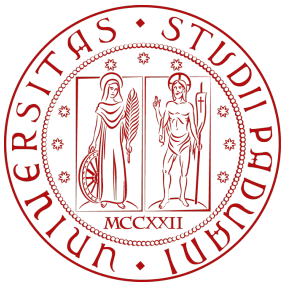


UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Dipartimento di Ingegneria Industriale

Corso di Laurea Magistrale in Ingegneria Aerospaziale

Tesi di Laurea Magistrale

DEVELOPMENT OF A PARTICLE TRACKING METHOD FOR COMPLEX FLOW GEOMETRIES

Relatore:
Prof. Francesco Picano

Laureando:
Lorenzo Martinuzzo
2085519

Correlatore:
Ing. Federico Dalla Barba

Anno Accademico 2023/2024

Abstract

Turbulent and multiphase flows have a fundamental role in the design phase not only in the engineering field. Nowadays, the investigation of such flows is mainly based on the numerical simulations, thanks to lower costs, times and complexity compare to experimental investigations.

In this thesis project, multiphase flows are considered, with a generic fluid as carrier and solid particles as secondary phase. Since dilute condition are hypothesized, one-way fluid-particle coupling is used.

The flow is simulated with the Direct Numerical Simulation (DNS) implemented in CaNS, an open-source code. For the purposes of simulating complex flow geometries, an Immersed Boundary Method (IBM) is adopted, in particular an Eikonal equation IBM is used, coupled with a Signed Distance Function.

The particles motion is simulated in parallel to the flow by solving the equations of motion. In addition to the particle motion, also the collision over rigid bodies is modelled. These rigid walls are modelled with the IMB, taking advantage of the Signed Distance Function properties.

The numerical method developed for this thesis project is validate through a qualitative comparison with results available in literature, in particular the flow over a cylinder is tested.

This page is intentionally left blank

Sommario

Flussi turbolenti e flussi multifase ricoprono un ruolo fondamentale nella progettazione in ambito ingegneristico e non solo. La loro investigazione oggi si basa principalmente su tecniche di simulazione numerica, riducendo tempi, costi e complessità rispetto ad indagini sperimentali.

In questo progetto di tesi si considerano flussi multifase in cui la fase principale è un generico fluido e la fase secondaria sono delle particelle solide. Sono state ipotizzate condizioni diluite e pertanto viene utilizzato un modello di interazione fluido-particella a singola via.

Il flusso è simulato tramite tecniche di Simulazione Numerica Diretta (DNS) implementate nel codice *open-source* CaNS. Al fine di simulare geometrie complesse si ricorre a tecniche di *Immersed Boundary Method* (IBM), in particolare viene adottato un IBM tramite equazione Iconale e *Signed Distance Function*.

Il moto delle particelle viene simulato parallelamente al flusso, risolvendo le equazioni del moto. Oltre al moto delle particelle viene modellata anche la collisione di queste su pareti rigide modellate tramite il metodo IBM e grazie alle proprietà della *Signed Distance Function*.

Il metodo numerico sviluppato per il progetto di tesi viene validato tramite confronto qualitativo con i risultati disponibili in letteratura, in particolare viene testato il flusso attorno ad un cilindro.

This page is intentionally left blank

Contents

List of Figures

List of Tables

Acronyms

BC Boundary Condition. 17, 29

CaNS Canonical Navier-Stokes. 3, 38, 39, 49, 57

DNS Direct Numerical Solution or Direct Navier-Stokes. 9–12, 37, 57

ESA European Space Agency. 1

FD Finite Difference. 25

FE Finite Element. 25

FFT Fast Fourier Transform. 38

FV Finite Volume. 25

IB Immersed Boundary. 31, 39, 40, 49

IBM Immersed Boundary Method. 3, 4, 17, 25, 26, 32–34, 37, 49, 57, IX

LES Large Eddy Simulation. 9, 10, 12, 58

LS Level-Set. 16, 17

N-S Navier-Stokes. 9, 10, 17, 58

RANS Reynolds-Averaged Navier–Stokes. 9, 10, 12, 13

SDF Signed Distance Function. 4, 32–34, 37, 39–42, 57, IX

VOF Volume Of Fluid. 16, 17

WM-LES Wall-Modelled Large Eddy Simulation. 58

This page is intentionally left blank

Introduction

The numerical simulation of multiphase flows is a key problem for many industrial applications like the prediction of erosion due to dust particles in jet engines [31], [33]. Another significant field of application is the study of atmospheric phenomena with the aim of modelling the atmosphere of planets like Mars, that is of fundamental importance for the future of space explorations.

Martian atmosphere has peculiar characteristics that distinguish it form the Earth atmosphere, starting from the very low density, and therefore a significantly larger ratio between rock dust density and atmosphere density. Furthermore, dust storm events on Mars are very vast and can cover large regions influencing the solar irradiation which is one of the mechanisms that permits dust particles lifting due to the strong temperature gradient [9].

Despite having wildly different atmosphere, Mars cloud patterns have been found to be surprisingly Earth-like, pointing to similar formation processes. Images from ESA Mars Express [10] reveal a particular phenomenon on Mars. They show that the martian dust storms are made up of regularly spaced smaller cloud cells, arranged like grains or pebbles. The same texture is also seen in clouds in Earth's atmosphere.

These familiar textures are formed by convection, whereby hot air rises because it is less dense than the cooler air around it. The type of convection observed here is called closed-cell convection, when air rises in the centre of small cloud pockets, or cells. The gaps of sky around the cloud cells are the pathways for cooler air to sink below the hot rising air.

On Earth, the rising air contains water which condenses to form clouds. The dust clouds imaged by Mars Express show the same process, but on Mars the rising air columns contain dust rather than water. The Sun heats dust-laden air causing

it to rise and form dusty cells. The cells are surrounded by areas of sinking air which have less dust. This gives rise to the granular pattern also seen in the image of clouds on Earth.

Some of these storms have been observed in 2019 and analyzed by Sánchez-Lavega et al. [28] which identified three main typologies of storms:

1. Textured dust storms with cellular or granular patterns, compact or organized in spiral systems as in **Figure 1**;
2. Irregularly shaped or filamentary;
3. Flushing arc-shaped dust storms.

All these morphologies have been observed at the edge of Mars North Polar Cap.

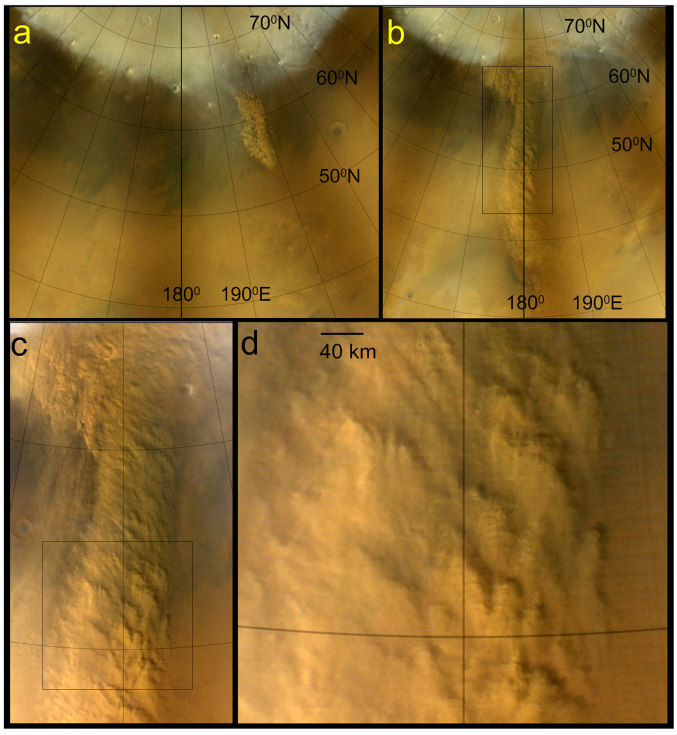


Figure 1: Example of a textured dust storm. a and b show the evolution in two consecutive sol, c and d show granular texture details with successive magnifications [28].

First type storms were tracked for several days following the formation, evolution, disappearing and reappearing. This storm occupied a total area of about

$4.3 \times 10^5\ km^2$. The first appear of this storm was formed by three parallel filaments aligned with the north-south direction. At smaller scales are distinguishable regularly distributed and smaller dust masses. This formation have then disappeared and reformed as a single-band dust storm with a prominent cell pattern but with less regularity. After this it has reformed as a spiral. The final appear of this storm showed a compact-textured dust storm. All this shapes of the storm have similar cellular patterns and the three filament form instead of a spiral form is triggered by small changes in the atmospheric conditions.

The second type of storm firstly appeared as an irregular dust storm and then evolved as filamentary storm with some more compact and denser areas with some granular texture that suggests a mechanism of cell formation similar to the one observed above.

The latter case showed an arc extending in a region outside the North Polar Cap and with no polar ices. In this case no granular patterns were observed. The curved arc shape suggests some development of cyclonic vorticity.

Different dynamical mechanisms and instabilities are involved in the formation of these storms. The primary disturbance is the meridional temperature gradient between the area still occupied by ice and the one where CO_2 has already evaporated. On a large scale, spiral and filamentary formations are the result of a baroclinic instability.

Another mechanism is dry convection at a local-scale in the Planetary Boundary Layer during the hours of maximum insulation heating. This convection produces the so called "Dust Devils" [32]. Dust Devils are convective vortices, similar to the ones observed on Earth, that on Mars reach higher altitudes due to low atmospheric pressure, large vertical temperature gradient (leading to thermal instability) and absence of moisture and vegetation. These plumes are able to transport dust particles up to altitudes of tens of kilometers triggering dust cloud formation.

Furthermore, the modelling of collision between particles and walls is a phenomena to take into account for the modelling of multiphase flows and this is the subject of this Thesis Project. The collision tracking method was implemented in the *CaNS* code to work with an Immersed Boundary Method (IBM).

The IBM is a simple and efficient approach to simulate complex geometries because it does not require the creation of complex mesh around the solid body,

but makes use of some changes to the equations near the body boundary. The boundaries of the immerse body could be identified with a Signed Distance Function (SDF), which properties could be also used to model the collision of the particle with the immersed boundary.

The thesis document begins with some theoretical background on turbulent flows and then numerical simulation in **chapter 1** and some basics about multiphase flows in **chapter 2**. Then various IBM method are described to introduce the one used for the thesis project (**chapter 3**). On **chapter 4** it is described the model implemented for the collision of the particles on a rigid body. Finally the results of the simulations and the validation of the model are presented in **chapter 5**.

Chapter 1

Turbulent Flows and Numerical Methods

Turbulent flows can be observed frequently in our everyday life: the water in a river, the smoke for a chimney or the water behind a boat are some examples. This turbulent phenomena are unsteady, irregular, random and chaotic. The turbulence appears on many different dimensional scales, if we look an image of a waterfall, we can observe eddies from the dimensions of the order of the waterfall size itself up to the resolution of the camera. An example is the turbulent jet in **Figure 1.1**, where many different dimensional scales are observable.

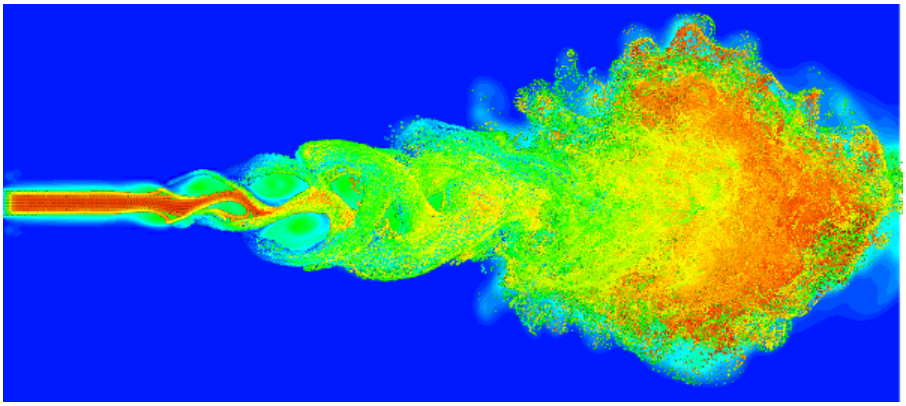


Figure 1.1: Development of a turbulent flow in a jet. Different scales can be observed [35].

Turbulent flows are important in industrial applications because their behavior is rather different from a laminar flow, e.g. turbulent flows can transport and mix

fluid much more effectively than laminar flows as demonstrates by the Reynolds experiment [27].

The ultimate object of studying turbulent flows is to obtain a theoretical description or a model of the phenomena. Due to the extreme complexity of turbulence, numerical solution techniques are of dominant importance [26]. Different approaches are possible for solving turbulent flows with different accuracy and computational cost.

1.1 Turbulent Flows

The main properties of turbulent flows are presented below and are in contraposition to laminar flows:

1. Absence of spatial symmetry and vorticity (**Figure 1.2**);
2. Multi-scale structure: eddies of different size (6 to 8 order of magnitude) influencing each other:
3. Turbulent flows appear as non-deterministic due to Deterministic Chaos Theory: small perturbation in a point of the domain could produce huge effects in a different position;
4. Unsteady regime with spatial irregularities;
5. Instantaneous fields have bigger fluctuations of velocity when increasing the Reynolds number due to the presence of smaller vortexes.

1.1.1 Energy Cascade and Kolmogorov Theory

The turbulent energy is associated to the eddies of a turbulent flow, but having multi-scale eddies, it is important to define the distribution of the energy among the different scales.

The first theory was proposed by Richardson which theorised the energy cascade in 1922: biggest eddies transfer energy to the smallest, so the energy is injected in the system by the biggest eddies and then dissipated by the smallest. This theory

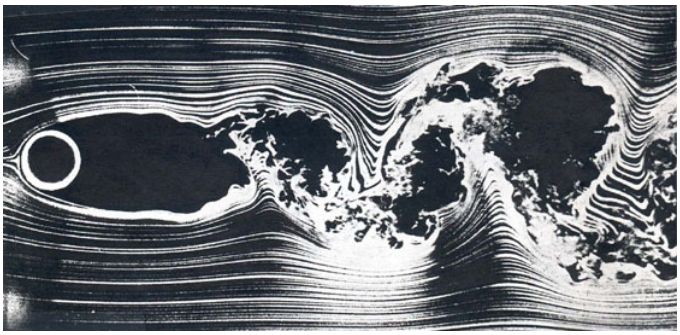


Figure 1.2: Turbulent flow around a cylinder. There is no symmetry with respect to the streamwise axis of the cylinder [7].

has been formalised in 1941 by Kolmogorov (K41 Theory) [18], [17] and the main results are presented below.

Let us consider a body with characteristic dimension L_0 and immersed in a flow with velocity U_0, viscosity ν and Reynolds number $Re = \frac{U_0 L_0}{\nu} >> 1$. The dimension l represents the size of the eddy. The biggest eddies have size $l_0 \sim L_0$, the smallest ones have size η (**Figure 1.3**).

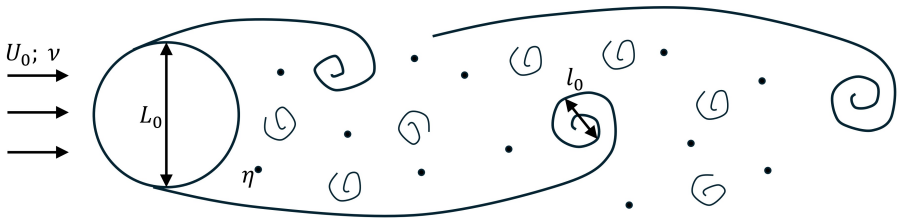


Figure 1.3: Schematic representation of the scales involved in a turbulent flow.

First it can be state that the macro-scales $l \sim l_0$ maintain dependence from the geometry of the problem and the direction of the flow, instead, the micro-scales have universal characteristics with homogeneity and isotropic properties.

Considering the biggest scales with dimension $l_0 \sim L_0$, from experimental results is known that $u_{l_0} \simeq 0.1 \div 0.3 \cdot U_0 \sim U_0$, so

$$Re_{l_0} = \frac{u_{l_0} l_0}{\nu} \sim \frac{U_0 L_0}{\nu} >> 1 \tag{1.1}$$

and the energy flux is:

$$\frac{k_{l_0}}{\tau_{l_0}} \sim \frac{u_{l_0}^3}{l_0} \sim \frac{U_0^3}{L_0} \sim \varepsilon \tag{1.2}$$

where k_l is the kinetic energy and τ_l is the characteristic time of the scale l.

Hypothesis zero: For $Re >> 1$ and eddy scales $l << l_0$ the turbulent flow is locally homogeneous and isotropic. The statistics are universal and they depend at most on ε and ν.

First hypothesis of similarity: The smallest scales η depend uniquely on both ε and ν.

As result of this hypothesis it is possible to obtain expressions for the length η, the characteristic speed u_η and the characteristic time τ_η by performing a dimensional analysis:

$$\eta = \left(\frac{\nu^3}{\varepsilon}\right)^{\frac{1}{4}} \tag{1.3}$$

$$u_\eta = (\nu\varepsilon)^{\frac{1}{4}} \tag{1.4}$$

$$\tau_\eta = \left(\frac{\nu}{\varepsilon}\right)^{\frac{1}{2}} \tag{1.5}$$

From these relations it is possible to calculate the Reynolds number:

$$Re_\eta = \frac{u_\eta \eta}{\nu} = 1 \tag{1.6}$$

From this result it is clear that at the smallest scales the effect of viscosity is as relevant as the inertia effects. Evaluating the energy dissipated it is possible to find that:

$$\varepsilon_\eta \sim \varepsilon \tag{1.7}$$

and therefore all the energy is dissipated at these scales, called also Kolmogorov dissipative scales.

Second hypothesis of similarity: The scales $\eta << l << l_0$ are universal, but do not depend on ν, thus depend only on ε.

Performing the dimensional analysis also for these inertial scales it is possible to obtain:

$$u_l = (\varepsilon l)^{\frac{1}{3}} \tag{1.8}$$

$$\tau_l = \left(\frac{l^2}{\varepsilon}\right)^{\frac{1}{3}} \tag{1.9}$$

As before it is possible to evaluate the Re:

$$Re_l = \frac{u_l l}{\nu}\left(\frac{l}{\eta}\right)^{\frac{4}{3}} \tag{1.10}$$

that is a high value and increases with l, so the inertial effects are dominant on the viscosity. The energy transported by the inertial scales can be evaluated as:

$$\varepsilon_l \sim \frac{k_l}{\tau_l} \sim \varepsilon \tag{1.11}$$

therefore all the energy is transported by the inertial scales.

To summarize, all the energy introduced by the biggest scales is transferred by the inertial scales to the Kolmogorov scales where it is dissipated as schematise in **Figure 1.4**.

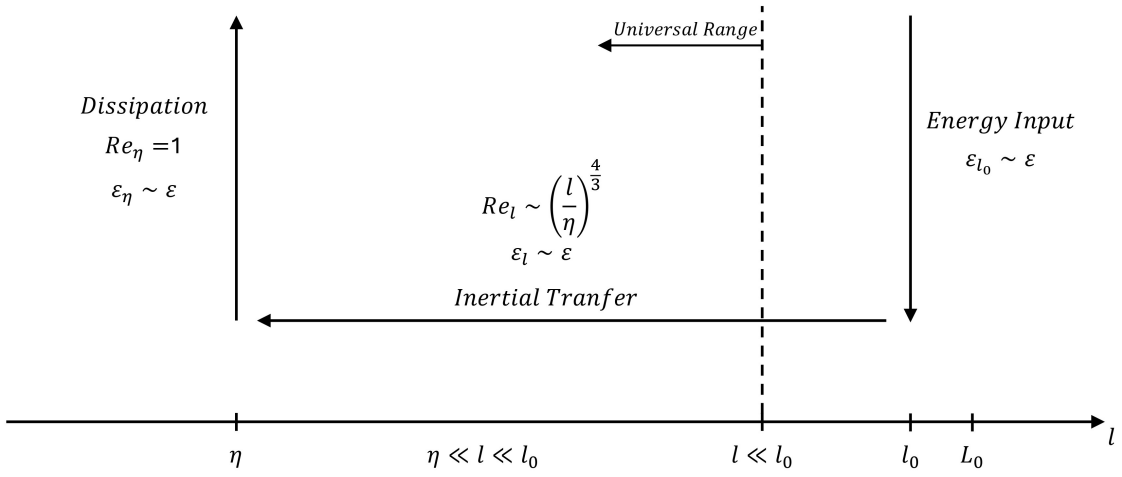


Figure 1.4: Energy transfer through the different eddies sizes.

1.2 Numerical Techniques for Turbulent Flows

As mentioned before, turbulent flows are extremely important in industrial applications, therefore it is necessary to find some methodologies to evaluate all the properties of the flow.

In the past turbulent flows were studied with an experimental approach, but with the need of higher detail level and reduction of costs and time, numerical methods play an important role.

The numerical solution of turbulent flows can be performed following two main approaches: the direct solution of the Navier-Stokes (N-S) equations with DNS or the modeling of the turbulence in LES and RANS [12].

In the **DNS** approach the N-S equations are solved directly on a 3D mesh with a time-depending simulation. No models for turbulence are required, but the computational cost is high.

RANS techniques resolve the averaged N-S equations and require a model to describe the Reynolds stress tensor in order to close the problem. The output of these simulations are only the average field, but the simulation time is order of magnitude lower than DNS.

LES methods are intermediate between RANS and DNS. The biggest eddies are directly simulated, instead the smallest and their effect on the system are modelled.

All these methods have specific performances and characteristics. Some aspects can be considered for choosing one method rather than the others [26] for the specific application:

- Level of description, e.g. instantaneous fields or average fields;
- Completeness, e.g. simulation parameters are or not flow dependant;
- Cost and ease of use;
- Range of applicability;
- Accuracy.

Some more details about these methodologies will be provided in the following sections.

1.2.1 Direct Numerical Solution

The most accurate approach is to solve the Navier-Stokes equation without any assumption and the numerical discretization in time and space as only one approximation. If we consider incompressible flow, the equations to solve are:

$$\begin{cases} \vec{\nabla} \cdot \vec{U} = 0 \\ \\ \dfrac{D\vec{U}}{Dt} = -\dfrac{\vec{\nabla} p}{\rho} + \nu \nabla^2 \vec{U} + \vec{f} \end{cases} \tag{1.12}$$

In Direct Numerical Solution or Direct Navier-Stokes (DNS), since no turbulence model is used, the simulation domain must be as large as the biggest eddies. At the

same time, a good simulation must capture all the kinetic energy dissipation that happens at the smallest scales, so the computational grid must be on the order of the Kolmogorov scale. In particular the space discretization requirement is $\Delta \leq 2\eta$ [12].

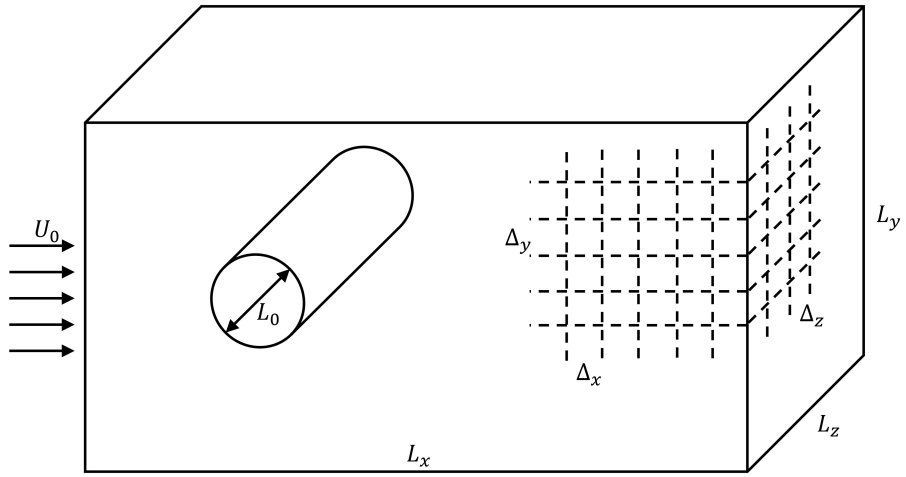


Figure 1.5: Example of a simulation domain for a cylinder immersed on a fluid flow, main dimensions and grid discretization are highlighted.

It is immediately clear that the computational requirement of a DNS is huge. Let us consider a simple domain like the one on **Figure 1.5**. The domain has dimensions $L_x \sim L_y \sim L_z \sim L_0$ in the three directions and the discretizations are $\Delta_x \sim \Delta_y \sim \Delta_z \sim \eta$. Considering the results of the K41 theory, the number of spatial subdivision along a direction can be evaluated as:

$$N_x = \frac{L_x}{\Delta_x} \propto \frac{L_0}{\eta} \propto Re^{3/4} \tag{1.13}$$

Along the three directions, the total number of subdivisions is given by:

$$N_{TOT} = N_x \cdot N_y \cdot N_z \propto N_x^3 \propto Re^{9/4} \tag{1.14}$$

Furthermore, the time discretization must be considered. If we call T the simulation time and Δt the time step, we can estimate the required number of time steps:

$$N_{\Delta t} = \frac{T}{\Delta t} \propto \frac{T_0}{\tau_\eta} \propto Re^{1/2} \tag{1.15}$$

As a consequence, the total computational cost is given by:

$$Computational\ Cost = N_{TOT} \cdot N_{\Delta t} \propto Re^{11/4} \approx Re^3 \tag{1.16}$$

This shows that the DNS approach is suitable only for small problems with low Reynolds Number.

1.2.2 Large Eddy Simulation

In Large Eddy Simulation (LES), the larger turbulent structures are directly resolved, instead the smallest scales are modelled. In terms of computational cost this method lies between DNS and RANS, but is more reliable than RANS for flows with significant unsteadiness.

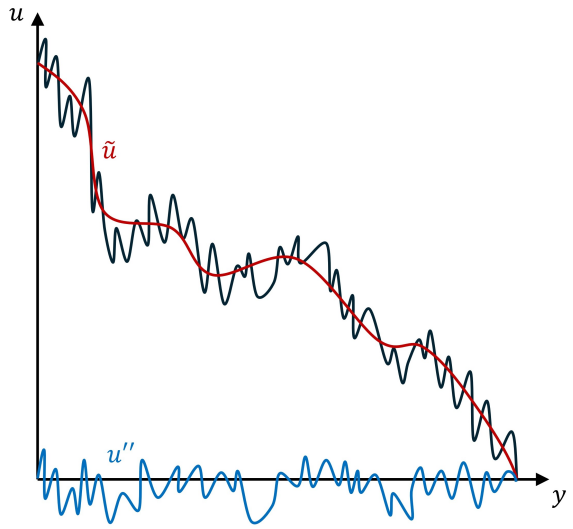


Figure 1.6: Velocity filtering in LES. In red the filtered velocity, in blue the residual velocity.

The basic concept of LES is the filtering of the velocity to eliminate the high frequency fluctuations and the decomposition of the velocity in a filtered velocity $\widetilde{u}$ and a residual velocity u'' (**Figure 1.6**) so that $u = \widetilde{u} + u''$. Two types of filtering can be implemented:

1. Implicit filtering: the grid itself is the spatial filter. The Navier-Stokes equations are directly resolved and then some corrections are applied to take into account the sub-grid scales that cannot be resolved;

2. Explicit filtering: a low pass filter (like Top-Hat or Gaussian filters) is applied to the Navier-Stokes equations and then a closure model is implemented to close the filtered LES.

The filtered Navier-Stokes equations are:

$$\begin{cases} \vec{\nabla} \cdot \vec{\widetilde{u}} = 0 \\ \\ \dfrac{D\vec{\widetilde{u}}}{Dt} = -\dfrac{\vec{\nabla}\widetilde{p}}{\rho} + \vec{\nabla} \cdot \left(2\nu \overset{\rightrightarrows}{\widetilde{E}}\right) - \vec{\nabla} \cdot \overset{\rightrightarrows}{\tau^R} \end{cases} \tag{1.17}$$

where $\tau_{ij}^R = \widetilde{u_i u_j} - \widetilde{u}_i \widetilde{u}_j$ is the residual stress tensor that is unknown and must be modelled. Different models are available like the Smagorinsky model.

1.2.3 Reynolds-Averaged Navier-Stokes

The Reynolds-Averaged Navier–Stokes (RANS) approach is the one with the smallest computational cost, but is less accurate and produces only averaged fields. In this approach the RANS equations are resolved, these equations are obtained by applying a Reynolds averaging to the Navier-Stokes equations. The velocity can be divided in the average and fluctuation components following the Reynolds decomposition: $u(x,t) = U(x,t) + u'(x,t)$. The RANS equations are:

$$\begin{cases} \vec{\nabla} \cdot \vec{U} = 0 \\ \\ \dfrac{D\vec{U}}{Dt} = -\dfrac{\vec{\nabla}p}{\rho} + \vec{\nabla} \cdot \left(2\nu \overset{\rightrightarrows}{E}\right) - \vec{\nabla} \cdot < \vec{u'}\vec{u'} > \end{cases} \tag{1.18}$$

where $< \vec{u'}\vec{u'} >$ is the Reynolds stress tensor, which terms are unknown and need to be model to close the problem.

Many models are available and a lot of them are based on the Boussinesq hypothesis:

$$- < u_i' u_j' >= -\frac{2}{3} k \delta_{ij} + 2\nu_T E_{ij} \tag{1.19}$$

Now a model for the turbulent viscosity ν_T must be found. Some of the models based on this assumption are the $Spalart-Allmaras$ (1 eq.), $k-\varepsilon$, $k-\omega$, $k-\omega-SST$ (2 eqs.) and $k-kl-\omega$ (3 eqs.).

Another approach is the Reynolds Stress model. In this case the exact transport equation for the Reynolds stress is obtained from the Navier-Stokes equations. This equation has some unknowns that need to be modelled, so 5 additional equations need to be solved.

Chapter 2

Multiphase Flows

In industrial applications many systems involve multiphase media, being them the combination of liquids and gases, non-miscible liquids or liquids and solids. Some examples are vapor-water mixture in cooling systems in thermal facilities, the combustion chamber of a liquid fuel rocket engine that contains a mixture of vaporizing droplets and combusting gases or many natural phenomena like clouds on Earth or dust clouds on Mars.

Typically the overall medium behaves as a fluid [1] also in presence of a solid phase, so to study these flows it is possible to use the same approach of classic fluids. Furthermore, these flows generally have random nature and velocity fluctuation similar to turbulent flows. Therefore, all this aspects are the base for studying multiphase flows.

The focus aspect of multiphase flows is the definition of a model to describe the behaviour of these flows. There are three approaches to obtain these models [2]:

- Experimentally: laboratory experiments with appropriate measurements;
- Theoretically: using mathematical equations;
- Computationally: solving the problem thanks to the computational capabilities of modern computer technologies.

Frequently experiments are too expensive and the laboratory scale is too different from the real case study that the results may be of low quality. This explains why the theoretical/computational approaches are dominant in understanding multiphase flows physics.

2.1 Modelling Multiphase Flows

A multiphase flow is a system in which multiple materials are involved. These materials have different state of matter, different thermodynamic state or different compounds which cannot mix. Each of this material is referred as a phase of the system. The different phases can interact with each other exchanging mass, momentum and energy. Two kinds of phases can be distinguished: a carrier phase that is the most abundant phase and a dispersed phase that is less abundant and is transported by the carrier.

Different techniques have been developed to numerically solve different typologies of multiphase flows like solid particle laden flows, droplets laden flows, immiscible dense system and dilute rather than dense systems. Two main categories of numerical methods can be identified:

- Hybrid Eulerian-Lagrangian methods: Navier-Stokes equations are solved for the carrier, Lagrangian equations are used to evolve the particle state;
- Eulerian methods: Navier-Stokes equations are solved for both the carrier and the secondary phase

Two of the main Eulerian methods used for immiscible dense fluids are the Volume Of Fluid (VOF) and Level-Set (LS) methods. In both of them the Navier-Stokes equations are solved on a fixed computational grid to evolve the motion of all the phases. The momentum equation takes into account a force to model the surface tension at the phase interface as function of the interface curvature and the surface tension coefficient. It is clearly necessary to take into account the different properties of the phases and to know the position and curvature of the interface.

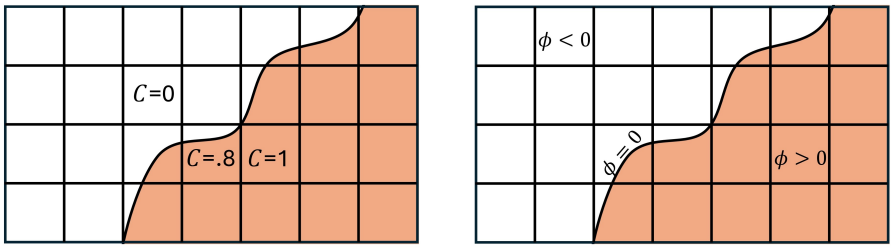


Figure 2.1: Graphic interpretation of the Volume Of Fluid method (left) and Level-Set method (right).

The VOF method evolves an advection equation for the volume fraction C_s:

$$\frac{\partial C_s}{\partial t} + \vec{u} \cdot \vec{\nabla} C_s = 0 \tag{2.1}$$

In each cell of the computational grid the volume fraction is computed (**Figure 2.1**) and the intensive properties are evaluated as volume fraction average on each involved phase. The interface location must be reconstructed. This is the most difficult aspect of this method. The VOF method is more accurate in terms of mass conservation.

The LS method simplifies the interface description by introducing an auxiliary field ϕ defined as positive in a phase, negative in the other and zero at the interface (**Figure 2.1**). An advection equation for this field is evolved:

$$\frac{\partial \phi}{\partial t} + \vec{u} \cdot \vec{\nabla} \phi = 0 \tag{2.2}$$

With this method the interface location is immediately defined, so LS is more accurate in reconstructing the interface.

Among the Hybrid Eulerian-Lagrangian methods, Immersed Boundary Method (IBM) and Point-Particle Approximation are the state of the art for solid particles.

In the IBM the N-S equations are solved on a fixed grid describing the carrier flow. The momentum equation is corrected with a fictitious force f_{IBM} that is applied in the proximity of the particle to impose the flow motion to follow locally the velocity of the particle. In this way the non-slip and non-penetration BCs are implemented and it is not necessary to have the particle surface coinciding with the computational grid (as for classical BC by setting $\vec{u} = 0$ on the node of the particle surface).

The Point-Particle Approximation is the model used for implementing the particle tracking method object of this thesis and will be discussed in the following paragraph 2.3.

2.2 Particle Laden Flows

In particle laden flows (also valid for droplets laden flows), it is possible to distinguish four types of interaction among the involved phases. In particular we can identify four coupling mechanisms (**Figure 2.2**):

1. The carrier phase forces the particles in term of mass, momentum and energy;

2. The particles force the carrier phase;

3. Particles interacts with other particles via the carrier phase, e.g. lubrication forces, wakes;

4. Direct interaction between particles, like collision or adhesion.

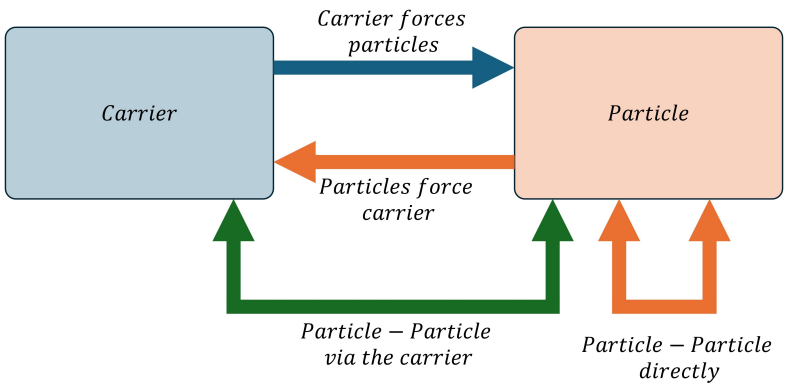


Figure 2.2: Scheme of the interaction between carrier and dispersed phase. The arrows highlight the four possible couplings between the carrier and the particles.

These couplings are more or less relevant depending on the mass and volume fraction of the dispersed phase. Volume fraction is defined as:

$$\phi = \frac{V_d}{V} \tag{2.3}$$

where V_d is the volume occupied by the dispersed phase and $V = V_d + V_c$ is the total volume of the system, that is the sum of the dispersed phase volume and the carrier volume. In the same way we can define the mass fraction:

$$\psi = \frac{m_d}{m} \tag{2.4}$$

and again m_d is the mass of the dispersed phase and $m = m_d + m_c$ is the total mass of the system.

According tho mass and volume fraction, three main scenarios could occur (**Figure 2.3**):

1. One way coupling: it happens in very diluted conditions such that particle-particle interaction and the effects of the disperse phase on the carrier are negligible. Only the carrier can influence the particles (only coupling 1);

2. Two way coupling: it applies at diluted conditions. Due to the low volume fraction, the particle-particle interaction is negligible, but the higher mass fraction entails the need of considering the influence of the particles on the carrier (couplings 1 and 2);

3. Four way coupling: in dense regime the volume fraction is high enough to require to take into account the effect of the mutual interaction between particles (couplings 1, 2, 3 and 4).

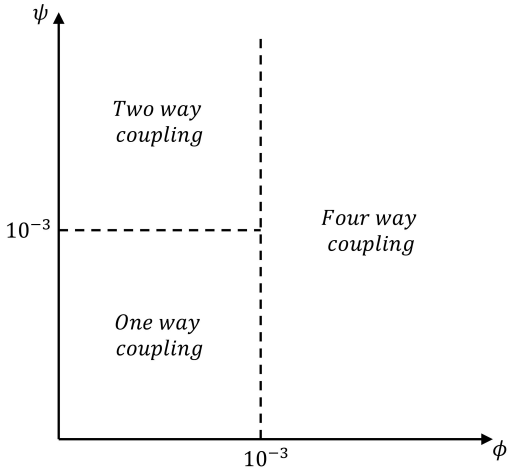


Figure 2.3: Approximate subdivision of the coupling typologies depending on mass fraction ψ and volume fraction ϕ.

2.3 Particle Equations of Motion

To discuss the motion of a particle in a fluid we can start considering the particle as a rigid sphere of radius r_p. We assume that from a macroscopic point of view, the finite size of the particle does not influence the flow. In a general case, also

including gravity force, the equation of motion of a particle can be written as [22]:

$$\frac{\overrightarrow{dU_p}}{dt} = -\frac{3}{4}\frac{C_D}{d_p}\left(\frac{\rho}{\rho_p}\right)\left|\overrightarrow{U_p} - \vec{U}\right|\left(\overrightarrow{U_p} - \vec{U}\right) + C_L\frac{\rho}{\rho_p}\left[\left(\overrightarrow{U_p} - \vec{U}\right) \times \omega\right] + \left(1 - \frac{\rho}{\rho_p}\right)\vec{g} \tag{2.5}$$

All the quantities with subscript p refer to particle properties, $d_p = 2 \cdot r_p$ is the diameter of the particle, ρ and ρ_p are the density of the carrier and of the particle, $\vec{U}$ and $\overrightarrow{U_p}$ are the velocities of the fluid and of the particle, ω is the vorticity of the fluid. Fluid properties are intended at the position of the particle.

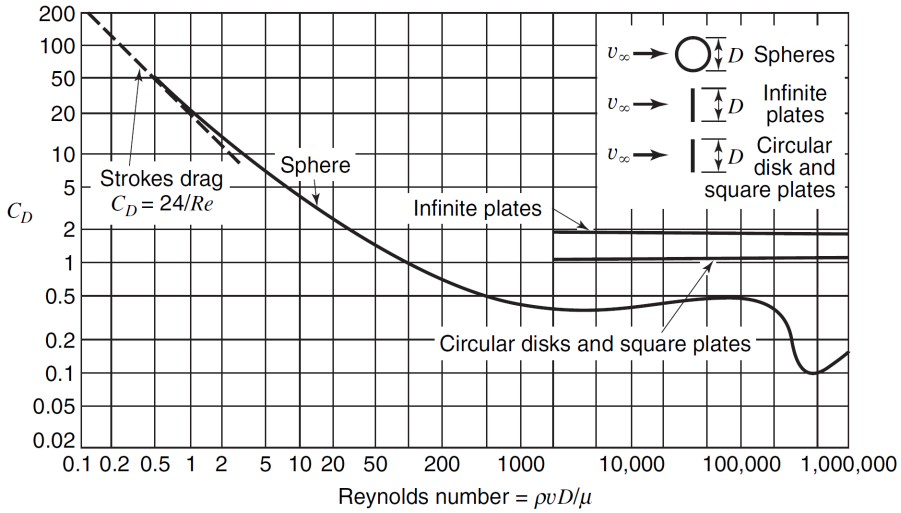


Figure 2.4: Drag coefficient C_D as a function of Reynolds number for different shapes [34].

The drag component can be estimated for the Stokes flow (laminar and at low Reynolds number) as [21]:

$$C_D = \frac{24}{Re_p} \tag{2.6}$$

To include also higher Reynolds number, a corrected Stokes drag coefficient can be used:

$$C_D = \frac{24}{Re_p}\left(1 + 0.15Re_p^{0.687}\right) \tag{2.7}$$

where the particle Reynolds number is defined as:

$$Re_p = \frac{d_p\left|\overrightarrow{U_p} - \vec{U}\right|}{\nu} \tag{2.8}$$

The lift force can be neglected for small particles, instead may be not negligible for bigger particles, so its estimation is reported. The lift force is calculated by means of the lift coefficient C_L that is a function of Reynolds number. We can evaluate C_L as [13]:

$$C_L = \begin{cases} C_{L_{McL}} = \left[5.816\left(\dfrac{Sr_p}{2Re_p}\right)^{0.5} - 0.875\dfrac{Sr_p}{2}\right]\dfrac{3}{4Sr_p}\dfrac{J(\varepsilon)}{2.255} & for\ Re_p < 1 \\ C_{L_{McL}}\dfrac{5-Re_p}{4} + C_{L_{KK}}\dfrac{Re_p - 1}{4} & for\ 1 < Re_p < 5 \\ C_{L_{KK}} = \left[K_0\left(\dfrac{Sr_p}{2}\right)^{0.9} + K_1\left(\dfrac{Sr_p}{2}\right)^{1.1}\right]\dfrac{3}{4Sr_p} & for\ Re_p > 5 \end{cases} \tag{2.9}$$

In **Equation 2.9** it is possible to note that the lift coefficient depends on the dimensionless parameter Sr_p:

$$Sr_p = \frac{\left|\left(\vec{U} - \vec{U_p}\right) \times \omega\right| d_p}{\left|\vec{U} - \vec{U_p}\right|^2} \tag{2.10}$$

The term $C_{L_{McL}}$ refers to the coefficient as evaluated by McLaughlin [23]. The function $J(\varepsilon)$ was introduced by McLaughlin to extend the validity of the equation also in the case that Re_p is not negligible and it depends on $\varepsilon = (Sr_p/Re_p)^{0.5}$. The term $C_{L_{KK}}$ is calculated as done by Kurose and Komori [19], where the coefficients K_0 and K_1 are function of Re_p as illustrated in **Table 2.1**. The lift coefficient is linearly interpolated between the McLaughlin value and the Kurose and Komori one in the range $1 < Re_p < 5$.

From now we introduce the hypothesis of small particle, so that the lift contribution can be neglected and also we assume no gravitational effects. Following this

Re_p	K_0	K_1
1	$4.815 \cdot 10^{-1}$	3.578
5	$-7.830 \cdot 10^{-1}$	1.746
10	$-9.408 \cdot 10^{-2}$	$1.886 \cdot 10^{-1}$
50	$-1.141 \cdot 10^{-1}$	$1.533 \cdot 10^{-1}$
100	$-1.823 \cdot 10^{-1}$	$1.242 \cdot 10^{-1}$
200	$-4.269 \cdot 10^{-1}$	$2.455 \cdot 10^{-1}$
300	-1.112	1.101
400	$-9.983 \cdot 10^{-1}$	$9.250 \cdot 10^{-1}$
500	$-6.926 \cdot 10^{-1}$	$5.305 \cdot 10^{-1}$

Table 2.1: Constant values for particle lift determination as presented by Kurose and Komori [19].

consideration, the **Equation 2.5** simplifies to:

$$\begin{aligned}
\frac{d\vec{U}_p}{dt} &= -\frac{3}{4}\frac{C_D}{d_p}\left(\frac{\rho}{\rho_p}\right)\left|\vec{U_p} - \vec{U}\right|\left(\vec{U_p} - \vec{U}\right) \\
&= -\frac{3}{4}\frac{24}{Re_p}\left(1 + 0.15Re_p^{0.687}\right)\frac{1}{d_p}\frac{\rho}{\rho_p}\left|\vec{U_p} - \vec{U}\right|\left(\vec{U_p} - \vec{U}\right) \\
&= -18\frac{\nu}{d_p\left|\vec{U_p} - \vec{U}\right|}\left(1 + 0.15Re_p^{0.687}\right)\frac{1}{d_p}\frac{\rho}{\rho_p}\left|\vec{U_p} - \vec{U}\right|\left(\vec{U_p} - \vec{U}\right) \\
\frac{d\vec{U}_p}{dt} &= \left(1 + 0.15Re_p^{0.687}\right)\left(\vec{U} - \vec{U_p}\right)\frac{18\mu}{d_p^2\rho_p}
\end{aligned} \tag{2.11}$$

In **Equation 2.11** we can identify the inverse of the momentum response time of the particle (valid for low Reynolds numbers):

$$\tau_p = \frac{d_p^2\rho_p}{18\mu} \tag{2.12}$$

The momentum response time is the time it takes for a particle or droplet to respond to a change in velocity. More specifically, the momentum response time is the time required for a particle, after being released from rest, to reach 63% of the free-stream velocity [11]. In other words, this parameter is a metric of the particle inertia.

To better describe the inertia effects of the particle it is necessary to consider also the characteristic time scale of the particle at a certain length scale, so another important non-dimensional parameter is the Stokes Number:

$$St = \frac{\tau_p}{\tau_l} \tag{2.13}$$

where $\tau_l = l/u_l$ is the characteristic time of the flow, l and u_l are the characteristic length and velocity at scale l.

If $St \ll 1$, the response time of the particles in the flow is much less than the characteristic time. Therefore, the particles will have enough time to respond to changes in flow velocity and the velocities of the particles and fluid will be nearly equal. If $St \gg 1$, the particles will not have enough time to respond to the velocity change of the fluid; furthermore, the particle velocity will not be noticeably affected by the change in the fluid velocity.

Recalling what was discussed above, the one way coupling is adopted in the thesis project since dispersed flow is assumed. A possible way to define if the flow is dense or dispersed is to consider the average time between particle-particle collision τ_{pp} [8]. If:

$$\frac{\tau_p}{\tau_{pp}} < 1 \tag{2.14}$$

the flow is dispersed, therefore particle motion is dominated by fluid forces and the particles have enough time to respond to fluid forces before the next collision. Instead, if:

$$\frac{\tau_p}{\tau_{pp}} > 1 \tag{2.15}$$

the flow is dense, and the particles do not have enough time to respond to fluid forces before the next collision.

Finally, the position of the particle is obtained integrating the velocity:

$$\frac{d\vec{x}_p}{dt} = \vec{U}_p \tag{2.16}$$

Chapter 3

Immersed Boundary Method

In many engineering applications are involved complex geometries that are not easily describable with Cartesian grids. When the geometry is regular, the grid choice is simple because the grid lines can easily follow the direction given by the boundary. If the geometry is complicated, the choice of the gird is not predictable a priori and depends on the adopted discretization method, e.g. an algorithm designed for curvilinear orthogonal grids cannot be used for non-orthogonal grids [12]. Some methods can be used to simplify the simulation of complex geometries, among these a valuable approach is the Immersed Boundary Method (IBM).

Generally speaking, an Immersed Boundary Method is a method that simulates viscous flows with immersed boundaries on a computational grid that do not conform to the shape of these boundaries. Basically, one first has to identify cells which are cut by wall boundaries (and possibly also their immediate neighbors) and then has to enforce the correct wall boundary conditions at correct locations; this leads to fix the velocity either in the cut cell, or in the ghost-cell inside of the body, such that the velocity profile on a certain number of nodes gives the specified wall velocity where it crosses the wall boundary. To better understand the operating principle of an IBM, let us consider the simple example of **Figure 3.1** [24].

The **conventional** approach (**Figure 3.1a**) adopts a grid that conforms to the body. First, a surface grid is generated to describe the body boundary Γ_b, then this is used as boundary condition for generating the grid in the fluid domain Ω_f. At the end the problem is solved with one among Finite Difference (FD), Finite Volume (FV) or Finite Element (FE) method.

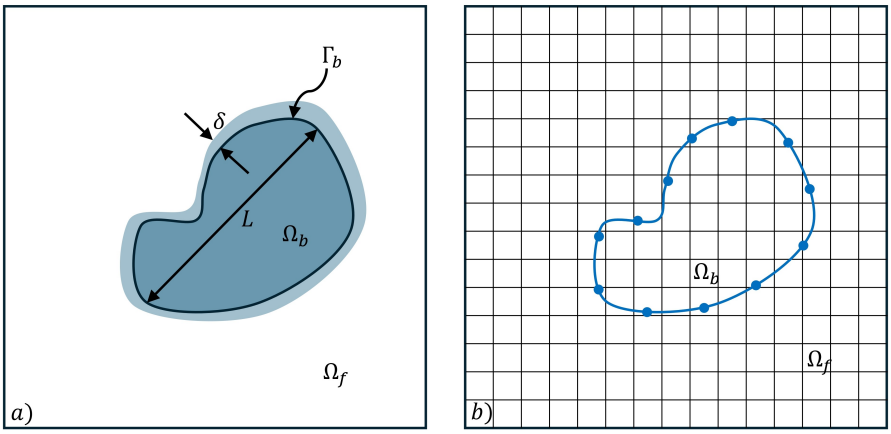


Figure 3.1: a) Schematic showing a generic body past which flow is to be simulated. b) Schematic of body immersed in a Cartesian grid on which the simulation is performed [24].

In the **IBM** approach we consider a Cartesian grid non-conformal to the body (**Figure 3.1b**) generated with no concern about the presence of the body. Since the grid has no boundary on the solid wall of the body, a correction is applied at the equations in the vicinity of the body in order to take into account the boundary conditions.

An IBM is advantageous in handling grid generation for complex geometries compared to a traditional body conformal grid. The simulation of moving bodies is also easier thanks to IBM since it is not necessary to recreate the entire grid at every time step.

3.1 Boundary Condition Implementation

The key aspect of IBM is the definition of the boundary conditions. Let us consider the Navier-Stokes equations:

$$\begin{cases} \vec{\nabla} \cdot \vec{U} = 0 \\ \\ \dfrac{\partial \vec{U}}{\partial t} + \vec{U} \cdot \vec{\nabla}\vec{U} = -\dfrac{\vec{\nabla} p}{\rho} + \nu \nabla^2 \vec{U} \end{cases} \tag{3.1}$$

$$with \quad \vec{U} = \vec{U_\Gamma} \quad on \quad \Gamma_b \tag{3.2}$$

We can compact the notation as:

$$\mathcal{L}\left(\underline{U}\right) = 0 \quad on \quad \Omega_f \tag{3.3}$$

$$\underline{U} = \underline{U}_\Gamma \quad on \quad \Gamma_b \tag{3.4}$$

where $\underline{U} = \left(\vec{U}, p\right)$ and $\mathcal{L}$ refers to the Navier-Stokes equations. The boundary conditions are considered by adding a forcing term to **Equation 3.3**. Two main methodologies can be adopted to implement boundary conditions:

Continuous forcing approach: The forcing term $\underline{f_b}$ is included in **Equation 3.3** so that $\mathcal{L}\left(\underline{U}\right) = \underline{f_b}$ and this equation is solved for all the domain $(\Omega_f + \Omega_b)$. Note that the forcing term $\underline{f_b} = \left(\overrightarrow{f_m}, f_p\right)$ applies to both momentum and pressure. The obtained equations are then discretized on the grid.

Discrete forcing approach: The **Equation 3.3** is discretized on the grid and then a correction term is applied in the cells near the immersed boundary being $[L']\{\underline{U}\} = \{\underline{r}\}$ with $[L']$ the modified discrete operator and $\{\underline{r}\}$ the known term associated to boundary conditions.

3.2 Continuous Forcing Approach

The continuous forcing approach works well for elastic boundaries, instead, rigid body presents some criticality. Moreover these methods solve the Navier-Stokes equations also inside the immersed body and this means increasing the computational cost more and more at increasing Reynolds number.

3.2.1 Flows with Elastic Boundaries

The Navier-Stokes equations are solved on the fixed Cartesian grid. The immersed boundary is represented as a set of elastic fibers [25]. The position of these fibers is expressed in a Lagrangian manner in some nodes k:

$$\frac{\partial \overrightarrow{X_k}}{\partial t} = \vec{U} \tag{3.5}$$

The immerse boundary effect on the fluid in managed by transmitting the fiber stress to the surrounding fluid with a forcing term in the momentum equation:

$$\overrightarrow{f_m}(\vec{x},t) = \sum_k \overrightarrow{F_k}(t) d \left| \vec{x} - \overrightarrow{X_k} \right| \tag{3.6}$$

where d is a distribution function (different versions have been tested and can be found in literature) and $\vec{F}$ are the stresses on the elastic fibers. Since the position $\overrightarrow{X_k}$ does not coincide with the Cartesian grid, the forcing term is distributed on the surrounding nodes (**Figure 3.2**).

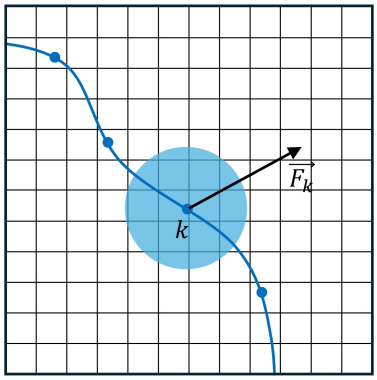


Figure 3.2: Transfer of the forcing from boundary point k to surrounding fluid nodes. The forcing is distributed in the shaded region [24].

3.2.2 Flows with Rigid Boundaries

To describe rigid boundaries, a first possibility is to follow the arguments done for elastic boundary, but assuming really high stiffness. A second approach is to consider the boundary linked to an equilibrium position with a spring [20] which force is:

$$\overrightarrow{F_k}(t) = -k \left(\overrightarrow{X_k} - \overrightarrow{X_k}^e(t) \right) \tag{3.7}$$

where k is the spring constant and $\overrightarrow{X_k}^e$ is the equilibrium position of the boundary Lagrangian point. High values of k are prone to stability problem, instead, low values of k cause an undesired residual elasticity of the boundary.

A generalization of the above is implemented modelling the forcing term as [14]:

$$\overrightarrow{F_k}(t) = \alpha \int_0^t \vec{u}(\tau)\, d\tau + \beta \vec{u}(t) \tag{3.8}$$

where α and β are chosen to represent correctly the boundary conditions. From a physical point of view **Equation 3.8** represents a damped oscillator.

3.3 Discrete Forcing Approach

Discrete forcing methods apply the boundary conditions directly on the discretized equations. These methods best reproduce the immersed boundary and do not introduce stability issues. Since Navier-Stokes equations and the equation for the boundary are uncoupled, there is no need to solve the flow field inside the immerse boundary, saving computational cost.

3.3.1 Indirect Boundary Conditions Imposition

Imposing boundary conditions indirectly permits to extract the forcing term from a first attempt solution of the problem [15]. We can consider the discretized solution of Navier-Stokes equation ignoring the presence of the immersed boundary: $[L]\{\underline{U}^*\} = 0$. This equation is solved at each step and $\{\underline{U}^*\}$ in the predicted velocity. The forcing term is defined as:

$$\{\underline{f}'_b\} \approx \{\underline{r}\} + [L]\{\underline{U}^*\} - [L']\{\underline{U}^*\} = \{\underline{r}\} - [L']\{\underline{U}^*\} \tag{3.9}$$

where:

$$\{\underline{r}\} = \{\underline{U}_\Gamma\} d \left|\overrightarrow{X_k} - \vec{x}_{i,j}\right| \tag{3.10}$$

$$[L'] = [L] + ([I] - [L])\, d \left|\overrightarrow{X_k} - \vec{x}_{i,j}\right| \tag{3.11}$$

This approach is advantageous since no input parameters need to be specified by the user. On the other hand, the forcing still extends to the fluid region due to the presence of the distribution function d and this method is sensible to the numerical solution technique adopted.

3.3.2 Direct Boundary Conditions Imposition

Direct BC imposition is advantageous at high Reynolds number when a good boundary layer estimation is desired, so when local accuracy is of fundamental importance. This is done by modifying the computational stencil near the immersed

boundary. Different methods have been developed and these can be divided into two main families discussed below.

Ghost-Cell Finite-Difference Approach

First, a Ghost cell is a cell within the immersed body that has at least a neighbour cell in the fluid domain (cell G in **Figure 3.3**). For each ghost cell, an interpolation scheme is implemented to include the immersed boundary. For example a bilinear (or trilinear in 3D) scheme for a generic variable ϕ is:

$$\phi = C_1 x_1 x_2 + C_2 x_1 + C_3 x_2 + C_4 \tag{3.12}$$

where the four C_i coefficients depend on the value of ϕ at nodes F_1, F_2 and F_3 in the fluid domain and on B_2 (or B_1) on the immersed boundary. Note that in **Figure 3.3**, points P_1 and P_2 are points on the boundary with the x and y coordinate of the ghost node, B_1 is a node in the midway between P_1 and P_2 and $B2$ in the node along the normal direction from G to the boundary.

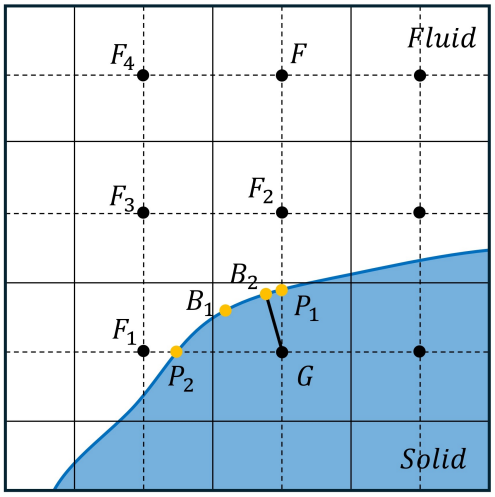


Figure 3.3: Representation of the characteristic points in the vicinity of the immersed boundary for the Ghost-cell approach [24].

For higher accuracy, in particular at high Reynolds with sparse grid, higher-order interpolation schemes should be adopted like, for example, a linear-quadratic interpolation respectively in the tangential and normal direction to the boundary.

Independently form the interpolation scheme used, the value at the ghost node is given by:

$$\sum_i \omega_i \phi_i = \phi_G \tag{3.13}$$

where ω_i are known coefficients function of the geometry. This equation is the one applied in the vicinity of the immersed boundary and simultaneously solved with the Navier-Stokes equations for the fluid.

Cut-Cell Finite-Volume Approach

The cut-cell approach is necessary to guarantee a strict observance of conservation laws in the vicinity of the immersed boundary. The control volume is created following these steps (see **Figure 3.4-Left** as reference):

1. Cells of the Cartesian grid that are cutted by the IB are identified to determine the intersection between these;
2. Cutted cells which center is within the fluid are reshaped discarding the portion lying in the solid;
3. Pieces of cut cell which centre is in the solid are absorbed by the neighbouring cells.

As a result a trapezoidal control volume is obtained.

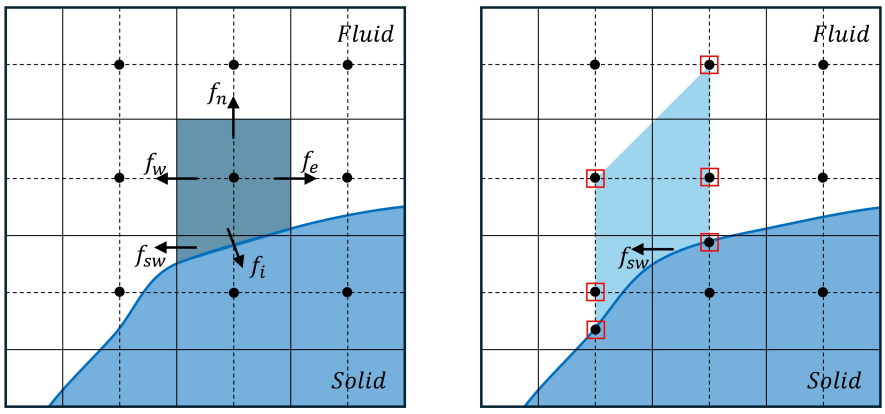


Figure 3.4: Left: Finite volume cell near the immersed boundary with face fluxes highlighted. Right: interpolation stencil for f_{sw}, red boxes mark the six nodes for the stencil [36].

The finite-volume discretization scheme need the knowledge of the flux integrals (mass, convective and diffusive) and pressure gradients on cell faces. The approach [36] is to express a generic variable ϕ with a two-dimension polynomial interpolation and evaluate fluxes f based on this interpolation. For example the flux f_{sw} in **Figure 3.4-Right** can be evaluated as:

$$\phi = C_1 x_1 x_2^2 + C_2 x_2^2 + C_3 x_1 x_2 + C_4 x_1 + C_5 x_2 + C_6 \tag{3.14}$$

where C_i are unknown coefficients function of ϕ in the six stencil nodes. An equation similar to **Equation 3.13** is solved for f_{sw}. The same process can be used to evaluate f_e and f_i.

3.4 Signed Distance Function

The Signed Distance Function (SDF) is a mathematical instrument that is useful to identify the immersed boundary and will be of fundamental importance in the IBM method discussed in **section 3.5**.

Let us consider a body Ω with boundary $\partial\Omega$ immersed in a generic domain. The Signed Distance Function is the orthogonal distance of a given point $\vec{x}$ to the boundary of a subdomain Ω in a generic domain, with the sign determined by whether or not $\vec{x}$ is in the interior of Ω. Hence the SDF ϕ is defined as [37]:

$$\phi(\vec{x}) = \begin{cases} d(\vec{x}) & \vec{x} \in \Omega \\ 0 & \vec{x} \in \partial\Omega \\ -d(\vec{x}) & \vec{x} \notin \Omega \end{cases} \tag{3.15}$$

where $d(\vec{x})$ is the minimum distance of $\vec{x}$ form $\partial\Omega$:

$$d(\vec{x}) = \min_{\vec{x}_p \in \partial\Omega} ||\vec{x} - \vec{x}_p|| \tag{3.16}$$

An example of SDF for a circumference in a 2D space is presented in **Figure 3.5**.

Some properties of the sign distance function will be useful in the continuation:

Eikonal equation If Ω is a subset of the Euclidean space $\mathbb{R}^n$ with smooth boundary, then the signed distance function is differentiable almost everywhere, and its gradient satisfies the eikonal equation:

$$\left|\vec{\nabla}\phi\right| = 1 \tag{3.17}$$

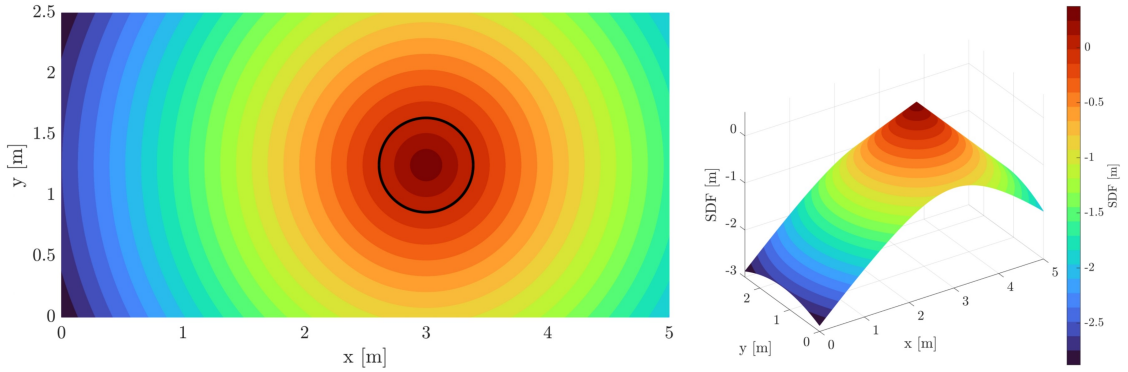


Figure 3.5: SDF of a circumference with origin in $x = 3.0[m]$ and $y = 1.25[m]$ and radius $r = 0.375[m]$. The black line is the circumference.

Normal direction If the boundary of Ω is of class C^k for $k \leq 2$ then d is C^k on points sufficiently close to the boundary of Ω. In particular, on the boundary f satisfies:

$$\vec{\nabla}\phi(\vec{x}) = \vec{n}(\vec{x}) \tag{3.18}$$

where $\vec{n}$ is the inward (pointing positive values of ϕ) normal vector field. The signed distance function is thus a differentiable extension of the normal vector field.

3.5 Eikonal Equation Method

The method illustrated in this section [6] is the one used for the simulations object of the thesis project. This method can be classified as a Discrete Forcing, Direct Boundary Condition Imposition, Cut-Cell Approach IBM. This method uses the Signed Distance Function to identify the position and the direction normal to the immersed boundary. The Eikonal Equation Method has some advantages:

- It is easy to implement;
- It is easy to apply for complex 3D geometries;
- It requires no additional time step restrictions;
- It is very easy to implement in parallel codes.

First, we recall the governing equation of the problem i.e. the Navier-Stokes equations for incompressible flows:

$$\begin{cases} \dfrac{\partial u_i}{\partial x_i} = 0 \\ \\ \dfrac{\partial u_i}{\partial t} + \dfrac{\partial u_i u_j}{\partial x_j} + \dfrac{1}{\rho}\dfrac{\partial p}{\partial x_i} = \dfrac{\partial \tau_{ij}}{\partial x_j} \end{cases} \tag{3.19}$$

In vector form become:

$$\overrightarrow{\overrightarrow{I}}\frac{\partial \vec{q}}{\partial t} + \frac{\overrightarrow{F_i^e}}{\partial x_i} = \frac{\partial \overrightarrow{F_i^v}}{\partial \vec{x_i}} \tag{3.20}$$

where:

$$\vec{q} = \begin{Bmatrix} p \\ \vec{u} \end{Bmatrix}; \quad \overrightarrow{F_i^e} = \begin{Bmatrix} u_i \\ \vec{u}u_i + \dfrac{p}{\rho} \end{Bmatrix}; \quad \overrightarrow{F_i^v} = \begin{Bmatrix} 0 \\ \tau_{ij} \end{Bmatrix} \tag{3.21}$$

$$\overrightarrow{\overrightarrow{I}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.22}$$

This IBM uses the SDF to simply identify which nodes are in the fluid domain and which are in the immersed body and therefore need a different treatment.

The first step is to obtain the solution inside the body by finding the steady state solution (with respect to a pseudo time τ) of the equation:

$$\frac{\partial \vec{q}}{\partial \tau} = \hat{n} \cdot \vec{\nabla}\vec{q} \tag{3.23}$$

By doing so, the solution in the fluid near the body is propagated inside the immersed body. The solution is extrapolated in few cells inside the body and this is sufficient to enforce the boundary conditions.

The extrapolated solution must be adapted to enforce the boundary conditions on the body surface before the flux computation. Two different procedures exist for the no-slip and free-slip conditions.

• **No-slip condition** The velocities in the nodes with $\phi > 0$ are modified as:

$$\vec{u} \rightarrow -\vec{u} \tag{3.24}$$

• **Slip condition** The velocities in the nodes with $\phi > 0$ are locally modified as:

$$\vec{u} \rightarrow \vec{u} - 2\,(\hat{n} \cdot \vec{u})\,\hat{n} \tag{3.25}$$

Once this procedure is performed, the flux is computed as normally done with finite difference or finite volume method. In this way we can obtain the correct Eulerian flux on body interface $\vec{\tilde{f}}_{i+1/2}$ (between $\vec{p}_i$ and $\vec{p}_{i+1}$, see **Figure 3.6** as reference). Finally the flux is extrapolated or interpolated in the position $\vec{p}_{i+1/2}$ through the value at position $\vec{p}_{i-1/2}$:

$$\vec{f}_{i+1/2} = \vec{f}_{i-1/2} + \Delta x \frac{\vec{\tilde{f}}_{i+1/2} - \vec{f}_{i-1/2}}{\Delta x/2 + d_x} \tag{3.26}$$

where Δx is the distance between point $\vec{p}_i$ and $\vec{p}_{i+1}$ and d_x is the distance from $\vec{p}_i$ and the intersection of the body with segment $[\vec{p}_i, \vec{p}_{i+1}]$. The term d_x can be computed as:

$$d_x = \frac{-\phi_i}{\phi_{i+1} - \phi_i} \Delta x \tag{3.27}$$

A similar approach is adopted also for viscous fluxes.

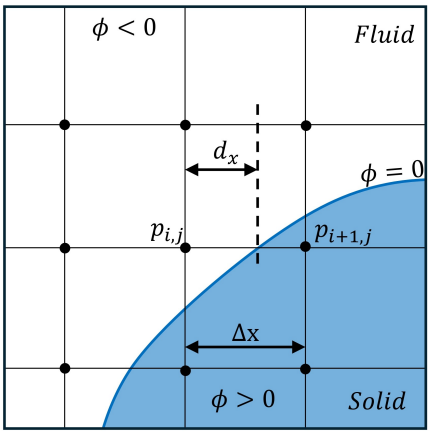


Figure 3.6: Scheme of the computational grid for Eikonal method [6].

Chapter 4

Collision Model

The main focus of this thesis project is the implementation of a collision model for the impact of solid particles on rigid wall described with a Signed Distance Function and to be coupled with the Eikonal IBM.

The collision model has been implemented on an open source DNS solver in FORTRAN90/95. The particle motion is described with the equation illustrated in **section 2.3**, then the collision is implemented to change particle velocity when the impact with a rigid body is detected. All the details of the particle tracking method are reported in the following sections.

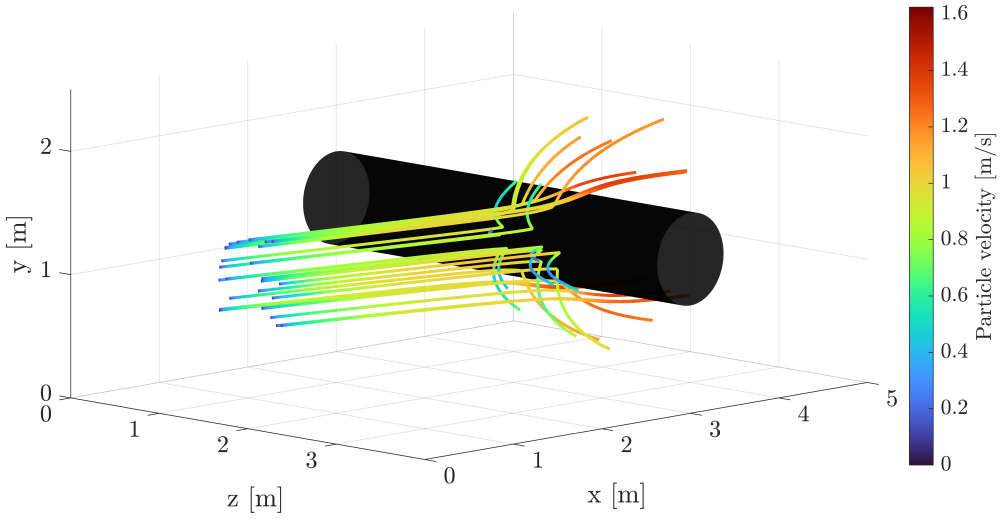


Figure 4.1: Simple example of particles evolution over a cylinder. The velocity field is a generic field in steady condition.

A first basic example of this implementation is given in **Figure 4.1**, where the particle trajectory is computed, instead, the flow field is a generic given steady field to simplify the problem and obtain a first validation of the collision model.

4.1 CaNS Solver

The collision model has been implemented in an open source code for massively-parallel numerical simulations of turbulent flows: the Canonical Navier-Stokes (CaNS) [4].

CaNS uses a second-order, finite-difference pressure correction scheme, where the pressure Poisson equation is solved with the method of eigenfunction expansions. This approach allows for very efficient FFT-based solvers in problems with different combinations of homogeneous pressure boundary conditions.

The numerical algorithm implemented in CaNS solves the Navier-Stokes equations for incompressible, Newtonian flows with unit density:

$$\begin{cases} \vec{\nabla} \cdot \vec{U} = 0 \\ \dfrac{\partial \vec{U}}{\partial t} + \vec{U} \cdot \vec{\nabla}\vec{U} = -\vec{\nabla} p + \nu \nabla^2 \vec{U} \end{cases} \tag{4.1}$$

These equations are solved in a structured Cartesian grid. Standard second-order finite-difference schemes are used for spatial discretization over staggered grids to avoid pressure-velocity decoupling. The equations a coupled through a pressure-correction method. The time integration is performed with a low-storage, three-step Runge-Kutta (RK3) scheme. The substeps of the Runge-Kutta method are:

$$\vec{U}^* = \vec{U}_k + \Delta t \left(\alpha_k \overrightarrow{AD}^k + \beta_k \overrightarrow{AD}^{k-1} - \gamma_k \vec{\nabla} p^{k-1/2} \right) \tag{4.2}$$

$$\nabla^2 p^* = \frac{\vec{\nabla} \cdot \vec{U}^*}{\gamma_k \Delta t} \tag{4.3}$$

$$\vec{U}^k = \vec{U}^* - \gamma_k \Delta t \vec{\nabla} p^* \tag{4.4}$$

$$p^{k+1/2} = p^{k-1/2} + p^* \tag{4.5}$$

where $\overrightarrow{AD} = -\vec{U} \cdot \vec{\nabla}\vec{U} + \nu \nabla^2 \vec{U}$, $\vec{U}$ is the predicted velocity and p^* is the correction pressure. The method has three stages, so $k = 1,\ 2,\ 3$; where $k = 1$ is the time step n and $k = 3$ is the time step $n+1$. The coefficient of the Runge-Kutta scheme, α_k and β_k are given in **Table 4.1** and $\gamma_k = \alpha_k + \beta_k$.

k	α_k	β_k
1	8/15	0
2	5/12	−17/60
3	3/4	−5/12

Table 4.1: Coefficients of low-storage, three stage Runge-Kutta scheme.

4.2 Particle Motion

The motion of the particles in the fluid domain is computed thanks to the equations discussed in **section 2.3**:

$$\begin{cases} \dfrac{d\vec{U}_p}{dt} = \left(1 + 0.15Re_p^{0.687}\right)\left(\vec{U} - \vec{U}_p\right)\dfrac{1}{\tau_p} \\ \dfrac{d\vec{x}_p}{dt} = \vec{U}_p \end{cases} \tag{4.6}$$

From the first we obtain the particle velocity and from the second the particle position at a given time step.

These equations are evolved in time domain with a Runge-Kutta scheme. In particular a low storage Runge-Kutta scheme is implemented in CaNS to reduce the memory required.

4.3 Collision

Once the particle motion is defined, we want to model the collision of a particle with an Immersed Boundary that is represented with a SDF. The collision is modelled under some hypothesis:

- The collision is a pure elastic collision;
- The wall where the particle collides is locally assumed as a flat plate;
- The particle is assumed to be a mass point, so no rigid body and moment of inertia effects are considered.

Despite being modelled as a mass point, the particle is characterised by a radius. This is necessary to avoid the particle to remain blocked in the vicinity of the

immersed boundary (or vibrate around this position) where the velocity is near zero. Therefore the radius is considered to activate the collision before this situation could happen and the collision is calculated when any point of the particle enter the Immersed Boundary.

The procedure to model the particle collision after the collision is detected, can be subdivided into three main steps:

I. Definition of the direction normal to the Immersed Boundary at the position of the collision;

II. Particle shift to take into account the real time at which the collision happens between two computational time steps;

III. Inversion of the velocity component normal to the Immersed Boundary.

All these steps are discussed in detail in **subsection 4.3.1**, **4.3.2** and **4.3.3**.

4.3.1 Normal Direction to Immerse Boundary

The first step necessary to model the collision is to identify the normal direction at the point of the collision on the Immersed Boundary. To do this we can take advantage of the properties of the SDF. In particular we know that the gradient of the SDF in a certain point is equal to the normal direction to the field in that point. Therefore, if $\vec{p} = \{x_p, y_p, z_p\}^T$ is the position of the collision, we have:

$$\hat{n}(\vec{p}) = \vec{\nabla}\phi(\vec{p}) = \begin{Bmatrix} \dfrac{\partial\phi}{\partial x} \\ \dfrac{\partial\phi}{\partial y} \\ \dfrac{\partial\phi}{\partial z} \end{Bmatrix} = \begin{Bmatrix} n_1 \\ n_2 \\ n_3 \end{Bmatrix} \tag{4.7}$$

Implementation

The position $\vec{p}$ of the particle at the collision is known by the solution of the particle equations of motion. Now we need to find some support points where evaluating the SDF in order to perform the derivative with a finite difference scheme.

In detail, considering the x direction, we perform the interpolation of the SDF at two points (**Figure 4.2**):

$$\phi\left(x_{i^+}, y_p, z_p\right) = \phi\left(x_p + \delta, y_p, z_p\right) = \phi_i^+ \tag{4.8}$$

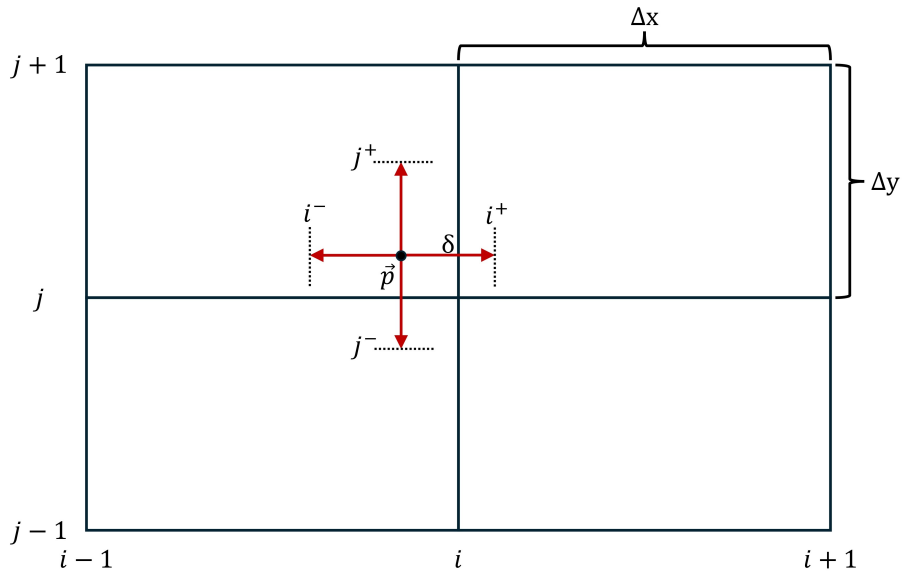


Figure 4.2: Schematic example of a 2D case for the evaluation of the direction normal to the immersed boundary.

$$\phi\left(x_{i^-}, y_p z_p\right) = \phi\left(x_p - \delta, y_p, z_p\right) = \phi_i^- \tag{4.9}$$

where δ is a small displacement. Thanks to the SDF evaluated on these two points it is possible to evaluate the gradient of the Signed Distance Function along the x direction. Using a centered first derivative finite difference stencil we obtain:

$$n_1(\vec{p}) = \frac{\partial \phi}{\partial x}(\vec{p}) = \frac{\phi_i^+ - \phi_i^-}{2\delta} \tag{4.10}$$

The same for the other directions:

$$n_2(\vec{p}) = \frac{\partial \phi}{\partial y}(\vec{p}) = \frac{\phi_j^+ - \phi_j^-}{2\delta} \tag{4.11}$$

$$n_3(\vec{p}) = \frac{\partial \phi}{\partial z}(\vec{p}) = \frac{\phi_k^+ - \phi_k^-}{2\delta} \tag{4.12}$$

In this way we have identified the direction normal to the immersed boundary at the location of the collision. This information will be fundamental in the following step of the collision model.

4.3.2 Particle Offset

As said before, the collision is identified when the particle has yet entered the immersed boundary, therefore at that time the particle should have yet changed

its trajectory due to the collision. We need to take into account this effect related to the finite time discretization used to solve the problem, indeed, we know the particle position at the time step t_i and t_{i+1}, but not at the exact time at which the collision happens t^* (**Figure 4.3**). To do this, we move the particle outside the immersed body of a distance equal to the particle penetration. This assumption is valid only for elastic collision.

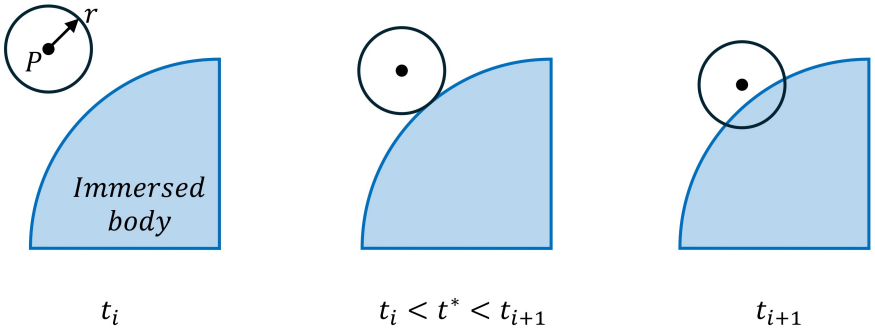


Figure 4.3: Temporal evolution of particle position with respect to the immersed boundary. Times t_i and t_{i+1} are two simulation time steps, t^* is the unknown time at which the particle collides with the body.

Implementation

First we identify the position on the surface of the particle to evaluate how deep the particle has penetrated the immersed body:

$$\vec{p_r} = \vec{p} - r\hat{n} \tag{4.13}$$

where $\vec{p}$ is the position of the particle center, $\vec{p_r}$ is the position on the particle surface along the normal direction and r is the radius of the particle.

On this position we interpolate the SDF which identifies the magnitude of the particle penetration on the solid body:

$$\phi_r = \phi(\vec{p_r}) \tag{4.14}$$

Since with an elastic collision only the velocity normal to the impact surface is modified, the position along the tangential direction does not need to be revised. Instead, the position along the normal direction must be changed, but since the

normal direction velocity has the same magnitude and opposite sign, we can translate the particle to have the $\vec{p_r}$ position symmetric to the immersed boundary along the normal direction.

Looking at **Figure 4.4**, we can notice that the distance between $\vec{p_r}$ and the position on the immersed boundary along the normal direction is: $\phi_r \hat{n}$. So we have to move the particle two times this distance along the normal direction, so the position of the particle after the collision is:

$$\vec{p} = \vec{p}_{old} + 2\phi_r \hat{n} \tag{4.15}$$

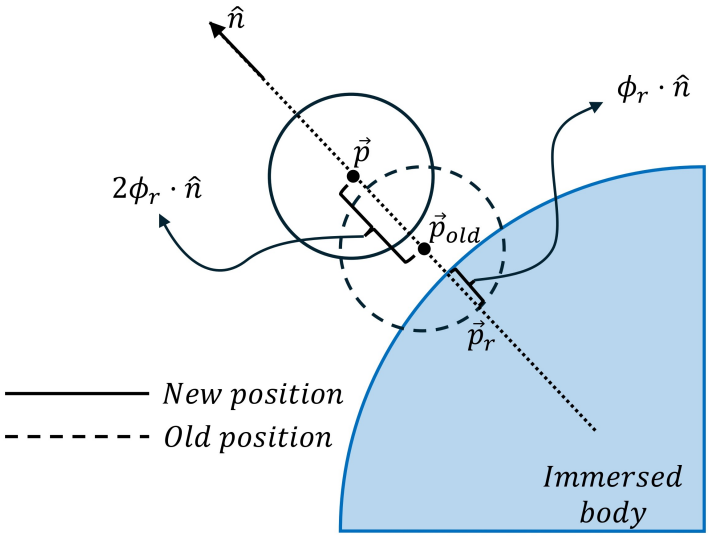


Figure 4.4: Detailed representation of the procedure to move the particle at the correct position after the collision is detected.

4.3.3 Velocity Change after the Collision

In this work we assume elastic collision, so the kinetic energy of the particle is conserved. As said before we assume the wall to be locally flat, therefore the velocity component parallel to the surface is unchanged and the normal component has opposite sign after the collision (**Figure 4.5**). Since point mass particles are assumed, there is no need to take into account rotational motion and related phenomena.

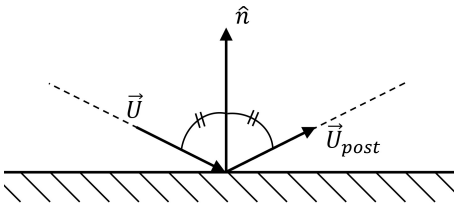


Figure 4.5: Velocity vector before and after the collision of a mass point on a rigid flat wall.

Implementation

To implement the collision, it is necessary to find the normal and the tangential components of the velocity with respect to the body surface in order to manipulate them as illustrated above for the elastic collision.

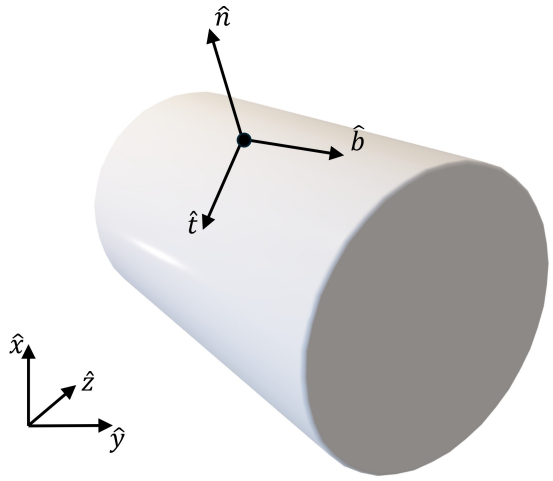


Figure 4.6: Tangent, Normal and Binormal reference system centered on a generic point of a cylinder. Tangent versor and Binormal versor define a plane tangent to the surface of the cylinder.

First, it is necessary to identify the *tnb* (Tangent, Normal and Binormal) reference system locally at the position of the collision (**Figure 4.6**). The normal versor $\hat{n}$ has yet been defined in **subsection 4.3.1**. The tangential versor $\hat{t}$ can be found using one of the following:

$$\vec{t} = \begin{Bmatrix} t_1 \\ t_2 \\ t_3 \end{Bmatrix} = \begin{Bmatrix} n_1 \\ n_2 \\ n_3 \end{Bmatrix} \times \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} 0 \\ n_3 \\ -n_2 \end{Bmatrix}, \quad or \tag{4.16}$$

$$\vec{t} = \begin{Bmatrix} t_1 \\ t_2 \\ t_3 \end{Bmatrix} = \begin{Bmatrix} n_1 \\ n_2 \\ n_3 \end{Bmatrix} \times \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix}, \quad or \quad \vec{t} = \begin{Bmatrix} t_1 \\ t_2 \\ t_3 \end{Bmatrix} = \begin{Bmatrix} n_1 \\ n_2 \\ n_3 \end{Bmatrix} \times \begin{Bmatrix} 0 \\ 0 \\ 1 \end{Bmatrix} \tag{4.17}$$

The three versions are used as a function of the normal versor $\hat{n}$ in order to avoid singularity in calculating $\hat{t}$. Then the tangential vector must be normalized. Let us consider the first case **Equation 4.16**:

$$\hat{t} = \frac{1}{\sqrt{t_1^2 + t_2^2 + t_3^2}} \begin{Bmatrix} 0 \\ n_3 \\ -n_2 \end{Bmatrix} = \begin{Bmatrix} 0 \\ n_3/A \\ -n_2/A \end{Bmatrix} \tag{4.18}$$

where $A = \sqrt{n_2^2 + n_3^2}$.

Now the binormal versor can be found:

$$\vec{b} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \end{Bmatrix} = \hat{t} \times \hat{n} = \begin{Bmatrix} n_2^2 + n_3^2 \\ -n_1 n_2 \\ -n_1 n_3 \end{Bmatrix} \tag{4.19}$$

and by normalizing it we have:

$$\hat{b} = \frac{1}{\sqrt{b_1^2 + b_2^2 + b_3^2}} \begin{Bmatrix} n_2^2 + n_3^2 \\ -n_1 n_2 \\ -n_1 n_3 \end{Bmatrix} = \begin{Bmatrix} A/B \\ -n_1 n_2/AB \\ -n_1 n_3/AB \end{Bmatrix} \tag{4.20}$$

where $B = \sqrt{n_1^2 + n_2^2 + n_3^2}$, but since $\hat{n}$ is a versor with unitary norm it is $B = 1$.

Now we want to project the velocity in the *tnb* reference system. After this the normal component of the velocity is explicit, so its sign is changed. Finally, with the base change matrix we bring again the velocity to the global reference system.

However, it is interesting to note that by doing all these calculations analytically, a very simplified form of this procedure can be obtained. First, we introduce the velocity in the global reference system:

$$\vec{U}_{xyz} = \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \end{Bmatrix} \tag{4.21}$$

and its projection on the *tnb* reference system:

$$\vec{U}_{tnb} = \begin{bmatrix} t_1 & t_2 & t_3 \\ n_1 & n_2 & n_3 \\ b_1 & b_2 & b_3 \end{bmatrix} \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \end{Bmatrix} = \begin{Bmatrix} U_2 \dfrac{n_3}{A} - U_3 \dfrac{n_2}{A} \\ -U_1 n_1 - U_2 n_2 - U_3 n_3 \\ U_1 A - U_2 \dfrac{n_1 n_2}{A} - U_3 \dfrac{n_1 n_3}{A} \end{Bmatrix} \tag{4.22}$$

where the velocity component in the normal direction $U_{tnb}(2)$ has yet been inverted.

The base change matrix is:

$$R = \begin{bmatrix} t_1 & n_1 & b_1 \\ t_2 & n_2 & b_2 \\ t_3 & n_3 & b_3 \end{bmatrix} \tag{4.23}$$

and therefore the velocity after the collision is $\vec{U}_{post} = R\vec{U}_{tnb}$:

$$\vec{U}_{post} = \begin{Bmatrix} -n_1\left(U_1 n_1 + U_2 n_2 + U_3 n_3\right) + U_1 A^2 - U_2 n_1 n_2 - U_3 n_1 n_3 \\ -n_2\left(U_1 n_1 + U_2 n_2 + U_3 n_3\right) + U_2 \frac{n_3^2}{A^2} - U_3 \frac{n_2 n_3}{A^2} - U_1 n_1 n_2 + U_2 \frac{n_1^2 n_2^2}{A^2} + U_3 \frac{n_1^2 n_2 n_2}{A^2} \\ -n_3\left(U_1 n_1 + U_2 n_2 + U_3 n_3\right) - U_2 \frac{n_2 n_3}{A^2} + U_3 \frac{n_2^2}{A^2} - U_1 n_1 n_3 + U_2 \frac{n_1^2 n_2 n_3}{A^2} + U_3 \frac{n_1^2 n_3^2}{A^2} \end{Bmatrix} \tag{4.24}$$

Equation 4.24 can be strongly simplified by noting that:

$$\sqrt{n_1^2 + n_2^2 + n_3^2} = 1 \implies n_1^2 + n_2^2 + n_3^2 = 1 \tag{4.25}$$

The steps to simplify **Equation 4.24** are omitted for conciseness, but the final result is:

$$\vec{U}_{post} = \begin{Bmatrix} U_1 - 2n_1\left(U_1 n_1 + U_2 n_2 + U_3 n_3\right) \\ U_2 - 2n_2\left(U_1 n_1 + U_2 n_2 + U_3 n_3\right) \\ U_3 - 2n_3\left(U_1 n_1 + U_2 n_2 + U_3 n_3\right) \end{Bmatrix} \tag{4.26}$$

As a result of all these calculations, it is clear that for the evaluation of the velocity after the elastic collision of the particle, **Equation 4.26** is sufficient. Hence, it is not necessary to evaluate tangent $\hat{t}$ versor and binormal $\hat{b}$ versor and it is not necessary to change two times the reference system, having a reduction of the computations necessary to evaluate the velocity after the collision.

Chapter 5

Simulations and Results

In this chapter we want to illustrate some results obtained from simulations carried out with the code discussed in **chapter 4**. First a preliminary validation of the collision is illustrated with a simplified model for two case studies. Then the results of complete flow-particle simulations are illustrated and are compared with some results available on literature in order to validate the proposed model.

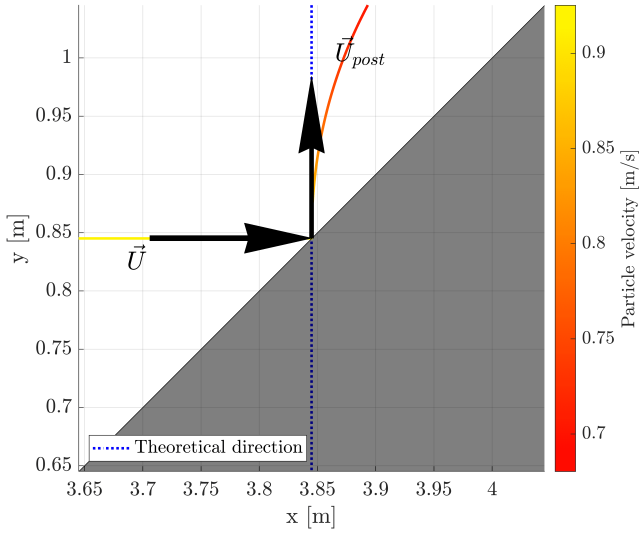


Figure 5.1: Validation of the collision model on a ramp under uniform steady flow. The dotted blue line illustrates the theoretical direction of the particle right after the collision. The two arrows represent the velocity of the particle at the time step before ($\vec{U}$) and after ($\vec{U}_{post}$) the collision. The trajectory of the particle is colored according to its velocity. The shaded area is the rigid body, i.e. the ramp.

5.1 Validation through Simplified Simulations

In order to have a first validation of the collision model implemented, a simplified 2D simulation has been performed. In particular we simulated a ramp tilt of 45° with respect to the flow direction. The flow is not simulated, but we assumed a simplified steady, uniform velocity field in order to have the particle impacting the ramp with a velocity parallel to x axis and so the expected velocity direction after the collision is parallel to the y axis.

As we can see in **Figure 5.1**, the velocity after the collision $\vec{U}_{post}$ computed with this simplified code is exactly superimposed to the theoretical direction. In this particular case, since the collision is elastic and due to the simple geometry, the theoretical direction of the particle velocity after the collision is clearly along the y axis. We consider as reference the velocity right after the collision, since later the particle is influenced by the flow velocity and the trajectory is deviated.

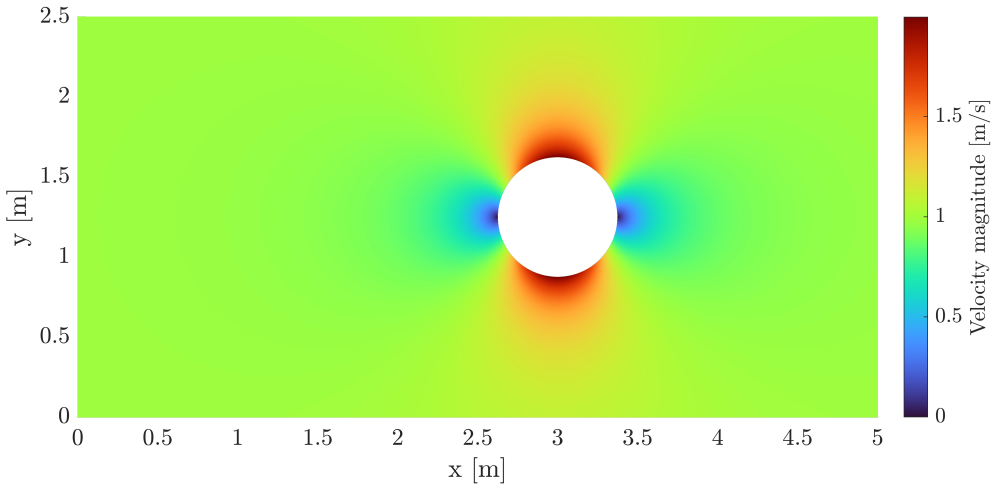


Figure 5.2: Potential flow around the cylinder used for a simplified validation of the collision model.

As a second test, we simulated with the same approach the collision on a cylinder in a 2D space (i.e. on a circumference). Also in this case we considered a steady flow, in particular the potential flow over the cylinder (**Figure 5.2**) and only the particle motion equations are evolved over this flow.

In this case, the direction of the particle after the collision is not obvious. The theoretical direction is evaluated by computing the angle between the velocity and

the direction normal to the cylinder and then applying a rotation of the velocity about two time this angle.

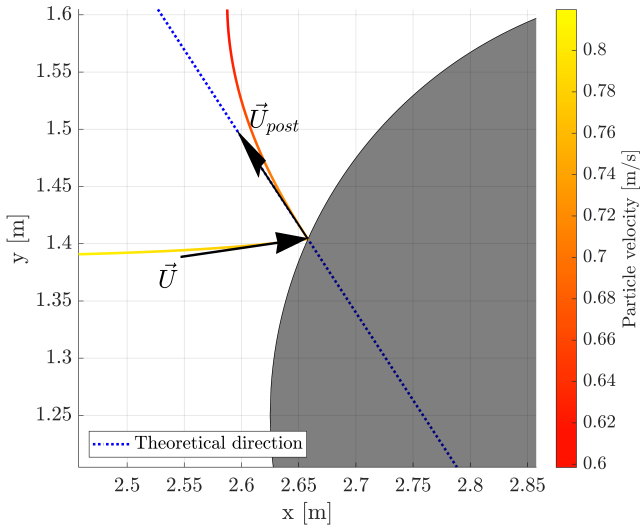


Figure 5.3: Validation of the collision model on a cylinder under potential flow. The dotted blue line illustrates the theoretical direction of the particle right after the collision. The two arrows represent the velocity of the particle at the time step before ($\vec{U}$) and after ($\vec{U}_{post}$) the collision. The trajectory of the particle is colored according to its velocity. The shaded area is the rigid body, i.e. the cylinder.

The result of this test can be observed in **Figure 5.3**, again the numerical result for the velocity after the collision is overlapped to the theoretically expected direction.

5.2 Model Validation

To have a better validation of the model, we performed some simulations coupling the CaNS code with the Eikonal IBM and the collision model. So in these simulations the flow field is solved with the pressure-correction method implemented in CaNS, the immerse body is described with the IBM approach and the collision is modelled as discussed in **section 4.3**.

These simulation has been made on a parallelepipedon shaped domain with a cylinder as Immersed Boundary. The cylinder has diameter $D = 0.2$ $[m]$ and is located at a distance $5D$ from the inlet. The domain has dimensions $40D$ along x,

$24D$ along z and $D/13$ along y (**Figure 5.4**). The domain is discretized with 256 nodes along x direction, 4 nodes along y direction and 128 nodes along z direction. Each face of the domain has been set with a periodic boundary condition and a forcing velocity has been activated to maintain the fluid motion. All the particle are seeded at the begin of simulations with a random position and zero velocity. The number of particles introduced in the domain is so that diluted condition are guaranteed.

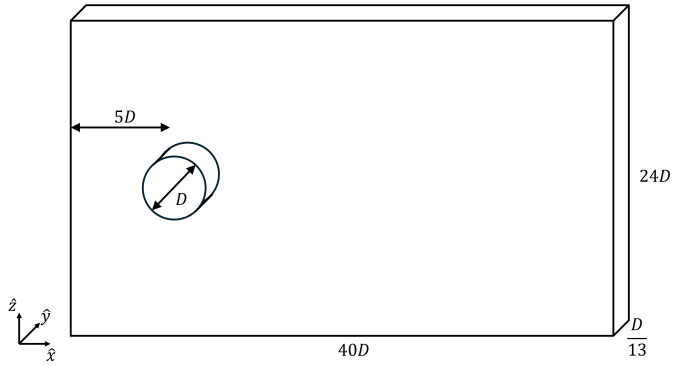


Figure 5.4: Domain dimensions adopted for the validation simulations.

The simulation is set up with a Reynolds number $Re = 150$, the undisturbed velocity is $\vec{U}_0 = \{1,0,0\}^T$ and therefore, the viscosity is:

$$\nu = \frac{DU_0}{Re} = 0.001333\ [m^2/s] \tag{5.1}$$

We are interested in simulating different Stokes numbers to understand the effect of different particle inertia in the particle trajectory. We simulated four different scenarios varying the Stokes number: $St = 0.5$; $St = 1$; $St = 2.8$ and $St = 33.5$. In these simulations we assume always the same densities for the fluid and for the particles: $\rho_f = 1.3\ [kg/m^3]$ and $\rho_p = 3510\ [kg/m^3]$ and so $\rho_r = \rho_p/\rho_f$ is fixed. Thus to change the Stoke number we have to change the radius of the particles r_p, indeed:

$$St = \tau_p \frac{1}{\tau_l} = \frac{d_p^2 \rho_p}{18\rho_f \nu} \frac{U_0}{D} \tag{5.2}$$

$$r_p = \frac{1}{2}\sqrt{\frac{18\nu D}{\rho_r U_0}}\sqrt{St} \tag{5.3}$$

The radius used for the simulations are reported in **Table 5.1**

St	r_p
0.5	$4.7140452079 \cdot 10^{-4}$
1	$6.6666666667 \cdot 10^{-4}$
2.8	$1.1155467020 \cdot 10^{-3}$
33.5	$3.8586123009 \cdot 10^{-3}$

Table 5.1: Simulated radius corresponding to the different Stokes numbers.

All these settings have been chosen to replicate the simulations performed by Schuster [29] to have some references and to validate the model. However these simulations have some differences in the implementation that must be taken into account when observing the results. The main differences between our simulations and the ones by Schuster are:

- We consider diluted condition, instead the reference is in dense condition;
- We consider elastic collision, the reference considers inelastic condition;
- We neglect the effects of the particle on the fluid, the reference considers these;
- We seed the particles at the beginning of the simulation, the reference seeds particles on the inlet at every iteration.

All these aspects have an influence on the results. Since we assume diluted condition, it is not necessary to consider the collision between particles, instead in dense condition it is necessary to consider also this phenomena, indeed this is done by Schuster.

The difference about the elastic/inelastic collision has an impact due to the influence on the rebound of the particle after the collision and also the energy of the particle after the collision is different since elastic collisions are conservative and inelastic collisions are non-conservative.

The influence of the particles on the fluid has a significant impact on the results. This is not considered in our code since we assume a one-way coupling, but is implemented on the Schuster code. This difference gets bigger by increasing the Stokes number as will be clear later.

Moreover, Schuster seeds the particle at the inlet at each time step and takes the results when the turbulence is completely established. Due to some limitation of our code this is not possible, so we take the results after few time steps. This is necessary because the particle are all seeded simultaneously at the beginning of the simulation, and due to the periodic boundary condition, when exiting the domain at the bottom the particle enter again at the inlet with unchanged velocity and y and x position. This mean that the particle reentering into the domain have been influenced by them previous history, changing the results.

The results about the particle distribution around the cylinder obtained by Schuster in the reference paper are illustrate in **Figure 5.5**.

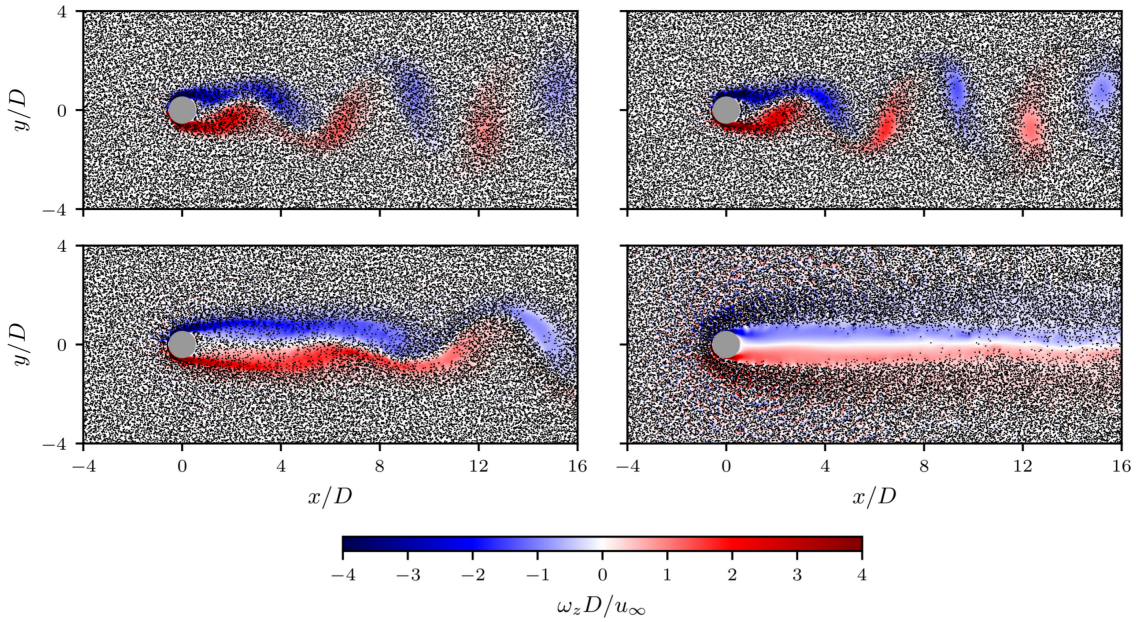
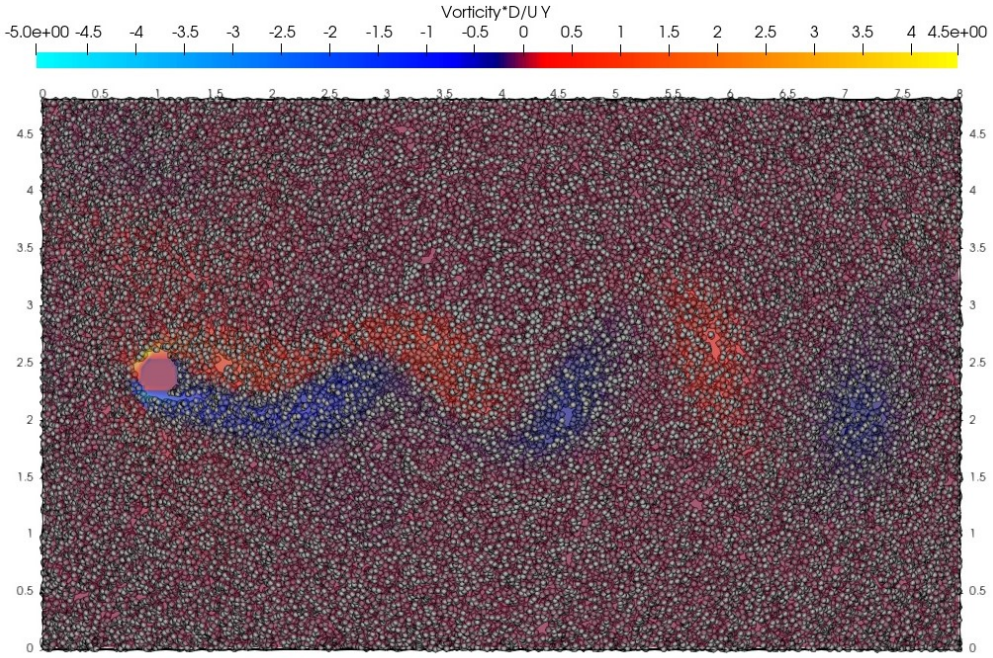


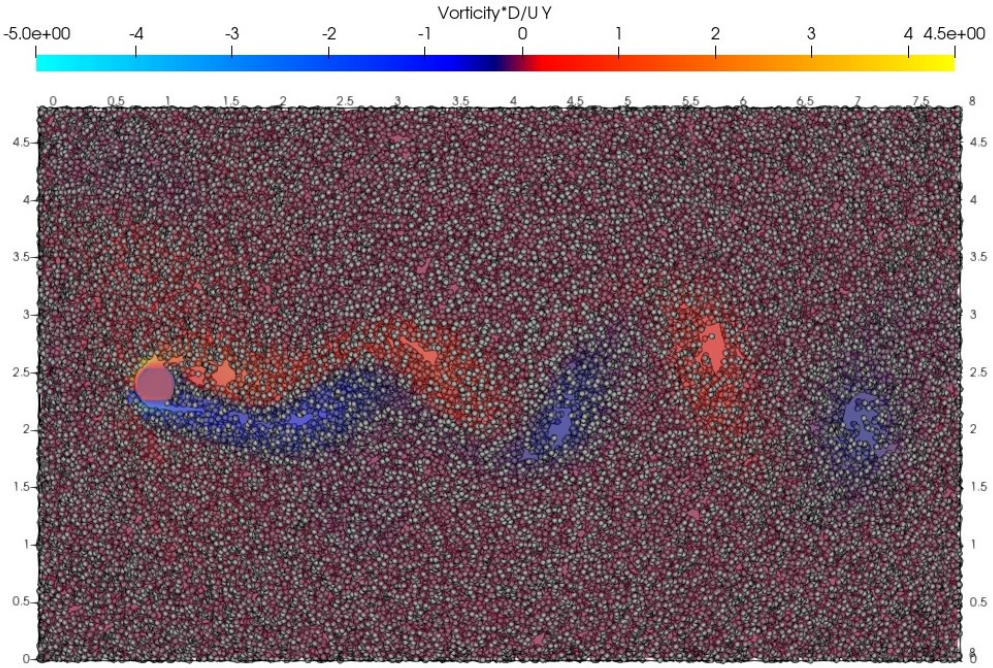
Figure 5.5: Snapshot of particle positions and vorticity field at fluid Reynolds number $Re = 150$. $St = 0.5$: top/left; $St = 1$: top/right; $St = 2.8$: bottom/left; $St = 33.5$: bottom/right [29].

First, let us consider the results for $St = 0.5$ (**Figure 5.6a**). The flow vorticity field is very similar between our simulations and the reference one, this because these low inertia particle are not effective in modifying the flow. In term of particle distribution we observe in both cases a uniform distribution over the flow domain, also in the vortex regions.

Looking at the results for $St = 1$ (**Figure 5.6b**) we note again that the flow is not influenced by the presence of the particle and the vorticity field is unchanged

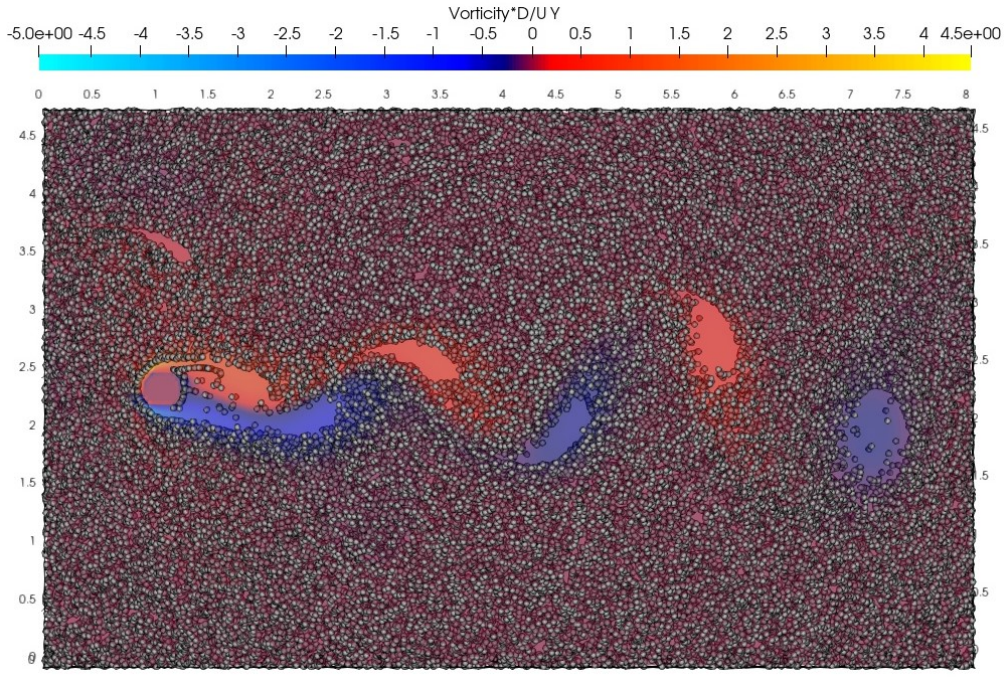


(a) $St = 0.5$

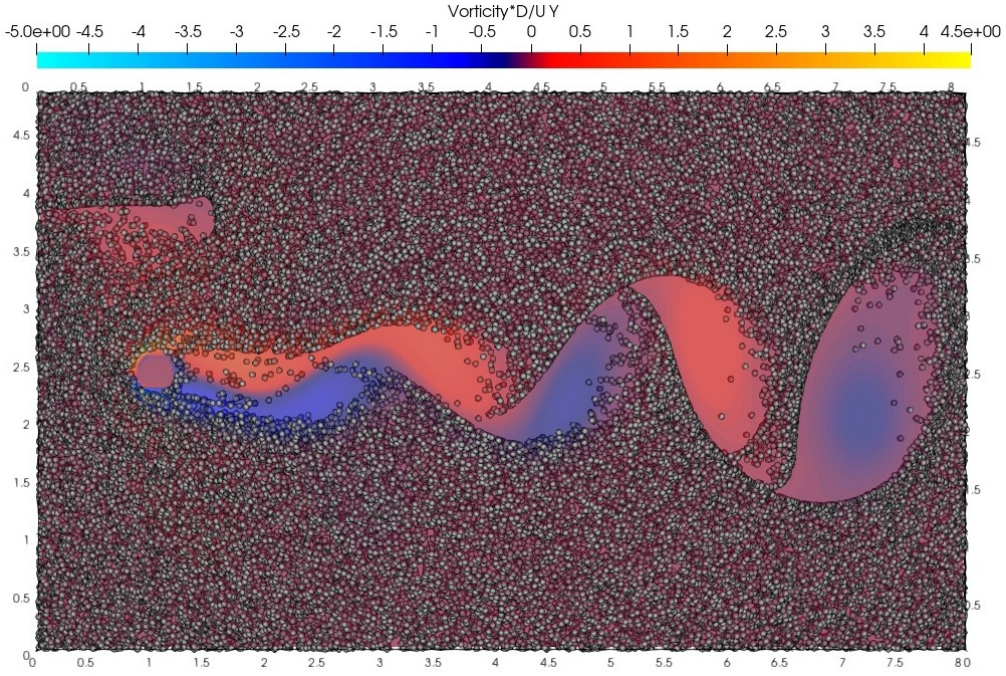


(b) $St = 1$

Figure 5.6: Particle position and normalized flow vorticity $(\omega_y D)/U_0$ for two different Stokes numbers.



(a) $St = 2.8$



(b) $St = 33.5$

Figure 5.7: Particle position and normalized flow vorticity $(\omega_y D)/U_0$ for two different Stokes numbers.

with respect to the $St = 0.5$ case. We can observe that the particles distribution is not homogeneous as before over the whole flow domain, but some lacks of particles appear in the vortex region.

Simulations at higher Stokes number show some differences, mainly due to the fact that high inertia particles can affect significantly the flow field. Looking at the results for $St = 2.8$ (**Figure 5.7a**) and $St = 33.5$ (**Figure 5.7b**) it is clear that the vorticity field is completely different, therefore it is not suitable to compare the results for the particle motion.

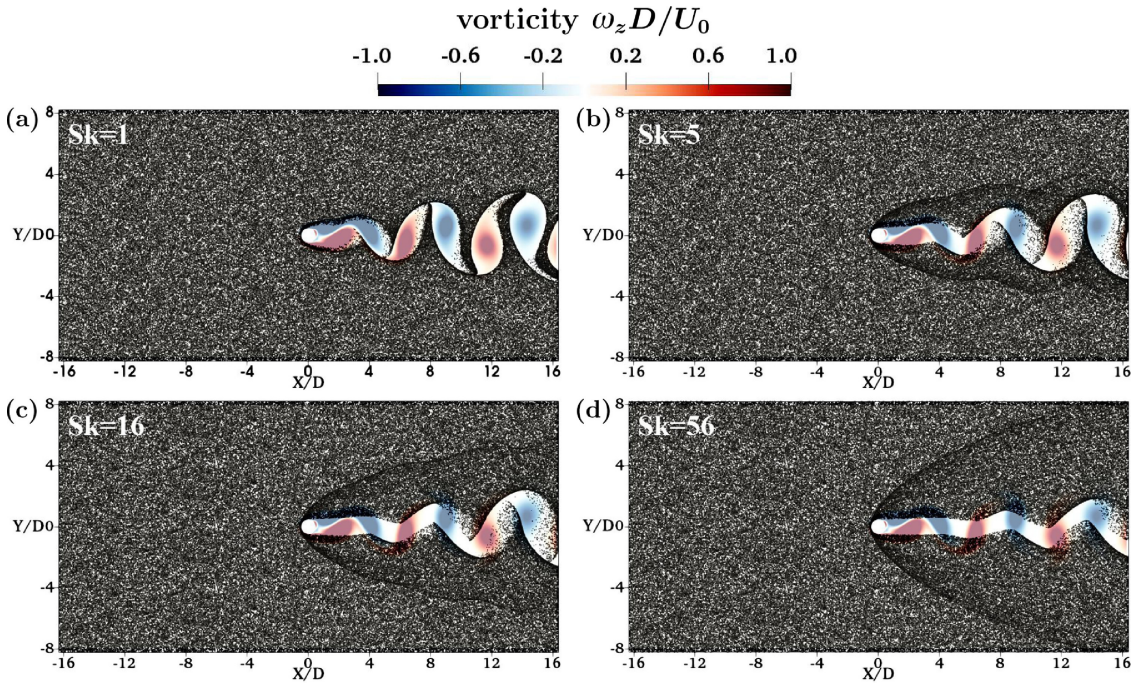


Figure 5.8: Snapshot of particle positions and vorticity field at fluid Reynolds number $Re = 100$. (a) $St = 1$; (b) $St = 5$; (c) $St = 16$; (d) $St = 56$ [30].

Higher Stokes number are more similar to the results obtained by Shi [30] and reported in **Figure 5.8**. These simulations consider a one-way coupling for particle-flow interaction, so the flow field is not influenced by the particles and is independent from the Stokes number, but also in this case an inelastic collision is considered. For these high Stokes numbers the regions without particles are larger since the particles are less influenced by the fluid, so the vortex regions have no particles.

At high Stokes numbers is possible to observe the bow shock particle clustering. This can be noted at $St = 33.5$ form Schuster simulations (**Figure 5.5**) and also

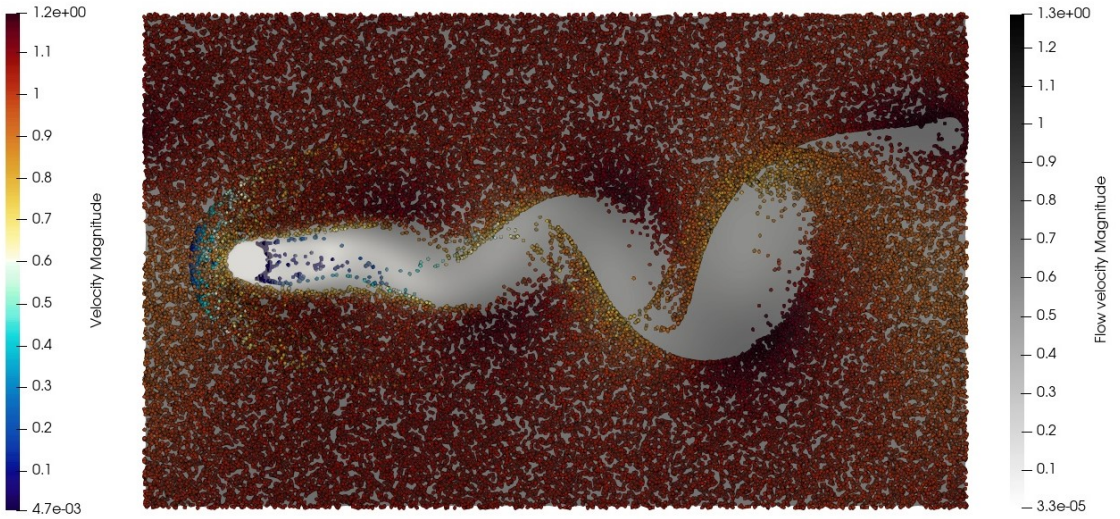


Figure 5.9: Particle velocity magnitude and flow velocity magnitude for $St = 33.5$. Color scale for particle velocity, gray scale for flow velocity.

for $St > 5$ in Shi simulations (**Figure 5.8**). The bow shock is a concentration of particles originating near the body and extending upstream of the cylinder. The bow shock forms due to the presence of particles bounced after the collision increasing the local density of the particles.

Due to the lower density of particles in our simulations, this phenomena is more difficult to be noted, indeed Shi solved the problem on a 3D domain and then projected all the particles on a single plane, since the statistics are independent on the z direction, making this phenomena clearly visible. However, looking at the snapshot of particle velocity in **Figure 5.9** it is possible to observe the presence of the bow shock. In our results the bow shock is more distant form the cylinder due to the fact that we do not consider any inelastic rebound coefficient.

Chapter 6

Conclusions

The main focus of this thesis project is the implementation of a model to describe the collision of a solid particle on a rigid wall described with a SDF. The basis of this code is the open source DNS solver, CaNS, over which some modules have been added to obtain the final implementation. In particular we added the SDF to identify the immersed boundary, the Eikonal IBM to solve the flow over an immersed body and the particle module to describe particle motion and to model the collision.

This code is promising for its ease of implementation and flexibility. It showed a good stability and with the possibility of being massively parallelized is very fast. This code also opens the possibility of being extended including other features making it able to solve more complex and more realistic flows, for example including inelastic collision or more complex coupling mechanisms between the carrier and the particles.

Despite being a new implementation, the code shows good results from a qualitative point of view. The flow configuration that we solved proposes results similar to what is possible to find in literature for different values of the Stokes number. The results obtained are comparable with the references in terms of particles distribution over the vortexes, dependence on the Stokes number and bow shock formation.

6.1 Outlook

This code still requires some improvements and further validation, also from a quantitative point of view. After this, it is possible to implement inelastic collision to obtain faithful to reality results. This can be done by taking into account a restitution coefficient on both normal and tangential velocities to model the energy loss due to the collision [3] [16].

Another improvement is the implementation of the particle influence on the carrier, so to obtain a two-way coupling. Considering also the mutual interaction between particles it could be possible to implement a four-way coupling. These would allow to extend the validity of the code also for higher particles mass fraction and volume fraction (i.e. for dense systems).

Actually this code solves directly the Navier-Stokes equations, but it could be possible to implement a LES approach to solve more complex problems in reasonable time although a lower accuracy. Following this path a WM-LES approach could be taken into account to further reduce the computational cost [5].

Bibliography

[1] Fabien Anselmet and Roland Borghi. *Turbulent multiphase flows with heat and mass transfer*. eng. 1st ed. ISTE. London, England: ISTE Ltd, 2014.

[2] Christopher E. Brennen. *Fundamentals of Multiphase Flow*. Cambridge University Press, 2005, pp. i–iv.

[3] Ze Cao and Danesh K. Tafti. «Alternate method for resolving particle collisions in PRS of freely evolving particle suspensions using IBM». In: *International Journal of Multiphase Flow* 177 (2024), p. 104862.

[4] Pedro Costa. «A FFT-based finite-difference solver for massively-parallel direct numerical simulations of turbulent flows». In: *Computers & Mathematics with Applications* 76.8 (2018), pp. 1853–1862.

[5] A. Dehbi. «Validation against DNS statistics of the normalized Langevin model for particle transport in turbulent channel flows». In: *Powder Technology* 200.1 (2010), pp. 60–68.

[6] Andrea Di Mascio and Stefano Zaghi. «An immersed boundary approach for high order weighted essentially non-oscillatory schemes». In: *Computers & Fluids* 222 (2021), p. 104931.

[7] Milton Van Dyke. «An Album Fluid Motion». In: *Journal of Fluid Mechanics* 127 (1983), pp. 562–563.

[8] Michaelides E., Crowe C.T., and Schwarzkopf J.D. *Multiphase Flow Handbook*. 2nd ed. Boca Raton: CRC Press, 2016.

[9] ESA. *Mars dust storm*. Available on line, accessed 05-05-2024. 2018. URL: `https://www.esa.int/About_Us/ESAC/Mars_dust_storm`.

[10] ESA. *Martian dust storms churn up Earth-like clouds*. Available on line, accessed 05-05-2024. 2022. URL: https://www.esa.int/Science_Exploration/Space_Science/Mars_Express/Martian_dust_storms_churn_up_Earth-like_clouds.

[11] Amir Faghri and Yuwen Zhang. *Fundamentals of Multiphase Heat Transfer and Flow*. eng. 1st ed. 2020. Cham: Springer International Publishing, 2020.

[12] Joel H. Ferziger, Milovan Perić, and Robert L. Street. *Computational Methods for Fluid Dynamics*. 4th. Springer, 2019.

[13] Andrea Giusti, Francesco Lucci, and Alfredo Soldati. «Influence of the lift force in direct numerical simulation of upward/downward turbulent channel flow laden with surfactant contaminated microbubbles». In: *Chemical Engineering Science* 60.22 (2005). 7th International Conference on Gas-Liquid and Gas-Liquid-Solid Reactor Engineering, pp. 6176–6187.

[14] D. Goldstein, R. Handler, and L. Sirovich. «Modeling a No-Slip Flow Boundary with an External Force Field». In: *Journal of Computational Physics* 105.2 (1993), pp. 354–366.

[15] D. Goldstein, R. Handler, and L. Sirovich. «LES in complex geometries using boundary body forces». In: *Center for Turbulence Research - Proceedings of the Summer Program 1998* (1998), pp. 171–186.

[16] B. Imre, S. Räbsamen, and S.M. Springman. «A coefficient of restitution of rock materials». In: *Computers & Geosciences* 34.4 (2008), pp. 339–350.

[17] A. N. Kolmogorov. «Dissipation of Energy in the Locally Isotropic Turbulence». In: *Proceedings: Mathematical and Physical Sciences* 434.1890 (1991), pp. 15–17.

[18] A. N. Kolmogorov. «The Local Structure of Turbulence in Incompressible Viscous Fluid for Very Large Reynolds Numbers». In: *Proceedings: Mathematical and Physical Sciences* 434.1890 (1991), pp. 9–13.

[19] Ryoichi Kurose and Satoru Komori. «Drag and lift forces on a rotating sphere in a linear shear flow». In: *Journal of Fluid Mechanics* 384 (1999), pp. 183–206.

[20] Ming-Chih Lai and Charles S. Peskin. «An Immersed Boundary Method with Formal Second-Order Accuracy and Reduced Numerical Viscosity». In: *Journal of Computational Physics* 160.2 (2000), pp. 705–719.

[21] Cristian Marchioli, Maurizio Picciotto, and Alfredo Soldati. «Influence of gravity and lift on particle velocity statistics and transfer rates in turbulent vertical channel flow». In: *International Journal of Multiphase Flow* 33.3 (2007), pp. 227–251.

[22] Martin R. Maxey and James J. Riley. «Equation of motion for a small rigid sphere in a nonuniform flow». In: *The Physics of Fluids* 26.4 (Apr. 1983), pp. 883–889.

[23] John B. McLaughlin. «Inertial migration of a small sphere in linear shear flows». In: *Journal of Fluid Mechanics* 224 (1991), pp. 261–274.

[24] Rajat Mittal and Gianluca Iaccarino. «Immersed Bounday Methods». In: *Annual Review of Fluid Mechanics* 37 (2005), pp. 239–261.

[25] C. S. Peskin. «The Fluid Dynamics of Heart Valves: Experimental, Theoretical, and Computational Methods». In: *Annual Review of Fluid Mechanics* 14.Volume 14, 1982 (1982), pp. 235–259.

[26] Stephen B. Pope. *Turbulent Flows*. Cambridge University Press, 2000.

[27] Osborne Reynolds. «An Experimental Investigation of the Circumstances Which Determine Whether the Motion of Water Shall Be Direct or Sinuous, and of the Law of Resistance in Parallel Channels». In: *Philosophical Transactions of the Royal Society of London* 174 (1883), pp. 935–982.

[28] A. Sánchez-Lavega et al. «Cellular patterns and dry convection in textured dust storms at the edge of Mars North Polar Cap». In: *Icarus* 387 (2022), p. 115183.

[29] D. Schuster, E. Climent, and U. Rüde. «Particle laden flows around a circular cylinder from the hydrodynamic to granular regime». In: *International Journal of Multiphase Flow* 165 (2023), p. 104487.

[30] Zhaoyu Shi et al. «Bow shock clustering in particle-laden wetted cylinder flow». In: *International Journal of Multiphase Flow* 130 (2020), p. 103332.

[31] Heike Sommerfeld et al. «High velocity measurements of particle rebound characteristics under erosive conditions of high pressure compressors». In: *Wear* 470-471 (2021), p. 203626.

[32] Shefali Uttam et al. «Characteristics of convective vortices and dust devils at gale crater on Mars during MY33». In: *Planetary and Space Science* 213 (2022), p. 105430.

[33] *Measured Particle Rebound Characteristics Useful for Erosion Prediction*. Vol. 3: Coal, Biomass and Alternative Fuels; Combustion and Fuels; Oil and Gas Applications; Cycle Innovations. Turbo Expo: Power for Land, Sea, and Air. The American Society of Mechanical Enigneers. Apr. 1982.

[34] James R. Welty et al. *Fundamentals of Momentum, Heat and Mass Transfer, 5th Edition*. Wiley Global Education, 2007.

[35] Jun Xia, K. Luo, and Suresh Kumar. «Large-Eddy Simulation of Interactions Between a Reacting Jet and Evaporating Droplets». In: *Flow, Turbulence and Combustion* 80 (July 2008), pp. 133–153.

[36] T. Ye et al. «An Accurate Cartesian Grid Method for Viscous Incompressible Flows with Complex Immersed Boundaries». In: *Journal of Computational Physics* 156.2 (1999), pp. 209–240.

[37] Weihong Zhang and Ying Zhou. «Chapter 2 - Level-set functions and parametric functions». In: *The Feature-Driven Method for Structural Optimization*. Ed. by Weihong Zhang and Ying Zhou. Elsevier, 2021, pp. 9–46.