = 4.94deg$. Vertical position in (a) and inlcination in (b) of the structure computed on grids G2 and G3. A time step of $0.005s$ and a damping factors of $b_{t,3} = 25$ for the heave and $b_{r,2} = 0.35$ for the pitch have been used.

Table 5.4: Description of the three grids employed in the RAO

Grid	**Grid size**	**Near body spacing [m]**	**Source region spacing [m]**	ϵ **[m]**
G1	$214 \times 85 \times 233$	$\Delta X = \Delta Y = 0.020$,	$\Delta X = 0.08$	0.04
	(4.2M)	$\Delta Z = 0.0025$	$\Delta X = 0.08$	0.04
G2	$248 \times 85 \times 233$	$\Delta X = \Delta Y = 0.020$,	$\Delta X = 0.16$	0.04
	(4.9M)	$\Delta Z = 0.0025$	$\Delta X = 0.16$	0.04
G3	$508 \times 169 \times 217$	$\Delta X = \Delta Y = 0.010$,	$\Delta X = 0.15$	0.04
	(18.6M)	$\Delta Z = 0.0050$	$\Delta X = 0.15$	0.04

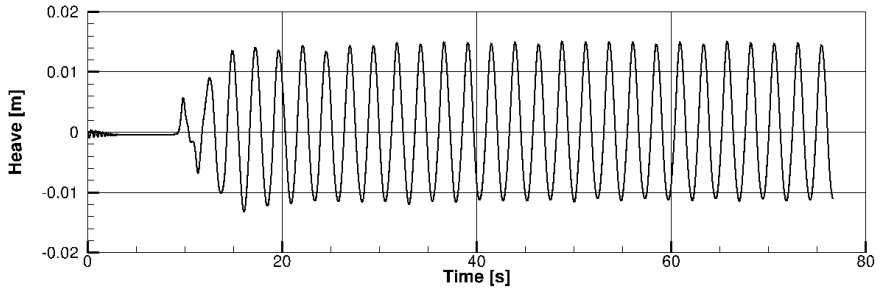
Table 5.5: Description of parameters used in each of the test cases to compute the RAO.

T[s]	**A[m]**	**L[m]**	**Grid used**
0.91	0.020	1.28	G1
0.97	0.012	1.47	G1
0.99	0.015	1.54	G1
1.02	0.013	1.61	G1
1.03	0.013	1.66	G1
1.50	0.009	3.52	G2
1.80	0.010	4.79	G2
2.30	0.014	6.97	G3
2.40	0.0725	7.40	G3
2.425	0.00125	7.50	G3
2.425	0.0017	7.50	G3
2.45	0.0085	7.61	G3
2.45	0.0121	7.61	G3
2.50	0.0086	7.82	G3
2.60	0.013	8.23	G3
2.70	0.0135	8.65	G3

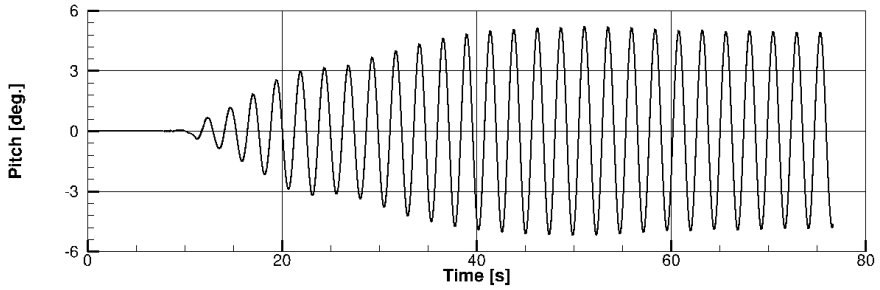
5.2.2 Turbine-wave interactions: response amplitude operator

In this section, we study the platform response when subject to a wave fields of different frequencies by performing a so called response-amplitude operator (RAO). A RAO is a transfer function used to predict the structural response that a given structure will exhibit when subject to different wave conditions.

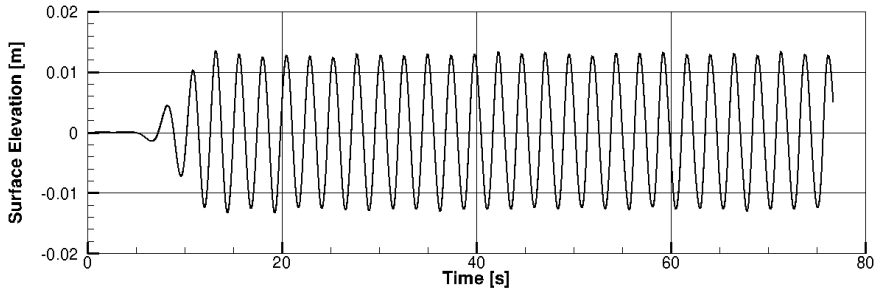
To carry out the RAO using the present computational method we first define a series of test cases characterized by the period T and the amplitude A of the incident wave heading to the floating structure. For each case, the simulation is started with a



(a) Turbine response in heave



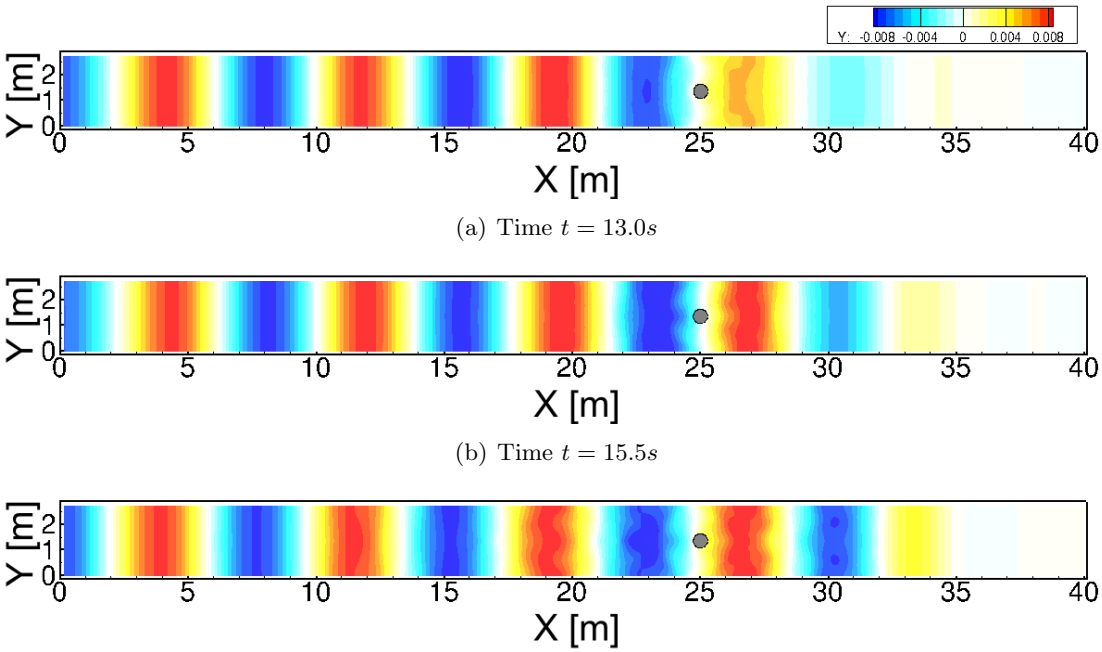
(b) Turbine response in pitch



(c) Surface elevation at $Z = 18m$

Figure 5.11: Wave/body interactions for the case of incident waves of period $T = 2.425s$. Computed structural response of the strucure in heave (a), in pitch (b), and surface elevation at a point located at $Z = 15m$ (c). The results have been computed on grid G2 using a time step of $0.0025s$.

calm free surface and the structure at rest. The wave forcing method, applied at the source region centered on the origin ($X = 0$), progressively begins to generate the wave field, which achieves the desired amplitude approximately after three oscillations. Then, for computing the RAO amplification factors for each of the wave cases, we take the



(a) Time $t = 13.0s$

(b) Time $t = 15.5s$

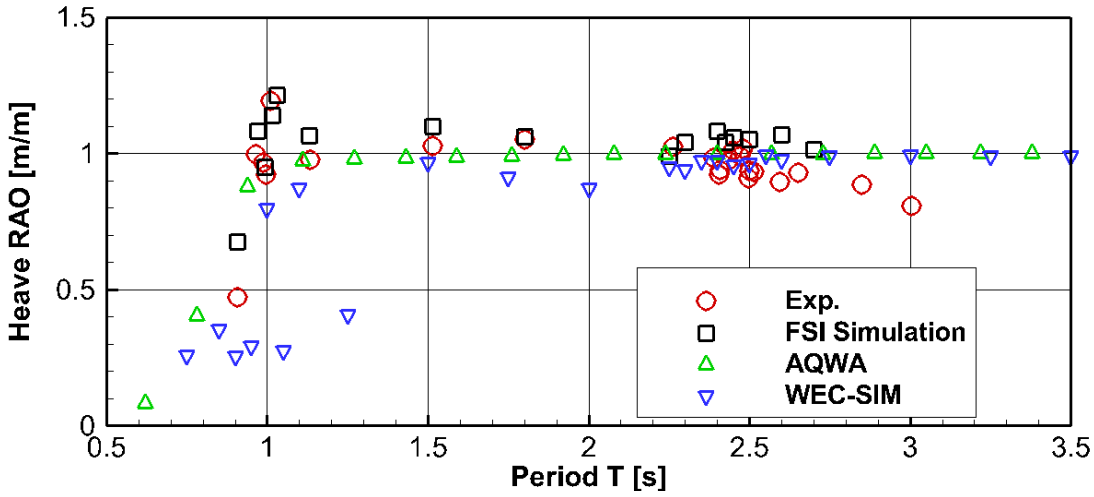
(c) Time $t = 32.5s$

Figure 5.12: Wave/body interactions for the case of incident waves of period $T = 2.425s$. Surface elevation contours at different instances in time. The results have been computed on grid G2 using a time step of $0.0025s$.

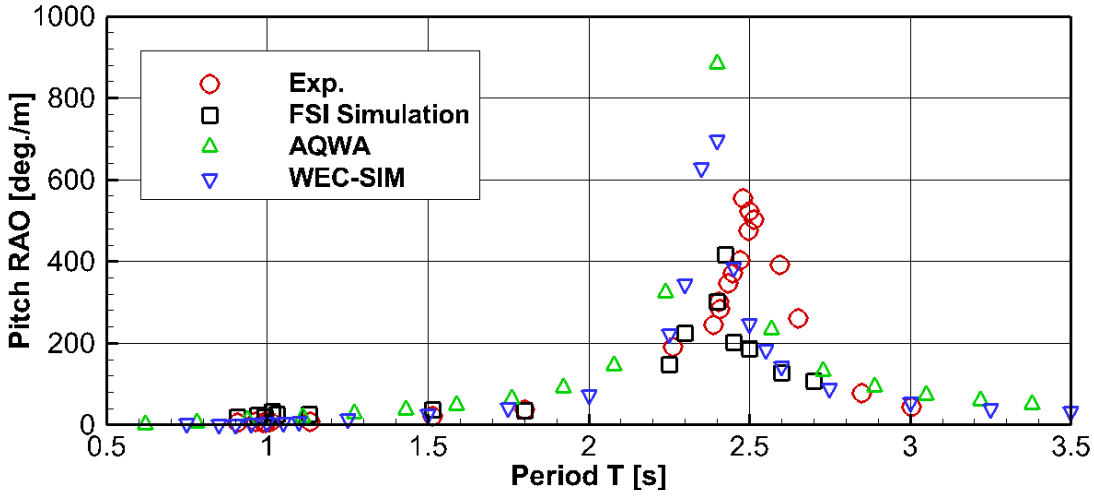
maximum amplitude of the oscillations of the structure, measured from peak to peak, for both, heave and in pitch, after the initial transient effects have been dissipated.

The wave period of the several test cases considered for the RAO ranges from $0.9s$ to $2.7s$ to be able to capture the RAO near the heave and pitch natural periods, which with the decay tests were shown to be $1s$ for heave and $2.45s$ for pitch. The list of computed test cases is given in Table 5.5.

Similar to the heave decay tests, the computational domain is a fraction of the SAFL wave basin. While the basin width of $2.743m$ and water depth of $1.37m$ are taken from the real dimensions, the length does not span the full dimension, which would be very costly. The sponge layer, applied at the beginning and end of the basin, ensures that the waves are not reflected. The basin length is defined based on the requirements of the forcing method and the sponge layer. The source region dimension ϵ_x needs to be equivalent to $L/2$, and the length of the sponge layer x_s between L and $2L$. Also, the



(a) Heave RAO



(b) Pitch RAO

Figure 5.13: Response amplitude operator of the floating structure. Normalized structural response of the strucure in heave (a) and in pitch (b) when subject to incidint monochromatic waves of varying wave period. The simulation results from the present FSI model have been computed on grid G2 using a time step of $0.0025s$.

test section where the structure is positioned needs to be located at least $2L$ from the source region to allow proper development of the wave field. Based on these restrictions, we defined three lengths of the computational domain to accommodate the different RAO cases, which have a wide range of wavelengths (from $1.28m$ to $9.26m$). Using the

dispersion relation, the wavelength L corresponding to the $0.9s$ period wave is $1.28m$, and the wavelength for the $2.7s$ period wave is $9.26m$. The length of the computational domain for the low L cases is $18.6m$ ($X = [-6.0, 12.6]$), for the intermediate L cases is $36.6m$ ($X = [-12.0, 24.6]$), and for the large L cases is $75.0m$ ($X = [-30.0, 45.0]$). We define a different non-uniform grid for each of the three computational domains: G1 of size $214 \times 85 \times 233$ (4.2M) for the short domain, G2 of size $248 \times 85 \times 233$ (4.9M) for the intermediate domain, and G3 of size $478 \times 169 \times 217$ (18.6M) for the long domain. All three grids follow a three-region structure with two inner regions of constant grid spacing and an outer region within witch the spacing progressively increases towards the boundaries. One of the inner regions encloses the wave generation source region and the other region encloses the floating structure. In the source region ΔX is 0.08 for grid G1, 0.16 for grid G2, and 0.15 for grid G3. In the near-body region the grid spacing is $\Delta X = \Delta Y = 0.020$ and $\Delta Z = 0.0025$ for G1, $\Delta X = \Delta Y = 0.020$ and $\Delta Z = 0.0025$ for G2, and $\Delta X = \Delta Y = 0.010$ and $\Delta Z = 0.0050$ for G3.

The dimensions, near body spacing, and interface thickness for the three grids is summarized in Table 5.4. Also, in Table 5.5, we indicate the grid employed for each of the RAO cases.

As we stated in the introduction, in addition to validate the present FSI model for wave-body interactions, we also seek to compare its accuracy to other models, based on lower order assumptions, which typically used by the offshore industry. We thus performed a simulation of the RAO of the floating turbine model using the following two numerical codes: (1) the commercial hydrodynamic software ANSYS-AQWA [168]; and (2) the open source WEC-Sim [169, 170].

ANSYS-AQWA is a potential flow solver based on the boundary element method. As such, it neglects the viscosity of the fluid and assumes linear wave theory and small amplitude motions. Although ANSYS-AQWA has a time domain solver for the structural response, we used the linear frequency domain solver. Frequency domain hydrodynamic codes are an efficient tool to compute the hydrodynamic coefficients required by other time domain solvers such as WEC-Sim.

In ANSYS-AQWA, the structural response is computed linearly by solving the following rigid body EoM in the frequency domain (i=1,2, ... , 6):

$$Y_i(\omega)\left[-\omega^2(m_{ii}+A_{ii}+j\omega(B_{ii}+\Lambda_i)+k_{ii})\right]=F_{e,i}(\omega), \tag{5.1}$$

where ω is the wave angular frequency, m_{ii} the diagonal terms of the mass matrix, A_{ii} is the added mass matrix, B_{ii} the radiation damping coefficients, Λ_i the viscous/additional damping coefficients, and $F_{e,i}(\omega)$ is the wave excitation force.

In the present case, Λ_i is taken from the experimental decay tests of [165]; for heave is $43.766N\cdot s/m$ and for pitch $0.366kg\cdot m^2/s^2$. Note that these values differ from the additional damping incorporated into the Computational Fluid Dynamics (CFD)-FSI method. While the CFD-FSI methods can, in principle, account for the eddy making damping and wave making damping, the potential based flow solvers completely neglects all contributions of the damping, which has to be fully incorporated artificially.

The fact that the center of mass of the floating turbine is below the pitch center of rotations is not straightforward to model in ANSYS-AQWA. We thus assume that the center of rotations of the floating system is at the center of gravity of the structure. Such assumption may have some effect in the ANSYS-AQWA pitch solution, but however, it does not alter the computation of the hydrodynamic coefficients (F_e, A_{ii}, and B_{ii}) to be used by WEC-Sim.

WEC-Sim solves the following time domain EoM (i=1,2,...,6):

$$m_{ij}\frac{\partial^2 Y^i}{\partial t^2}=F^i_{ext}+F^i_{rad}+F^i_v+F^i_B+F^i_m, \tag{5.2}$$

where m_{ij} are the components of the mass matrix, F^i_{ext} are the wave excitation force components, F^i_{rad} the force components due to radiated waves, F^i_v the viscous damping force components, F^i_B the net buoyancy force components, and F^i_m the force components due to mooring connections. The F^i_{ext} and F^i_{rad} forces are computed using data from the previous frequency domain solution computed using ANSYS-AQWA. The details of the method are given in [170, 171].

In WEC-Sim, we considered the actual location of the CG and the pitch axis. Also, since WEC-Sim accounts for some non-linearities of the problem, we expect it to provide a more accurate solution than that form ANSYS-AQWA.

In Figure 5.13 we present the heave and pitch RAO results, including the experimental data, the FSI computation, and the ANSYS-AQWA and WEC-Sim results. As it is shown in the heave plot in Figure 5.13(a), the FSI model predicts the resonance peak when the wave period is $T = 1.03s$ with a corresponding RAO of 1.216. This values are nearly identical to the experimental results where the maximum amplification factor of 1.19 occurs for an incident wave of period $1.01s$. In contrast, the lower order methods, ANSYS and WEC-Sim, are unable to accurately predict the heave RAO near the natural frequency. Particularly, WEC-Sim has some values near the heave peak with very low amplification factor far below the experimental data.

Before analyzing the computed RAO results in pitch, we first need to look into the experimental data. As shown in Figure 5.13(b), the exact experimental RAO and period of the incident wave of the resonance peak is not clearly defined. The following four experimental realizations with period close to the peak where tested: $T = 2.480s$, $T = 2.498s$, $T = 2.500s$, and $T = 2.515s$, with the respective amplification factor being $553deg/m$, $475deg/m$, $521deg/m$, and $501deg/m$. The amplitude of the incident wave for these four cases is nearly identical ($0.00725m$, $0.00745m$, $0.00725m$, and $0.00725m$, respectively). The fact that the amplification factor oscillates when slightly increasing the wave period, means that the peak period and amplitude is subject to a certain degree of randomness. Thus we consider the experimental amplification factor for pitch to be within the range $[475, 553]deg/m$. The average value of these four realizations is $512.5deg/m$. Similarly, the period of the incident wave is within the range $[2.480, 2.515]deg/m$ and the average value is $2.498s$.

In the same Figure 5.13(b), the RAO results in pitch computed with the FSI model predict the resonance peak to occur when the period of the incident wave is $T = 2.425s$. This value, compared to the averaged experimental pitch natural period of $2.498s$, is 2.9% lower. This is consistent with the observations of the decay tests that our model was slightly under-predicting the pitch decay period by 2% as a result of considering a simplified geometry of the floating system. Looking at the RAO of the peak, the FSI result is shown to be at $417deg/m$, which is 18.6% below the experimental value taken from averaging the four near-peak cases.

With regards to the lower order models results, both ANSYS-AQWA and WEC-Sim, capture the resonance peak for a period of the incident wave of $T = 2.4s$, which

is 3.9% below the experimental measurements. Comparing the amplification factor at the pitch resonance peak with that from the experiments, ANSYS-AQWA, with a value of $884 deg/m$, over-predicts it by 63.5% and WEC-Sim, with a value of $696 deg/m$, over-predicts it by 35.6%. As it was expected from the fact that WEC-Sim consider non-linearities of the problem and that in the ANSYS-AQWA model the center of rotation was not taken at the actual position but at the CG position, the WEC-Sim results in pitch are significantly better than those from ANSYS-AQWA.

One of the features that we could capture with the present FSI model that is not accounted for by linear models is the effect of the amplitude of the incident wave given a fixed wave period. This phenomenon was already documented in previous works such us in Jung et al. [155]. They carried out, experimentally, the RAO in roll of a rectangular barge restricted to move in a single DoF. They observed that for two cases of incident wave with equal period, the RAO is larger for the case of lower wave amplitude. Also, this phenomenon was observed to be maximum at the peak frequency and to minimize far from the peak. In our pitch RAO results computed with the FSI model we observed the exact same phenomenon. We computed the RAO at the pitch peak period ($T = 2.425s$) with two cases of different wave amplitude, $0.0125m$ and $0.0170m$. The RAO for the case of lower wave amplitude is $417 deg/m$ and for the case of larger wave amplitude $297 deg/m$. We performed the same analysis to a case in which the incident wave period is far from the peak ($T = 2.45s$) to show how this effect, far from the peak, diminishes. For that wave period, a case with wave amplitude of $0.0085m$ resulted in a amplification factor of $202 deg/m$ and a case with wave amplitude of $0.0121m$ to a slightly smaller amplification factor of $194 deg/m$. Note that in Figure 5.13(b) only the wave case that results in a maximum RAO are displayed.

Finally, in Figure 5.11 we present the structural response in, heave (a) and in pitch (b), as well as the time evolution of the surface elevation at a point located at $X = 15m$ (c). These results correspond to the case in which the incident wave has a period of $2.425s$ and an amplitude of $0.00125m$. A common behavior observed in this figure and in most of the remaining wave cases is that while the pitch response grows progressively, the heave response grows sharply in just 3 to 4 oscillations. This sharp start of the heave motion results in a fictitious maximum amplitude usually at the fourth oscillation. Since this is an artifact effect that could have been removed by using a ramping function, we

decided not to consider the first four oscillations in the calculation of the maximum amplitude of the oscillation. For the same case, we show in Figure 5.12 the surface elevation contours at different instances in time. The figure shows how the wave field evolves in time, going from a clean state, free of disturbances, to a state in which diffracted, radiated, and reflected waves have spread.

In summary, the results we presented herein demonstrate that the FSI model of [167] can reasonably capture wave-body interactions with overall better agreements than that predicted by the two lower order models considered, which were based on the potential flow theory.

Chapter 6

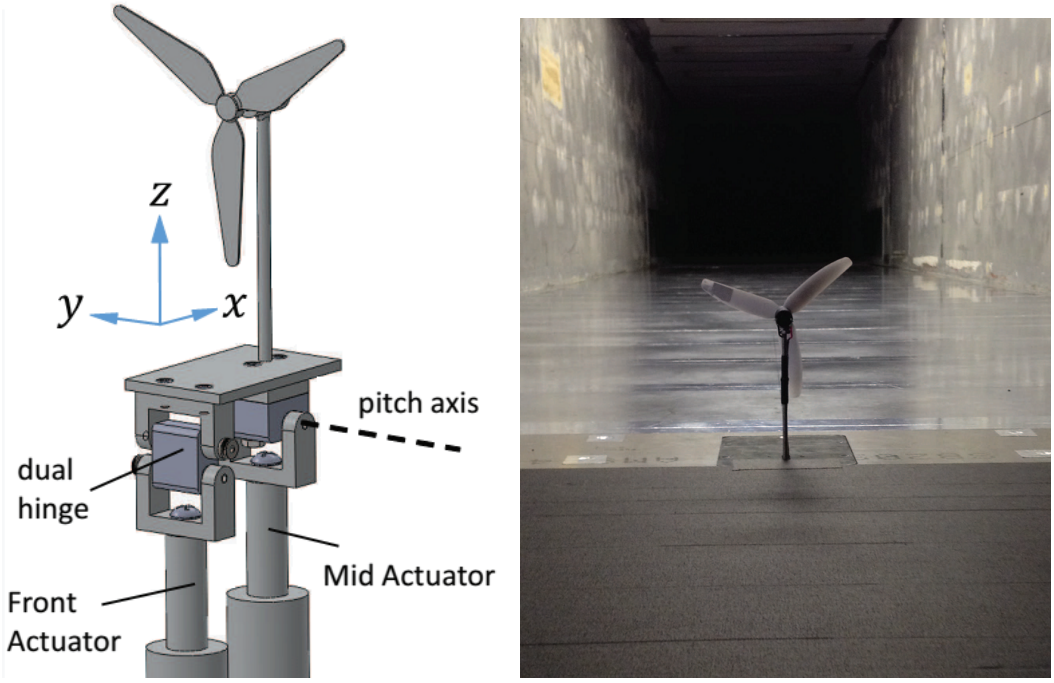
Simulation of an oscillating wind turbine under prescribed motion

In this chapter, we perform LES of a wind turbine that undergoes an oscillatory prescribed motion in the pitch directions aiming to investigate the effect of these motion to the turbine wake.

6.1 Description of the oscillating turbine case

The turbine settings and flow conditions for this test case are taken identical to those from the phase II experiments of Feist et al. [165]. As already discussed in section 5.1, the experiments of [165], studying an offshore floating wind turbine, are decomposed in two quasi-coupled phases. The first phase, briefly summarized in section 5.1, neglects the wind flow and only considers the hydrodynamics of the floating system under the wave conditions representative from the PNW offshore climate. In the phase II of the experiments, a wind tunnel experiment was carried out to study the effects of the rigid body motions recorded in phase I to the air flow behind the turbine.

In the experimental study, the miniature wind turbine of phase II experiments was mounted on an actuator system that allowed to prescribe heaving and pitching oscillatory motions to the turbine, as depicted in figure 6.1(a). This turbine system was positioned at the test section, of length $16m$, height $1.5m$, and width $1.7m$, of the



(a) (b)

Figure 6.1: (a) Schematic description of the actuator system that induce the heaving and pitching motions to the turbine model. (b) View of the test section of the SAFL wind tunnel with the experimental turbine model. These two figures are reproduced from [165].

SAFL atmospheric boundary layer wind tunnel, as illustrated in figure 6.1(b). The inflow conditions facing the turbine are a neutral and stable boundary layer that develops along the length of the tunnel test section. Using the law of the wall ($W = w_*/\kappa \ln y/y_0$, where κ is the Von Kármán constant taken as 0.4 and y_0 is the surface roughness taken as $0.03mm$), the frictional Reynolds number computed as $Re_\tau = u_*\delta/\nu$ was estimated to be approximately $Re_\tau \approx 0.3 \times 10^4$, and the frictional velocity $u_* = 0.1m/s$. The height of the boundary layer is approximately $0.4m$ and the free stream velocity $2.73m/s$.

The turbine model for this case has a 3-blade rotor of type GWS/EP-6030x3 with a diameter of $0.128m$ and a hub height of $0.104m$ (see [172] for details). This hub height corresponds to a fourth of the boundary layer height, which is a common value observed in utilily scale turbines. The hub velocity is $2.26m/s$.

6.2 Inflow conditions

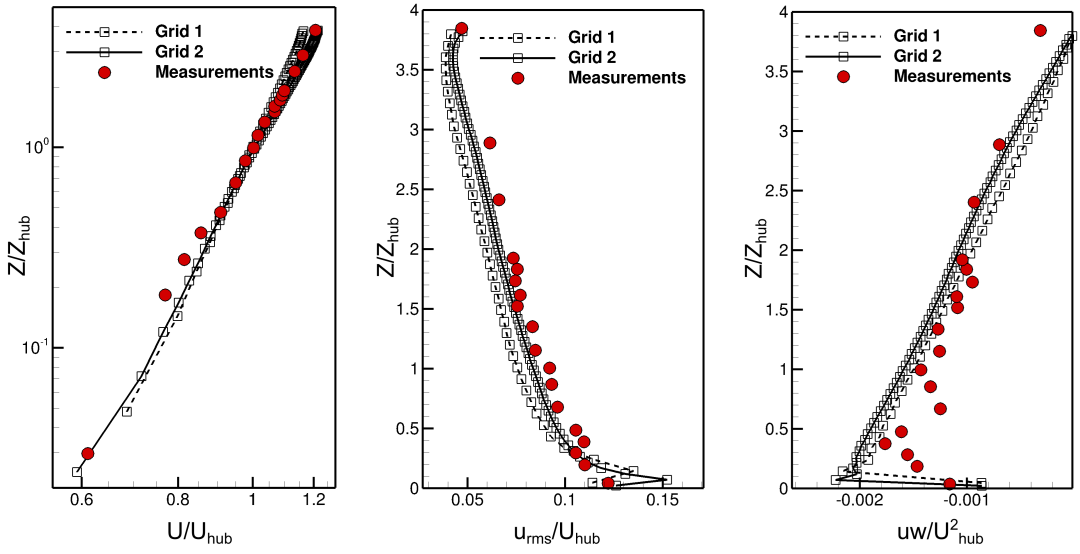


Figure 6.2: Upstream flow conditions for the oscillating turbine case. Vertical profiles of (a) stream-wise averaged velocity, (b) stream-wise turbulence intensity, and (c) primary Reynolds shear stress. The computed results are from the channel flow precursor simulation and the measured data was taken from the experimental work of [165].

To obtain the turbulent boundary layer flow conditions equivalent to the experiments, we first perform a precursor simulation, using LES, of the classical channel flow case as described in [173]. For the channel flow case, we consider a rectangular domain of dimensions $2m$, $1.2m$, and $0.4m$ in the stream-wise (X), span-wise (Y), and vertical (Z) directions, respectively. We consider two uniform meshes of different resolutions: Grid 1 is of size $61 \times 121 \times 41$ and Grid 2 of size $121 \times 241 \times 81$.

We set no-slip wall boundary conditions at the bottom boundary, slip-wall conditions at the top boundary, and periodic boundary conditions at the inlet, outlet and lateral boundaries. Since the vertical grid resolution is not sufficient to resolve the bottom wall boundary layer, we applied the wall model of [122] to reconstruct the velocity boundary condition.

In the present simulation the friction Reynolds number was set equal to that from

the experiments, ($Re_\tau = 3000$), which corresponds to a Reynolds number flow of $Re = 137900$ as estimated with the following expression given in [173]:

$$Re_\tau \approx 0.09 Re^{.88}. \tag{6.1}$$

Using the Reynolds number definition for this test case, $Re = 2\delta U/\nu$, where δ is the channel half height, and U the flow bulk velocity, and taking the kinematic viscosity of air as $\nu = 1.6 \times 10^{-5}$, the flow bulk velocity is estimated as 2.758. The time step size is taken as 0.001.

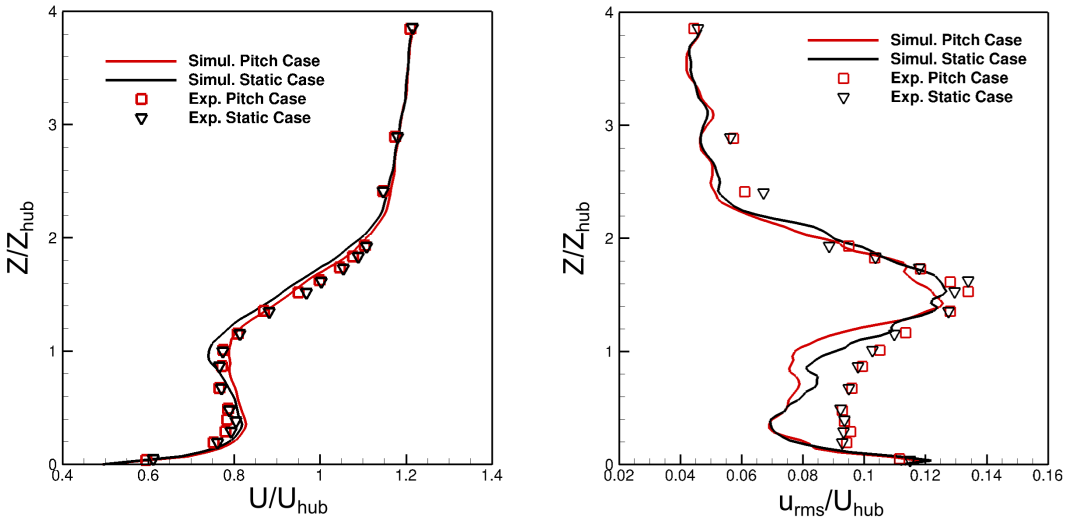
In figure 6.2, we present the vertical profiles of stream-wise average velocity, stream-wise turbulence intensity, and turbulence shear stress, computed on grids 1 and 2 and the experimental data. As seen in the figure, the results computed on the finer mesh, grid 2, are in overall better agreement with the measurements than those computed on grid 1. Particularly, the stream-wise velocity profile presented in log-scale in figure 6.2(a), shows that the finer mesh can better predict the slope of the logarithmic velocity profile from the experiments. The Turbulence intensity shown in figure 6.2(b) is also better predicted with the finer mesh.

The degree of agreement of these results is similar to other numerical studies of this type available in the literature (see for example [145]). We thus take the results computed on grid 2 as inflow conditions for the turbine simulation case.

6.3 Results of the oscillating turbine case

In this section we perform LES of the flow past the model wind turbine. We consider two cases: a first case in which the turbine is kept fixed and a second case in which the turbine oscillates in the pitch DoF under prescribed motion. For both cases, the flow around the turbine rotor is not resolved but modeled with the actuator line method as described in section 2.1.8. Also, in this simulation we neglect the effect of the tower and rotor hub. The tip-speed ratio (TSR) of the turbine is considered fixed and equal to $\lambda = 4.5$ for both cases. The TSR is defined as $\lambda = \omega R/U$, where ω is the rotor angular velocity, R is the rotor diameter, and W is the inflow reference velocity.

In the prescribed motion case the turbine oscillates with respect to the pitch axis, which is located $0.039m$ below the wall. The amplitude of the oscillations is of $4deg$

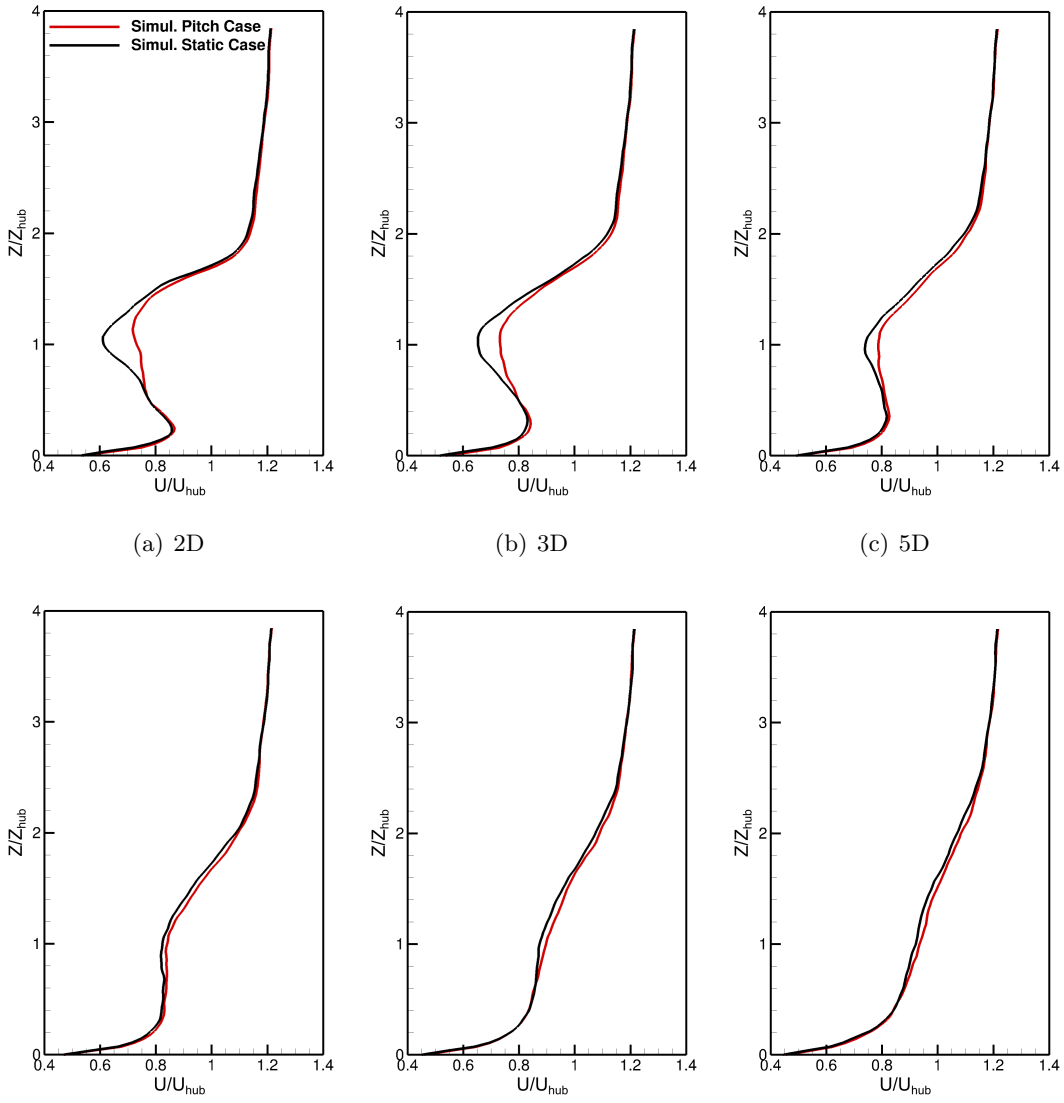


(a) (b)

Figure 6.3: Vertical profiles of averaged stream-wise velocity in (a) and turbulence intensity in (b). The results correspond to $5D$ downstream of the turbine and centered on $X = 0$. The measured data is from [165].

and the frequency of oscillation equals to $3Hz$. This is a turbine motion case taken from the work of [165] for which measured data is available. It was defined using the experimental RAO results, estimating the structural response of a relatively large wave. In the wave basin scale, the wave was chosen to be of amplitude equal to $0.11m$ and of period equal to $2s$.

The computational domain for this case is similar to that from the rectangular channel flow simulation presented in the previous section. While the span-wise and vertical directions remain unchanged, $1.2m$ and $0.4m$, respectively, the stream-wise direction is defined to a longer length of $3m$ ($X = [0, 3]m$, $Y = [-0.6, 0.6]m$, and $Z = [0, 0.4]m$, in the stream-wise, span-wise, and vertical directions respectively). A non-uniform grid of size $171 \times 241 \times 109$ is employed. The mesh is composed of a two-region structure with an inner region of constant grid spacing enclosing the turbine and most part of the turbine wake and an outer region within which the spacing increases progressively towards the boundaries. In the inner region, defined in the intervals $X = [0, 1.5]m$, $Y = [-0.15, 0.15]m$, and $Z = [0, 0.17]m$, the spacing is equal to $0.005m$.

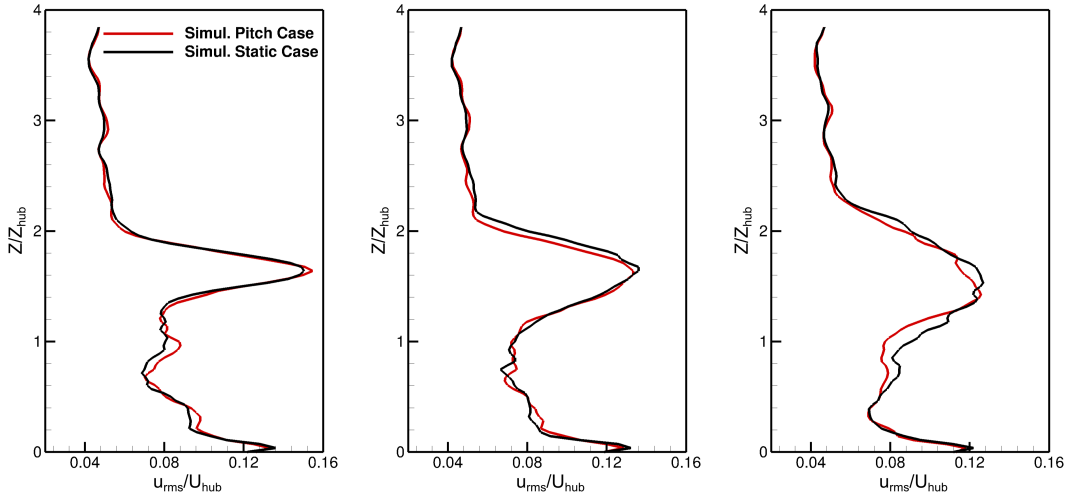


(a) 2D (b) 3D (c) 5D

(d) 7D (e) 10D (f) 14D

Figure 6.4: Vertical profiles of averaged stream-wise velocity at different distances downstream of the turbine and centered on $X = 0$. The measured data is from [165].

Similar to the previous channel flow case, the boundary conditions employed are no-slip wall for the bottom boundary, slip wall for the top boundary, and periodic in the remaining boundaries. For the bottom wall the wall model strategy has been applied

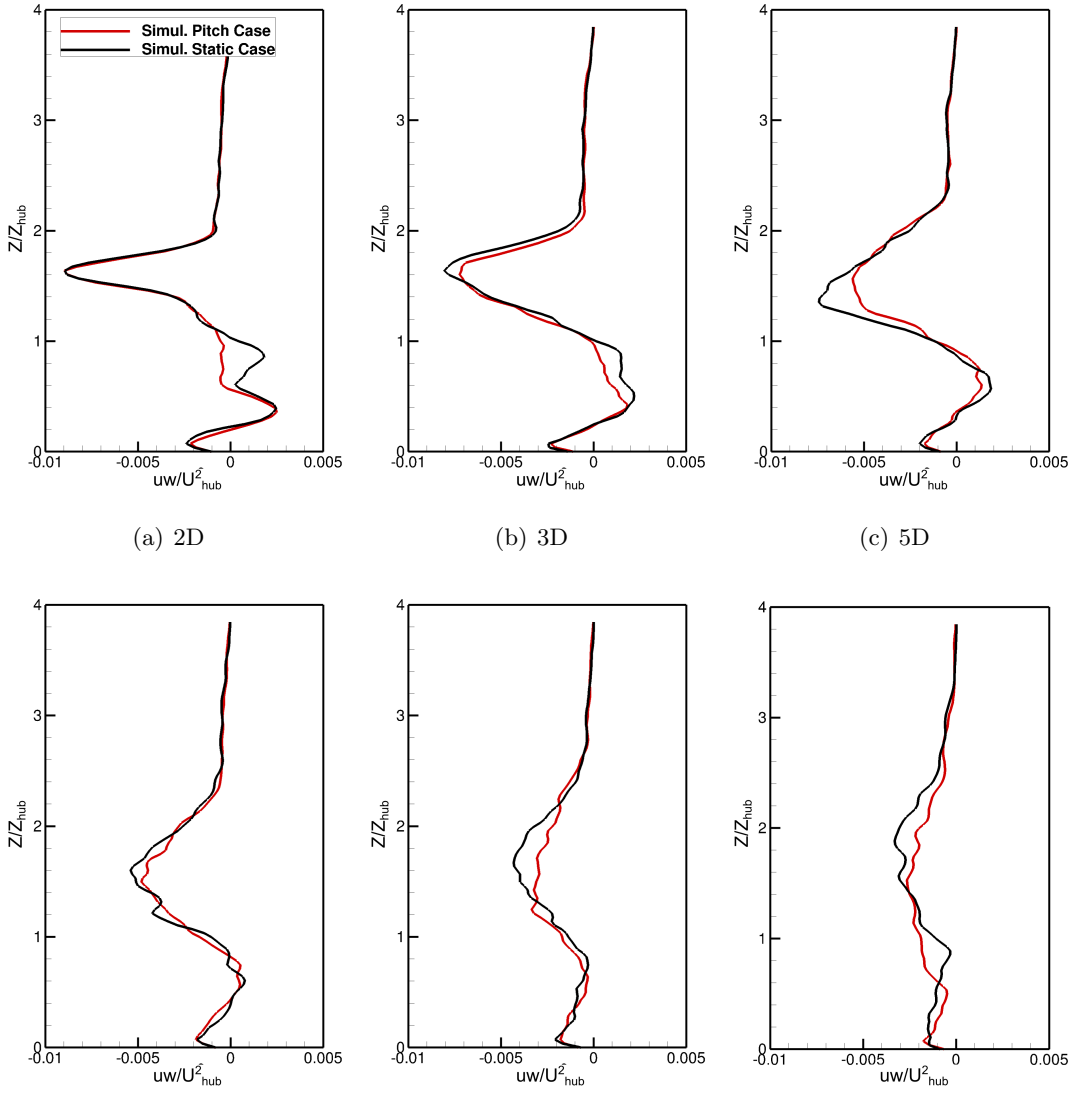


(a) 2D (b) 3D (c) 5D

(d) 7D (e) 10D (f) 14D

Figure 6.5: Vertical profiles of stream-wise turbulence intensity at different distances downstream of the turbine and centered on $X = 0$. The measured data is from [165].

and the time step of the simulation is $0.005s$. To feed the inflow velocities from the precursor simulation to the turbine simulation case at each time step, we used linear interpolation as the grid nodes of the two cases do not coincide.



(a) 2D (b) 3D (c) 5D

(d) 7D (e) 10D (f) 14D

Figure 6.6: Vertical profiles of shear stress at different distances downstream of the turbine and centered on $X = 0$. The measured data is from [165].

In figure 6.3 we present the vertical profiles of stream-wise averaged velocity and stream-wise turbulence intensity for the static turbine case and the pitch oscillating case at 5D downstream of the turbine, which is where experimental data is available. As shown in the figure, the computed results compare well with the experimental results

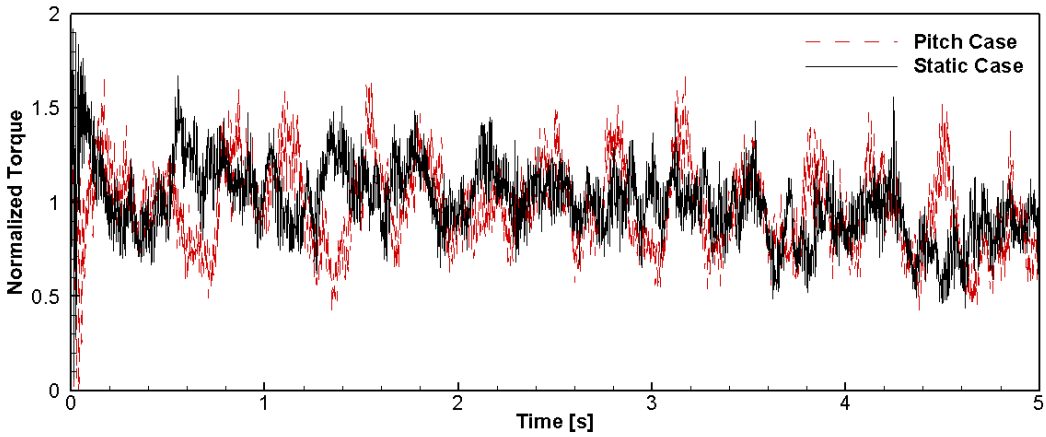


Figure 6.7: Time evolution of the computed torque of the turbine model case. The torque has been normalized by the averaged torque of the static case.

considering that the turbine has been modeled with the actuator line method. The computed results from the static case and the pitching case are nearly identical, reinforcing the fact that the turbine pitching motion has minimal effects on the averaged velocity and turbulence statistics, as it was already seen in the experiments.

In figures 6.4, 6.5, and 6.6, the computed vertical profiles of averaged velocity, turbulence intensity, and shear stress, are plotted for different distances downstream of the turbine. From the velocity profile plots (figure 6.4), we observe that the velocity deficit in the near-wake region is slightly lower in the pitch case than in the static case, and this effect diminishes far down from the turbine. The time-averaged turbine torque is also consistent with this results, as it is 3% lower in the pitching case, showing that the oscillating turbine case captures slightly less momentum from the air flow. It is also interesting to see the time evolution of the turbine torque, shown in figure 6.7. The torque for the pitching case oscillates at the same frequency as the turbine pitching motion. When the system pitches in the clockwise direction, i.e. the turbine rotor moves forward towards the flow direction, the torque gets lower, and when the turbine pitches counterclockwise, i.e. the turbine rotor moves backwards against the flow direction, the torque gets higher.

In conclusion, the averaged quantities show minimal differences between the two cases. In future studies we will investigate the fluctuations induced in the wake by the

pitching motion, by looking into phase averaged results, as done in [165]. We will also consider other cases with turbine motions in other DoF.

Chapter 7

FSI simulation of an offshore floating wind turbine

The aim of this simulation case is to demonstrate the full capabilities of the proposed far-field/near-field computational framework by simulating an offshore floating wind turbine under realistic wind and wave conditions representative from a site-specific environment such as the PNW.

7.1 Definition of the offshore environmental conditions

To determine the wind velocity and the wave field that is representative from the environmental conditions of the PNW, we use the data measurements taken from the Station 46041 of the National Data Buoy Center. In particular, [174] provides an annual JPD of the most commonly occurring waves elaborated with several decades of wave measurements. According to the JPD, the most common waves occurring more than 10% of the time have a dominant period of about 10.0 to 12.9s, and a wave height between 1.5 and 2.4s. Based on that, we choose, for the wave field, a broadband wave spectrum of dominant peak period $T_{peak} = 12.75s$, which using the dispersion relation, is equivalent to a peak wavelength of $L_{peak} = 251m$.

With regards to the wind field, [174] provides monthly averaged values of wind speed taken at $5m$ above the MSL. The values show that the averaged wind speed is considerably lower during warmer season (August and September), approximately $3.8m/s$, and

increases to averaged values of about $7m/s$ during the colder season (December and January). We thus adopt an intermediate wind speed of $5m/s$ at $5m$ above the MSL. Using the law of the wall, the extrapolated wind speed at the turbine hub height (133m) corresponds to approximately $9m/s$.

7.2 The floating wind turbine

We consider a large floating wind turbine system, consisting of 13.2MW wind turbine installed on a tri-column triangular platform. The turbine rotor, which is a design by Sandia National Laboratories, has a $200m$ long diameter and a three SNL100-00 blades (see Griffith and Ashwill [166] for details). The hub height with respect to the MSL of the turbine is $133.5m$.

The floating platform that supports the turbine, designed by Principle Power, is a scaled up version (using Froude number similitude and a scale factor of $\lambda = 1.4$) of the the OC4 semi-submersible design presented in the work of Robertson et al. [175]. It is composed of a main central column of $9.1m$ diameter and $36.4m$ height and three offset cylindrical columns of $16.8m$ diameter that are interconnected through pontoons and cross braces. The offset column, which are spaced $70m$ apart from each other, have a base of $33.6m$ diameter and $8.4m$ height. The geometry of the four columns is illustrated in figure 7.1.

7.2.1 Hydrodynamic properties of the floating turbine

In order to introduce all the floating turbine parameters required for computing the 6 DoF dynamics of the floating wind turbine, we first rewrite the rigid body EoM given by equations (2.12) and (2.12), in matrix form, including case specific external forces. The 6 DoF EoM, can be written in the inertial frame of reference and in principal axis, as follows:

$$\mathbf{M}\frac{\partial^2 \mathbf{q}}{\partial t^2} = \mathbf{F}_{fluid} + \mathbf{F}_{gyro} + \mathbf{F}_{mooring} + \mathbf{F}_{actline} \tag{7.1}$$

where $\mathbf{M}$ is the 6×6 mass matrix, $\mathbf{q}$ the 6 DoF position vector (includes 3 displacements q_x, q_y, and q_z, and 3 rotations q_{θ_x}, q_{θ_y}, and q_{θ_z}), and $\mathbf{F}_{fluid}$, $\mathbf{F}_{actline}$, $\mathbf{F}_{gyro}$, and $\mathbf{F}_{mooring}$ are the vectors containing forces and moments due to the fluid, the rotor thrust force,

the gyroscopic effects of the spinning rotor, and the mooring system, respectively.

The mass matrix when the equations are in principal axis have zero non-diagonal terms, and reads as follows:

$$\mathbf{M} = \begin{Bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_x & 0 & 0 \\ 0 & 0 & 0 & 0 & I_y & 0 \\ 0 & 0 & 0 & 0 & 0 & I_z \end{Bmatrix}. \tag{7.2}$$

In the present case, we take the mass of the floating turbine system to be $m = 3.7 \times 10^7 kg$, and the inertia $I_x = 3.64 \times 10^{10} kg \cdot m^2$, $I_y = 3.64 \times 10^{10} kg \cdot m^2$, and, $I_z = 6.39 \times 10^{10} kg \cdot m^2$. The center of gravity is located $18.84m$ below the MSL and the platform draft is $28m$.

$\mathbf{F}_{fluid}$ is computed by integrating the fluid pressure and shear stresses at the surface body as described in section 2.1.6. The thrust force, when using the actuator line model described in section 2.1.8, can be computed in the following manner:

$$\mathbf{F}_{actline} = \sum_{n_l} (\mathbf{n}_t \mathbf{L} a + \mathbf{n}_t \mathbf{D} a) \tag{7.3}$$

where $\mathbf{L}$ and $\mathbf{D}$ are the lift and drag coefficients computed at each line element using equations (2.47) and (2.48), respectively, n_l is the total number of elements (including all blades), a is the length of each element, and $\mathbf{n}_t$ is the rotor normal direction pointing towards the stream-wise direction.

The moments due to the gyroscopic effects of the spinning rotor are computed using the method described in [176]. The method, which is based on the assumption that the

turbine system undergoes small rotations, reads as follows:

$$\mathbf{F}_{gyro} = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ I_p \Omega \frac{\partial q_{\theta_y}}{\partial t} \\ -I_p \Omega \frac{\partial q_{\theta_x}}{\partial t} \\ 0 \end{Bmatrix} \tag{7.4}$$

where I_p is the rotational inertia of the rotor, $6.471 \times 10^8 kg \cdot m^2$ for the present turbine, and Ω is the angular velocity of the rotor determined by the tip speed ration and the inflow velocity.

The calculation of the last necessary term for the turbine simulation, $\mathbf{F}_{mooring}$, is described in the subsequent section.

7.2.2 The mooring system

The floating platform is secured in place using a mooring system consisting of three catenary lines distributed symmetrically with respect to the platform vertical axis as illustrated in figure 7.2. The design is taken from Robertson et al. [175], with the difference that it has been properly scaled to accommodate the larger dimensions of the present turbine design as of $\lambda = 1.4$. The mooring cables are attached to the upper part of the base columns at a location corresponding to a water depth of $19.6m$ and a radial distance of $57.2m$. The other end of the cables is attached to the sea bottom at a radial distance of $1172m$ and a depth of $280m$.

To incorporate the mooring system into the computational framework we opt for the linearized model of [175]. In the linearized model, the forces induced by the entire mooring system, $F^{mooring}$, are estimated in the following manner:

$$\mathbf{F}^{mooring}(\mathbf{q}) = \mathbf{F}^{mooring,0} - \mathbf{C}^{mooring}\mathbf{q}, \tag{7.5}$$

where, $\mathbf{F}^{mooring,0}$ is the force vector of the mooring system when the system is in equilibrium, and $\mathbf{C}^{mooring}$ is the 6×6 linearized restoring matrix. Robertson et al. [175] provide the values of $\mathbf{F}^{mooring,0}$ and $\mathbf{C}^{mooring}$ for the present mooring system, estimated

using a linearized perturbation analysis in FAST. Using an scaling factor of λ^3 for the forces, a factor of λ^4 for the moments, a factor of λ for the lengths, and a unit factor for the angles, the scaled $\mathbf{F}^{mooring,0}$ and $\mathbf{C}^{mooring}$ read as follows:

$$\mathbf{F}^{mooring,0} = \left\{ \begin{array}{c} 0 \\ 0 \\ -5.05 \times 10^6 \\ 0 \\ 0 \\ 0 \end{array} \right\}, \tag{7.6}$$

and

$$\mathbf{C}^{mooring} = \left\{ \begin{array}{ccc} 1.39 \times 10^5 N/m & 0 & 0 \\ 0 & 1.39 \times 10^5 N/m & 0 \\ 0 & 0 & 3.74 \times 10^4 N/m \\ 0 & 2.94 \times 10^6 Nm/m & 0 \\ -2.94 \times 10^6 Nm/m & 0 & 0 \\ 0 & 0 & 0 \end{array} \right.$$
$$\left. \begin{array}{ccc} 0 & -2.96 \times 10^5 N/rad & 0 \\ 2.96 \times 10^5 N/rad & 0 & 0 \\ 0 & 0 & 0 \\ 3.35 \times 10^8 Nm/m & 0 & 0 \\ 0 & 3.35 \times 10^8 Nm/m & 0 \\ 0 & 0 & 4.49 \times 10^8 Nm/m \end{array} \right\}. \tag{7.7}$$

We note that the linearized mooring model assumes small motions. This assumption is reasonable given that the objective of the present simulation is for demonstrating the capabilities of the framework. In future simulation we will consider more elaborate models, treating individual lines and considering non-linear effects.

7.3 Simulation results

In this simulation we employ the far-field/near-field approach developed herein for studying floating structures under ocean wind and waves. The near-field computational domain is $2675m$ long in the stream-wise direction ($X = [-550m, 2125m]$) and $1750m$ wide in the span-wise direction ($Y = [-875m, 875m]$), the water depth is $280m$ and the air column above the free surface is $1000m$. The source region has a length ϵ_x of $224m$ and is centered on $X = 0$. The floating turbine is positioned downstream of the sponge layer at $X = 900m$ and centered on $Y = 0m$. We use a non-uniform mesh of size $163 \times 207 \times 259$ that is schematically described in figure 7.3. In the stream-wise direction, the spacing Δx is constant at the following two regions: (1) at the source region ($X = [-112m, 112m]$) in which the spacing is $\Delta x = 5.6m$; and (2) at the region containing the floating structure and defined by $X = [848.2m, 951.8m]$ in which the spacing is $\Delta x = 5.18m$. From the end of the first region ($X = 112m$) to the beginning of the second region ($X = 848.2m$), Δx varies smoothly across the two values, and outside of these two regions the spacing increases progressively towards the inlet and outlet boundaries. In the vertical direction the spacing Δz follows the same spacing pattern also with two regions of constant spacing: (1) the region from $Z = -40m$ to $Z = 20m$ which has spacing Δz equal to $2m$ and comprises the floating platform and the free surface; and (2) the region from $Z = 130m$ to $Z = 250m$ which has spacing Δz equal to $5m$ and comprises most of the rotor. Finally, in the span-wise direction the spacing Δy is constant and equal to $5m$ at a single region spanning from $Y = -110m$ to $Z = 110m$ and enclosing the floating turbine. The stretching ration used in all direction is always limited to 1.05 and its variation follows a hyperbolic function. The thickness of the interface is set to $\epsilon = 4m$, the gravity to $g = 9.81m/s^2$, and the density and dynamic viscosity for the water to $1000kg/m^3$ and $1.0 \times 10^{-3}Pas$, respectively, and for the air $1.2kg/m^3$ and $1.8 \times 10^{-5}Pas$. Slip-wall boundary conditions is adopted at the 6 boundaries, and the sponge layer method with thickness $200m$ is applied at the four lateral walls.

The geometrical parts of the platform considered in the simulation and its dimensions are presented in figure 7.1 which also shows the structural mesh used for discretizing the structure in the CURVIB method. The structural elements interconnecting the four

columns have been neglected due to its small size in comparison to the large dimensions of the columns.

The turbine rotor in the present simulation is treated with the actuator line model implementation of [145]. The turbine is simulated with a constant tip speed ratio of 8, which given a hub height incoming velocity of about $9m/s$, is estimated to be close to the optimal value for performance.

The wind and wave conditions are developed with a precursor simulation using the far-field model. For the air phase of the far-field model, the domain size is $6280m$, $3140m$, and $1000m$, in the stream-wise, span-wise, and vertical direction, respectively. A non-uniform mesh of size $128 \times 128 \times 128$ is used. While in the stream-wise and span-wise directions the spacing is constant, in the vertical direction the grid cells are clustered near the free surface. The mesh of the HOS for simulating the wave field is uniform and of size 769×769. The free surface initial condition is a broadband wave spectrum of type Joint North Sea Wave Observation Project (JONSWAP) and wave period peak $T_{peak} = 12.75s$. The wind field, which is driven by a constant pressure gradient such that the velocity at height $5m$ is $6m/s$, has been solved in a coupled manner with the wave field. The far-field simulation has been advanced about 300000 time steps, with a time step size of $0.545s$, before start feeding to the near-field domain. That was to ensure that fully developed wind and wave conditions were achieved. The wave spectrum for the developed wave field is shown in figure 7.4.

In figure 7.5 we present near-field results of the floating wind turbine including the free surface and the stream-wise velocity at a horizontal plane at hub-height. As seen in the figure, the wave field clearly shows the formation of radiated waves induced by the motion of the platform. The structural response of the floating structure in the six DoF is given in figure 7.6. Note that surge, sway, and heave correspond to the transnational DoF in the stream-wise, span-wise, and vertical direction, respectively, and roll, pitch, and yaw, rotations with respect to the X, Y, and Z axis, respectively. Looking at the pitch response in sub-figure 7.6(b), the turbine is slightly inclined towards positive angles as a result of the wind effect.

This test case clearly illustrates the ability of the method to simulate wind-wave-body interactions using real environmental conditions and a complex floating structure. Some of the models that have been integrated in the framework for this simulation,

such as the mooring system model or the computation of the gyroscopic forces, are simple models that assume low amplitude motions. The flexibility of the code, however, allows easy implementations of more sophisticated methods for individual aspects of the application.

More in depth results analyzing the water and air flow around this and other floating turbines are certainly necessary to get better insights of the problem. The computational framework we presented is a unique tool that allows, for the first time, to study offshore and nearshore application considering as many coupled physical phenomenon.

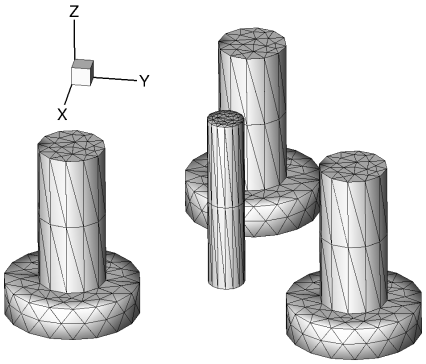


Figure 7.1: Mesh of the floating platform used in the IB method in the offshore floating wind turbine case.

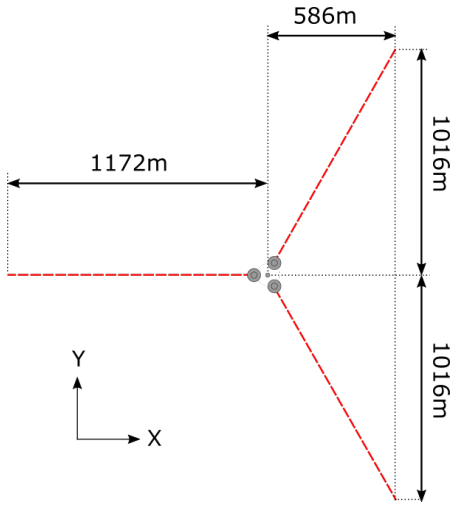


Figure 7.2: Schematic description of the mooring system composed of three catenary lines employed in the floating turbine case.

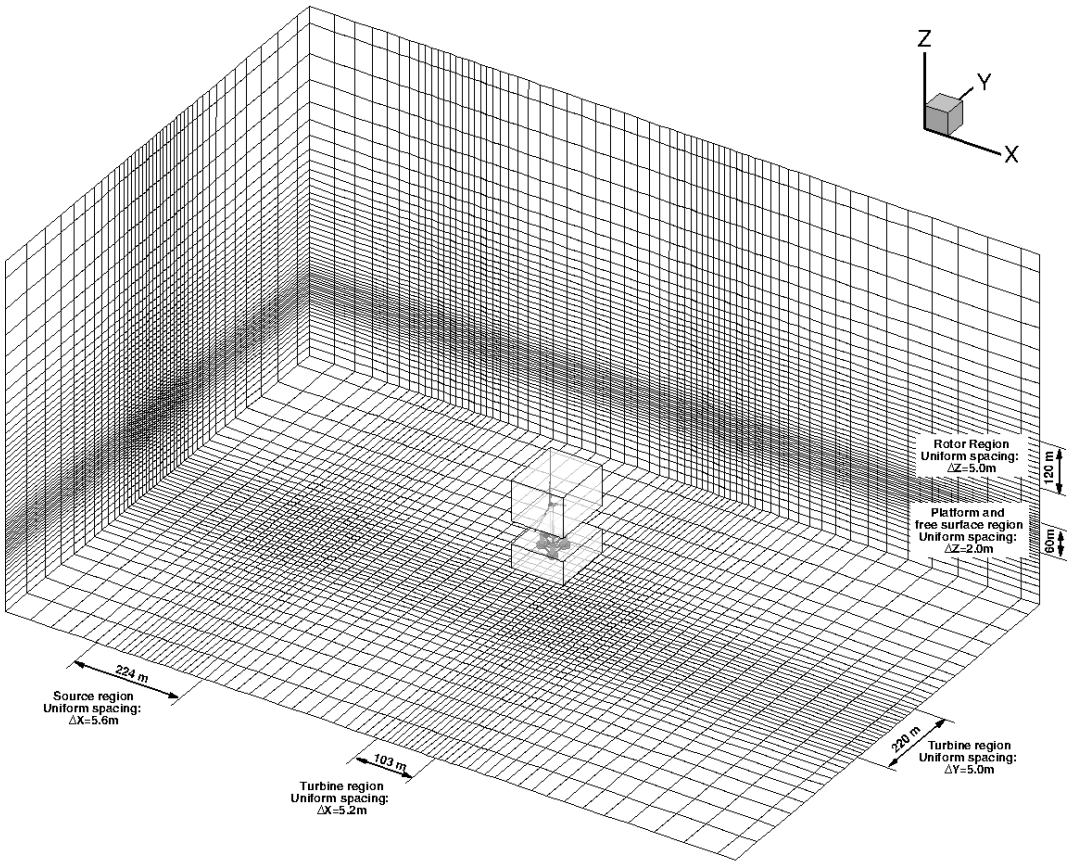


Figure 7.3: Schematic description of the fluid mesh used in the far-field domain of the offshore floating wind turbine case. The rectangular boxes indicate the two regions of constant grid spacing where the floating turbine is located. In this figure, for every grid line shown four are skipped.

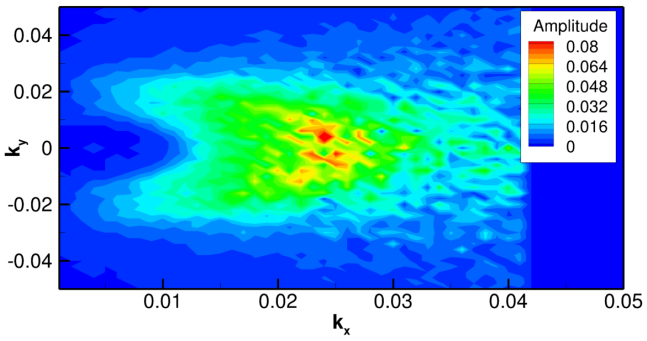
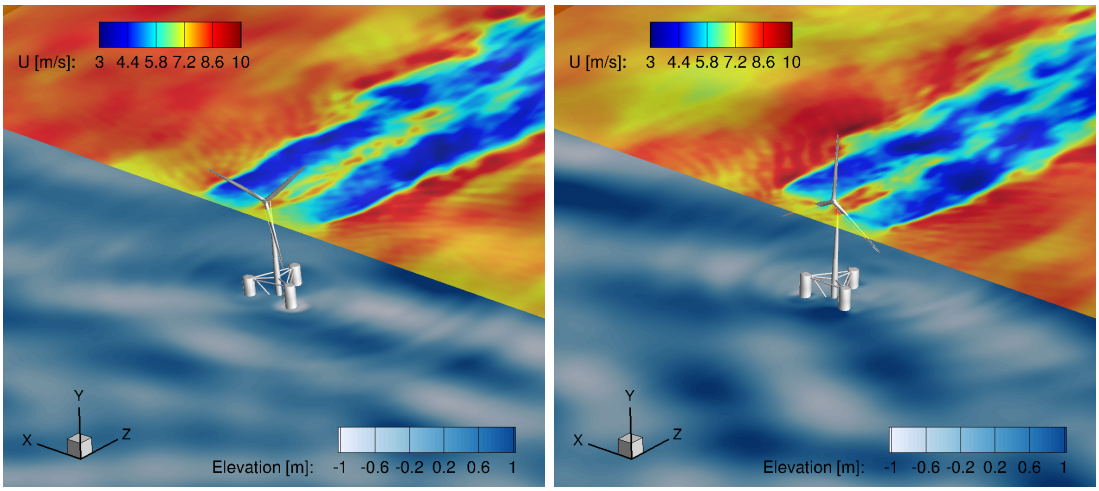
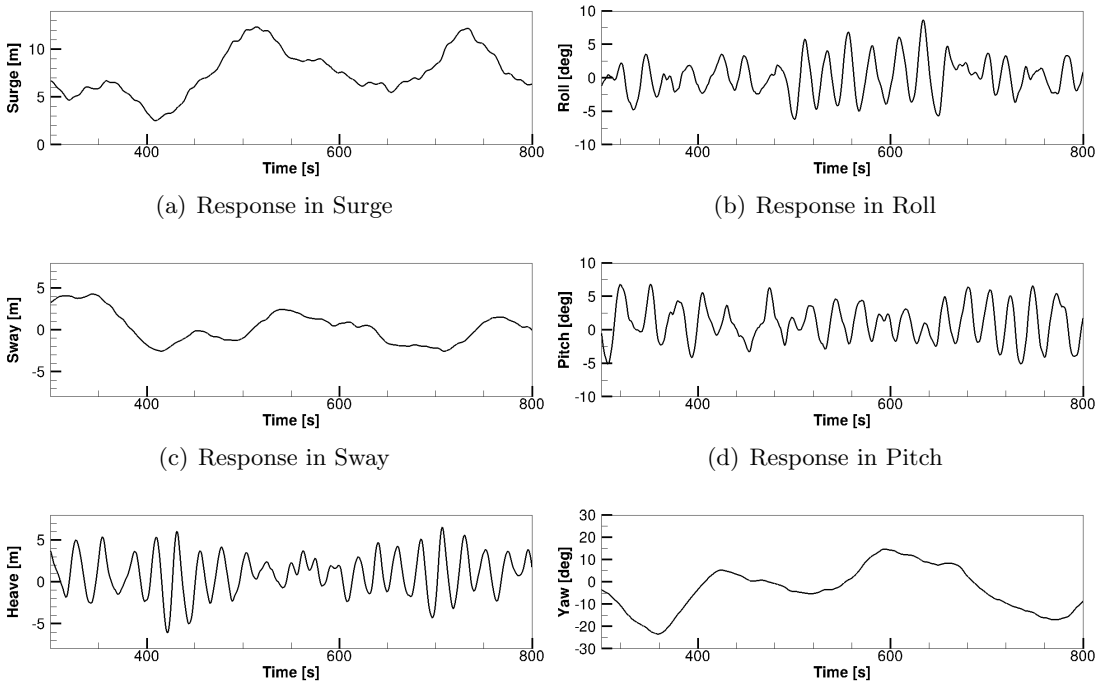


Figure 7.4: Definition of the broadband wave spectrum represented by wave amplitude contours as function of the directional wavenumbers. Computed at the far-field domain at a time for which the flow is fully developed ($t = 163500s$).



(a) Time $t = 273s$ (b) Time $t = 685s$

Figure 7.5: Offshore floating wind turbine case. 3D view of the floating wind turbine with the free surface colored with elevation contours and a horitzonal plane at hub heigth of stream-wise velocity.



(a) Response in Surge

(b) Response in Roll

(c) Response in Sway

(d) Response in Pitch

(e) Response in Heave

(f) Response in Yaw

Figure 7.6: Offshore floating wind turbine case. Structural response of the floating turbine system in the six DoF.

Chapter 8

Summary, conclusions and future work

8.1 Summary and conclusions

We developed, validated, and demonstrated the predictive capabilities of a novel computational framework that can simulate real life complex floating structures and its interaction with realistic ocean waves and wind fields. To efficiently deal with the computational challenge of large disparity of scales associated to such type of problems we adopted a far-field/near-field domain decomposition approach, i.e., a large-scale domain with periodic boundary conditions, known as far-field domain, in which the offshore ocean conditions are efficiently developed, and a reduced scale domain with high grid resolution, known as near-field domain, where the floating structure is located. In the far-field domain we applied the two-fluid method of Yang and Shen [1, 2], allowing to obtain fully developed wind and wave condition in an efficient manner as the waves are resolved with a potential flow based HOS method. In the near-field domain, a novel FSI model for simulating arbitrarily complex floating rigid bodies interacting with non-linear free surface flows was developed and validated.

The proposed near-field model, published in [167], integrates the FSI-CURVIB method of Borazjani et al. [6] with a level set approach along with a new method for calculating forces due to pressure on submerged structures in two-phase flows. The

so-called pressure projection boundary condition (PPBC) method was shown to mitigate the difficulties encountered when calculating the force on the structure using the standard method developed by Borazjani et al. [6], which employs integration of the pressure on the surface of the volume consisting of all immersed boundary grid cells in the CURVIB method around the structure. While this standard approach works well in single-phase flows, in two-phase flow problems with submerged bodies it does not account for the force imparted on the structure due to the large density difference between the air and water enclosed between the immersed boundary surface and the body and gives rise to a first-order accurate calculation of the pressure force. The proposed PPBC approach employs the normal momentum equation to the body to obtain a more accurate representation of the pressure field on the body via a series of successive projection steps. Numerical tests clearly showed that the proposed method not only reduces the error in the calculation of the force by nearly one order of magnitude relative to the standard approach but also yields near second-order accurate convergence rate for the force and results that are in significantly better agreement with experimental measurements. While the PPBC method was developed and demonstrated herein in the context of the CURVIB approach it is general and should be readily applicable to other sharp-interface immersed boundary methodologies.

To demonstrate the accuracy of the coupled FSI, level set implementation for free surface-body interactions we simulated a series of test cases, including forced motion problems and coupled FSI problems. We showed that for all simulated cases the proposed method is able to replicate with good accuracy the structural response of several laboratory experiments including a free decay test of a circular cylinder, a roll decay test of a rectangular structure, and a free falling wedge impacting the free surface.

In order to prevent the formation of instabilities in problems involving complex air-water interface phenomena, the use of a large interface strip thickness ϵ as well as reduced reinitialization time steps sizes $\Delta\tau$ was found necessary. To reduce the computational cost for implementing these remedies, we systematically investigated the influence of using an overall reinitialization time lower than the time required for full reinitialization of the interface strip. We showed that for the falling cylinder case using a time equivalent to reinitializing 10% of the interface strip was able to predict in sufficiently high accuracy the formation of the breaking wave formed at the side of the

cylinder.

The most challenging case we simulated was that of a wedge impinging on the free surface. Large pressure gradients and forces develop on the structure as it impinges on the surface and decelerates rapidly, which made the solution of the FSI problem especially challenging. Even for this case, however, our method was able to obtain converged solutions but required the use of strong coupling FSI in conjunction with the Aitken method. The latter technique was critical for efficient FSI iterations, since it reduced the number of strong-coiling sub-iterations required for convergence by fifty percent or more.

Our simulations elucidated the rich 3D dynamics resulting in the air phase as the waves induced by the impinging on the free surface wedge break, and showed that most of the energy from the breaking waves is ultimately transferred to the air phase. Massive separation off the cusp of the breaking waves gave rise to complex coherent structures dominated by loops, arch, and hairpin vortices forming an intertwined web of vortical structures that ultimately lead to the flow transitioning to turbulence. The resulting turbulent flow in the air phase was found to grow upward and persist at a significant elevation above the free surface. All these findings are in accordance with the recent findings of Iafrati et al. [154] who also showed that flow separation off breaking waves is the key mechanism for producing turbulence in the air phase. Overall the computed results demonstrated the potential of our method as a powerful tool for simulating the coupled interaction of complex floating structures with a free surface.

Our simulations for the wedge case exposed a limitation of the method when structure over-topping occurs. Our method was able to capture this complex phenomenon, which leads to pockets of water entrapped on the flat surface of the wedge in the air phase. As these pockets, however, spread laterally and grow thinner ultimately reach the scale of resolution of the level set method (the thickness of the interface that can be resolved on a given grid) and spuriously disappear. This is an inherent limitation of the level set approach and can only be resolved by local grid refinement, which can be made practical via an adaptive mesh refinement approach.

To couple the two decomposed domains, the far-field and the near field domains, we adopted a one-way coupling approach, feeding the wind and wave fields that have been fully developed at the far-field domain, to the near-field domain. We opted for a

one-way loosely coupled approach, in contrast to a two-way strongly coupled approach, considering the fact that the far-field domain is applied to generate large-scale offshore flows in which the presence of a single or several marine structures should only have local effects and not alter the ocean environmental conditions.

To incorporate the far-field waves into the near-field domain, we employed an internal wave generation method consisting in applying a pressure force on the free surface in form of source term in the momentum equation. This approach, known as pressure forcing method, was initially proposed by Guo and Shen [39] to generate, suppress, and maintain water waves in a computational approach in which the free surface is treated with a sharp interface method. In this work, we extended the pressure forcing method by adapting it to the diffused interface level set method of Kang and Sotiropoulos [96]. Wave reflections at the lateral boundaries are prevented by using a sponge layer method. We validated the forcing method by applying it to simulate various wave cases of increasing complexity including simple monochromatic waves, monochromatic waves with several frequencies, three-dimensional directional waves with a single or multiple frequencies, and a broadband spectrum. Some of these cases involved transferring the wave field from the far-field domain to the near-field domain, which showed that the two domains were successfully coupled. On the other hand, the coupling of the wind field between the two domains was implement by feeding, at every time step, the velocity at the inlet plane of the near-field domain.

Once the wave generation method was implemented and validated we showed the capability of the method to study wave-body interactions. We applied the framework to replicate the experiments of [165] investigating a barge style floating wind turbine under different wave cases. By simulating a set of free decay test, we showed that the method can capture the natural periods of the system in heave and in pitch with very good agreement when compared to the experimental measurements. A very small discrepancy of less than 2% was observed between the natural period of the simulations and experiments in the pitch DoF. The discrepancy could be explained by the geometry simplification, which may have a small effect on the drag force, and thus on the computed natural period. This discrepancy was not observed on the heave DoF. For one of the heave decay cases we also compared the free surface elevation at two nodal locations showing excellent agreement. The decay tests also served to calibrate the amount of

artificial damping introduced in the system to account for the frictional damping of the experimental apparatus and the damping due to the fact of using a simplified geometry. In contrast to the methods based on potential flow theory in which all damping needs to be added artificially, we showed that our approach already captures most part of the damping, and only a small contribution needs to be added artificially.

Once the optimum value of additional damping was obtained, as in [167], we performed a grid sensitivity study for all decay cases. For the single DoF free decay tests for which experimental data was available, we demonstrated that the computed solution was converging towards a grid independent solution. This solution was in very good agreement with the experimental data. An important conclusion that we could extract from these tests is that when the amplitude of the motion diminishes, approaching very small values, coarse grids tend to be inaccurate or insensible to such small amplitude motions, and finer grid resolution are required. This effect was taken in consideration when selecting the grid resolution to be used in the subsequent simulations of the wave-body interactions. After validating the code for each of the single DoF, we presented simulation results for a decay test, combining an initial heave displacement with an initial inclination. In that case, in addition to the structural response, we showed vorticity contours, illustrating the vortex shedding around the body edges and demonstrating the ability of the method to capture complex flow features around the floating body. Also, we presented free surface elevation contours showing the evolution of the platform radiated waves forming complex free surface elevation patterns.

With the RAO results, we demonstrated the validity of the method for simulating wave-body interactions. We compared the FSI results of the RAO with the experimental results and the simulation results performed using two widely used methods based on the potential flow theory, ANSYS-AQWA and WEC-Sim. We showed that the FSI solution is able to predict the results in overall better agreement with the experimental data. Also, the CFD-FSI method was shown to provide additional information not captured by the lower order methods such as the flow features formed at the vicinity of the structure.

Both, high order CFD-FSI models and lower order based methods can have a mutual benefit if employed in a wise manner. On the one hand, the CFD-FSI framework is able to provide a great deal of information which may be useful for developing and

calibrating low-dimensional dynamic models of floating platforms. On the other hand, since the CFD-FSI model is computationally very intensive, lower order models may be employed to obtain a quick first estimation of the floating platform response under different scenarios of wind and wave conditions. This first approximation can be useful to select the most relevant wind/wave scenarios to be computed in high resolution with the CFD-FSI framework.

It is important to note that the turbine model case we studied was characterized by low amplitude motions, which are conditions favorable for the lower order models. We expect that for more severe environmental conditions only CFD-FSI models would provide reasonable results. The cost of the FSI model is significantly higher, but, can certainly be justified with the higher level of accuracy, the wider range of applicability, and extra features of the problem that can be captured.

To investigate the effect of turbine rigid body motions to the turbine wake we simulated the case of an oscillating wind turbine model studied experimentally in the SAFL wind tunnel in the second phase of the experiments of [165]. In particular, we simulated two cases, a static turbine case and a pitching turbine case. In line with the experimental data, we showed through simulations that the turbine pitching motion has minimal effect to the averaged stream-wise velocity profiles and turbulence statistics. We leave as a future work to further study this case by looking into phase averaged results and to simulate more cases involving motions in the other DoF.

Finally, to demonstrate the full potential of the framework. We applied it to simulate a $13.2MW$ offshore floating wind turbine under realistic site-specific ocean wind and wave conditions. The method was able to capture the turbine response in the 6 DoF, considering the platform-wave interactions, the effect of the turbulent wind on the turbine, and the gyroscopic effect of the spinning rotor.

8.2 Future work

The computational model we presented is a powerful tool that can be applied to a wide range of offshore and nearshore applications. However, there are still limitations and further developments that will need to be addressed as part of the future work.

The key further development that would result in a drastic improvement in the computational efficiently of the framework is the extension to an adaptive mesh refinement (AMR) approach. In such approach, one could refine the grid at the sensitive regions where fine resolutions are required, such as near the free surface interface and at the vicinity of a floating structure, while treating other regions of the domain with a much coarse resolution. For example, with the AMR approach, the operational floating turbine case, modeled herein with the actuator line method (presented in chapter 7), could be studied using a fully rotor resolving approach. This had not been possible using the present method along with the limited computational resources that we had available.

The diffused level set approach employed in this work has some limitations. For example, in some extreme situations, such as in wave breaking, structural over-topping, and air/water entrainments, the mass is not properly conserved. This can be circumvented by coupling the level set method with the VOF method. A coupled VOF-level set method can take advance of the well proven mass conserving properties of the VOF method along with the benefits of the level set method in obtaining a more accurate free surface interface position and slope. Another aspect that needs further investigation in such diffused interface approach, is in the treatment of the boundary layer formed at the air domain due to the wind flow on top of the waves. In this situation, the free surface is seen by the air domain as a no-slip wavy wall, for what the grid resolution required to resolve the viscous sub-layer becomes prohibitively large for large Reynolds number flows. Currently, there is no wall model strategy available in the literature to address this problem. For example, one could develop a wall model that can enforce the shear stress on the free surface by incorporating a distributed eddy viscosity at the nodes located within the diffused interface.

With regards to offshore floating wind turbine applications, there are various aspects to consider in future studies. As a way to demonstrate the capabilities of the framework we simulated an operational floating wind turbine under realistic wind and wave conditions. However, this results could not be validated due to the lack of measurements. Also, future studies should include simulations of the problem at a wind farm scale, considering several arrays of floating wind turbines. Such case would require the implementation of additional methods for addressing more complex mooring systems such as mooring webs. Additionally, the turbine blade and tower are currently treated

in the present model as rigid bodies. It is well known that turbine blades undergo large deformations, affecting the wake of the turbine. A more complex structural solver for dealing with flexible bodies (such as the blades) should be considered. In fact, a versatile and efficient finite elements solver based on thin-walled structural theory has been recently developed and is being incorporated into the framework.

References

[1] Di Yang and Lian Shen. Simulation of viscous flows with undulatory boundaries. part i: Basic solver. *Journal of Computational Physics*, 230(14):5488–5509, June 2011.

[2] Di Yang and Lian Shen. Simulation of viscous flows with undulatory boundaries: Part II. coupling with other solvers for two-fluid computations. *Journal of Computational Physics*, 230(14):5510–5531, June 2011.

[3] Kester Gunn and Clym Stock-Williams. Quantifying the global wave power resource. *Renewable Energy*, 44:296–304, August 2012.

[4] Marc N. Schwartz, Donna Heimiller, Steve Haymes, and Walt Musial. *Assessment of offshore wind energy resources for the United States*. Citeseer, 2010.

[5] U.S. Department of Energy. Marine and hydrodykinetic technology database. `http://en.openei.org/wiki/Marine_and_Hydrokinetic_Technology_Database`. Accessed: 2014-05-15.

[6] Iman Borazjani, Liang Ge, and Fotis Sotiropoulos. Curvilinear immersed boundary method for simulating fluid structure interaction with complex 3D rigid bodies. *Journal of Computational Physics*, 227(16):7587–7620, August 2008.

[7] Liang Ge and Fotis Sotiropoulos. A numerical method for solving the 3D unsteady incompressible NavierStokes equations in curvilinear domains with complex immersed boundaries. *Journal of Computational Physics*, 225(2):1782–1809, August 2007.

[8] C. W. Hirt and B. D. Nichols. Volume of fluid (VOF) method for the dynamics of free boundaries. *Journal of computational physics*, 39(1):201–225, 1981.

[9] Stanley Osher and James A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *J. Comput. Phys.*, 79(1):12–49, November 1988.

[10] Jianming Yang and Frederick Stern. Sharp interface immersed-boundary/level-set method for wavebody interactions. *Journal of Computational Physics*, 228(17):6590–6616, September 2009.

[11] T. Sebastian and M. A. Lackner. Characterization of the unsteady aerodynamics of offshore floating wind turbines. *Wind Energy*, 2012.

[12] A. Babarit, J. Hals, M. J. Muliawan, A. Kurniawan, T. Moan, and J. Krokstad. Numerical benchmarking study of a selection of wave energy converters. *Renewable Energy*, 41:44–63, May 2012.

[13] Sébastien Gueydon and Wei Xu. Floating wind turbine motion assessment. In *OCEANS 2011*, pages 1–10. IEEE, 2011.

[14] G. Colicchio, M. Greco, and O. M. Faltinsen. A BEM-level set domain-decomposition strategy for non-linear and fragmented interfacial flows. *International Journal for Numerical Methods in Engineering*, 67(10):1385–1419, September 2006.

[15] Philip L-F Liu, Yong-Sik Cho, Michael J Briggs, Utku Kanoglu, and Costas Emmanuel Synolakis. Runup of solitary waves on a circular island. *Journal of Fluid Mechanics*, 302:259–285, 1995.

[16] D. H. Peregrine. Long waves on a beach. *Journal of Fluid Mechanics*, 27:815–827, 1967.

[17] James T Kirby, Fengyan Shi, Babak Tehranirad, Jeffrey C Harris, and Stephan T Grilli. Dispersive tsunami waves in the ocean: Model equations and sensitivity to dispersion and coriolis effects. *Ocean Modelling*, 62:39–55, 2013.

[18] M. S. Longuet-Higgins and E. D. Cokelet. The deformation of steep surface waves on water. i. a numerical method of computation. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 350(1660):1–26, July 1976.

[19] Carlos Alberto Brebbia and Jose Dominguez. *Boundary elements: an introductory course*. WIT press, 1996.

[20] George Z Gu and Hsiang Wang. Numerical modeling for wave energy dissipation within porous submerged breakwaters of irregular cross section. *Coastal Engineering Proceedings*, 1(23), 1992.

[21] S. T. Grilli, J. Skourup, and I. A. Svendsen. An efficient boundary element method for nonlinear water waves. *Engineering Analysis with Boundary Elements*, 6(2):97–107, June 1989.

[22] Takumi Ohyama and Kazuo Nadaoka. Modeling the transformation of nonlinear waves passing over a submerged dike. *Coastal Engineering Proceedings*, 1(23), 1992.

[23] Mei-Ying Lin, Chin-Hoh Moeng, Wu-Ting Tsai, Peter P. Sullivan, and Stephen E. Belcher. Direct numerical simulation of wind-wave generation processes. *Journal of Fluid Mechanics*, 616:1–30, 2008.

[24] H. Liu and K. Kawachi. A numerical study of undulatory swimming. *Journal of Computational Physics*, 155(2):223–247, November 1999.

[25] N. Repalle, K. Thiagarajan, and M. Morris-Thomas. CFD simulation of wave run-up on a spar cylinder. In *16th Australasian Fluid Mechanics Conference (AFMC)*, pages 1091–1094, 2007.

[26] Phung Dang Hieu, Tanimoto Katsutoshi, and Vu Thanh Ca. Numerical simulation of breaking waves using a two-phase flow model. *Applied Mathematical Modelling*, 28(11):983–1005, November 2004.

[27] H. C. Yign Noh, H. C. Hong Sik Min, and Siegried Raasch. Large eddy simulation of the ocean mixed layer: The effects of wave breaking and langmuir circulation. *Journal of Physical Oceanography*, 34(4):720–735, April 2004.

[28] Pierre Lubin, Stphane Vincent, Stphane Abadie, and Jean-Paul Caltagirone. Three-dimensional large eddy simulation of air entrainment under plunging breaking waves. *Coastal engineering*, 53(8):631–655, 2006.

[29] Jianming Yang and Frederick Stern. Large-eddy simulation of breaking waves using embedded-boundary/level-set method. In *45th AIAA aerospace sciences meeting and exhibit*, pages 2007–1455, 2007.

[30] Erik Damgaard Christensen, Henrik Bredmose, and Erik Asp Hansen. *Transfer of Boussinesq waves to a Navier-Stokes solver*, volume 4, pages 917–926. American Society of Mechanical Engineers, 2009.

[31] Erik Damgaard Christensen, JH Jensen, and Stefan Mayer. Sediment transport under breaking waves. In *COASTAL ENGINEERING CONFERENCE*, volume 3, pages 2467–2480. ASCE AMERICAN SOCIETY OF CIVIL ENGINEERS, 2001.

[32] G. Wei and J. Kirby. Time-dependent numerical code for extended boussinesq equations. *Journal of Waterway, Port, Coastal, and Ocean Engineering*, 121(5):251–261, 1995, http://ascelibrary.org/doi/pdf/10.1061/

[33] P. Lin and P. L. F. Liu. Internal wave-maker for navier-stokes equations models. *Journal of waterway, port, coastal, and ocean engineering*, 125(4):207–215, 1999.

[34] N. Garcia, J. L. Lara, and I. J. Losada. 2-d numerical analysis of near-field flow at low-crested permeable breakwaters. *Coastal Engineering*, 51(10):991–1020, November 2004.

[35] J. L. Lara, N. Garcia, and I. J. Losada. RANS modelling applied to random wave interaction with submerged permeable structures. *Coastal Engineering*, 53(56):395–417, April 2006.

[36] P. Lin and S. Karunarathna. Numerical study of solitary wave interaction with porous breakwaters. *Journal of Waterway, Port, Coastal, and Ocean Engineering*, 133(5):352–363, 2007.

[37] Ge Wei, James T. Kirby, and Amar Sinha. Generation of waves in boussinesq models using a source function method. *Coastal Engineering*, 36(4):271–299, May 1999.

[38] Junwoo Choi and Sung Bum Yoon. Numerical simulations using momentum source wave-maker applied to RANS equation model. *Coastal Engineering*, 56:1043–1060, 2009.

[39] Xin Guo and Lian Shen. On the generation and maintenance of waves and turbulence in simulations of free-surface turbulence. *Journal of Computational Physics*, 228(19):7313–7332, October 2009.

[40] Moshe Israeli and Steven A. Orszag. Approximation of radiation boundary conditions. *Journal of Computational Physics*, 41(1):115–135, 1981.

[41] M. W. Gee, U. Kttler, and W. A. Wall. Truly monolithic algebraic multigrid for fluidstructure interaction. *International Journal for Numerical Methods in Engineering*, 85(8):987–1016, 2011.

[42] Andrew T. Barker and Xiao-Chuan Cai. Scalable parallel methods for monolithic coupling in fluidstructure interaction with application to blood flow modeling. *Journal of Computational Physics*, 229(3):642–659, February 2010.

[43] Annalisa Quaini Santiago Badia. Modular vs. non-modular preconditioners for fluidstructure systems with large added-mass effect. *Computer Methods in Applied Mechanics and Engineering*, pages 4216–4232, 2008.

[44] Bjrn Hbner, Elmar Walhorn, and Dieter Dinkler. A monolithic approach to fluidstructure interaction using space-time finite elements. *Computer Methods in Applied Mechanics and Engineering*, 193(23-26):2087–2104, June 2004.

[45] Fotis Sotiropoulos and Xiaolei Yang. Immersed boundary methods for simulating fluid-structure interaction. *Progress in Aerospace Sciences*, 2013.

[46] Yue Yu, Hyoungsu Baek, and George Em Karniadakis. Generalized fictitious methods for fluidstructure interactions: Analysis and simulations. *Journal of Computational Physics*, 245:317–346, July 2013.

[47] J. Yang, S. Preidikman, and E. Balaras. A strongly coupled, embedded-boundary method for fluidstructure interactions of elastically mounted rigid bodies. *Journal of Fluids and Structures*, 24(2):167–182, February 2008.

[48] Jean Donea, Antonio Huerta, J.-Ph. Ponthot, and A. Rodrguez-Ferran. Arbitrary LagrangianEulerian methods. In *Encyclopedia of Computational Mechanics*. John Wiley & Sons, Ltd, 2004.

[49] Karl W. Schulz and Yannis Kallinderis. Unsteady flow structure interaction for incompressible flows using deformable hybrid grids. *Journal of Computational Physics*, 143(2):569–597, July 1998.

[50] Q Zhu, MJ Wolfgang, DKP Yue, and MS Triantafyllou. Three-dimensional flow structures and vorticity control in fish-like swimming. *Journal of Fluid Mechanics*, 468:1–28, 2002.

[51] Mehmet Sahin and Kamran Mohseni. An arbitrary lagrangian-eulerian formulation for the numerical simulation of flow patterns generated by the hydromedusa aequorea victoria. *J. Comput. Phys.*, 228(12):4588–4605, July 2009.

[52] Balasubramaniam Ramaswamy and Mutsuto Kawahara. Arbitrary LagrangianEulerianc finite element method for unsteady, convective, incompressible viscous free surface fluid flow. *International Journal for Numerical Methods in Fluids*, 7(10):1053–1075, 1987.

[53] W. P. Breugem and B. J. Boersma. Direct numerical simulations of turbulent flow over a permeable wall using a direct and a continuum approach. *Physics of fluids*, 17:025103, 2005.

[54] V. Roubtsova and R. Kahawita. The SPH technique applied to free surface flows. *Computers & Fluids*, 35(10):1359–1371, December 2006.

[55] A. Souto-Iglesias, L. Delorme, L. Prez-Rojas, and S. Abril-Prez. Liquid moment amplitude assessment in sloshing type problems with smooth particle hydrodynamics. *Ocean Engineering*, 33(1112):1462–1484, August 2006.

[56] A. Larese, R. Rossi, E. Oate, and S. R. Idelsohn. Validation of the particle finite element method (PFEM) for simulation of free surface flows. *Engineering Computations*, 25(4):385–425, 2008.

[57] S.r. Idelsohn, E. Oate, and F. Del Pin. The particle finite element method: a powerful tool to solve incompressible flows with free-surfaces and breaking waves. *International Journal for Numerical Methods in Engineering*, 61(7):964–989, 2004.

[58] Charles S. Peskin. The immersed boundary method. *Acta Numerica*, 11:479–517, 2002.

[59] D. K. Clarke, H. A. Hassan, and M. D. Salas. Euler calculations for multielement airfoils using cartesian grids. *AIAA Journal*, 24(3):353–358, March 1986.

[60] Ann S. Almgren, John B. Bell, Phillip Colella, and Tyler Marthaler. A cartesian grid projection method for the incompressible euler equations in complex geometries. *SIAM Journal on Scientific Computing*, 18(5):1289–1309, 1997.

[61] Stphane Popinet. Gerris: a tree-based adaptive solver for the incompressible euler equations in complex geometries. *Journal of Computational Physics*, 190(2):572–600, 2003.

[62] M. P. Kirkpatrick, S. W. Armfield, and J. H. Kent. A representation of curved boundaries for the solution of the NavierStokes equations on a staggered three-dimensional cartesian grid. *Journal of Computational Physics*, 184(1):1–36, 2003.

[63] Randall J. Leveque and Zhilin Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis*, 31(4):1019–1044, 1994.

[64] Sheng Xu and Z. Jane Wang. An immersed interface method for simulating the interaction of a fluid with moving boundaries. *Journal of Computational Physics*, 216(2):454–493, 2006.

[65] D. V. Le, B. C. Khoo, and J. Peraire. An immersed interface method for viscous incompressible flows involving rigid and flexible boundaries. *Journal of Computational Physics*, 220(1):109–138, 2006.

[66] J. Mohd-Yusof. Combined immersed-boundary/B-spline methods for simulations of ow in complex geometries. *Annual Research Briefs. NASA Ames Research Center= Stanford University Center of Turbulence Research: Stanford*, pages 317–327, 1997.

[67] Iman Borazjani, Fotis Sotiropoulos, Eric D. Tytell, and George V. Lauder. Hydrodynamics of the bluegill sunfish c-start escape response: three-dimensional simulations and comparison with experimental data. *The Journal of Experimental Biology*, 215(4):671–684, February 2012. PMID: 22279075.

[68] Iman Borazjani, Fotis Sotiropoulos, Edwin Malkiel, and Joseph Katz. On the role of copepod antennae in the production of hydrodynamic force during hopping. *The Journal of Experimental Biology*, 213(17):3019–3035, 2010.

[69] I Borazjani and F Sotiropoulos. On the role of form and kinematics on the hydrodynamics of self-propelled body/caudal fin swimming. *The Journal of Experimental Biology*, 213(1):89–107, 2010.

[70] Trung Bao Le, Iman Borazjani, Seokkoo Kang, and Fotis Sotiropoulos. On the structure of vortex rings from inclined nozzles. *Journal of Fluid Mechanics*, 686:451–483, 2011.

[71] Suresh Behara, Iman Borazjani, and Fotis Sotiropoulos. Vortex-induced vibrations of an elastically mounted sphere with three degrees of freedom at re = 300: hysteresis and vortex shedding modes. *Journal of Fluid Mechanics*, 686:426–450, 2011.

[72] E.A. Fadlun, R. Verzicco, P. Orlandi, and J. Mohd-Yusof. Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. *Journal of Computational Physics*, 161(1):35–60, June 2000.

[73] Jianming Yang and Elias Balaras. An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries. *Journal of Computational Physics*, 215(1):12–40, 2006.

[74] Yu-Heng Tseng and Joel H. Ferziger. A ghost-cell immersed boundary method for flow in complex geometry. *Journal of Computational Physics*, 192(2):593–623, 2003.

[75] Ali Khosronejad, Seokkoo Kang, Iman Borazjani, and Fotis Sotiropoulos. Curvilinear immersed boundary method for simulating coupled flow and bed morphodynamic interactions due to sediment transport phenomena. *Advances in Water Resources*, 34(7):829–843, July 2011.

[76] Seokkoo Kang, Iman Borazjani, Jonathan A. Colby, and Fotis Sotiropoulos. Numerical simulation of 3D flow past a real-life marine hydrokinetic turbine. *Advances in Water Resources*, 39:33–43, April 2012.

[77] Iman Borazjani, Liang Ge, Trung Le, and Fotis Sotiropoulos. A parallel overset-curvilinear-immersed boundary framework for simulating complex 3D incompressible flows. *Computers & Fluids*, 77:76–96, April 2013.

[78] Rajat Mittal and Gianluca Iaccarino. Immersed boundary methods. *Annual Review of Fluid Mechanics*, 37(1):239–261, 2005.

[79] Anvar Gilmanov and Fotis Sotiropoulos. A hybrid cartesian/immersed boundary method for simulating flows with 3D, geometrically complex, moving bodies. *Journal of Computational Physics*, 207(2):457–492, August 2005.

[80] Anvar Gilmanov and Sumanta Acharya. A hybrid immersed boundary and material point method for simulating 3D fluid-structure interaction problems. *International Journal for Numerical Methods in Fluids*, 56(12):2151–2177, April 2008.

[81] Anvar Gilmanov, Sumanta Acharya, and Timur Gilmanov. Flow–structure interaction simulations for ballutes in supersonic flow. *AIAA Paper*, 2906, 2009.

[82] K.M.T. Kleefsman, G. Fekken, A.E.P. Veldman, B. Iwanowski, and B. Buchner. A volume-of-fluid based simulation method for wave impact problems. *Journal of Computational Physics*, 206(1):363–393, June 2005.

[83] S. Haeri and J. S. Shrimpton. On the application of immersed boundary, fictitious domain and body-conformal mesh methods to many particle multiphase flows. *International Journal of Multiphase Flow*, 40:3855, 2012.

[84] Elias Balaras. Modeling complex boundaries using an external force field on fixed cartesian grids in large-eddy simulations. *Computers & Fluids*, 33(3):375–404, 2004.

[85] Ming-Chih Lai and Charles S. Peskin. An immersed boundary method with formal second-order accuracy and reduced numerical viscosity. *Journal of Computational Physics*, 160(2):705–719, May 2000.

[86] Linwei Shen, Eng-Soon Chan, and Pengzhi Lin. Calculation of hydrodynamic forces acting on a submerged moving object using immersed boundary method. *Computers & Fluids*, 38(3):691–702, March 2009.

[87] J. Sanders, J. E. Dolbow, P. J. Mucha, and T. A. Laursen. A new method for simulating rigid body motion in incompressible two-phase flow. *International Journal for Numerical Methods in Fluids*, 67(6):713–732, 2010.

[88] Iman Borazjani. *Numerical Simulations of Fluid-Structure Interaction Problems in Biological Flows*. PhD thesis, University of Minnesota, 2008.

[89] Salih Ozen Unverdi and Grtar Tryggvason. A front-tracking method for viscous, incompressible, multi-fluid flows. *Journal of Computational Physics*, 100(1):25–37, May 1992.

[90] Tao Ye, Wei Shyy, and Jacob N. Chung. A fixed-grid, sharp-interface method for bubble dynamics and phase change. *Journal of Computational Physics*, 174(2):781–815, December 2001.

[91] James Edward Pilliod Jr. and Elbridge Gerry Puckett. Second-order accurate volume-of-fluid algorithms for tracking material interfaces. *Journal of Computational Physics*, 199(2):465–502, September 2004.

[92] A. Iafrati. Numerical study of the effects of the breaking intensity on wave breaking flows. *Journal of Fluid Mechanics*, 622:371, February 2009.

[93] Wusi Yue, Ching-Long Lin, and Virendra C. Patel. Numerical simulation of unsteady multidimensional free surface motions by level set method. *International Journal for Numerical Methods in Fluids*, 42(8):853–884, 2003.

[94] H. Liu, S. Krishnan, S. Marella, and H.S. Udaykumar. Sharp interface cartesian grid method II: a technique for simulating droplet interactions with surfaces of arbitrary shape. *Journal of Computational Physics*, 210(1):32–54, November 2005.

[95] P. M. Carrica, R. V. Wilson, and F. Stern. An unsteady single-phase level set method for viscous free surface flows. *International Journal for Numerical Methods in Fluids*, 53(2):229–256, 2007.

[96] Seokkoo Kang and Fotis Sotiropoulos. Numerical modeling of 3D turbulent free surface flow in natural waterways. *Advances in Water Resources*, 40:23–36, May 2012.

[97] J. A. Sethian and Peter Smereka. Level set methods for fluid interfaces. *Annual Review of Fluid Mechanics*, 35(1):341–372, 2003.

[98] Mark Sussman and Elbridge Gerry Puckett. A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows. *Journal of Computational Physics*, 162(2):301–337, August 2000.

[99] Xiaofeng Yang, Ashley J. James, John Lowengrub, Xiaoming Zheng, and Vittorio Cristini. An adaptive coupled level-set/volume-of-fluid interface capturing method for unstructured triangular grids. *Journal of Computational Physics*, 217(2):364–394, September 2006.

[100] S. Tanaka and K. Kashiyama. ALE finite element method for FSI problems with free surface using mesh re-generation method based on background mesh. *International Journal of Computational Fluid Dynamics*, 20(3-4):229–236, March 2006.

[101] Kwang Paik. *Simulation of fluid-structure interaction for surface ships with linear/nonlinear deformations*. PhD thesis, University of Iowa, January 2010.

[102] Linwei Shen and Eng-Soon Chan. Numerical simulation of fluidstructure interaction using a combined volume of fluid and immersed boundary method. *Ocean Engineering*, 35(89):939–952, June 2008.

[103] E. Walhorn, A. Klke, B. Hbner, and D. Dinkler. Fluidstructure coupling within a monolithic model involving free surface flows. *Computers & structures*, 83(25):2100–2111, 2005.

[104] P. B. Ryzhakov, R. Rossi, S. R. Idelsohn, and E. Oate. A monolithic lagrangian approach for fluidstructure interaction problems. *Computational Mechanics*, 46(6):883–899, August 2010.

[105] Kwang Hyun Lee. *Responses of floating wind turbines to wind and wave excitation.* Thesis, Massachusetts Institute of Technology, 2005. Thesis (S.M.)–Massachusetts Institute of Technology, Dept. of Ocean Engineering, 2005.

[106] Elizabeth N. Wayman. *Coupled dynamics and economic analysis of floating wind turbine systems.* PhD thesis, Massachusetts Institute of Technology, 2006.

[107] Finn Gunnar Nielsen, Tor David Hanson, and Bjorn Skaare. Integrated dynamic analysis of floating offshore wind turbines. In *25th International Conference on Offshore Mechanics and Arctic Engineering*, pages 671–679. American Society of Mechanical Engineers, 2006.

[108] Thomas Zambrano, Tyler MacCready, Taras Kiceniuk Jr, Dominique G. Roddier, and Christian A. Cermelli. Dynamic modeling of deepwater offshore wind turbine structures in gulf of mexico storm condition. In *Proceedings of OMAE2006*, Hamburg, Germany, 2006.

[109] Jason M. Jonkman. Dynamics of offshore floating wind turbinesmodel development and verification. *Wind Energy*, 12(5):459–492, 2009.

[110] Di Yang, Charles Meneveau, and Lian Shen. Large-eddy simulation of offshore wind farm. *Physics of Fluids (1994-present)*, 26(2), February 2014.

[111] Di Yang, Charles Meneveau, and Lian Shen. Effect of downwind swells on offshore wind energy harvesting a large-eddy simulation study. *Renewable Energy*, May 2014.

[112] Di Yang and Lian Shen. Simulation of viscous flows with undulatory boundaries: Part II. coupling with other solvers for two-fluid computations. *Journal of Computational Physics*, 230(14):5510–5531, June 2011.

[113] Niels N Sørensen and Martin OL Hansen. Rotor performance predictions using a navier-stokes method. *AIAA paper*, 98:0025, 1998.

[114] N. N. Srensen, J. A. Michelsen, and S. Schreck. NavierStokes predictions of the NREL phase VI rotor in the NASA ames 80 ft 120 ft wind tunnel. *Wind Energy*, 5(2-3):151–169, April 2002.

[115] Jeppe Johansen, NN Sorensen, JA Michelsen, and S Schreck. Detached-eddy simulation of flow around the nrel phase-vi blade. In *ASME 2002 Wind Energy Symposium*, pages 106–114. American Society of Mechanical Engineers, 2002.

[116] Nilay Sezer-Uzol and Lyle N. Long. 3-d time-accurate CFD simulations of wind turbine rotor flow fields. *AIAA paper*, 394:2006, 2006.

[117] Nilay Sezer-Uzol, Ankur Gupta, and Lyle N. Long. 3-d time-accurate inviscid and viscous CFD simulations of wind turbine rotor flow fields. In *Parallel Computational Fluid Dynamics 2007*, number 67 in Lecture Notes in Computational Science and Engineering, pages 457–464. Springer Berlin Heidelberg, January 2009.

[118] Frederik Zahle, Niels N. Srensen, and Jeppe Johansen. Wind turbine rotor-tower interaction using an incompressible overset grid method. *Wind Energy*, 12(6):594–619, September 2009.

[119] Y. Bazilevs, M. C. Hsu, I. Akkerman, S. Wright, K. Takizawa, B. Henicke, T. Spielman, and T. E. Tezduyar. 3D simulation of wind turbine rotors at full scale. part i: geometry modeling and aerodynamics. *International Journal for Numerical Methods in Fluids*, 65(1-3):207–235, 2011.

[120] Y. Bazilevs, M. C. Hsu, J. Kiendl, R. Wchner, and K. U. Bletzinger. 3D simulation of wind turbine rotors at full scale. part II: fluidstructure interaction modeling with composite blades. *International Journal for Numerical Methods in Fluids*, 65(1-3):236–253, 2011.

[121] Seokkoo Kang, Xiaolei Yang, and Fotis Sotiropoulos. On the onset of wake meandering for an axial flow turbine in a turbulent open channel flow. *Journal of Fluid Mechanics*, 744:376–403, April 2014.

[122] Seokkoo Kang, Anne Lightbody, Craig Hill, and Fotis Sotiropoulos. High-resolution numerical simulation of turbulence in natural waterways. *Advances in Water Resources*, 34(1):98–113, January 2011.

[123] J. N. Srensen, W. Z. Shen, and X. Munduate. Analysis of wake states by a full-field actuator disc model. *Wind Energy*, 1(2):73–88, December 1998.

[124] Christian Masson, Arezki Smali, and Christophe Leclerc. Aerodynamic analysis of HAWTs operating in unsteady conditions. *Wind Energy*, 4(1):1–22, January 2001.

[125] Marc Calaf, Charles Meneveau, and Johan Meyers. Large eddy simulation study of fully developed wind-turbine array boundary layers. *Physics of Fluids (1994-present)*, 22(1):015110, January 2010.

[126] Xiaolei Yang, Seokkoo Kang, and Fotis Sotiropoulos. Computational study and modeling of turbine spacing effects in infinite aligned wind farms. *Physics of Fluids*, 24(11):115107–115107–28, November 2012.

[127] Yu-Ting Wu and Fernando Port-Agel. Large-eddy simulation of wind-turbine wakes: Evaluation of turbine parametrisations. *Boundary-Layer Meteorology*, 138(3):345–366, March 2011.

[128] Jens Norkr Sorensen and Wen Zhong Shen. Numerical modeling of wind turbine wakes. *Journal of Fluids Engineering*, 124(2):393–399, May 2002.

[129] B. Sanderse, S.p. van der Pijl, and B. Koren. Review of computational fluid dynamics for wind turbine wake aerodynamics. *Wind Energy*, 14(7):799–819, 2011.

[130] Massimo Germano, Ugo Piomelli, Parviz Moin, and William H. Cabot. A dynamic subgridscale eddy viscosity model. *Physics of Fluids A: Fluid Dynamics*, 3(7):1760–1765, July 1991.

[131] Stanley J. Osher and Ronald P. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2003 edition, October 2002.

[132] A. Iafrati and E. F. Campana. Free-surface fluctuations behind microbreakers: spacetime behaviour and subsurface flow field. *Journal of Fluid Mechanics*, 529:311–347, 2005.

[133] Mark Sussman and Emad Fatemi. An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. *SIAM J. Sci. Comput.*, 20(4):1165–1191, February 1999.

[134] Guang-Shan Jiang and Chi-Wang Shu. Efficient implementation of weighted ENO schemes. *Journal of Computational Physics*, 126(1):202–228, June 1996.

[135] Mark Sussman, Emad Fatemi, Peter Smereka, and Stanley Osher. An improved level set method for incompressible two-phase flows. *Computers & Fluids*, 27(56):663–680, June 1998.

[136] S. Armfield and R. Street. An analysis and comparison of the time accuracy of fractional-step methods for the NavierStokes equations on staggered grids. *International Journal for Numerical Methods in Fluids*, 38(3):255–282, January 2002.

[137] M. N. Huxley. Exponential sums and lattice points III. *Proceedings of the London Mathematical Society*, 87(03):591–609, 2003.

[138] A. Ivic, E. Krtzel, M. Khleitner, and W. G. Nowak. Lattice points in large regions and related arithmetic functions: Recent developments in a very classic topic. *arXiv preprint math/0410522*, 2004.

[139] Johannes Gualtherus van der Corput. *Over roosterpunten in het platte vlak:(de beteekenis van de methoden van Voronoï en Pfeiffer)*. PhD thesis, P. Noordhoff, 1919.

[140] Iman Borazjani and Fotis Sotiropoulos. Vortex induced vibrations of two cylinders in tandem arrangement in the proximity-wake interference region. *Journal of fluid mechanics*, 621:321–364, 2009. PMID: 19693281.

[141] W. Cabot and P. Moin. Approximate wall boundary conditions in the large-eddy simulation of high reynolds number flow. *Flow, Turbulence and Combustion*, 63(1-4):269–291, 2000.

[142] Meng Wang and Parviz Moin. Dynamic wall modeling for large-eddy simulation of complex turbulent flows. *Physics of Fluids*, 14(7), 2002.

[143] Jung-Il Choi, Roshan C. Oberoi, Jack R. Edwards, and Jacky A. Rosati. An immersed boundary method for complex incompressible flows. *Journal of Computational Physics*, 224(2):757–784, June 2007.

[144] Bruce M. Irons and Robert C. Tuck. A version of the aitken accelerator for computer iteration. *International Journal for Numerical Methods in Engineering*, 1(3):275–277, 1969.

[145] Xiaolei Yang, Fotis Sotiropoulos, Robert J Conzemius, John N Wachtler, and Mike B Strong. Large-eddy simulation of turbulent flow past wind turbines/farms: the virtual wind simulator (vwis). *Wind Energy*, 2014.

[146] V. E. Zakharov. Stability of periodic waves of finite amplitude on the surface of a deep fluid. *Journal of Applied Mechanics and Technical Physics*, 9(2):190–194, 1968.

[147] Douglas G Dommermuth and Dick KP Yue. A high-order spectral method for the study of nonlinear gravity waves. *Journal of Fluid Mechanics*, 184:267–288, 1987.

[148] C. Mei Chiang, Michael Stiassnie, and Dick K.-P. Yue. *Theory and Applications of Ocean Surface Waves:Part 1: Linear AspectsPart 2: Nonlinear Aspects: 23*. WSPC, expanded ed edition edition, July 2005.

[149] Di Yang, Charles Meneveau, and Lian Shen. Dynamic modelling of sea-surface roughness for large-eddy simulation of wind over ocean wavefield. *Journal of Fluid Mechanics*, 726:62–99, 2013.

[150] E. Bou-Zeid, C. Meneveau, and M. Parlange. A scale-dependent Lagrangian dynamic model for large eddy simulation of complex turbulent flows. *Physics of Fluids*, 17(2), 2005.

[151] Pengzhi Lin. A fixed-grid model for simulation of a moving body in free surface flows. *Computers & Fluids*, 36(3):549–561, March 2007.

[152] M. Greenhow and S. Moyo. Water entry and exit of horizontal circular cylinders. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, pages 551–563, 1997.

[153] Soichi Ito. *Study of the transient heave oscillation of a floating cylinder.* PhD thesis, MIT, 1971.

[154] A. Iafrati, A. Babanin, and M. Onorato. Modulational instability, wave breaking, and formation of large-scale dipoles in the atmosphere. *Physical Review Letters*, 110(18), 2013.

[155] K. H. Jung, K. A. Chang, and H. J. Jo. Viscous effect on the roll motion of a rectangular structure. *Journal of engineering mechanics*, 132(2):190–200, 2006.

[156] E. M. Yettou, A. Desrochers, and Y. Champoux. Experimental study on the water impact of a symmetrical wedge. *Fluid dynamics research*, 38(1):47–66, 2006.

[157] L. Xu, A. W. Troesch, and R. Peterson. Asymmetric hydrodynamic impact and dynamic response of vessels. *Journal of Offshore Mechanics and Arctic Engineering*, 121(2):83–89, May 1999.

[158] GX Wu, H Sun, and YS He. Numerical simulation and experimental study of water entry of a wedge in free fall motion. *Journal of Fluids and Structures*, 19(3):277–289, 2004.

[159] Simon G Lewis, Dominic A Hudson, Stephen R Turnock, and Dominic J Taunton. Impact of a free-falling wedge with water: synchronized visualization, pressure and acceleration measurements. *Fluid Dynamics Research*, 42(3):035509, June 2010.

[160] M. Greenhow. Wedge entry into initially calm water. *Applied Ocean Research*, 9(4):214–223, 1987.

[161] Xiaoming Mei, Yuming Liu, and Dick K.P. Yue. On the water impact of general two-dimensional sections. *Applied Ocean Research*, 21(1):1–15, 1999.

[162] R Zhao, O Faltinsen, and J Aarsnes. Water entry of arbitrary two-dimensional sections with and without flow separation. In *21st symposium on naval hydrodynamics*, 1997.

[163] M Doring, G Alessandrini, and P Ferrant. Sph simulationsof floating bodies in waves. In *Proceedings of the 19th International Workshop on Water Waves and Floating Bodies*, 2004.

[164] JCR Hunt, AA Wray, and P Moin. Eddies, stream, and convergence zones in turbulent flows. *Center For Turbulence Research*, Report CTR-S88, 1988.

[165] Chris Feist, Antoni Calderer, Kelley Ruehl, Fotis Sotiropoulos, and Michele Guala. Platform kinematics and wake evolution of a floating wind turbine: a quasi coupled wind-wave experimental study. *Journal of Fluid Mechanics, Submitted*, 2015.

[166] D Todd Griffith and Thomas D Ashwill. The sandia 100-meter all-glass baseline wind turbine blade: Snl100-00. *Sandia National Laboratories, Albuquerque, Report No. SAND2011-3779*, 2011.

[167] Antoni Calderer, Seokkoo Kang, and Fotis Sotiropoulos. Level set immersed boundary method for coupled simulation of air/water interaction with complex floating structures. *Journal of Computational Physics*, 277:201–227, 2014.

[168] AQWA ANSYS. Version 15.0; ansys. *Inc.: Canonsburg, PA, USA November*, 2013.

[169] Kelley Ruehl, Carlos Michelen, Samuel Kanner, Michael Lawson, and Yi-Hsiang Yu. Preliminary verification and validation of wec-sim, an open-source wave energy

converter design tool. In *ASME 2014 33rd International Conference on Ocean, Offshore and Arctic Engineering*, pages V09BT09A040–V09BT09A040. American Society of Mechanical Engineers, 2014.

[170] Michael Lawson, Yi-Hsiang Yu, Ruehl Kelley, Carlos Michelen, et al. Development and demonstration of the wec-sim wave energy converter simulation tool. 2014.

[171] Wec-sim web page. http://wec-sim.github.io/WEC-Sim/, 2015. Accessed: 2015-10-22.

[172] Leonardo P Chamorro and Fernando Porté-Agel. A wind-tunnel investigation of wind-turbine wakes: boundary-layer turbulence effects. *Boundary-layer meteorology*, 132(1):129–149, 2009.

[173] Stephen B Pope. *Turbulent flows*. Cambridge university press, 2000.

[174] National data buouy center, 2015.

[175] A Robertson, J Jonkman, M Masciola, H Song, A Goupee, A Coulling, and C Luan. Definition of the semisubmersible floating system for phase ii of oc4. *Offshore Code Comparison Collaboration Continuation (OC4) for IEA Task*, 30, 2012.

[176] Christian Jensen, Lars Damkilde, and Ronnie Refstrup Pedersen. Numerical simulation of gyroscopic effects in ansys. *Aalborg University, Denmark*, 2011.

Appendix A

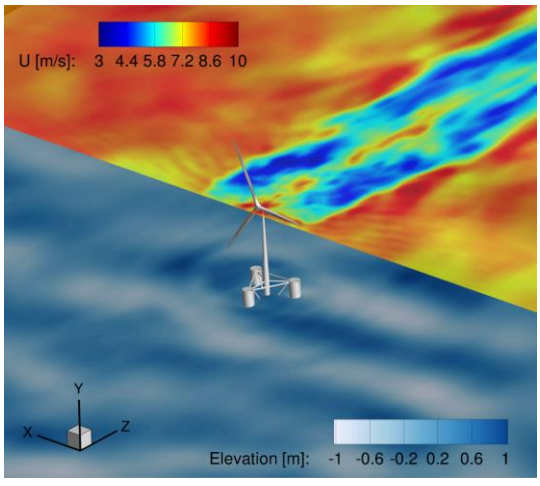
VFS-Wind User Manual

VFS

User Manual

Wind Version 1.0

October 30, 2015



University of Minnesota
Driven to Discover℠

Sandia National Laboratories

Acknowledgements

VFS is the result of many years of research work by several graduate students, post-docs, and research associates that have been involved in the Computational Hydrodynamics and Biofluids Laboratory directed by professor Fotis Sotiropoulos. The preparation of the present manual has been supported by the U.S. Department of Energy (DE-EE 0005482) and the University of Minnesota Initiative for Renewable Energy and the Environment (RM-0005-12).

Current Code Developers

Dennis Angelidis

Antoni Calderer

Anvar Gilmanov

Ali Khosronejad

Trung Le

Xiaolei Yang

Former Code Developers

Iman Borazjani

Liang Ge

Seokkoo Kang

Contributors to this manual

Antoni Calderer (Saint Anthony Falls Lab.)

Ann Dallman (Sandia National Laboratories)

D. Todd Griffith (Sandia National Laboratories)

Thomas Herges (Sandia National Laboratories)

Kelley Ruehl (Sandia National Laboratories)

License

Contents

Chapter 1

Introduction

Virtual Flow Simulator (VFS) is a three-dimensional (3D) incompressible Navier-Stokes solver based on the Curvilinear Immersed Boundary (CURVIB) method developed by Ge and Sotiropulos [1]. The CURVIB is a sharp interface type of immersed boundary (IB) method that enables the simulation of fluid flows with the presence of geometrically complex moving bodies. In IB approaches, the structural body mesh is superposed on the underlying Eulerian fluid mesh that is kept fixed. This approach circumvents the limitation of classical body fitted methods, in which the fluid mesh adapts to the body and thus limited to relatively simple geometries and small amplitude motions.

A particularity of the CURVIB method with respect to most sharp interface IB methods is that it is formulated in generalized curvilinear coordinates. This feature allows application of a body-fitted approach for the simpler boundaries while maintaining the ability to incorporate complex and moving geometries. For instance, in wind energy applications, one could take advantage of this feature when simulating the site specific geometry of a wind farm. The fluid mesh can follow the actual topography of the terrain while treating the turbines as immersed bodies.

VFS also integrates a two-phase flow solver module based on the level set method that allows simulation of coupled free surface flows with water waves, winds, and floating structures [2, 3]. This module was designed to simulate offshore floating wind turbines considering the non-linear effects of the free surface with a two-phase flow solver, the coupled 6 degrees of freedom (DoF) dynamics of the floating structure, and the ability to incorporate turbulent wind and wave conditions representative of realistic offshore environments.

The CURVIB method has been applied to a broad range of applications, such as cardiovascular flows [4, 5, 6], river bed morphodynamics [7, 8, 9], and wind and hydro-kinetic turbine simulations [10, 11]. For wind energy applications, a turbine can be resolved by immersing the geometry with the IB method or by using one of the different rotor parametrization models implemented.

The current version of the manual is for the VFS-Wind version of VFS. This version of the code includes the base solver and all the necessary libraries for simulating land based and offshore wind farms. The parts that are not included in this version

are the modules for sediment transport, bubbly flows, and elastic body deformations.

The organization of this user manual is as follows: in Chapter 2, the main governing equations and numerical methods employed by VFS are briefly described. In Chapter 3, the user is introduced to the basic steps to start using VFS. In Chapter 4, the source code organization is introduced with a description of the main functions in each module. In Chapter 5, the code input parameters are described. Finally, in Chapter 6, a series of application cases are documented.

Chapter 2

Overview of the Numerical Algorithms

2.1 The Flow Solver

The code solves the spatially-filtered Navier-Stokes equations governing incompressible flows of two immiscible fluids. The equations adopt a two-fluid formulation based on the level set method and are expressed in generalized curvilinear coordinates as follows ($i, j, k, l = 1, 2, 3$):

$$J\frac{\partial U^i}{\partial \xi^i} = 0, \tag{2.1}$$

$$\begin{aligned}\frac{1}{J}\frac{\partial U^j}{\partial t} &= \frac{\xi_l^i}{J}\left(-\frac{\partial}{\partial \xi_j}\left(U^j u_l\right) + \frac{1}{\rho\left(\phi\right)Re}\frac{\partial}{\partial \xi^j}\left(\mu\left(\phi\right)\frac{\xi_l^j \xi_l^k}{J}\frac{\partial u_l}{\partial \xi^k}\right) - \right. \\ &\left. -\frac{1}{\rho\left(\phi\right)}\frac{\partial}{\partial \xi^j}\left(\frac{\xi_l^j p}{J}\right) - \frac{1}{\rho\left(\phi\right)}\frac{\partial \tau_{lj}}{\partial \xi^j} - \frac{\kappa}{\rho(\phi)We^2}\frac{\partial h(\phi)}{\partial x_j} + \frac{\delta_{i2}}{Fr^2}\right),\end{aligned} \tag{2.2}$$

where ϕ is the level set function defined below, ξ^i are the curvilinear components, ξ_l^i are the transformation metrics, J is the Jacobian of the transformation, U^i are the contravariant volume fluxes, u_i are the Cartesian velocity components, ρ is the density, μ is the dynamic viscosity, p is the pressure, τ_{lj} is the sub-grid scale (SGS) tensor, κ is the curvature of the interface, δ_{ij} is the Kronecker delta, h is the smoothed Heaviside function, and Re, Fr, and We are the dimensionless Reynolds, Froude, and Weber numbers, respectively, which can be defined as:

$$Re = \frac{UL\rho_{water}}{\mu_{water}}, Fr = \frac{U}{\sqrt{gL}}, We = U\sqrt{\frac{\rho_{water}L}{\sigma}} \tag{2.3}$$

where U and L are the characteristic velocity and linear dimension, ρ_{water} and μ_{water}, the density and dynamic viscosity of the water phase, g the gravitational acceleration, and σ the surface tension.

The level set function ϕ is a signed distance function, adopting positive values on the water phase and negative values on the air phase. The density and viscosity are taken to be constant within each phase, and transition smoothly across the interface, which is smeared over a distance 2ϵ, as follows:

$$\rho(\phi) = \rho_{air} + (\rho_{water} - \rho_{air})\, h(\phi), \tag{2.4}$$

$$\mu(\phi) = \mu_{air} + (\mu_{water} - \mu_{air})\, h(\phi), \tag{2.5}$$

where $h(\phi)$ is the smoothed Heaviside function given in [12] and defined as:

$$h(\phi) = \begin{cases} 0 & \phi < -\epsilon, \\ \frac{1}{2} + \frac{\phi}{2\epsilon} + \frac{1}{2\pi}\sin(\frac{\pi\phi}{\epsilon}) & -\epsilon \leq \phi \leq \epsilon, \\ 1 & \epsilon < \phi. \end{cases} \tag{2.6}$$

Note that using the above equations, one can recover the single fluid formulation, by setting the density and viscosity to a constant value throughout the whole computational domain.

The free surface interface, given by the zero level of the distance function ϕ, can be modeled by solving the following level set equation proposed by Osher and Sethian [13]:

$$\frac{1}{J}\frac{\partial \phi}{\partial t} + U^j \frac{\partial \phi}{\partial \xi^j} = 0. \tag{2.7}$$

After solving the above level set advection equation, the distance function no longer maintains a unit gradient $|\nabla\phi| = 1$, which is a requirement to ensure conservation of mass between the two phases. To remedy this situation, the code solves the mass conserving re-initialization equation proposed by Sussman and Fatemi [14]. A detailed description of the method in the context of curvilinear coordinates can be found in Kang and Sotiropoulos [15].

The flow equations (2.1) and (2.2) are solved using the fractional step method of Ge and Sotiropoulos [1]. The momentum equations are discretized in space and time with a second-order central differencing scheme for the viscous, pressure gradient, and SGS terms, a second-order central differencing or a third-order WENO scheme for the advective terms, and the second-order Crank-Nicholson method for time advancement as follows:

$$\frac{1}{J}\frac{\mathbf{U}^* - \mathbf{U}^\mathbf{n}}{\triangle t} = \mathbf{P}(p^n, \phi^n) + \frac{1}{2}(\mathbf{F}(\mathbf{U}^*, \mathbf{u}^*, \phi^{n+1}) + (\mathbf{F}(\mathbf{U}^\mathbf{n}, \mathbf{u}^\mathbf{n}, \phi^n)), \tag{2.8}$$

where n indicates the previous time step, $\triangle t$ the time step size, $\mathbf{F}$ the right hand side of Eq.(2.2) excluding the pressure term, and $\mathbf{P}$ the pressure term. The continuity condition, discretized with three-point central differencing scheme, is enforced in the second stage of the fractional step method with the following pressure Poisson equation:

$$-J\frac{\partial}{\partial \xi^i}\left(\frac{1}{\rho(\phi)}\frac{\xi_l^i}{J}\frac{\partial}{\partial \xi^j}\left(\frac{\xi_l^j \Pi}{J}\right)\right) = \frac{1}{\triangle t} J \frac{\partial U^{j,*}}{\partial \xi^j}, \tag{2.9}$$

where Π is the pressure correction, used as follows, to update the pressure and the velocity field resulting from the first step of the fractional step method:

$$p^{n+1} = p^n + \Pi, \tag{2.10}$$

$$U^{i,n+1} = U^{i,*} - J\Delta t \frac{1}{\rho(\phi)} \frac{\xi_l^i}{J} \frac{\partial}{\partial \xi^j} \left(\frac{\xi_l^j \Pi}{J} \right), \tag{2.11}$$

The momentum equations are solved using an efficient matrix-free Newton-Krylov solver, and the Poisson equation with a generalized minimal residual (GMRES) method preconditioned with multi-grid.

The level set equations (2.7) are discretized with a third-order WENO scheme in space, and second-order Runge-Kutta in time. The re-initialization step uses a second-order ENO scheme. A detailed description of the method can be found in [15].

2.2 The CURVIB Method

The code can simulate flows around geometrically complex moving bodies with the sharp interface CURVIB method developed by Ge and Sotiropoulos [1]. The method has been thoroughly validated in many applications, such as fluid-structure interaction (FSI) problems [16, 17], river bed morphodynamics [7, 8, 9], and cardiovascular flows [4, 5, 6].

In the CURVIB method, the bodies are represented by an unstructured triangular mesh that is superposed on the background curvilinear or Cartesian fluid grid. First the nodes of the computational domain are classified depending on their location with respect to the position of the body. The nodes that fall inside the body are considered structural nodes and are blanked out from the computational domain. The nodes that are located in the fluid but in the immediate vicinity of the structure are denoted as IB nodes, where the boundary condition of the velocity field is reconstructed. The remaining nodes are the fluid nodes where the governing equations are solved.

The velocity reconstruction is performed in the wall normal direction with either linear or quadratic interpolation in the case of low Reynolds number flows when the IB nodes are located in the viscous sub-layer. While, the velocity reconstruction uses the wall models described by [18, 19, 20] in high Reynolds number flows when the grid resolution is not sufficient to accurately resolve the viscous sub-layer.

The distance function ϕ also needs to be reconstructed at the body-fluid interface. This is done by setting gradient $\Delta\phi$ to be zero at the cell faces that are located between the fluid and IB nodes as described in [15].

2.3 The Structural Solver and the Fluid-Structure Interaction Algorithm

The original FSI algorithm implementation for single phase flows is described in Borazjani, Ge, and Sotiropoulos [16], and the extension to two-phase free surface flows in Calderer, Kang, and Sotiropoulos [2].

The code solves the rigid body equations of motion (EoM) in 6 DoF, which can be written in Lagragian form and in principle axis as follows (i=1,2,..6),

$$M\frac{\partial^2 Y^i}{\partial t^2} + C\frac{\partial Y^i}{\partial t} + KY^i = F_f^i + F_e^i \tag{2.12}$$

where Y^i represents the coordinates of the Lagrangian vector describing the motion of the structure. For the translational DoFs, Y^i are the Cartesian components of the body position, M is the mass matrix, C is the damping coefficients matrix, K is the spring stiffness coefficient matrix, F_f^i are the forces exerted by the fluid, and F_e^i are the components of the external force vector. For the rotational DoFs, Y^i are relative angle components of the body, M represents the moment of inertia, and F_f^i and F_e^i are the moments around the rotation axis, induced by the fluid and by the external forces, respectively.

The forces and moments that the fluid exerts on the rigid body are computed by integrating the pressure and the viscous stresses along the surface Γ of the body as follows

$$F_f = \int_\Gamma -pnd\Gamma + \int_\Gamma \tau_{ij} n_j d\Gamma \tag{2.13}$$

$$M_f = \int_\Gamma -\epsilon_{ijk} r_j p n_k d\Gamma + \int_\Gamma \epsilon_{ijk} r_j \tau_{kl} n_l d\Gamma \tag{2.14}$$

where p denotes the pressure, τ the viscous stress, ϵ_{ijk} the permutation symbol, r the position vector, and n the normal vector.

The EoM (Eqs. 2.12) are coupled with the fluid domain equations through a partitioned FSI approach. The time integration can be done explicitly with loose coupling (LC-FSI), or implicitly with strong coupling (SC-FSI). The Aitken acceleration technique of [21] allows for significant reduction in the number of sub-iterations when the SC-FSI algorithm is used. A detailed description of both time-integration algorithms is given in [16].

2.4 Turbine Parameterizations

The actuator disk and actuator line models implemented in the code for parameterizing turbine rotors are given, respectively, in Yang, Kang, and Sotiropoulos [10] and in Yang et al. [22]. The basic idea of these models is to extract from the flow field the kinetic energy that is estimated to be equivalent to that from a turbine rotor, without the need to resolve the flow around its geometry. To introduce such kinetic energy

reduction into the flow, a sink term, affecting the fluid nodes located at the vicinity of the turbine, is considered in the right hand side of the momentum equations.

2.4.1 Actuator Disk Model

In the actuator disk model, the turbine rotor is represented by a circular disk that is discretized with an unstructured triangular mesh. The body force of the disk per unit area is the following

$$F_{AD} = -\frac{F_T}{\pi D^2/4}, \tag{2.15}$$

where D is the rotor diameter and F_T is the thrust force computed as

$$F_T = \frac{1}{2}\rho C_T \frac{\pi}{4} D^2 U_\infty^2, \tag{2.16}$$

where U_∞ is the turbine incoming velocity, $C_T = 4a(1-a)$ is the thrust coefficient taken from the one-dimensional momentum theory, and a is the induction factor. The incoming velocity U_∞ is also computed from the one-dimensional momentum theory as

$$U_\infty = \frac{u_d}{1-a}, \tag{2.17}$$

where u_d is the disk-averaged stream-wise velocity computed as

$$u_d = \frac{4}{\pi D^2} \sum_{N_t} u(X) A(X), \tag{2.18}$$

where N_t is the number of triangular elements composing the disk mesh, $A(X)$ is the area of each element, and $u(X)$ is the velocity at the element centers. The fluid velocity at the disk ($u(X)$) requires interpolation from the velocity values at the surrounding fluid mesh points, as the nodes from the fluid and disk meshes do not necessarily coincide. If we consider X to be the coordinates of the actuator disk nodes and x the coordinates of the fluid mesh nodes, the interpolation, which uses a discrete delta function, reads as follows

$$u(X) = \sum_{N_D} u(x) \delta_h(x - X) V(x), \tag{2.19}$$

where δ_h is a 3D discrete delta function, $V(x)$ is the volume of the corresponding fluid cell, and N_D is the number of fluid cells involved in the interpolation.

Finally, the body force F_{AD}, which is computed at the disk mesh nodes, needs to be distributed over the fluid cells located in the immediate vicinity using the following expression:

$$f_{AD}(x) = \sum_{N_D} F_{AD}(X) \delta_h(x - X) A(x). \tag{2.20}$$

2.4.2 Actuator Line Model

In the actuator line method, each blade of the rotor is modeled with a straight line, divided in several elements along the radial direction. In each of the elements, the lift (L) and drag (D) forces are computed using the following expressions:

$$L = \frac{1}{2}\rho C_L C V_{rel}^2, \tag{2.21}$$

$$D = \frac{1}{2}\rho C_D C V_{rel}^2, \tag{2.22}$$

where C_L, C_D are the lift and drag coefficients, respectively, taken from tabulated two-dimensional (2D) airfoil profile data, C is the chord length, and V_{ref} is the incoming reference velocity computed as

$$V_{rel} = (u_z, u_\theta - \Omega r) \tag{2.23}$$

where u_z and $u_\theta - \Omega r$ are the components of the velocity in the axial and azimuthal directions, respectively, Ω the angular velocity of the rotor, and r the distance to the center of the rotor.

To compute the reference velocity at the line elements, similarly to the actuator disk method, the velocity at the fluid mesh is transferred to the line elements using a discrete delta function as given by equation (2.19). Once the lift and drag forces are computed at each of the line elements, the distributed body force in the fluid mesh can be computed using the following equation:

$$f_{AL}(x) = \sum_{N_L} F(X)\delta_h(x - X)A(x). \tag{2.24}$$

where N_L is the number of segments composing one of the actuator lines, $F(X)$ is the projection of L and D, expressed in actuator line local coordinates, into Cartesian coordinates.

2.5 Large-Eddy Simulation

The description of the Large-eddy simulation (LES) model implemented in the code is extensively described in Kang et al. [23]. The sub-grid stress term in the right hand side of the momentum equation resulting from the the filtering operation is modeled with the Smagorinsky SGS model of [24] which reads as follows

$$\tau_{ij} - \frac{1}{3}\tau_{kk}\delta_{ij} = -2\mu_t \overline{S}_{ij}, \tag{2.25}$$

where μ_t is the eddy viscosity, the overline indicates the grid filtering operation, and $\overline{S_{ij}}$ is the large-scale strain-rate tensor. The eddy viscosity can be written as

$$\mu_t = C_s \Delta^2 |\overline{S}|, \tag{2.26}$$

where Δ is the filter width taken from the box filter, $|\overline{S}| = (2\overline{S}_{ij}\overline{S}_{ij})^{1/2}$ is the magnitude of strain-rate tensor, and C_s is the Smagorinsky constant computed dynamically with the Smagorinsky model of Germano et al. [25].

2.6 Wave Generation

The code uses an internal generation method as described in [3]. As illustrated in Figure 2.1, a surface force is applied at the so called source region, generating waves that propagate symmetrically to both stream-wise directions. A sponge layer method is used at the lateral boundaries to dissipate the waves and prevent reflections. The area between the source region and the sponge layer can be used to study body-wave interactions. Both the sponge layer force and the wave generation force are introduced in the code using a source term in the right hand side of the momentum equations.

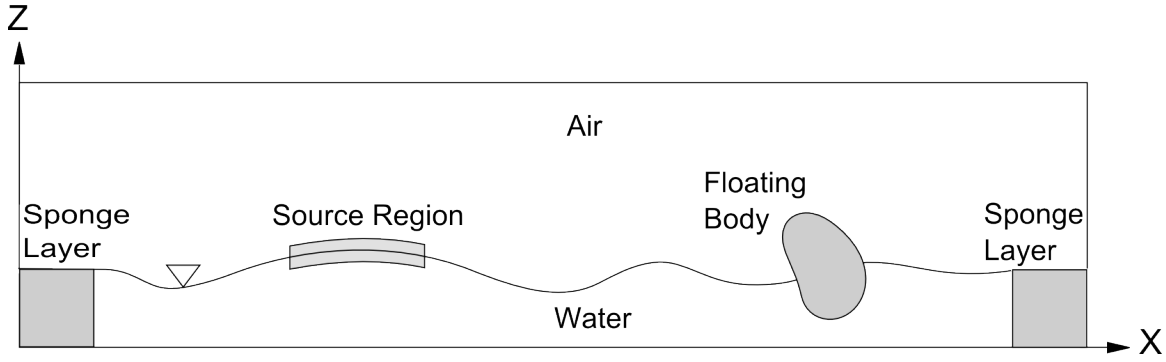


Figure 2.1: Schematic description of the wave generation method using the free surface forcing method [3].

To generate the following surface elevation,

$$\eta(x, y, t) = A\cos(k_x x + k_y y - \omega t + \theta), \tag{2.27}$$

where k_x and k_y are the components of the wavenumber vector K, θ is the wave phase, A is the wave amplitude, and ω the wave frequency, the forcing term reads as follows

$$S_i(x, y, t) = n_i(\phi)P_0\delta(x, \epsilon_x)\delta(\phi, \epsilon_\phi)sin(\omega t - k_y y - \theta), \tag{2.28}$$

where P_0 is a coefficient that depends on the wave and fluid characteristics,

$$P_0 = A\frac{g^2}{\omega^2}\frac{\epsilon}{f(\epsilon_x, k_x)}\frac{2\rho_w}{\rho_a + \rho_w}\frac{k_x}{\left(k_x^2 + k_y^2\right)^{1/2}}, \tag{2.29}$$

δ is a distribution function defined as:

$$\delta(\alpha, \beta) = \begin{cases} \frac{1}{2\beta}\left[1 + \cos\left(\frac{\pi\alpha}{\beta}\right)\right] & \text{if } -\beta < t < \beta \\ 0 & \text{otherwise.} \end{cases}, \tag{2.30}$$

and $f(\epsilon_x, k_x)$ is

$$f(\epsilon_x, k_x) = \frac{\pi^2}{k_x\left(\pi^2 - \epsilon_x^2 k_x^2\right)}\sin(k_x\epsilon_x). \tag{2.31}$$

With the present forcing method, wave fields with multiple components, such as a broadband wave spectrum, can be incorporated into the fluid domain. This method enables to simulate the interaction of floating structures with complex wave fields. The wave fields can either be originated in a far-field precursor simulation or taken from theoretical or measured data.

The sponge layer method for dissipating the waves at the boundaries reads as follows:

$$S_i(x,y,t) = -\left[\mu C_0 u_i + \rho C_1 u_i \left|u_i\right|\right] \frac{\exp\left[\left(\frac{x_s - x}{x_s}\right)^{n_s}\right] - 1}{\exp(1) - 1} \text{ for } (x_0 - x_s) \leq x \leq x_0, \tag{2.32}$$

where x_0 denotes the starting coordinate of the source region, x_s is the length of the source region, and C_0, C_1, and n_s are coefficients to be determined empirically. In [3], n_s is 10, C_0 is 200000, and C_1 is 1.0.

Chapter 3

Getting Started

In this chapter, we describe how to install the libraries required for VFS to work and how to run a simulation case.

3.1 Installing PETSC and Required Libraries

VFS is implemented in C and is parallelised using the Message Passing Interface (MPI). The Portable, Extensible Toolkit for Scientific Computation (PETSC) libraries are used for the code organization and to facilitate its parallel implementation. Also we use the library HYPRE for solving the Poisson equation. Before the code can be compiled, the following libraries need to be properly installed.

- PETSC version 3.1
- Blas and Lapack
- openmpi
- HYPRE

Note that when installing the PETSC libraries there is the option to automatically install the other required libraries in the case that they are not already in the computer. The PETSC web page [26] (http://www.mcs.anl.gov/petsc/documentation/installation.html) gives a detailed description on how to install all of these libraries. We briefly outline the steps for installing PETSC in the command line of a linux machine in the case that none of the aforementioned libraries have been previously installed:

1. Create a directory where to download the PETSC source files:
   mkdir source
2. Create the installation directory:
   mkdir system

3. Download the PETSC source code from the PETSC server in the source folder:
   wget http://ftp.mcs.anl.gov/pub/petsc/release-snapshots/petsc-3.1-p6.tar.gz

4. Unzip the PETSC source code in the source folder:
   tar xvfz petsc-3.1-p6.tar.gz

5. Execute the PETSC configuration script:
   config/configure.py --with-cc=mpicc --with-cxx=mpicxx --with-fc=mpif90 --download-f-blas-lapack=1 --download-mpich=1 --download-hypre=1 --prefix=installation_folder --with-shared=0 --with-debugging=0

6. If the previous action executes successfully, PETSC will print on the screen the subsequent steps.

   Note that PETSC can be installed with the debugging option either active or inactive. It is recommended that PETSC is installed without debugging because it will be used in production mode. The PETSC installation with debugging may be needed for developing purpose but it compromises the speed of the code.

3.2 Compiling the Code

To compile the code and generate an executable file we include a file named "makefile". This file is general and can work for any linux-based computer without being modified. It basically links all the source code files (*.c, *.h) and the necessary libraries (PETSC, HYPRE, etc). Given that in every computer the compiler libraries are located in different directories, the user has to create a new text file with the name: "makefile.local". This file is read by makefile and should contain the following user dependent information:

```
1 MPICXX       = mpicxx
2 HOME    = /Your_Home_Folder/
3 ACML    = /Your_ACML_Folder/
4 PETSC   = /Your_PETSC_Folder/
5 HYPRE   = /Your_HYPRE_Folder/
6 USE_TECPLOT=1 ##1 if tecplot is intalled, otherwise set to 0
7 TEC360HOME=/
```

If all libraries have been successfully installed and properly referenced in the file "makefile.local" the code should compile by typing the following command in the linux shell:

make

Alternatively, one can add the option -j to increase the computation speed by taking advantage of the several processors as follows:

make -j

Either of the last two commands will generate the executable source file "**vwis**".

In addition to the executable file for running the code, it is also necessary to compile the executable file for post-processing the resulting output data. The post-processing file can generate result files in both Tecplot format (plt) and Totalview format (vtk). The advantage of Totalview over Tecplot is that it is open source and can be download from the Totalview webpage.

3.2.1 Compiling the Post-Processing File for Totalview Only

This may be useful when Tecplot 360 is not installed in the computer. For creating the post-processing file which is only suitable for generating Totalview output files, first set the tecplot option in the file "makfile.local" to inactive as follows "USE_TECPLOT=0".

Then type the following command in the linux shell:

make data

3.2.2 Compiling the Post-Processing File for Tecplot and Totalview

If this is the case, in the previously described file "makfile.local", the tecplot option has to be active as follows "USE_TECPLOT=1".

In addition, one needs to download the library file "libtecio.a" from the tecplot library webpage (http://www.tecplot.com/downloads/tecio-library/) and add it to the code directory.

The post-processing file named "**data**" should then be created with the following command:

make data

3.3 Running VFS

3.3.1 File Structure Overview

All the files that are necessary for running VFS should be located in a user defined folder. The required input files are the following:

grid.dat or xyz.dat The structured mesh for the fluid domain.

bcs.dat The option file for setting the BCs of the fluid boundaries.

vwis Executable file obtained upon code compilation.

Submission script The linux script for submitting the job in a linux based cluster.

control.dat The file containing most of the control options.

ibmdata00, ibmdata01, etc. The mesh files for the immersed bodies, if any.

data The post-processing executable file

The fluid grid file, the immersed bodies grid files, the boundary conditions file and the control file are described extensively in Section 4.2.

The remainder of this chapter describes the compiling process for obtaining the executable vwis file, and the submission script for running the code.

3.3.2 Execute the Code

Each cluster may have different job resource manager systems although the most common is PBS. If it is not PBS, your system manager may provide instructions about the resource manager in use. There is ample documentation online as well.

In the case that your system uses the PBS system, the user can submit a job in the cue with the example script shown below:

```
1  #!/bin/bash
2  ### Job name
3  ### Mail to user
4  #PBS -k o
5  #PBS -l nodes=1:ppn=16,walltime=4:00:00
6  #PBS -j oe
7
8  cd $PBS_O_WORKDIR
9
10 /openmpi_directory/mpirun --bind-to-core  ./vwis>& err
```

In the script above, the job will use 1 node of 16 cpus per node (ppn). The job maximum duration will be 4 hours (walltime). The name of the file executed is "vwis", and the on screen information will be stored in "err" file.

The command for submitting this script is

qsub script_name.sh

One can check the status of the job,

showq

To finalize the job type

qdel job_id

3.3.3 Simulation Outputs

vfieldX.dat, ufieldX.dat, pfieldX.dat, nvfieldX.dat, lfieldX.dat, cs_X.dat

These are binary files containing the flow variables at the whole computational domain at a given time step indicated by "X". These files will be exported at every "tio" times steps, where "tio" is a control option. The content in each of these files is summarized in Table 3.1.

Table 3.1: Description of the instantaneous output results

File Name	Containing variable	Description of the Variable
vfield'x'.dat	Ucont	Contravarian velocity components (fluxes)
ufield'x'.dat	Ucat	Cartesian velocity components
pfield'x'.dat	P	Pressure field
nvfield'x'.dat	Nvert	If IB is used, it indicates the classification of nodes. 3 is structure node, 1 is IB node, and 0 is fluid node.
lfield'x'.dat	level	The distance function used in the levelset method to track the interface
cs_'x'.dat	Cs	The eddy viscosity coefficient when using LES.

Converge_dU

This text file contains the following information:

- The first number displayed in each of the lines is the time step number.
- Second column is the algorithm that the line refers to (momenum, poisson, levelset, IBMSERA0)
  - momentum: Momentum equation solver.
  - poisson: Poisson solver for the second step of the fractional step method.
  - levelset: If using the level set method it refers to the Reinitialization equation.
  - IBMSERA0: Refers to the searching algorithm for node classification when at least one immersed body is present.
- The third column is the computational time in seconds that it takes the algorithm to complete.
- The fourth column, if any, is the convergence of the corresponding solver. In the case of the Poisson solver the convergence is the maximum divergence and is indicated with "Maxdiv=".

Kinetic_Energy.dat

This file exports a text file with two columns. The first column displays the time, and the second column the total kinetic energy of the whole computational domain calculated as follows

$$KE = \sum_{i=0}^{N_i} \sum_{j=0}^{N_j} \sum_{k=0}^{N_k} u_{ijk}^2 + v_{ijk}^2 + w_{ijk}^2, \tag{3.1}$$

where N_i,N_j, and N_k is the number of grid nodes in the i, j, and k directions, respectively, and u_{ijk}, v_{ijk}, and w_{ijk}, are the Cartesian velocity components computed at the cell centers. The function **KE_Output** is responsible for creating this file.

FSI_position00, FSI_position01, etc

This is a text file containing information about the immersed body location, velocity and forces for the linear degrees of freedom in the x, y, and z direction. It consists of 10 columns as follows:

$$ti, Y_x, \dot{Y_x}, F_x, Y_y, \dot{Y_y}, F_y, Y_z, \dot{Y_z}, F_z, \tag{3.2}$$

where ti is the time step number, Y is the body position with respect to its initial position, $\dot{Y}$ is the body velocity, and F the force that the fluid imparts to the immersed body. If multiple immersed bodies are used, the code will print a file for each of the bodies, labeled with the body number.

FSI_Angle00, FSI_Angle01, etc.

This output file is similar to the FSI_position file, but for the rotational degrees of freedom. It contains information about the immersed body rotation angle and angular velocity of the body. It consists of 7 columns as follows:

$$ti, \Theta_x, \dot{\Theta}_x, \Theta_y, \dot{\Theta}_y, \Theta_z, \dot{\Theta}_z, \tag{3.3}$$

where ti is the time step number, Θ is the body rotation with respect to its starting position and $\dot{\Theta}$ is the body angular velocity. If multiple immersed bodies are used, the code will print a file for each of the bodies, labeled with the body number.

Force_Coeff_SI00

This text file contains information about the forces that the fluid imparts to the immersed body in the three linear degrees of freedom, x, y, and z. The file has 10 columns with the following data:

$$ti, F_x, F_y, F_z, Cp_x, Cp_y, Cp_z, Ap_x, Ap_y, Ap_z, \tag{3.4}$$

where ti is the time step number, F denotes the fluid force applied to the body, Cp the normalized force coefficient, and Ap the area of the body projected in

the corresponding direction which has been used for computing Cp. Again, a different file is exported for each additional immersed body.

Momt_Coeff_SI00
This file is equivalent to Force_Coeff_S00 but in the rotational degrees of freedom. The information exported is the following:

$$ti, M_x, M_y, M_z, \tag{3.5}$$

where ti is the time step number and M denotes the moments applied to the structure.

surface000_00_nf.dat, surface000_01_nf.dat, ...
These are tecplot ASCII files containing the surface of the corresponding IB body at the time step indicated at the first number of the file name. It basically contains the x, y, and z coordinates of the nodal points of the triangular mesh.

sloshing.dat
Data file with information about the free surface elevation in the center of the tank for the sloshing case. The first column is the simulation time [sec], the second is the computed surface elevation [m] at the tank center, the third is the expected theoretical solution at the tank center, and the final column is the error.

3.3.4 Post-Processing

Once VFS reaches a time step at which data are output (multiple of "tio", or output time step), the output file can be post-processed. Post-processing consists of converting the output files (ufieldxx.dat, vfieldxx.dat, pfieldxx.dat, nvfieldxx.dat, lfieldxx.dat, lfieldxx.dat, etc), which are in binary form, to a format which is readable for a visualization software such as TecPlot360 or Totalview.

Create plt Files for Tecplot360

To post-process the data, the executable "data" should be used with the following command:

mpiexec ./data -tis 0 -tie 50000 -ts 100

where -tis is the start timestep, -tie is the end timestep and -ts is the time step interval at which the files were generated.

The above step will generate n number of result files (for each timestep) compatible with tecplot with names similar to: Resultxx.plt where xx indicates the timestep. The .plt file can be opened in tecplot and worked upon.

The following webpage contains several tutorials in flash video on how to use tecplot: http://www.genias-graphics.de/cms/tp-360-tutorials.html

Create vtk Files for Totalview

If the user wants to visualize the files using Totalview, the option -vtk 1 should be included as follows

mpiexec ./data -tis 0 -tie 50000 -ts 100 -vtk 1

Averaged results

By default, the plt and vtk files contain instantaneous data results even if the code has performed averaging of the results (-averaging set to 1, 2, or 3 at the time that the code is executed). To include the averaged results in the post-processed file the option -avg needs to be activated when executing the file "data". The option -avg can be set to 1, 2, and 3, depending on the amount of information to be included in the post-processed file, having an impact on the overall file size. If -avg is set to 1, the post-processed file contains only averaged velocity and turbulence intensities (U, V, W, uu, vv, ww, wv, vw, uw); if it is set to 2, the post-processed file contains the same as the case for -avg 2 plus the averaged pressure and pressure fluctuations; if it is set to 3, the file contains the same variables as in -avg 2 plus averaged vorticity. Note that for post-processing the data using the options -avg 2 and 3, the code should have been executed using the option -averaging with a value equal or larger than the -avg value. An example on how to process averaged results is as follows:

mpiexec ./data -tis 0 -tie 50000 -ts 100 -avg 1

3.3.5 The Post-Processed File

Instantaneous Results

The variables of the post-processed results file, regardless of being Tecplot or Totalview formatted, are summarized in Table 3.2.

Table 3.2: Description of the instantaneous output results in the postprocessed file.

Variable	**Description**
X, Y, Z	Coordinates of the fluid grid
U, V, W	Velocity components at the grid cell centers
UU,	Velocity magnitude
P	Pressure field
Nvert	If IB is used, it indicates the classification of nodes. 3 is structure node, 1 is IB node, and 0 is fluid node.
level	The distance function used in the levelset method to track the interface

Averaged Results

The variables of the post-processed results file, regardless of being Tecplot or Totalview formatted, are summarized in Table 3.3.

Table 3.3: Description of the time averaged output results in the post-processed file.

Variable	**Description**
X, Y, Z	Coordinates of the fluid grid
U, V, W	Averaged velocity components at the grid cell centers
uu, vv, ww, uv, uw, vw	Turbulence intensities
P	Averaged pressure field
Nvert	If the IB method is used, it indicates the classification of nodes. 3 is structure node, 1 is IB node, and 0 is fluid node.
level	The distance function used in the levelset method to track the interface

Chapter 4

Code Input Parameters

4.1 Units

In terms of the units that the code uses, the user needs to differentiate between two cases, when the level set method is active (levelset option set to 1) and when it is not active (levelset option set to 0). In the case that the level set method is not active, the Navier-Stokes equations are solved in the dimensionless form. In this case, it is recommended that the reference length of the domain and the reference velocity of the flow are both equal to one.

In the case that the level set method is in use, the equations solved have real dimensions, and the units are in MKS (meters-kilograms-seconds system).

4.2 VFS Input Files

The VFS code requires several input files that have to be located by default at the cases directory. The input files are the following:

control.dat

bcs.dat

grid.dat

ibmdata00, ibmdata01, ibmdata02, etc. (if IB method is in use)

4.2.1 The control.dat File

The file control.dat is a text file that is read by the code upon initiation. It contains most of the input variables for the different modules of the code. The order in which the different options are included in the control file is not relevant, however it is recommended to group the options in categories as proposed below.

The control options start with a dash "-" symbol. If for any case the same option is introduced twice the code will take the value introduced latest in the file. If a control option wants to be kept in the control file for later use it can be commented by introducing a "!" sign in the start of the line.

Time Related Options

dt (double)

Time step size for the solution of the Navier-Stokes equations. The CFL number has to be smaller than 1 although values less than 0.5 are recommended. When using the level set method, the CFL number is more restrictive and in that case the CFL should be lower than 0.05.

tio (int)

The code exports the complete flow field data every time step that is a multiple of the value of this parameter.

totalsteps (int)

Total number of time steps to run before ending the simulation.

rstart (int)

This option is for restarting a simulation. The value given to this parameter is the time step number for which the simulation will restart. This option can only be used if results from a previous simulation are present in the folder. Note that even if the value is set to zero, the simulation will restart from a previous run, in that case, from a zero time step.

rs_fsi (int 0, 1)

This parameter can be used when restarting a simulation (rstart active) and FSI module is in use. If rs_fsi is set to 1, the code will read the file FSI_DATA corresponding to time step rstart. This file contains information regarding the body motion such as body position, velocity, forces, etc.

delete (int 0, 1)

If this option is set to 1, the code keeps in the hard drive only the result files from the two last exported time step, deleting the files from previously exported time steps.

averaging (int 0, 1, 2, 3)

If this parameter is set to 1, 2, or 3, the code performs time averaging of the flow field. Time averaging is typically started when the flow field is fully developed which occurs when the kinetic energy of the flow is stabilized. Therefore, averaging is a two stage process. In the first stage, the simulation is started with the averaging option set to zero and advanced to a point in which the flow is developed. In the second stage, the flow results from stage 1 are used to restart the simulation and start the averaging. The first step to do in stage 2 is to rename the flow field files to be used from stage 1 (ufieldXXXXXX.dat, vfieldXXXXXX.dat, ...) to the file name corresponding to time step 0 files (ufield000000.dat, vfield000000.dat, ...). Then, the simulation can be restarted by activating the averaging option and setting the rstart option to 0 (restart from time step 0). The reason that the files need to be renamed is because of the way that the code performs averaging. The code uses the current time step for dividing the velocity sum and obtaining the averaged results. So if averaging is started at a non-zero time step, the number of velocity summands will not correspond to the current time step number and the average will not be correct. As long as the averaging has been started at time step zero, there is no problem with restarting the simulation during stage 2. The different values for this parameter, 1, 2, and 3, refer to the amount of information that is averaged and exported to the results files. If average is set to 1, only averaged velocities (U, V, and W) and turbulence intensities (uu, vv, ww, uw, vw, uv) are processed. If the parameter is set to 2, in addition to the averaged velocities and turbulence intensities, the averaged pressure and pressure fluctuations are computed. Finally, if the parameter is set to 3, the processed variables include the ones from option 2 plus the averaged vorticity vector. As already indicated in section 3.3.4, to post-process the results and include the averaged results to the output file, the option "avg" needs to be activated. The value given to "avg" should be lower or equal to the value given to the option "averaging".

Options for Boundary Contitions

inlet (int)

The inlet option defines the inflow profile type when the inlet plane is set to inflow mode (see bcs.dat description). It also sets the velocity initial conditions to the one corresponding to the inlet profile.

1: Uniform inflow with velocity value determined by the option flux.

13: This option is used for performing channel flow simulation. It sets the velocity initial condition to follow a log law. The domain height (channel half height) has to be 1.

100: Imports inflow from external file. The external files must have a cross plane grid geometry equivalent to the one of the current simulation grid.

perturb (int 0, 1)

If a non-zero initial condition is given, this option perturbs the initial velocity by adding random velocity values which are proportional to the local streamwise velocity component.

wallfunction (int)

Use wall model at the walls of the immersed bodies. It basically interpolates the velocity at the IB nodes using a wallfunction.

ii_periodic, jj_periodic, kk_periodic (int 0,1)

Consider periodic boundary conditions in the corresponding direction. When this option is chosen in the control file, the corresponding boundaries in bcs.dat need to be set to any non-defined value such as 100.

flux (double)

0: Sets the velocity at the inlet boundary to 1.

Non-zero value = Sets the flux at the inlet boundary. The flux is defined as the bulk inlet velocity divided by the area of the inlet cross section. The units are m^3/s or non-dimensional depending on whether level set is used.

Level Set Method Options

levelset (int 0,1)

Activates the levelset method. If used, the solved Navier-Stokes equations are in dimensional form.

dthick (double)

If using the level set method, this parameter defines half the thickness of the air/water interface. The fluid properties adopt their corresponding value in each phase, and vary smoothly across this interface. Typical values adopted by "dthick" are on the order of 2 times the vertical grid spacing. Larger values may be necessary in extreme cases.

sloshing (int 0, 1, 2)

Sets the initial condition of the sloshing problem in a tank and exports to a file (sloshing.dat) the free surface elevation at the center of the tank.

1: Sets the initial condition for the 2D sloshing problem in a tank.

2: Sets the initial condition for the 3D sloshing problem in a tank.

0: Sloshing problem is not considered.

level_in (int 0, 1, 2)

Flat initial free surface with elevation defined by level_in_height.

1: The free surface normal is in the z direction.

2: The free surface normal is in the y direction.

level_in_height (double)

When the level_in option is active this parameter determines the free surface vertical coordinate which is uniform.

fix_level (int 0,1)

1: The free surface is considered but kept fixed. In this case, the level set equation is not solved.

fix_outlet, fix_inlet (int 0,1)

When using inlet and outlet boundary conditions, activating any of these parameters will keep constant the free surface elevation at the corresponding boundary.

levelset_it (int)

Number of times to solve the reinitialitzation equation for mass conservation. Higher number may be useful in cases involving high curvature free surface patterns.

levelset_tau (double)

Parameter to define the pseudo-time step size used in the reinitialization equation. The pseudo time step is levelset_tau times the minimum grid spacing.

rho0, rho1 (double)

Density of the water and the air respectively.

mu0, mu1 (double)

Dynamic viscosity of the water and the air respectively.

stension (int 0,1)

If active it considers the surface tension at the free surface interface.

Modelling Options and Solver Parameters

les (int 0,1,2)

Activating the LES model.

1: Smagorinsky - Lilly model.

2: Dynamic Smagorinsky model (recommended).

imp (int 1,2,3,4)

Type of solver for the momentum equation. The only value supported is 4 which corresponds to the Implicit solver. Other values correspond to obsolete approaches and are not guaranteed to work.

imp_tol (double)

Tolerance for the momentum equation. A value smaller than 1.0×10^{-5} would be recommended.

poisson (int -1,0,1)

Selection of the Poisson solver. The only value supported is 1, other values correspond to obsolete approaches and are not guaranteed to work.

poisson_it (int)

Maximum number of iteration for solving the Poisson equation. If the tolerance set by the option poisson_tol is reached the Poisson solver is completed before reaching poisson_it iterations.

poisson_tol (double)

Tolerance for the maximum divergence of the flow.

ren (double)

This parameter defines the Reynolds Number in the case of non-dimensional simulations. When using the level set method, the equations solved have dimensions and "ren" is not in use.

inv (int 0,1)

1: Neglects the viscous terms in the RHS of the momentum equation, and thus the flow is considered inviscid.

Immersed Boundary Method Options

imm (int 0,1)

Activate the immersed boundary method. The code will expect a structural mesh (ibmdata00).

body (int)

If using the immersed boundary method, this parameter determines the number of bodies considered. There must be the same number of IB meshes (e.g., ibmdata00, ibmdata01,...,ibmdataXX, where XX is the number of bodies).

thin (int)

Option for simulating bodies with very sharp geometries where the resolution is not sufficient to resolve the depth.

x_c, y_c, z_c (double)

Initial translation of the immersed body position.

angle_x0, angle_y0, angle_z0 (double)

Initial rotation of the immersed body position with respect to the center of rotation defined by x_r, y_r, and z_r. Note that the initial rotation is applied after the translation for which x_r, y_r, and z_r are defined.

Fluid-Structure Interaction Options

fsi (int 0,1)

1: Activates the ability to move the structure in a single translational DoF. Select the desired DoF by setting one of the following options to 1: dgf_ax, dgf_ay, or dgf_az.

forced_motion (int 0, 1)

When "fsi" is active, the parameter controls whether to use prescribed motion or FSI motion.

0: The motion of the structure is computed in a coupled manner with the flow.

1: The motion of the structure is prescribed. Both the position of the structure in time as well as the velocity in time are specified in the function Forced_Motion which is located in the code source file fsi_move.c. In this function the motion is defined with an analytic expression. If a user needs

to set a specific motion which can be defined through a mathematical expression it needs to be implemented by editing this function. Obviously, if the code is edited it also needs to be recompiled.

rfsi (int 0,1)

1: Activates the ability to move the structure in a single rotational DoF. Select the desired rotational DoF by setting rotdir.

rotdir (int 1,2,3)

When rfsi is active, the parameter selects the axis of rotation.

1: Rotation along the x axis.

2: Rotation along the y axis.

3: Rotation along the z axis.

fsi_6dof (int 0,1)

1: Activates the ability to move the structure in up to six DoF. Each of the six DoF can be activated independently of each other with the following control options: dgf_x, dgf_y, dgf_z, dgf_ax, dgf_ay, dgf_az, (described below). Note that any combination of the six DoF is valid regardless of the translational DoF or rotational DoF. One can obtain the same results as using the "fsi" option or the "rfsi" option by activating only one of the 6 DoF.

dgf_ax, dgf_ay, dgf_az (int 0, 1)

In the case of a single translational DoF (fsi 1), the desired translational DoF is specified by setting one of these to 1.

When using fsi_6dof multiple DoF can be activated.

dgf_x, dgf_y, dgf_z (int 0, 1)

In the case of a single rotational DoF (rfsi 1), the desired rotational DoF is specified by setting one of these to 1.

str (int 0, 1)

0: When either fsi or rfsi are active, the parameter uses the loose coupling algorithm.

1: When either fsi or rfsi are active, the parameter uses the strong coupling algorithm.

red_vel, damp, mu_s (double)

Parameters to be used in the test case "VIV of an elastically mounted rigid cylinder".

body_mass (double)

Mass of the structure.

body_inertia_x, body_inertia_y, body_inertia_z (double)

Moment of inertia with respect to the center of rotation defined by x_r, y_r, and z_r.

body_alpha_rot_x, body_alpha_rot_y, body_alpha_rot_z (double)

Damping coefficient for the rotational DoFs.

body_alpha_lin_x, body_alpha_lin_y, body_alpha_lin_z (double)

Damping coefficient for the translational DoFs.

body_beta_rot_x, body_beta_rot_y, body_beta_rot_z (double)

Elastic spring constant for the rotational DoFs.

body_beta_lin_x, body_beta_lin_y, body_beta_lin_z (double)

Elastic spring constant for the translational DoFs.

x_r, y_r, z_r (double)

Center of rotation in the rotational DoFs.

fall_cyll_case (int 0, 1)

Option for the falling cylinder test case.

Wave Generation Options

wave_momentum_source (int 0, 1, 2)

When using the level set method, the parameter activates the wave generation module, based on the method of Guo and Shen (2009).

0: Waves are not generated.

1: Waves are read from an external file.

2: A single monochromatic wave is generated.

wave_angle_single (double)

In the case that wave_momentum_source is equal to 2, the parameter sets the wave initial phase.

wave_K_single (double)

In the case that wave_momentum_source is equal to 2, the parameter sets the wave number.

wave_depth (double)

In the case that wave_momentum_source is active, the parameter sets the water depth. This parameter is used in the dispersion relation to compute the wave frequency.

wave_a_single (double)

In the case that wave_momentum_source is equal to 2, the parameter sets the wave amplitude.

wave_ti_start (int)

Time step to start applying the wave generation method.

wave_skip (int)

This option is used in the case that wave_momentum_source is equal to 1 (waves are imported from external files). The external data file to be imported, which has been generated using an external code, may have a time step size not equivalent to the time step of the present code simulation. Usually the time step size in the wave data is significantly larger than that from the present simulation ($\Delta t_{wave-data} = \Delta t_{simulation} \times b$, where b is an integer). By setting wave_skip to b, the code will import a new wave data file every wave_skip time steps, and since wave_skip=b, the time of the simulation will match the time of the wave data.

wind_skip (int)

This option is equivalent to wave_skip but for importing the inlet wind profile when the option air_flow_levelset is equal to 2.

wave_start_read (int)

This parameter is useful for restarting the simulation in the case that wave_momentum_source is equal to 1 (waves are imported from external files). The code will import the wave file corresponding to the wave time step wave_start_read, instead of importing the starting wave data file (WAVE_info000000.dat).

wind_start_read (int)

This parameter is useful for restarting the simulation in the case that air_flow_levelset is equal to 2 (inlet wind profile is imported from external files). The code will import the wind data file corresponding to the wind time step wind_start_read, instead of importing the first wind data file (WAVE_wind000000.dat), .

wave_recicle (int)

In the case that wave_momentum_source is equal to 1 (waves are imported from external files) and the wave time step reaches this number, the wave time step is recycled to 0, which means that the code will import the wave data corresponding to time step 0 (WAVE_info000000.dat).

wind_recicle (int)

In the case that air_flow_levelset is equal to 2 (wind is imported from external files) and the wave time step reaches this number, the wind data time step is recycled to 0, which means that the code will import the wind data corresponding to time step 0 (WAVE_wind000000.dat).

wave_sponge_layer (int 0, 1, 2)

1: Sponge layer method is only applied at the x boundaries.

2: Sponge layer method is applied at the four lateral boundaries.

wave_sponge_xs (double)

Length of the sponge layer applied at the x boundaries.

wave_sponge_x01 (double)

Starting x coordinate of the first sponge layer applied at the x boundary.

wave_sponge_x02 (double)

Starting x coordinate of the second sponge layer applied at the x boundary.

wave_sponge_ys (double)

Length of the sponge layer applied at the y boundaries.

wave_sponge_y01 (double)

Starting y coordinate of the first sponge layer applied at the y boundary.

wave_sponge_y02 (double)

Starting y coordinate of the second sponge layer applied at the y boundary.

Turbine Parameterization Options

rotor_modeled (int 0, 1, 2, ..., 6)

Activate the turbine modeling option with the following parameterization approach:

1: Actuator disk model using the induction factor as input parameter.

2: Option for development purpose. This option is currently obsolete.

3: Actuator line model.

4: Actuator disk model using thrust coefficient as a input parameter.

5: Actuator surface model (under development).

6: Actuator line model with an additional actuator line for computing the reference velocity.

turbine (int)

Number of wind/hydro-kinetic turbines to be modeled.

reflength (double)

Reference length of the turbine. The code will divide the imported turbine diameter (from the mesh file and Turbine.inp) by this amount.

rotatewt (int 1,2,3)

Direction to which the turbines point to.

1: i direction

2: j direction

3: k direction

r_nacelle (double)

This parameter represent the radius of the turbine nacelle. The code will ignore the rotor effect within this radius.

num_foiltype (int)

Number of foil types used along the turbine blade. VFS requires a description file named FOIL00, FOIL01, ..., for each foil type as described in Section $4.2.6.

num_blade (int)

Number of blades in the turbine rotor.

refvel_wt, refvel_cfd (double)

These parameters do not have any effect in the simulation, and are only for normalizing the output profiles. One can set them to 1 and normalize when plotting the data.

loc_refvel (int)

Distance upstream of the turbine in rotor diameters where the turbine incoming velocity or reference velocity is computed.

deltafunction (int)

Type of smoothing function within which the pressure due to the rotor is applied.

halfwidth_dfunc (double)

Half the distance for which the turbine effect is smoothed. The value is expressed in number of grid nodes.

4.2.2 The bcs.dat File

The bcs.dat file is another text file with information about the boundary conditions of the fluid domain boundaries. Lets denote the six boundaries as Imin, Imax, Jmin, Jmax, Kmin, and Kmax, corresponding to the starting and ending boundary in the i, j and k directions, respectively.

The format of the bcs.dat file is a single line with the 6 integers corresponding to each of the boundaries. This number can adopt the following values:

Table 4.1: Options for the bcs.dat file

Boundary condition type	**Value**
Slip Wall	10
No slip wall	1
No slip with wall modelling, smooth wall	-1
No slip with wall modelling, rough wall	-2
[1]Periodic boundary conditions	100
[1]Inlet	5
Outlet	4

The bcs.dat file has the following aspect: Imin-value Imax-value Jmin-value Jmax-value Kmin-value Kmax-value

Example. Simulation case with slip wall at the Imin, Imax, Jmin, Jmax boundaries, and inlet and outlet along the k direction:
10 10 10 10 5 4

[1]Require additional information in the control file. Further details can be found in the corresponding section.

4.2.3 The Grid File

The grid file (grid.dat) is formatted with the standard PLOT3D. The file can be imported in binary form or in ASCII form. For importing the grid in binary form the option binary in the control.dat has to be set to 1, otherwise the code expects the ASCII form.

Any grid generator software that is able to export PLOT3D grids may be suitable for VFS. However, Pointwise is recommended.

When creating the mesh, the user needs to pay attention to the orientation of two different sets of coordinate systems. The Cartesian components which are indicated in Figure 4.1 with x, y, and z, and the curvilinear components which are attached to the mesh and are referred to as i, j, and k. Both coordinate systems should be right-hand oriented.

The recommended axis combination between Cartesian components and grid coordinates is depicted in Figure 4.1.

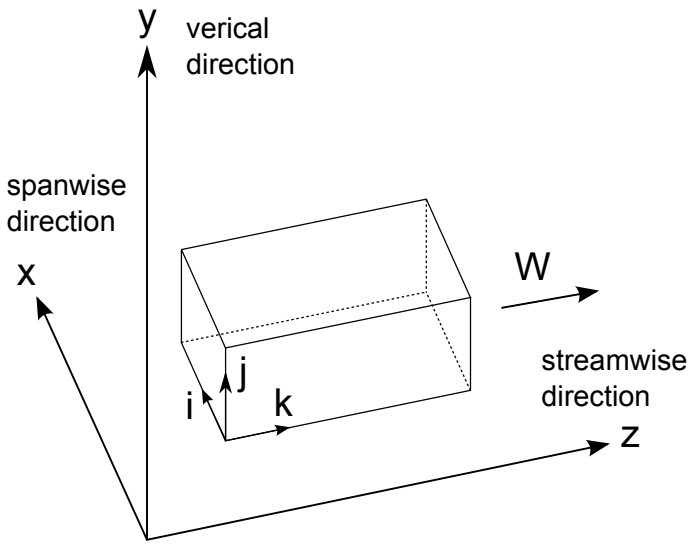


Figure 4.1: Axis orientation

A third format type that VFS can handle is "SEGMENT". This format is suitable only for cases where the fluid mesh is Cartesian, as the only information that the mesh file stores are the grid points of the three axis. An obvious advantage of this approach is that the mesh file size is much smaller than the "PLOT3D" formatted files.

To use "SEGMENT" format, the grid file should be named "xyz.dat" and the option "xyz" in the control file should be set to 1. In the first three lines, the "xyz.dat" file contains the number of grid nodes of the mesh for each of the three axis N_x, N_y, and N_z. The values are followed by the three coordinate of the points in the X axis, then the coordinates of the points in the Y axis, and finally the coordinates of the points in the Z axis as follows:
X_{x1} Y_{x1} Z_{x1}
X_{x2} Y_{x2} Z_{x2}

...
X_{xN_x} Y_{xN_x} Z_{xX_x}
X_{y1} Y_{y1} Z_{y1}
X_{y2} Y_{y2} Z_{y2}
...
X_{yN_y} Y_{yN_y} Z_{yX_y}
X_{z1} Y_{z1} Z_{z1}
X_{z2} Y_{z2} Z_{z2}
...
X_{zN_z} Y_{zN_z} Z_{zX_z}

4.2.4 The Immersed Boundary Grid File

The immersed boundary method allows one or more immersed bodies to be incorporated into the computational domain. If more than one body is considered, by default, each body has its own body mesh and its name is ibmdata00 for the first body, ibmdata01 for the second body, etc.

The body mesh is an unstructured surface mesh with triangular nodes. The format is the standard UCD. When generating an immersed boundary mesh one needs to consider the following:

- The normal direction of the triangular elements must point towards the flow.
- In general, a triangular mesh with triangles of similar sizes as the fluid background mesh is recommended. If the immersed boundary is a flat wall, the triangular mesh may be coarser than the fluid mesh without loss of accuracy.

4.2.5 The Wave Data External File

The wave generation module allows the incorporation of broadband wave fields with large number of frequency components (wave_momentum_source set to 1). The origin of the wave data can be from a precursor simulation (far-field simulation) using an external wave code or from real measurements. A broadband wave field composed of $NZMOD/2 \times (NYMOD + 1)$ wave frequencies, where $NZMOD/2$ is the number of frequencies in the Z direction and $2 \times (NYMOD + 1)$ is the number of frequencies in the X direction) can be given by the following expression

$$\eta(z,x) = \sum_{k_z=1}^{k_z=NZMOD/2} \sum_{k_x=-NXMOD/2}^{k_x=NXMOD/2} a_{k_z,k_x} cos(k_z{*}PEZ{*}z + k_x{*}PEX{*}x + \theta_{k_z,k_x}) \quad (4.1)$$

where η is the free surface elevation, k_z and k_x indicate the directional wave numbers, a is the wave amplitude, and (θ) is the wave phase. PEZ and PEX are coefficients to scale the wave numbers, which are usually set to 1.

The code will read the wave data file named WAVE_infoXXXXXX.dat, where XXXXXX represents the time step of the wave data. The first line of the wave data file contains the time of the wave, NZMOD, NXMOD, PEZ, and PEX. The second line is where the actual wave data starts. The wave file is as follows:
$time_{wave}$ $NZMOD$ $NXMOD$ PEZ PEX
$a_{k_z=1,k_x=0}$ $\theta_{k_z=1,k_x=0}$
$a_{k_z=2,k_x=0}$ $\theta_{k_z=2,k_x=0}$
...
$a_{k_z=NZMOD/2,k_x=0}$ $\theta_{k_z=NZMOD/2,k_x=0}$
$a_{k_z=1,k_x=1}$ $\theta_{k_z=1,k_x=1}$
$a_{k_z=2,k_x=1}$ $\theta_{k_z=2,k_x=1}$
...
$a_{k_z=NZMOD/2,k_x=1}$ $\theta_{k_z=NZMOD/2,k_x=1}$
$a_{k_z=1,k_x=-1}$ $\theta_{k_z=1,k_x=-1}$
$a_{k_z=2,k_x=-1}$ $\theta_{k_z=2,k_x=-1}$
...
$a_{k_z=NZMOD/2,k_x=-1}$ $\theta_{k_z=NZMOD/2,k_x=-1}$
$a_{k_z=1,k_x=2}$ $\theta_{k_z=1,k_x=2}$
$a_{k_z=2,k_x=2}$ $\theta_{k_z=2,k_x=2}$
...
$a_{k_z=NZMOD/2,k_x=2}$ $\theta_{k_z=NZMOD/2,k_x=2}$
$a_{k_z=1,k_x=-2}$ $\theta_{k_z=1,k_x=-2}$
$a_{k_z=2,k_x=-2}$ $\theta_{k_z=2,k_x=-2}$
...
$a_{k_z=NZMOD/2,k_x=-2}$ $\theta_{k_z=NZMOD/2,k_x=-2}$
...
$a_{k_z=1,k_x=NXMOD/2}$ $\theta_{k_z=1,k_x=NXMOD/2}$
$a_{k_z=2,k_x=NXMOD/2}$ $\theta_{k_z=2,k_x=NXMOD/2}$
...
$a_{k_z=NZMOD/2,k_x=NXMOD/2}$ $\theta_{k_z=NZMOD/2,k_x=NXMOD/2}$
$a_{k_z=1,k_x=-NXMOD/2}$ $\theta_{k_z=1,k_x=-NXMOD/2}$
$a_{k_z=2,k_x=-NXMOD/2}$ $\theta_{k_z=2,k_x=-NXMOD/2}$
...
$a_{k_z=NZMOD/2,k_x=-NXMOD/2}$ $\theta_{k_z=NZMOD/2,k_x=-NXMOD/2}$

As discussed in the control option for the wave module, the wave time step size may no be equal to the time step of the present code simulation. Usually the time step in the wave data is significantly larger than that from the present simulation ($\Delta t_{wave-data} = \Delta t_{simulation} \times b$, where b is an integer). By setting wave_skip to b, the code will import a new wave data file every wave_skip time steps, and since wave_skip=b, the time of the simulation will match the time of the wave data.

If the wave frequencies do not vary in time, one could use a single wave data file by setting the option wave_skip to a very large value.

The wave module also allows the wind field associated with a wave field pre-computed simulation to be incorporated by setting air_flow_levelset to 2. In such a case, the code will read the wave data file named WAVE_windXXXXXX.dat, with XXXXXX representing the time step of the wind data. The first line of the wind data file contains the time of the far-field simulation, the number of grid points in the vertical direction NY, and the number of grid points in the horizontal direction NX. The actual wave data starts in line two. The overall structure of the file is as follows:
$time_{far-field}$ NY NX
$X_{0,0}$ $Y_{0,0}$ $Z_{0,0}$ $U_{0,0}$ $V_{0,0}$ $W_{0,0}$
$X_{0,1}$ $Y_{0,1}$ $Z_{0,1}$ $U_{0,1}$ $V_{0,1}$ $W_{0,1}$
...
$X_{0,NX}$ $Y_{0,NX}$ $Z_{0,NX}$ $U_{0,NX}$ $V_{0,NX}$ $W_{0,NX}$
$X_{1,0}$ $Y_{1,0}$ $Z_{1,0}$ $U_{1,0}$ $V_{1,0}$ $W_{1,0}$
$X_{1,1}$ $Y_{1,1}$ $Z_{1,1}$ $U_{1,1}$ $V_{1,1}$ $W_{1,1}$
...
$X_{1,NX}$ $Y_{1,NX}$ $Z_{1,NX}$ $U_{1,NX}$ $V_{1,NX}$ $W_{1,NX}$
...
$X_{NY,0}$ $Y_{NY,0}$ $Z_{NY,0}$ $U_{NY,0}$ $V_{NY,0}$ $W_{NY,0}$
$X_{NY,1}$ $Y_{NY,1}$ $Z_{NY,1}$ $U_{NY,1}$ $V_{NY,1}$ $W_{NY,1}$
...
$X_{NY,NX}$ $Y_{NY,NX}$ $Z_{NY,NX}$ $U_{NY,NX}$ $V_{NY,NX}$ $W_{NY,NX}$

4.2.6 The Files Required for the Turbine Rotor Model

The Turbine.inp Control File

The Turbine.inp file is a text file containing input parameters used by the rotor model. The file has two lines, the first line is ignored by VFS and only used for informative purposes by listing the input variable names. The second line is the control value corresponding to the variable listed in line 1 as shown in the example below.

```
1 nx_tb-ny_tb-nz_tb-x_c-y_c-z_c-indf_axis-Tipspeedratio-J_rot ...
2 0.0 0.0 1.0 0.0 0.104 0.256 0.36 4.5 14380000 ...
```

nx_tb, ny_tb, nz_tb (double)

Normal direction of the turbine rotor plane.

x_c, y_c, z_c (double)

The turbine rotor initial translation.

indf_axis (double)

Induction factor when using the actuator disk model (rotor_model=1).

Tipspeedratio (double)

Tip speed ratio when using the actuator line model (rotor_model=3,6). The tip-speed ratio (TSR) can adopt negative values which indicate that the turbine is rotating counterclockwise with respect to the stream-wise axis.

J_rotation (double)

Rotor moment of inertia used only when the option "turbinetorquecontrol" is active.

r_rotor (double)

Radius of the turbine rotor.

CP_max (double)

Maximum power coefficient of the turbine; used only when the option "turbinetorquecontrol" is active.

TSR_max (double)

Refers to the maximum TSR of the turbine; used only when the option "turbinetorquecontrol" is active.

angvel_fixed (double)

Rotor rotational speed when the option "fixturbineangvel" is active. The variable angvel_fixed can adopt negative values which indicate that the turbine is rotating counterclockwise with respect to the stream-wise axis.

Torque_generator (double)

Turbine torque, used only when the option "turbinetorquecontrol" is active.

pitch (double)

Pitch angle of the turbine blades when using actuator line or actuator surface models.

CT (double)

Thrust coefficient used with rotor_modelled 4.

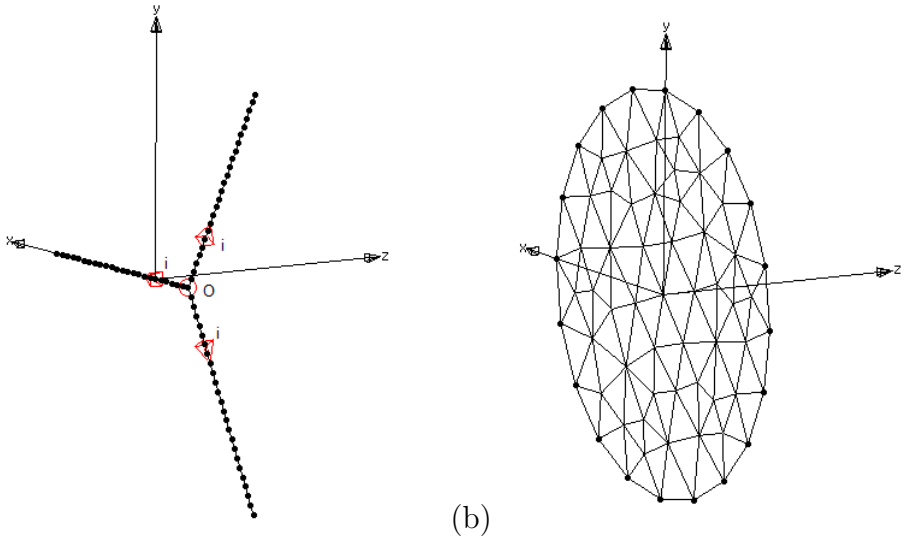
The acldata000 Mesh File

The acldata000 file is an ASCII data file containing the mesh of the turbine model. When using the actuator line model, the file consists of n segments, where n is the number of rotor blades, as shown in Figure 4.2(a). The ASCII data file uses the SEGMENT format.

In the case of the actuator disc models, the mesh is a UCD formatted unstructured triangular mesh, and the rotor is represented with a circle as shown in Figure 4.2(b).

The turbine center, o, of this mesh could be located directly at the actual position of the turbine, or alternatively, centered at the origin and later translated with the control options x_c, y_c, and z_c defined in the rotor model control file "turbine.inp".

In the actuator line model the coordinate attached to the segment i has to point towards the tip of the blade. In the actuator disk model, the direction normal to the rotor has to point towards the direction of the flow.



(a) (b)

Figure 4.2: Representation of the "acldata000" mesh used to represent the blades in the (a) actuator line model and (b) actuator disk model.

The Urefdata000 Mesh File

The Urefdata000 file is a UCD formatted triangular mesh equivalent to the actuator disk acldata000 file. The purpose of this file is to compute the inflow reference velocity required for both the actuator disk and actuator line models.

The disk dimensions and normal direction in Urefdata000 have to match the dimensions of the turbine rotor defined in "turbine.inp".

The code positions the disk upstream of the actual turbine location. At every time step, the velocity of the flow is transferred to each triangular element of this mesh. By adding the velocity at all triangular elements and dividing by the surface of the disk, the code computes the turbine reference velocity (disk average velocity). As an

example, when using the actuator line, the reference velocity is used for determining the TSR of the simulation.

The CD00, CD01, CD02, ..., and CL00, CL01, CL02, ... Files

These files contain the lift and drag coefficients at each profile along the turbine blades when using the actuator line model (rotor_model 3 or 6). The first two lines in the file are descriptive and the third line defines the number of data points in the file. Starting at line 4, the angle of attack (column 1), in degrees, and Lift/Drag coefficient (column 2) are listed as shown in the example below.

```
1 Airfoil type: DU97-W-300
2 Drag coefficients
3 31
4 7.50925697358677e-001    1.41002221673661e-002
5 2.25610960256727e+000    2.10614663046161e-002
6 4.32615403604048e+000    2.79782769686497e-002
7 6.20902246358924e+000    2.78301653912614e-002
8 ...
9 8.98528141199704e+001    2.13806467538879e+000
```

The FOIL00, FOIL01, FOIL02, ... Files

These files contain the angle of attack and chord length for each profile used along the turbine blades when using the actuator line model (rotor_model 3 or 6). The first two lines in the file are descriptive and the third line defines the number of data points in the file. Starting at line 4, the distance from the blade section to the turbine hub in non-dimensional units or in meters (column 1), the blade section angle of attack in degrees (column 2), and the profile chord length in non-dimensional units or meters (column 3) are listed as shown in the example below.

```
1  Turbine type: Clipper 2.5 MW
2  Airfoil type: Circular
3  7
4  0.000  9.5  2.400
5  2.800  9.5  2.400
6  3.825  9.5  2.385
7  4.950  9.5  2.259
8  6.950  9.5  2.338
9  8.950  9.5  2.339
10 10.800  9.5  2.848
```

Chapter 5

Library Structure

5.1 The Source Code Files

The source code is structured in several files with extension ".c" and one file with extension ".h". The header file (variables.h) is included at the beginning of any other ".c" file and contains all the function prototypes, global variable definitions, and structure definitions. The ".c" files contain the subroutines which are generally grouped by code module. A brief description of the ".c" files is presented as follows:

main.c
: Main code file where the code is initialized and finalized

bcs.c
: Subroutines for specifying boundary conditions

compgeom.c, ibm.c, ibm_io.c, variables.c
: Subroutines for the IB method

fsi.c, fsi_move.c
: Subroutines for the FSI module

level.c, distance.c
: Subroutines for simulating two-phase free surface flows with the levelset method

wave.c
: Subroutines for the wave module (also to simulate wind over waves, in which the wind is imported from a far-field simulation)

data.c
: Subroutines for post-processing and visualizing the results

wallfunction.c
: Subroutines for the wall modeling

rotor_model.c
Contains all the subroutines that are necessary for simulating a wind turbine or a hydro-kinetic turbine using the actuator disk or actuator line models

les.c
Subroutines for the turbulent models

solvers.c, implicitsolver.c, momentum.c, poisson.c, poisson_hypre.c, rhs.c, rhs2.c, timeadvancing.c, timeadvancing1.c
Contains all the subroutines used by the flow solver, including the momentum and the Poisson equations

init.c
Subroutines for initializing the code variables

metrics.c
The subroutines for computing the grid Jacobian and metrics

5.2 The Flow Solver

To describe the basic elements of the flow solver we present a code flow chart of VFS, which displays the order in which the relevant functions of the code are called. This code flow chart corresponds to the most simple case that VFS can simulate and no additional module is considered. An example would be to perform Direct Numerical Simulation of the channel flow case.

As in any "c" code, the so called main function is the entry point or where the software starts the execution. In the code flow chart presented below the functions, emphasized in bold, are indented such that the functions from a lower level are called by the function of the above level.

- **main** (pre-processing)
  In the first part of this main function, the code pre-processing is performed as follows:
  - **MG_Initial**
    Reads the structured grid file (grid.dat), partitions the domain within the cpus, allocates memory for the main variables in a partitioned form. Also reads the boundary conditions file (bcs.dat).
    * **FormMetrics**
      Computes the metrics and Jacobians of the transformation given by equation (2.2).
  - **Calc_Inlet_Area**
    Computes the inlet area corresponding to section k=0. The code was designed such that the stream-wise direction is k.

- **SetInitialGuessToOne**
  Sets the initial velocity of the whole computational domain at time=0 to a specific profile defined by the variable inlet.
- **Contra2Cart**
  Using the Cartesian velocity components at the cell centers, the contravariant velocity components at the cell faces are calculated through interpolation.

- **main** (time iteration)
  At this point of the main function, the time-stepping loop starts.

  - **Flow_Solver**
    This function solves for the velocity and pressure fields to advance to time step ti+1.
    * **Calc_Minimum_dt**
      Calculates and prints the minimum time step size (dt) required such that the CFL number is equal to 1.
    * **Pressure_Gradient**
      Reads the pressure field and computes the pressure gradient.
    * **Formfunction_2**
      Forms the right hand side of the momentum equation.
    * **Implicit_MatrixFree**
      Solves the momentum equation.
    * **PoissonSolver_Hypre**
      Solves the Poisson equation in the second step of the fractional step method to obtain the pressure correction.
    * **UpdatePressure**
      The pressure correction is applied to obtain the pressure field.
    * **Projection**
      Corrects the velocity to make it divergence free.
    * **IB_BC**
      Sets most of the boundary conditions. Note, however, that other functions such as "Implicit_MatrixFree" and "Contratocart" also deal with a part of the boundary conditions.
    * **Divergence**
      Checks and prints the maximum divergence to the output file Convergence_du.
    * **Contra2Cart**
      Using the Cartesian velocity components at the cell centers, the contravariant velocity components at the cell faces are calculated through interpolation.

    - **Calc_ShearStress**
      Computes and outputs the shear stress.
    - **KE_Output**
      Exports the total kinetic energy of the whole computational domain to the file Kinetic_Energy.dat.
    - **Ucont_P_Binary_Output**
      Writes the flow field results to files provided that the time step is a multiple of the control option "tiout".

- End of time-stepping loop

  – **MG_Finalize**
    This function is called right before ending the code to di-allocate all the memory created during the execution of the code.

5.3 Code Modules

In the present section the main functions used by the different modules of the code are reviewed. All modules follow a common structural pattern. First, a group of functions are called for pre-processesing purposes, then, a second set of functions are called with the purpose of advancing the solution in time.

- **Subroutines for pre-processing**. Upon initiation of the program and before starting the time iteration, a set of functions are called to: (1) import the module specific input files (if any); and (2) initialize and allocate memory for the necessary variables. This process happens only once in the beginning of the main function located in the file "main.c".

- **Subroutines for time advancing**. After the initial pre-processing part is completed, the code is ready to start advancing in time. Then a second set of functions are used to compute, at every time step, the necessary elements involved in the corresponding module. This part is generally executed from the function Flow_Solver located in "solvers.c".

5.3.1 The Level Set Method Module

- **Subroutines for pre-processing**. In this module, the pre-processing basically consists of initializing the level set main variables and setting the free-surface initial condition.

  – **MG_Initial** Initializes the levelset variables. The levelset main variable is named "level[k][j][i]", which is defined as the distance from the current cell center to the closest point of the free surface interface. It adopts a positive value in the water phase and negative value in the air phase. The free surface interface coincides with the zero level.

- **Levelset_Function_IC** Sets the initial position of the free-surface.

- **Subroutines for time advancing**. Here the time-advancing involves solving an advection equation to find the new location of the free surface interface and a mass conserving reinitialization to ensure that the mass within the two phases is conserved.

  - **Advect_Levelset** Solves the levelset advection equation.
    - **Levelset_Advect_RHS** Forms the right hand side terms of the advection equation.
  - **Reinit_Levelset** Solves the mass conserving reinitizlization equation.
    - **Init_Levelset_Vectors** Creates temporary arrays for solving the reinitialization equation.
    - **Solve_Reinit_explicit** Solves the equation in an explicit form.
      - **Distance_Function_RHS** Forms the right hand side terms of the reinitialization equation.
    - **Destroy_Levelset_Vectors** Deletes the temporary arrays.
  - **Compute_Density** Updates the density and viscosity values of the fluid with the values corresponding to the new location of the free surface. The function executes the functions Advect_Levelset and Reinit_Levelsetfree, which update the free surface location.
  - **Compute_Surface_Tension** Applies the surface tension at the free surface.
  - **Levelset_BC** Sets the boundary conditions of the free surface. The function is called both before and after solving the advective and the reinitialization equations.
  - **Calc_free_surface_location** Exports the free surface elevation to the external file FreeSurfaceElev_XXXXXX.dat (XXXXXX refers to the time step) at every "tiout" time steps.

5.3.2 The Large-Eddy Simulation (LES) Method Module

- **Subroutines for pre-processing**. In this module the pre-processing basically consists of initializing the LES main variables.

  - **MG_Initial** Initializes the LES model main variables.

- **Subroutines for time advancing**. Here the time-advancing involves computing the new eddy viscosity, which is added to the diffusion term of the momentum equation.

- **Compute_Smagorinsky_Constant_1** Computes the Smagorinsky constant C_s.
- **Compute_eddy_viscosity_LES** Computes the eddy viscosity μ_t by applying equation (2.26).
- **Formfunction_2** Adds the eddy viscosity term to the right hand side of the momentum equation.

5.3.3 The Immersed Boundary (IB) Method Module

- **Subroutines for pre-processing**. In this module the pre-processing consists of initializing the IB method variables and importing the IB mesh.
  - **main** Initializes the primary variables for the IB method.
  - **ibm_read_ucd** Reads and imports the body triangular mesh (ibmdata00, ibmdata01, ...).
  - **ibm_search_advanced** Performs a classification of the fluid nodes depending on its position with respect to the structure. This classification is stored in the variable "nvert". If nvert is 0 the node belongs to the fluid domain and the equations are solved; if nvert is 3, the node belongs inside the structural domain and the node is blanked from the computational domain; if nvert is 1, the node is an IB node, which belongs in the fluid domain but is located at the immediate vicinity of the structure. IB nodes are where the velocity boundary condition of the body are specified.
  - **ibm_interpolation_advanced** Computes the velocity boundary conditions at the IB nodes. This computation can be done using linear interpolation or using a wall model.
    - **noslip** Applies the no-slip-wall boundary condition using linear interpolation.
    - **freeslip** Applies the slip-wall boundary condition using linear interpolation. Used when the inviscid option is active.
    - **wall_function** Applies a wall model assuming a smooth wall. Used when the option wallfunction is active.
    - **wall_function_roughness** Applies a wall model assuming a rough wall. Used when the wallfunction option is active and rough_set is specified.

- **Subroutines for time advancing**. The time-advancing part depends on whether the body is moving or not. While the velocity boundary condition at the IB nodes has to be recomputed at every time step, the classifications of nodes has to be recomputed only if the body is moving.

- **ibm_search_advanced** This function does not need to be called if the body is not moving. Otherwise, this function needs to be called at every time step, if the body is moving, to update the node classification once the body position has been updated.
- **ibm_interpolation_advanced** The velocity at the IB nodes has to be updated at every time step.

5.3.4 The Fluid-Structure Interaction (FSI) Algorithm Module

- **Subroutines for pre-processing**. In this module the pre-processing consists of initializing the FSI variables and applying an initial motion to the body.
  - **FsiInitialize** Initializes the variables for the body motion; either the motion is prescribed or determined using FSI.
  - **FSI_DATA_Input** Reads the external file "DATA_FSIXXXXXX_YY.dat". (XXXXXX refers to the time step and YY to the body number). This process is necessary when the simulation is restarted. The option rstart_fsi needs to be active.
  - **Elmt_Move_FSI_TRANS** This function applies a linear translation to the body mesh in a single DoF. The function is called when the single translational DoF module is in use. In the pre-processing, the function is used to apply an initial translation to the body either when starting the simulation or when restarting.
  - **Elmt_Move_FSI_ROT** This function applies a rotation to the body mesh in a single DoF. The function is called when the single rotational DoF module is in use. In pre-processing, the function is used to apply an initial rotation to the body, either when starting the simulation or when restarting.
    - **rotate_xyz** This function applies a rotation to a given point with respect to a center of rotation in a given direction.
  - **Elmt_Move_FSI_ROT_TRANS** This function applies the structural motion in any of the six DoF to the body mesh. The function is called when the six DoF module is in use. During pre-processing, the function is used to apply an initial motion to the body either when starting the simulation or when restarting.
    - **rotate_xyz6dof** This function applies a rotation to a given point with respect to a center of rotation in the three axial directions.
- **Subroutines for time advancing**. The time-advancing part depends on whether the body is moving or not. As already discussed for the IB method

module, the velocity boundary condition at the IB nodes has to be recomputed at every time step, and the classifications of nodes has to be recomputed only if the body is moving.

- **Struc_Solver** This function computes and updates the new position of the body. The function is called within the main function at every time step.
  - **Calc_forces_SI** Computes the force and moments that the fluid imparts to the body.
  - **Calc_forces_SI_levelset** Computes the force and moments that the fluid imparts to the body. It replaces **Calc_forces_SI** when the level set method is in use.
  - **Forced_Motion** Computes the position and velocity of the structure using the prescribed motion mode. Both the position and the velocity are specified through an analytic expression. Needs to be followed by a call to either the function **Elmt_Move_FSI_TRANS** or **Elmt_Move_FSI_ROT_TRANS**.
  - **Calc_FSI_pos_SC** Solves the EoM in a single translational DoF. Needs to be followed by a call to **Elmt_Move_FSI_TRANS**.
  - **Calc_FSI_pos_SC_levelset** Solves the EoM in a single translational DoF when the levelset method is active. Needs to be followed by a call to **Elmt_Move_FSI_TRANS**.
  - **Calc_FSI_pos_6dof_levelset** Solves the six DoF EoM. Needs to be followed by a call to **Elmt_Move_FSI_ROT_TRANS**
  - **Calc_FSI_Ang** Solves the EoM in a single rotational DoF. Needs to be followed by a call to **Elmt_Move_FSI_ROT**.
  - **Forced_Rotation** Computes the rotation and angular velocity of the structure using the prescribed motion mode through an analytic expression. Needs to be followed by a call to **Elmt_Move_FSI_ROT**.
  - Note that after the motion has been applied to the body mesh, the function **ibm_search_advanced** needs to be applied to update the fluid mesh node classification.
- **FSI_DATA_Output** At every "tiout" time step, it exports the body motion information in the file "DATA_FSIXXXXXX_YY.dat". (XXXXXX refers to the time step and YY to the body number).

5.3.5 The Wave Generation Module

- **Subroutines for pre-processing**. In this module the pre-processing consists of initializing the variables for the wave generation method and for specifying the inlet wind from the far-field precursor simulation.

- **Initialize_wave** Initializes the variables for the wave generation method.
- **Initialize_wind** Initializes the variables for importing the wind field from the far field precursor simulation.

- **Subroutines for time advancing**. In this module the code needs to import the wave data and, if necessary, the wind data, at the time steps for which it needs to be updated (every wave_skip and wind_skip time steps, respectively). Then the imported wave/wind information is applied.
  - **WAVE_DATA_input** Sets the water wave field information (amplitude, frequencies, direction angle, ...) to be simulated.
    If the option wave_momentum_source is 1, the function imports the wave information from an external file.
    If wave_momentum_source is equal to 2, the function uses the information given in the control file.
  - **WAVE_Formfuction2** Adds the pressure force in the right hand side of the momentum equation in the form of a source term.
  - **WAVE_SL_Formfuction2** Applies the sponge layer method at the side wall boundaries that are specified in the control file.
  - **WIND_DATA_input** Reads the external file containing information of the wind field of the far-field simulation to be applied at the inlet of the present simulation.
    - **WIND_vel_interpolate** Function to perform bi-linear interpolation to obtain the velocity values at the present fluid mesh which generally differs from the far-field fluid mesh.

5.3.6 The Rotor Turbine Modeling Module

Actuator Disk Model

The actuator disk model is activated by setting rotor_modeled to 1 (the model input is the induction factor) or to 4 (the input is the thrust coefficient).

- **Subroutines for pre-processing**. In the turbine modelling module the pre-processing subroutines import the turbine model input file, and initialize the corresponding variables, allocating memory if necessary. Again, this process happens only once in the beginning of the main function located in the file "main.c".
  - **main** Initializes variables and imports the turbine control file "Turbine.inp".
    - **disk_read_ucd** Imports the disk mesh. The function is called first to import the actual turbine mesh, named acddata000, and then to import the disk mesh for the reference length, named Urefdata000.

    * **Pre_process** This functions searches the fluid cells that are at the vicinity of the disk mesh and it is called every time step provided that the disk changes its position.

- **Subroutines for time advancing**. After the previous part is completed and the code starts advancing in time, the code computes the necessary elements involved in the turbine models at every time step, such as the interaction forces between the fluid and the turbine rotor or updates the new position of the rotor. These subroutines are called in the function Flow_Solver located in "solvers.c".

  – **Uref_ACL**
  Calculates the reference velocity (U_ref). This value corresponds to the space averaged velocity along a disk of the same diameter as the rotor and located some distance upstream of the turbine. The value is multiplied by the disk normal that points downstream.

  – **Calc_U_lagr**
  Interpolates the velocity from the fluid mesh to the Lagrangian points at the rotor model mesh.

  – **Calc_F_lagr**
  Computes the actuator line forces at each element of the Lagrangian mesh.

  – **Calc_forces_rotor**
  Computes the overall turbine forces.

  – **Calc_F_eul**
  Transfers the forces from the Lagrangian mesh to the fluid mesh.

Actuator Line Model

The actuator line model is activated by setting rotor_modeled to 3 (the reference velocity is computed within a disk located upstream of the turbine) or to 6 (the reference velocity is computed within a line mesh instead of a disk).

- **Subroutines for pre-processing**. Equivalent to the actuator disk model with the difference that the turbine blades are represented with a one-dimensional mesh and the blade profile information is required.

  – **main** Initializes variables and imports the turbine control file "Turbine.inp".
    * **ACL_read_ucd** Imports the actuator line mesh file named "acldata000".
    * **disk_read_ucd** Imports the disk mesh file for computing the reference velocity named "Urefdata000".
    * **Pre_process** This function searches the fluid cells that are at the vicinity of the actuator line mesh or the reference velocity disk/-line mesh. The function is called every time that the disk/line mesh changes its position.

    * **airfoil_ACL** Imports the airfoil information.

- **Subroutines for time advancing**.
  - **Uref_ACL**
    Calculates the reference velocity (U_ref) for the actuator line model. This value corresponds to the space averaged velocity along a disk of the same diameter as the rotor and located some distance upstream of the turbine. The value is multiplied by the disk normal that points downstream.
  - **Calc_turbineangvel**
    Calculates the rotational velocity of the turbine based on the U_ref velocity value.
  - **rotor_Rot**
    Applies a rotation to the turbine equivalent to the rotation velocity times the time step dt.
  - **Pre_process**
    Updates the new location of the turbine.
  - **refAL_Rot**
    Applies a constant rotation to the reference line located upstream of the turbine.
  - **rotor_Rot_6dof_fsi**
    If the 6 DoF FSI module is active, this function applies the same motion to the actuator line as was applied to the floating platform.
  - **Calc_U_lagr**
    Interpolates the velocity from the fluid mesh to the Lagrangian points at the rotor model mesh.
  - **Calc_F_lagr_ACL**
    Computes the actuator line forces at each element of the Lagrangian mesh.
  - **Calc_forces_ACL**
    Computes the overall turbine forces.
  - **Calc_F_eul**
    Transfers the forces from the Lagrangian mesh to the fluid mesh.

Chapter 6

Applications

6.1 3D Sloshing in a Rectangular Tank

6.1.1 Case Definition

This test aims to validate the implementation of the level set method which is used in the code to track the motion of the free surface. The test consists of a 3D sloshing of liquid in a tank with the dimension of $L \times L$ and a mean flow depth of D (see Figure 6.1). The initial free surface elevation is of Gaussian shape and is given by

$$\varrho(x, z) = D + \eta_0(x, z), \tag{6.1}$$

where

$$\eta_0(x, z) = H_0 \exp\left\{\left(x - \frac{L}{2}\right)^2 + \left(z - \frac{L}{2}\right)^2\right\}, \tag{6.2}$$

H_0 is the initial hump height, and κ is the peak enhancement factor.

In the computation, the following parameters are used: $L = 20$, $\kappa = 0.25$, and $H0 = 0.1$. The free-slip boundary conditions are applied at all boundaries. This condition is signified by the value 10 in bcs.dat. The number of grid points in the x, y, and z directions are 201, 41, and 201, respectively. Uniform grid spacing of $\Delta x = \Delta z = 0.1$ is employed in the x and z directions, while stretched grid is used in the y direction. The initial hump height (0.1 m) is resolved by approximately five vertical grid nodes. The time step used for the computation is $\Delta t = 0.001s$ and the value of ϵ (free surface thickness) is set to $0.03m$. The solution of the free surface elevation at the center of the tank is given in Figure 6.3.

Further details about the simulation as well as the analytical solution of the problem can be found in Kang and Sotiropoulos [15].

6.1.2 Main Parameters

The main parameters in the control file for setting this test case are listed in Table 6.1.2.

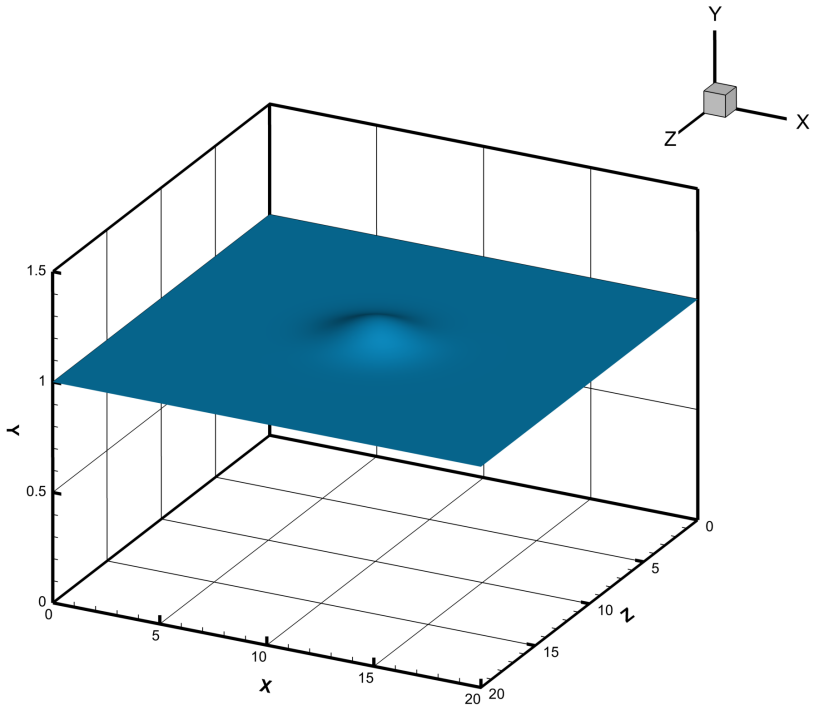


Figure 6.1: Schematic description of the domain and the initial condition of the free surface.

Table 6.1: Parameters in control.dat file for the sloshing case.

Parameter	Option in control file	Value
Time step size	dt [sec]	0.001
Read xyz.dat type mesh	xyz	1
Activate level set method	levelset	1
Set density of water and air	rho0, rho1 [kg/m3]	1000, 1
The viscosity is neglected	inv	1
Set gravity in y direction	gy [m2/s]	-9.81
Activate 3D Sloshing test case option	sloshing	2
Initialize flat free surface at elevation set by level_in_height	level_in	2
Set level set interface thickness	dthick [m]	0.03
Number of level set reinitializations	levelset_it	15
Reinitialization time step size	levelset_tau [sec]	0.05

By activating the sloshing option, two actions are implemented in the code. First, the initial condition of the distance function, and thus the free surface elevation, is set to the one corresponding to the present case. Then, at every time step after the flow solution is updated, it computes the analytical solution of the free surface position at the tank center and exports it along with the computed solution in a file named sloshing.dat described below.

Also note that because the level set method is active, the equations are solved with dimensions.

6.1.3 Validation

The output value for comparison in this test case is the time evolution of the free surface elevation at the tank center. This value can be checked at the post-processed data file containing the whole domain solution, however, this information is only available for the few exported time steps defined by the input option -tio. Another approach that is more convenient for evaluating the surface elevation in the tank center is by checking the output data file (sloshing.dat). This file exports information at every time step and consists of an ASCII data file with four columns. The first column is the simulation time [sec], the second is the computed surface elevation [m] at the tank center, the third is the expected theoretical solution also at the tank center, and the last is the error. If the purpose of running this test case is for validation only, it is not necessary to post-process any flow-field data. The surface elevation at the tank center is plotted in Figure 6.3.

Although Figure 6.3 shows the solution for 60000 time steps (60s), for validation purpose, it is not necessary to run the simulation for that long. With the proposed time step size, between 6000 and 8000 iterations (equivalent to 6 to 8 sec) should be sufficient to demonstrate that the solution is valid. As a reminder, the total number of time steps for which the code runs is specified at the control.dat file with the variable "totalsteps".

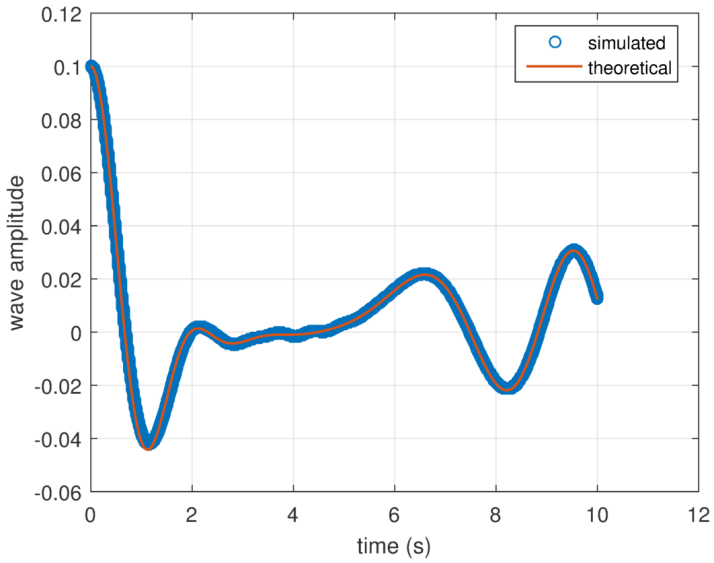


Figure 6.2: Comparison of the computed and analytic free surface elevation at the center of the tank. (symbol: computed solution, solid line: analytical solution).

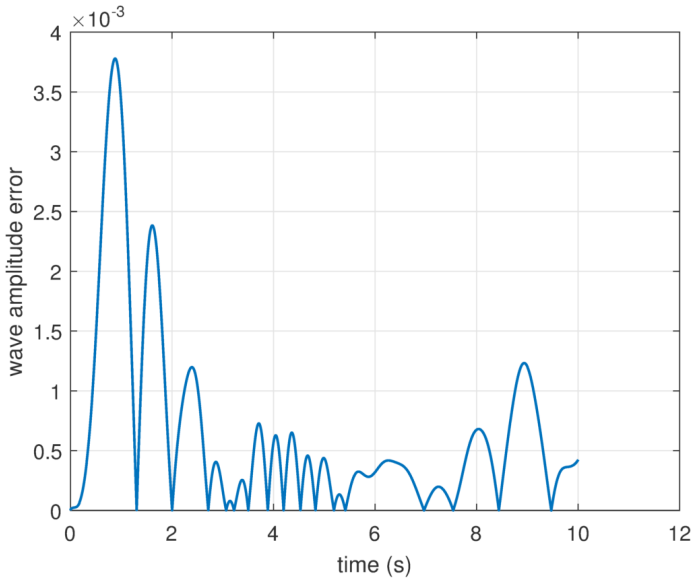


Figure 6.3: Error in the computed free surface elevation.

6.2 2D VIV of an Elastically Mounted Rigid Cylinder

6.2.1 Case Definition

Vortex induced vibration (VIV) of an elastically mounted rigid cylinder is a well-known benchmark case for validating FSI codes. The schematic description of the problem is shown in Figure 6.4. The problem consists of a 2D rigid cylinder of diameter D that is elastically mounted in a uniform flow of velocity U. The cylinder is allowed to move in the direction perpendicular to the flow with one degree of freedom. Additional details can be found in Borazjani, Ge, and Sotiropoulos [16].

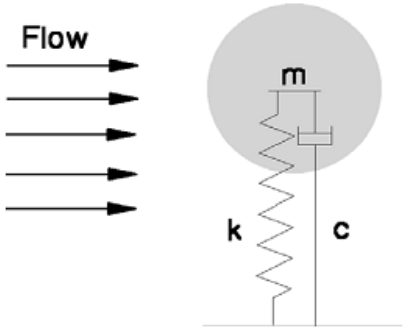


Figure 6.4: Schematic description of the elastically mounted rigid cylinder in the free stream. This figure was reproduced from [16].

A rectangular computational domain with dimensions $32D \times 16D$ is considered.

The cylinder is initially positioned at $8D$ from the inlet and centered. The boundary conditions for the side walls are slip wall (defined by 10 in bcs.dat), uniform flow is prescribed at the inlet (5 in bcs.dat), and a convective boundary condition is applied at the outlet (4 in bcs.dat). A non-uniform grid is used with 281×241 nodes in the streamwise and span-wise directions, respectively. The code requires at least 5 grid nodes (current test case employs 6) in the span-wise direction to carry out a two-dimensional simulation since the code is fully three-dimensional in addition to slip-wall boundary conditions along the span-wise walls. A square box with constant grid spacing of $0.02D$ and 50×50 nodes centered on the cylinder is used. Outside of the box the grid is gradually stretched towards the boundaries.

6.2.2 Main Parameters

The main parameters in the control file for setting this case are listed in Table 6.2. See Borazjani, Ge, and Sotiropoulos [16] for the definition and details of the parameters. In contrast to the previous case the level set method is not employed and the solved equations are all non-dimensional.

Table 6.2: Parameters in control.dat file for the VIV case.

Parameter	**Option in control file**	**Value**
Time step size	dt	0.01
Reynolds number ($RE = U * D/$)	ren	150
Reduced velocity of 6	red_vel	1.0472
Raduced mass of 2	mu_s	0.25
Cylinder damping	damp	0.0
Activate IB method	imm	1
Use of one body	body	1
Activate FSI module	fsi	1
Activate proper degree of freedom	dgf_y	1
Apply an initial translation of the cylinder to the actual position. The cylinder mesh was initially defined at the origin and needs to be translated to the desired location.	y_c, z_c	8.0, 8.0

6.2.3 Validation

The VIV case can be validated by comparing the position of the cylinder in time. Similar to the previous case, there is no need to post-process the flow field data in order to get the cylinder position. The cylinder position is provided at every time step in the FSI_position00 file. The FSI_position00 file is a text file containing information about the immersed body location, velocity, and forces for the linear

degrees of freedom in the x, y, and z direction. The file consists of 10 columns as follows:

$$ti, Y_x, \dot{Y}_x, F_x, Y_y, \dot{Y}_y, F_y, Y_z, \dot{Y}_z, F_z, \tag{6.3}$$

where ti is the time step number, Y is the body position with respect to its initial position Y/D, $\dot{Y}$ is the body velocity, and F is the force that the fluid imparts to the immersed body. For this test case, 6000 time steps should be sufficient.

In the test case folder, a successful run file name _FSI_position00_good_run is provided to quickly validate the case. Figure 6.5 provides a plot showing a comparison with the provided successful run file. The maximum amplitude of the cylinder is approximately 0.49.

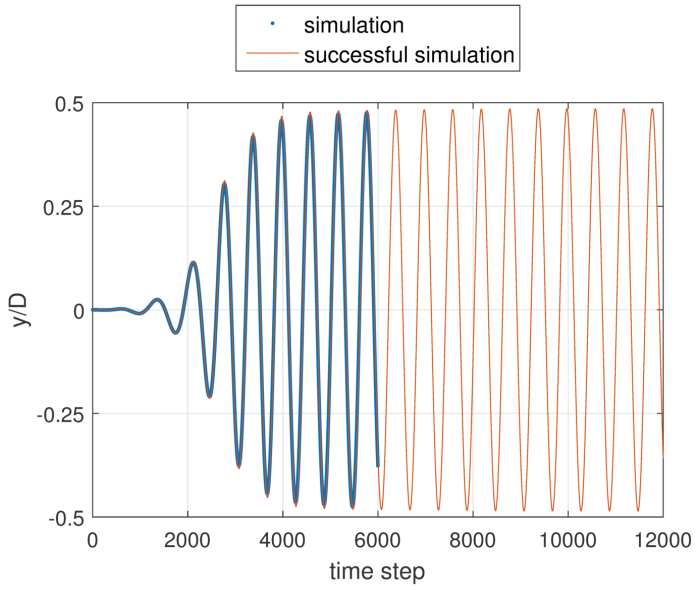


Figure 6.5: Displacement of the cylinder for a successful simulation.

6.3 2D Falling Cylinder (Prescribed Motion)

6.3.1 Case Definition

This test case is to validate the integration of the CURVIB and level set methods. The case involves a body moving across the air/water interface with prescribed motion. Note that the FSI algorithm is not used currently. This test case considers an infinitely long 2D cylinder of radius $R = 1m$ moving with constant downward speed in an infinitely wide domain and crossing the free surface from a gas to liquid phase.

The configuration provided currently is identical to that simulated by Yang and Stern [28] where an identical cylinder, initially positioned above the free surface at a distance $h = 1.25m$, moves downwards with constant velocity of $u = -1m/s$. The liquid and gas densities are $\rho_0 = 1kg/m^3$ and $\rho_1 = 1 \times 10^3 kg/m3$, respectively. Both

the liquid and gas are considered inviscid, while the gravity is set to $gz = -1m/s^2$ and the time is normalized as $T = ut/h$.

The 2D computational domain is $40R \times 24R$ in the horizontal and vertical directions, respectively. A non-uniform grid consisting of an inner region, centered on the cylinder, within which the mesh is uniform and an outer region where the grid is gradually stretched. The inner region is the rectangular domain defined by $[-5, 5]$ and $[-4, 2.6]$ in the horizontal and vertical directions, respectively. Within this inner domain uniform grid spacing is employed along both directions, which is equal to $0.05R$. Outside of this inner domain the mesh is stretched gradually away from the cylinder using the hyperbolic stretching function with a stretching ratio kept below 1.05. The total number of nodes is $360 \times 255 \times 6$. A time step of $0.01s$ and a free surface thickness ϵ of $0.04m$ are used. For more detail about this test case refer to Calderer et al. [2]

6.3.2 Main Parameters

The main parameters in the control file for setting this case are listed in Table 6.3.

Table 6.3: Parameters in control.dat file for the 2D falling cylinder with prescibed motion test case.

Parameter	**Option in control file**	**Value**
Time step size	dt [s]	0.01
Activate level set method	levelset	1
Set density of liquid and gas	rho0, rho1 [kg/m3]	1, 0.001
Set gravity in z direction	gz [m/s2]	-1.0
Thickness of the free surface interface	dthick [m]	0.04
Initialize flat free surface at elevation set by level_in_height	level_in	1
Initial free surface elevation	level_in_height [m]	0.0
Set levelset interface thickness	levelset_it	10
Number of level set reinitializations	levelset_tau [s]	0.05
Activate IB method	imm	1
Use of one body	body	1
Activate fluid structure interaction module	fsi	1
Activate prescribed motion function	forced_motion	1
Activate Falling cylinder case. This option basically prescribes the velocity of the body to be constant and downward.	fall_cyll_case	1
Activate proper degree of freedom	dgf_ay	1
Initial translation of the ib at the right position	z_c	1.25

6.3.3 Validation

The falling cylinder case can be validated by observing the free surface elevation at four instances in time as shown in Figure 6.6. These plots are from the post processed cylinder position (Nv) and free surface (level = 0) at time step T = 125, 250, 375, and 500.

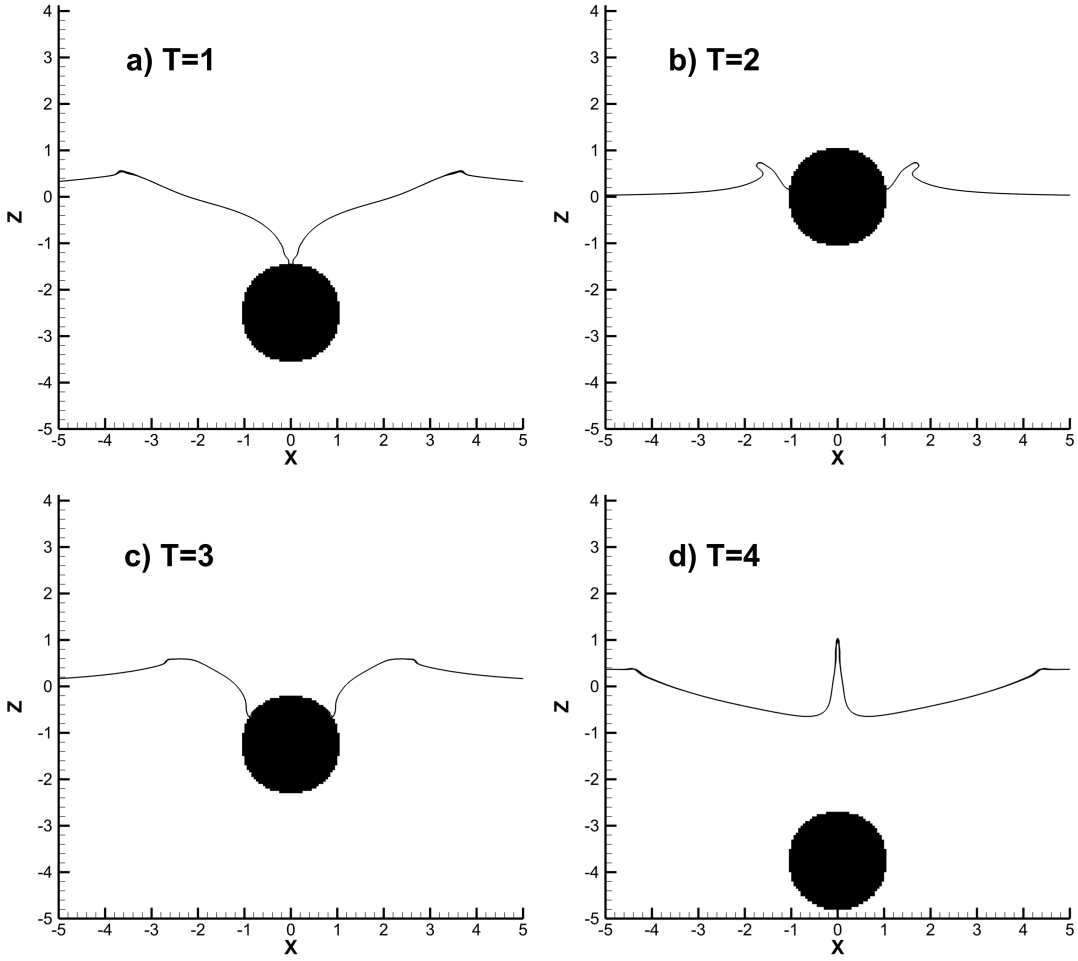


Figure 6.6: Water entry of a horizontal circular cylinder moving with prescribed velocity. Free surface position at different non-dimensional times T calculated by the present method.

6.4 3D Heave Decay Test of a Circular Cylinder

6.4.1 Case Definition

To validate the coupled FSI algorithm for simulating complex floating structures a free heave decay test of a horizontal cylinder is provided. This same test case

was studied experimentally by Ito [29]. A horizontal circular cylinder of diameter $D = 0.1524m$ and density $\rho = 500kg/m^3$ is partially submerged with its center positioned $0.0254m$ above the free surface of a rectangular channel (see Figure 6.7 for a schematic representation).

The computational domain is a $27.4m$ long, $2.59m$ wide, and $1.22m$ deep channel with initially stagnant water. The cylinder movement is restricted to the vertical degree of freedom while allowed to oscillate freely. A non-uniform grid is employed with $436 \times 8 \times 261$ nodes in the horizontal, vertical, and span-wise directions, respectively.

The grid is uniform with spacing equal to $0.02D$ in a rectangular region centered around and containing the body defined by $[0.3, 0.3]$ in the horizontal direction and $[0.2, 0.2]$ in the vertical direction. The grid is stretched using a hyperbolic function, where the ratio never exceeds 1.05, in the domain outside of the uniform grid region. The interface thickness used is $0.065D$ and the simulation was carried out employing the loose coupling FSI algorithm and a time step size of $0.0005s$.

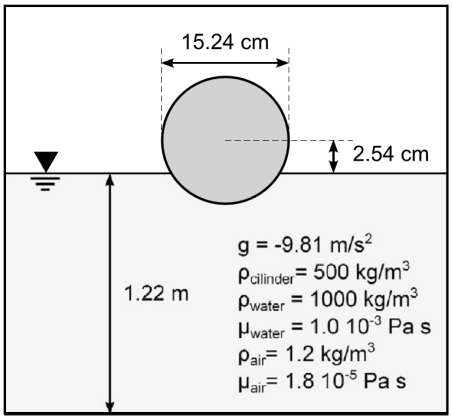


Figure 6.7: Schematic description of the cylinder case. This figure was reproduced from [2].

6.4.2 Main Parameters

The main parameters in the control file for setting this case are listed in table 6.4.

6.4.3 Validation

In the heave decay test case of a horizontal cylinder it is simplest to compare the vertical position of the cylinder in time as shown in Figure 6.8 using the FSI_position00 file's vertical position column as described in the VFS Manual Section \$3.3.3 and in the test case of the 2D VIV of an elastically mounted rigid cylinder in section \$6.2. An FSI_position00_good_run file is provided for comparison and validation.

Table 6.4: Parameters in control.dat file for the 3D heave decay of a circular cylinder test case.

Parameter	**Option in control file**	**Value**
Time step size	dt [s]	0.0005
Activate Dynamic Smagorinski LES model	les	2
Activate level set method	levelset	1
Set density of water and air	rho0, rho1 [kg/m3]	1000, 1.0
Set viscosity of water and air	mu0, mu1 [Pa s]	1e-3, 1.8e-5
Set gravity in z direction	gy [m/s2]	-9.81
Thickness of the free surface interface	dthick [m]	0.006
Initialize flat free surface at elevation set by level_in_height	level_in	2
Initial free surface elevation	level_in_height [m]	0.0
Set level set interface thickness	levelset_it	20
Number of level set reinitializations	levelset_tau [s]	0.01
Activate IB method	imm	1
Use of one body	body	1
Activate fluid structure interaction module	fsi	1
Activate proper degree of freedom	dgf_y	1
Mass of the cylinder	body_mass [kg]	23.63
Initial translation of the ib at the right position	y_c [m]	0.0254

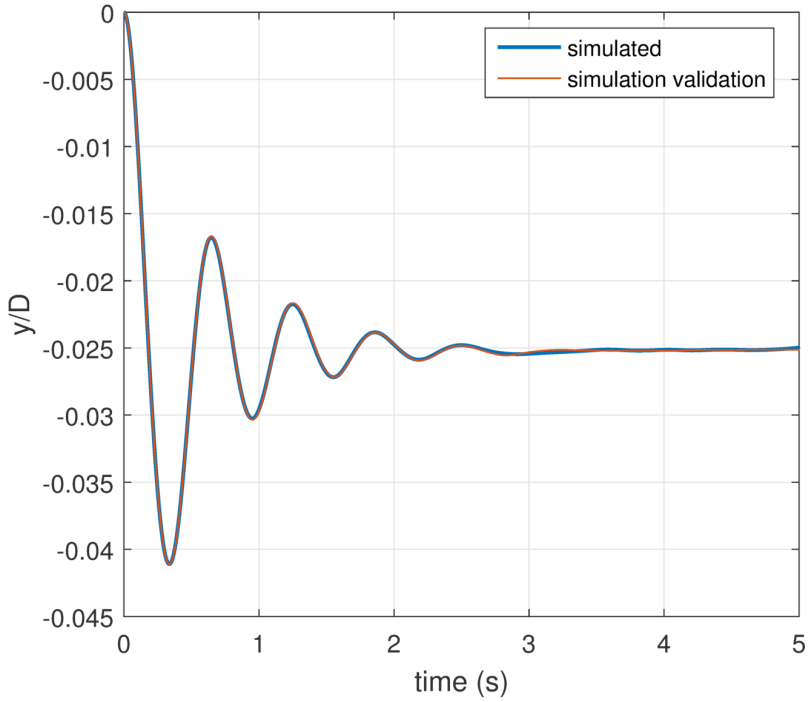


Figure 6.8: Normalized position of the cylinder in time computed with the present code.

6.5 2D Monochromatic Waves

6.5.1 Case Definition

The simulated generation and propagation of a progressive monochromatic wave in a 2D rectangular channel is used to validate the wave generation method of Guo and Shen [30] implemented in this code. A linear wave of amplitude $0.01m$ and wavelength of $1.2m$, for which the analytic solution is known from linear wave theory, is considered. A two-dimensional domain of length equal to $24m$, water depth of $2m$, air column above the water of $1m$, and gravity equal to $gy = -9.81m/s^2$ is simulated using a non-uniform grid size of 40×177 in the longitudinal and vertical direction, respectively. The grid is uniform in a rectangular region centered on $z = 0$ and spanning $24m$ along the z direction ($[12, 12]$), and containing the free surface along the vertical direction y ($[0.15, 0.15]$). Within this region the grid spacing is 0.06 and 0.005 in the horizontal and vertical directions, respectively, while outside of this region the grid spacing increases progressively with a stretching ratio limited to 1.05.

The time step of the simulation is $0.002s$ with the thickness of the interface set to $0.02m$. The source region is centered on the origin. Sponge layers with length equal to $3m$ are defined at the two ends of the computational domain and slip boundary conditions are implemented at the bottom and top boundaries (10 in bcs.dat).

Details of the method implementation are given in Calderer et al. [3]. The analytical solution is

$$\eta(z,t) = Acos(\omega t - kx), \tag{6.4}$$

where η is the surface elevation, A is the wave amplitude, $k = 2\pi/L$ is the wave number, d is the water depth, and ω is the angular frequency solved for with the dispersion relation as follows:

$$\omega = \sqrt{kgtanh(kd)}. \tag{6.5}$$

6.5.2 Main Parameters

The main parameters in the control file for setting this case are listed in Table 6.5.

6.5.3 Validation

The free surface elevation (height of level = 0 in the post-processed data files) and its comparison with the analytical solution are presented in Figure 6.12. Note that in the source region the computed results are not expected to follow the analytical free surface pattern.

Table 6.5: Parameters in control.dat file for the 2D monochromatic wave test case.

Parameter	**Option in control file**	**Value**
Time step size	dt [s]	0.002
Activate level set method	levelset	1
Set density of water and air	rho0, rho1 [kg/m3]	1000, 1.0
Set viscosity of water and air	mu0, mu1 [Pa s]	1e-3, 1.8e-5
Set gravity in z direction	gy [m/s2]	-9.81
Thickness of the free surface interface	dthick [m]	0.02
Initialize flat free surface at elevation set by level_in_height	level_in	1
Initial free surface elevation	level_in_height [m]	0.0
Set levelset interface thickness	levelset_it	15
Number of level set reinitializations	levelset_tau [s]	0.05
Activate the wave generation method for single wave frequency	wave_momentum_source	2
Time step for which wave generation method begins	wave_ti_start	1
Wave Direction. If 0rad waves travel in x direction	wave_angle_single [rad.]	0
Wavenumber	wave_K_single [1/m]	5.23599
Water depth	wave_depth [m]	2
Wave amplitude	wave_a_single [m]	0.01
Activate wave sponge layer method at the x boundaries for wave suppression.	wave_sponge_layer	1
Length of the sponge layer	wave_sponge_xs [m]	3
Starting coordinate of the first x sponge layer.	wave_sponge_x01 [m]	-12
Starting coordinate of the second sponge layer.	wave_sponge_x02 [m]	9

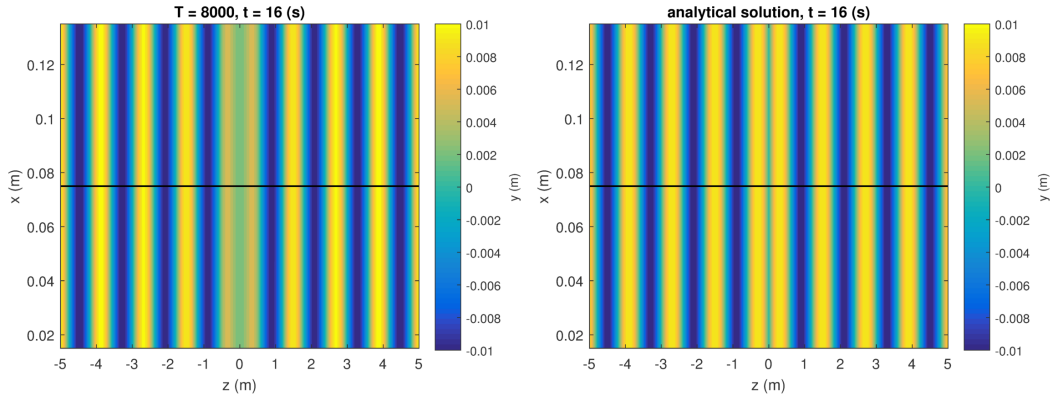


Figure 6.9: Generation of monochromatic waves. Contours of free surface elevation, computed (left) and analytical solution (right).

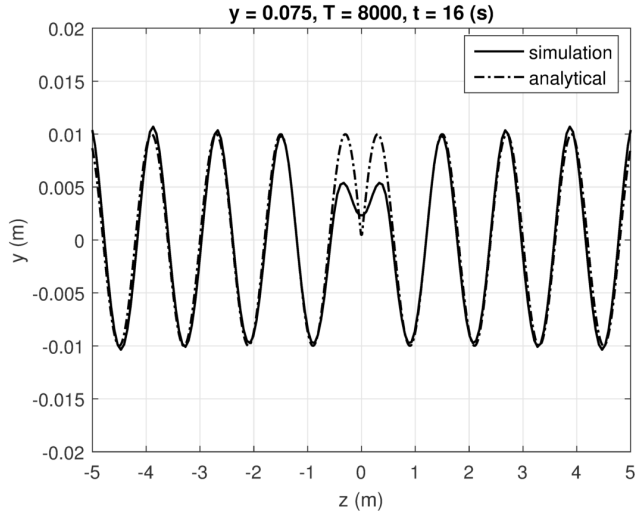


Figure 6.10: Generation of monochromatic waves. Computed and analytical free surface elevation.

6.6 3D Directional Waves

6.6.1 Case Definition

This test case validates the implemented forcing method for wave generation by generating a linear directional wave field in a 3D basin of constant depth. The wave amplitude is $A = 0.01m$, the wavelength is $L = 1.2m$, and the wave direction is $\beta = 25deg$. The domain length is $24m$ ($-12m \leq z \leq 12m$) in the longitudinal direction, $12m$ ($-6m \leq x \leq 6m$) in the span-wise direction, and the depth of water and air is $2m$ and $1m$, respectively. A non-uniform grid size of $121 \times 139 \times 201$ in the x, y, and z directions, respectively, is employed consisting of an inner rectangular region with uniform grid spacing and an outer region within which the mesh is gradually stretched towards the boundaries. The inner region ($-6m \leq z \leq 6m$, $-6m \leq x \leq 6m$, $-0.1m \leq y \leq 0.1m$), which contains the source region and part of the propagated waves, has a constant grid spacing of $0.1m$, $0.1m$, and $0.005m$, in the x, y and z directions, respectively. The source region is centered on $z = 0$ and spans the entire domain along the x direction. The time step is $0.005s$, the interface thickness is $0.02m$, and the gravity is $g = -9.81m/s^2$. The sponge layer method with length $3m$ is applied at the Z boundaries and periodic boundary conditions is applied at the X boundaries. Details of the method implementation are given in Calderer et al. [3]. The analytical solution is

$$\eta(z, x, t) = Acos(\omega t - k_x x - k_z z) \tag{6.6}$$

where η is the surface elevation, A is the wave amplitude, $k = 2\pi/L$ is the wave number, $k_x = kcos(\beta)$, $k_y = ksin(\beta)$, d is the water depth, and ω is the angular frequency solved for with the dispersion relation as follows

$$\omega = \sqrt{kgtanh(kd)}. \tag{6.7}$$

6.6.2 Main Parameters

The main parameters in the control file for setting this case are listed in Table 6.6.

6.6.3 Validation

The free surface elevation (height of level = 0) and its comparison with the analytical solution is presented in Figure **??**.

Table 6.6: Parameters in control.dat file for the 3D directional wave test case.

Parameter	**Option in control file**	**Value**
Time step size	dt [s]	0.002
Activate level set method	levelset	1
Set density of water and air	rho0, rho1 [kg/m3]	1000, 1.0
Set viscosity of water and air	mu0, mu1 [Pa s]	1e-3, 1.8e-5
Set gravity in z direction	gy [m/s2]	-9.81
Thickness of the free surface interface	dthick [m]	0.04
Initialize flat free surface at elevation set by level_in_height	level_in	2
Initial free surface elevation	level_in_height [m]	0.0
Set number of pseudo time steps to solve the reinitialization equation for the level set method	levelset_it	15
Number of level set reinitializations	levelset_tau [s]	0.05
Activate the wave generation method for single wave frequency	wave_momentum_source	2
Wave Direction. If set to $0rad$ waves travel in z direction	wave_angle_single [rad.]	0.5235988
Wavenumber	wave_K_single [1/m]	5.23599
Water depth	wave_depth [m]	2.0
Wave amplitude	wave_a_single [m]	0.01
Activate wave sponge layer method at the x boundaries for wave suppression.	wave_sponge_layer	1
Length of the sponge layer	wave_sponge_zs [m]	1.2
Starting coordinate of the first z sponge layer.	wave_sponge_z01 [m]	-12.0
Starting coordinate of the second sponge layer.	wave_sponge_z02 [m]	10.6

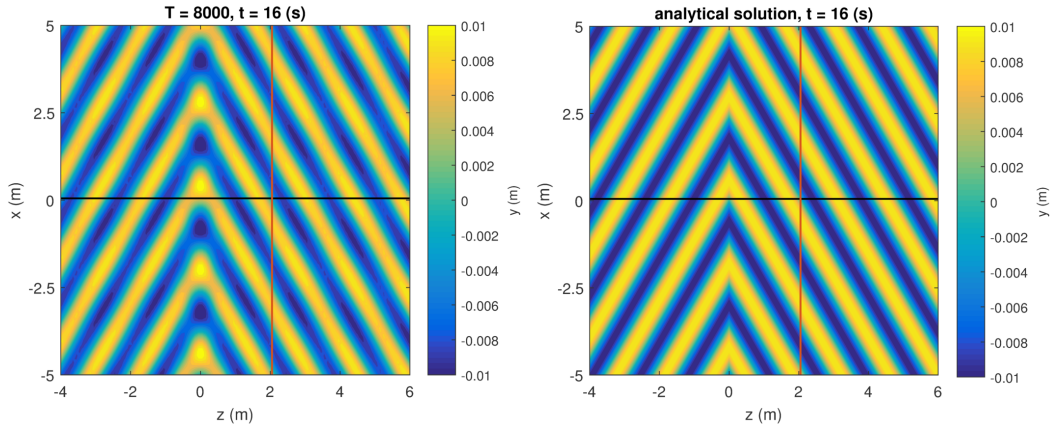


Figure 6.11: Generation of directional progressive waves in a 3D tank. Contours of computed (left) and analytical (right) free surface elevation.

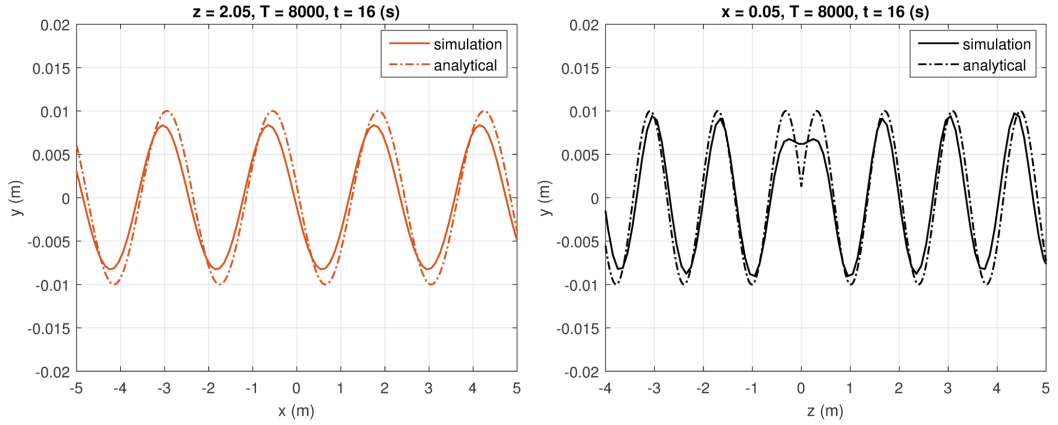


Figure 6.12: Generation of directional progressive waves in a 3D tank. Profiles of surface elevation.

6.7 Floating Platform Interacting with Waves

This test case involves a barge style floating platform model that is installed in the Saint Anthony Falls Laboratory main channel. The channel acts as a wave basin.

The floating platform has a cylindrical shape and is allowed to move in two degrees of freedom, pitch and heave. Rotations are in respect to the center of gravity of the structure. The platform has two small cylindrical masses located underneath which are also considered in this simulation.

In this particular case we study the response of the structure under a given incoming wave frequency. Monochromatic waves of amplitude $0.00075m$ and wave-number 0.8039 are generated at the source region located at $z = 0$. The platform is located at $z = 30m$ and oscillates as a response of the forces induced by waves both in the vertical direction Y and rotating along the X axis. Using the dispersion relation and considering that the water depth is $1.37m$ the wave period is $T = 2.5s$.

The value that we are interested for validating this simulation is the maximum amplitude of the oscillation in both the pitch and heave directions. These can be seen at the files FSI_angle00 and FSI_position00. Figure 6.13 shows the experimental results for several incoming wave frequencies known as Response amplitude operator (RAO). In Figure 6.13 the values are normalized by the incident wave height as follows

$$RAO_{heave} = Y_{max}/H_{wave} \tag{6.8}$$

and

$$RAO_{pitch} = \phi_{max}/H_{wave} \tag{6.9}$$

where Y_{max} is the maximum vertical displacement measured from peak to peak, H_{wave} is the incident wave height, and ϕ_{max} is the maximum rotation angle between two consecutive peaks.

Note that for this case the computed wave amplitude may be slightly inferior to that specified at the control file. This result is due to the fact the waves are in a shallow water regime. This fact will not alter the results as long as the actual simulated wave height is considered in the normalization.

6.8 Clipper Wind Turbine

6.8.1 Case Definition

This test case is for introducing and validating the actuator line model for turbine parameterization. The test case involves the simulation of the Clipper Liberty 2.5MW wind turbine which was built during the EOLOS project and is located in UMore Park, Rosemount, MN. The turbine has a diameter of $96m$ and a hub height of $80m$. Details can be found in [11]. For validation purpose we propose the case with uniform inflow which makes the case simple as it does not require any precursor simulation and the boundary conditions at the top wall, bottom wall and side walls are free slip. Two cases with different TSR, 5.0 and 8.0, are tested.

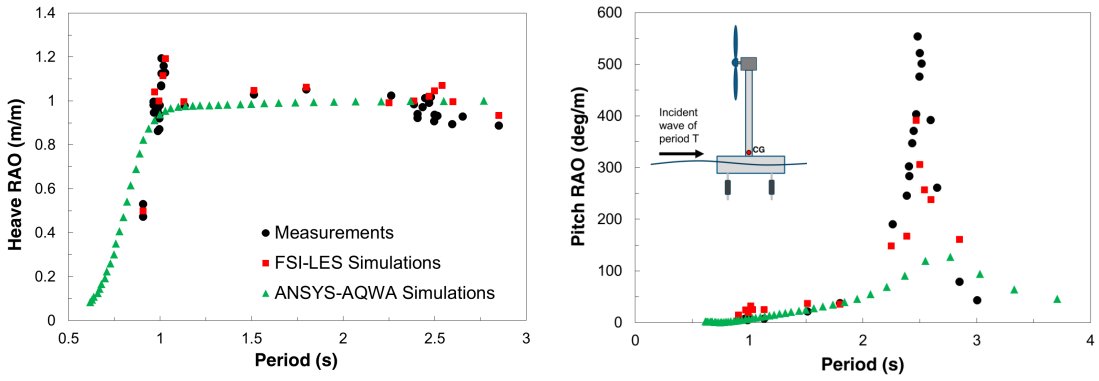


Figure 6.13: Response amplitude operator for the floating turbine.

The simulation is carried in a non-dimensional form being the reference length the turbine diameter. The grid dimensions are 2 units in the y and x directions, and 4 units in the z direction, which corresponds to a dimensional domain of 192m and 384m, respectively. The grid is uniform and the spacing is 0.05 units in all three directions which is equivalent to a grid size of $41 \times 41 \times 81$. Note that the grid file (grid.dat or xyz.dat) has already non-dimensionalized size. The simulation is set with a unit non-dimensional velocity equivalent to a real velocity of 8m/s.

In contrast to the fluid mesh, the actuator line mesh (acldata000) can be constructed with the real turbine dimenions ($96m$) and non-dimensionalized by setting the turbine reference length option "reflength_wt" in control.dat to 96.0. This will divide the actuator line mesh dimensions by "reflength_wt". Alternatively one could generate a rotor mesh already with the non-dimensionalised units and choose "reflength_wt" equals to 1.0.

6.8.2 Main Case Parameters

Since the main solver parameters have already been discussed in previous test cases, only the parameters related to the wind turbine rotor model will be addressed. When using the rotor model the control options for setting the case are located not only in control.c but also in Turbine.inp. The former parameters are summarized in Table 6.7, with the latter in Table 6.8. The parameters in Turbine.inp that are not discussed in the Table 6.8 are not used in the simulation.

This test case requires averaging, and therefore has to be executed in two stages as described in Section $4.2.1. In the first stage the flow is fully developed, while in the second stage the time averaging is performed.

For developing the flow field, it is sufficient to perform 5000 time steps. For time-averaging the flow field, 2000 time steps are enough. For time-averaging, set the option in the control file "averaging"" to 3, rstart to 0, and rename the result files from the last instantaneous time step (ufield005000.dat, vfield005000.dat, pfield005000.dat, nvfield005000.dat, and cs_field005000.dat) to the value corresponding to time zero (

Table 6.7: Parameters in control.dat file for the Clipper wind turbine.

Parameter	**Option in control file**	**Value**
Time step size	dt [s]	0.0025
Activate the actuator line model	rotor_modeled	6
Number if blades in the rotor	num_blade	3
Number of foil types along the blade	num_foiltype	5
Number of turbines	turbine	1
Reference length of the turbine	reflength_wt	96.
Nacelle diameter	r_nacelle	1.3
Reference velocity of the case. It is only used for dimensionalizing the turbine model output files	refvel_wt	8.0
Distance upstream of the turbine in rotor diameters where the turbine incoming velocity or reference velocity is computed	loc_refvel	0.5
Direction to which the turbine points to	rotatewt	1
Type of smoothing delta function	deltafunc	10
Delta function width in cell units	halfwidth_dfunc	2.0
Activate constant turbine rotation mode	fixturbineangvel	1

Table 6.8: Parameters in Turbine.inp file for the Clipper wind turbine.

Parameter	**Option in Turbine.inp file**	**Value**
Normal direction of the turbine rotor plane.	nx_tb, ny_tb, nz_tb	0.0 0.0 1.0
The turbine rotor initial translation.	x_c, y_c, z_c	0.0 0.0 0.0
Tip speed ration	Tipspeedratio	5
Radius of the rotor corresponding to the acldata000 mesh	r_rotor	48.0
Tip speed ratio when FixTipSpeedRatio option in control.dat is active, otherwise is not used	TSR_max	8.3
Rotor angular velocity when fixturbineangvel option in control.dat is active, otherwise is not used	angvel_fixed	10.0
Angle of pitch of the blades in degrees	pitch[0]	1.0

ufield000000.dat, vfield000000.dat, ...).

Also, when restarting the flow field for averaging, set rstart_turbinerotation as 1 and rename TurbineTorqueControl005001_000.dat as TurbineTorqueControl000001_000.dat.

6.8.3 Validation

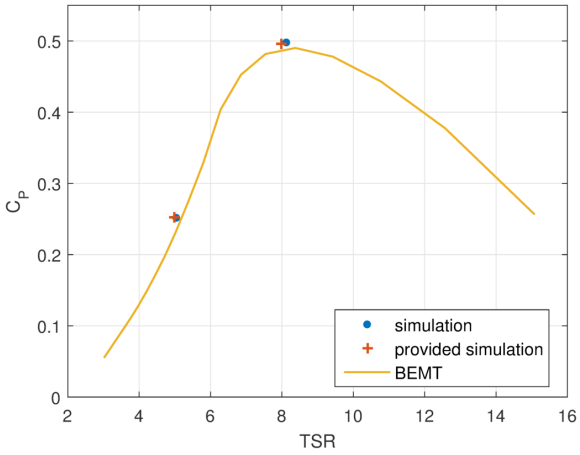


Figure 6.14: Cp coefficient of the Clipper turbine as a function of the TSR.

For validation the case the power coefficient (Cp) of the turbine is compared with the theoretical power coefficient obtained using blade element momentum theory.

A folder named "Tecplotfiles" is provided to assist in generating the Cp comparison. Executing the tecplot macro file ReadSaveCp.mcr a file CP.txt is created with the averaged Cp coefficient. The CP value for the TSR 5 and TSR 8 cases is 0.25 and 0.50, respectively, as shown in figure 6.14.

6.9 Model Wind Turbine Case

6.9.1 Case Definition

This test case is to further demonstrate the validity of the actuator line model by analyzing the turbulence statistics in the turbine wake. The test case consists of a miniature wind turbine simulation with geometry equivalent to the model turbine tested experimentally in the Saint Anthony Falls wind tunnel by [31]. The diameter of the turbine is $0.15m$ and the turbine hub height is $0.125m$. The rotor blade cross section is approximated as a NACA0012, although the real blade geometry is unknown. The tower is neglected and the nacelle is modeled by extending the actuator line effect to the center point of the rotor. The inflow velocity at hub height is $2.2m/s$ and the TSR is 4.1. Reynolds number based on rotor diameter D and the incident velocity at the hub height U_{hub} is 4.2×10^4. For more details about the case and the turbine geometry see [22].

The computational domain dimensions are $30D$ in the streamwise direction, $12D$ in the spanwise direction, and $3D$ in the vertical direction. The turbine is positioned $2D$ after the inlet plane and centered on the transverse direction. The grid is considered uniform along the spanwise and vertical directions. In the streamwise direction,

the mesh is uniform from the inlet to 12D downstream and stretched towards the outlet. Grid size is $201 \times 121 \times 31$, which is equivalent to 10 grid points per turbine diameter.

The velocity profile specified at the inlet is from a pre-computed simulation consisting of a channel flow. An example of such channel flow simulation is provided in the test case in Section $6.10. The channel flow case presented in $6.10 is illustrative of how to prepare fully developed inlet flow conditions, although the case parameters may not exactly coincide with the precursor simulation used for the present simulation.

The bottom wall cannot be resolved with the current grid resolution and is treated with a wall model.

6.9.2 Main Case Parameters

Table 6.9: Parameters in control.dat file for the wind turbine model simulation.

Parameter	**Option in control file**	**Value**
Time step size	dt [s]	0.001
Activate the actuator line model	rotor_modeled	6
Number if blades in the rotor	num_blade	3
Number of foil types along the blade	num_foiltype	2
Number of turbines	turbine	1
Reference length of the turbine	reflength_wt	1.5
Nacelle diameter	r_nacelle	0.0
Reference velocity of the case. It is only used for dimensionalizing the turbine model output files	refvel_wt	8.0
Distance upstream of the turbine in rotor diameters where the turbine incoming velocity or reference velocity is computed	loc_refvel	1.0
Direction to which the turbine points to	rotatewt	1
Type of smoothing delta function	deltafunc	0
Delta function width in cell units	halfwidth_dfunc	2.0
Activate constant turbine rotation mode	fixturbineangvel	1

As in the previous test case, time averaging is required (see Section $4.2.1). For developing the flow field, it is sufficient to perform 5000 time steps. For time-averaging the flow field, 2000 time steps are sufficient. For time-averaging, set the option in the control file "averaging"" to 3, rstart to 0, and rename the result files from the last instantaneous time step (ufield005000.dat, vfield005000.dat, pfield005000.dat, nvfield005000.dat, and cs_field005000.dat) to the value corresponding to time zero (ufield000000.dat, vfield000000.dat, etc.).

Table 6.10: Parameters in Turbine.inp file for the wins turbine model simulation.

Parameter	**Option in Turbine.inp file**	**Value**
Normal direction of the turbine rotor plane.	nx_tb, ny_tb, nz_tb	0.0 0.0 1.0
The turbine rotor initial translation.	x_c, y_c, z_c	1.0 0.0833 0.2
Tip speed ration	Tipspeedratio	4.0
Radius of the rotor corresponding to the acldata000 mesh	r_rotor	0.025
Tip speed ratio when FixTipSpeedRatio option in control.dat is active, otherwise is not used	TSR_max	8.3
Angle of pitch of the blades in degrees	pitch[0]	1.0

Also, when restarting the flow field for averaging, set rstart_turbinerotation as 1 and rename TurbineTorqueControl005001_000.dat as TurbineTorqueControl000001_000.dat.

6.9.3 Validation

In this test case, vertical profiles of velocity and turbulence statistics at different downstream locations are compared with measured data from the experiment of [31]. To facilitate the creation of these figures a Tecplot macro file named "ExtractLines_3D.mcr", that automatically generates the comparison figures, is provided in the sub-folder "Tecplotfiles". The macro file calls the averaged results file "ResultsXXXXXX.plt" (XXXXXX refers to the time step), which may have to be edited based on the results available. As an example, figure 6.15 shows the vertical profiles of averaged velocity, Reynolds shear stresses, and turbulence intensity $5D$ behind the turbine.

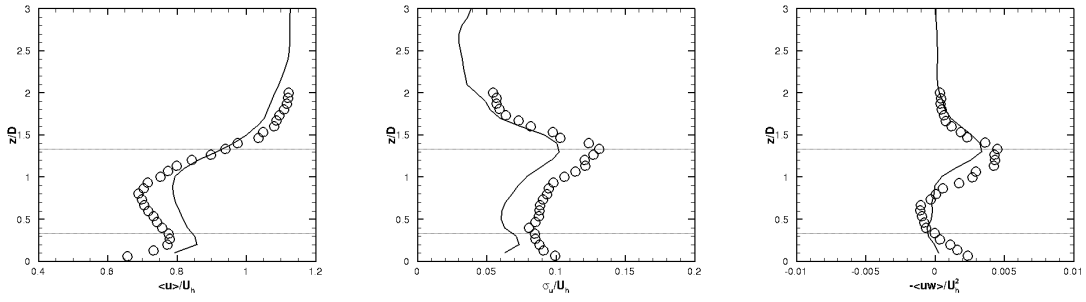


Figure 6.15: Vertical profiles of averaged stream-wise velocity (left), Reynolds stresses (center), and Turbulence intensity (right) at a distance $5D$ behind the turbine.

6.10 Channel Flow

6.10.1 Case Definition

This is the classical channel flow case as extensively described in [32]. A rectangular duct of height $H = 2h$ with uniform flow is considered. This test case can have a dual purpose: (1) to test the wall model at the bottom and/or top walls, or (2) as a precursor simulation to generate fully developed turbulent flow conditions to be fed at the inlet of other cases such as the wind turbine model case in section \$6.9.

The domain is $1.2m$ in the spanwise direction (x), $0.4m$ in the vertical direction (y), and $2m$ in the streamwise direction. A uniform grid is used with size $121 \times 41 \times 61$. Note that the height of $0.4m$ corresponds to the channel half height and slip-wall boundary conditions are used at the top boundary. Fully developed turbulent boundary conditions can be achieved using periodic boundary conditions in the streamwise and spanwise directions and no slip wall boundary condition at the bottom boundaries.

The flow can be characterized with the Reynolds number defined as

$$Re = 2\delta U/\nu, \tag{6.10}$$

where δ is the channel half height and U is the bulk velocity. The flow case can also be defined with the friction Reynolds number defined as

$$Re_\tau = u_\tau \delta/\nu, \tag{6.11}$$

with $u_\tau = \sqrt{\tau_w/\rho}$, and τ_w the shear stress at the wall. The Reynolds number flow can be related to the friction Reynolds number with the following expression

$$Re_\tau \approx 0.09 Re^{.88}. \tag{6.12}$$

In the present simulation the friction Reynolds number is $Re_\tau = 3000$ which using (6.12) corresponds to a Reynolds number flow of $Re = 137900$. With the kinematic viscosity of air taken as $\nu = 1.6 \times 10^{-5}$, using equation (6.10) the flow bulk velocity is 2.7584.

6.10.2 Main Case Parameters

The main parameters used in the present simulation control file are summarized in Table 6.11.

6.10.3 Validation

This test case is validated by comparing the vertical time-averaged profiles of streamwise velocity and turbulence intensity. Time averaging is performed as described in Section \$4.2.1 or as shown in previous test cases. The flow is first executed for about

Table 6.11: Parameters in control.dat file for the channel flow simulation.

Parameter	**Option in control file**	**Value**
Time step size	dt [s]	0.001
This parameter corresponds to $1/\nu$	ren	62500
Dynamic LES modelling active	les	2
Periodic boundary conditions in the streamwise and spanwise directions	kk_periodic, ii_periodic	1, 1
Initial condition set to uniform flow	inlet	1
The inlet flux to obtain a bulk velocity of 2.785. This takes in account the domain cross section area which is $0.48m^2$	flux	1.324
Introduces a random perturbation to the initial velocity field	perturb	1
Activate the wall model at the bottom boundary	viscosity_wallmodel	1
When active it exports to an external file the velocity field at the inlet plane at every time step. First it is set to 0 while the flow is developed. In the second stage the case is reinitialized with this option active.	save_inflow	0, 1
This parameter defines the number of times steps that are stored in an individual file	save_inflow_period	500
Folder where to store the exported velocity files. The code will create in the specified directory a folder named inflow.	path_inflow	"./"

4000 times steps to achieved developed conditions (check the file Kinetic_Energy.dat to ensure that the kinematic energy is stabilized). Then rename the flow field files to the corresponding to time step zero and restart the simulation with the time averaging option active. Also, when restarting, activate the option save_inflow to export the velocity field into the inlet.

A tecplot macro file is provided in the test case folder (ExtractLines_3D.mcr) which extracts vertical profiles of the data from the post-processed data file (Result010000-avg.plt). About 5000 times steps may be sufficient for time averaging although 10000 may provide smoother results. One can use the option ikavg equal to 1 to do space averaging in the streamwise and spanwise directions, when post-processing the average results in addition to the avg equals to 1 option.

Also note that the code output file provides the actual computed values of Re_{tau}

and u_τ. These values are printed at every time step and can be found, at the output file, by searching for the word "Bottom". The two values, are indicated as u* and Re*, respectively. However, the Re* value printed by the code for this case is not the real Re_{tau}. The code assumes that the vertical dimension is 1. So, to obtain the real value of Re_{tau}, the given Re* has to be multiplied by the channel half height δ, equals to 0.4 for this case. These two values are not exact and tend to oscillate in time. Errors of about 10% are within an admissible range.

The averaged stream-wise velocity profile can be compared with the logarithmic law of the wall (see figure 6.16) given by

$$u^+ = \frac{1}{\kappa} \ln(y^+) + 5.0 \tag{6.13}$$

where $\kappa \approx 0.41$ is the Von Karman constant, $u^+ = u/u_\tau$, and $y^+ = yu_\tau/\nu$.

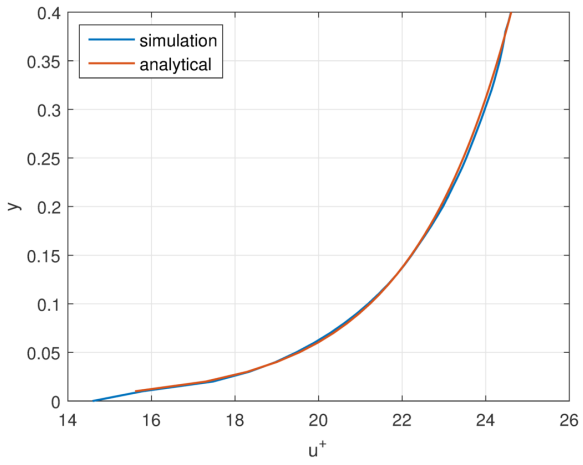


Figure 6.16: Vertical profile of averaged stream-wise velocity.

Bibliography

[1] L. Ge and F. Sotiropoulos, "A numerical method for solving the 3D unsteady incompressible NavierStokes equations in curvilinear domains with complex immersed boundaries," *Journal of Computational Physics*, vol. 225, pp. 1782–1809, Aug. 2007.

[2] A. Calderer, S. Kang, and F. Sotiropoulos, "Level set immersed boundary method for coupled simulation of air/water interaction with complex floating structures," *Journal of Computational Physics*, vol. 277, pp. 201–227, 2014.

[3] A. Calderer, X. Guo, L. Shen, and F. Sotiropoulos, "Coupled fluid-structure interaction simulation of floating offshore wind turbines and waves: a large eddy simulation approach," in *Journal of Physics: Conference Series*, vol. 524, p. 012091, IOP Publishing, 2014.

[4] T. B. Le and F. Sotiropoulos, "Fluid–structure interaction of an aortic heart valve prosthesis driven by an animated anatomic left ventricle," *Journal of computational physics*, vol. 244, pp. 41–62, 2013.

[5] T. B. Le, *A computational framework for simulating cardiovascular flows in patient-specific anatomies*. PhD thesis, UNIVERSITY OF MINNESOTA, 2011.

[6] I. Borazjani and F. Sotiropoulos, "The effect of implantation orientation of a bileaflet mechanical heart valve on kinematics and hemodynamics in an anatomic aorta," *Journal of biomechanical engineering*, vol. 132, no. 11, p. 111005, 2010.

[7] A. Khosronejad, S. Kang, I. Borazjani, and F. Sotiropoulos, "Curvilinear immersed boundary method for simulating coupled flow and bed morphodynamic interactions due to sediment transport phenomena," *Advances in water resources*, vol. 34, no. 7, pp. 829–843, 2011.

[8] A. Khosronejad, S. Kang, and F. Sotiropoulos, "Experimental and computational investigation of local scour around bridge piers," *Advances in Water Resources*, vol. 37, pp. 73–85, 2012.

[9] A. Khosronejad, C. Hill, S. Kang, and F. Sotiropoulos, "Computational and experimental investigation of scour past laboratory models of stream restoration rock structures," *Advances in Water Resources*, vol. 54, pp. 191–207, 2013.

[10] X. Yang, S. Kang, and F. Sotiropoulos, "Computational study and modeling of turbine spacing effects in infinite aligned wind farms," *Physics of Fluids (1994-present)*, vol. 24, no. 11, p. 115107, 2012.

[11] X. Yang, J. Annoni, P. Seiler, and F. Sotiropoulos, "Modeling the effect of control on the wake of a utility-scale turbine via large-eddy simulation," in *Journal of Physics: Conference Series*, vol. 524, p. 012180, IOP Publishing, 2014.

[12] S. J. Osher and R. P. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2003 ed., Oct. 2002.

[13] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations," *J. Comput. Phys.*, vol. 79, pp. 12–49, Nov. 1988.

[14] M. Sussman and E. Fatemi, "An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow," *SIAM J. Sci. Comput.*, vol. 20, pp. 1165–1191, Feb. 1999.

[15] S. Kang and F. Sotiropoulos, "Numerical modeling of 3D turbulent free surface flow in natural waterways," *Advances in Water Resources*, vol. 40, pp. 23–36, May 2012.

[16] I. Borazjani, L. Ge, and F. Sotiropoulos, "Curvilinear immersed boundary method for simulating fluid structure interaction with complex 3D rigid bodies," *Journal of Computational Physics*, vol. 227, pp. 7587–7620, Aug. 2008.

[17] I. Borazjani and F. Sotiropoulos, "Vortex induced vibrations of two cylinders in tandem arrangement in the proximity-wake interference region," *Journal of fluid mechanics*, vol. 621, pp. 321–364, 2009. PMID: 19693281.

[18] W. Cabot and P. Moin, "Approximate wall boundary conditions in the large-eddy simulation of high reynolds number flow," *Flow, Turbulence and Combustion*, vol. 63, no. 1-4, pp. 269–291, 2000.

[19] M. Wang and P. Moin, "Dynamic wall modeling for large-eddy simulation of complex turbulent flows," *Physics of Fluids*, vol. 14, no. 7, p. 2043, 2002.

[20] J.-I. Choi, R. C. Oberoi, J. R. Edwards, and J. A. Rosati, "An immersed boundary method for complex incompressible flows," *Journal of Computational Physics*, vol. 224, pp. 757–784, June 2007.

[21] B. M. Irons and R. C. Tuck, "A version of the aitken accelerator for computer iteration," *International Journal for Numerical Methods in Engineering*, vol. 1, no. 3, pp. 275–277, 1969.

[22] X. Yang, F. Sotiropoulos, R. J. Conzemius, J. N. Wachtler, and M. B. Strong, "Large-eddy simulation of turbulent flow past wind turbines/farms: the Virtual Wind Simulator (VWiS): LES of turbulent flow past wind turbines/farms: VWiS," *Wind Energy*, pp. n/a–n/a, Aug. 2014.

[23] S. Kang, A. Lightbody, C. Hill, and F. Sotiropoulos, "High-resolution numerical simulation of turbulence in natural waterways," *Advances in Water Resources*, vol. 34, pp. 98–113, Jan. 2011.

[24] J. Smagorinsky, "General circulation experiments with the primitive equations: I. the basic experiment*," *Monthly weather review*, vol. 91, no. 3, pp. 99–164, 1963.

[25] M. Germano, U. Piomelli, P. Moin, and W. H. Cabot, "A dynamic subgrid-scale eddy viscosity model," *Physics of Fluids A: Fluid Dynamics (1989-1993)*, vol. 3, no. 7, pp. 1760–1765, 1991.

[26] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, and H. Zhang, "Petsc web page," 2014. http://www.mcs.anl.gov/petsc.

[27] H. T. Ahn and Y. Kallinderis, "Strongly coupled flow/structure interactions with a geometrically conservative ALE scheme on general hybrid meshes," *Journal of Computational Physics*, vol. 219, pp. 671–696, Dec. 2006.

[28] J. Yang and F. Stern, "Sharp interface immersed-boundary/level-set method for wave-body interactions," *Journal of Computational Physics*, vol. 228, pp. 6590–6616, Sept. 2009.

[29] S. Ito, *Study of the transient heave oscillation of a floating cylinder*. PhD thesis, MIT, 1971.

[30] X. Guo and L. Shen, "On the generation and maintenance of waves and turbulence in simulations of free-surface turbulence," *Journal of Computational Physics*, vol. 228, pp. 7313–7332, Oct. 2009.

[31] L. P. Chamorro and F. Porté-Agel, "A wind-tunnel investigation of wind-turbine wakes: boundary-layer turbulence effects," *Boundary-layer meteorology*, vol. 132, no. 1, pp. 129–149, 2009.

[32] S. B. Pope, *Turbulent flows*. Cambridge university press, 2000.