National Technical University of Athens
School of Naval Architect and Marine Engineer
Division of Marine Hydrodynamics

Diploma Thesis

The Overset Grid Assembly (Chimera) of Unstructured Partitioned Grids coupled with the Artificial Compressibility Roe Solver and Applications in Marine Hydrodynamics

Academic Year 2021-2022

Student: Spyridon Zafeiris

Supervisor: Georgios Papadakis, Assistant Professor

Student Registration Number: 08117003

July 2022, Athens

Abstract

In the following thesis, the problem of the numerical flow simulation around moving bodies is tackled. The flow solution concerns incompressible one or two phase fluids without surface tension in the interface. These types of flows are largely applied to marine and oceanographic applications and as a consequence, there has been a need for a robust handling of the grid movement due to wall translation and rotation. Moreover, due to the storage intensity of large sized grids and the massive computational load needed from the computer to perform the required floating point operations, the following work is implemented in parallel computer-architectures. The implementation is conducted via the unstructured parallel CFD code provided by the laboratory coupled with an overset grid assembly library.

Acknowledgements

This work signals the end of my undergraduate studies and is a result of long-lasting efforts of many people who contributed and I would like to thank.

First of all, I would like to thank my supervisor Prof. G. Papadakis for his precious help in many matters throughout the development of my thesis. Aside from instructions and directives in all aspects of the field with which I learned and practised, he was very helpful considering the brainstorming and the solution-seeking process from start to finish. This work would not be possible without his help and the help of his PhD student D. Ntouras who solved many technical problems that occurred and explained various details on the solver and CFD generally.

Many thanks also goes to my friends, Kyriakos, Andreas, Aggelos, Giorgos, Vasilis and Thanasis who gladly supported the work I prepared and with whom I exchanged many ideas in various research topics.

Special credits goes to Penny, because her support gives me strength to achieve my aims.

Lastly, I would like to thank my family for all the support and trust in every decision I made, giving me enough courage to continue and achieve something that for me, seems a great accomplishment. Their appreciation in what I do is one of the most valuable gifts.

Contents

List of Figures

List of Tables

Nomenclature

α	volume fraction
δ_{ij}	Kronecker delta
Γ_e	Jacobian Matrix
$\mathbf{g}$	gravity acceleration vector
$\mathbf{u}$	velocity vector
P_i	process i (computing unit)
μ	dynamic viscosity
$\overrightarrow{F}_{\cdot}$	vector form of flux function
$\overrightarrow{Q}$	vector form of primitive variables
$\overrightarrow{S}_{\cdot}$	vector form of source term
$\overrightarrow{U}$	vector form of state variables
$\overleftrightarrow{\sigma}$	Cauchy stress tensor
$\overleftrightarrow{\tau}$	strain tensor
ρ	density of fluid
$AABB_i$	Axis-Aligned Bounding Box belonging to process P_i
D_k	control volume k
e	total energy of fluid
OBB_i	Oriented Bounding Box belonging to process P_i
p	pressure

Chapter 1

Introduction

The research field of fluid dynamics has been in the center of many scientific and engineering domains in the sense that there exist a plethora of physical phenomena involving fluid interactions.

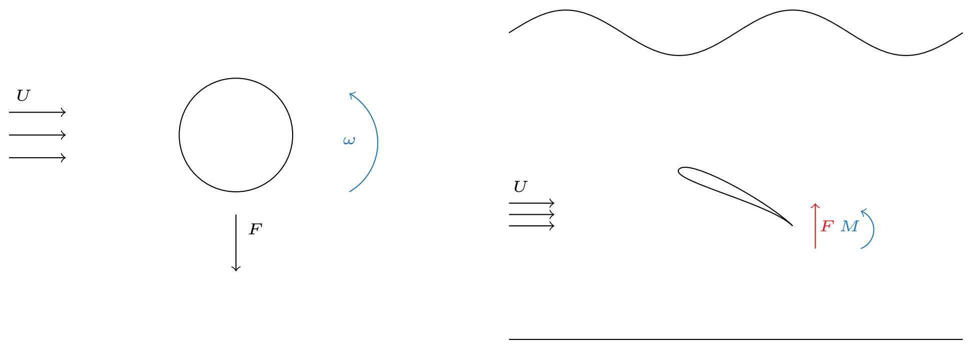
Evidently, the CFD industry has grown over the years with a variety of research successes pertaining to answers to numerous engineering instances in the aviation, transport, manufacturing, and health industries, as well as in the climate domain and many more.

Although the following study devotes a significant portion to the idea behind its methodology and a verification of its applicability, it focuses mostly on flows observed in marine and oceanic engineering applications. Of course, further integration with other engineering and scientific fields is obvious.

1.1 Motivation & Problem

The performance of the ship's conventional propulsion system (namely, the propeller) and the kinematics of a floating body are two of the most notable maritime applications that are the focus of the following thesis.

Today, every area of heavy industry is being impacted by the issue of climate change. Efficiency is on the scope as a benchmark for improvement. The marine industry, where the scalability of the transporting means infer vast amounts of fossil fuel produced energy and the footprint is similar with other heavy industrial sections, is not an exception to the rule when it comes to system efficiency improvement. Therefore, in-depth research and modelling are required for the dynamical system of the self-propelled ship, the oscillations of marine floating structures, and the energy extraction devices.



Computational fluid dynamics serves as a helpful instrument with numerous extending capabilities in this process.

More specific applications that are of crucial importance, involve solid moving boundaries such as the propeller or energy-saving devices like the flapping foil [1], the flettner rotor [2] and many others that are subject to a periodic motion that either drives a generator, or produces propulsive force, reducing thus the demand.

The concept of the overset grid approach, which is revisited in this thesis, serves the role of bridging the stationary observed flow with the body-induced interactions on a complex geometric domain.

1.2 Solving with CFD

Regarding fluid dynamics generally, conservation laws that describe the kinematics and dynamics of a continua with a fluid behaviour are still to be analytically solved in the most general case. As a consequence, there have been significant advancements in the study of phenomena with boundaries, turbulent flows, and multi-phase interactions with the aid of computing power and the capability to approximate numerically the fully nonlinear conservation laws.

The process of simulating numerically a certain CFD case, in a nutshell, involves creating a discretised domain of solution (mesh creation) with a proper geometric approximation of the boundary. The underlying equations are then also discretised and solved with machine precision on a computer.

Unavoidably, when a problem becomes more complicated, the discretization procedure becomes a more difficult task, creating new problems.

For the special case of flows in the vicinity of moving solid boundaries, a temporally static mesh, consisting of non-overlapping elements, is unable to represent the time-dependent solution domain $\mathcal{D}(t)$ and as a result, a mesh motion is applied mainly near the boundary.

This mesh motion method, which is frequently referred to as "mesh morphing", can be a useful strategy with little overall computing expense that monitors the time-varying solution domain $\mathcal{D}(t)$.

However, mesh morphing is a limited alternative for flows with several solid surfaces moving in a particular motion since it causes significant deformations in each geometric cell.

The first alternative to address the dynamic change in the geometric domain entails repeatedly generating meshes for every fraction of translation and rotation, with the drawbacks of excessive computing costs and little to zero control over the grid resolution.

In the second alternative, multiple overlapping meshes are used, one for each body, each with its own motion (body driven), and possibly even with an overlapping volume. This approach, called "*chimera*" after the mythological creature, was first put forth by *Joseph L. Steger et. al.* [3], [4] and is now the standard for complex simulations combining body motion and potentially flexible borders.

1.2.1 Unstructured Grids

As more intricate geometric applications are modelled using CFD, the necessity for geometric versatility over the cell density across the entire domain has drawn attention to the development of unstructured meshes. The unstructured cells can differ from the ones contained in structured meshes in that they can have any number of bounding vertices, and the solver can then obtain the necessary data from the cell connectivity in a global index style.

In contrast to structured meshes, unstructured cells may have many different orientations with respect to the fluid momentum vector and their shape and size has to be closely monitored be the researcher. More precisely, highly skewed cells with flattened shapes are not very preferable. A poorly designed unstructured mesh can lead to extra numerical diffusion, a fundamental issue observed in numerical methods.

The question is “how does the cell skewness affects the numerical performance?”. In the majority of the semi-discrete expressions for a weak formulation, special attention is given in terms arising from the application of the divergence theorem, i.e. the surface integral terms. The orientation of the face with respect to the left and right cell centre will eventually determine the accuracy of the flux approximation or the face value approximation. In the case of non-linear flux function terms, the system’s non-linearity is solely depicted on this flux function, whereas, in the case of the face value, its use is seen in the calculation of a cell-centered gradient (diffusion terms calculation).

Keeping in mind these features of unstructured discretisation, extensive mesh morphing will also come with an extra cost of accuracy performance.

Chimera methods, will try to overcome this problem by using well-designed (taylor-made for each body) meshes along with sophisticated interpolation techniques.

However, a fundamental issue with multiple, overlapping unstructured grids is grid connectivity, namely the sharing and ownership of field information.

1.2.2 Parallel Implementation

For such numerical algorithms, the computational and storage requirements of a fluid dynamics simulation present an extra challenge. The MPI protocol must be used to divide the computing demand among a number of storage independent computing units (namely, CPUs). The major objective of the scalability criterion is to lessen the communication bottleneck caused by data transfer among computer units, which may be accomplished with the use of the right library and a practical algorithm. As a result, parallel integration increases processing performance and makes it possible to run sophisticated 3D unstable simulations.

1.3 Moving Meshes

1.3.1 Structured Meshes

The most fundamental component for describing a body movement through a fluid, is the expression of the conservation laws in the moving reference frame, usually referred as *Arbitrary Eulerian-Lagrangian (ALE)*. This method can be used to solve the problem of a single body moving through the three-dimensional space.

When a body with a different motion is added to the general case, the moving reference frame is no longer unique, making it impossible for the *(ALE)* expression to solve the approximation problem.

The use of multiple composite grids had another big advantage, as specified in [5] in the time when mesh creation was more of a manual procedure.

Prior to the development of algorithms for unstructured mesh creation, mesh points had to be produced on structured grids in compliance with a given geometry (for example, an airfoil) by means of particular geometric transformations, typically in the complex domain, such as the *Schwartz-Cristoffel* transformation,[6], or other algorithms, with the restriction that the geometry would have to be developable (and therefore the mapping from a parametric space would be conformal). Additionally, these algorithms would typically limit the researcher to two-dimensional simulations.

When dealing with many borders and more complex (non-developable) geometries, these techniques were challenging to implement. For that reason, the overset grid method came as a solution, indicating that every boundary-forming structure (or substructure) could be discretised in its vicinity and then superimposed over a cartesian grid where field communication was applied.

As an illustration, we can consider discretising the volume surrounding the propeller and its shaft (one body). The researcher was able to construct a volume mesh around each blade using structured composite meshes, a mesh around the cylindrical shape of the shaft, and ultimately a cartesian mesh covering the entire volume of solution.

1.3.2 Composite Grids

Naturally, there is a cost associated with the solution of the aforementioned complex problem. Field variables are now (partially) interpolated from one grid to another, which is one more source of numerical diffusion in addition to the approximation error of the discretisation of the non-linear conservation laws (e.g., finite differences). These errors are distinct from approximation ones (where non-linear terms are projected onto a finite dimensional functional space), and for the interpolation problem, point values are typically the only available information.

In two-dimensional problems (probably unsteady), the effects of interpolation errors are observed through the disability of the system to conserve its energy (internal and mechanical). Or even if the spatial and temporal discretisation have non-conservative properties, composite grids tend to weaken the accuracy and the convergence rate.

Conservativity and accuracy preservability on composite meshes has been a topic addressed by [5], [7], [8] and [9]. The majority of the research described above either concentrate on the performance of grid data interpolation or the coupling with the solver and the corrector steps for fluxes. For example in [7], the mass loss is balanced by correcting the mass fluxes on an iterative procedure. Other flux correction techniques were implemented in [10] aside from interpolation schemes. The coupling with the solver can be implemented in many ways and is dependent on the current solution methodology (e.g., the pressure-coupling procedure). Of course, there is an additional source of computing weight that belongs in this iterative convergence process along with the processing cost of interpolating and solving the partial differential equations.

Leaving solver coupling solutions aside, there are a variety of interpolation techniques that, due to their high degree of accuracy and computational complexity, tend to be computationally intensive. The decision to use high order interpolation methods is made in an effort to avoid numerical diffusion by offering an approximation with a higher order than the solver's.

1.3.3 Interpolation Techniques

The problem of multivariate interpolation as mentioned before, relies on a given topology of the interpolating nodes, the density of the cloud (of nodes) and of course the order of solution in the inter-node regions. If the structure of nodes is given and their expressions involve the counting of indices i, j, k, then a parametric volume ξ, η, ζ can be constructed in which the abscissa are mapped from physical space while respecting the appropriate parameterisations. In contrast, in the case of clustered node data (unstructured manner), there are two possible solutions.

Asimuth free approximation The first solution which is widely applicable and probably takes the less computational effort is the definition of the interpolating weights in a radial-dependent way. The accuracy of this type of interpolation is unaffected by the direction of contributions, and typically a shape parameter for the basis functions is defined, implying a dependence on the node density of the nearby region. For optimal results, this shape parameter needs to be properly defined. Weight functions without shape parameters do exist, and they do solve the problem of defining these scaling parameters, but they come with the trade-off of anticipating a suboptimal result (or, more precisely, whether it is of acceptable accuracy or not), and the data interpolation is heavily reliant on the dispersed node multitude throughout the desired domain.

Many researchers who are mentioned above, have studied these techniques (for example the Inverse Distance Weighting or the Radial Basis Functions).

Lagrangian to Eulerian to Lagrangian Methods The second solution is derived from the existence of non-classical solvers, namely particle solvers (vortex particles) and hybrid solvers (Particle and Mesh). As far as hybrid solvers are concerned, the solution near the boundaries is obtained by the use of a classic Eulerian solver over a body fitted discretised mesh, whereas regions without walls are formulated using particles that carry the field variables while their movement and interactions are determined by the conservation laws, expressed in the Lagrangian frame of reference. Such methods have been tested and thoroughly explored in [11], [12] and [13]. Some issues with these formulations relate to the field data transfer from the grid to the particles or the particle relocation, which is actually done to reduce the amount of trajectory dislocations (or particle position conflicts).

Generally, particles are often arranged in a regular cartesian grid, and when they have to be relocated (or redistributed), their placements have close to normal spacing between neighbours, indicating that they are not dispersed randomly. This property permits the use of particular kernels derived from the theory of splines, specifically the cardinal splines developed by *I.J. Schoenberg* [14], [15] that are of high order and preserve up to an order of moments over their domain of support, like the kernel shown below (M_6^* function).

$$M_6^*(x) = \begin{cases} 0 & |x| > 3 \\ -\frac{1}{24}(|x|-2)(|x|-3)^3(5|x|-8), & 2 \le |x| \le 3 \\ \frac{1}{24}\left(25|x|^3 - 114|x|^2 + 153|x| - 48\right)(|x|-1)(|x|-2), & 1 \le |x| \le 2 \\ -\frac{1}{12}\left(25|x|^4 - 38|x|^3 - 3|x|^2 + 12|x| + 12\right)(|x|-1), & |x| \le 1 \end{cases} \tag{1.1}$$

When using such functions on a tensor product for the representation in many dimensions, the accuracy is sufficient enough to preserve the conservation properties that mesh-less methods offer. Another characteristic of such kernels is that they contain, by definition, the interpolation restrictions (identical values over interpolating nodes), which makes the calculation on the new structured grid simple and free of a linear system solution.

In the case of overset methods, these values can be re-projected on the overset grid immediately following their projection on the structured grid (which of course can be unstructured and with no regularity).

This concept, apart from being a direct implementation of the hybrid paradigm, is computationally intensive and usually involves large inter-processor data communication.

Robustness is a major goal in this thesis, and this type of interpolation is not addressed.

Chapter 2

Methodology

2.1 Navier-Stokes Equations

Firstly, the general boundary value-initial value problem for (in general) compressible fluid motion by means of conservation laws is introduced.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{2.1}$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \oplus \mathbf{u}) = \rho g + \nabla \cdot \overset{\leftrightarrow}{\sigma} + F \tag{2.2}$$

$$\frac{\partial \rho \left(\frac{1}{2}u^2 + e\right)}{\partial t} + \nabla \cdot \left(\rho \mathbf{u} \left(\frac{1}{2}u^2 + e\right)\right) = \rho \mathbf{u} \mathbf{g} + \nabla \cdot \left(\overset{\leftrightarrow}{\sigma} \cdot \mathbf{u}\right) - \nabla \cdot \mathbf{q} + w\rho \tag{2.3}$$

$$f(p, V, T) = 0 \tag{2.4}$$

$$\mathbf{u}\left(\mathbf{x}\right) = 0, \quad \mathbf{x} \in \partial \mathcal{D} \tag{2.5}$$

In the equations 2.1, 2.2, 2.3 and 2.4, the state variables $\{\rho, \mathbf{u}, e\}$ stand in for density, velocity components, and total internal energy, respectively. When squared, the bold vector expression in the velocity components is absent. For all the equations, $\mathbf{x}$ belongs to the domain of interest $\mathcal{D}(t)$.

The Cauchy stress tensor is expressed with $\overset{\leftrightarrow}{\sigma}$ while the heat flux is represented with $\mathbf{q}$.

Initial values for the variables and boundary conditions—typically of the *Dirichlet* type (no slip condition) or *Neumann* type (no through condition)—are introduced along with the governing equations.

2.1.1 Incompressibility

In the case of incompressible fluids, the equation 2.6 depicts the so-called divergence-free condition (continuity equation), acting as a constraint on the velocity field. Equation 2.7 is simplified with respect to the constant space and time derivatives of the density, and equation 2.3 is trivial and may be omitted as the energy terms are calculated through the equation of state 2.4.

While further simplifications are done, the incompressible continuity and Navier-Stokes Equations emerge.

$$\nabla \cdot \mathbf{u} = 0 \tag{2.6}$$

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\,\mathbf{u} = \rho g + \frac{1}{\rho}\nabla \cdot \overset{\leftrightarrow}{\sigma} + \frac{1}{\rho}F \tag{2.7}$$

2.1.2 Material Equations

In addition, as the current work is devoted to the overset paradigm, further simplifications are done considering viscous forces.

Since the fluid is thought to be of the *Newtonian* type, the viscous stresses adhere to Hooke's linear material law, and the shear component of the tensor is a scaled function of the velocity field's strain rate.

$$\sigma_{ij} = -p\delta_{ij} + \tau_{ij} \tag{2.8}$$

$$\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \tag{2.9}$$

Equations 2.6 become:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\,\mathbf{u} = -\frac{1}{\rho}\nabla p + \rho g + \nu \Delta \mathbf{u} + \frac{1}{\rho}F \tag{2.10}$$

2.1.3 Free Surface Flows

Taking a step further, multi-phase flows are examined, especially flows referred as continuous-continuous Eulerian.

Regarding free surface flows (which fall under the aforementioned category), numerous techniques have been developed throughout the years, primarily falling into two main types.

- surface tracking methods
- surface capturing methods

The first category, which was developed by [16] and used in previous years, entails representing the free surface as a moving boundary with appropriate boundary conditions and a solution focusing only on the liquid phase. Grid deformation occurs when this strategy is used.

The family of surface capturing methods involves different techniques such as the Volume of Fluid (VOF) method [17], the level-set method [18] and the Marker-and-Cell method [19] which, in contrast to surface tracking methods, treat the material surface as a density discontinuity and both phases as solved.

In the case of (VOF) the *volume fraction* is presented.

$$\alpha_l = \frac{\rho_m - \rho_a}{\rho_w - \rho_a} \tag{2.11}$$

Subscripts a & w refer to air and water respectively.

The density and dynamic viscosity fields are then expressed as a convex combination of the *volume fraction.*

$$\rho_m = \alpha_l \rho_w + (1 - \alpha_l)\, \rho_a \tag{2.12}$$

$$\mu_m = \alpha_l \mu_w + (1 - \alpha_l)\, \mu_a \tag{2.13}$$

It is pointed out that surface tension is omitted, and thus the pressure field is continuous across the surface. Also, phase changes and therefore mass transfer across the field are out of the current scope.

As mentioned before, the free surface is considered to be a material surface and the volume fraction is subject to pure advection:

$$\frac{\partial \alpha_l}{\partial t} + (\mathbf{u} \cdot \nabla)\, \alpha_l = 0 \tag{2.14}$$

The last equation 2.14 closes the system of equations for the incompressible two phase flow problem.

2.1.4 Governing Equations

Lastly, we present the governing incompressible equations for the two phase field collectively, while dropping the ∇ operator and reducing to the Einstein notation.

$$\partial_j u_j = 0 \tag{2.15}$$

$$\partial_t u_i + \left(u_j \partial_j\right) u_i = -\frac{1}{\rho}\partial_i p + \frac{1}{\rho} g_i + \nu \Delta u_i + F_i \tag{2.16}$$

$$\partial_t \alpha_l + \left(u_j \partial_j\right) \alpha_l = 0 \tag{2.17}$$

2.1.5 Conservative Form

In order to derive the discrete system, we return to the conservative form along with the transport equation for the volume fraction.

$$\partial_j \cdot u_j = 0 \tag{2.18}$$

$$\frac{\partial \rho_m u_i}{\partial t} + \partial_j \cdot \left(\rho_m u_i u_j\right) + \partial_j p = \partial_j \cdot \sigma_{ij} + F_{B_i}, \qquad i = 1, 2, 3 \tag{2.19}$$

$$\frac{\partial \alpha_l}{\partial t} + \partial_j \cdot \left(u_j \alpha_l\right) = 0 \tag{2.20}$$

The above system is suitable for employing the Green-Gauss theorem. Integrating over a control volume, yields:

$$\int_{D_k} \partial_j \cdot u_j \,\mathrm{d}D_k = 0 \tag{2.21}$$

$$\frac{\partial}{\partial t}\int_{D_k} \rho_m u_i \,\mathrm{d}D_k + \int_{D_k} \partial_j \cdot \left(\rho_m u_i u_j\right) \,\mathrm{d}D_k + \int_{D_k} \partial_j p \,\mathrm{d}D_k = \int_{D_k} \partial_j \cdot \sigma_{ij} \,\mathrm{d}D_k + \int_{D_k} F_{B_i} \,\mathrm{d}D_i \tag{2.22}$$

$$\frac{\partial}{\partial t}\int_{D_k} \alpha_l \,\mathrm{d}D_k + \int_{D_k} \partial_j \cdot \left(u_j \alpha_l\right) \,\mathrm{d}D_k = 0 \tag{2.23}$$

Introducing the outward normal vector **n** and applying the Green-Gauss theorem, the following expression arises.

$$\int_{\partial D_k} u_j \cdot n_j \,\mathrm{d}S_k = 0 \tag{2.24}$$

$$\frac{\partial}{\partial t}\int_{D_k} \rho_m u_i \,\mathrm{d}D_k + \int_{\partial D_k} \rho_m u_i \left(u_j \cdot n_j\right) \,\mathrm{d}S_k + \int_{\partial D_k} n_j p \,\mathrm{d}S_k = \int_{\partial D_k} \sigma_{ij}^T \cdot n_j \,\mathrm{d}S_k + \int_{D_k} F_{B_i} \,\mathrm{d}D_i \tag{2.25}$$

$$\frac{\partial}{\partial t}\int_{D_k} \alpha_l \,\mathrm{d}D_k + \int_{\partial D_k} \alpha_l \left(u_j \cdot n_j\right) \,\mathrm{d}S_k = 0 \tag{2.26}$$

Presenting the above in a vector form and performing the variable transformation to the primitive variables, gives:

$$\Gamma_e \frac{\partial}{\partial t}\int_{D_k} \vec{Q} \,\mathrm{d}D_k + \int_{\partial D_k} \left(\vec{F_c} - \vec{F_v}\right) \mathrm{d}S_k = \int_{D_k} \vec{S_q} \,\mathrm{d}D_k \tag{2.27}$$

$$\vec{U} = \left[\ 0 \quad \rho\vec{u} \quad \alpha_l\ \right]^T, \qquad \vec{Q} = \left[\ p \quad \vec{u} \quad \alpha_l\ \right]^T \tag{2.28}$$

$$\frac{\partial \vec{U}}{\partial t} = \frac{\partial U}{\partial Q}\frac{\partial \vec{Q}}{\partial t} = \Gamma_e \frac{\partial \vec{Q}}{\partial t} \tag{2.29}$$

$$\Gamma_e = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \rho I_{3\times 3} & \vec{u}\delta\rho \\ 0 & 0 & 1 \end{bmatrix} \qquad \vec{F}_c = \begin{bmatrix} V \\ \rho_m u\delta V + pn_x \\ \rho_m v\delta V + pn_y \\ \rho_m w\delta V + pn_z \\ \alpha_l \delta V \end{bmatrix} \tag{2.30}$$

$$\vec{F}_v = \begin{bmatrix} 0 \\ \tau_{xx}n_x + \tau_{xy}n_y + \tau_{xz}n_z \\ \tau_{yx}n_x + \tau_{yy}n_y + \tau_{yz}n_z \\ \tau_{zx}n_x + \tau_{zy}n_y + \tau_{zz}n_z \\ 0 \end{bmatrix}$$

In the above expressions which follow the nomenclature of [20], the following expressions appear:

$$V_n = \vec{u} \cdot \vec{n}$$
$$V_g = \vec{u}_{vol} \cdot \vec{n}$$
$$\delta V = V_n - V_g$$

2.2 Finite Volume Method

Some of the most commonly used numerical methods for hyperbolic systems that have flourished for many decades are the finite difference method and the finite element methods. The most important difference between them is that the finite element methods rely on taking the weak formulation of the equation and integrating over a control volume, performing that way the projection of the solution to a finite space. Finite differences, on the other hand, rely on approximating the differential operators acting on the solving variables. One can generalize and assume that the finite difference method (FDM) is a form of a Petrov-Galerkin approximation where the test function is the *Dirac-delta* function and the spacial (mainly derivative) operators, and therefore the function itself is projected onto a polynomial space through interpolation on a given stencil (arbitrary order of spacial accuracy).

Aside from these usual methods, meshless methods have found ground of applicability in advection-diffusion problems, etc., but with certain difficulties when solid bodies are present.

A special case for the finite element methods is the use of piece-wise constant test functions and solution approximations. Additionally, the conservative form of the equations is crucial for the use of the Green-Gauss theorem, such that the discretization problem is reduced mainly to computing numerical fluxes across the cell interfaces.

2.2.1 Velocity-Pressure Coupling

In contrast with the compressible continuity & Navier Stokes equations 2.1,2.2, 2.3 where the system of equations is well-coupled and the conservation laws are able to converge equivalently for all the state variables via an implicit or explicit temporal scheme, the incompressible manner of the equations cannot ensure that the pressure field, although approximated, is consistent with the velocity field, not before the divergence-free constraint is applied.

For that matter, there have been proposed a number of ways to numerically implement pressure-linked equations with the most known being the SIMPLE algorithm developed by S. V. Patankar & D. B. Spalding [21] and the general method of operator splitting (PISO) proposed by A. J. Chorin [22].

In the current work, the solver uses the artificial compressibility method (AC) proposed by A. J. Chorin [23] which yields a hyperbolic nature to the existing system of equations introducing a pseudo-time derivative for the proper coupling within each of the equations. The time-fictitious problem is then approximated implicitly or explicitly until convergence is achieved.

The advantage of the current hyperbolic reformulation is that both the incompressible NS equations with the continuity equation and the advection of the volume fraction are approximated and updated simultaneously, giving the possibility to treat the discontinuous problem as an approximation of the Riemann problem for a system of equations.

Artificial Compressibility

With the introduction of the pseudo-time term the system of equations 2.27 has now the following expression:

$$\Gamma \frac{\partial}{\partial \tau} \int_{D_k} \vec{Q} \, \mathrm{d}D + \Gamma_e \frac{\partial}{\partial t} \int_{D_k} \vec{Q} \, \mathrm{d}D_k + \int_{\partial D_k} \left(\vec{F_c} - \vec{F_v} \right) \mathrm{d}S_k = \int_{D_k} \vec{S_q} \, \mathrm{d}D_k \tag{2.31}$$

$$\Gamma = \begin{bmatrix} \frac{1}{\beta \rho_m} & 0 & 0 \\ 0 & \rho_m I_{3\times 3} & \vec{u}\delta\rho \\ \frac{\alpha_l}{\beta \rho_m} & 0 & 1 \end{bmatrix} \tag{2.32}$$

2.2.2 Spacial Discretisation

Volume Integral Approximation

The current solver (MaPFlow) uses cell-centered values and therefore the volume integral is approximated as the average value times the volume. This integral approximation is referred in the bibliography as the midpoint rule, which is 2^{nd} order accurate for any control volume.

$$\overline{\vec{Q}} = \frac{1}{D_i} \int_{D_i} \vec{Q}(\mathbf{x}, t) \; \mathrm{d}D \tag{2.33}$$

Face Reconstruction

Considering the inviscid fluxes, their values are computed as a function of the left and right cell values, as:

$$F(\vec{Q}) = f\left(\vec{Q}_L, \vec{Q}_R\right) \tag{2.34}$$

For the values of the left and right cells, a linear interpolation scheme is employed except for the case of the vicinity of the free surface where the pressure gradient is discontinuous and the volume fraction is of course highly discontinuous.

$$\mathbf{u}_L = \mathbf{u}_i - \nabla\mathbf{u}_i \cdot \mathbf{r}_i \tag{2.35}$$
$$\mathbf{u}_R = \mathbf{u}_j + \nabla\mathbf{u}_j \cdot \mathbf{r}_j \tag{2.36}$$

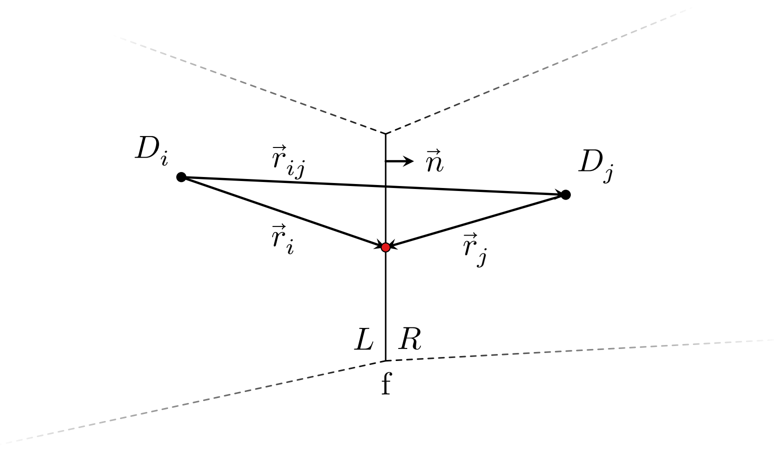


Figure 2.1: schematic representation of face reconstruction

For the case of the inviscid fluxes $\vec{F}_c$ the approximate Riemann solver of Roe [24] is employed.

$$\vec{F}_{c,f} = \frac{1}{2}\left(\vec{F}_c\left(\vec{Q}_R\right) + \vec{F}_c\left(\vec{Q}_L\right)\right) - \frac{1}{2}\Gamma\left|\underline{A_c}\right|_f \left(\vec{Q}_R - \vec{Q}_L\right) \tag{2.37}$$

Where $\underline{A_c}$ is the preconditioned Jacobian Matrix.

$$A_c = \frac{\partial\vec{F}_c}{\partial\vec{Q}} = \Gamma\Gamma^{-1}A_c = \Gamma\underline{A}_c \tag{2.38}$$

Where Γ is the precondition matrix given by Kunz [25].

For the case of viscous fluxes, a straightforward cell-averaging value is obtained.

2.2.3 Temporal Discretisation

The solver focuses mainly on implicit temporal schemes, both on pseudo-time and real time discretisation.

Firstly, the internal iteration is performed until the total residual (unsteady & spacial terms) converges to zero.

$$\Gamma \frac{\partial \left(\overrightarrow{Q}^{*} D_i \right)}{\partial \tau} + \overrightarrow{R}^{*} = 0 \tag{2.39}$$

$$\overrightarrow{R}^{*} = \overrightarrow{R}_{D_i} \left(\overrightarrow{Q}^{*} \right) + \Gamma_e \frac{\partial \left(\overrightarrow{Q}^{*} D_i \right)}{\partial t} \tag{2.40}$$

Then the internal iterator updates all the values based on the iterative procedure.

The time discretisation is second order implicit in time and obeys the geometric conservation law (GCL) for the case of temporally changing control volumes.

2.3 The concept of Overset Grid Assembly

In *CFD* simulations with multiple overlapping grids that are subject to temporal motion, the proper field data communication is the main goal along with sufficient accuracy to achieve convergence as well as a computational efficiency criterion. Since the first appearance of the method, many software packages & libraries have been developed utilizing the algorithms for performing the grid assembly, such as SUGGAR++ [26], CHIMPS [27] and the most known *Chimera Grid Tools* attached to the solver OVERFLOW [28]. Many of these packages can perform the grid communication with a certain limitation, such as lack of handling unstructured grids or no parallel implementation.

2.3.1 TIOGA

In this thesis, the overset algorithms belong to TIOGA (Topology Independent Overset Grid Assembly) library developed by J. Sitaraman et. al. [29], which is a node-centred parallel MPI library with the capability of handling complex unstructured partitioned grids in a scalable way. The library, due to these advantages, performs in a general way within vast numerical simulation instances without being solver-specific. Lastly, the library is open source and the user is capable of performing the interpolation by himself.

In this thesis, the development lies more on matching the library with the solver and not the construction of the overset grid assembly algorithms. These algorithms were thoroughly researched and optimised by J. Sitaraman el. al. [29]. The most valuable part in this process is the ability to extract the grid assembly results (i.e. donors & receptors). That way, we can devote a large part on addressing the interpolation problem in various numerical cases of interest.

2.3.2 Definition

Consider a number of subdomains, namely $\mathcal{D}_i \subseteq \Omega,\ i < n$ that are subject to a geometric partitioning (such as triangulation) $\mathcal{T}_i$.

The goal of the overset grid assembly has a number of tasks.

Terminology

Characterization of points & cells:

- Query Point
- Donor Cell
- Candidate Donor Cell
- Hole Point
- Field Point

Tasks:

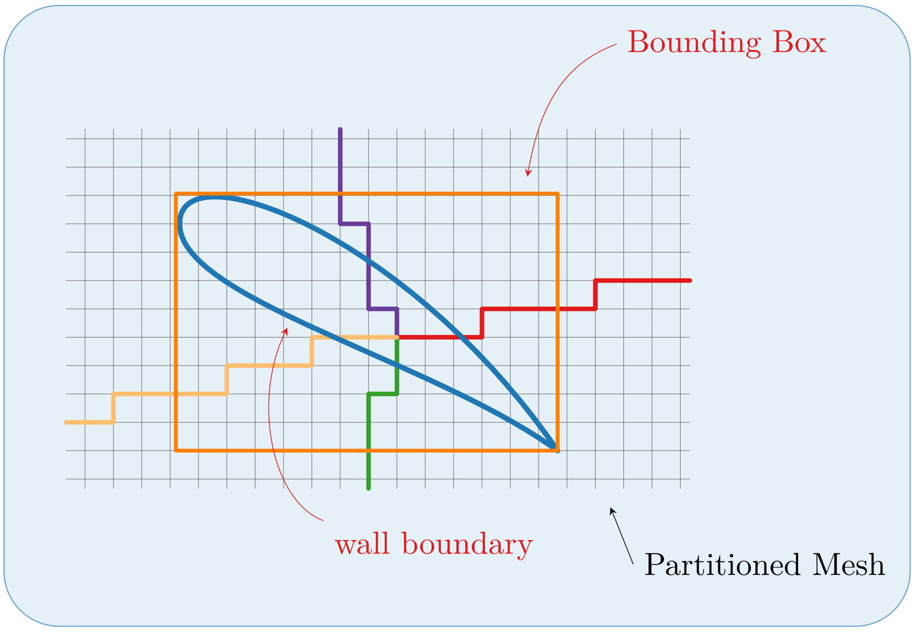
- Hole-Cutting
- Query identification
- Donor Search
- Point-type Assignment

Task Implementation

This indicates that the algorithm design is based on the principle of least data communication, and since the library is intended for generally unstructured grids, the main idea for all tasks is based on the creation and communication of an Auxiliary Grid (AG) created to carry out the search and the identification.

2.3.3 Hole-Cutting

The first algorithm that is implemented in the library (TIOGA) [29] identifies nodes that lie within the closed surface of the wall boundary.



Bounding Box Construction

Beginning with the identification of the hole cut, i.e. the smallest bounding box that surrounds all the wall nodes, all processes that obtain wall nodes shall build their own bounding box and communicate their coordinates:

$$BB_i := \left\{ \begin{matrix} A^i_{min} = (x_{min}, y_{min}, z_{min}) \\ A^i_{max} = (x_{max}, y_{max}, z_{max}) \end{matrix} \right\} \tag{2.41}$$

to all the other processes via broadcasting.

These (local) bounding boxes, along with those shared by communication, are treated as sets, and their union is considered the geometry that is to be bounded by a larger bounding box. This ensures that the final box will contain all the wall nodes and is, of course, also known to every process P_i.

Sub-block Division

Every process, starting with the global bounding box, divides BB_i iteratively into smaller sub-blocks until no sub-block contains both wall and outer nodes.

This way, the algorithm can mark the sub-blocks that lie entirely inside the wall boundary and the sub-blocks that have a possible overlap with wall faces.

Performing flood-fill

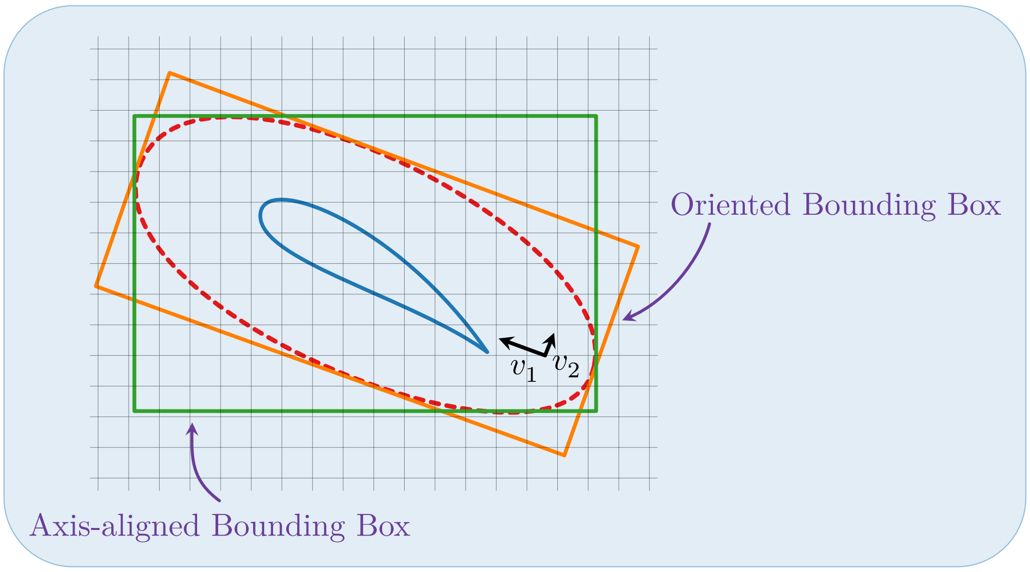
In order to mark which sub-blocks are totally inside the wall boundary and which do not, a layer is added externally in the bounding box, tagged as outer and the rest as inside. Moving from the outside region to the centre, the algorithm marks the sub-blocks that contain wall nodes, leaving only sub-blocks with zero overlap tagged as ”inside”.

2.3.4 Overset Node Identification

The next task is the query point identification. These overset points (named *query*) are associated with the mesh-block of the process P$_i$ and its potential geometric association with the mesh block of the process P$_j$, $i \neq j$ is examined. The first criterion is the possible overlap of the two mesh-blocks, and the second is that the point has to lie in the region of possible overlap.

In order to give a geometric intuition of the topology of the query points, all the points close to the outer (possibly overset) boundary have to be found and marked.

The algorithm starts with the creation of axis-aligned bounding boxes and oriented bounding boxes for all the mesh-blocks. Oriented bounding boxes have their three axes parallel with the three eigenvectors of the geometric covariance matrix for each mesh block. Auxiliary grids are then created with cell resolution defined by the number of vertices in the mesh-block. Necessary information for these boxes is broadcast through all processes P$_j$, $j = 1, N$. Potential overlap over different mesh-blocks is checked via point containment in an axis-aligned bounding box or an oriented bounding box.



$$\begin{aligned}
&Cov\,(X) = X^T X && \\
&det\,(X^T X - \lambda I_{3\times3}) = 0 && \Rightarrow \text{find } \lambda_i,\ i = 1,3 \\
&(X^T X - \lambda_i I_{3\times3})\,\mathbf{v}_i = 0 && \Rightarrow \text{find } \mathbf{v}_i,\ i = 1,3 \\
&\text{define OBB by its orientation } \{v_1, v_2, v_3\} &&
\end{aligned} \tag{2.42}$$

Overlap Criterion

Every process P$_i$ checks whether any of its sub-blocks that contain vertices overlap with sub-blocks of the auxiliary grid belonging to P$_j$. These sub-blocks are tagged and sent to process P$_j$. From the side of P$_i$, a check is performed whether these sub-blocks contains points from its mesh-block. Points that qualify for this procedure are stored as query points.

Also, during this overlap test, if a point of P$_j$ is inside both of $AABB_i$ and OBB_i then it is also marked as query.

Donor Search

Once the query point list is established, further information about cell connectivity and cell resolution is obtained.

A mapping function is created, connecting auxiliary grid sub-blocks with cells that overlap. This mapping is termed in [29] as the Exact Inverse Map. The case that the cell centre does not overlap is also examined, considering a possible intersection with the cell bounding box, establishing a cell point.

This mapping limits the donor search algorithm only on the listed (candidate donor) cells.

The donor cell is found following the line-walk algorithm (or stencil jumping algorithm) which briefly performs the following tasks:

- Defines the line (vector) pointing from starting point (cell point or cell center) to the query point
- checks for intersections over a cell face and then moves to the neighbouring cell
- checks whether the line intersects a mesh-block boundary
- chooses the closest cell for donor taking into account the above limitations

The structure of the algorithm identifying donor cells follows the line to face or line to line orientation logic to target the specific topology of the unstructured manner of the cells and its mesh-block boundaries.

Chapter 3

Development of the Chimera Branch

3.1 Compatibility

The most work-intensive part of introducing a library into an existing code, mainly with a large data structure, is obtaining the data in the correct way for use.

Depending on the structure of a certain *CFD* software and the numerical implementation, there are plenty of ways to introduce geometric data and field variables. Most finite volume methods, due to their sole degree of freedom per cell, tend to store field values at the cell centre and perform calculations on every face. The cell type identification (whether it holds a boundary condition, a symmetry condition, etc.) is done by firstly identifying the types of the faces (boundary face, mesh-block boundary, etc.). By appropriately tagging the mesh faces, the partitioned grid is adequately defined. Therefore, cell connectivity is achieved through face-left and right-oriented cells.

On the other hand, for finite element structured solvers such as Spectral Elements codes or Nodal Discontinuous Galerkin implementations, where every cell may obtain a larger number of degrees of freedom, it is basically essential that the structure is node-based (considering field values storing and topology identification).

It is obvious that developing a node-centered data structure results in a more generic and flexible approach, whereas the cell connectivity is based on node index mappings.

Another issue of compatibility that arises when trying to match a solver data structure with a library is the element sequence for all the grids per partition.

In unstructured meshes, it is usually valuable to reorder the cell sequence with the scope of resulting in a diagonally-dominant linear system matrix. For that matter, the grid-partitioning algorithms are used for proper grid division sets and suitable reordering. The grids are read by the solver and joined together as a result of these processes. Then the partitioning is performed and the grid number of the cell sequence is not an ascending function. Due to this, the grid number has to be kept in track during the start of the simulation.

The current library (TIOGA) [29] registers grids that are sorted by grid number and cell type (number of nodes defining the cell). For that reason, proper index mappings were produced to remedy this incompatibility issue.

It is noted at this point that it may be more efficient to apply the geometric grid partitioning algorithm separately to each grid for the hope of better load balance, especially in the case of the grid assembly.

3.1.1 Hash Tables

In order to create a suitable cell sequence for the library index mappings, appropriate algorithms have to be introduced for every job. These algorithms have to be implemented either by tagging elements or by sorting based on a mapping array.

During the simulation, it will be useful to obtain results and transfer cell indices from the library to the solver, making use of these mapping functions. For that reason, every index mapping will have its inverse introduced. However, some of the mappings are not bijective, meaning that they do not match one-to-one items. Hash tables are useful in the sense that the mapped information is basically a stored array of indices, and if there is a smaller number of mapped values, the inverse array will simply match the mapping size.

These mapping arrays are constructed at the start of the simulation, and they are used without being redefined. For that reason, computing performance optimisation during their evaluation is of minor importance.

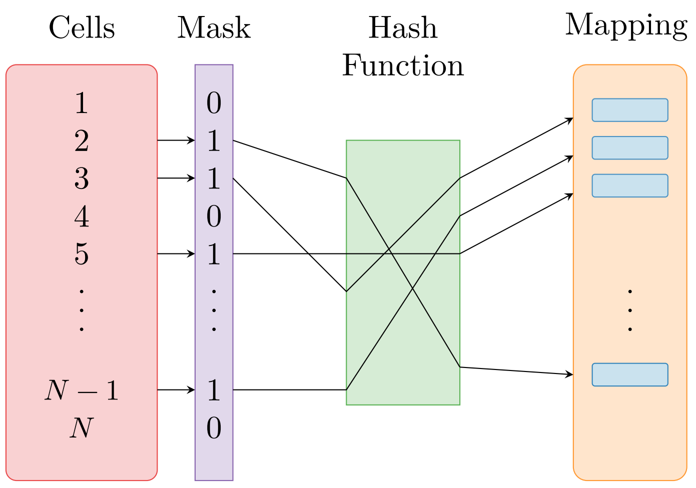


Figure 3.1: schematic of general mapping sequence

3.1.2 Cell Marking

The first step is to decide which parts of the mesh are useful to the library and which parts may be only a performance bottleneck. On a general application, this is a job that is done by the library.

In the current work, the first grid is always considered to be the background grid (containing the far-field boundaries) and with no nodes tagged as wall (hole-cutting is not performed for this grid).

This means that the hole-cutting algorithm will only be performed for the body-fitted grids. This assumption greatly simplifies the implementation algorithm in the developing section.

Considering body-fitted grids, near wall boundary cells and nodes are useless for the OGA algorithm as they are far from the overset boundary and their resolution is often different from the overset cell resolution (by scales in volume). In addition, these cells may often be highly skewed, as in many cases, boundary layer resolving is desirable. This can cause performance issues in query identification and donor search.

For that reason, a more broad area of cells is given to the library, tagged as wall, that is distant by a small number of cell-lengths to the overset boundary.

To determine the cut-off distance of the cell marking in a general way, a basic cell length has to be introduced. This length is defined as the square root of the maximum surface for all the overset boundary faces. After this identification, a distance search is conducted by providing the maximum distance of the search (times the above length). Of course, there is no reason to consider that every process has the closest cells to its portion of the overset boundary.

To circumvent this issue in parallel implementation, the following steps are performed.

a) Every process P_i that owns overset boundary faces for the grids $2, 3, 4$, ... broadcasts all of these faces to every process $\mathrm{P}_j, j = 1, N$.

b) Process P_i finds the maximum face surface of all the boundary faces and takes the square root as a reference length times a portion (user defined). This arithmetic procedure is identical to all the processes $\mathrm{P}_j, j = 1, N$. This portion is usually ranging from $1 \div 9$.

c) Calculate the distance from every cell of the process P_j to every overset boundary face. If the distance of one cell to one boundary face is lower than the defined length, the cell is tagged as 1, meaning that it will be registered in TIOGA, otherwise it is tagged as 0 and is ignored by the library.

By performing this operation, computational effort is saved for cells which we know that there is no chance of being near an overset region. Therefore, the hole-cutting examines less cells for a bigger bounding box.

This part is the most computing intensive, firstly due to broadcasting so many data and secondly due to the double loop over cells and faces.

At this point, the first mask function is defined, which appends only the *tagged as 1* cells to a new array with a length equal to the total number of cells in the starting array (NTE_{tioga} out of NTE).

3.1.3 Sorting

The next step involves the numbering so that the TIOGA library receives the cells and the connectivity.

Firstly, cells are sorted by grid number and the grid mapping is constructed. This function does not change the length of the resulting array. Sorting is performed both on cells and faces, as well as for nodes.

Additionally, an array is created where the total number of cells of a specific type is concentrated (tetrahedra, pyramids, prisms, and hexahedra) for the next sorting.

Before registering the volume mesh cells, incrementing loops are performed for wall and overset boundary node registering.

For the volume mesh, every previous mapping is used on a composition function after the accumulated number of cells/nodes is subtracted. Now that the nodes are index-compatible with the library, the node connectivity is registered and the type of cell mapping is constructed (following if conditions).

After registering every grid, the resolutions are specified by some reference quantity indicating which grid is the coarser.

It is noted that new arrays for storing the geometry are introduced, increasing by a large amount the storage intensity of every simulation.

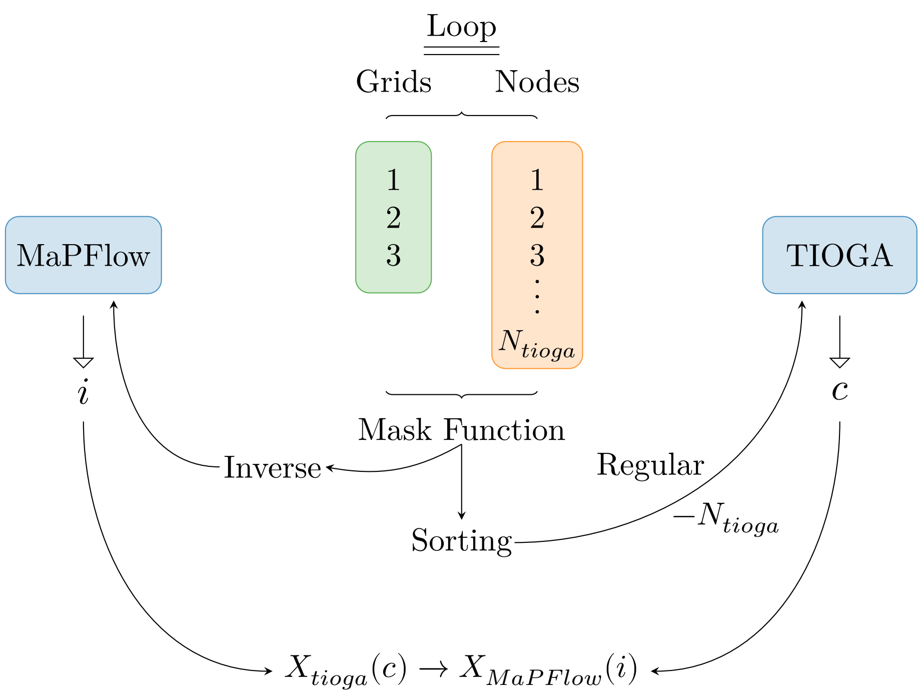


Figure 3.2: Schematic of the index mapping for coordinate assignment

3.2 Interpolation Schemes

3.2.1 Interpolation Algorithm

At this point, the overset grid assembly is established and the TIOGA library has provided a donor cell for every cell that is tagged as a receptor. Of course, along with the cell index, other necessary information that is obtained is the grid number of the donor cell and the index number of the process P_i to which it belongs. All the necessary data are then communicated from process P_i to process P_j such as:

- cell center coordinates $\mathbf{x}_d$
- cell center variables $\vec{Q}$
- cell center gradients $grad\left\{\vec{Q}\right\}$

Auxiliary data are also communicated such as:

- cell index (for storing nearest neighbours)
- iblank value
- cell volume (shape parameter calculation for RBF)

Variable interpolation is always done from cells to cells without any interaction with cell faces. This is preferred due to the solver being cell-based and the cell-centered value being considered the numerical approximation of the solution.

After this point, the solver calculates the fluxes through the left and right states from either an approximated or an interpolated value of the cell center.

3.2.2 Radial Basis Interpolation (RBF)

The current interpolation scheme is widely applicable, especially in multi-dimensional scattered data interpolation problems (astrophysics, image processing). The scheme may be implemented in a local or global manner as many parallel algorithms have been developed. The algorithm is based on multiplying the node value with a function of a radial dependence (and possibly with a shape parameter $\varepsilon > 0$). After building the basis of the finite space that the field will be projected onto, a linear system has to be solved, which unfortunately consists of a dense matrix of coefficients (usually). The method ensures, that if the nodes are distinctive among the rest (coordinates do not coincide) the matrix of coefficients will always be positive definite and the system $A\mathbf{x} = B$ is solvable.

Given N number of interpolating points in 3D space (generally $N - D$) with coordinates and values $\mathbf{x}_k,\ y_k,\ k = 1, N$ respectively, the radial basis function is built.

$$\varphi_k\left(\mathbf{x};\, \varepsilon\right) = \varphi\left(||\mathbf{x} - \mathbf{x}_k||;\, \varepsilon\right) \tag{3.1}$$

In many cases, the shape parameter is acting as a scaling function on the norm $||\cdot||$. Usually, the Euclidean norm is selected for this instance.

The interpolated function is constructed.

$$s\left(\mathbf{x}\right) = \sum_{k=1}^{N} c_k\, \varphi_k\left(\mathbf{x};\, \varepsilon\right) \tag{3.2}$$

The linear system of equations is built by applying the following constraints.

$$\begin{aligned}
& s\left(\mathbf{x}_i\right) = y_i, \quad i = 1, N \\
\Rightarrow & \sum_{j=1}^{N} c_j\, \varphi^{\varepsilon}\left(||\mathbf{x}_i - \mathbf{x}_j||\right) = y_i \\
& A\,\mathbf{c} = \mathbf{y} \\
& A_{ij} = \varphi^{\varepsilon}\left(||\mathbf{x}_i - \mathbf{x}_j||\right)
\end{aligned}$$

The role of the shape parameter is to control the active support of the basis function. For example, if $\varepsilon \to 0$ then the basis functions tend to be flattened and the matrix of coefficients has a very large condition number. However, when $\varepsilon \to \infty$ then the basis function has a sharp spike shape and contributions to the approximation point are practically zero.

Some of the most usual radial functions are:

- Gaussian

$$\varphi(\mathbf{x}) = exp\left\{-(\varepsilon\mathbf{x})^2\right\} \tag{3.3}$$

- Multiquadric

$$\varphi(\mathbf{x}) = \sqrt{1 + (\varepsilon\mathbf{x})^2} \tag{3.4}$$

- Inverse

$$\varphi(\mathbf{x}) = \frac{1}{\left(1 + (\varepsilon\mathbf{x})^2\right)^p}, \quad p > 0 \tag{3.5}$$

- Polyharmonic Splines ($r = ||\mathbf{x}||_2$).

$$\varphi(r) = \begin{cases} r^k & , k \text{ odd} \\ r^k \log r & , k \text{ even} \end{cases} \tag{3.6}$$

The case of polyharmonic splines is an example where a shape parameter is absent. This feature makes this type of radial basis appealing in the case that the density of scattered points is not uniform and a variable shape parameter may be employed. Another reason that the shape parameter is not the best option is that it has to be defined by taking into consideration the donor cell volume and the distances for every donor. The donor cell volume is an unknown quantity to the processor P_i who receives donor information and therefore it has to be communicated.

3.2.3 Inverse Distance Weighting (Shepard's Method)

This type of interpolation uses a distance weight for every neighbouring interpolating point. The smoothing function satisfies explicitly the interpolation constraints for every interpolating node, and therefore weights do not have to be found by solving a linear system.

The weights are defined as:

$$w_i(\mathbf{x}) = \frac{1}{\left(||\mathbf{x} - \mathbf{x}_i||_2\right)^p}, \quad p > 0 \tag{3.7}$$

The approximated function is the average with respect to all the weights.

$$s(\mathbf{x}) = \frac{\sum_{i=1}^{N} w_i\left(\mathbf{x}\right) y_i}{\sum_{i=1}^{N} w_i\left(\mathbf{x}\right)} \tag{3.8}$$

The choice of the exponent p has an effect on the resulting approximation and acts as a shape parameter, similar to ε defined in 3.2.1. The value of the exponent, however, is not considered to be of great importance and, as a common practice, its value for 3D applications is chosen so that $p > 3$. The choice of the exponent p has an effect on the resulting approximation and acts as a shape parameter, similar to ε defined in 3.2.1. The value of the exponent, however, is not considered to be of great importance and, as a common practice, its value for 3D applications is chosen so that $p > 3$.

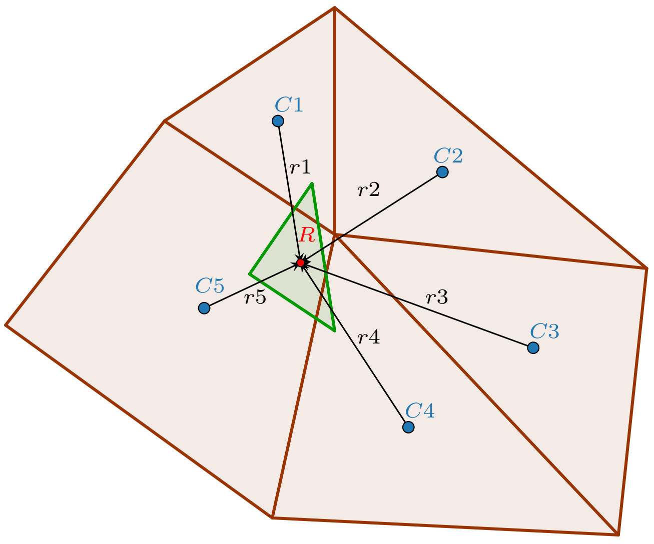


Figure 3.3: Inverse Distance Weighting schematic

3.2.4 Nearest Neighbour

The easiest interpolation scheme, which, in contrast with the previous, is not of high order type, assigns directly the value of the nearest neighbour to the value of the receptor point. The order of accuracy is very low, and as a result, this method is used with success in two-phase flows where a discontinuity is present.

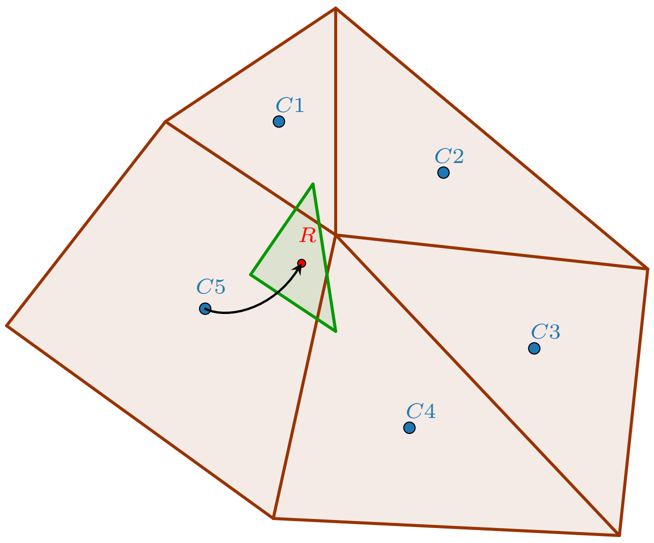


Figure 3.4: Nearest Neighbour Schematic

This type of approach, although applicable in any case, is often insufficient (as far as accuracy is concerned) as it may introduce numerical diffusion. Aside from accuracy issues, it remains a robust way to communicate data among grids.

3.2.5 Axis-Aligned Local Interpolation

The last and most interesting type of local interpolation scheme that was developed, nick-named *Axis-Aligned* has an exception which makes it distinctive among all previous schemes. This scheme does not treat donor cell values and positions as in a pure scatter point interpolation problem (which would make more sense in the case of a Lagrangian interpretation). As a matter of fact, this scheme uses further information already provided by the solver, such as the field data gradients, and tries to distinguish between steep and smooth interpolation directions.

This feature helps by providing a more intrinsic approach, especially in the case that a density jump is expected to occur.

Firstly, the direction of the highest rate of change has to be found. Introducing the vector of the gradient $\vec{n}$ with a starting point $A \equiv \mathbf{x}_R$, the first thing that is noted is that "*if a field discontinuity (jump) is present close to the receptor point, it would most likely appear along the direction of* $\vec{n}$".

Consider N_d number of closest donors to the receptor point. It must be first checked that the receptor point lies within a certain region among donors. That region is considered in our case the Bounding Box constructed by the minimum and maximum values of coordinates (x_d, y_d, z_d) of all donors. This constraint would also be present for every type of local interpolation scheme, e.g. tri-linear. This check is done by straight-forward coordinate checking.

The gradient of the receptor point is approximated by distance-averaging every gradient component while taking into account all the donor cell centres (N_d in number).

Now that the direction of the steepest change is established and just because we have a starting point A and a normal $\vec{n}$, we can fully define a perpendicular plane, denoted from now on as $\{A, \vec{n}\}$.

This plane is now used to define a left and right orientation for the donor cell centers, that is, whether they lie left or right with respect to the plane and a left-to-right reference, the direction of the normal $\vec{n}$.

Once donor cell centers are tagged as L or R (left or right) we project their coordinates on the axis that passes through $\vec{n}$ and A. Next, field variables are approximated on the projected points via a first order Taylor expansion (using the already known gradient).

Once coordinates and field variables are projected and aligned on the axis, the interpolation problem is now reduced to a 1D approximation case with N_d abscissas and field values where a discontinuity is present.

In the worst case, a discontinuity has to be tracked along the axis and a proper approximation on A has to be produced, whereas in the best case, we have a smooth solution where a quasi-approximation has to be found.

The steps of the algorithm are listed below for compactness.

1) Set a number of donors for the local approximation: N_d.

2) Find the N_d nearest donor cell indices.

3) Construct a bounding box from the nearest donor min and max values.

$$BB = \{\mathbf{x}_{min}, \mathbf{x}_{max}\} \tag{3.9}$$

4) Check whether the receptor point A lies inside the BB and if not, then iterate by increasing the number of donors from the total batch.

$$x_{i_{min}} \leq A_i \leq x_{i_{max}}, \qquad i = 1,3 \tag{3.10}$$

5) Use distance averaging to approximate the receptor gradient for all components of $\vec{Q}$.

$$r_d = ||\mathbf{x}_A - \mathbf{x}_d||_2 \tag{3.11}$$

$$\{\nabla Q_i\}_k \approx \frac{\sum_{d=1}^{N_d} r_d \cdot \{\nabla Q_{d,i}\}_k}{\sum_{d}^{N_d} r_d}, \quad k = 1,2,3 \quad i = 1, dim\{Q\} \tag{3.12}$$

6) define the direction $\vec{n}$ as the approximated normalised gradient from (3.4) for every field variable.

$$\vec{n} = \frac{\nabla Q_i}{||\nabla Q_i||} \tag{3.13}$$

7) Create a parametric expression of the axis perpendicular to the plane $\{A, \vec{n}\}$ that passes through A.

$$\mathbf{x} = A + t \cdot \vec{n}, \qquad t \in \mathbb{R} \tag{3.14}$$

8) Create the vector $\overrightarrow{AD} = \overrightarrow{OA} - \overrightarrow{OD}$ and project $\overrightarrow{AD}$ on the axis.

$$AD \cdot \vec{n} = |AD|\,|\vec{n}| \cos\theta = t_0 \tag{3.15}$$

9) if t_0 is negative, tag as (L) left whereas if positive tag as (R) right.

10) Calculate the projected coordinates of donor cell centers.

$$\mathbf{x}_{d,proj} = A + t_0 \cdot \vec{n} \tag{3.16}$$

11) Every component of Q is then projected using first order Taylor's expansion.

$$Q_{d,proj} = Q_d + \nabla Q \cdot (\mathbf{x}_{proj} - \mathbf{x}_d) \tag{3.17}$$

12) Solve the 1D interpolation problem where the parametric values are t_0 (for every donor).

13) Track the discontinuity (if existent) and perform a suitable interpolation.

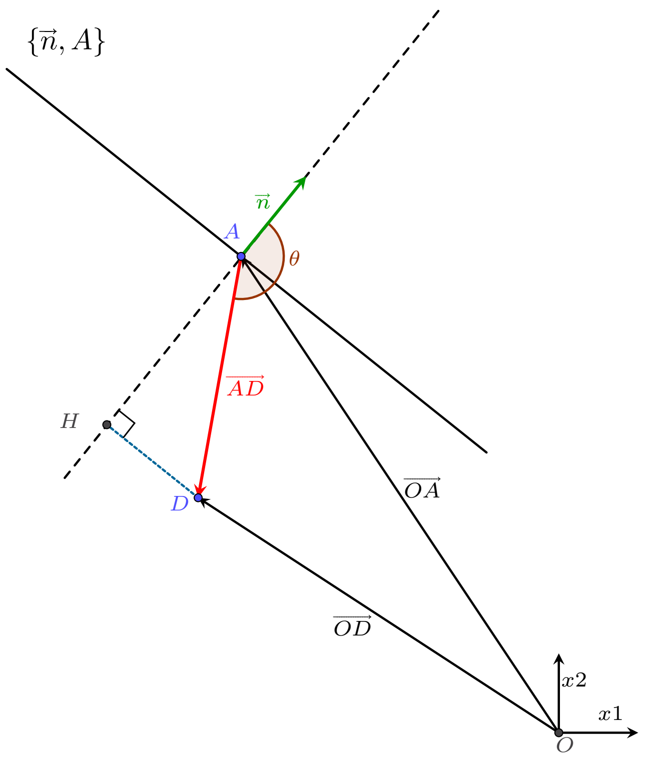


Figure 3.5: Geometric interpretation of the value and coordinate projection on the axis of interest

For the case that the average gradient has a zero length (below a tolerance level), there are two possible causes. The first one is that every donor gradient has a zero length, and therefore the field has a near zero change over this area. The second reason might be that cell centre gradients have both positive and negative components considering the whole batch. For the last case, it is considered that the problem is grid-dependent and the resolution is insufficient to accurately approximate the solution along with its gradients. Therefore, grid refinement has to be performed. For that reason, it is preferred to choose a small number of influencing donors for every receptor so as to minimise the maximum donor distance with respect to cell dimensions.

It has to be mentioned that the cell-centered gradients are computed by the solver using the divergence theorem for scalar fields over the cell of interest.

$$\int_{D_k} \nabla Q \, \mathrm{d}D = \int_{\partial D_k} Q \cdot \mathbf{n} \, \mathrm{d}S \tag{3.18}$$

The face values for the equation 3.18 are computed, usually by averaging the values of the left and right cells or even by calculating their convex combination (linear interpolation). Again, the volume integral is approximated by the constant cell-centered value times its volume (midpoint rule), which is always 2^{nd}-order accurate.

For the case of a zero length gradient, the algorithm bypasses steps from 7 to 13 and a usual multivariate interpolation method is used. This is chosen mainly because a steep direction is non-existent. Now that the field is considered sufficiently smooth, high-order interpolation schemes can be used for increased accuracy.

The performance of the current method will be evaluated in the validation chapter 4.

Another thing to keep in mind about the axis-aligned algorithm is that in many parts, the solver's spatial accuracy is preserved and only the simplest further approximations are performed, such as the gradient approximation, which is of different nature (involves only the approximation of the direction vector), and the direct interpolation with given accuracy.

Lastly, it has to be mentioned that it is not necessary for the receptor point to lie inside a particular bounding box of contributing donors. In this case, an orientation is again found and all the donors are positioned left (L) or right (R). In that case, there are some differences involving the interpolation patterns.

The first is that convex combinations such as linear fitting cannot be used and an upwind or downwind approximation is unavoidable. The second difference is that, due to this upwind or downwind value association, special treatment has to be given to the pressure field, which will be explained in the following paragraphs 3.2.6.

3.2.6 Account for Hydrostatic Pressure

Continuing in the two-phase paradigm, especially for the case in which the gravitational force is of great importance, the hydrostatic difference of the total pressure has to be tracked when going from donor data to receptor. This treatment has to be implemented only when the accuracy of the interpolation is below 1^{st} order and its purpose is to produce a better approximation while keeping in mind that errors deriving from hydrostatic differences play a big role in the overall convergence. In fact, the pressure correction is not implemented in the case of radial basis functions or inverse distance weighting, where the contributions have a weight which depends on the radius in a non-linear way, and therefore the approximation has a higher order of accuracy.

For that matter, hydrostatic pressure consideration is performed only in the nearest neighbour and axis-aligned algorithms (upwind or downwind cases).

$$\rho_m = \alpha_l \rho_w + (1 - \alpha_l)\, \rho_a \tag{3.19}$$

$$\delta p = \rho_m g \, (z_d - z_r) \tag{3.20}$$

$$p_r = p_d + \delta p \tag{3.21}$$

The hydrostatic approximation may seem to be–at least by a first glance–a straightforward pressure correction step from donor to receptor as shown in 3.19.

As mentioned before, there is no issue when using a linear convex combination of two surrounding donors (when the interpolation is expressed with the axis-aligned algorithm) or any high-order scheme. In the case that the interpolation is performed by a direct value assignment (or an upwind-like method), a problem arises in the hydrostatic term which concerns the origin of the density value. The value of the density contains within itself the value of the volume fraction, which is the 5^{th} state variable in the approximation problem and is due to be updated when interpolating (meaning that the donor and the receptor may have different values for the volume fraction). This will not be a problem with the nearest neighbour algorithm where the value of the volume fraction is the same between the receptor and the (closest) donor, and therefore consistency is achieved.

In contrast, when using the axis aligned algorithm, there could be a case where the volume fraction will be interpolated as a convex linear combination (is between a left and right donor) and the pressure will have a direct assignment (upwind or downwind) because of the absence of the left or right donor. This may occur because, for every variable, a different plane will be constructed and the donors are not always both left and right of it.

Revising the above, there exists a case where the hydrostatic addition term 3.19 could have two different values, one where the density takes the volume fraction of the donor and one where the density takes the volume fraction of the receptor.

This may seem a minor problem; however, errors due to hydrostatic inaccuracies are the ones with the highest magnitude (in fact, by scales), mainly because of the large density difference between the two fluids (a scale of a thousand). For example, the error in the u_x component at some point on the overset grid would be of order $\mathcal{O}(10^{-2})$, whereas the pressure error would be of order $\mathcal{O}(10^{1})$. This phenomenon is amplified when the solver uses compressive flux reconstruction schemes, which means that the density differences between donor and receptor are bigger. The result is a pressure source near the free surface level in the overset region even with no wave present (hydrostatic simulation).

This case was observed just because the donor values were always used by default on density assignment to avoid unbounded occasions and high errors.

To remedy this unfortunate case, every time that the interpolation routine is called, the code checks with *if* conditions whether this special case is present (checks for left and right cells for the pressure). From that point, first the volume fraction is interpolated (which is the 5^{th} state variable) and then the pressure and the velocity components. In that way, the hydrostatic term can be calculated with respect to the receptor value of volume fraction and then pressure addition may be done safely.

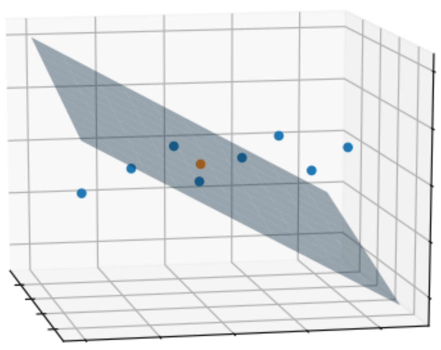


Figure 3.6: Representation of the possible donor orientation with respect to the plane

3.2.7 Flexibility of the Interpolation Process

Now that all the schemes are presented, it has to be noted that not all of them are applicable in all cases, especially when free surface effects are present.

In order to give an example, suppose that a simple first-order Taylor expansion is addressed as an alternative for the low-order nearest neighbour. This means that the field approximation has the following expression.

$$Q_{rec} = Q_{don} + \nabla Q_{don} \cdot (\mathbf{x}_{rec} - \mathbf{x}_{don}) \tag{3.22}$$

In the case of the first order expansion, it is observed that the solution is not generally bounded and the biggest issue is the derivation of the gradient approximation.

Such gradients, especially in flows with density jumps and high z-linear dependency (hydrostatic problems), may give poor approximations and, most importantly of all, be unbounded. This issue can of course be addressed by a limiter that has to be defined on a stencil over donors (and has to be communicated to the receptor-belonging process P_i) but this problem is not addressed in this thesis, as it was preferred that the process of data interpolation is with a more natural way (steep and smooth directions) in strictly steep areas and a more traditional way when dealing with smooth regions (tri-linear or higher order).

Another final note that separates the global interpolation schemes from local is that, usually global schemes tend to achieve an order of scaling with an exponent strictly greater than 1, for example $\mathcal{O}\left(N^2\right)$ or $\mathcal{O}\left(N^3\right)$ (if an LU-decomposition is applied on the matrix of coefficients). This suggests that the nature of the algorithm demands a strong dependence on the number of degrees of freedom. However, local approximation algorithms use only neighbouring information, which is usually about the same in quantity for every control volume, and therefore the dependence on the overall degrees of freedom is weak, giving the opportunity to achieve a near $\mathcal{O}(N)$ scaling order. This algorithmic criterion for the approximation is also desired (and vastly applied) in the nature of every classical solver (finite elements, finite volume).

3.3 Data Storage & Coupling

In this part of the development, the appropriate coupling between the solver and the library is achieved, and therefore the proper information for the cell and node identification is gathered. The mask function for every cell identification is called **iblank** value and is an integer with the following indication.

a) -1: overset node

b) 0: node inside of wall

c) 1: node with solver solution

The first three tasks for the OGA are:

1) register grids

2) preprocess grids

3) perform grid connectivity

These identification integers are obtained by the third subroutine and are the most essential identifier for the interpolation process.

3.3.1 Identification of Cells

These iblank values are used to give instructions to the solver. The strategy behind the coupling is as follows.

Due to the fact that the solver is face-based and the approximated values are stored in the cell centers, the iblank values are assigned to the cell centers.

Generally, the solver for every different iblank value performs different tasks.

iblank	process
-1	values are interpolated at the start of the iteration
0	the solver ignores these cells
1	values derive from the finite volume discretisation

Unfortunately, the library marks every overset cell with lower resolution as a receptor node, which is of $iblank = -1$. This means that there will be a large number of donors and receptors for the communication and interpolation workload. The solver only needs a strip (or a thick sheet for 3D) of overset cells to pass the information to the other mesh, and therefore, loops are performed where useless cells are marked as $iblank = 0$. This process reduces a significant amount of computational and communication time.

The solver in every iteration, calls the following subroutines in the listed order:

1) perform interpolation

2) communicate inter-processor boundary data

3) set boundary conditions

4) calculate gradients at cell centers

5) calculate fluxes

6) calculate left hand side & right hand side

7) update with given algorithm

3.3.2 Donor information

After the grid assembly, the requisite overset information is extracted from the library. This information is for every donor cell and consists of the number of donors that the process P_i contains, their cell number (local value), their receptor process P_j and cell number (for the communication) and the donor points along with every cell (in the finite volume representation, nodes are of no use). Lastly, the target grid is given for the proper grid value match.

3.3.3 Gradient Calculation

As it was mentioned in the interpolation section 3.2, the gradients in every cell are calculated by applying the Green-Gauss theorem on the control volume and then approximating the values on every face. For every face, a left (L) and right (R) cell are used with its values and coordinates to approximate this value. Using the L and R values, usually the cell value is approximated through a linear interpolation, while special treatment is applied to pressure in the case that a gravitational force exists.

There are some cases where one of the two cells (L or R) has an iblank value of zero and therefore has no values. In that case, the face values are comprised only of the other cell values, and the approximation is only of first order.

More specifically, in the algorithm, gradients are calculated only where cells have an iblank value of 1 in the body-fitted grids (where -1 are only cells in the outer boundary) and ignore cells with iblank equals 0 in the first grid (cartesian). That way, values are taken from the approximated or interpolated cells and consistency is achieved.

3.3.4 Flux Calculation

Similar to the gradient approximations, fluxes are only calculated when both of the L and R cells have iblank values different than 0. No flux corrections are applied, and the solver uses interpolated values to solve the approximate Riemann problem (for the inviscid fluxes) or perform value averaging (for viscous fluxes). This step is the most error-intensive and depends strongly on the interpolation process as a flux reconstruction is applied for the L and R states that are projected in some way on the face center.

3.3.5 Left and Right hand side calculation

Regarding value updating on the internal iteration, the solver fills left (linearised) and right-hand side contributions for every cell of the mesh. When the chimera algorithms are applied, the solver ignores cells that are not solved, namely with $iblank < 1$. In these receptor cells, LHS and RHS values are always zero, so no contribution to the iteration error exists from this family of cells.

3.3.6 Summarising

Concluding, the proper connection between the solver and the TIOGA library is achieved when following the steps described above, and a large part of the development time was dedicated there.

After achieving a functioning coupled solver, basic simulations are run and the first test cases are examined just for proof of concept and accurate flow prediction. Meanwhile, further adjustments and corrections were performed, mainly in the structure of the code and the data storage management. By gradually increasing the complexity of the simulations (3D simulations, steady and unsteady) and their meshes, the *chimera* solver is further tested up to its' “breaking” limit.

The largest step for testing the software is depicted in the first test case involving free surface flows. From this part and after, the work was focused mainly on the interpolation process and its aspects.

Chapter 4

Validation

The validation chapter contains all the simulations that were performed in this thesis. The testing and debugging were done on a simple NACA 0012 in uniform angled flow, usually with a visual criterion and quantitative examination. However, the first validating case starts with the examination of the vortex street generation on a stationary 2D projected cylinder 4.1.1. Following, a comparison with an analytical solution is documented 4.1.2 before the introduction of two-phase cases 4.1.3.

The second part is devoted to pure 3D phenomena, with cases such as the open water propeller 4.2.1 and the heave decay of a freely floating sphere from a starting height 4.2.2.

4.1 2D cases

4.1.1 Laminar cylinder

One of the simplest cases for validating the basic performance of the methodology in its aspects is the uniform laminar steady-inlet flow around a cylinder and the production of vortex shedding.

The reason for this choice is attributed to the examination of the feasibility of the current method to communicate field variables between overset grids. A well-transferred (interpolated) field will be validated based on existing contour configurations from single grid simulations.

This case study was the first to be examined thoroughly using comparisons from existing literature (simulations and experiments) during the development process. In this subsection, no comparisons were made between different interpolating schemes, and the only type that was used was the *radial basis function interpolation.*

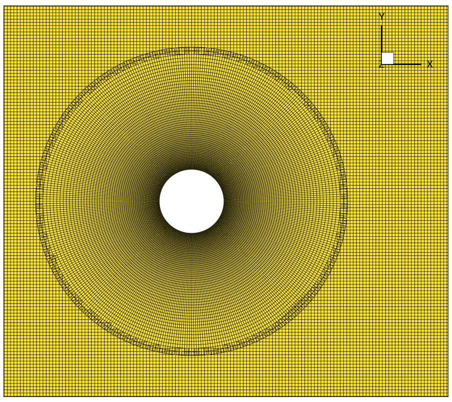
Simulation details

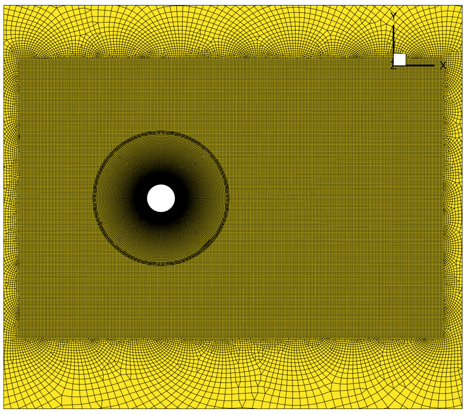
For the case of the body-fitted grid:

The cylinder grid has a radial symmetry and the outer (overset) boundary has a diameter five times the cylinder diameter (5 m).

Number of wall boundary vertices	392
Number of outer boundary vertices	392
Number of Cells for the first grid	19992
Height of first cell	5 mm

The background grid has a dense uniform zone mesh mostly consisting of rectangles with a side length of $(50 - 150\ mm)$ with a y-range and a x-range close to three times the body-fitted grid. The far-field is located at a distance of 50 cylinders from the cylinder center. Outside of the dense zone, the maximum dimension of the cells is increased by up to five times the cylinder diameter in the far-field region.


(a) Grid size in the vicinity of the overset boundary


(b) Grid size near the uniform size dense zone

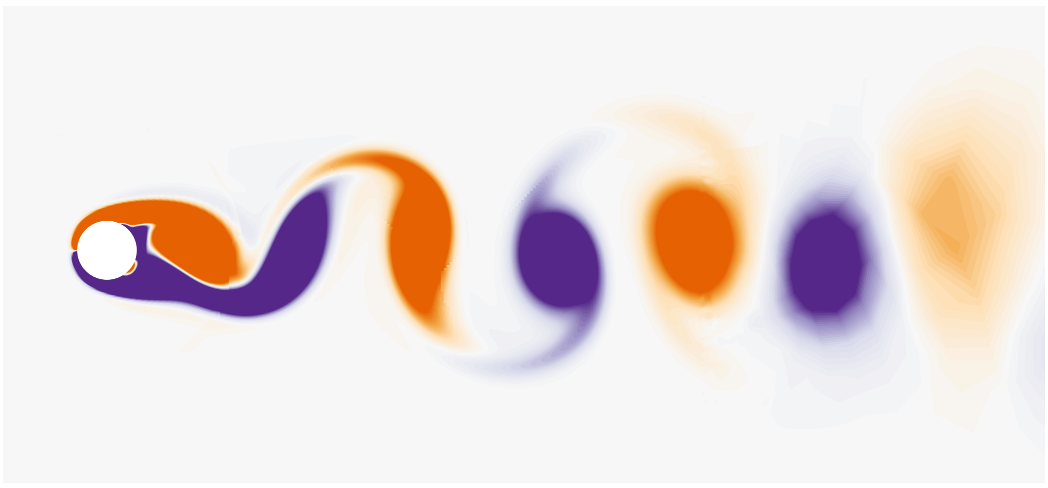
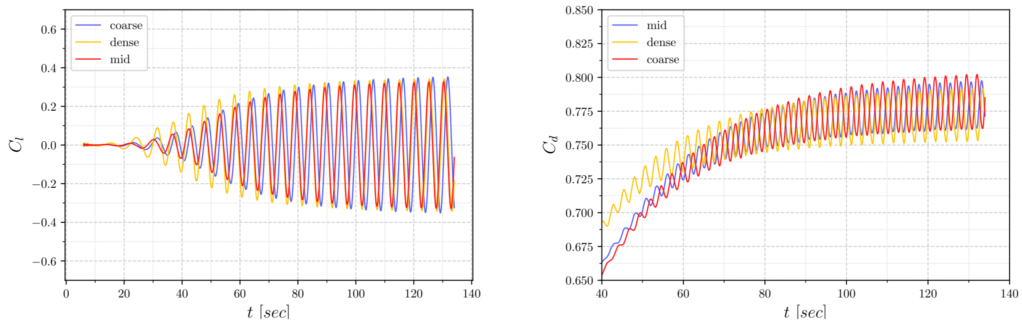


Figure 4.2: Vortex street generation due to unsteady vortex shedding past a cylinder

Grid Independence Simulations

The first step of this validation is to execute the same simulation input parameter with various first cell heights and various background grid cell sizes. The choice of the current work limits the values to: $5,\ 10,\ 20\ mm$.

The validation criterion of the grid independency is the value of lift and drag coefficients over time as well as the dominant frequency present in the current phenomenon.



(a) Lift Coefficient over time (b) Drag Coefficient over time

Figure 4.3: Comparison for 3 different meshes

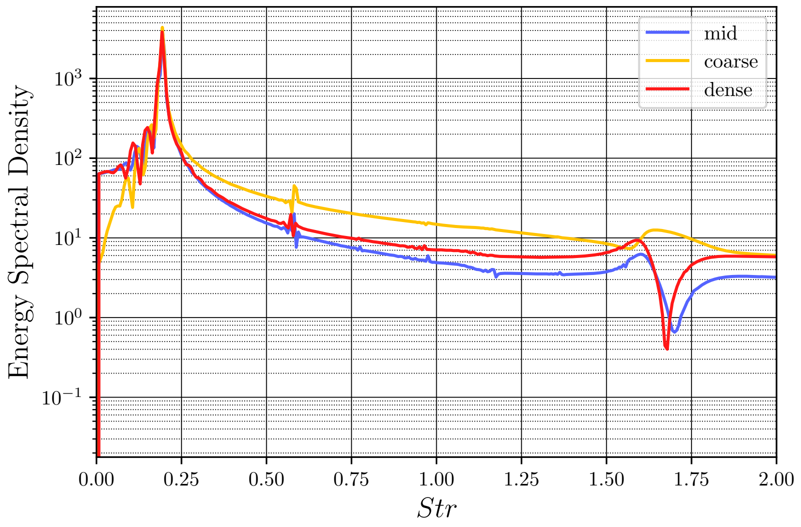


Figure 4.4: Square root of Energy Spectral Density of the Fast fourier transform of $C_l - time$ signal in semi-log y scale over normalised frequency ($strouhal\ number$)

In the last diagram 4.4 and in more detail, the observed peak value (dominant normalised frequency), which is translated as the periodic frequency of the vortex shedding interference for a cylinder of diameter $D = 1\ m$ and a downstream velocity of $V = 1\ m/sec$ is listed down 4.1 for the three cases:

$$Str = \frac{fD}{V} \equiv f \tag{4.1}$$

grid resolution	coarse	middle	dense
Strouhal Number	0.19377	0.19377	0.19288

Table 4.1: Grid dependency comparison of the peak value of the strouhal number.

Comparison

Moving on to the validation of the results, computational experiments were conducted for the case of three *Reynolds* numbers, specifically ($Re = 100, 150, 200$).

The calculation of the *Reynolds* number is done by using the cylinder diameter as the characteristic length.

$$Re = \frac{VD}{\nu} \tag{4.2}$$

The comparison of the performance of the overset grid values communication is based again on the *Strouhal* number and the reference cases are experiments conducted by Norberg ([30]) as well as side-by-side comparison with the sole-grid MaPFlow solver.

The reference simulation in MaPFlow was conducted with a mesh consisting of 38962 cells, mostly of rectangular type and in a radial manner.

Lastly, the comparison includes the empirical relationship between Re and Str number in the laminar cylinder case by *Williamson*, [31].

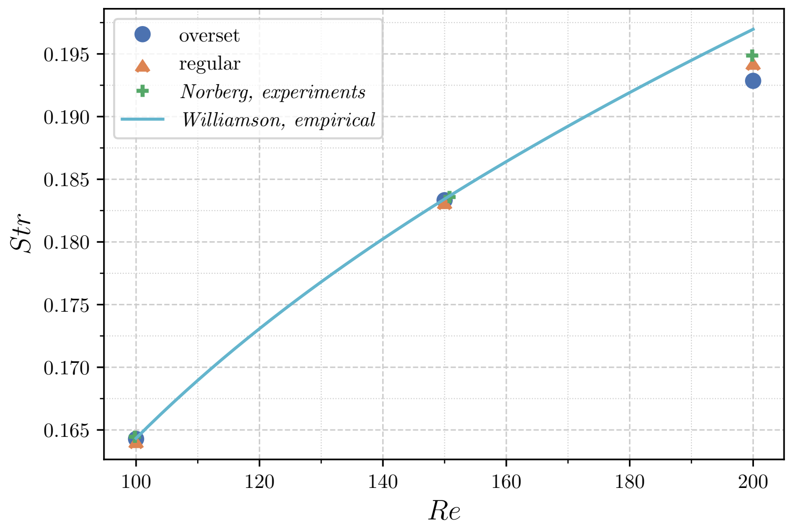


Figure 4.5: Comparison of the calculated strouhal number over Reynolds Number

The presented results verify the widely validated relationship between the two non-dimensional numbers in the case of the laminar vortex street generation of a flow past an infinite (span-wise) cylinder.

4.1.2 Lamp-Oseen Vortex

In this part of the validation, the comparison of different interpolation schemes is presented. As reviewed in the previous section, four types of interpolation are compared: the first two types have a global manner by taking into account a large number of scattered donor points, whereas the last two schemes are of a local nature.

The case of the Lamp-Oseen Vortex is considered as a reference analytical solution to the 2D problem involving the presence of viscosity [32].

Regarding the setup of the solution, consider a vortex centred at $\mathbf{x}_c$ with the distribution of the following expression:

$$\mathbf{r} = \mathbf{x} - \mathbf{x}_c\,, \qquad r = ||\mathbf{r}||_2\,, \qquad \omega = \frac{\Gamma}{\pi}\frac{1}{\sigma^2 + 4\nu t}\, exp\left\{\frac{-r^2}{\sigma^2 + 4\nu t}\right\} \tag{4.3}$$

For $t = 0$ the initial value of the vortex distribution is calculated. The flow is considered laminar and the circumferential velocity and pressure have the following expressions:

$$u_\theta = \frac{\Gamma}{2\pi r}\left[1 - exp\left\{\frac{-r^2}{\sigma^2 + 4\nu t}\right\}\right] \tag{4.4}$$

$$p = \rho u_\theta^2 \log(r) + p_\infty \tag{4.5}$$

The numerical experiment is performed by introducing a rigid cylinder which induces the above velocity and pressure fields for $r > R$, where $\sigma = R$ is the cylinder radius. The following assumption is done by implying equal circumferential velocity at $r = R$ in the expression 4.4 as the circumferential velocity of a particle attached in the cylinder perimeter (expression of the no-slip condition).

The angular velocity (Ω) of the rotating cylinder respects the value of the circumferential velocity in the perimeter and is derived by the following relation.

$$u_\theta(r = R, t)\,\hat{\theta} = (\Omega\hat{z}) \times \mathbf{r} \tag{4.6}$$

$$\Omega = \frac{u_\theta}{r} \tag{4.7}$$

Additionally the following parameters are set.

$$R = 0.5 \tag{4.8}$$

$$\Gamma = \pi \tag{4.9}$$

$$Re = 200 \tag{4.10}$$

In the present work, the spatial convergence is tested where the L_2 and L_∞ norms are displayed over grid density.

The grid density is assumed to be equal to the fraction of cylinder radius R over first cell height h_{first}, namely R/h_{first}.

$$L_2 = \sqrt{\sum_{k \in cells} \left(Q_k - Q_{an}\left(\mathbf{x}_k\right)\right)^2 \cdot D_k} \tag{4.11}$$

$$L_\infty = \max_{cells} \left|Q_k - Q\left(\mathbf{x}_k\right)\right| \cdot \sum_{cells} D_k \tag{4.12}$$

All of the simulations are run with a constant $CFL = 0.5$ and completion is set on the 20th time step. The value of the time-step compatible with the CFL condition is taken by using as a discretisation distance the smallest cell length. Then, norms are calculated on the last time step, and at that point, the comparison is presented.

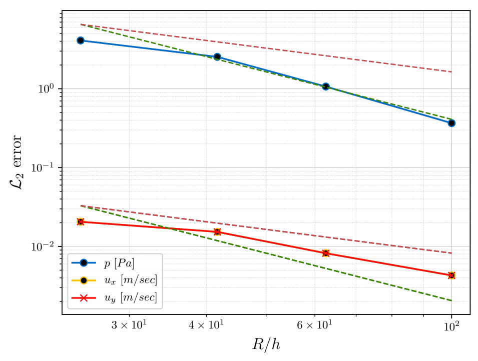
During the convergence rate study, four different grid refinements are performed.

Additionally, results are presented for one grid (no overset grids with a total mesh circular motion) for visual comparison.

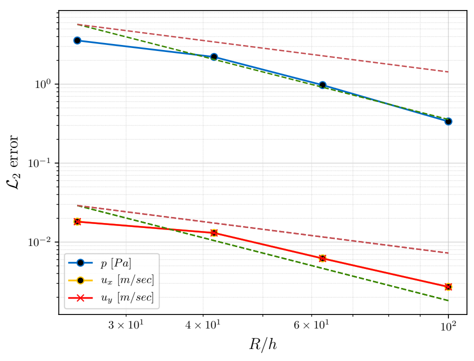
The scaling of the following convergence study could also be normalised by the total domain volume, which is a constant quantity on all occasions and was therefore omitted.

The dashed lines in the graphs represent the first order and the second order angle between x and y axis.

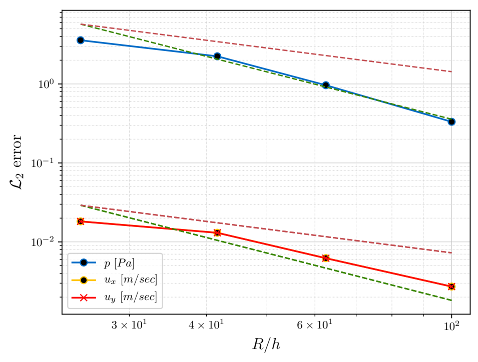
For the overset case, beginning with the L_2 norm:



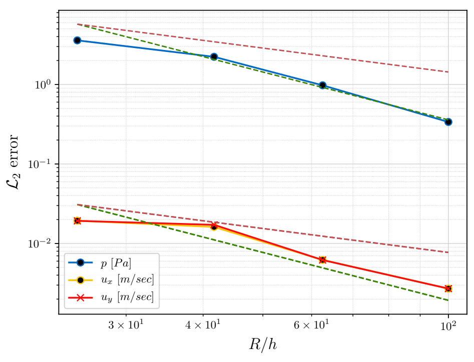
(a) Inverse Distance Weighting



(b) Radial Basis Functions

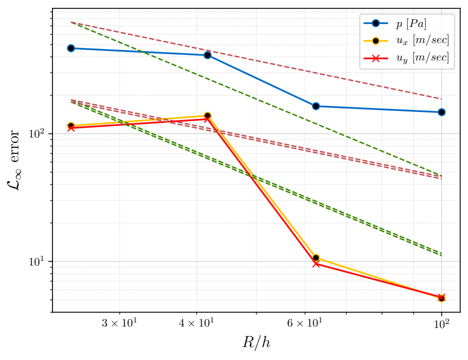


(a) Nearest Neighbor

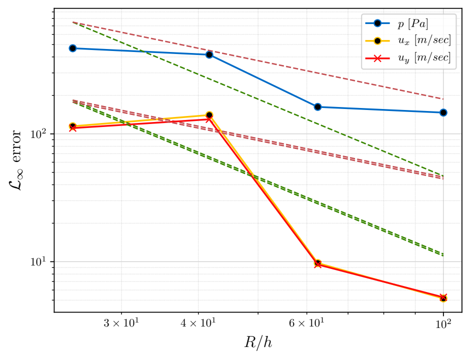


(b) Axis-Aligned

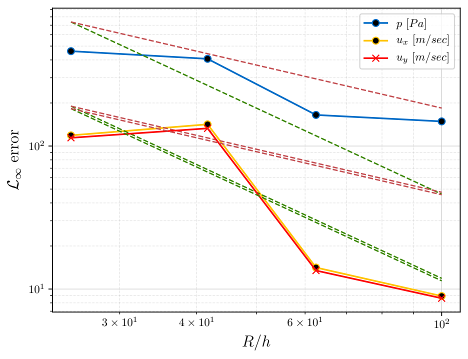
In the case of the $\mathcal{L}_\infty$ norm:



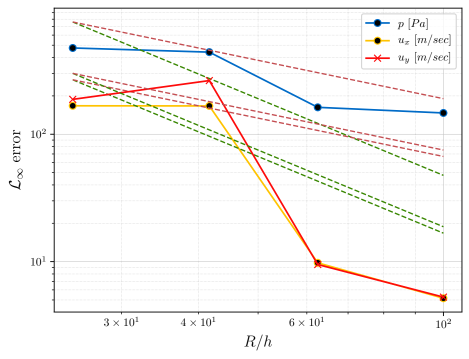
(a) Inverse Distance Weighting



(b) Radial Basis Function



(a) Nearest Neighbor



(b) Axis-Aligned

Regarding the above results and starting with the $\mathcal{L}_2$ norm, it is observed that each and every one of the described schemes can retain by some degree the convergence rate of the solver. Furthermore, the inverse distance weighting scheme seems to introduce some numerical diffusion in the velocity components, whereas the axis-aligned method has a well retained convergence rate.

The performance of the inverse distance method was an expected result, as this particular type of interpolation is known for smooth approximations on imaging with a smoothing indicator, the exponent p. This exponent has to be calibrated when expecting highly accurate results.

In the following thesis, this parameter was not calibrated as it does not lie among the problems of the methodology. Of course, its value would be dependent on scatter point densities and field fluctuations and, of course, a tuning much like this of the shape parameter tuning in *RBF* could be done with the difference that it is no longer a scaling constant and the definition by mesh volume or minimum distance would have a different expression.

Similar conclusions can be brought in the case of the $\mathcal{L}_\infty$ norm. Exceptions involve poorer performance for the nearest neighbour method (as it is expected due to low order) and the fact that pressure tends to follow a smaller convergence rate.

4.1.3 Wave Propagation

Until this point, the validation has a basic standard that follows famous cases and analytical solutions.

The next case study that is examined takes the difficulty level up by one step by introducing discontinuities in the solution field. The reason for the introduction of such a case is the examination of the performance of the axis-aligned interpolation scheme over its standard competitor, the nearest neighbour.

This comparison is done by measuring the wave amplitude with gauges (similar to the experimental procedure) before and after the presence of the small grid.

It is noted that all the simulations in the current case are inviscid and the discontinuity problem is tackled using the Euler Equations.

As a reference, a simulation will be run with only one grid and the initial loss of amplitude will be noted.

Numerical Experiment setup

To perform the following numerical experiment, let us consider a plane free surface that is at rest in two-dimensional space. The concept is to generate a wave of one harmonic frequency ω that is compliant with one of the existing wave theories (e.g., the Airy Theory of linear waves). The entire domain is of rectangular type and is distinguished by three zones:

a) Wave Generation Zone

b) Solution Domain

c) Wave absorption Zone

In the first zone, the creation of the propagating waves is performed through source terms, which is the momentum equation. The generation zone is located on the left and near the far-field boundary. The length of this zone is approximately one wave length, and the source term has the following expression:

$$\vec{S}_w = C_w \rho_m \left(\mathbf{u} - \mathbf{u}_{tar}\right), \qquad \mathbf{x} \in \mathcal{D}_{gen} \tag{4.13}$$

The entity C_w is a function of the normalised length from the far-field up to the solution interface. Following the notation of [20], the expression of C_w is presented.

$$C_w = \alpha \frac{\exp\{x_r^n\} - 1}{e - 1} \tag{4.14}$$

$$x_r = \frac{x_s - x}{x_s - x_e} \tag{4.15}$$

The value of α is not to be greater than 200 with an $\mathcal{O}\left(10^2\right)$ order of magnitude.

As equation 4.15 points out, x_r states the normalised length and finally the exponent n usually takes values ranging from $2:5$ which is used for regulating the slope of the generation function.

The target field $\mathbf{u}_{tar}$ is a velocity field produced by a semi-analytical solution from the Stokes Theory and the cnoidal Theory presented thoroughly in [33].

The second zone is the zone where we study the overset performance. It is the regular domain without any sources or sinks for the solution. The length of this zone is approximately two wavelengths, and the overset grid spans up to around $\frac{4}{5}\lambda$ length.

The absorption zone, similarly to the generation one, uses the expression 4.13 with the difference that this time the goal is the damping of the vertical component of velocity. In our case the $\mathbf{u}_{tar}$ has a zero x-component and a desired v_∞ z-component.

The water tank has a constant depth of $h = 5\ m$ and the entire length of the domain spans in 4.5λ length.

The harmonic wave characteristics are:

Wave Characteristics		
H	0.05	m
λ	8	m
T	2.264	sec
g	9.81	m/sec^2
k	0.7854	-
ω	2.775	rad/sec

The grid structure consists of hexahedra (rectangular in 2D) cells which have an increasing y-span as they become more distant from the undisturbed free surface level. The resolution is such that in the vicinity of the flat free surface, the mesh consists of 15 cells per wave height H. Respectively, the x-span of the cells is constant over the regular domain and equal to $\lambda/150\ m$. In contrast, in the generation and absorption zones, the cell length in the x direction is gradually increasing while moving closer to the far-field boundary. At the bottom, no-through boundary conditions are imposed, and at every other boundary surface, the boundary condition is of the far field inlet type.

Considering the time discretisation, a second order implicit (in time) scheme is employed with a constant time step at $\Delta t = 5 \cdot 10^{-3}\ [sec]$ which is approximately equal to $T/450$, where T is of course the wave period.

For a more intuitive and fairer comparison, the smaller grid, which spans for two times the wave amplitude on both sides of the flat free surface (z-direction), has the same resolution and grid structure as the background grid. The only difference is that during the mesh creation, the parameters for length increasing are changed by a very small amount, and after the mesh creation, the grid is also shifted in the x-direction by a fraction of the cell size. This is done to avoid any topological conflicts in the TIOGA library when performing the various steps for the grid assembly. Otherwise, all query points would have a twin cell centre in the other grid and an error would occur.

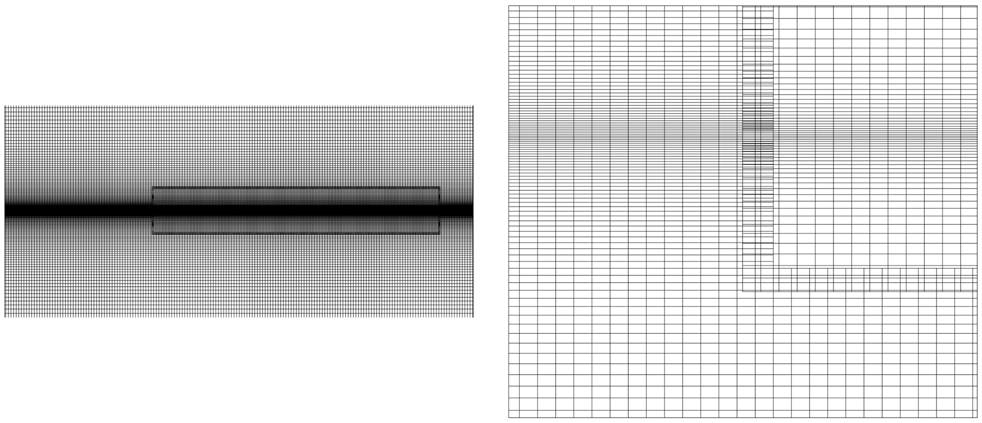


Figure 4.10: Details of the mesh creation for the current simulations

The comparison is performed with the following procedure:

- Two gauges are placed before and after the inner grid that measure the free surface elevation.
- The simulation is ran, firstly with only one grid on the pure solver and without any interpolation. After a time interval where the phenomenon is time-harmonic, the free surface elevation signal is extracted by the two gauges and is compared on a graph.
- Next, the simulations with the overset grid are ran, firstly using the nearest neighbour scheme and then using the axis-aligned scheme.
- The signals from the last gauge (after passing through the overset grid) are extracted and gathered along with the previous signals.
- The performance is tested by comparing the loss of amplitude for the second gauge with reference the first gauge.

The resulting dynamic pressure field after a certain amount of time is visualised below.

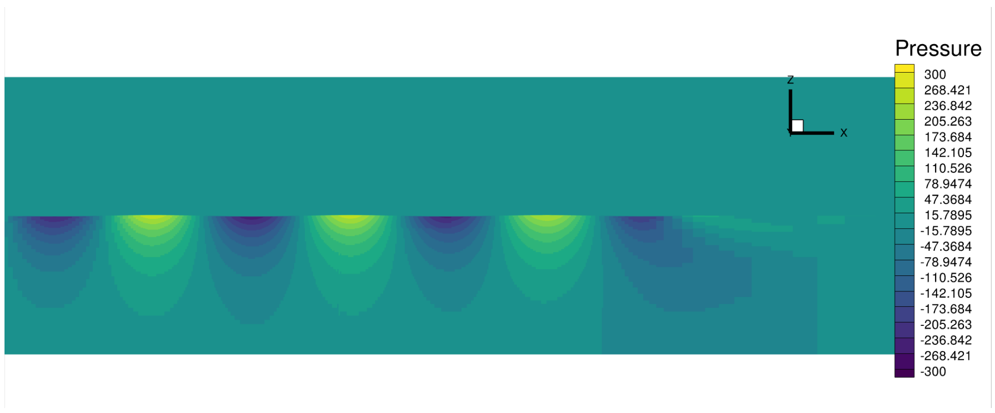
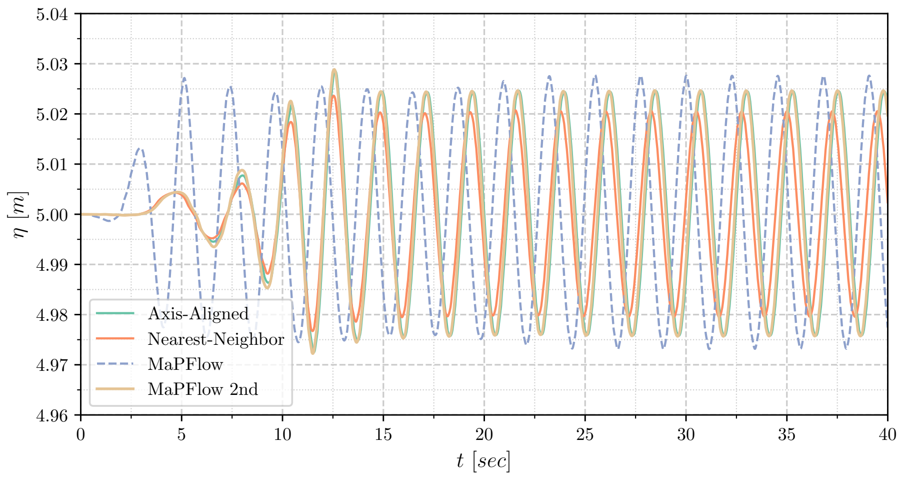


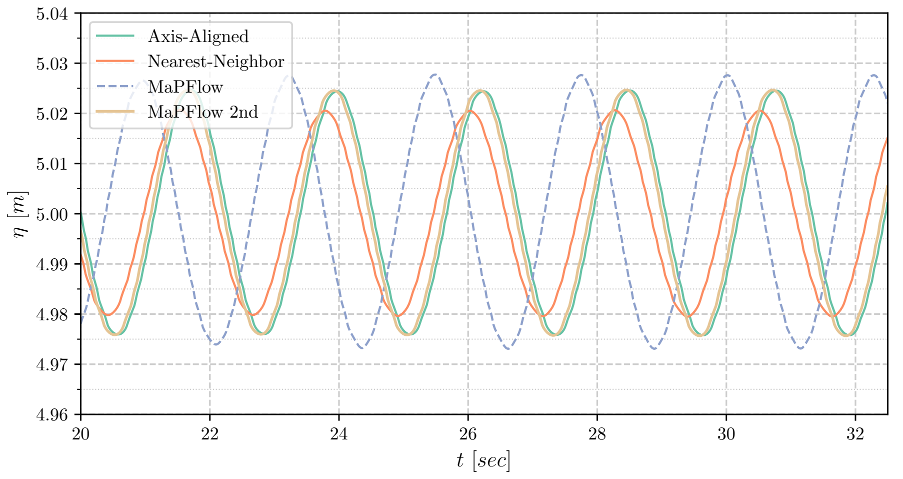
Figure 4.11: Contour Plot of the dynamic component of the pressure field in $[Pa]$.

The plot 4.11 above is presented for better understanding of the generation (left) and the absorption zone (right), as well as the periodic nature of the dynamic pressure field.

Following, the comparison over the gauge signals is presented.



(a) Free surface elevation signals before (dashed) and after the inner grid



(b) Free surface elevation signals before (dashed) and after the inner grid (focused in a smaller time interval)

Figure 4.12: Station free surface elevation signal

By examining the above results of the stations' free surface elevation η [m], it is, first of all, noted that the pure solver simulation (only one grid) which is marked as *MaPFlow* and *MaPFlow* 2^{nd} in the graph for the first and second gauge respectively, has a small amplitude reduction. This implies that numerical diffusion exists, which is of course dependent on the solver and the discretisation. The quantification of the numerical diffusion (on a pure advective phenomenon) is presented exactly with this wave amplitude reduction.

It is also noted that there is an improvement in the energy loss between the newly presented scheme and its nearest neighbour. This is clearly visible in the second graph 4.12b where the results indicate that the interpolation accuracy retains the order of spacial accuracy of the solver by a fair amount. Quantified results of the above performance are presented in the table below.

Simulations	Wave Height $[m]$	Percentage of Reduction
MaPFlow 1^{st}	0.055	– %
MaPFlow 2^{nd}	0.052	5.599 %
Axis Aligned	0.049	10.602 %
Nearest Neighbor	0.041	24.544 %

Table 4.2: Wave height reduction percentage from the 1^{st} station to the 2^{nd} normalised with the first station height.

This comparison table presents the total height loss both from approximation and from interpolation as it is compared with the first station amplitude. The error that is caused by the solver itself is a common feature for all the low-order solvers that focus on the approximation of the volume fraction. A significant part in the performance of the solver, as mentioned in [20], plays the face reconstruction problem, especially in the case of the volume fraction, where it is treated with many proposed so-called ”compressive” schemes that minimise the numerical diffusion while trying to retain spacial accuracy.

To compare the interpolation error only between the two methodologies, the 2^{nd} station is taken as a reference, and the results are presented in the following table.

Simulations	Wave Height $[m]$	Percentage of Reduction
MaPFlow 2^{nd}	0.052	– %
Axis Aligned	0.049	5.300 %
Nearest Neighbour	0.041	20.068 %

Table 4.3: Wave height percentage reduction with respect to the second station.

It has to be noted that the effects of data interpolation are not only shown in the amplitude reduction but also in the phase change.

This effect will be shown, again, by performing a Fast Fourier Transform on the stations’ elevation signals for all the cases.

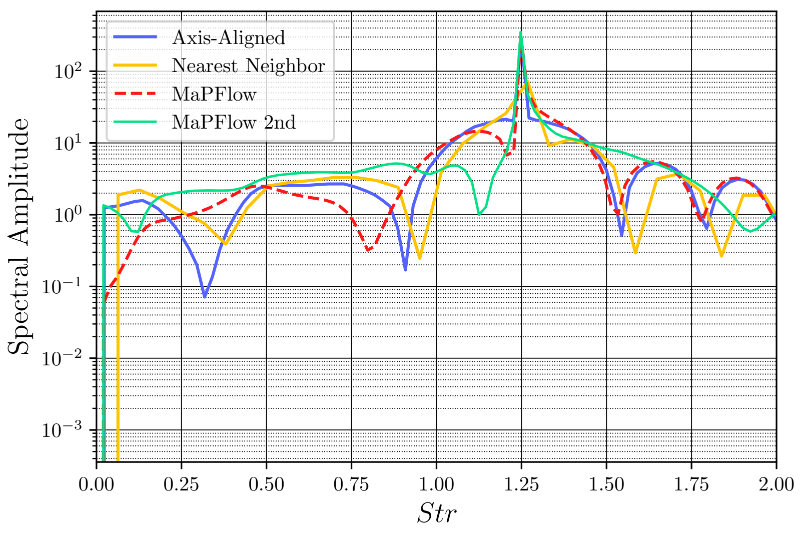


Figure 4.13: Dominant normalised frequency of the surface elevation signal for the first gauge and the second in 3 cases

The *Strouhal* number in this case is the frequency times the wave period.

$$Str = f \cdot T \tag{4.16}$$

The dominant (normalised) frequency above is compared in all cases and presented in a tabular form.

First Gauge	1.2578	
MaPFlow	1.2578	+0%
Axis Aligned	1.2625	+0.37%
Nearest Neighbour	1.2679	+0.81%

Table 4.4: Comparison of strouhal number with the first gauge

4.2 3D Cases

The next step of the validation process involves the solution of pure 3D problems in the engineering and scientific sectors.

The key feature of the methodology that is mentioned repeatedly in this thesis is scalability, i.e., the ability of the software to scale on multiple memory-separate processing units. Apart from this, the ability of the library is tested by finding the desirable number of donors and receptors on a pure 3D unstructured mesh consisting of many different cell types.

These cases were the second type of simulation tested during the development process. This was done right after the data communication problem was solved and the interpolation scheme was at an acceptable level (in terms of accuracy). Furthermore, many issues were remedied and tested concerning the library and the information that was extracted from it.

In this section of the validation, two cases are examined; the first is a steady case and the second is an unsteady one. The first case involves the prediction of coefficients of the open water propeller (thrust & torque coefficients), which are a global approximation indicator for the performance of the method. The second case is the sphere drop test, where the solver is coupled with the rigid body dynamics equations to study the heave decay of a sphere which is dropped from a distance from the free surface. The last case, which is also the hardest in terms of performance and accuracy, measures the heave response and the radiation wave amplitude through several free surface stations.

4.2.1 Open Water Propeller

Starting from the case of the open water propeller, the numerical experiments were conducted on a B-series *Wagenigen* propeller with the following features:

D	0.160	m
P/D	0.880	$-$
A_E/A_0	0.628	$-$
Z	4	$-$
D_h	0.028	m

Table 4.5: propeller particulars

In the table 4.5 the symbols represent the following.

- D: propeller nominal diameter
- P: pitch
- A_E/A_O: blade area ratio
- Z: number of blades
- D_h: hub diameter

The comparison contains a steady (relative reference frame) single grid *MaPFlow* simulation as well as experiments on the aforementioned propeller.

The simulations are run with a constant Reynolds number:

$$Re = \frac{V \cdot D}{\nu} = 5.5 \cdot 10^5 \tag{4.17}$$

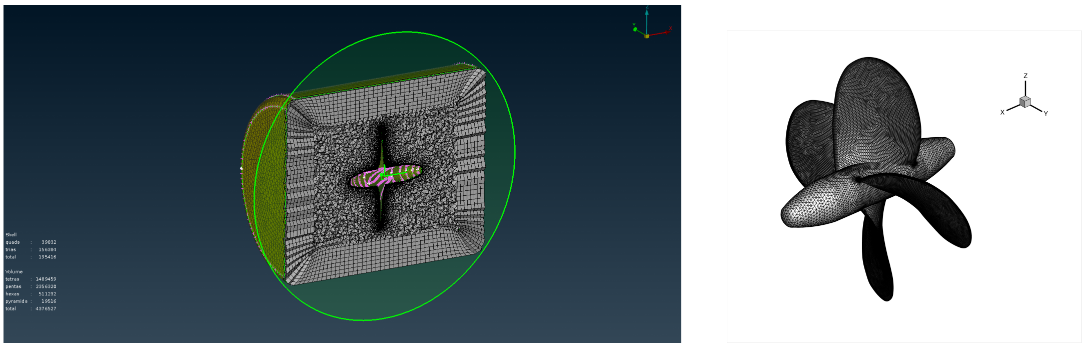
With V we refer to the magnitude of the inlet velocity vector, which is parallel to the x axis.

The simulation is performed in an unsteady configuration with a rotating body fitted grid around the propeller and a stationary background grid.

The fluid is considered viscous and a turbulence model is required. In our case, the $k-\omega$ SST (RANSE) model was used.

The boundary layer was fully resolved with a max y^+ value measured at 0.65 approximately.

The configuration of the mesh is presented in the following figures, taken from the post processing.



```
Shell
quads     :    39032
trias     :   156384
total     :   195416

Volume
tetras    :  1489459
pentas    :  2356320
hexas     :   511232
pyramids  :    19516
total     :  4376527
```

Figure 4.14: Representation of the volume and surface mesh of the propeller grid.

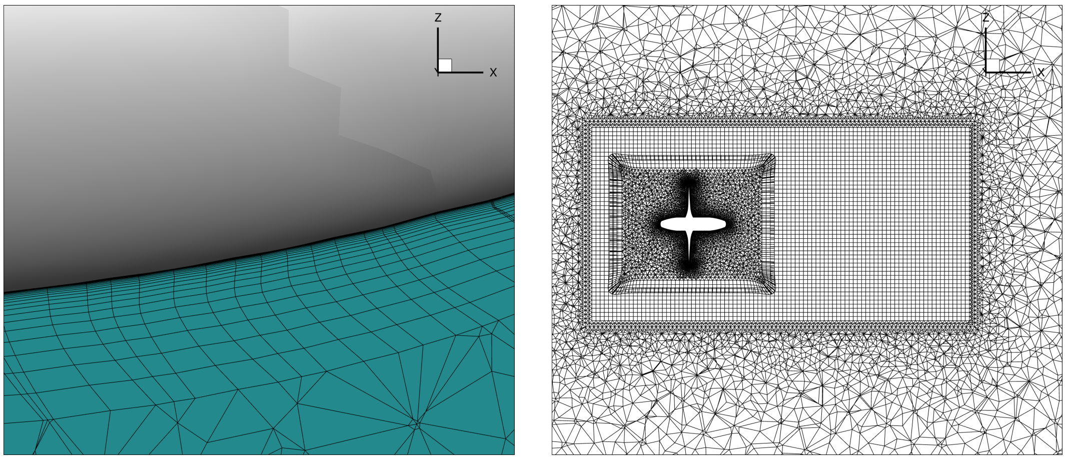


Figure 4.15: Details of the grid connection (right) and the boundary layer discretisation (left).

All the simulations are run on a dimensional standard with the following specifications:

J	$V\ [m/sec]$	$\omega\ [rad/sec]$	$n\ [rps]$	Re	$\nu\ [m^2/sec]$
0.4008	1.6890	165.4991	26.34	$5.5E+05$	$4.9135E-07$
0.5006	2.0840	163.4885	26.02	$5.5E+05$	$6.0625E-07$
0.5994	2.4800	162.4832	25.86	$5.5E+05$	$7.2145E-07$
0.6987	2.8150	158.2106	25.18	$5.5E+05$	$8.1891E-07$
0.8028	3.2010	156.5770	24.92	$5.5E+05$	$9.3120E-07$

The first validation property that was used in the visualisation part of the flow is the Q-criterion. This variable represents a measure for the vortex creation in a 3D flow problem.

The Q-criterion is defined as follows:

$$\nabla \mathbf{u} = \Omega\left(\mathbf{u}\right) + S\left(\mathbf{u}\right) \tag{4.18}$$

$$\Omega\left(\mathbf{u}\right) = \frac{1}{2}\left(\nabla \mathbf{u} - \nabla \mathbf{u}^T\right) \tag{4.19}$$

$$S\left(\mathbf{u}\right) = \frac{1}{2}\left(\nabla \mathbf{u} + \nabla \mathbf{u}^T\right) \tag{4.20}$$

$$Q\left(\mathbf{u}\right) = \frac{1}{2}\left(\left|\left|\Omega\left(\mathbf{u}\right)\right|\right|_2 - \left|\left|S\left(\mathbf{u}\right)\right|\right|_2\right) \tag{4.21}$$

$$\left|\left|A_{n\times n}\right|\right|_2 := \varrho\left(A^* A\right) = \max\left\{\lambda_1, \lambda_2, \dots, \lambda_n\right\}$$

$$\left(\lambda_1, \lambda_2, \dots, \lambda_n\right) \quad \rightarrow \quad det\left(A^* A - \lambda I_{3\times 3}\right) = 0 \tag{4.22}$$

In this definition, Ω and S refer to the rotation rate and the strain rate, respectively, whereas the Euclidean norm of a matrix $A_{n\times n}$ (the so-called spectral norm) is defined as the spectral radius of the conjugate matrix times itself, i.e. the maximum of the modulus of its eigenvalues (measured in the case that the eigenvalues are in the complex plane).

The Q-criterion is not the unique way to identify vortex cores in three-dimensional flows. Another identification standard could be the λ_2-criterion which checks whether at least two of the eigenvalues of the matrix $S\left(\mathbf{u}\right)^2 + \Omega\left(\mathbf{u}\right)^2$ are negative.

Taking this measure into account, we visualise the field after a large set of unsteady iterations in the following graph.

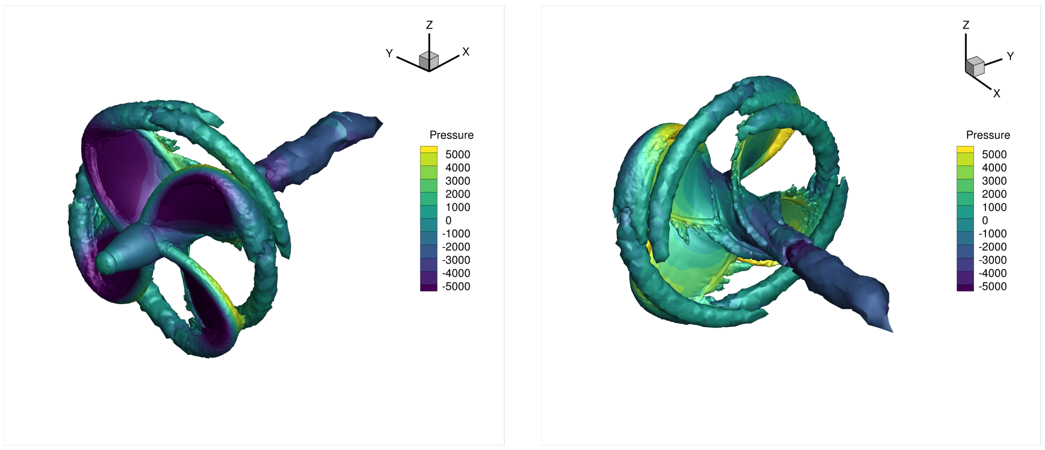


Figure 4.16: Iso-surface of the Q-criterion in values ±15000 along with projected contour of pressure with front view (left) and back view (right)

In these iso-surface graphs, it can be observed that the vortex creation happens at the tip of every blade and is moving away along a cylindrical path. In another note, it is clear to identify the low pressure zones of the blades (front side) and the high pressure zones (back side). As far as the values of Q are concerned, their magnitude implies that the vortex creation is very weak and the simulation converges into a steady flow configuration.

Additionally, in order to focus on the field communication between the two grids, we present in a contour slice the values of the x-component of the velocity.

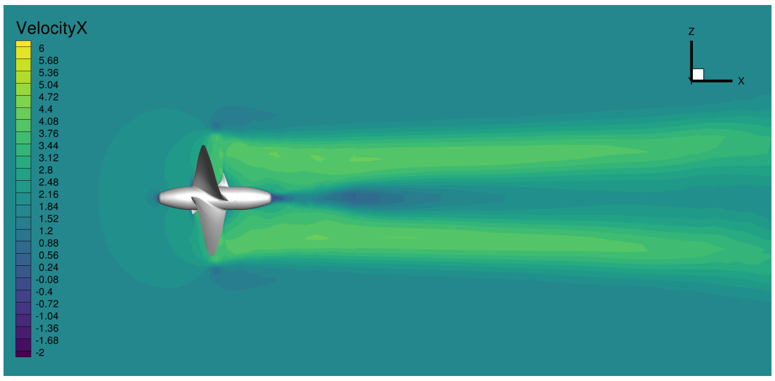


Figure 4.17: Contour plot on a $z-x$ section slice of the x-component of the velocity field

In this contour, it is without special notice that the flow accelerates behind the lifting body, indicating the creation of longitudinal thrust in the negative x-direction.

Thrust & Torque approximation After the evaluation of the flow characteristics visually, the comparison is made for global measures such as the total thrust and torque that are exerted by the rotating body.

The dimensionless measures for the propeller with which we build the thrust and torque[1] diagram are:

$$J = \frac{V}{nD} \tag{4.23}$$

$$k_T = \frac{T}{\rho n^2 D^4} \tag{4.24}$$

$$k_Q = \frac{Q}{\rho n^2 D^5} \tag{4.25}$$

$$\eta = \frac{J}{2\pi}\frac{k_T}{k_Q} \tag{4.26}$$

The simulations are ran in five values of the advance coefficient J in which we attempt to validate with the single grid (relative frame of reference) simulation and experiments. Both the simulations with the single grid and the experiments are conducted and assessed by Ntouras et. al. [34].

In the case that the simulations are run on a single grid with a steady-state configuration, convergence is achieved from some point in the iteration process, and therefore the body forces converge smoothly up to a tolerance level.

However, this is not the case in the unsteady regime, where even if the physical phenomenon has a very low characteristic strouhal number (pseudo-steady), when executing an unsteady simulation with a specific resolution, some

[1]The standard symbol for the mechanical torque in the majority of the bibliography concerning ship propulsion is Q, so it is not to be confused with the Q-criterion symbol, which will not be used furthermore in this thesis.

part of the energy cascade spectrum is resolved and the presence of eddies is inevitable. This dependency on the time and space resolutions can be easily shown in large-scale high-fidelity simulations (DNS or LES) at low angles of attack on airfoil sections where a vortex street is created near the trailing edge after a sufficient amount of time. The vortex creation visualised above depicts exactly this characteristic.

In order to find the part of the simulation that the longitudinal force is considered to have converged around a single value, we first observe and note the frequency of the force fluctuations and then create a chunk of time points where we have solution output (how many unsteady iterations from peak to bottom etc.). Then the mean value is calculated for every chunk. The criterion for stopping the iterations and measuring the solution values is a minimum tolerance level for the rate of change of the mean value function.

The comparison for the k_T, k_Q, η variables is shown below in the graph as well as tabulated.

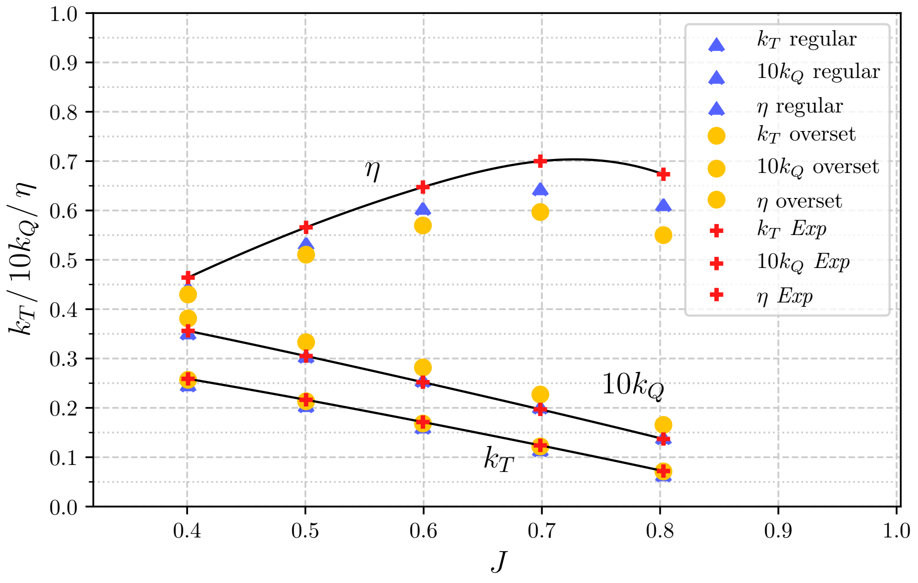


Figure 4.18: Comparison of the thrust, torque coefficients and efficiency grade

Comparison CFD-overset

J	δk_T	δk_Q	$\delta\eta$	δk_T %	δk_Q %	$\delta\eta$ %
0.4008	−0.0118	−0.0632	0.0464	−1.18	−6.32	4.64
0.5006	−0.0074	−0.0685	0.0561	−0.74	−6.85	5.61
0.5994	−0.0001	−0.0727	0.0686	−0.01	−7.27	6.86
0.6987	0.0043	−0.0844	0.0835	0.43	−8.44	8.35
0.8028	0.0037	−0.1252	0.1115	0.37	−12.52	11.15

Table 4.6: Comparison of the coefficients for the overset simulation vs the single grid CFD simulation normalised with the single grid CFD values

Comparison overset-experiments

J	δk_T	δk_Q	$\delta\eta$	δk_T %	δk_Q %	$\delta\eta$ %
0.4008	−0.0075	0.0716	−0.0738	−0.7507	7.16	−7.38
0.5006	−0.0138	0.0933	−0.0979	−1.3840	9.33	−9.79
0.5994	−0.0178	0.1163	−0.1201	−1.7830	11.63	−12.01
0.6987	−0.0169	0.1528	−0.1473	−1.6949	15.28	−14.73
0.8028	−0.0110	0.2108	−0.1832	−1.1022	21.08	−18.32

Table 4.7: Comparison of the coefficients for the overset simulation vs the experiment measurements normalised with the experiment values

As it can be observed, the method adequately approximates the total thrust that is exerted by the propeller, whereas there seems to be an over-estimation of the torque. This small overestimation affects the mechanical efficiency grade. Considering the load calculations, these are calculated by integrating over the solid wall boundary (using the mid-point rule) the pressure field and the viscous stresses plus gravitational forces (absent in our case). It is noted that the load calculation consists of integrating over the same field variables with the difference of the cross product with the position vector. This can cause a scaling in the approximation error and a more fine surface mesh could give more precise results.

$$\vec{F} = \int_{S_b} pI_{3\times3}\cdot\vec{n}\,\mathrm{d}S + \int_{S_b} \overset{\leftrightarrow}{\tau}\cdot\vec{n}\,\mathrm{d}S \quad \left(+ \int_{V_b} \rho_m \mathbf{g}\,\mathrm{d}V \right) \tag{4.27}$$

$$\vec{M} = \int_{S_b} \vec{r}\times\left(pI_{3\times3}\cdot\vec{n}\right)\,\mathrm{d}S + \int_{S_b} \vec{r}\times\left(\overset{\leftrightarrow}{\tau}\cdot\vec{n}\right)\,\mathrm{d}S \quad \left(+ \int_{V_b} \vec{r}\times\rho_m \mathbf{g}\,\mathrm{d}V \right) \tag{4.28}$$

Scalability

The last part that is addressed in this simulation is the scalability of the solver-library coupling.

However, it has to be pointed out that the scalability of this coupling has not been optimised and far more work has to be done to achieve a close to optimal result.

Both the solver and the library, have proven their skills for scalability by up to $10k$ processors. In this test, we simply document the results for their coupling with the intention of providing a global examination in our development.

In this scope, we provide the scale up factors and the elapsed time over the number of computing units.

The scaling factor is calculated with the first simulation running at 5 processors and not from 1 single processor. This is done due to a lack of sufficient storage space for such a large simulation.

Suppose we have two simulations with N_1 & N_2 number of processors and $N_2 > N_1$. The scaling factor is defined as:

$$Scale\ \% = \frac{\text{elapsed time}_{N_1}}{\text{elapsed time}_{N_2}} \bigg/ \frac{N_2}{N_1} \cdot 100 \tag{4.29}$$

Additionally, the speed-up percentage is calculated as:

$$speedup_i\ \% = \frac{t_0}{t_i} \cdot 100 \tag{4.30}$$

$$t_i : \quad \text{elapsed time for time step in simulation } i \tag{4.31}$$

The calculated elapsed times for all the simulations are gathered in the following table:

run	CPUs	Elapsed Time	
1	5	116.91	*sec*
2	10	64.21	*sec*
3	20	35.38	*sec*
4	40	17.79	*sec*
5	80	9.86	*sec*
6	200	4.79	*sec*
7	400	3.81	*sec*
8	600	3.42	*sec*

Table 4.8: Execution time per time step

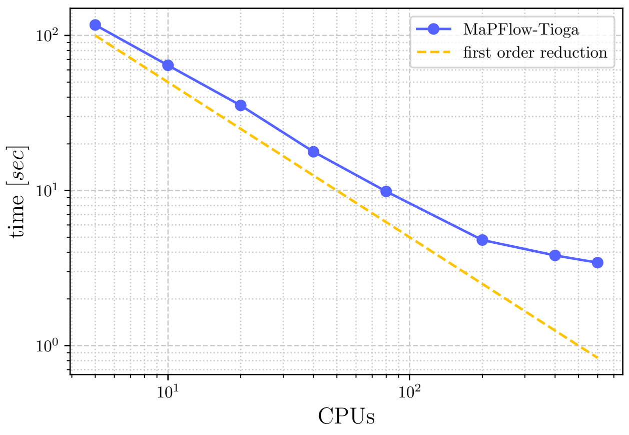


Figure 4.19: Elapsed time of execution over the number of CPUs on a logarithmic scale

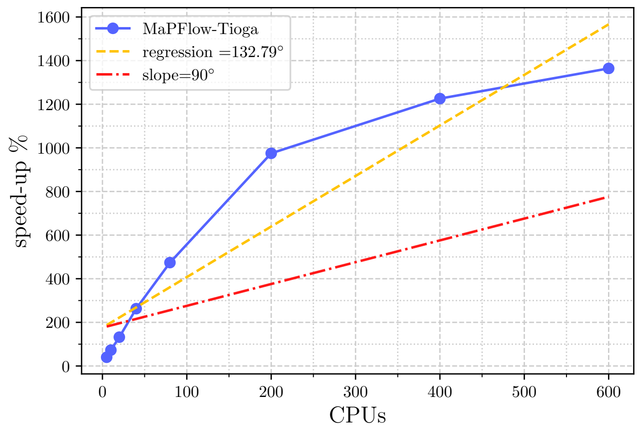


Figure 4.20: Total speed-up factor percentage along with its linear regression and a reference slope of 90°

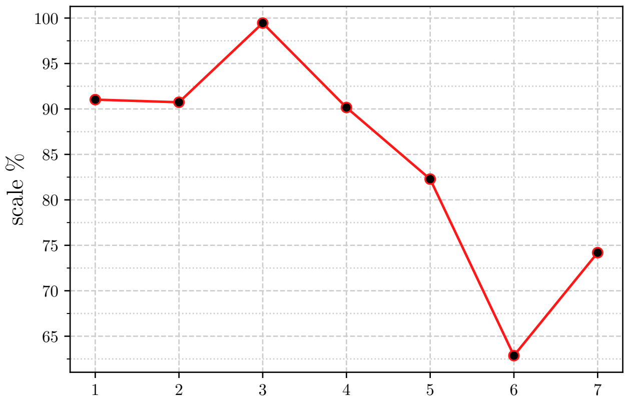


Figure 4.21: total scaling factor for every two executions in an ascending order of CPUs

As we observe in the graph 4.19 the elapsed time reduction is present for a significant number of processors. The biggest effect of the library-solver coupling that should be addressed concerns the graphs 4.20, 4.21 where a significant reduction in scalability can be observed. However, it has to be mentioned, at this point, that the elapsed time, along with the DoF (degrees of freedom) per CPU, is the actual indicator of the total efficiency at the point where we would wish that a time step has an elapsed time of execution at around $\mathcal{O}\left(10^0\right) - \mathcal{O}\left(10^1\right)$. This is the case for the current simulation at around 200 processes. Of course, the amount of DoF per CPU affects the total inter-CPU communication time and is considered a bottleneck when it is too small. It was observed in the whole process of the development that the optimal amount of DoF per CPU is close to $\mathcal{O}\left(10^4\right)$.

4.2.2 Heave decay of the floating sphere

Moving to the last validation case, we focus on the elementary physical phenomenon that involves sphere lateral dynamics on the free surface without forcing oscillations. This case is considered the most difficult among all the cases in this chapter. It is expected to test the limits of both the solver and the library, whether examining accuracy or computing efficiency.

This last numerical experiment was chosen mainly because it encompasses a medley of aspects such as: **a)** a pure unsteady 3D simulation **b)** presence of free surface **c)** rigid body dynamics. The second reason that the heave decay experiment is being studied is that there have been thorough examinations of the subject with plenty of experimental and simulation results. These results are documented for all experiments (both numerical and experimental) in [35] where various parties from different laboratories participate to produce results concerning body dynamics and radiation measurements.

The case setup reads as follows:

"Suppose there is a sphere of diameter $D = 300\ mm$ that its center of gravity is placed at $H_0\ mm$ from the undisturbed free surface of an experimental tank. The tank is adequately large (to avoid remaining reflections) with a constant depth of $d = 900\ mm$. At the time instant marked as $t_0 = 0\ sec$ the sphere is left free with zero initial vertical velocity and experiences the effects of all the hydrodynamic forces while freely oscillating."

The first variable that is observed and measured is the heave response signal, and the second is the free surface elevation in three different stations positioned at $x = 0$ (all of them) and $y = 600, 1200$ and $1800\ mm$.

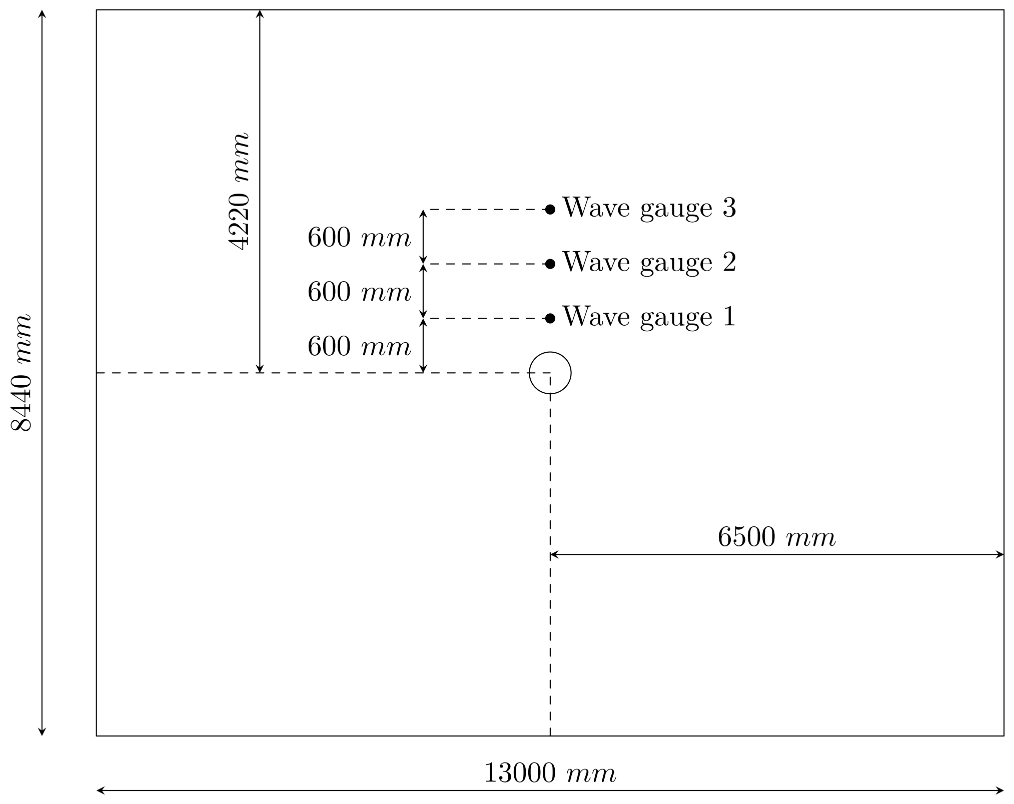


Figure 4.22: Schematic representation of the water tank for the numerical experiment

As shown in the schematic 4.22, the sphere is located at the centre of the tank with centre coordinates $(0, 0, 0)$.

The scope of the simulation case is to try and appropriately approximate two-phase flows near wall boundaries and the effects of the volume fraction near the boundary layer mesh. As far as the mesh resolution is concerned, for every starting position height of the sphere, we try to keep approximately $15-20$ cells within the wave height. Just because the case does not involve a propagation wave and is rather a radiation problem, we try to keep this number of cells compatible with the wave that has the dominant frequency and wave number. This rule will apply both in the z-direction and for the xy-surface mesh.

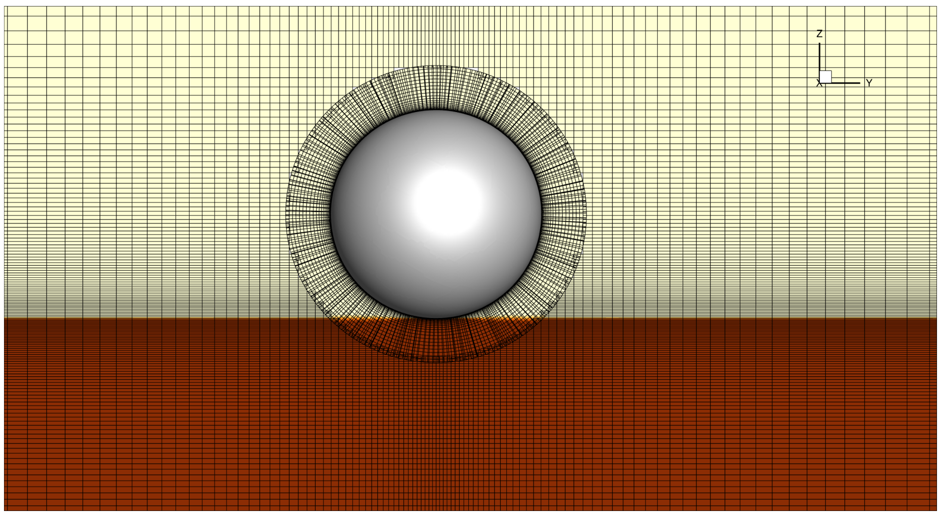


Figure 4.23: Contour slice with projected mesh on time $t = 0$ sec

4.2.3 Simulation setup

In order to simplify the setup for the rigid body dynamics, it is assumed that only one degree of freedom exists (out of six), which is an adequately accurate assumption for a perfectly azimuth-symmetric numerical experiment (however not in an experimental setup).

To perform the CFD-RBD [2] coupling, a total iteration number is first defined before the start of the simulation. From that point, each time-step is divided into sub-steps in which the CFD and the RBD solver exchange forces and CoG position and velocity, respectively.

The time integration for the RBD solver is performed by the Newmark-beta method with coefficients $\gamma = 0.5$ & $\beta = 0.25$.

$$M\ddot{x}_3 = F_{tot} \tag{4.32}$$

$$F_{tot} \text{ from } 4.27 \tag{4.33}$$

The mass M in equation 4.32 could be the experimental mass as it was measured in [35] experiments. However, further accuracy can be achieved while simultaneously eliminating a potential shift in the position of equilibrium ($z = 0$ mm), when using the numerical mass of the object, that is the

[2]RBD: Rigid Body Dynamics

total hydrostatic force from the fluid to the stationary ($H_0 = 0\ mm$) sphere divided by the measured acceleration of gravity.

In order to evaluate the simulation parameters after the mesh creation, we try to measure some flow characteristics by performing preliminary simulations. From these simulations, the dominant wave length and the maximum fluid velocities can be extracted. Taking into account the maximum values of the velocities and the smallest cell volume, the *Courant-Friedrich-Lewy* number is chosen, and therefore the appropriate time step:

$$CFL_{max} = \frac{u_{max} \cdot \Delta t}{\Delta x_{min}} \tag{4.34}$$

$$\Rightarrow \Delta t = \frac{CFL_{max} \cdot \Delta x_{min}}{u_{max}} \tag{4.35}$$

The flow is considered viscous in all cases, and the densities and viscosities are specified in [35] and more specifically in tables 1,2 and 4. We present the main features of the simulation input collectively.

Description	Properties	Values	Unit
diameter	D	300	mm
mass	m	7056	g
center of gravity	CoG	$(0, 0, -34.8)$	mm
acceleration of gravity	g	9.82	m/sec^2
starting altitude	H_0	$(30, 90, 150)$	mm
depth	d	900	mm
density of water	ρ_w	998.2	kg/m^3
density of air	ρ_a	1.2	kg/m^3
water kinematic viscosity	ν_w	$1.0 \cdot 10^{-6}$	m^2/sec
air kinematic viscosity	ν_a	$15.1 \cdot 10^{-6}$	m^2/sec

Table 4.9: Flow particulars

As far as the turbulence model is concerned, there have been some observations during the execution and development.

The *Reynolds* number was calculated using the solid sphere diameter and maximum velocity during heave decay, which is the velocity of the object when passing through the hydrostatic position for the first time. As for the viscosity, the kinematic viscosity of water is used.

Generally, when working with two-phase flow phenomena, special attention has to be given to the modelling of the existing turbulence. Classic eddy viscosity models such as the $k-\omega$ or the SST model (and their combination) [36], [37] makes use of two equations for the shear stress transport that involve the turbulent kinetic energy k and the dissipation rate ω. These models are mentioned briefly in Appendix A 5.2. When working with two-phase flows, such models tend to overproduce the turbulent kinetic energy near the free surface interface.

For that reason, modified turbulence models exist in which a source term is added to account for this specific strange effect. The most well known model introduced by B. Devolder *et. al.* [38] introduces the following term to the transport equation of the turbulent kinetic energy.

$$G_b = -\frac{\nu_t}{0.85}\frac{\partial \rho}{\partial x_i} g_i \tag{4.36}$$

In fact, generally in MaPFlow, every time that a simulation is set up involving two phases of fluids along with a turbulent regime, the above modification is always employed.

However, it was observed that, with the presence of a solid boundary and with extreme conditions such as free surface piercing, the problem of overproduction of turbulent kinetic energy is far from solved. This effect, in the case of $0.5D$ starting height, can cause instabilities during the iterative procedure of pseudo-time and the sudden increase in the local CFL number, that can (conditionally) crash the simulation.

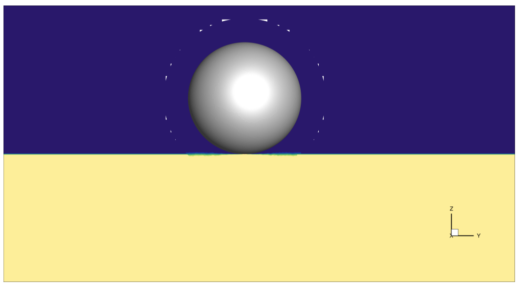
For this first reason, we try to execute the $0.5D$ simulation in the laminar case with a first cell height of about $y^+ = \mathcal{O}(1)$. This assumption, in the case of smaller starting heights ($0.1D$, $0.3D$) is adequately compatible in the results with the laminar hypothesis.

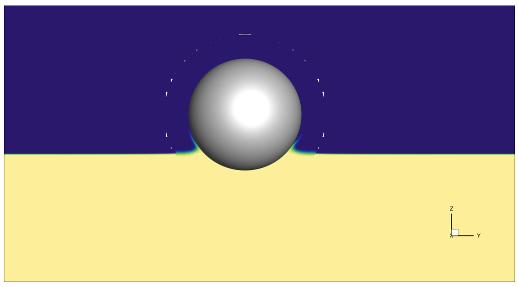
Another difficulty that is remedied during this family of simulations is the choice of the flux reconstruction scheme. Generally, as explained in chapter 2, two-phase flows with a high density difference and the presence of a gravitational field demand a less (numerically) diffusive reconstruction scheme for the volume fraction, mainly with the intention of constructing a more compressive free surface effect with a steeper density jump. These types of flux reconstruction schemes usually fluctuate between the upwind and the downwind state while taking into account the angle between the gradient of the normalised variable and the outward normal vector to the face. That way, accuracy is achieved in smoother regions away from the free surface, and a surface compression with a steeper volume fraction in the vicinity of the free surface is constructed.

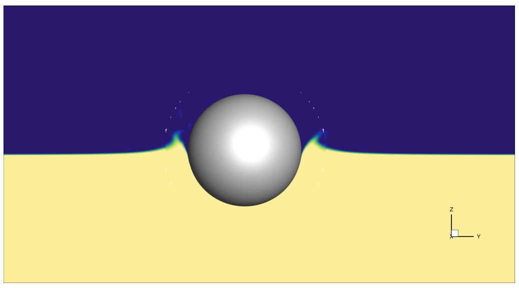
Some of these schemes are: the HRIC [39], STACS [40], MGDS [41], CICSAM [42] and BICS [43].

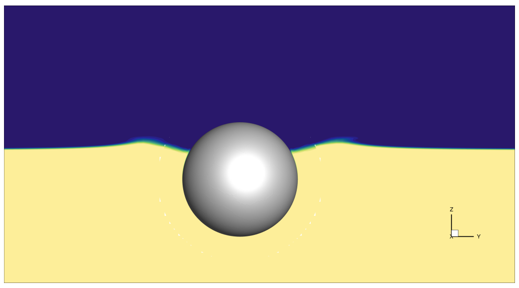
It has to be noted that, because there might be rapid changes in the interactions between the solid body and the free surface (such as the surface piercing effect), a more compressive scheme, which would work exceptionally well in the simple case of a wave propagating through a tank, might produce time-dependent discretisation restrictions. The scheme that was chosen for the simulations and proved to give eminent results with a relatively normal time step is the High Resolution Interface Capturing Scheme (HRIC).

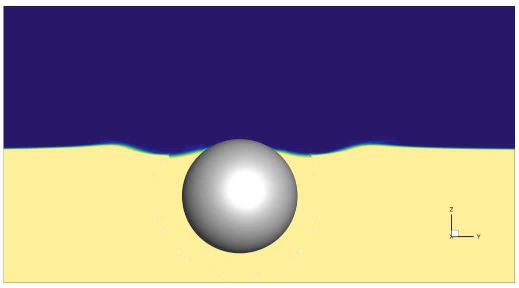
To document more effectively all the results between our simulations and the ones listed in [35], the comparison is presented, for every starting height H_0 separately for experiments, the CFD simulations and the non-linear BEM. Both signals (heave response and station free surface elevation) are compared in their time sequence and in their energy spectral density. Especially for the heave signals, the decaying cosine is mirrored on the negative time axis (horizontal axis) to create a wavelet type signal, which is then subject to a Fast Fourier Transform.

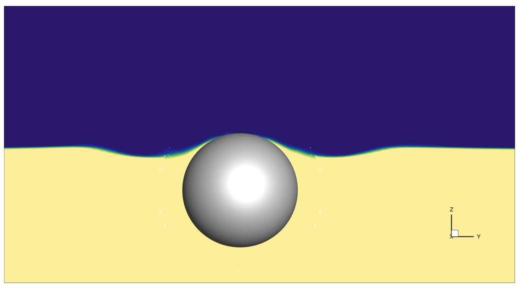

(a) $t = 0$ *sec*


(b) $t = 0.1$ *sec*


(a) $t = 0.2$ *sec*


(b) $t = 0.3$ *sec*


(a) $t = 0.4$ *sec*


(b) $t = 0.5$ *sec*

The above snapshots represent the contour slice of the density in various time instants for a starting height of $H_0 = 0.5D = 150\ mm$.

4.2.4 Starting height at: 0.1D

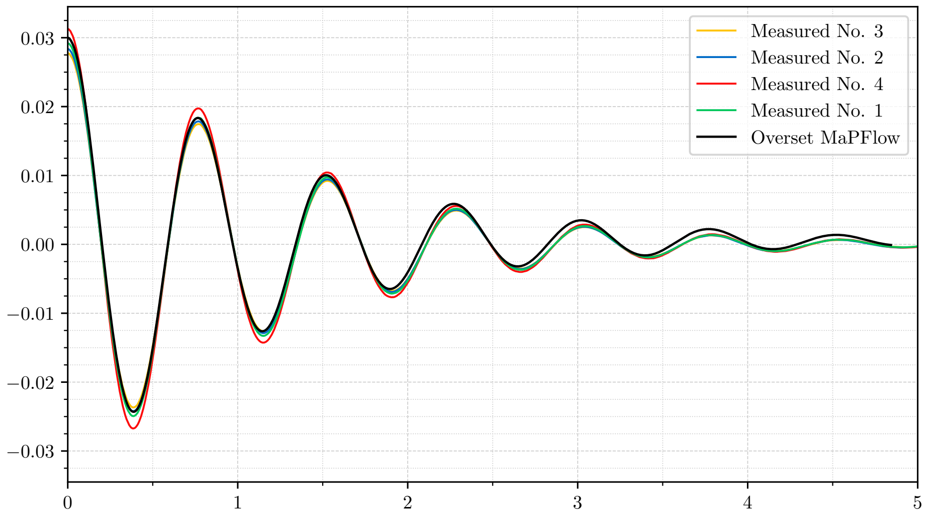


Figure 4.27: Heave decay compared with experiments at starting height $H_0 = 0.1D$.

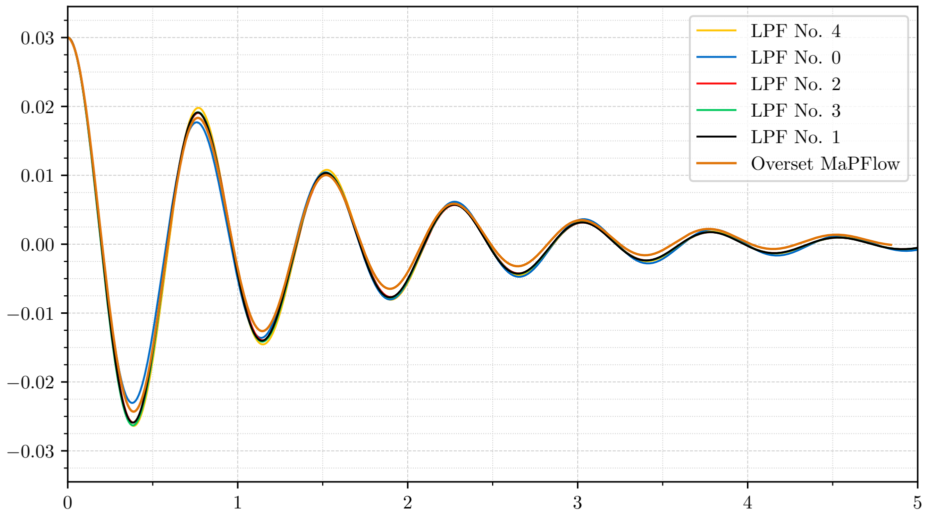


Figure 4.28: Heave decay compared with non-linear BEM solvers at starting height $H_0 = 0.1D$.

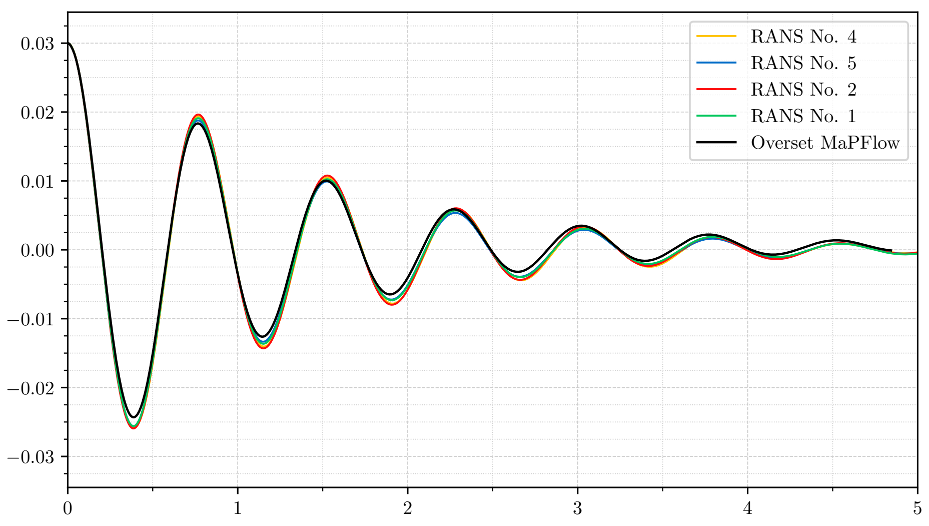


Figure 4.29: Heave decay compared with RANSE solvers at starting height $H_0 = 0.1D$.

4.2.5 Starting height at: 0.3D

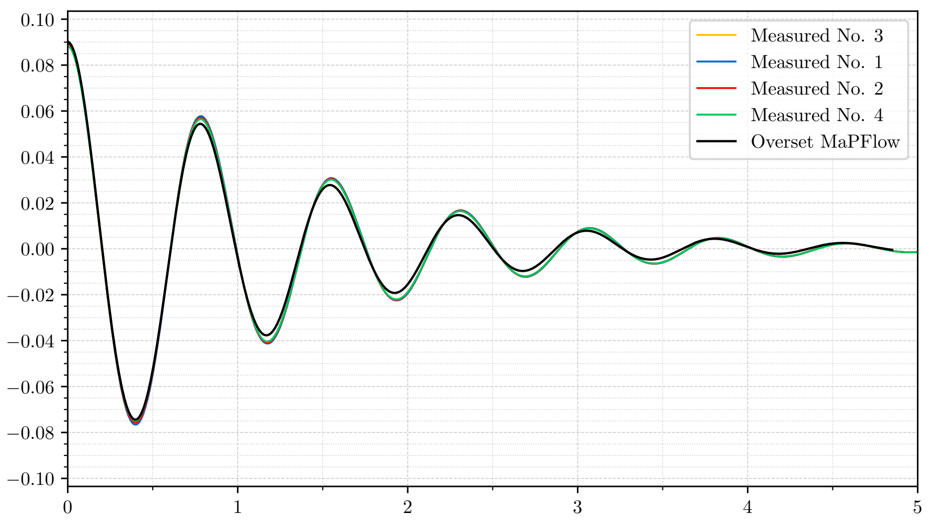


Figure 4.30: Heave decay compared with experiments at starting height $H_0 = 0.3D$.

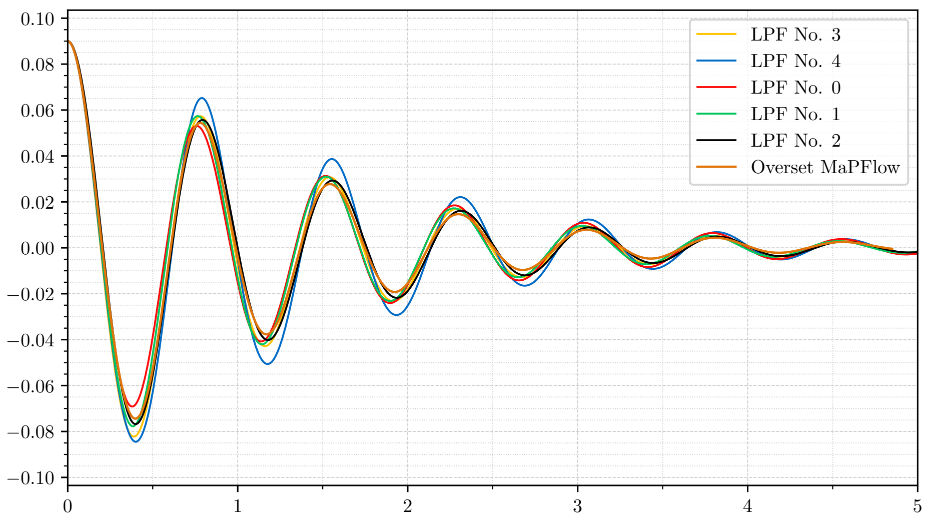


Figure 4.31: Heave decay compared with non-linear BEM solvers at starting height $H_0 = 0.3D$.

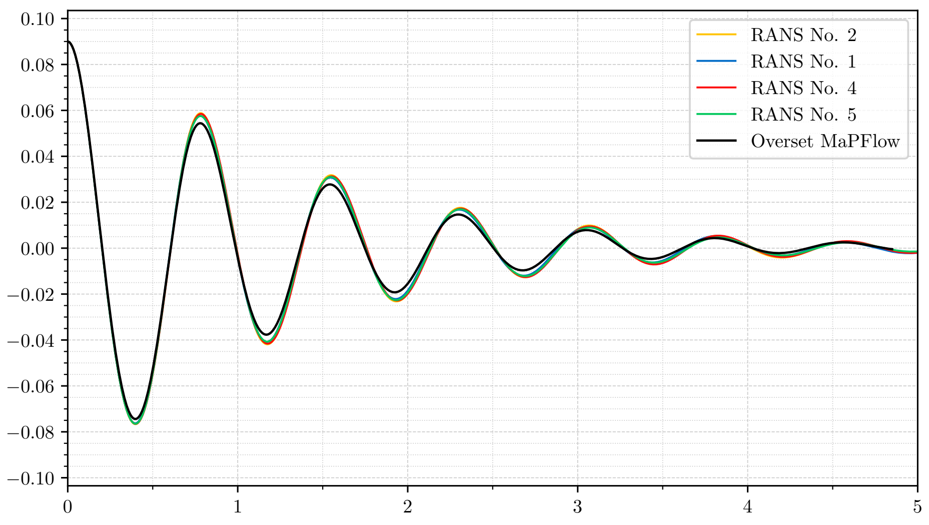


Figure 4.32: Heave decay compared with RANSE solvers at starting height $H_0 = 0.3D$.

4.2.6 Starting height at: 0.5D

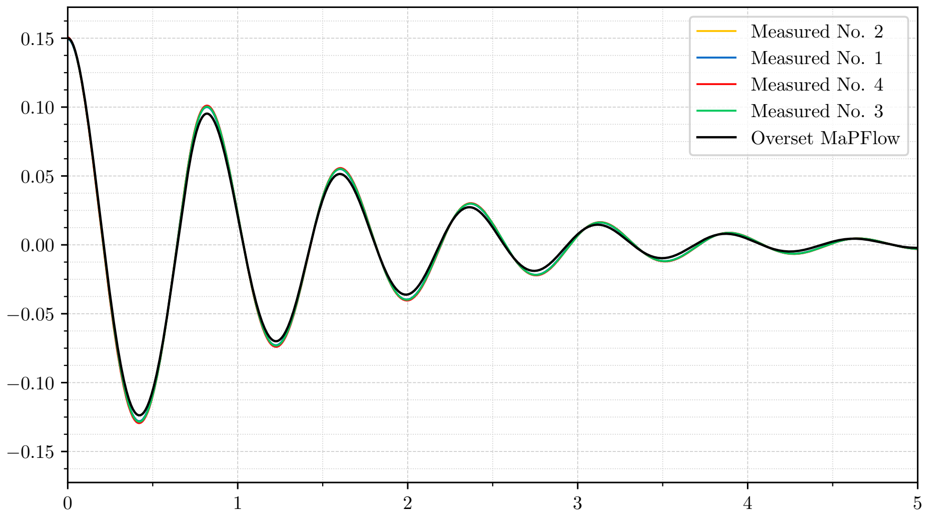


Figure 4.33: Heave decay compared with experiments at starting height $H_0 = 0.5D$.

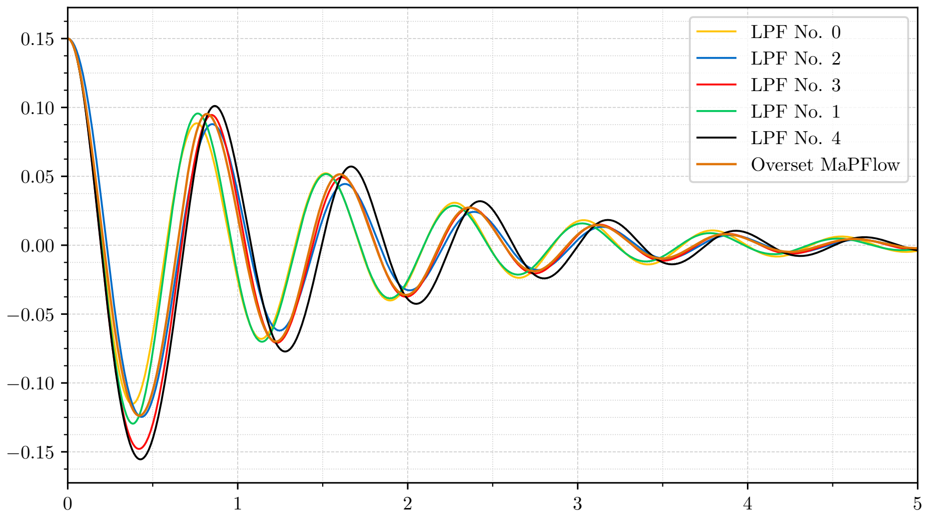


Figure 4.34: Heave decay compared with non-linear BEM solvers at starting height $H_0 = 0.5D$.

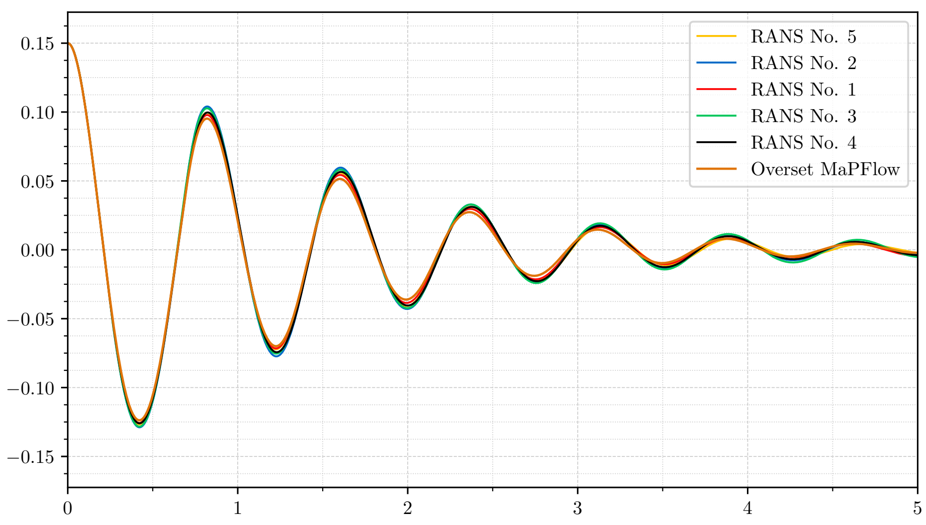


Figure 4.35: Heave decay compared with RANSE solvers at starting height $H_0 = 0.5D$.

On a general note, results deriving from the overset method and the experiments are in adequate agreement in all starting positions. In addition, results from non-linear BEM and pure mesh morphing RANSE simulations are also in agreement, especially in the case of $0.1D$ starting position. At higher starting points, differences in the heave decay amplitude and phase are worsened for the case of non-linear BEM because of effects such as wave-breaking and large dependencies on shear forces on the sphere (which are omitted in the irrotational inviscid hypothesis).

Along with the pure heave comparison, a cumulative graph is also presented for the normalised response over normalised time. The time is divided by the natural period of the free damping oscillation occurring when solving the linear hydrodynamic response in the frequency domain with constant hydrodynamic coefficients. The solution to this linear problem is a linear combination of harmonics multiplied by a damping exponent. The value of the natural damping period as investigated in [35] is calculated.

$$T_{e0} = 0.76 \; sec \tag{4.37}$$

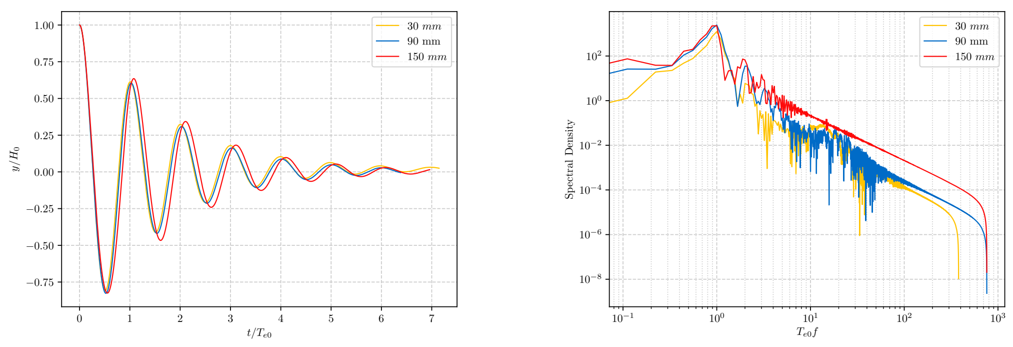


Figure 4.36: Comparison of the normalised heave position in the starting heights and dominant divided frequency.

It may be noted that, as far as the dominant normalised frequency is concerned, it has the expected value of unity, suggesting that the sphere is oscillating within very small tolerance levels of its natural frequency (no external force applied) as shown in the graph 4.36.

4.2.7 Free Surface Elevation

The free surface elevation results, which are measured in three different stations as shown in figure 4.22, are more dependent on the spatial and temporal discretisation, which is able to resolve only a limited spectra of wavelengths that characterise a dispersive gravity-induced radiating wave. For that matter, differences between signals from the gauges (in the numerical experiment) are amplified due to this effect.

However, the grid sensitivity criterion is not addressed in this part of the validation because of the absence of more specific data from the validating numerical experiments (such as cell first heights –from the undisturbed free surface– and cell lengths).

Lastly, the capturing of the free surface (for post processing and evaluation) is done by tracking the vertical coordinate where the volume fraction has a value of 0.5 (usually through interpolation).

4.2.8 Starting height at: 0.1D

In the graphs below, the free surface elevation is presented for three gauges and three starting heights (as the labels indicate). The comparison is done among all experimental and simulation data derived from RANS simulations. More precisely, on the left hand, the comparison is done between experimental data and whereas on the right hand, RANSE simulations are used for comparison.

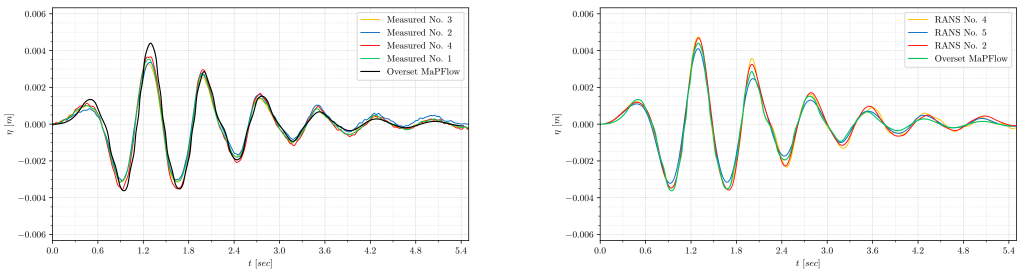


Figure 4.37: Free surface elevation time series of the **1st** station for starting height at $H_0 = 0.1D$.

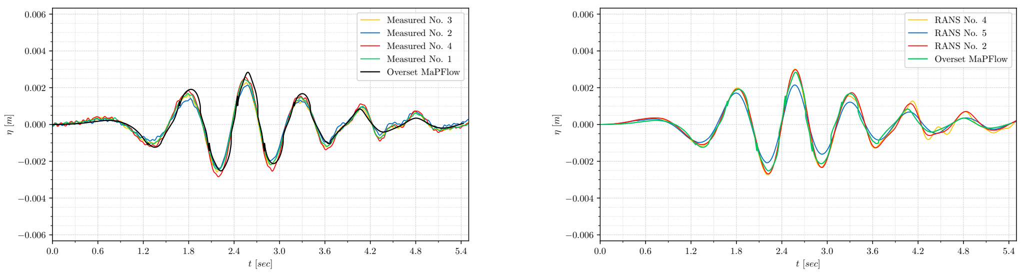


Figure 4.38: Free surface elevation time series of the **2nd** station for starting height at $H_0 = 0.1D$.

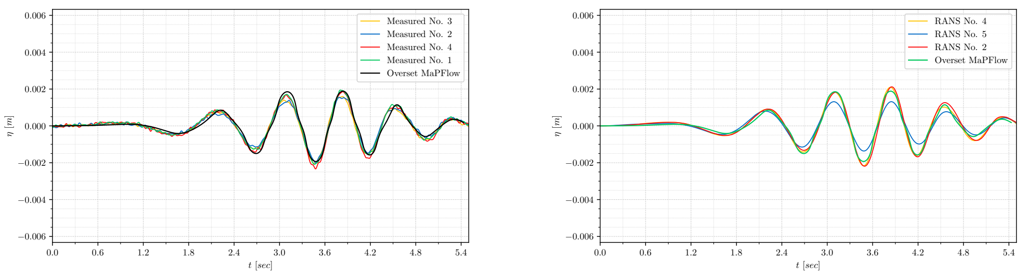


Figure 4.39: Free surface elevation time series of the **3rd** station for starting height at $H_0 = 0.1D$.

4.2.9 Starting height at: 0.3D

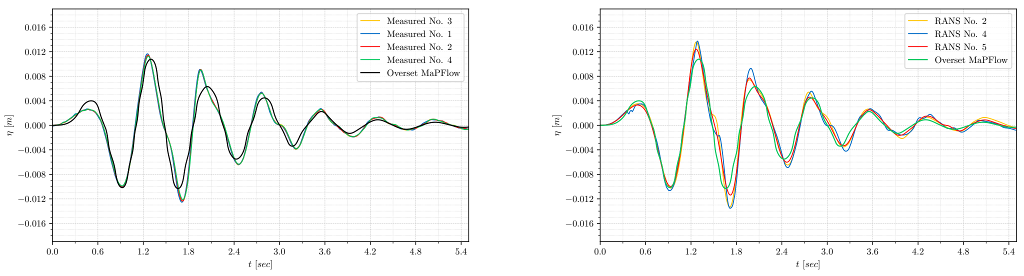


Figure 4.40: Free surface elevation time series of the **1st** station for starting height at $H_0 = 0.3D$.

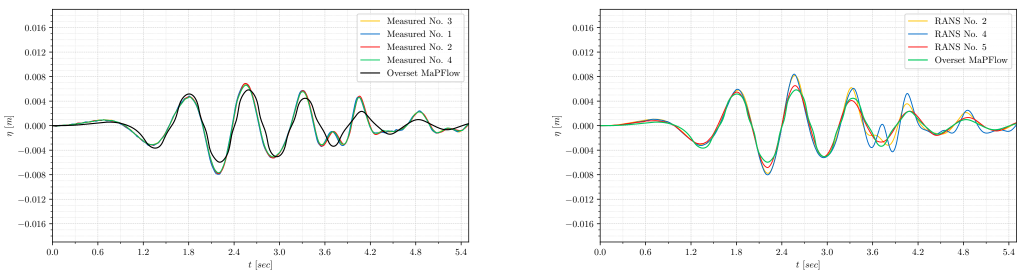


Figure 4.41: Free surface elevation time series of the **2nd** station for starting height at $H_0 = 0.3D$.

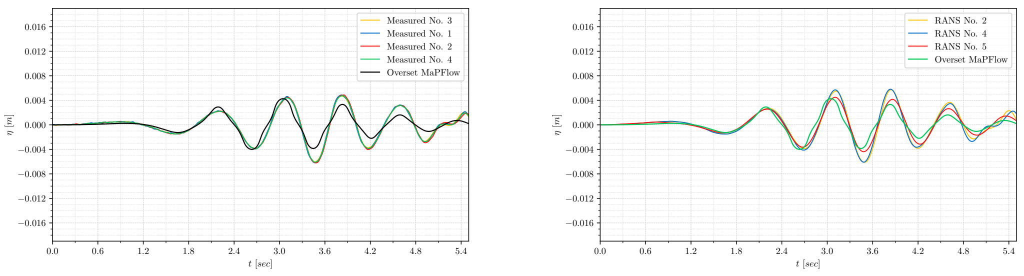


Figure 4.42: Free surface elevation time series of the **3rd** station for starting height at $H_0 = 0.3D$.

4.2.10 Starting height at: 0.5D

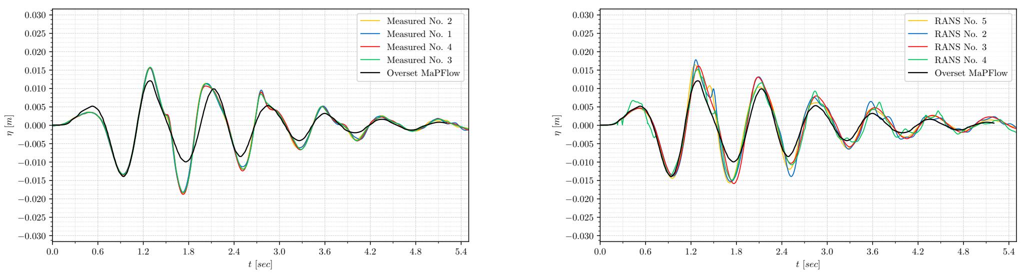


Figure 4.43: Free surface elevation time series of the **1st** station for starting height at $H_0 = 0.5D$.

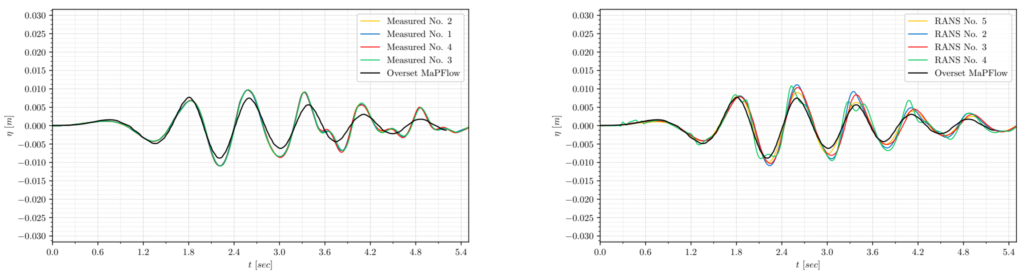


Figure 4.44: Free surface elevation time series of the **2nd** station for starting height at $H_0 = 0.5D$.

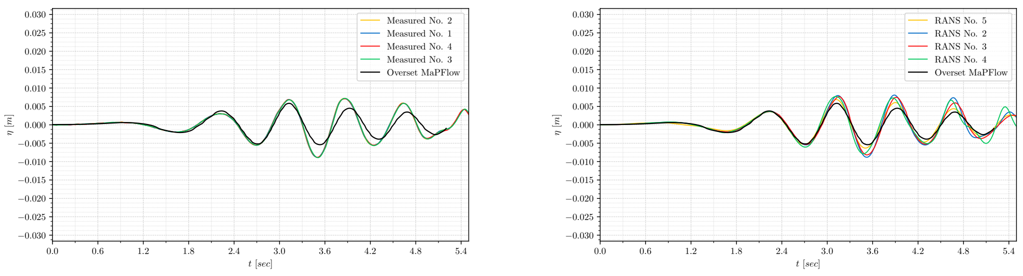


Figure 4.45: Free surface elevation time series of the **3rd** station for starting height at $H_0 = 0.5D$.

Chapter 5

Conclusion

5.1 Summary

A large portion of the work in this thesis was dedicated to both developing and understanding the underlying structure of a numerical methodology. One of the most desired aims for this work was to introduce a versatile solver-library coupling with the intention of understanding more complex fluid-structure phenomena. The structure of the resulting solver has great adaptivity in the case that the researcher wants to introduce a higher complexity factor. Moreover, *chimera* methods have advantages when compared with a famous competitor, namely the *immersed boundary methods*, and that is the explicit interpretation of a turbulence model in the vicinity of the boundary layer.

More precisely, immersed boundary methods usually introduce a source term with a mask function that fluctuates between two values when the cell is inside or outside the solid boundary. Apart from getting around the problem of complex moving geometries with great success, these methods lack the ability to reproduce near boundary shear stress fluidic interactions in different scales when applied in high-Reynolds number flows and a direct numerical simulation cannot be avoided.

Apart from the developed result, special treatment and focus had to be given to the field approximation through interpolation and the effects of special cases of two-phase fluids or gravity effects.

The robustness and accuracy of the solver were thoroughly tested in various cases, and a plethora of results and exported data validate these experiments with other numerical and experimental work provided in various publications.

5.2 Future Work Recommendation

As mentioned before, the current work aims at the flexibility of introducing complexity factors. By tackling the problem of an abstract movement of a body, researchers are faced with a more broad family of cases that describe physical phenomena. Some of these complexity factors that can be introduced and would be a valuable development for future work are the introduction of the body's elasticity. Though the coupling of CFD with structural finite element solvers seems straightforward (in the sense of algorithmic complexity), high-deformable objects with abstract motions could be formulated using deformable (morphing) meshes in the body-fitted grid along with the grid assembly structure for the communication of the field variables through the far field.

Other future work recommendations could easily involve hybrid solver coupling, with different levels of accuracy in each grid. More precisely, a useful interpretation in development would be the use of unstructured (triangulated) versatile grids for the body fitted meshes and high-order AMR[1] elements for the background mesh. This separation of mesh types could introduce high-fidelity solvers such as spectral elements, CWENO[2] reconstructions or nodal Discontinuous Petrov-Galerkin formulations with the intention of providing a low (numerically) diffusive representation of the fluid flow variables away from the vicinity of a solid boundary. By achieving this coupling, a researcher could easily study interactions of multiple bodies in vortical flows or vortex generation and vortex damping cases introduced by energy-saving devices.

[1]AMR: adaptive mesh refinement

[2]CWENO: central weighted non-oscillatory

Appendix A

Turbulence Modelling

A.1 Eddy Viscosity RANS Models

The most famous family of turbulence models that introduce a physical closure to phenomena with the presence of turbulence are the Eddy viscosity models. In this appendix, the two classic RANS models are briefly referenced. However, there have been many combinations of those and modified siblings which are widely used nowadays. Lastly, apart from RANS models, a wide range of applications require spatial filtering, introducing a filtering kernel for the sub-ranges, giving rise to the LES (large eddy simulations) models, which will not be mentioned in this thesis.

Generally, in the RANS family, a decomposition is introduced in the sense of a time-integrated mean value and a perturbed value:

$$u\left(\mathbf{x},t\right) = \overline{u}\left(\mathbf{x}\right) + u'\left(\mathbf{x},t\right) \tag{A.1}$$

The operator that produces the filtered function $\overline{u}\left(\mathbf{x}\right)$ is called a Reynolds operator and obeys the following rules:

$$\langle f+g\rangle = \langle f\rangle + \langle g\rangle \tag{A.2}$$
$$\langle af\rangle = a\langle f\rangle \tag{A.3}$$
$$\langle\langle f\rangle g\rangle = \langle f\rangle\langle g\rangle \tag{A.4}$$
$$\left\langle\frac{\partial f}{\partial t}\right\rangle = \frac{\partial\langle f\rangle}{\partial t} \tag{A.5}$$
$$\left\langle\frac{\partial f}{\partial x}\right\rangle = \frac{\partial\langle f\rangle}{\partial x} \tag{A.6}$$

Applying the Reynolds filtering on equations 2.15 and 2.7, reads:

$$\partial_j\overline{u}_i = 0 \tag{A.7}$$
$$\partial_t\overline{u}_i + \overline{u}_j\partial_j\overline{u}_i + u'_j\partial_i u'_i = \overline{f}_i - \frac{1}{\rho}\partial_i\overline{p} + \nu\partial_j^2\overline{u}_i \tag{A.8}$$

By eliminating the time-derivatives of a time-invariant mean velocity field and revisiting the definition of the strain-rate tensor S_{ij}:

$$\partial_j \overline{u}_i = 0 \tag{A.9}$$

$$\rho \overline{u}_j \partial_j \overline{u}_i = \rho \overline{f}_i + \partial_j \left(-\overline{p} \delta_{ij} + 2\mu \overline{S}_{ij} - \rho \overline{u_i' u_j'} \right) \tag{A.10}$$

The term $\overline{u_i' u_j'}$ which is known as the Reynolds stress divided by density, depicts every sub-grid to sub-grid interaction and is modelled by a transport equation. A classical RANS turbulence model will provide a transport equation for these Reynolds stresses. Additionally, the following definitions are important.

$$\epsilon = \nu \overline{\frac{\partial u_i'}{\partial x_k} \frac{\partial u_j'}{\partial x_k}} \qquad \text{Turbulent dissipation rate} \tag{A.11}$$

$$k = \frac{1}{2} \left(\overline{(u')^2} + \overline{(v')^2} + \overline{(w')^2} \right) \qquad \text{Turbulent kinetic energy} \tag{A.12}$$

$$\mu_t \qquad \text{dynamic eddy viscosity} \tag{A.13}$$

Some of these models are listed below.

A.1.1 K-epsilon Turbulence Model

The $k - \epsilon$ model introduces a transport equation for the specific dissipation rate and the turbulence kinetic energy.

$$\frac{\partial \rho k}{\partial t} + \frac{\partial \rho k u_i}{\partial x_i} = \frac{\partial}{\partial x_j} \left(\frac{\mu_t}{\sigma_k} \frac{\partial k}{\partial x_j} \right) + 2\mu_t E_{ij} E_{ij} - \rho \epsilon \tag{A.14}$$

$$\frac{\partial \rho \epsilon}{\partial t} + \frac{\partial \rho \epsilon u_i}{\partial x_i} = \frac{\partial}{\partial x_j} \left(\frac{\mu_t}{\epsilon} \frac{\partial \epsilon}{\partial x_j} \right) + C_{1\epsilon} \frac{\epsilon}{k} 2\mu_t E_{ij} E_{ij} - C_{2\epsilon} \rho \frac{\epsilon^2}{k} \tag{A.15}$$

$$\mu_t = \rho C_\mu \frac{k^2}{\epsilon} \tag{A.16}$$

The above equations contain constants that have been approximated through many different turbulent flow cases.

A.1.2 k-omega Turbulence Model

This model introduces the specific dissipation rate of turbulent kinetic energy to thermal energy ω. The transport equations read as follows:

$$\frac{\partial \rho k}{\partial t} + \frac{\partial \rho k u_i}{\partial x_i} = \rho P - \beta^* \rho \omega k + \frac{\partial}{\partial x_i} \left[\left(\mu + \sigma_k \frac{\rho k}{\omega} \right) \frac{\partial k}{\partial x_i} \right] \tag{A.17}$$

$$\frac{\partial \rho \omega}{\partial t} + \frac{\partial \rho \omega u_i}{\partial u_i} = \frac{\alpha \omega}{k} P - \beta^* \rho \omega^2 + \frac{\partial}{\partial x_i} \left[\left(\mu + \sigma_\omega \frac{\rho k}{\omega} \right) \frac{\partial \omega}{\partial x_i} \right] + \frac{\rho \sigma_d}{\omega} \frac{\partial k}{\partial x_i} \frac{\partial \omega}{\partial x_i} \tag{A.18}$$

$$P = \tau_{ij} \frac{\partial u_i}{\partial x_j} \tag{A.19}$$

Bibliography

[1] *Ευάγγελος Φίλιππας*. Hydrodynamic analysis of ship and marine biomimetic systems in waves using gpgpu programming. 2019.

[2] Michael Traut, Paul Gilbert, Conor Walsh, Alice Bows, Antonio Filippone, Peter Stansby, and Ruth Wood. Propulsive power contribution of a kite and a flettner rotor on selected shipping routes. *Applied Energy*, 113:362–372, 2014.

[3] Joseph L Steger and John A Benek. On the use of composite grid schemes in computational aerodynamics. *Computer Methods in Applied Mechanics and Engineering*, 64(1-3):301–320, 1987.

[4] Joseph L Steger, F Carroll Dougherty, and John A Benek. A chimera grid scheme. 1983.

[5] G Chesshire and William D Henshaw. Composite overlapping meshes for the solution of partial differential equations. *Journal of Computational Physics*, 90(1):1–64, 1990.

[6] Elwin Bruno Christoffel. Sul problema delle temperature stazionarie e la rappresentazione di una data superficie. *Annali di Matematica Pura ed Applicata (1867-1897)*, 1(1):89–103, 1867.

[7] Hidajet Hadzic. *Development and application of finite volume method for the computation of flows around moving bodies on unstructured, overlapping grids*. Technische Universität Hamburg, 2006.

[8] Ashesh Sharma, Shreyas Ananthan, Jayanarayanan Sitaraman, Stephen Thomas, and Michael A Sprague. Overset meshes for incompressible flows: On preserving accuracy of underlying discretizations. *Journal of Computational Physics*, 428:109987, 2021.

[9] Jacob A Crabill, Jayanarayanan Sitaraman, and Antony Jameson. A high-order overset method on moving and deforming grids. In *AIAA Modeling and Simulation Technologies Conference*, page 3225, 2016.

[10] Dominic DJ Chandar. On overset interpolation strategies and conservation on unstructured grids in openfoam. *Computer Physics Communications*, 239:72–83, 2019.

[11] George Papadakis and Spyros G Voutsinas. A strongly coupled eulerian lagrangian method verified in 2d external compressible flows. *Computers & Fluids*, 195:104325, 2019.

[12] Philippe Parmentier, Grégoire Winckelmans, and Philippe Chatelain. A vortex particle-mesh method for subsonic compressible flows. *Journal of Computational Physics*, 354:692–716, 2018.

[13] Georges-Henri Cottet, Petros D Koumoutsakos, et al. *Vortex methods: theory and practice*, volume 8. Cambridge university press Cambridge, 2000.

[14] IJ Schoenberg. Cardinal interpolation and spline functions. *Journal of Approximation theory*, 2(2):167–206, 1969.

[15] JJ805872 Monaghan. Extrapolating b splines for interpolation. *Journal of Computational Physics*, 60(2):253–262, 1985.

[16] Akihiko Takizawa, Seiichi Koshizuka, and Shunsuke Kondo. Generalization of physical component boundary fitted co-ordinate (pcbfc) method for the analysis of free-surface flow. *International Journal for Numerical Methods in Fluids*, 15(10):1213–1237, 1992.

[17] Cyril W Hirt and Billy D Nichols. Volume of fluid (vof) method for the dynamics of free boundaries. *Journal of computational physics*, 39(1):201–225, 1981.

[18] Mark Sussman, Peter Smereka, and Stanley Osher. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational physics*, 114(1):146–159, 1994.

[19] Francis H Harlow and J Eddie Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *The physics of fluids*, 8(12):2182–2189, 1965.

[20] Dimitris Ntouras and George Papadakis. A coupled artificial compressibility method for free surface flows. *Journal of Marine Science and Engineering*, 8(8):590, 2020.

[21] Suhas V Patankar and D Brian Spalding. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. In *Numerical prediction of flow, heat transfer, turbulence and combustion*, pages 54–73. Elsevier, 1983.

[22] Alexandre Joel Chorin. Numerical solution of the navier-stokes equations. *Mathematics of computation*, 22(104):745–762, 1968.

[23] Alexandre Joel Chorin. A numerical method for solving incompressible viscous flow problems. *Journal of computational physics*, 135(2):118–125, 1997.

[24] Philip L Roe. Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of computational physics*, 43(2):357–372, 1981.

[25] Robert F Kunz, David A Boger, David R Stinebring, Thomas S Chyczewski, Jules W Lindau, Howard J Gibeling, Sankaran Venkateswaran,

and TR0972 Govindan. A preconditioned navier–stokes method for two-phase flows with application to cavitation prediction. *Computers & fluids*, 29(8):849–875, 2000.

[26] Ralph Noack, David Boger, Robert Kunz, and Pablo Carrica. Suggar++: An improved general overset grid assembly capability. In *19th AIAA Computational Fluid Dynamics*, page 3992. 2009.

[27] Juan Alonso, Seonghyeon Hahn, Frank Ham, Marcus Herrmann, Gianluca Iaccarino, Georgi Kalitzin, Patrick LeGresley, Ken Mattsson, Gorazd Medic, Parviz Moin, et al. Chimps: A high-performance scalable module for multi-physics simulations. In *42nd AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, page 5274, 2006.

[28] Pieter Buning and Thomas Pulliam. Cartesian off-body grid adaption for viscous time-accurate flow simulations. In *20th AIAA computational fluid dynamics conference*, page 3693, 2011.

[29] Beatrice Roget and Jayanarayanan Sitaraman. Robust and efficient overset grid assembly for partitioned unstructured meshes. *Journal of Computational Physics*, 260:1–24, 2014.

[30] Christoffer Norberg. Effects of reynolds number and a low-intensity freestream turbulence on the flow around a circular cylinder. *Chalmers University, Goteborg, Sweden, Technological Publications*, 87(2):1–55, 1987.

[31] Charles HK Williamson. Defining a universal and continuous strouhal–reynolds number relationship for the laminar vortex shedding of a circular cylinder. *The Physics of fluids*, 31(10):2742–2744, 1988.

[32] CW Oseen. Uber die wirbelbewegung in einer reibenden flussigkeit. *Ark. Mat. Astro. Fys.*, 7, 1912.

[33] JD Fenton. The numerical solution of steady water wave problems. *Computers & Geosciences*, 14(3):357–368, 1988.

[34] D Ntouras, G Papadakis, D Liarokapis, G Trachanas, and G Tzabiras. Numerical and experimental investigation of a model scaled propeller. *Trends in Maritime Technology and Engineering Volume 1*, pages 409–415, 2022.

[35] Morten Bech Kramer, Jacob Andersen, Sarah Thomas, Flemming Buus Bendixen, Harry Bingham, Robert Read, Nikolaj Holk, Edward Ransley, Scott Brown, Yi-Hsiang Yu, et al. Highly accurate experimental heave decay tests with a floating sphere: A public benchmark dataset for model validation of fluid–structure interaction. *Energies*, 14(2):269, 2021.

[36] David C Wilcox. Formulation of the kw turbulence model revisited. *AIAA journal*, 46(11):2823–2838, 2008.

[37] Florian R Menter. Two-equation eddy-viscosity turbulence models for engineering applications. *AIAA journal*, 32(8):1598–1605, 1994.

[38] Brecht Devolder, Pieter Rauwoens, and Peter Troch. Application of a buoyancy-modified k-ω sst turbulence model to simulate wave run-up around a monopile subjected to regular waves using openfoam®. *Coastal Engineering*, 125:81–94, 2017.

[39] Samir Muzaferija. Computation of free surface flows using interface-tracking and interface-capturing methods. *Nonlinear water-wave interaction. Computational Mechanics, Southampton*, 1998.

[40] M Darwish and F Moukalled. Convective schemes for capturing interfaces of free-surface flows on unstructured grids. *Numerical heat transfer, part B: Fundamentals*, 49(1):19–42, 2006.

[41] Patrick Queutey and Michel Visonneau. An interface capturing method for free-surface hydrodynamic flows. *Computers & fluids*, 36(9):1481–1510, 2007.

[42] Onno Ubbink. Numerical prediction of two fluid systems with sharp interfaces. 1997.

[43] Jeroen Wackers, Barry Koren, Hoyte Christiaan Raven, A Van der Ploeg, AR Starke, GB Deng, Patrick Queutey, Michel Visonneau, Takanori Hino, and Kunihide Ohashi. Free-surface viscous flow solution methods for ship hydrodynamics. *Archives of Computational Methods in Engineering*, 18(1):1–41, 2011.