Specialist's degree
M.S.
The School of Engineering and Applied Science

A Resolved CFD-DEM Coupling Method for Simulation Two-Phase Fluids Interaction with Arbitrary Shaped Bodies.

by Anastasiia Sarmakeeva

B.S. in Applied Mathematics and Computer Science, July 2014, Udmurt State University

A Dissertation submitted to

The Faculty of
The School of Engineering and Applied Science
of The George Washington University
in partial satisfaction of the requirements
for the degree of Doctor of Philosophy

Januray 8, 2024

Dissertation directed by

Lorena A. Barba
Professor of Mechanical and Aerospace Engineering

The School of Engineering and Applied Science of The George Washington University certifies that Anastasiia Sarmakeeva has passed the Final Examination for the degree of Doctor of Philosophy as of December 18, 2023. This is the final and approved form of the dissertation.

A Resolved CFD-DEM Coupling Method for Simulation Two-Phase Fluids Interaction with Arbitrary Shaped Bodies.

Anastasiia Sarmakeeva

Dissertation Research Committee:

Lorena A. Barba, Professor of Mechanical and Aerospace Engineering, Dissertation Director

Elias Balaras, Professor of Engineering and Applied Science, Committee Member , **DEFINE CO-CHAIR TITLE**, Dissertation Co-Director

Kausik Sarkar, Professor of Mechanical and Aerospace Engineering, Committee Member

Thomas Lichtenegger, Department of Particulate Flow Modelling, Committee Member

Full Name, Title, Dissertation Director/Dissertation Co-Director/Committee Member

Name of External Examiner, Professorial Title, Name of External University (or Name, Job Title, Name of External Company), Committee Member

Dedication

Include a fancy quote or dedication.

Acknowledgments

For me, this journey has been longer and more sophisticated than anyone I know could have imagined. First, I'd like to thank my family for their support, especially my mom, dad, and brother. I would not be here without you.

I must also thank my advisor, Professor Lorena Barba. Her introduction to Almadena Chtchelkanova clarified that I should complete my work in the U.S. under her guidance. I am grateful for her support and guidance throughout my time at GW.

I am also grateful to Professors Thomas Lichtenegger, his support at PFM lab made compuatational part of my research more efficient and I learned a lot from our meetings.

I'd like to thank as well my lab mates: Paulina Rodriguez, Ting-Gyu Wang, Olivier Mesnard, Natalia Clementi, and Pi-Yueh Chang.

Special thanks go to researchers Leonid Tonkov and Alena Chernova, as well as Alexander Novikov. Their support and knowledge were invaluable when I began this journey.

Lastly, I want to thank my partner Lawrence Kirk and my Russian-speaking grad friends Anna Medvedeva, Maria Siulova, and Anna Webber. Your support made this journey significantly easier.

Abstract

A Resolved CFD-DEM Coupling Method for Simulation Two-Phase Fluids Interaction with Arbitrary Shaped Bodies.

This is the abstract. It contains some random text from the `lipsum` package. You may safely remove the `lipsum` package once you write your thesis.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

List of Symbols

Preface

This is the preface. It's another front matter page that offers additional detail into your work. Typically, only one (preface OR prologue OR foreword) is used. You can remove the other sections by deleting them inside `tex/frontmatter.tex` or using the appropriate show or hide commands.

Chapter 1: Introduction

1.1 Background

Each year, landslides cause more than 5000 deaths [**?**]. The frequency and severity of flooding continue to escalate due to the effects of drought and global warming, making landslides an ongoing concern. This work has a goal to provide a tool to better understand and make predictions of landslide process which would include granular media, water, and air. With the final goal is to make simulations of the process efficient and reproducible.

This work proposes a new method for simulation of the process of landslides falling into water, to better understand the physical involved mechanisms and get more accurate predictions of the resulting impact. The proposed method will focus on simulating the behavior of small particles, rocks, and boulders of arbitrary shpae as they interact with each other and with the water, using computational fluid dynamics techniques. By improving our understanding of this process, **we** hope to provide valuable insights that can inform risk management strategies and help reduce the impact of landslides on communities and infrastructure.

The occurrence of landslides inshore areas is a complex phenomenon.For accurate simulation in is common in literature rto use a fluid-structure interaction (FSI) [**?**] approach, wherein the fluid component is modeled using computational fluid dynamics (CFD) methods and the granular media component is modeled using solid mechanics methods. The challenge lies in combining these two components in a way that facilitates communication between them. The interdependence between the fluid and granular media makes the modeling process time-consuming and complex. This research aims to develop an approach for effectively integrating these two components to improve the accuracy and efficiency of landslide simulations. Specifically, the proposed method

will utilize advanced numerical techniques to accurately capture the fluid-granular media interaction and ensure that both components communicate seamlessly with each other. The outcome of this research could be applied for hazard assessment and risk management in areas prone to landslides. Although, a couple of works were published in recent years [**?**] [**?**], current work not only uses a different approach for computational simulations but also uses open-source code and libraries with a goal being reproducible [**?**]. Below I will provide the three aims of this work and the methods which will be used to achieve them:

Aim 1: Choose an efficient method to run simulations.

In order to select the most appropriate and efficient method for running simulations, several key aspects were considered:

1. **Coupling method:** Identify a suitable method for coupling the solid and fluid simulations, considering the characteristics of a landslide and granular media movement. This involved comparing different coupling techniques, such as one-way and two-way coupling, as well as investigation different levels of coupling, such as resolved and unresolved CFD-DEM approaches.

2. **Free surface reconstruction:** I need to evaluate various techniques for accurately capturing the free-surface in multiphase flow simulations. This include comparing different methods for free-surface reconstruction in terms of their accuracy, computational efficiency, and ease of implementation in the final solver.

3. **Stability and robustness:** Assess the stability and robustness of the chosen methods, especially when considering the influence of the coupling between solid and fluid simulations. This may involve analyzing the convergence properties and sensitivity to numerical parameters such as time step size and grid resolution.

4. **Scalability and parallelism:** Investigate the scalability and parallel performance of the chosen methods, ensuring that they can effectively utilize available

computational resources and accommodate large-scale simulations.

5. **Implementation complexity:** Consider the complexity of implementing the chosen methods within an existing simulation framework or developing a new one. This includes assessing the compatibility of the methods with existing software libraries and tools, as well as the effort required for code development and maintenance.

By carefully evaluating these factors, the most efficient method for running simulations can be chosen, enabling accurate and reliable results while minimizing computational costs.

Aim 2: Implement and run simulations using chosen method.

The implementation could be done incorporating third party libraries or modifying existing ones. After implementation, the model needs to be supported by a verification and validation procedure, which includes a parallel option for simulation to optimize performance. Once the implementation is complete, the results needed to be analyzed and interpreted. Stability of the solver is supported by grid convergence study as well. After that **we** tested the solver running multispherical bodies intercating with fluid and the last step was application to real-world problem to test its effectiveness and usefulness. It was experiment where granular mass falling into water as in works [**?**] or [**?**].

Aim 3: Make code developed slover efficient and reproducible.

The final aim was to create repro-pack following a tradion of reproducible research at Barba group [**?**]. **insert oliver check list and link to the repo**

The source code which is used for simulations and post processing scrips should easy accessible on GitHub [**?**] along with containers [**?**] to reproduce main experiments. The container encapsulate all dependencies, making possible for others to replicate the main experiments. Any initial conditions, boundary conditions, and material properties necessary for the simulations should be documented as well as detailed instructions

on how to run the simulations, including necessary command line arguments or configuration settings. Detailed explanations of the theoretical models used is provided in the next chapter, the implementation of code, and the interpretation of results provided in the chapter 3.

1.2 Overview and literature review

The signigicant amount of methods and approaches that were considered during the research process to have a better understanding of the research methodology and the rationale.

The field of landslide research is vast, with a lot of academic papers and numerous approaches proposed in recent years. The primary criterion for the simulation aspect of this study was to ensure the use of open-source code to support the requirement for reproducibility. As the landslide process involves a solid component interacting with fluid, it was necessary to choose the most suitable method for simulating both solid and fluid components with a free surface in an optimized and compatible manner. It is essential to consider the variety of free surface reconstruction methods available, as they differ in terms of numerical stability and the ability to capture the interface between fluids.

1.3 Methods for fluid simulation

Fluid simulation plays a critical role in numerous engineering and scientific fields, ranging from designing more efficient aircraft to predicting weather patterns. Various methods are available for fluid simulation, including Smoothed Particle Hydrodynamics (SPH) [**?**] [**?**], Finite Element Method (FEM) [**?**], Volume of Fluid (VOF) [**?**], and Lattice Boltzmann Method (LBM) [**?**]. Each method has its advantages and limitations.

Smoothed Particle Hydrodynamics (SPH) is a Lagrangian method that models fluid as a collection of particles, which interact with each other based on their relative

positions and velocities. SPH has proven effective in simulating fluid flows with complex geometries and free surfaces [**?**], such as ocean waves [**?**] and splashing water [**?**]. The method requires a large number of particles to accurately capture the fluid behavior, which can result in high computational costs and make it challenging to simulate larger-scale systems. Another limitation of SPH is that it relies on artificial viscosity to handle fluid viscosity, which can lead to inaccurate results, especially when the flow is complex [**?**]. Additionally, the artificial viscosity can cause unwanted damping effects on the fluid motion. Furthermore, SPH struggles with accurately modeling fluid-solid interactions [**?**], causing inaccuracy in boundary treatment. The placement of boundary particles, for instance, can affect the accuracy of the results, and it can be challenging to correctly represent complex geometries.

The Finite Element Method (FEM) is another fluid simulation approach. It is an Eulerian method that discretizes the fluid domain into finite elements [**?**], where fluid properties are represented by a set of equations. FEM can handle complex geometries and boundary conditions and has been widely used in simulating fluid-structure interactions. However, FEM requires a high level of mesh refinement to accurately capture the fluid behavior near solid boundaries, which can result in significant computational overheads [**?**]. Due to this reason, we did not consider this method in our work.

LBM [**?**] is a relatively new method that models fluid as a set of particles moving on a lattice. It is so called meshless method. LBM has demonstrated excellent performance in simulating fluid flows with complex geometries and boundary conditions and is particularly useful in modeling microfluidic systems [**?**]. However, LBM is not be the best choice since the focus of our application is on landslides and the way to simulate arbitrarily shaped bodies using LBM could be quite complicated.

Finally, VOF is an Eulerian method that tracks the fluid interface by solving a transport equation for each fluid phase with different density parameters. The

method conceptually marks cells between 0 and 1. To enable the method to handle complex geometries, free surfaces, and multiphase flows, various approaches have been suggested, such as the Marker and Cell method [**?**], Geometric Reconstruction [**?**], High-Resolution Interface Capturing [**?**], or Level Set [**?**]. We will discuss these methods more thoroughly in the following sections. VOF, in combination with free surface reconstruction methods, offers several advantages over other methods. These advantages include accurate tracking of the fluid interface, modeling surface tension effects, and maintaining computational efficiency.

After considering the advantages and limitations of each method and the specific requirements of our simulation, we have concluded that the VOF method is the most appropriate choice for our application. But we need to choose appropriate method for the free-surface reconstruction.

1.4 Method for free-surface representation

Free-surface simulation is a crucial technique in studying fluid dynamics and is widely used in many fields, such as aerospace, naval architecture, civil engineering and medical devices. There are several methods available for simulating free-surface flows, each with their own advantages and limitations. There are two general approaches when using the Finite Volume Formulation (or considering Eulerian formulation): interface-tracking methods and interface-capturing.

The first category includes the Front Tracking Method (FTM)[**?**] and Marker and Cell (MAC) Method [**?**]. While interface-tracking methods could be effective in some scenarios, they have crucial disadvantages compared to interface-capturing methods. One of the most important ones is the limited ability to handle topological changes such as merging, breaking, or the creation of new interfaces, which can be challenging. Such events often require advanced mesh manipulation techniques, which can be difficult to implement and may introduce instabilities. Although CFD is generally complex,

interface tracking methods often involve complex algorithms and data structures to explicitly track and manage the moving interface mesh or markers. This can lead to higher computational costs and may require more memory and processing time. Ensuring good mesh quality and resolution near the interface is critical for interface tracking methods. Maintaining an optimal mesh can be difficult, especially in cases of large deformations or high curvature. Due to the complexity of managing the interface mesh or markers, interface tracking methods can be more challenging to parallel and scale efficiently on high-performance computing systems.

In contrast, interface-capturing methods like algebraic [**?**] and geometric VOF [**?**] formulation, Level Set Method (LSM) [**?**], and a group of sharp interface methods [**?**] offer a more straightforward and flexible approach to handling complex topologies, with potentially lower computational costs. However, they may suffer from a smeared interface representation and require additional techniques to sharpen the interface and maintain mass conservation.

Some well-known geometric reconstruction methods include the Piecewise Linear Interface Calculation (PLIC) [**?**] method, which reconstructs the interface as a line in 2D or a plane in 3D within each interfacial cell. This is one of the most widely used methods for reconstructing interfaces in VOF simulations. The Height Function Method [**?**] reconstructs the interface by calculating the height of the fluid in each cell, which is then used to derive the interface normal and curvature. This technique is particularly effective in structured Cartesian meshes. Youngs' Method [**?**] represents the interface as a piece-wise linear surface within each cell, allowing for time-dependent multi-material flow simulations with large fluid distortions. The isoAdvector [**?**] method involves two main steps: exploiting an iso-surface concept for modeling the interface inside cells in a geometric surface reconstruction step and modeling the motion of the face-interface intersection line for a general polygonal face to obtain the time evolution within a time step of the submerged face area. As a plus, it has an open-source

implementation as a part of OpenFOAM [?].

These are just a few examples of the geometric reconstruction schemes used in the context of multiphase flow simulations. Each method has its advantages and disadvantages, and the choice of the most suitable method depends on factors like the problem's complexity, computational resources, and accuracy requirements.

Algebraic reconstruction schemes are used in interface capturing methods, particularly within the context of the Volume of Fluid (VOF) method. These methods compute the fluxes algebraically without the need for geometric reconstruction of the interface. Some well-known algebraic reconstruction methods include Compressive Schemes, which use the information of the interface's orientation (interface normal) with respect to the cell face to compute the face flux. One example of a compressive scheme is the High-Resolution Interface Capturing (HRIC) method [?]. Tangent of Hyperbola for Interface Capturing (THINC) [?] schemes utilize a hyperbolic tangent function to represent the phase indicator function. The volume fraction is then obtained using a numerical approximation, such as volume-averaged, polynomial, or hyperbolic tangent representation. The CICSAM (Compressive Interface Capturing Scheme for Arbitrary Meshes)[?] is an extension of the compressive VOF method for unstructured meshes. It uses the orientation of the interface normal to compute the face flux. Slope-Limiter-Based Methods [?] use slope limiters to control the steepness of the volume fraction gradient within the cells, which helps to avoid numerical oscillations and maintain the sharpness of the interface. Multidimensional Universal Limiter for Explicit Solution (MULES)[?] is a numerical scheme where the advection term is modified to compress the surface and reduce smearing. It helps in achieving a higher-order scheme for more accurate advection at the surface. These algebraic reconstruction schemes offer varying degrees of accuracy and computational efficiency. The choice of the most suitable method depends on factors like problem complexity, computational resources, and accuracy requirements.

The Level Set method is a popular numerical technique used for tracking interfaces and shapes in various applications, including two-phase flow simulations. It is widely applicable to a variety of problems, including fluid mechanics, computer graphics, and image processing. The method has its own set of advantages, such as accurate computation of interface properties. The method allows precise calculation of normals and curvature, which are crucial in many applications involving interfacial flows. Another benefit of the method is implicit interface representation, which makes it easier to handle complex topologies such as merging, breaking, and self-intersecting interfaces. The Level Set method can be easily extended to work with Adaptive Mesh Refinement (AMR), allowing for better resolution in areas of interest and reduced computational cost. On the other hand, there are several disadvantages that may make the method not the first choice. One of the biggest issues is that it does not inherently conserve mass for each phase, which is a critical requirement in numerical modeling of realistic two-phase flows. Although there are techniques to limit mass loss, the problem cannot be completely eliminated. Moreover, solving the Level Set equation can be computationally expensive, especially for large-scale problems or those requiring high accuracy, and the Level Set function needs to be reinitialized frequently to maintain a signed distance function, which can introduce additional computational overhead and may lead to a loss of accuracy. Additionally, coupling the Level Set method with other physics, such as fluid flow, can be challenging and may require the use of additional numerical techniques to address issues like mass conservation or surface tension.

Based on literature review the most general and accurate method is the geometric VOF, which in has implementation [**?**], which was chosen for free surface reconstruction. Because of the complexity of the method, it is necessary to describe it in more details which will be done in the next chapter.

1.5 Methods for granular media simulation

Several methods are available for simulating granular media, each with its own advantages and limitations. These methods include the Discrete Element Method (DEM)[**?**], Finite Element Method (FEM), Smoothed Particle Hydrodynamics (SPH)[**?**], Lattice Boltzmann Method (LBM) [**?**] [**?**], and Molecular Dynamics (MD).

DEM is a method that is widely used for studying the behavior of granular materials under different loading conditions. In this approach, granular media are modeled as a collection of discrete particles interacting with each other through contact forces. One of DEM's strengths is its ability to capture the details of individual particle interactions and contact forces, making it suitable for studying granular materials with complex geometries and boundary conditions.

In contrast, FEM is useful for simulating large-scale systems with solid bodies and complex geometries and boundary conditions. While FEM is capable of capturing the behavior of granular materials under different loading conditions, it is less accurate in capturing the details of individual particle interactions.

The Smoothed Particle Hydrodynamics Method could be used not only for fluid simulation, but also for prediction of behaviour of granular media as a fluid with individual particles representing the grains. This method is well suited for simulating the dynamics of granular flows and mixing processes. However, its accuracy in capturing the detailed contact forces between individual particles is limited, which restricts its applicability to certain types of granular materials.

Lattice Boltzmann Method falls into the same category as SPH group of methods. It is an efficient method for simulating large-scale systems with complex geometries and can accurately capture the fluid-like behavior of granular materials. However, it can be computationally expensive for high-resolution simulations, and its accuracy in capturing the detailed contact forces between individual particles is limited.

As well as we Molecular Dynamic Methods is useful for studying the behavior of granular materials at a microscopic scale and can capture the effects of thermal fluctuations and chemical reactions. But it is computationally expensive and may not be practical for simulating large-scale granular systems.

Taking into account the strengths and limitations of each method, we have chosen DEM with multi-spherical bodies, which called clumps in the literature, as our method for simulating granular media. DEM is able to capture the details of individual particle interactions and contact forces, and the addition of clump features allows for the modeling of cohesive forces between particles, which can be important in certain types of granular materials.

1.6 Coupling approach.

Coupling methods are commonly used in simulations where multiple physical phenomena are involved. There are different ways to couple simulations, and each method has its own advantages and disadvantages. Some common coupling methods include one-way coupling, two-way coupling, and fully coupled simulations.

In one-way coupling, one simulation provides input to another simulation, but the second simulation does not affect the first simulation. This method is useful when one simulation has a negligible effect on the other simulation.

In two-way coupling, both simulations interact with each other, and the results of each simulation affect the other. However, the interactions between the simulations are limited to specific points in time, and the simulations do not interact with each other at every time-step.

In fully coupled simulations, the two simulations are fully integrated with each other, and the results of both simulations are obtained simultaneously. The simulations interact with each other at every time-step, and each simulation takes into account the effect of the other simulation.

In our work, we have chosen to use two-way coupling. In this way method would allow us to simulate the interactions between different physical phenomena while being computationally efficient. We believe that this choice will provide necessary level of accuracy while keeping the computational costs reasonable.

To run this type of simulation code based on CFDEMcoupling [**?**] was chosen as well as their approach of two-way coupling was chosen as a base. There are couple of main reasons why **we** decided to use this approach:

- CFDEM is open source project with good support
- Project based on OpenFoam - open source C++ programming library. Freedom to choose type of discretisation, boundary conditions, methods to project solid body on Eulerian mesh and even what approach to use to model free surface.
- DEM part is open source, based on molecular dynamic package [**?**] with options of create clump as a body and possibility to implement force interaction models for granular media.

Verification and validation procedures of CFDEMcoupling [**?**] already done by [**?**] and [**?**] for one phase fluid interacting with solids. Next step is to implement multiphase and apply method for the real world problem.

1.7 Reproducibility

Despite in recent years there were couple of academic papers published with similar approach [**?**], [**?**], but different method for free surface simulation. The authors of published works does not share their source code, data from simulations and even experiments which used for verification are impossible to repeat because of lack of experiment parameters. That makes works unreproducible and could deprive readers to fully understand how the proposed approach work. One of the goals of current research were to make sure that application is preproducible. A container for the

solver, scripts, data, configuration files are avaliable on github and dockerhub (**links**) and Jupyter notebooks in the manuscript repository.!!!!

1.8 Thesis outline

The research project aims to simulate the behavior of fluids, granular media, and their interactions. To achieve this goal, we need to carefully choose the appropriate numerical methods for each aspect of the simulation.

For fluid simulation, we considered various approaches, such as SPH, FEM, VOF, and Lattice Boltzmann. After analyzing the pros and cons of each method, we concluded that geometric VOF method with isoAdvector approach for phase reconstruction is the best choice.

Next, for free surface simulation, we evaluated several techniques, including level-set, front-tracking, and a customized version of the LSM. After careful consideration, we decided to use the isoAdvector approach with some modifications to meet the specific needs of the simulation.

Finally, for granular media simulation, we looked at various methods, such as DEM, MPM, and lattice-based models. Based on the analysis of each method's strengths and weaknesses, DEM with multi-spherical body features looks the most appropriate.

In addition, we decided to use two-way coupling as the coupling method for simulation interaction of the solid part and the two-phase fluid. This method allows the most accurate modeling of the interactions between the media.

To validate and verify simulation results, we will implement and run simulations with the methods developed and described in the previous steps. This will include parallel options for simulation and will be supported with verification and validation procedures.

Ultimately, the results of the simulations will be applied to a real-world problem, demonstrating the practical applications of this research.

In conclusion, this research project represents a comprehensive analysis of the different methods for fluid simulation, free surface simulation, and granular media simulation, culminating in the development of a customized simulation model. The findings of this study have the potential not only to advance our understanding of the behavior of fluids and granular media, with real-world applications in fields such as engineering and geology but also provide reproducible results which could be improved and used by other researchers.

Chapter 2: Background and Methods

The type of the considered problem called as Fluid Structure Interaction (FSI) it is well known in CFD field, attacked researchers and scientists with different approaches. The main criteria and difficulty for the problem which we consider in current thesis is that we solving two-way coupling simulation [**?**]. That mean that simulations from fluid part affect simulation from solid part and vice versa and we need to consider the best way for communication between these two parts.

We will start with Navier-Stokes system of equation for incompressible fluid which consist of momentum and continuity equation:

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\nabla p + \nabla \cdot \boldsymbol{\mu} + \rho \mathbf{g}, \tag{2.1}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{2.2}$$

where $\mathbf{u}$ is velocity vector field, ρ is a fluid density, p - pressure, μ is a viscosity and $\mathbf{g}$ gravity vector.

There is no analytical solution available for Navier Stokes equation, in fact solution existence and uniqueness have not been proven till these days [**?**]. The most common and stable approach for fluid simulation is the finite volume method (FVM) [**?**], where we use Eulerian frame of reference to solve momentum and continuum equations as a Navier-Stokes system.

2.1 The finite volume method

To solve **??** we need to run discretisation of solution domain which is in turn give us a computational mesh on which governing equations are subsequently solved. The procedure is consist of two parts: discretisation of time and discretisation of space.

For the space discretisation we use Finite Volume Method (FVM), where the main feature is that domain is subdivided into Control Volumes (which is does not overlap and cover the whole domain).

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\rho \mathbf{u} \phi) = \nabla \cdot (D \nabla \phi) + \nabla S, \tag{2.3}$$

where D - is diffusivity of the ϕ and S is the source term. If we will write the equation in integral form as:

$$\int_{V_C} \frac{\partial \phi}{\partial t} dV + \int_{V_C} \nabla \cdot (\rho \mathbf{u} \phi) dV = \int_{V_C} \nabla \cdot (D \nabla \phi) dV + \int_{V_C} S dV, \tag{2.4}$$

where V_C is a fixed control volume C and S its flux over the boundaries. Then applying Gauss theorem to the convective and diffusive term we will get

$$\frac{\partial}{\partial t} \int_{V_C} \phi dV + \int_{S_C} \nabla \cdot (\mathbf{u} \phi) d\mathbf{S} = \int_{S_C} \nabla \cdot (D \nabla \phi) d\mathbf{S} + \int_{V_C} S dV, \tag{2.5}$$

where integral over a surface S_C of control volume C integral with normal directed outwards from control volume.

2.1.1 Governing equations and domain discretization

Using finite difference approach to propagate in time and replacing $\mathbf{u}$ by an approximation of the Taylor series around the cell centre.

Eqn. **??** is semi-discretized form of eqn. **??**, in order to solve it we need to approximate each term:

$$\frac{\partial}{\partial t} \int_{V_C} \phi dV = \frac{\rho^{n+1} \phi^{n+1} - \rho^n \phi^n}{\Delta t} V_C \tag{2.6}$$

for the forward Euler scheme:

$$\int_{S_C} \nabla \cdot \rho \mathbf{u}_f \phi d\mathbf{S} \approx \sum_f \rho \mathbf{u}_f \phi_f \cdot \mathbf{S}_f \tag{2.7}$$

where $\mathbf{S}_f$ faces of the control volume element C and $\mathbf{u}_f$ is interpolated value at the corresponding face. Similar procedure applied for diffusive term if we assume that D is a scalar

$$\int_{S_C} \nabla \cdot (D \nabla \phi) d\mathbf{S} \approx \sum_f D(\nabla_f \phi) \cdot d\mathbf{S}, \tag{2.8}$$

where $\nabla_f \phi$ is the face normal gradient at face S_C. The source term as a function of ϕ could be integrated over a the control volume as

$$\begin{aligned} S(\phi) &= S_p \phi + S_u \\ \textstyle\int_{V_C} S(\phi) dV &= S_p V_C \phi + S_u V_C \end{aligned} \tag{2.9}$$

where S_p and S_u are source terms.

2.1.2 Temporal discretization

In order to propagate in time $[t, t + \Delta)$

$$\int_t^{t+\Delta t} \left[\frac{\partial}{\partial t} \int_{V_C} \phi dV + \int_{S_C} \nabla \cdot (\rho \mathbf{u} \phi) d\mathbf{S} \right] dt = \int_t^{t+\Delta t} \left[\int_{S_C} \nabla \cdot (D \nabla \phi) d\mathbf{S} + \int_{V_C} S dV \right] dt \tag{2.10}$$

Assuming that control volume does not change in time we can rewrite **??** as a discretized form of transport equation **??**. Using an implicit Euler formulation and substituting disrcetized spatial terms we will get:

$$\frac{\rho^{n+1} \phi_C^{n+1} - \rho \phi_C^n}{\Delta t} V_C + \sum_f \mathbf{S}_f \cdot \rho \mathbf{u}_f \phi_f = \sum_f D_f \left(\mathbf{S}_f \nabla_f \phi_f^{n+1} \right) + S_p V_C \phi_C^{n+1} + S_u V_C \tag{2.11}$$

where ϕ_C^{n+1} is the value of ϕ at the centre of control volume C at time $t + \Delta t$, ϕ_C^n is

the value of ϕ at the centre of control volume C at time t, $\mathbf{S}_f$ is the area of face f, $\rho\mathbf{u}_f$ is the interpolated velocity at the face f, ϕ_f is the interpolated value of ϕ at the face f, D_f is the diffusivity at the face f, S_p and S_u are source terms.

Face values have to be obtained from the neighboring cells, which is why interpolation scheme is play important role in accuracy and stability of the solution.

2.1.3 Solving linear system of algebraic equations

There are different approaches to get neighboring cell-face values. It could be widely used upwind scheme or it could be higher order scheme.

If we write **??** in general form as:

$$a_c\phi_c^{n+1} + \sum_N a_N\phi_N^{n+1} = b_c \tag{2.12}$$

where N denotes of neighboring cells of c, this system could be written in a matrix form as:

$$[A][\phi] = \mathbf{B} \tag{2.13}$$

where $[A]$ is the matrix containing the coefficients a_c and a_N, $[\phi]$ is a flux values and $\mathbf{B}$ contains the source terms. a_c are the diaconal compinets of the matrix $[A]$ and a_N are the off-diagonal components.

2.1.4 Solution of the system

Because momentum and continuity equation does not have pressure relation **??**, there is couple of common ways to treat this issue [**?**]. The most common is SIMPLE, PISO and PIMPLE algorithms. In this work we use Pressure Immplicit Splitting Operator (PISO) [**?**] due to its stability to time advantage. Equation

$$a_P^{\mathrm{u}}\mathbf{u}_P + \sum_N a_N^{\mathrm{u}}\mathbf{u}_N = -\nabla p, \tag{2.14}$$

can be solved as a system of linear algebraic equations in two steps on a staggered according to PISO approach. Usuially the pressure is calculated using Rhie-Chow[**?**] interpolation scheme. It is done to avoid pressure-velocity decoupling. The process of pressure-velocity coupling is could be described in two steps:

Step 1: Prediction step, to find intermediate velocity:

$$(H + A)\mathrm{u} = -\nabla p^n \tag{2.15}$$

where all non-diagonal elements stored in matrix $\mathbf{H}(\mathbf{u}) = -\sum_N a_N^{\mathrm{u}} \mathbf{u}_N$ and diagonal elements stored in matrix A. Then we can rewrite **??** as

$$\begin{aligned} a_P^{\mathrm{u}} \mathbf{u}_P &= \mathbf{H}(\mathbf{u}) - \nabla p \\ \mathbf{u}_P &= \left(a_P^{\mathrm{u}}\right)^{-1} \left(\mathbf{H}(\mathbf{u}) - \nabla p\right) \end{aligned} \tag{2.16}$$

where P is an index arbitrary velocity node, N is an index from neighbouring cells.

Step 2: Correction step, to find pressure and velocity:

Compute flux on a cell faces using Rhie-Chow [**?**] correction

$$\phi = \frac{H(\mathbf{u})}{a_P^u} \cdot S_f \tag{2.17}$$

where S_f is area of a cell face. From **??** to satisfy continuity equation $\nabla \cdot \mathbf{u} = 0$

$$(a_P^{\mathbf{u}})^{-1} \cdot \nabla^2 p^* = \nabla \cdot \phi, \tag{2.18}$$

then correct the velocity field:

$$\mathbf{u}^* = (a_P^{\mathbf{u}})^{-1}(H(\mathbf{u}) - \nabla p^*) \tag{2.19}$$

where $\mathbf{u}^*$ and p^*,an intermediate velocity and a pressure. Then repeat step 2, untill

convergence condition is fulfulled and last step is writing down $\mathbf{u}^{n+1}$ and p^{n+1} for the next step $n+1$.

2.2 Volume of fluid method for multiphase flow

For two phase flow simulation we are using as a base of Volume of Fluid Method (VOF) [**?**]. The main idea of the method is to track the interface between two fluids by solving the continuity equation for the volume fraction of one of the fluids. The volume fraction is defined as the ratio of the volume of one fluid to the total volume of the two fluids. The volume fraction is defined as the ratio of the volume of one fluid to the total volume of the two fluids. The volume fraction is defined as the ratio of the volume of one fluid to the total volume of the two fluids. The volume fraction is defined as the ratio of the volume of one fluid to the total volume of the two fluids. The volume fraction is defined as the ratio of the volume of one fluid to the total volume of the two fluids. The volume fraction is defined as the ratio of the volume of one fluid to the total volume of the two fluids. The volume fraction is defined as the ratio of the volume of one fluid to the total volume of the two fluids. The volume fraction is defined as the ratio of the volume of one fluid to the total volume of the two fluids. The volume fraction is defined as the ratio of the volume of one fluid to the total volume of the two fluids.

$$\begin{gathered}\tfrac{\partial(\rho\mathbf{u})_i}{\partial t} + \nabla\cdot(\rho\mathbf{u}\mathbf{u})_i = -\nabla p_i + \nabla\cdot\boldsymbol{\tau}_i + \rho_i\mathbf{g} \\ \nabla\cdot\mathbf{u}_i = 0\end{gathered} \tag{2.20}$$

where i is the phase number. VOF relay on finite volume method [**?**]. For simulation of the free surface we add phase convection equation.

$$\frac{\partial\alpha}{\partial t} + \nabla\cdot(\alpha\boldsymbol{u}) = 0 \tag{2.21}$$

Where to differentiate fluid phases we can use scalar Heaviside step function to mark mesh cells as:

$$\alpha(t, \mathbf{x}) = \begin{cases} 1, & \mathbf{x} \in \Omega_1 \\ 0, & \mathbf{x} \in \Omega_2, \cup\Gamma \end{cases} \tag{2.22}$$

where Ω_1, Ω_2 different fluid phases and Γ is a surface. The surface Γ is a free surface between two fluids. On the interface boundary we need to satisfy equilibrium state:

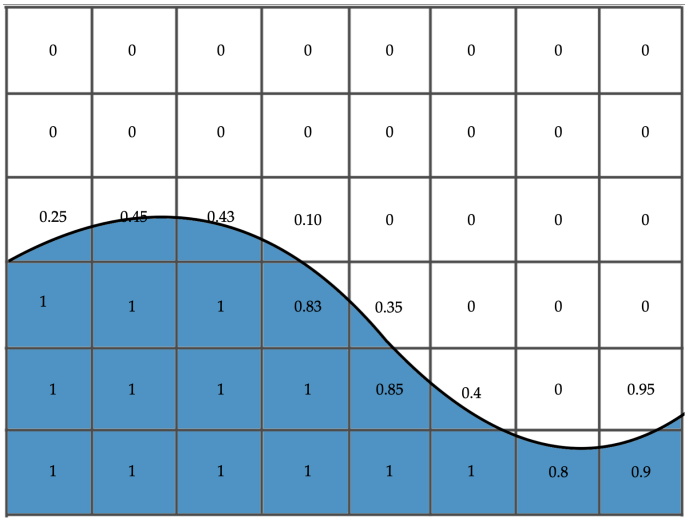


Figure 2.1: Schematic representation of volume of fluid mesh

$$\begin{gathered} (\boldsymbol{\tau}_1 - \boldsymbol{\tau}_2)\,\mathbf{e} = (p_1 - p_2 + \sigma K)\,\mathbf{e} \\ \mathbf{u}_1 = \mathbf{u}_2 \end{gathered} \tag{2.23}$$

where $\mathbf{e}$ - unit normal vector to Ω_1, K - surface curvature of Γ and σ is surface tension coefficient. Based on **??** density and pressure are:

$$\rho = \alpha\rho_{\text{water}} + (1-\alpha)\rho_{\text{air}}\; p = \alpha p_{\text{water}} + (1-\alpha)p_{\text{air}} \tag{2.24}$$

In the pure advection problem with a predetermined velocity field the specific values

of the fluid densities, ρ_A and ρ_B are immaterial, that is, the solution does not depend on them. To remove these insignificant parameters from the problem, we define the indicator field:

$$H(\mathbf{x}, t) \equiv \frac{\rho(\mathbf{x}, t) - \rho_B}{\rho_A - \rho_B} \tag{2.25}$$

where ρ_A and ρ_B are the densities of the two fluids.

2.3 Free surface reconstruction method

The method discussed herein is a sharp interface reconstruction technique, often referred to by various authors as the geometric Volume of Fluid (VOF) approach. In their work, Deen et al. [**?**] utilized Direct Numerical Simulations (DNS) to simulate multi-phase flow, employing a combination of the VOF method [**?**] and the Direct Forcing Immersed Boundary Method [**?**]. They employed a piece-wise linear interface method for free surface reconstruction, as in the approach presented by Brackbill [**?**].

A noted limitation of the VOF method, as discussed in [**?**], arises when the mutual distance between bubbles is less than the size of a computational cell. This condition, referred to as mutual coalescence, makes mass conservation challenging. Consequently, alternate methods such as the Multidimensional Universal Limiter with Explicit Solution (MULES) [**?**] and isoAdvector [**?**] have been considered. The study by Roenby et al. [**?**] provides a comprehensive comparison of these methods with proprietary solvers like ANSYS Fluent and STAR-CCM+, demonstrating their performance using a disk formation flow experiment. Roenby et al.'s work [**?**] gave promising results, with the source code available online. Owing to its superior performance and access to the source code, the free surface reconstruction approach from isoAdvector was tested and chosen.

The algorithm for isoAdvector,() as described in detail by Roenby et al. [**?**], consists of the following steps:

1. **Initialization:** The initial fluid distribution is represented by a field variable such as a color function or a volume fraction, which is set to 1 in cells filled with fluid and 0 in cells filled with air (or another fluid).

2. **Interface reconstruction:** The fluid-air interface is reconstructed in each cell based on the volume fraction. This is typically done using a geometric approach, which represents the interface as a planar surface within each cell

3. **Advection:** The interface is advected according to the local fluid velocity. This is done by sweeping a control volume through the cell in the friction of the velocity vector and computing the change in the volume fraction

4. **Correction:** After the advection step, the volume fractions are corrected to ensure mass conservation. This is done by adjusting the volume fraction in cells where the interface has been advected in or out.

5. **Update:** The new volume fractions are used to update fluid distribution for the next time step.

For isoAdvector the amount of each phase calculated as:

$$\alpha_i = \frac{1}{V_i} \int_{\Omega_i} H(\boldsymbol{x}, t) dV, \tag{2.26}$$

where V_i is the volume of cell i and Ω_i surface??? represents each cell to calculate the phase fractions at the next time step. The volume fraction of each phase is then updated as:

$$\alpha_i(t + \Delta t) = \alpha_i(t) - \frac{1}{V_i} \sum_{j \in B_i} s_{ij} \int_t^{t+\Delta t} \int_{F_j} H(\boldsymbol{x}, \tau) \boldsymbol{u}(\boldsymbol{x}, \tau) d\boldsymbol{S} d\tau \tag{2.27}$$

Here, the flux of α across each cell face is time-integrated and then summed. The symbol B_i denotes the set of all faces F_j associated with cell i. The factor s_{ij} is

employed to direct the flux outward from the cell. τ is the integration variable used over the time step, and dS is the differential area vector pointing out of the volume.

The factor s_{ij} can either be $+1$ or -1. Its purpose is to ensure that the product $s_{ij}dS$ always points outward from the cell boundary, even when the orientation of face j causes dS to point into the cell.

The right side integrals could be replaced with $\Delta V_j(t, \Delta t)$, which represents the total volume of fluid transported across face j during one time step in given by:

$$\Delta V_j(t, \Delta t) = \int_t^{t+\Delta t} \int_{F_j} H(\boldsymbol{x}, \tau)\boldsymbol{u}(\boldsymbol{x}, \tau)d\boldsymbol{S}d\tau \tag{2.28}$$

This allows us to experss the next step as:

$$\alpha_i(t + \Delta t) = \alpha_i(t) - \frac{1}{V_i} \sum_{j \in B_i} s_{ij}\Delta V_j(t, \Delta t) \tag{2.29}$$

In the context of fluid equations of motion treated by the finite volume method, the velocity field is naturally represented by cell-averaged values:

$$\mathbf{u}_i(t) \equiv \frac{1}{V_i} \int_{\mathcal{C}_i} \mathbf{u}(\mathbf{x}, t)dV \tag{2.30}$$

Given that the convective terms in the governing fluid equations denote the transport of properties like mass and momentum across cell faces, another crucial representation of the velocity field are the volumetric fluxes across mesh faces:

$$\phi_j(t) \equiv \int_{\mathcal{F}_j} \mathbf{u}(\mathbf{x}, t) \cdot d\mathbf{S} \tag{2.31}$$

where ϕ_j is the face flux across the face j.

2.3.1 The advection step

Many interface flows solver in the Navier-Stokes equations using a segregated solution approach, wherein the governing flow equations are solved sequentially for each time step. As a result, while advancing the interface from time t to $t + \Delta t$, we only have access to the velocity field information up to time t. However, to compute the updated αi as illustrated in equation **??**, it's necessary to understand the velocity field within the interval $[t,\ t + \Delta t]$. This implies that we must estimate how the velocity field evolves during this time step.

The most straightforward approach is to presume that the velocity field remains constant throughout the entire time step. This assumption simplifies the expression in equation **??** to $u(x, \tau) \approx u(x, t)$. Furthermore, in equation **??**, we posit that the dot product of u with the differential face normal vector, dS, on face F_j can be approximated in terms of the volumetric face flux, ϕ_j **??** as:

$$\mathbf{u}(\mathbf{x}, t) \cdot d\mathbf{S} \approx \frac{\phi_j(t)}{|\mathbf{S}_j|} dS \text{ for } \mathbf{x} \in \mathcal{F}_j \tag{2.32}$$

if we set face normal as in [**?**]:

$$\mathbf{S}_j \equiv \int_{\mathcal{F}_j} d\mathbf{S} \tag{2.33}$$

and substitute in into **??**, we will get:

$$\Delta V_j(t, \Delta t) \approx \frac{\phi_j(t)}{|\mathbf{S}_j|} \int_t^{t+\Delta t} \int_{\mathcal{F}_j} H(\mathbf{x}, \tau) dS d\tau \tag{2.34}$$

while the the rest of the **??** is the area of the face j submerged in one fluid, which we call $A_j(\tau)$:

$$A_j(\tau) \equiv \int_{\mathcal{F}_j} H(\mathbf{x}, \tau) dS = \int_{\mathcal{F}_j \cap \mathcal{A}(\tau)} dS \tag{2.35}$$

Replace $A_j(\tau)$ into **??** we will get:

$$\Delta V_j(t, \Delta t) \approx \frac{\phi_j(t)}{|\mathbf{S}_j|} \int_t^{t+\Delta t} A_j(\tau) d\tau \tag{2.36}$$

The transport of one fluid across a face is computed only once per face. For internal faces, this calculated value is utilized to update the volume fractions for both cells adjoining the face. This process ensures both local and global conservation of the two fluids, as discussed in [**?**].

2.4 Immersed boundary method

First the terminology was used in [**?**] to study blood flow through the heart. The method has since been applied to a variety of problems in fluid dynamics, solid mechanics, and biological systems. In current method solid body cover several computational cells. There are two primary approaches to handling fluid-structure interaction problems: body conformal mesh methods and fixed mesh methods. Body conformal mesh methods offer high accuracy but come with increased complexity in terms of implementation and computational cost and not sutable for simulation of granular media process. On the other hand, fixed mesh methods discretize the entire computational domain at once, with both fluid phases and solid sharing a single velocity and pressure field. This leads to more efficient calculations, making the fixed mesh approach the method of choice for this thesis. Figure **??** provides a schematic representation of a solid body within a 2D fluid domain, both before and after discretization.

2.4.1 Void fraction calculation

To account solid body on Eulerian mesh in CFDEMcoupling [**?**] used *smooth representation* [**?**] algorithm where the each sphere of multisperical body divided into a

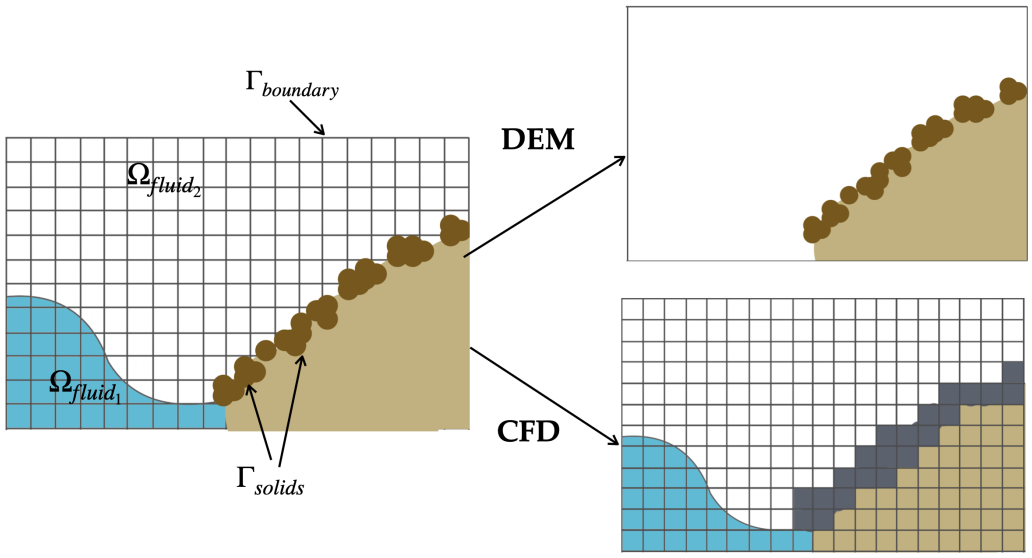


Figure 2.2: Particle in the fluid domain (left), representation of the particle in DEM (top) and CFD (bottom). Ω_{fluid_1} is the water domain, Ω_{fluid_2} the air phase, $\Omega = \Omega_{fluid}$. Γ_{solid} is the interface between fluid and solid.

core and corona and similar procedure done for each particle **??**.

The *voidFraction* β_i model is used to calculate clump forces and correct the fluid velocity field. The main challenge is that particle dynamics are calculated in a Lagrangian coordinate system, while fluid dynamics use an Eulerian coordinate system. To account for the solid body on the Eulerian mesh, the CFDEMcoupling employs a *smooth representation* algorithm [**?**], where each sphere of a multispherical body is divided into a core and a corona as shown in Figure **??**. A similar procedure is done for each particle. All cells detected within the particle's area have their relative positions with respect to the particle checked. If they are located inside the core, their volume fraction is set to one. For cells covered by the corona, a loop over all vertices belonging to the cell begins. If the vertex is inside the particle, the cell's volume fraction is increased by one-eighth (since each cell has eight vertices in the given case). If the vertex lies outside the corona, the intersecting point of the particle hull and the connection between the cell center and edge is computed. The relative length of the

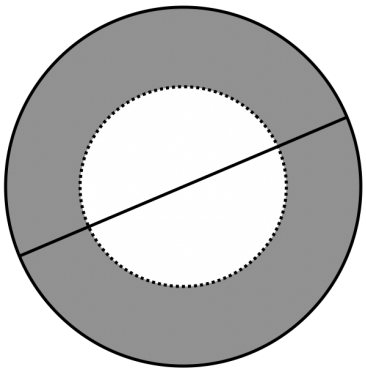


Figure 2.3: Schematic representation of the particle divided into core (white) and corona (gray).

line between the cell center and intersection, multiplied by one-eighth, is then added to the cell's current volume fraction. This method's verification has been made in the work by Kloss et al. [**?**].

First we need to assign a weight for every finite volume cell

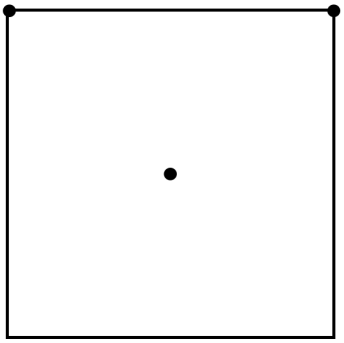
$$\beta_i = \frac{N_{vc,i} + N_{cc,i} N_{v,i}}{2 N_{v,i}} \tag{2.37}$$

where $N_{vc,i}$ and $N_{cc,i}$ are the number of vertices and centers intersecting the immersed body, respectively. $N_{v,i}$ one of the vertices, for haxahedral cell $i = 8$. The weight is used to calculate the velocity of the immersed body at the cell center. The velocity of the immersed body at cell i's center is calculated as:

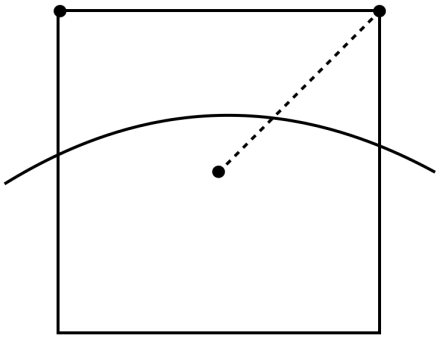
$$\boldsymbol{u}_{ib,i} = \boldsymbol{v}_{ib} + \frac{1}{N_{vc,\mathrm{i}} + N_{v,\mathrm{i}}} \left[\left(\sum_{j}^{N_{v,i}} \omega \times (\boldsymbol{x}_{v,j} - \boldsymbol{x}_{ib}) \right) + N_{v,i} \boldsymbol{\omega} \times (\boldsymbol{x}_{c,\mathrm{i}} - \boldsymbol{x}_{ib}) \right] \tag{2.38}$$

The velocity of cell i, denoted as $\boldsymbol{u}_{ib,i}$, comprises the translational velocity $\boldsymbol{v}_{ib}$ and the rotational velocity around the center of rotation $\boldsymbol{x}_{ib}$. The coordinates of the cell's center and vertices are represented by $\boldsymbol{x}_{c,\mathrm{i}}$ and $\boldsymbol{x}_{v,j}$, respectively. It is important to note that the angular component of the immersed body's velocity at cell i's position

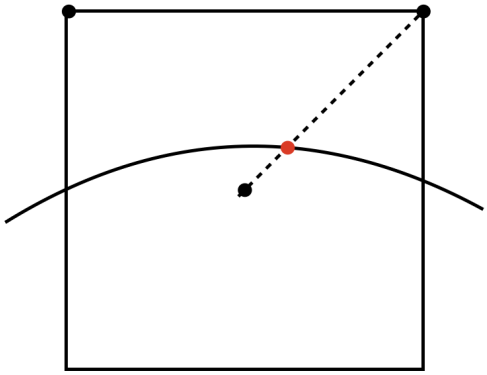
cannot be determined analytically due to the stair-casing effect. The expression within the brackets represents this angular velocity, which is not precisely known. However, for a fully covered cell, it is equivalent to the angular velocity at the cell center. Using



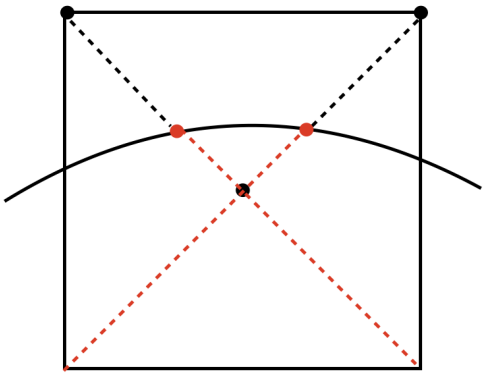
(a) Vertex is outside the particle



(b) Connecting a line between the cell center and vertex



(c) Intersecting point of the particle hull and the connection between the cell center and edge is computed



(d) Summing up the contributions of all vertices

Figure 2.4: Schematic representation of defining *voidFraction* process.

this methodology, the volume of the projected immersed body may not exactly match the volume enclosed by its surface mesh. Consequently, a halo layer is introduced to adjust the volume of the discretized immersed body. This layer corresponds to cells i where the body fraction β_i falls within the range of $]0, 1[$. Its size is modified to ensure

the volume remains unaffected by cell alignment. The entire process is summarized in the block diagram depicted on the figure **??** below.

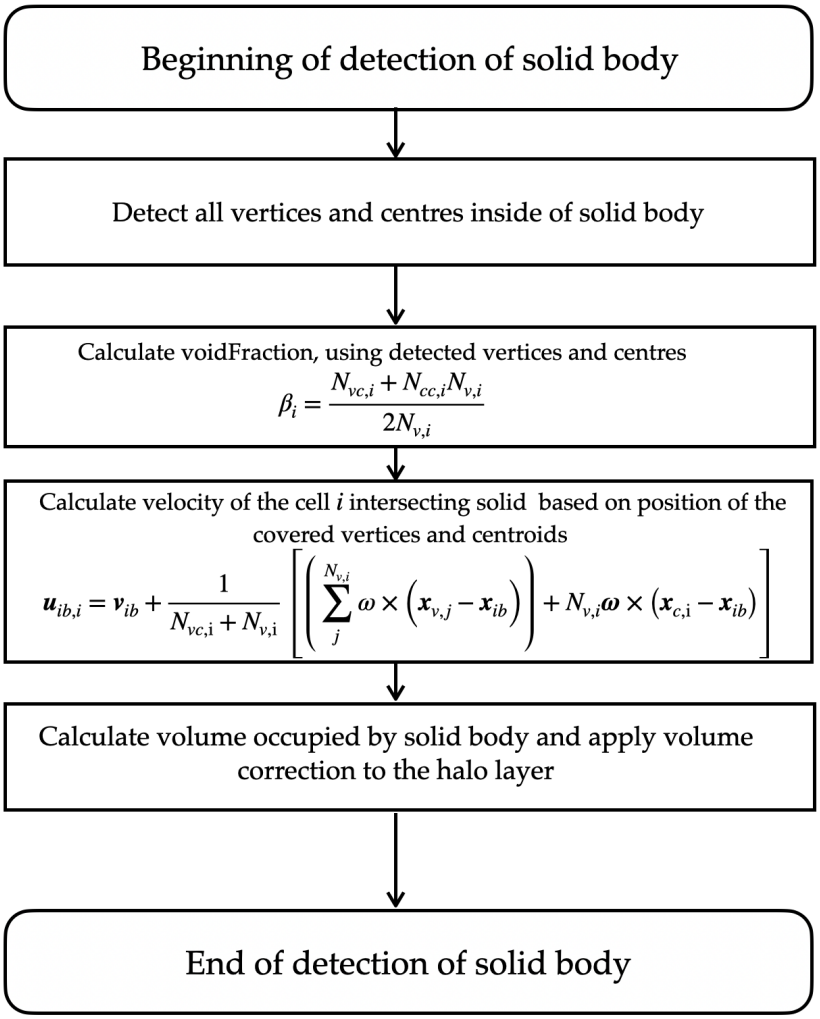


Figure 2.5: Schematic diagram for the defining process of the solid on the fluid mesh

2.5 Discrete element method

In the simulation of granular media and solid body motion, the Discrete Element Method (DEM) serves as a foundational approach. DEM models consist of two core components: the individual elements, which can be particles like sand grains or stone voussoirs, and the contacts between them. The first step in any DEM-based simulation

is to establish a geometrical model, specify the mechanical properties of the elements and contacts, and define the time-varying loads on the system.

While perfectly spherical elements are computationally convenient, they often result in unrealistic behavior, particularly in non-cemented assemblies. This has been understood since the 1980s, leading to the use of 'clumps' for representing non-spherical particles. A clump is essentially an aggregate of multiple overlapping spheres rigidly connected to approximate the shape of a complex particle. These clumps move as a single entity, effectively mimicking a particle with a more intricate shape. Graphical examples of clumps are shown in Figures **??**, **??**.

The use of clumps introduces some computational overhead due to the increased number of contact interactions and complexity in force calculations. However, this is still more efficient than directly simulating truly irregular shapes, which would require even more complex contact detection algorithms. The use of clumps is a straightforward and computationally efficient way to represent non-spherical particles in DEM simulations. By adjusting the number, size, and arrangement of spheres within a clump, various particle shapes can be approximated with varying levels of accuracy. Using clumps can increase the computational cost of DEM simulations compared to using simple spherical particles, as the number of contact interactions and the complexity of force calculations may increase. However, this approach is still more efficient than directly simulating truly irregular particle shapes, which would involve more complex contact detection and force calculations.

The particles motion formulated in Lagrangian frame of reference [**?**] with momentum and inertia equations:

$$m_i \frac{d\boldsymbol{u}_{\mathrm{p},i}}{dt} = \mathbf{F}_{i,n} + \mathbf{F}_{i,t} + \mathbf{F}_{i,f} + \mathbf{F}_{i,b} \tag{2.39}$$

$$I_i \frac{d\boldsymbol{\omega}_{\mathrm{p},i}}{dt} = \mathbf{r}_{i,c} + \mathbf{F}_{i,t} + \mathbf{T}_{i,r}, \tag{2.40}$$

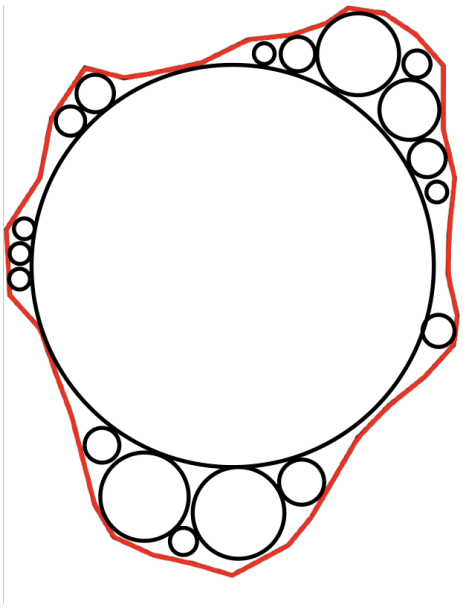


Figure 2.6: Schematic representation of the solid

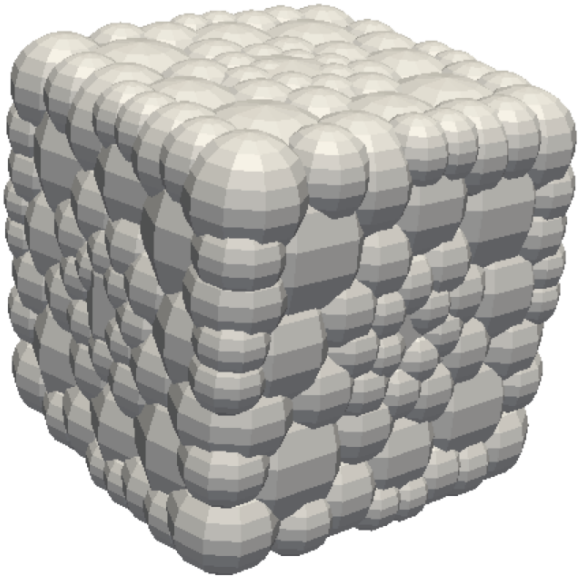


Figure 2.7: STL image of the solid body model

where $\mathbf{u}_{\mathrm{p},i}$ - translational velocity of the particle i, $\boldsymbol{\omega}_{\mathrm{p},i}$ - angular velocity of the particle i, $\mathbf{F}_{i,n}$ is the normal particle-particle contact force, $\mathbf{F}_{i,t}$ tangential particle-particle contact force. $\mathbf{F}_{i,f}$ force from surrounding fluid phase. All forces like gravity, electrostatic forces etc are summarised to $\mathbf{F}_{i,b}$ and $\mathbf{T}_{i,r}$ is an additional torque on the particle.

The second moment of inertia I_p is essential for calculating stress, for the case of a sphere, it could be calculated as:

$$I_p = \frac{2m_p r^2}{5} \tag{2.41}$$

In this application, the discrete element method used from LIGGGHTS® package, which is an open-source software designed for simulating granular flows and their interactions. LIGGGHTS® is built on C++ and based on LAMMPS® molecular dynamics simulator.

2.5.1 Time integration

In LIGGGHTS [**?**], the second-order accurate Velocity-Verlet method is employed for time integration. The method consists of two steps. First, the particle position r_i is integrated over the entire time step using the half-step velocity:

$$\mathbf{u}_{\mathbf{p},i}(t + \Delta t/2) = \mathbf{u}_{\mathbf{p},i}(t) + \frac{\Delta t}{2}\frac{\partial \mathbf{u}_{\mathbf{p},i}}{\partial t}(t) \tag{2.42}$$

$$\mathbf{r}(t + \Delta t) = \mathbf{r}(t) + \Delta t \mathbf{u}_{\mathbf{p},i}(t + \Delta t/2) \tag{2.43}$$

In the subsequent step, the remaining half-step of the velocity is integrated as follows:

$$\mathbf{u}_{\mathbf{p},i}(t + \Delta t) = \mathbf{u}_{\mathbf{p},i}(t + \Delta t/2) + \frac{\Delta t}{2}\frac{\partial \mathbf{u}_{\mathbf{p},i}}{\partial t}(t + \Delta t/2) \tag{2.44}$$

Here, $\partial \mathbf{u}_{p,i}/dt(t + \Delta t)$ is evaluated from the acting forces as described in **??**. This two-step process ensures accurate time integration of particle positions and velocities, taking into account the forces acting on each particle.

2.5.2 Particle–particle contact model

The used method is based on the theory of Cundall and Strack[**?**], where the complex collision behaviour of spheres is approximated by simple spring-dashpot interactions in normal (n) and tangential (τ) direction as

$$\boldsymbol{F}_{ij}^{int} = \boldsymbol{F}_{ij}^{n} \cdot n_{ij} + \boldsymbol{F}_{ij}^{\tau} \cdot \tau_{ij} \tag{2.45}$$

$$\boldsymbol{F}_{\mathrm{n}} = k_{\mathrm{n}}\delta_{\mathrm{n}} - \eta_{\mathrm{n}}\boldsymbol{u}_{\mathrm{p,n}}^{\mathrm{rel}} \tag{2.46}$$

$$\boldsymbol{F}_{\tau} = k_{\tau}\delta_{\tau} - \eta_{\tau}\boldsymbol{u}_{\mathrm{p},\tau}^{\mathrm{rel}} \tag{2.47}$$

where k is the spring coefficient from Herz contact theory, δ_n the particle overlap, η_τ the viscous damping coefficient from Mindlin-Deresiewicz [**?**] and $\mathbf{u}^{rel}$ the relative velocity of the colliding particles.

For a spherical particle, the moment of inertia I_i can be considered as a scalar because the sphere exhibits symmetry about all axes. In this case, the moment of inertia will be the same for any axis of rotation that passes through the sphere's center. This simplifies the mathematical description and calculations, as the direction of the axis of rotation doesn't need to be considered—the value of the moment of inertia will be constant in any case.

In vector form, the moment of inertia is usually represented by a tensor, which is generally a matrix. However, for a sphere, this matrix would be diagonal with identical diagonal elements, allowing it to be simplified to a scalar value.

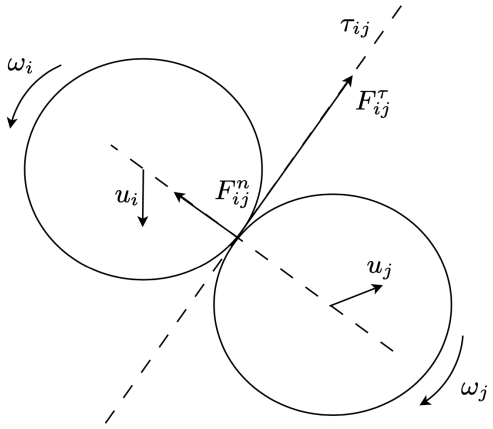


Figure 2.8: Force directions for a multi-spherical body.

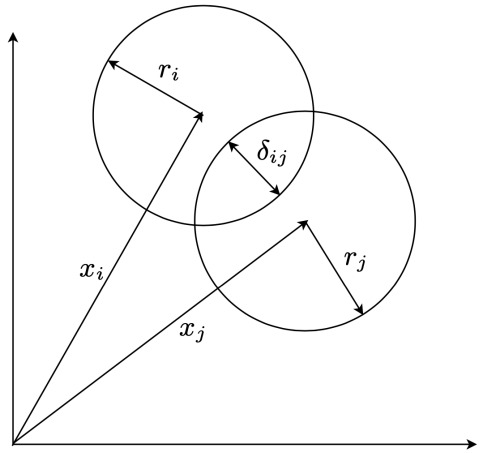


Figure 2.9: Schematic representation of contact model for a multi-spherical body.

$$
\begin{aligned}
k_{\mathrm{n}} &= \frac{4}{3} E_{\mathrm{eff}} \sqrt{r_{\mathrm{eff}} \delta_{\mathrm{n}}} \\
k_{\tau} &= 8 G_{\mathrm{eff}} \sqrt{r_{\mathrm{eff}}\, \delta_n}
\end{aligned}
\tag{2.48}
$$

where r_{eff} the effective radius, E_{eff} Young's modulus and G_{eff} shear modulus, which

defined for two colliding particles as:

$$\begin{aligned} r_{\text{eff}} &= \left(\frac{1}{r_i} + \frac{1}{r_j} \right)^{-1} \\ E_{\text{eff}} &= \left(\frac{1-\nu_{\text{p},i}^2}{E_i} + \frac{1-\nu_{\text{p},j}^2}{E_j} \right)^{-1} \\ G_{\text{eff}} &= \left(\frac{2-\nu_{\text{p},i}}{G_i} + \frac{2-\nu_{\text{p},j}}{G_j} \right)^{-1}, \end{aligned} \tag{2.49}$$

where ν is the Poisson's ratio and

$$\begin{aligned} \eta_{\text{n}} &= -\alpha(e)\sqrt{m_{\text{eff}} k_{\text{n}}} \\ \eta_{\tau} &= -\alpha(e)\sqrt{\frac{2}{3} m_{\text{eff}} k_{\tau}} \end{aligned} \tag{2.50}$$

more details could be found in [**?**] documentation.

2.6 Coupling

There are several methods which is available for solving the two-way coupling problem with CFDEMcoupling code, they listed below in increasing complexity order:

- The Immersed Boundary Method based on Distributed Lagrange Multipliers [**?**] was introduced by Shirgaonkar [**?**].
- Hager [**?**] adopted Shirgaonkar's approach, using solid body force correction with the PISO algorithm. The software implementation of this approach is provided as the CFDEMcoupling code [**?**].
- Blais [**?**] stores the solid body force as a separate field and adjusts the same implementation as Hager [**?**].
- The proposed approach in the current work for two-phase flow is based on the methods mentioned above and the VOF method with isoAdvector for free surface reconstruction.

In the paper by Balachandran et al. [**?**], a resolved CFD-DEM method was used for blood flow simulations. This work serves as an example of two-way coupling with a solid moving body having 6 degrees of freedom (DOF) for one fluid phase. The model of a red blood cell is treated as a cluster of overlapping spheres interconnected by a flexible mathematical bond. The inter-cellular interactions are modeled using a Hertz-Mendelian model, with attractive forces enforced through a cohesive force based on the Morse potential. A similar idea is used as the core concept for simulating granular media of artificial shape as rocks and boulders.

The resolved CFD-DEM method is technically an Immersed Boundary Method with a fictitious domain for the flow, where the DEM describes particle interactions. The full description of the system is as follows:

$$\frac{\partial(\rho\mathbf{u})_i}{\partial t} + \nabla p_i + \nabla\cdot(\rho\mathbf{u}\mathbf{u})_i - \nabla\tau_i + \rho_i\mathbf{g} = \mathbf{f}_s \quad in\ \Omega_f \cup \Omega_a \tag{2.51}$$

$$\nabla\cdot\mathbf{u}_i = 0 \quad in\ \Omega_f \cup \Omega_a \tag{2.52}$$

$$\frac{\partial\alpha}{\partial t} + \nabla\cdot(\alpha\mathbf{u}_i) = 0 \quad in\ \Omega_f \cup \Omega_a \tag{2.53}$$

$$m_p\frac{\mathrm{d}\mathbf{u_p}}{\mathrm{d}t} = m_p\mathbf{g} + F_p^f + \sum_{N_p} F_p^p + \sum_{N_w} F_p^w + F_p^{\mathrm{ext}} \quad in\ \Omega_s \tag{2.54}$$

$$\mathbf{u} = \mathbf{u}_\Gamma \quad on \in \Gamma, \mathbf{u} = \mathbf{u}_p \quad on\ \Omega_s \tag{2.55}$$

$$I_p\frac{\mathrm{d}\omega_\mathbf{p}}{\mathrm{d}t} = \mathbf{r}\times\left(F_p^f + \sum_{N_p} F_p^p + \sum_{N_w} F_p^w + F_p^{\mathrm{ext}}\right) \quad in\ \Omega_s \tag{2.56}$$

where $\mathbf{f}_s$ - is a force from the solid body side which calculated with the DEM, Ω_f, Ω_a - is the volume taken by fluid and air correspondingly. m_p - is the particle mass, F_f^p is the hydrodynamic forces acting on the particle and F_p^p and F_p^w are the particle–particle and particle–wall interaction forces, respectively. F_p^{ext} is the p external force acting on the particle which comes from cohesion, adhesion and other interaction forces. I_p is the second moment of inertia of the particle, ω_p is the angular velocity, and r is the

distance between the particle centroid and a point on the surface of the particle.

There are two primary categories of CFD-DEM: unresolved and resolved. In the unresolved method, the particle size is significantly smaller than the computational cell. Here, the particles' motion is simulated as if they were another fluid, and this is solved using the Navier-Stokes momentum equation [**?**]. In contrast, in the resolved method, the particle size is considerably larger than the computational cell, as shown in the figure below **??** For the resolved CFD-DEM there is two main approaches how

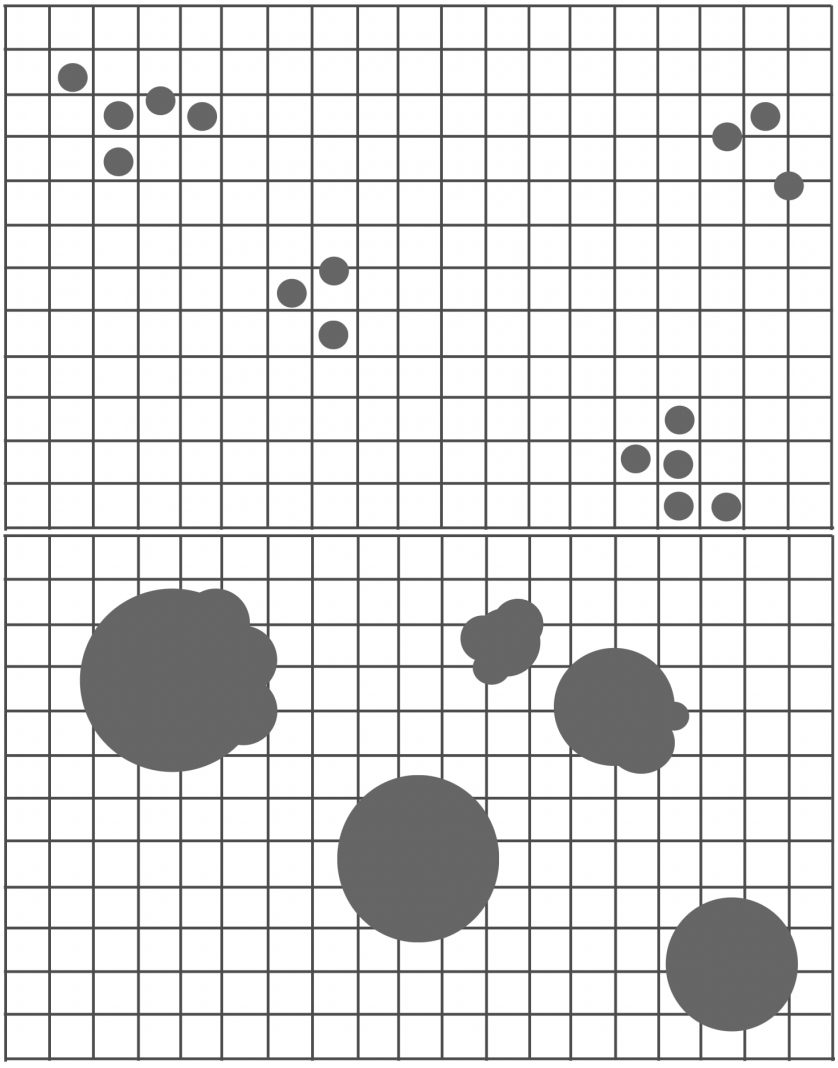


Figure 2.10: Particle representation approaches

to provide the coupling. One is direct forcing approach[**?**] which could be used to

modify the flow with a no-slip boundary condition which change the particle velocity. The main feature of the approach is dependence on the spatial discretization which used for IB flux calculation because the velocity field have to be divergence free.

Another one is continuous forcing approach [**?**] where the source term from the solid body **??** added before discretization, therefore no independence from spatial discretization.

$$F_B = \frac{\rho\beta\left(\mathbf{u_p} - \mathbf{u}_f\right)}{\kappa\Delta t} \tag{2.57}$$

where $\mathbf{u}_f$ and $\mathbf{u}_p$ are fluid and particle velocity correspondingly. The solid volume fraction β or so-called *voidFraction*, $\beta = 1$ when the cell fully occupied with the solid, or $\beta = 0$ if it not occupied and at the interface $0 < \beta < 1$. κ is the rigid body constraint.

Note that the method has been used for Fluid Structure Interaction (FSI) problems [**?**], [**?**], [**?**], but original part is combination of two phase flow with surface reconstruction method interaction with solid bodies of arbitrary shape.

2.6.1 Force term calculation

To account solid body forces **??** during the computation of the system of governing equations we will add particle position data, save in *voidFraction* field and with it as source term in momentum equation during calculation pressure and velocity cycle which is based on PISO algorithm, described before in **??** section.

step 1: Prediction step, to find intermediate velocity will include body force calculated on previous time step:

$$a_P^{\mathrm{u}}\mathbf{u}_P + \sum_N a_N^{\mathrm{u}}\mathbf{u}_N = -\nabla p + \mathbf{F}_i^{m*} \tag{2.58}$$

then let $\mathbf{H}(\mathbf{u}) = -\sum_N a_N^{\mathrm{u}} \mathbf{u}_N$ and rewrite **??** as

$$\begin{aligned} a_P^{\mathrm{u}} \mathbf{u}_P &= \mathbf{H}(\mathbf{u}) - \nabla p \\ \mathbf{u}_P &= \left(a_P^{\mathrm{u}}\right)^{-1} \left(\mathbf{H}(\mathbf{u}) - \nabla p\right) \end{aligned} \tag{2.59}$$

where P is an index arbitrary velocity node, N is an index from neighbouring cells, $\mathbf{F}_i^{m*}$ - is immersed boundary forcing term. We also need to solve phase convection equation at this step.

step 2: Correction step, Compute flux on a cell faces using Rhie-Chow [**?**] correction

$$\phi = \frac{H(\mathbf{u})}{a_P^u} \cdot S_f \tag{2.60}$$

where S_f is area of a cell face. From **??** to satisfy continuity equation $\nabla \cdot \mathbf{u} = 0$

$$(a_P^{\mathbf{u}})^{-1} \cdot \nabla^2 p^* = \nabla \cdot \phi \tag{2.61}$$

then correct the velocity field:

$$\mathbf{u}^* = (a_P^{\mathbf{u}})^{-1} (H(\mathbf{u}) - \nabla p^* + \mathbf{F}_i^{m*}) \tag{2.62}$$

where $\mathbf{u}^*$ and p^*,an intermediate velocity and a pressure. Finally, this forcing term is corrected using the difference between the current velocity and the prescribed one within the immersed body

$$\mathbf{F}_i^{m**} = \boldsymbol{F}_i^{m*} + \frac{\alpha \beta_i}{\Delta t} \left(\boldsymbol{u}_{ib,i} - \boldsymbol{u}_i^{m**} \right) \tag{2.63}$$

Then repeat step 2 until it converged or depends on how many correction steps we choose to run, after write down new velocity and pressure fields $\mathbf{u}^{m+1}$ and p^{m+1} for the next step $m+1$. Schematically algorithm shown on the figure **??** below.

In conculsion, although final algorithm including a lot of details, it build on already existing techniques which need to work all together....

smth elseeee!@!!

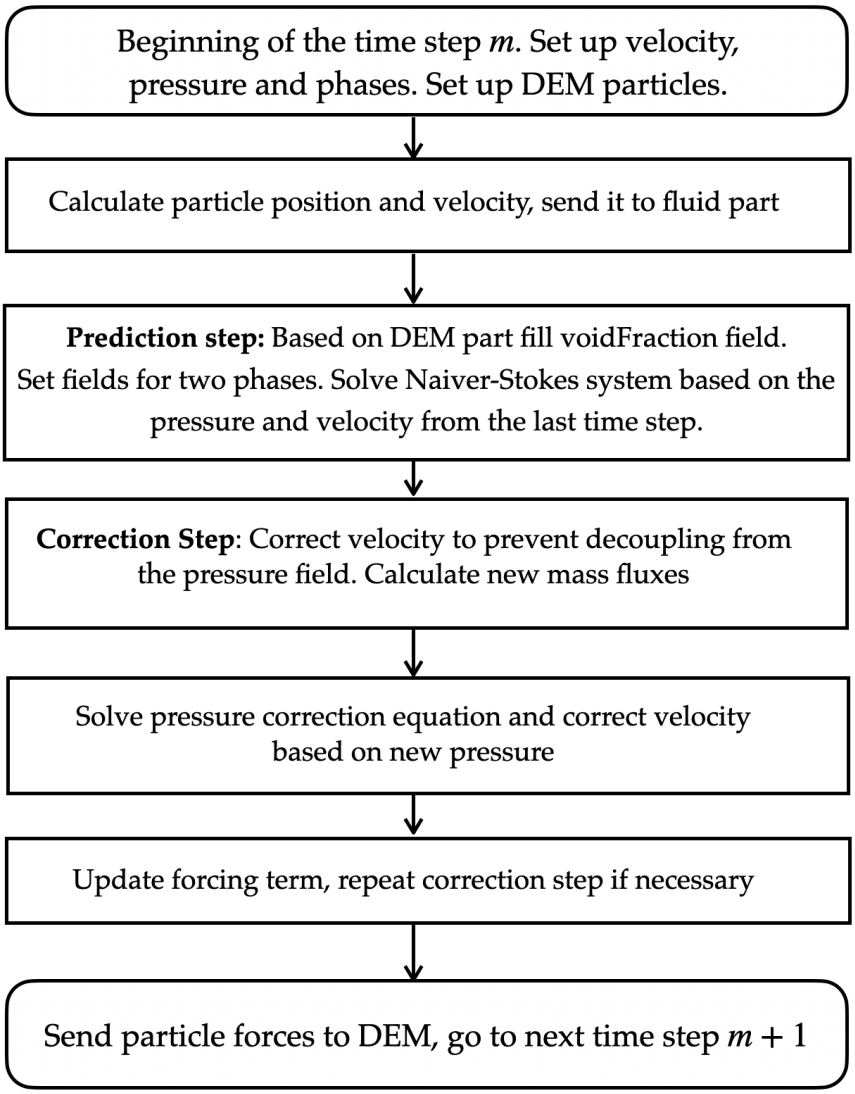


Figure 2.11: Schematic diagram for CFD-DEM algorithm.

Chapter 3: Computational Methods

3.1 Software: OpenFOAM and LIGGGHTS

3.2 Parallel Computations

3.3 Validation and Verification

3.4 Resolved two phase CFD-DEM validation

3.5 Implementation

An implementation of the method provided in the Chapter **??** based on OpenFOAM [**?**] open-source programming library which includes finite volume solvers and Discrete Element Method (DEM) used from LIGGGHTS® package which is coupled with OpenFOAM through CFDEMcoupling [**?**] implementation. LIGGGHTS® is based on LAMMPS [**?**] a classical molecular dynamics simulator based on the time integration algorithm using Verlet symplectic integrator [**?**]. One other benefit of using LAMMPS, that it supports large scale parallelism. Distributed-memory parallelism via MPI with differenet ways of spatial decomposition. This all give opportunity to run the whole simulation in parallel on HPC cluster.

3.5.1 cfddemSolverMultisphereIB

The schematic description is shown on the figure below **??**.

After initialization of DEM and CFD, we send DEM data to CFD part. Identify cells occupied with solid bodies and save this cells as a *voidFraction* field.

At the same time we initialize two phases, marking each cell of CFD in the range $[0, 1]$.

Then we starting PISO loop. First, solve momentum equation based on velocity

fluid field from previous step with two phases and *voidFraction* field also from the previous step or based on initial conditions, excluding pressure field.

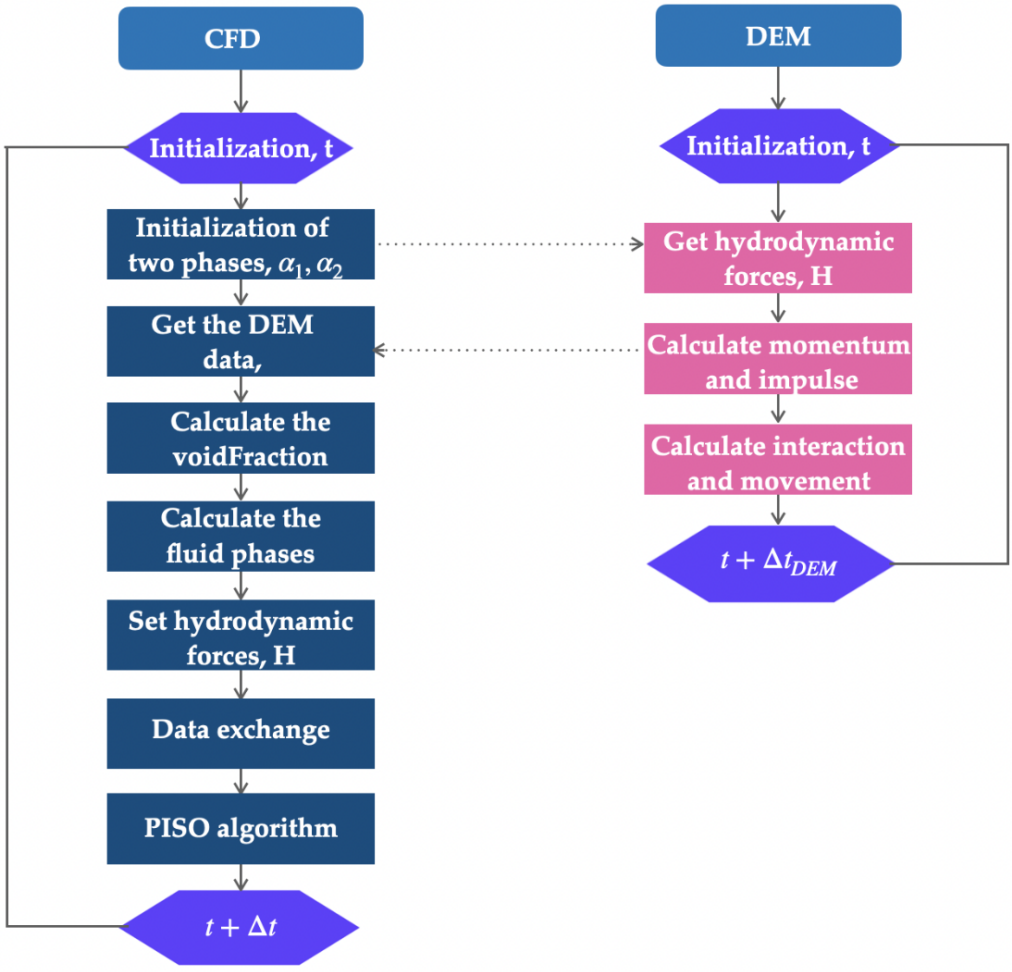


Figure 3.1: Schematic description of CFD-DEM solver.

The step of force calculation is the most important part of the simulation. Force calculated in function `IBpressureForce::setForce()`. Where we should first initialize pressure, then for all cells which we found related to the DEM part on FVM mesh apply:

$$F = -\nabla p \mathbf{u}(1 - voidFraction)$$

3.5.2 Time step calculation

To run simulations with CFDEMcoupling we need to set up time step for DEM and for CFD solvers separately. It is common to use Courant (or Courant-Friedrichs-Lewy) number as a reference to get a solution converge.

To calculate time step we will use relationship:

$$\Delta t = \frac{CFL \times \Delta x}{\mathbf{u}}, \tag{3.1}$$

where $\mathbf{u}$ - fluid velocity, Δt - time step, Δx - grid size and $CFL \in \mathbb{Q}$ is the Courant number. It indicates the number of cells a *fluid particle* passes in one time step. Consequently, a calculation can only be considered stable if the Courant number is smaller or equal to one.

3.5.3 Computational efficiency

The algorithm presented on the Fig **??** is not straightforward. We need to pass back and forth data from one system to another. Thanks to API incorporated by a CFDDEM developers team, there was no need to develop this part. API works as as follows:

- set up equation which needed to be solved on fluid side
- create a force model with CFDDEM, calculate how solid body affect fluid.
- set up LIGGGHTS code, receive data form OpenFOAM through CFDDEM force model, calculate the solid body motion.

The software combination is takes a lot of resources and API is done using MPI. Moreover all process could be run parallel on CPU. When we are increasing the domain size, the number of CPUs required for running the simulation increased due to an increase in the finite volume mesh size.

The DEM calculations takes 1/8 of all computational time. The VOF part takes 1/4 and parallel communication takes around same resources as DEM part. The main consumption, around half of all time, is coming from mapping the Lagrangian DEM data to a Eulerian field. This step is critical for immersed boundary method, because velocity of the solid bodies is required to support continuous forcing approach.

3.6 Preliminary computational results

We run of simulation to make sure that implemented solver work as we suppose. Otherwise we run validation experiment. For that reason we choose three different type of experiments:

- **A falling sphere example, 1 phase**. Experiment for one phase and one solid body.
- **A bouncing sphere example, water-air**. Experiment with two phases and one solid body. Density of the body smaller than one of the phase which make the body bounce and do not sink.
- **A falling multi-spherical body** into a water. This will be good example of that solver works.

3.6.1 A falling sphere example, 1 phase

For initial validation of created numerical model as a first step we compared one phase falling sphere simulation with the work [**?**]. Parameters of the fluid and particle which was used for the simulation provided in the table **??**. On the figure **??** shown simulation from the work [**?**] and on the figure **??** shown results of simulation of the current solver. Visually it can be seen that the velocity of the fluid field around both particles is almost the same. On the figure **??** provided comparison of trajectory in z-direction and velocity of the falling particle.

Table 3.1: Simulation Parameters

Simulation Part	Physical Parameters (units)	Value
Particle	Density (kg/m^3)	1200
	Diameter (cm)	0.167
	Initial height (cm)	3.5
Fluid	Density (kg/m^3)	1000
	Viscosity (m^2/s)	1e^{-6}

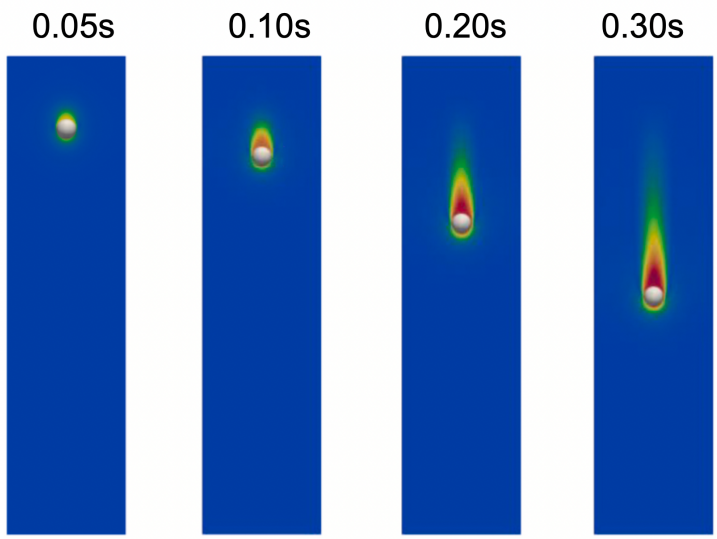


Figure 3.2: Falling sphere simulation in 1 phase fluid from the work [**?**].

3.6.2 A bouncing sphere example, water-air

The initial experiment comes from the work [**?**]. Where they using the tank parameters: length: 109.7 m, width 6.7 m, depth 3.2 m.

The sphere radius $r = 0.252$ m , height $z = r/10$ above the free surface. The sphere density 500 kg/m .

First, we tried to make regular mesh similar to Pathak etc.[**?**] work. For the domain size from Pathak etc.[**?**] we followed formula:

$$(length \cdot height \cdot width) \cdot r \cdot CPR, \tag{3.2}$$

where CPR - cell per radius of sphere.

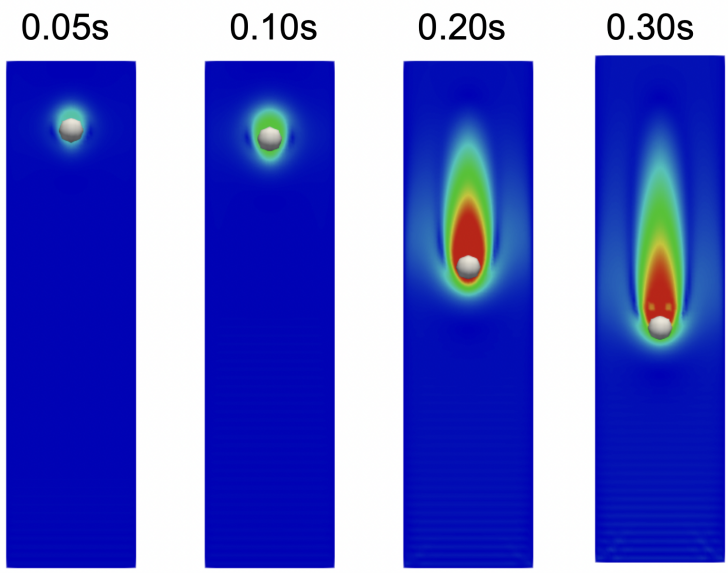


Figure 3.3: Falling sphere simulation in 1 phase fluid from the CFD-DEM simulation

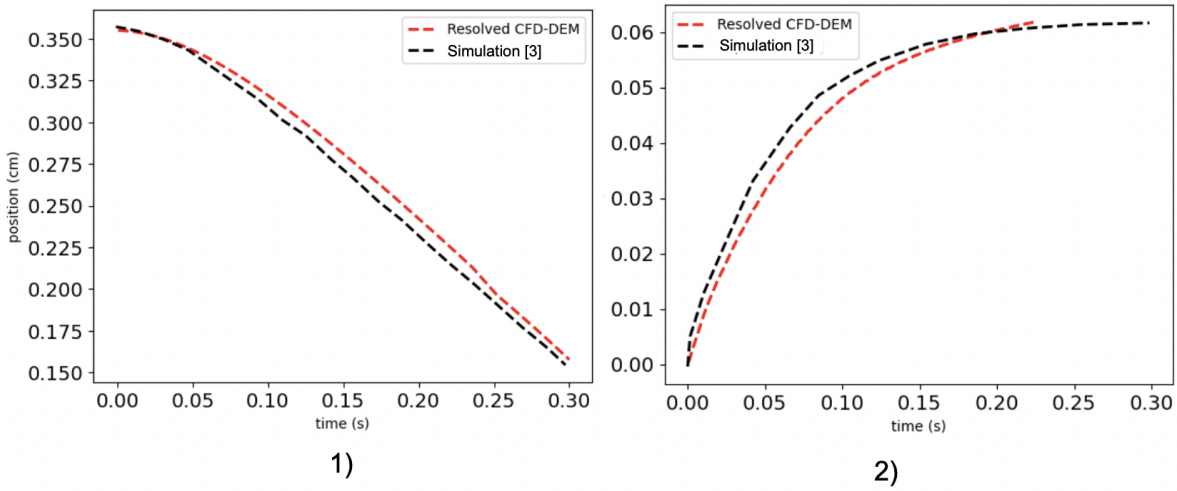


Figure 3.4: Falling sphere in 1 phase fluid 1) comparison in z-direction 2) velocity of the particle

For the lowest mesh resolution results of simulation shown on the figure below **??**. The main parameters shown in the table below

Results of error with physical experiment compared with original experiment [**?**] and [**?**].

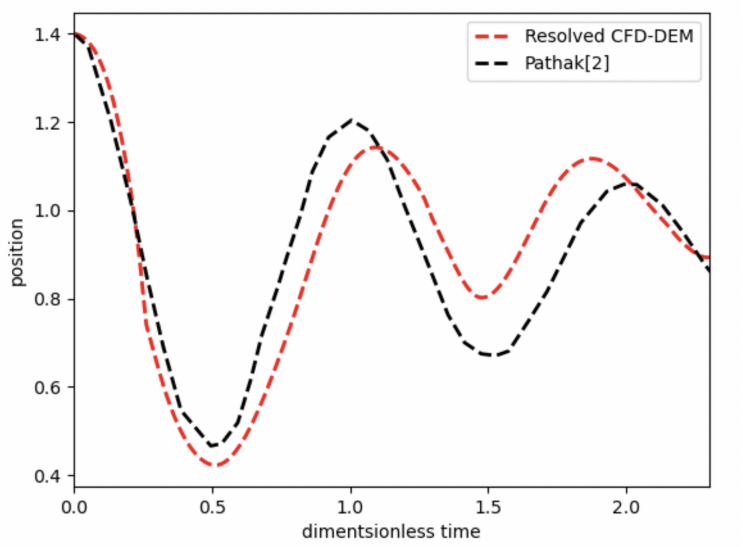


Figure 3.5: Bouncing sphere simulation for two-phase fluid from the CFD-DEM simulation

Table 3.2: Simulation Parameters

Simulation Part	Physical Parameters (units)	Value
Particle	Density (kg/m^3)	500
	Diameter (cm)	0.252
	Initial height (cm)	1.0252
Fluid	Density (kg/m^3)	1000
	Viscosity (m^2/s)	1e^{-6}
Air	Density (kg/m^3)	$1e-5$
	Viscosity (m^2/s)	1e^{-6}

3.6.2.1 Computational setup

For the numerical simulation we choose IsoAdvector [**?**] method for simulation of free surface which is described in a theoretical part. Applying second order Crank–Nicolson method for time differentiation . For interpolation diffusion and convection terms we used scheme which called *Gauss linear* in OpenFOAM, which is the second-order accurate scheme, the standard finite volume discretization using the Gaussian integration that requires the interpolation from cell centers to face centers. Pressure field tolerance was chosen as $1e-06$. With 3 PISO loops on each time step. The time

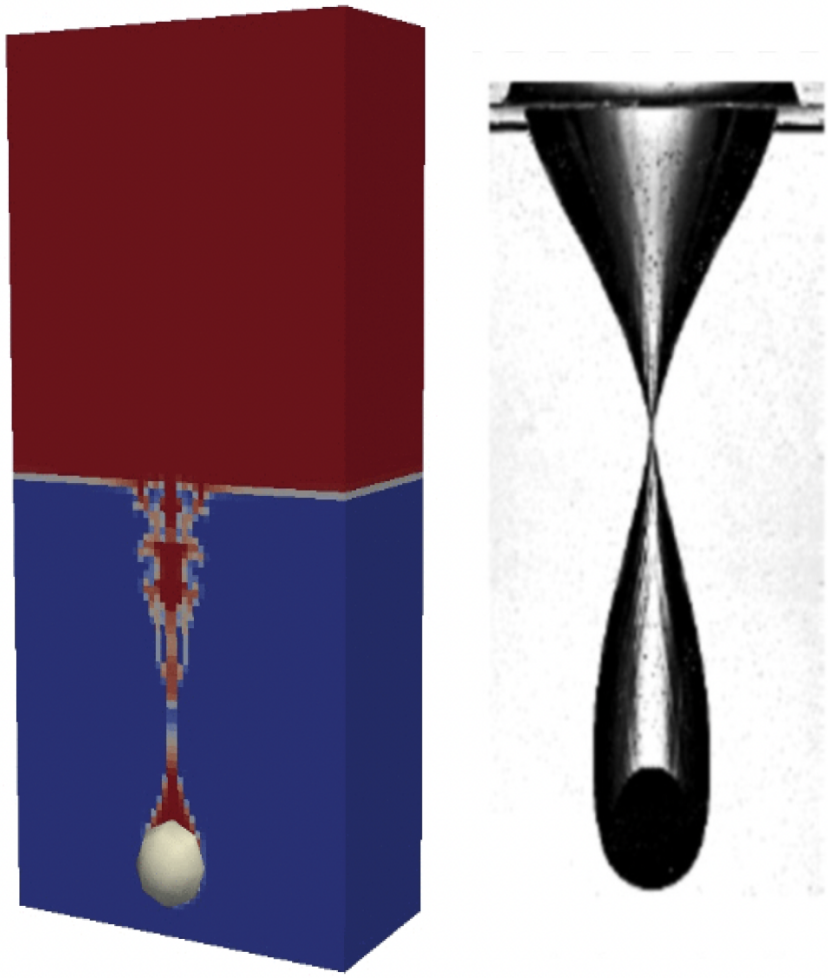


Figure 3.6: Falling sphere visualization results. On the right, cavity after falling sphere in the water [**?**].

step we choose $\Delta t = 1e-05$ for the VOF part and $\Delta t = 1e-06$ for the DEM part i.e communication was made every 10 time steps from the solid body part.

3.6.3 Multi-spherical body interaction with two-phase fluid

The primary objective of this experiment is to illustrate initial stability and functionality of the developed CFD-DEM code through simulating the interaction of a multi-spherical body, analogous to a stone, with water.

The simulation involves a multi-spherical body consists out of 2 overlapping spheres, positioned at a 0.4 m height above a fluid with the proprieties of water. The main

properties are provided in the table below.

Table 3.3: Simulation Parameters

Simulation Part	Physical Parameters (units)	Value
Particle 1	Density (kg/m^3)	500
	Diameter (m)	0.25
	Initial height (m)	(0.5, 0.5, 1.4)
Particle 2	Density (kg/m^3)	500
	Diameter (m)	0.25
	Initial height (m)	(0.2, 0.4, 1.4)
Fluid	Density (kg/m^3)	1000
	Viscosity (m^2/s)	1e^{-6}
Air	Density (kg/m^3)	$1e-5$
	Viscosity (m^2/s)	1e^{-6}

The body is released and allowed to fall into the water under gravity forces. The simulation captures the motion of the body and the fluid's response upon impact and during the subsequent settling period. The results of simulation showed on a figure **??** below.

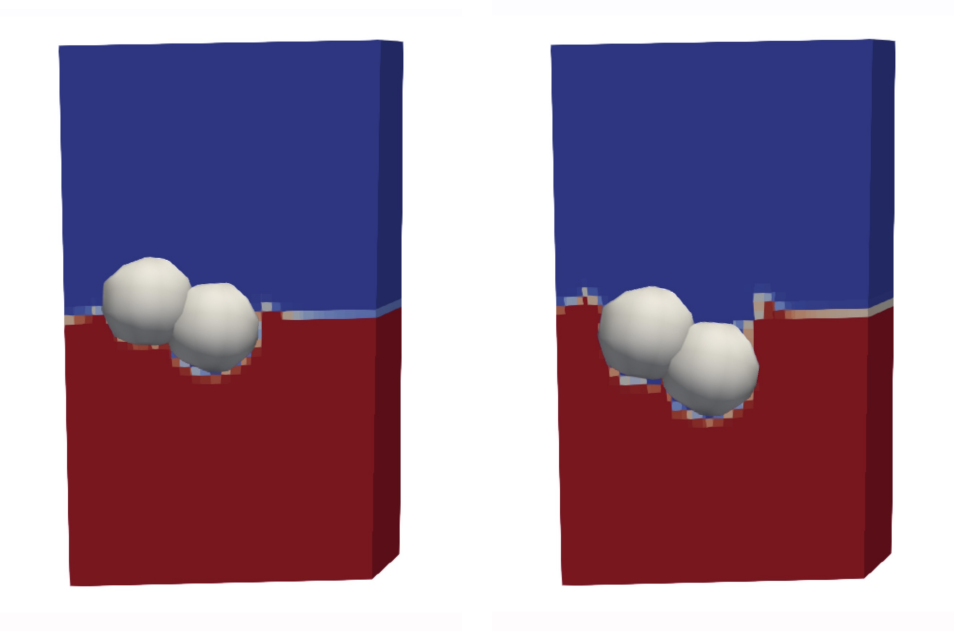


Figure 3.7: Falling 2 spheres.

The results of simulation showed numerical stability of the solver. The main

concern was the volume of fluid phases, it changes in the range less than 1%. The future work and conclusion provided in the next section.

3.6.4 Grid convergence analysis: Multi-spherical body interaction with two-phase fluid

According to Roache's definition of code verification [**?**], to confirm the ability to accurately solve the set of governing equations. A main strategy for validating the code is to convey a grid convergence study, which involves executing multiple simulations on successive finer grids. The discretization error is expected to approach zero asymptotically as the grid refined. The major point of interest in this context is the approximation order derived from the numerical method. For instance, if a second-order accurate method is used, it is reasonable to anticipate second-order convergence.

The discrepancy between the exact solution, denoted as f_{exact}, and the computed solution, represented as $f(h)$, as a result of the discretization error, can be expressed as follows:

$$f_{\text{exact}} - f(h) = c \cdot h^p + O\left(h^{p+1}\right) \tag{3.3}$$

Chapter 4: Results

4.0.1 Landslide Simulations

4.0.2 Sensitivity Analysis

Bibliography

[1] Github. `https://github.com`. Accessed: 2023-04-04.

[2] Stefan Adami, Xiangyu Y Hu, and Nikolaus A Adams. A generalized wall boundary condition for smoothed particle hydrodynamics. *Journal of Computational Physics*, 231(21):7057–7075, 2012.

[3] Cyrus K Aidun and Jonathan R Clausen. Lattice-boltzmann method for complex flows. *Annual review of fluid mechanics*, 42:439–472, 2010.

[4] Christopher R Anthony, Hansol Wee, Vishrut Garg, Sumeet S Thete, Pritish M Kamat, Brayden W Wagoner, Edward D Wilkes, Patrick K Notz, Alvin U Chen, Ronald Suryo, et al. Sharp interface methods for simulation and analysis of free surface flows with singularities: Breakup and coalescence. *Annual Review of Fluid Mechanics*, 55, 2023.

[5] Achuth Nair Balachandran Nair, Stefan Pirker, and Mahdi Saeedipour. Resolved cfd-dem simulation of blood flow with a reduced-order rbc model. *Computational Particle Mechanics*, pages 1–16, 2021.

[6] A Barreiro, AJC Crespo, JM Domínguez, and M Gómez-Gesteira. Smoothed particle hydrodynamics for coastal engineering problems. *Computers & Structures*, 120:96–106, 2013.

[7] Robert F Beck and Stergios Liapis. Transient motions of floating bodies at zero forward speed. *Journal of ship research*, 31(03):164–176, 1987.

[8] Romana Begum and M Abdul Basit. Lattice boltzmann method and its applications to fluid flow problems. *European Journal of Scientific Research*, 22(2):216–231, 2008.

[9] Ted Belytschko. Fluid-structure interaction. *Computers & Structures*, 12(4):459–469, 1980.

[10] Friedrich-Karl Benra, Hans Josef Dohmen, Ji Pei, Sebastian Schuster, and Bo Wan. A comparison of one-way and two-way coupling methods for numerical analysis of fluid-structure interactions. *Journal of applied mathematics*, 2011, 2011.

[11] Richard Berger, Christoph Kloss, Axel Kohlmeyer, and Stefan Pirker. Hybrid parallelization of the liggghts open-source dem code. *Powder technology*, 278:234–247, 2015.

[12] Bruno Blais, Manon Lassaigne, Christoph Goniva, Louis Fradette, and François Bertrand. A semi-implicit immersed boundary method and its application to viscous mixing. *Computers & Chemical Engineering*, 85:136–146, 2016.

[13] Carl Boettiger. An introduction to docker for reproducible research. *ACM SIGOPS Operating Systems Review*, 49(1):71–79, 2015.

[14] Jeremiah U Brackbill, Douglas B Kothe, and Charles Zemach. A continuum method for modeling surface tension. *Journal of computational physics*, 100(2):335–354, 1992.

[15] Shiyi Chen and Gary D Doolen. Lattice boltzmann method for fluid flows. *Annual review of fluid mechanics*, 30(1):329–364, 1998.

[16] Jack Chessa and Ted Belytschko. An extended finite element method for two-phase fluids. *J. Appl. Mech.*, 70(1):10–17, 2003.

[17] Clay Mathematics Institute. Millennium problems: Navier-stokes equation, 2022. `https://www.claymath.org/millennium-problems/navier-stokes-equation`, Last accessed on 2022-06-15.

[18] Peter A Cundall and Otto DL Strack. A discrete numerical model for granular assemblies. *geotechnique*, 29(1):47–65, 1979.

[19] Niels G Deen, Martin van Sint Annaland, and JAM Kuipers. Direct numerical simulation of complex multi-fluid flows using a combined front tracking and immersed boundary method. *Chemical engineering science*, 64(9):2186–2201, 2009.

[20] Suraj S Deshpande, Lakshman Anumolu, and Mario F Trujillo. Evaluating the performance of the two-phase flow solver interfoam. *Computational science & discovery*, 5(1):014016, 2012.

[21] Mu-Yu Duan and Ren-Qing Zhu. Numerical simulation of violent liquid sloshing in a tank based on youngs method. *Chuan Bo Li Xue*, 13(1):9–18, 2009.

[22] Adolfo Esteban, Joaquín López, Pablo Gómez, Claudio Zanzi, Johan Roenby, and Julio Hernández. A comparative study of two open-source state-of-the-art geometric vof methods. *Computers & Fluids*, 250:105725, 2023.

[23] Joel H Ferziger, Milovan Perić, and Robert L Street. *Computational methods for fluid dynamics*, volume 3. Springer, 2002.

[24] Robert A Gingold and Joseph J Monaghan. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly notices of the royal astronomical society*, 181(3):375–389, 1977.

[25] Roland Glowinski, T-W Pan, Todd I Hesla, and Daniel D Joseph. A distributed lagrange multiplier/fictitious domain method for particulate flows. *International Journal of Multiphase Flow*, 25(5):755–794, 1999.

[26] José González-Cao, Corrado Altomare, Alejandro JC Crespo, JM Domínguez, Moncho Gómez-Gesteira, and Dogan Kisacik. On the accuracy of dualsphysics to assess violent collisions with coastal structures. *Computers & Fluids*, 179:604–612, 2019.

[27] Alice Hager. Cfd-dem on multiple scales: An extensive investigation of particle–fluid interactions. *Johannes Kepler University Linz, Linz*, 2014.

[28] 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.

[29] 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.

[30] Meng Huang, Lilong Wu, and Bin Chen. A piecewise linear interface-capturing volume-of-fluid method based on unstructured grids. *Numerical Heat Transfer, Part B: Fundamentals*, 61(5):412–437, 2012.

[31] Hrvoje Jasak, Aleksandar Jemcov, Zeljko Tukovic, et al. Openfoam: A c++ library for complex physics simulations. In *International workshop on coupled methods in numerical dynamics*, volume 1000, pages 1–20. IUC Dubrovnik Croatia, 2007.

[32] Fabian Peng Kärrholm. Rhie-chow interpolation in openfoam. *Department of Applied Mechanics, Chalmers University of Technology: Goteborg, Sweden*, 2006.

[33] Christoph Kloss and Christoph Goniva. Liggghts–open source discrete element simulations of granular materials based on lammps. *supplemental proceedings: materials fabrication, properties, characterization, and modeling*, 2:781–788, 2011.

[34] Christoph Kloss, Christoph Goniva, Alice Hager, Stefan Amberger, and Stefan Pirker. Models, algorithms and validation for opensource dem and cfd–dem. *Progress in Computational Fluid Dynamics, an International Journal*, 12(2-3):140–152, 2012.

[35] Roland W Lewis, Perumal Nithiarasu, and Kankanhalli N Seetharamu. *Fundamentals of the finite element method for heat and fluid flow*. John Wiley & Sons, 2004.

[36] Cheng Liu, Ruoqing Gao, and Changhong Hu. An approximated volume of fluid method with the modified height function method in the simulation of surface tension driven flows. *AIP Advances*, 12(8):085308, 2022.

[37] Dingzhu Liu, Jinbo Tang, Hao Wang, Yang Cao, Nazir Ahmed Bazai, Huayong Chen, and Daochuan Liu. A new method for wet-dry front treatment in outburst flood simulation. *Water*, 13(2):221, 2021.

[38] Jia Mao, Lanhao Zhao, Yingtang Di, Xunnan Liu, and Weiya Xu. A resolved cfd–dem approach for the simulation of landslides and impulse waves. *Computer Methods in Applied Mechanics and Engineering*, 359:112750, 2020.

[39] Raymond D Mindlin and HERBERT Deresiewicz. Elastic spheres in contact under varying oblique forces. 1953.

[40] Rajat Mittal and Gianluca Iaccarino. Immersed boundary methods. *Annu. Rev. Fluid Mech.*, 37:239–261, 2005.

[41] Joe J Monaghan. Simulating free surface flows with sph. *Journal of computational physics*, 110(2):399–406, 1994.

[42] Andreia Borges Moreira, Agnès Leroy, Damien Violeau, and Francisco de Almeida Taveira-Pinto. Overview of large-scale smoothed particle hydrodynamics modeling of dam hydraulics. *Journal of Hydraulic Engineering*, 146(2):03119001, 2020.

[43] Xuan Nan, Zhihao Shen, Jingming Hou, and Guodong Li. High-resolution model of complexly shaped bodies motion using an ibm-vof-dem coupling method. *Powder Technology*, 413:118005, 2023.

[44] Engineering National Academies of Sciences and Medicine. *Reproducibility and Replicability in Science*. The National Academies Press, Washington, DC, 2019.

[45] Eugenio Oñate and J Garcıa. A finite element method for fluid–structure interaction with surface waves using a finite calculus formulation. *Computer methods in applied mechanics and engineering*, 191(6-7):635–660, 2001.

[46] Stanley Osher, Ronald Fedkiw, and K Piechor. Level set methods and dynamic implicit surfaces. *Appl. Mech. Rev.*, 57(3):B15–B15, 2004.

[47] Ashish Pathak and Mehdi Raessi. A 3d, fully eulerian, vof-based solver to study the interaction between two fluids and moving rigid bodies using the fictitious domain method. *Journal of computational physics*, 311:87–113, 2016.

[48] Sid Perkins. Death toll from landslides vastly underestimated. *Nature News*, 8, 2012.

[49] Charles S Peskin. Flow patterns around heart valves: a numerical method. *Journal of computational physics*, 10(2):252–271, 1972.

[50] Patrick J Roache. *Verification and validation in computational science and engineering*, volume 895. Hermosa Albuquerque, NM, 1998.

[51] Johan Roenby, Henrik Bredmose, and Hrvoje Jasak. Isoadvector: Geometric vof on general meshes. In *OpenFOAM®*, pages 281–296. Springer, 2019.

[52] DS Schwalbach, Thomas Shepard, Seamus Kane, David Siglin, Teegan Harrington, and JP Abraham. Effect of impact velocity and mass ratio during vertical sphere water entry. *Dev. Appl. Oceanic Eng*, 3:55–62, 2014.

[53] Xiaopeng Shang, Xuan Zhang, Thien-Binh Nguyen, and Tuan Tran. Direct numerical simulation of evaporating droplets based on a sharp-interface algebraic vof approach. *International Journal of Heat and Mass Transfer*, 184:122282, 2022.

[54] Zhihao Shen, Gang Wang, Duruo Huang, and Feng Jin. A resolved cfd-dem coupling model for modeling two-phase fluids interaction with irregularly shaped particles. *Journal of Computational Physics*, 448:110695, 2022.

[55] Anup A Shirgaonkar, Malcolm A MacIver, and Neelesh A Patankar. A new mathematical formulation and fast algorithm for fully resolved simulation of self-propulsion. *Journal of Computational Physics*, 228(7):2366–2390, 2009.

[56] A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. in 't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott, and S. J. Plimpton. LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. *Comp. Phys. Comm.*, 271:108171, 2022.

[57] Murilo F Tome and Sean McKee. Gensmac: A computational marker and cell method for free surface flows in general domains. *Journal of Computational Physics*, 110(1):171–186, 1994.

[58] Grétar Tryggvason, Bernard Bunner, Asghar Esmaeeli, Damir Juric, N Al-Rawahi, W Tauber, J Han, S Nas, and Y-J Jan. A front-tracking method for the computations of multiphase flow. *Journal of computational physics*, 169(2):708–759, 2001.

[59] Markus Uhlmann. An immersed boundary method with direct forcing for the simulation of particulate flows. *Journal of Computational Physics*, 209(2):448–476, 2005.

[60] Loup Verlet. Computer" experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Physical review*, 159(1):98, 1967.

[61] T Wacławczyk and T Koronowicz. Remarks on prediction of wave drag using vof method with interface capturing approach. *Archives of civil and mechanical engineering*, 8(1):5–14, 2008.

[62] Tomasz Wacławczyk and Tadeusz Koronowicz. Comparison of cicsam and hric high-resolution schemes for interface capturing. *Journal of theoretical and applied mechanics*, 46:325–345, 2008.

[63] Teng Xiao, Bin Xie, Xi Deng, and Yanping Du. Numerical simulations for incompressible turbulence cavitation flows with tangent of hyperbola interface capturing (thinc) scheme. *Physics of Fluids*, 34(2):022108, 2022.

[64] Steven T Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of computational physics*, 31(3):335–362, 1979.

[65] Feng Zhang, Alejandro Crespo, Corrado Altomare, José Domínguez, Andrea Marzeddu, Shao-ping Shang, and Moncho Gómez-Gesteira. Dualsphysics: a numerical tool to simulate real breakwaters. *Journal of Hydrodynamics*, 30:95–105, 2018.

Appendix A: Appendix

This is an example of an appendix. The only difference is the use of `\appendix` command at the start of this `tex` file. This automatically changes the chapter and section headings.

A.1 A section

The easiest method.

$$x_k = \frac{a_k + b_k}{2} \tag{A.1}$$

A.2 False Position

Nulla ac nisl. Nullam urna nulla, ullamcorper in, interdum sit amet, gravida ut, risus. Aenean ac enim. In luctus. Phasellus eu quam vitae turpis viverra pellentesque. Duis feugiat felis ut enim. Phasellus pharetra, sem id porttitor sodales, magna nunc aliquet nibh, nec blandit nisl mauris at pede. Suspendisse risus risus, lobortis eget, semper at, imperdiet sit amet, quam. Quisque scelerisque dapibus nibh. Nam enim. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nunc ut metus. Ut metus justo, auctor at, ultrices eu, sagittis ut, purus. Aliquam aliquam.

A.3 Starting the Appendices

Actually, using appendices is quite simple. Immediately after the end of the last chapter and before the start of the first appendix, simply enter the command `\appendix`. This will tell LaTeX to change how it interprets the commands `\chapter`, `\section`, *etc.*

Each appendix is actually a chapter, so once the `\appendix` command has been called, start a new appendix by simply using the `\chapter` command.

Note that the `\appendix` command should be called only once–not before the start of each appendix.

All the fancy referencing and tools still work. You only need to add the appendix command and all will be as it should be.

Appendix B: Another Appendix

Maecenas non massa. Vestibulum pharetra nulla at lorem. Duis quis quam id lacus dapibus interdum. Nulla lorem. Donec ut ante quis dolor bibendum condimentum. Etiam egestas tortor vitae lacus. Praesent cursus. Mauris bibendum pede at elit. Morbi et felis a lectus interdum facilisis. Sed suscipit gravida turpis. Nulla at lectus. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Praesent nonummy luctus nibh. Proin turpis nunc, congue eu, egestas ut, fringilla at, tellus. In hac habitasse platea dictumst.