POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

An Immersed Boundary method for fish-like swimming simulations

Tesi di Laurea Magistrale in
Ingegneria Matematica

Author: **Cássio Murakami**

Student ID: 990364
Advisor: Prof. Nicola Parolini
Co-advisors: dr. Giorgio Negrini
Academic Year: 2022-23

Abstract

The study of flows over immersed structures is a widely researched field with significant relevance in various industrial sectors and applications. This work focuses on the modelling and simulation of external flows over deformable bodies with prescribed kinematics. The mathematical model of the fluid employs the incompressible Navier-Stokes equations, adapted to incorporate the interaction with the immersed body. The numerical solution is implemented using the *open-source* software OpenFOAM, which utilizes the Finite Volume Method to discretize the problem. Additionally, the Immersed Boundary Method is employed to handle meshes that do not conform to the structure.

The numerical verification of the scheme and the calculation of forces were conducted using benchmark cases involving two-dimensional external flow around a cylinder and travelling wavy foils. Subsequently, the study focused on modelling the movement of a deformable body, reproducing the swimming of a fish while considering the constraints of inextensibility and mass conservation.

Throughout two-dimensional simulations of the fish-like swimming, an analysis was conducted on the effects of oscillation frequency, movement amplitude, and motion pattern. This analysis encompassed the computation of thrust force and the examination of the vorticity pattern in the wake, revealing its impacts on the swim. Ultimately, three-dimensional simulations were executed utilizing a realistic fish geometry, featuring a particular focus on the topology of the wake.

Keywords: Computational Fluid Dynamics, Fish swimming, Immersed Boundary Method

Abstract in lingua italiana

Lo studio dei flussi su strutture immerse è un campo ampiamente studiato e di notevole rilevanza in diversi settori industriali e applicazioni. Questo lavoro si concentra sulla modellazione e sulla simulazione di flussi esterni su corpi deformabili con cinematica prescritta. Il modello matematico del fluido impiega le equazioni di Navier-Stokes incomprimibili, adattate per incorporare l'interazione con il corpo immerso. La soluzione numerica è implementata utilizzando il software *open-source* OpenFOAM, che utilizza il metodo dei volumi finiti per discretizzare il problema. Inoltre, il Metodo della Superficie Immersa viene utilizzato per gestire le maglie non conformi alla struttura.

La verifica numerica dello schema e il calcolo delle forze sono stati condotti utilizzando casi di riferimento che coinvolgevano un flusso esterno bidimensionale attorno a un cilindro e lame ondulate in movimento. Successivamente, lo studio si è concentrato sulla modellazione del movimento di un corpo deformabile, riproducendo il nuoto di un pesce e considerando i vincoli di inestensibilità e conservazione della massa.

Durante le simulazioni bidimensionali del nuoto simile a quello di un pesce, è stata condotta un'analisi degli effetti della frequenza di oscillazione, dell'ampiezza del movimento e del modello di movimento. Questa analisi ha incluso il calcolo della forza di spinta e l'esame del modello di vorticità nella scia, rivelando il suo impatto sul nuoto. Infine, sono state eseguite simulazioni tridimensionali utilizzando una geometria realistica del pesce, con una particolare attenzione alla topologia della scia.

Parole chiave: Fluidodinamica Computazionale, Nuoto dei pesci, Metodo della Superficie Immersa

Contents

1 | Introduction

The study of fluid dynamics enables the analysis of the effects on an immersed body in a flow domain, which is relevant for numerous aerodynamic applications [28]. In particular, the modelling and simulation of fish-like swimming are not only of interest to biology but also hold relevance in engineering applications. Within the realm of applied sciences, the analysis of this scenario finds utility in optimizing underwater propulsion and enhancing aerodynamic designs to augment maneuverability performance [2].

In the context of empirical data, the study of fish swimming kinematics has been widely explored by biologists and analyzed using the Particle Image Velocimetry technique (PIV), as demonstrated in [29, 44], aiming to investigate the flow structure of the wake. In the realm of Computational Fluid Dynamics (CFD), in which results are obtained through computational simulations, it is customary to utilize previously acquired kinematic data to approximate fish motion. This approximation often involves the use of undulatory backward-travelling waves, as explained in [18, 46].

The numerical simulations of fish-like swimming can be classified into two classes. The first class investigates fish locomotion considering the self-propulsion. This approach considers the feedback of fluid forces acting on the body, which in turn generates a propulsive force and momentum that causes a change of motion in the structure. Studies that contemplate this case are [2, 3, 24]. The present work belongs to the second category of investigation, which focuses on evaluating the fixed swimming approach scenario. Within this category, studies such as [13] fix a flexible structure at a specific point in a uniform external flow and then impose the kinematic of the structure.

Additional studies have been conducted to gain a deeper understanding of fish motion, as demonstrated in works such as [9, 23]. These studies delve into the effects of multiple fish swimming nearby, particularly focusing on the wake generated by neighboring fish. Their objective is to determine whether the school of fish benefits from mutual motion. Furthermore, research efforts have expanded to include more complex three-dimensional simulations, as seen in the studies by [6, 8]. These simulations involve the creation of a more realistic model that incorporates refined geometry resembling the physiological

characteristics of fish.

To solve the system of partial differential equations arising from the physical principles of the real-world scenario, it is fundamental to establish a computational mesh that discretizes the space domain [39]. One approach is to adopt a background mesh that conforms to the geometry of the solid, aligning the grid with the body and enabling the immediate imposition of coupling conditions on the shared interface. A demonstration of this concept is found in the application of the Arbitrary Lagrange Eulerian Method (ALE) [10, 11, 16]. On the other hand, the present study will utilize an alternative approach that employs a fixed background mesh with a structured grid not necessarily aligned with the fluid-solid interface. This requires enforcing boundary conditions through numerical approximations, as performed by the Immersed Boundary Method (IBM) [34–36], which is an important approach that can be used to obtain the solution of a variety of problems [21, 37, 41, 42].

The objective of the present work is to apply the Immersed Boundary Method to investigate fish-like swimming. Focusing on the fixed swimmer method, a flapping airfoil is positioned in a uniform flow to analyze both the thrust generated by the induced flow and the wake structure. Utilizing the IBM developed in [30] within the OpenFOAM software, we will construct a numerical algorithm to prescribe the structure motion while considering the constraints of inextensibility and mass conservation.

The present study will be divided into various chapters. Chapter 2 will focus on developing the mathematical equations derived from the underlying physical principles. Its main objective is to derive the incompressible Navier-Stokes equations, taking into consideration the influence of an immersed body.

Chapter 3 is dedicated to the development of the Finite Volume Method (FVM) framework for the problem. It will detail the projection method that will be employed to discretize in space and solve the incompressible Navier-Stokes equations. Additionally, it will provide an overview of the Immersed Boundary Method and illustrate how it can be applied algebraically to implement the immersed boundary condition.

Chapter 4 will focus on modelling the fish-like swimming kinematics, covering aspects such as geometry, mass conservation, and backbone length conservation. It will also provide a detailed explanation of the algorithm implemented to achieve this objective.

Chapter 5 will verify the OpenFOAM Immersed Boundary Method algorithm through benchmark external flow scenarios involving a fixed cylinder. Subsequently, the chapter will present the results of the travelling wave foil simulations, encompassing both flat

plate and fish-body airfoil configurations.

Chapter 6 will present the results of the two-dimensional fish-like swimming simulations in OpenFOAM, taking into account the inextensibility and mass conservation constraints. Additionally, it will feature three-dimensional simulations and provide a discussion of the results.

2 | Fluid dynamics model

In this chapter, our main objective is to introduce the differential equations that govern fluid dynamics [14, 28]. We begin by defining the characteristics of the flow and subsequently introduce specific hypotheses. By applying these assumptions, we can derive equations from the physical principles, aiming to obtain the system of partial differential equations known as the incompressible Navier-Stokes equations.

We proceed to derive the Navier-Stokes equations, drawing from the principles of mass and momentum conservation, focusing, in particular on the case of incompressible flows. Then, we introduce the Immersed Boundary method, which is instrumental in describing the presence of an immersed body within the fluid medium.

For this discussion, assume that the domain occupied by the fluid at time $t \in [0, +\infty)$ is $D_t \subset \mathbb{R}^d$, where $d = 2, 3$ represents the spatial dimension of the problem. Let $\widehat{D} = D_0$ be the initial configuration of the fluid domain. The mapping between $\widehat{D}$ and D_t is defined by the Eulerian velocity field $\mathbf{u}(\mathbf{x}, t) : D_t \times [0, +\infty) \to \mathbb{R}^d$ as follows:

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{u}(\mathbf{x}, t), \quad \forall (\mathbf{x}, t) \in D_t \times [0, +\infty). \\ \mathbf{x} = \hat{\mathbf{x}}, \quad \forall \mathbf{x} \in \widehat{D},\ t = 0. \end{cases}$$

Let $\mathcal{L}_t : \widehat{D} \to D_t$ be the Lagrangian map that relates the point in the reference configuration $\hat{\mathbf{x}}$ to the corresponding point $\mathbf{x} = \mathcal{L}_t(\hat{\mathbf{x}})$ at time t in the deformed configuration. It is worth mentioning that if the velocity field $\mathbf{u}$ is regular enough, the map $\mathcal{L}_t$ is invertible. For visualization, consider Figure 2.1, which illustrates the correlation between points that are mapped in a scenario that involves arbitrary sections within the fluid domain, namely $\widehat{V} \subseteq \widehat{D}$, and $V_t \subseteq D_t$.

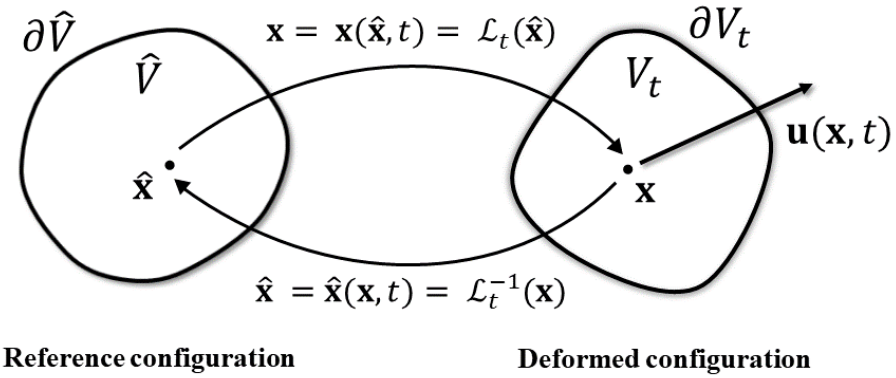


Figure 2.1: Map $\mathcal{L}_t$ between the reference configuration $\widehat{V}$ and the deformed configuration V_t.

To derive the fluid dynamic equations, we recall the Divergence Theorem and the Reynolds Transport Theorem that are going to be extensively applied in the formulations of this chapter.

Theorem 2.1. *(Divergence Theorem) Let V_t be a volume in space enclosed by a surface ∂V_t. Given a regular function $f : V_t \times [0, +\infty) \to \mathbb{R}$, a regular vector field $\mathbf{v} : V_t \times [0, +\infty) \to \mathbb{R}^d$, and a regular tensor field $\mathrm{T} : V_t \times [0, +\infty) \to \mathbb{R}^{d\times d}$, then*

$$\begin{aligned}\int_{V_t} (\nabla f)\mathrm{d}V &= \oint_{\partial V_t} (\mathbf{n}f)\mathrm{d}S,\\ \int_{V_t} (\nabla \cdot \mathbf{v})\mathrm{d}V &= \oint_{\partial V_t} (\mathbf{n} \cdot \mathbf{v})\mathrm{d}S,\\ \int_{V_t} (\nabla \cdot \mathrm{T})\mathrm{d}V &= \oint_{\partial V_t} (\mathrm{T}\mathbf{n})\mathrm{d}S.\end{aligned}$$

where $\mathbf{n}$ denotes the outward normal of ∂V_t.

Theorem 2.2. *(Reynolds Transport Theorem) Given the Eulerian velocity field $\mathbf{u}$, the Lagrangian map $\mathcal{L}_t$, and an arbitrary fluid portion V_t, then*

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} f\mathrm{d}V &= \int_{V_t} \frac{\partial f}{\partial t}\mathrm{d}V + \oint_{\partial V_t} f\mathbf{u} \cdot \mathbf{n}\mathrm{d}S,\\ \frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} \mathbf{v}\mathrm{d}V &= \int_{V_t} \frac{\partial \mathbf{v}}{\partial t}\mathrm{d}V + \oint_{\partial V_t} \mathbf{v}\mathbf{u} \cdot \mathbf{n}\mathrm{d}S,\\ \frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} \mathrm{T}\mathrm{d}V &= \int_{V_t} \frac{\partial \mathrm{T}}{\partial t}\mathrm{d}V + \oint_{\partial V_t} \mathrm{T}\mathbf{u} \cdot \mathbf{n}\mathrm{d}S.\end{aligned}$$

2.1. Incompressibility

The incompressibility hypothesis states that the volume of a given portion of the fluid is constant over time. In other words, the material derivative of the volume of any arbitrary partition $V_t \subseteq D_t$ is zero:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} \mathrm{d}V = 0. \tag{2.1}$$

Consider the Divergence Theorem 2.1 and the Reynolds Transport Theorem 2.2 in the particular case of a constant material element $f(\mathbf{x},t) : D_t \times [0,+\infty) \to \mathbb{R}$, such that $f(\mathbf{x},t) = 1,\ \forall(\mathbf{x},t) \in D_t \times [0,+\infty)$,

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} \mathrm{d}V = \frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} f\mathrm{d}V = \int_{V_t}\left(\frac{\partial f}{\partial t} + \nabla\cdot(f\mathbf{u})\right)\mathrm{d}V = \int_{V_t}\nabla\cdot\mathbf{u}\,\mathrm{d}V. \tag{2.2}$$

Then, evaluate (2.2) using the constraint (2.1), which results in the incompressibility integral formulation:

$$\int_{V_t}\nabla\cdot\mathbf{u}\,\mathrm{d}V = 0.$$

Since the integration is zero for any arbitrary V_t, it implies that the integrand should be null in D_t [15]. Finally, it is possible to develop the differential form of the incompressibility hypothesis:

$$\nabla\cdot\mathbf{u} = 0, \quad \forall(\mathbf{x},t) \in D_t \times [0,+\infty). \tag{2.3}$$

2.2. Mass conservation

Let $\rho(\mathbf{x},t) : D_t \times [0,+\infty) \to \mathbb{R}^+$ be the mass density of the fluid. The principle of mass conservation states that the total mass of a body remains constant throughout its motion. This means that the material derivative of the mass of any arbitrary volume $V_t \subseteq D_t$ is zero:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} \rho\mathrm{d}V = 0. \tag{2.4}$$

Consider the Divergence Theorem 2.1 and Reynolds Transport Theorem 2.2, then

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t} \rho\mathrm{d}V = \int_{V_t}\left(\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\mathbf{u})\right)\mathrm{d}V. \tag{2.5}$$

Thus, considering the property (2.4) evaluated at (2.5) it is obtained the integral formu-

lation of the mass conservation:

$$\int_{V_t} \left(\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) \right) \mathrm{d}V = 0.$$

Since the integration is zero for any arbitrary V_t, it is possible to state the differential form of the mass conservation principle:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \quad \forall (\mathbf{x}, t) \in D_t \times [0, +\infty). \tag{2.6}$$

Finally, using the incompressibility result (2.3), the differential form of mass conservation is reduced to

$$\frac{\mathrm{d}\rho}{\mathrm{d}t} = \frac{\partial \rho}{\partial t} + \mathbf{u} \cdot \nabla \rho = 0, \quad \forall (\mathbf{x}, t) \in D_t \times [0, +\infty),$$

which states that the density is constant along the trajectory of the flow.

2.3. Linear momentum conservation

Consider an arbitrary partition of the fluid domain denoted as $V_t \subseteq D_t$. In this context, let $\mathbf{f}(\mathbf{x}, t) : V_t \times [0, +\infty) \to \mathbb{R}^d$ represent the volume forces per unit mass acting within the fluid. To describe the behavior of the surface forces, we invoke the Cauchy principle, which asserts the existence of a stress field $\mathbf{S}(\mathbf{x}, t, \mathbf{n}) : D_t \times [0, +\infty) \times \mathcal{N} \to \mathbb{R}^d$, where $\mathcal{N} = \{\mathbf{n} \in \mathbb{R}^d : |\mathbf{n}| = 1\}$. Therefore, we can define the surface force per unit area that acts on the partition $\mathbf{s}(\mathbf{x}, t, \mathbf{n}) : \partial V_t \times [0, +\infty) \to \mathbb{R}^d$, where $\mathbf{n}$ is the outward normal vector.

According to Newton's second law, the rate of change of a body's linear momentum is directly proportional to the resultant force exerted upon it. For the body V_t, the linear momentum can be expressed as the integral of the quantity $\rho \mathbf{u}$ over the entire body.

The resultant of the volume forces can be determined by evaluating the integral of $\rho \mathbf{f}$ over the domain, while the resultant of the surface forces is obtained through integration of $\mathbf{s}$ along the surface of the body:

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{V_t} \rho \mathbf{u} \mathrm{d}V = \int_{V_t} \rho \mathbf{f} \mathrm{d}V + \oint_{\partial V_t} \mathbf{s} \mathrm{d}S. \tag{2.7}$$

To derive the integral form of the linear momentum equation, we apply the Reynolds

Transport Theorem 2.2:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_t}\rho\mathbf{u}\mathrm{d}V = \int_{V_t}\frac{\partial\rho\mathbf{u}}{\partial t}\mathrm{d}V + \oint_{\partial V_t}\rho\mathbf{u}(\mathbf{u}\cdot\mathbf{n})\mathrm{d}S.$$

Next, we consider the product identity involving a scalar and a dyadic:

$$\mathbf{u}(\mathbf{u}\cdot\mathbf{n}) = (\mathbf{u}\otimes\mathbf{u})\mathbf{n}. \tag{2.8}$$

Proof. $[\mathbf{u}(\mathbf{u}\cdot\mathbf{n})]_i = u_i\sum_{j=1}^{d}(u_j n_j) = \sum_{j=1}^{d}(u_i u_j)n_j.$ □

Then, using the identity (2.8), and then applying the Divergence Theorem 2.1 for tensors we have that:

$$\oint_{\partial V_t}\rho\mathbf{u}(\mathbf{u}\cdot\mathbf{n})\mathrm{d}S = \oint_{\partial V_t}(\rho\mathbf{u}\otimes\mathbf{u})\mathbf{n}\mathrm{d}S = \int_{V_t}\nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u})\mathrm{d}V.$$

Let $\mathrm{T}(\mathbf{x},t) : V_t\times[0,+\infty)\to\mathbb{R}^{d\times d}$ be the stress tensor field. Then, considering the Cauchy Stress Theorem [17] and applying the Divergence Theorem 2.1 for second order tensors we obtain:

$$\oint_{\partial V_t}\mathbf{s}\mathrm{d}S = \oint_{\partial V_t}\mathrm{T}\mathbf{n}\mathrm{d}S = \int_{V_t}\nabla\cdot\mathrm{T}\mathrm{d}V.$$

Finally, applying the results in (2.7) we arrive at the integral form of the linear momentum conservation:

$$\int_{V_t}\frac{\partial\rho\mathbf{u}}{\partial t}\mathrm{d}V + \int_{V_t}\nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u})\mathrm{d}V = \int_{V_t}\rho\mathbf{f}\mathrm{d}V + \int_{V_t}\nabla\cdot\mathrm{T}\mathrm{d}V. \tag{2.9}$$

The relation (2.9) must hold for any arbitrary portion $V_t\subseteq D_t$, which leads us to the differential form of the momentum equation in the conservative formulation:

$$\frac{\partial\rho\mathbf{u}}{\partial t} + \nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u}) = \rho\mathbf{f} + \nabla\cdot\mathrm{T}, \quad \forall(\mathbf{x},t)\in D_t\times[0,+\infty). \tag{2.10}$$

Considering the mass conservation relation (2.6), it is possible to analyze the time derivative of the linear momentum:

$$\frac{\partial\rho\mathbf{u}}{\partial t} = \frac{\partial\rho}{\partial t}\mathbf{u} + \rho\frac{\partial\mathbf{u}}{\partial t} = \rho\frac{\partial\mathbf{u}}{\partial t} - (\nabla\cdot\rho\mathbf{u})\mathbf{u}. \tag{2.11}$$

Moreover, it is possible to derive a formulation for the divergence of the dyadic product:

$$\nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = (\nabla \cdot \rho \mathbf{u})\mathbf{u} + \rho(\mathbf{u} \cdot \nabla)\mathbf{u}. \tag{2.12}$$

Proof. $[\nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u})]_i = \sum_{j=1}^{d} \frac{\partial(\rho u_i u_j)}{\partial x_j} = \sum_{j=1}^{d} \left(\frac{\partial(\rho u_j)}{\partial x_j} u_i + \rho u_j \frac{\partial u_i}{\partial x_j} \right).$ □

Finally, by combining the mathematical identities presented in (2.11) and (2.12) with the previous formulation (2.10), we arrive at the non-conservative formulation of the linear momentum equation:

$$\rho \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} \right) = \rho \mathbf{f} + \nabla \cdot \mathrm{T}, \quad \forall (\mathbf{x}, t) \in D_t \times [0, +\infty). \tag{2.13}$$

2.4. Angular momentum conservation

Consider a fixed pole $\mathbf{o} \in \mathbb{R}^d$ and define the vector $\mathbf{r}(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}^d$ such that $\mathbf{r}(\mathbf{x}) = \mathbf{x} - \mathbf{o}$. According to the angular momentum conservation principle, any change in the angular momentum of the body necessitates the application of an external torque:

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{V_t} \mathbf{r} \times \rho \mathbf{u} \mathrm{d}V = \int_{V_t} \mathbf{r} \times \rho \mathbf{f} \mathrm{d}V + \oint_{\partial V_t} \mathbf{r} \times \mathrm{T}\mathbf{n} \mathrm{d}S. \tag{2.14}$$

Consequently, it is possible to demonstrate that, for a vector field, the total derivative can be interchanged with the integral operation [17]. By subsequently applying the identity for the derivative of the vector product and utilizing the definition of the Eulerian velocity field $\mathbf{u} = {}^{\mathrm{d}\mathbf{x}}/_{\mathrm{d}t}$, we arrive at the following result:

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{V_t} \mathbf{r} \times \rho \mathbf{u} \mathrm{d}V = \int_{V_t} \left(\frac{\mathrm{d}\mathbf{r}}{\mathrm{d}t} \times \rho \mathbf{u} + \mathbf{r} \times \rho \frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} \right) \mathrm{d}V = \int_{V_t} \mathbf{r} \times \rho \frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} \mathrm{d}V.$$

It is important to highlight that this equality holds even when considering its scalar product with an arbitrary vector $\mathbf{w} \in \mathbb{R}^d$,

$$\int_{V_t} \mathbf{w} \cdot \mathbf{r} \times \rho \frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} \mathrm{d}V = \int_{V_t} \mathbf{w} \cdot \mathbf{r} \times \rho \mathbf{f} \mathrm{d}V + \oint_{\partial V_t} \mathbf{w} \cdot \mathbf{r} \times \mathrm{T}\mathbf{n} \mathrm{d}S.$$

Then, using the property of mixed product that allows cyclic permutation of factors, we obtain:

$$\int_{V_t} \mathbf{w} \times \mathbf{r} \cdot \rho \frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} \mathrm{d}V = \int_{V_t} \mathbf{w} \times \mathbf{r} \cdot \rho \mathbf{f} \mathrm{d}V + \oint_{\partial V_t} \mathbf{w} \times \mathbf{r} \cdot \mathrm{T}\mathbf{n} \mathrm{d}S.$$

Using an algebraic manipulation of tensorial calculus and the Divergence Theorem 2.1 it is possible to develop the surface integral into a volume integral:

$$\oint_{\partial V_t} \mathbf{w} \times \mathbf{r} \cdot \mathrm{T}\mathbf{n}\mathrm{d}S = \oint_{\partial V_t} \mathrm{T}^T(\mathbf{w} \times \mathbf{r}) \cdot \mathbf{n}\mathrm{d}S = \int_{V_t} \nabla \cdot (\mathrm{T}^T(\mathbf{w} \times \mathbf{r}))\mathrm{d}V. \tag{2.15}$$

Furthermore, consider the mathematical identity involving the divergence of the transpose of a tensorial operation:

$$\nabla \cdot (\mathrm{T}^T\mathbf{v}) = (\nabla \cdot \mathrm{T}) \cdot \mathbf{v} + \mathrm{T} : \nabla\mathbf{v}. \tag{2.16}$$

Proof. $\nabla \cdot (\mathrm{T}^T\mathbf{v}) = \sum_i \sum_k \frac{\partial T_{ki} v_k}{\partial x_i} = \sum_i \sum_k \left(\frac{\partial T_{ki}}{\partial x_i} v_k + T_{ki} \frac{\partial v_k}{\partial x_i} \right).$ □

Finally, applying the identity (2.16) into (2.15) considering $\mathbf{v} = \mathbf{w} \times \mathbf{r}$, and substituting the results in (2.14) we obtain that:

$$\int_{V_t} (\mathbf{w} \times \mathbf{r}) \cdot \left(\rho \frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} - \rho \mathrm{f} - \nabla \cdot \mathrm{T} \right) \mathrm{d}V = \int_{V_t} \mathrm{T} : \nabla(\mathbf{w} \times \mathbf{r})\mathrm{d}V.$$

Thus, using the local formulation of the linear momentum balance (2.13) and the Reynolds Transport Theorem 2.2 it is possible to show that the left-hand side of the equation is zero. Since it must hold for any generic partition V_t it is possible to obtain the differential formulation for the conservation of angular momentum:

$$\mathrm{T} : \nabla(\mathbf{w} \times \mathbf{r}) = 0, \quad \forall (\mathbf{x}, t) \in D_t \times [0, +\infty). \tag{2.17}$$

It is possible to prove that $\nabla(\mathbf{w} \times \mathbf{r})$ is the antisymmetric tensor associated with the arbitrary vector $\mathbf{w}$ [17]. Therefore, for (2.17) to hold true for any generic vector $\mathbf{w}$ the tensor T must be symmetric:

$$\mathrm{T} = \mathrm{T}^T, \quad \forall (\mathbf{x}, t) \in D_t \times [0, +\infty).$$

2.5. Constitutive relation

Assume that the Cauchy stress tensor T is continuously related to $\nabla\mathbf{u}$, and that as shown in section 2.4, the stress tensor T is symmetric. Furthermore, consider the principle of frame indifference, which states that the material response should be essentially indepen-

dent of the observer:

$$\mathrm{Q}\mathrm{T}(\nabla\mathbf{u})\mathrm{Q}^T = \mathrm{T}(\mathrm{Q}\nabla\mathbf{u}\mathrm{Q}^T), \quad \forall \mathrm{Q} \in \mathcal{R} \tag{2.18}$$

where $\mathcal{R} = \{\mathrm{Q} \in \mathrm{R}^{d\times d} : \mathrm{Q}^T = \mathrm{Q}^{-1}\}$.

Then, it is possible to show that the stress tensor T depends only on the symmetric part of $\nabla\mathbf{u}$, which is the rate of deformation tensor D defined as

$$\mathrm{D}(\mathbf{u}) = \mathrm{sym}(\nabla\mathbf{u}) = \frac{\nabla\mathbf{u} + \nabla^T\mathbf{u}}{2}.$$

Let $\alpha_i(\mathrm{I_D}, \mathrm{II_D}, \mathrm{III_D}) : \mathbb{R}^3 \to \mathbb{R}$, $i = 1, 2, 3$, be coefficients that depends exclusively on the invariants of D, where $\mathrm{I_D} = \mathrm{tr(D)}, \mathrm{II_D} = \frac{1}{2}((\mathrm{tr(D)})^2 - \mathrm{tr}(\mathrm{D}^2)), \mathrm{III_D} = \det(\mathrm{D})$. Then, in order to satisfy (2.18) considering the dependency only the symmetric part of $\nabla\mathbf{u}$, we conclude that T must possess the following structure:

$$\mathrm{T} = \alpha_1(\mathrm{I_D}, \mathrm{II_D}, \mathrm{III_D})\mathrm{I} + \alpha_2(\mathrm{I_D}, \mathrm{II_D}, \mathrm{III_D})\mathrm{D} + \alpha_3(\mathrm{I_D}, \mathrm{II_D}, \mathrm{III_D})\mathrm{D}^2.$$

Consider the assumption of a Newtonian fluid, that states that the relation between T and D is linear. Because of the linear relation $\alpha_3 = 0$, and also α_1 and α_2 must not depend on the nonlinear invariants $\mathrm{II_D}$ and $\mathrm{III_D}$. Moreover, applying the incompressibility hypothesis (2.3), then $\mathrm{I_D} = \mathrm{tr(D)} = \nabla \cdot \mathbf{u} = 0$. As a result, the coefficients α_i remain independent of the tensor D.

Let $p(\mathbf{x}, t) : D_t \times [0, +\infty) \to \mathbb{R}$ be the pressure field, and $\mu(\mathbf{x}, t) : D_t \times [0, +\infty) \to \mathbb{R}^+$ be the dynamic viscosity. Finally, consider that with no deformation, the internal stresses rely only on the pressure field, such that $\mathrm{T}(0) = -p\mathrm{I}$, and let $\alpha_1 = -p$, $\alpha_2 = 2\mu$. Thus we obtain the constitutive relation of the incompressible Newtonian fluid:

$$\mathrm{T} = -p\mathrm{I} + 2\mu\mathrm{D}(\mathbf{u}). \tag{2.19}$$

Additionally, examine the vector identity involving the divergence of the transpose of the gradient of a vector:

$$\nabla \cdot (\nabla^T\mathbf{u}) = \nabla(\nabla \cdot \mathbf{u}). \tag{2.20}$$

Proof. $[\nabla \cdot (\nabla^T\mathbf{u})]_i = \sum_{j=1}^{d} \frac{\partial}{\partial x_j}\left(\frac{\partial u_j}{\partial x_i}\right) = \sum_{j=1}^{d} \frac{\partial}{\partial x_i}\left(\frac{\partial x_j}{\partial x_j}\right) = \frac{\partial}{\partial x_i}(\nabla \cdot \mathbf{u})$. □

Considering the hypothesis of a homogeneous fluid, which implies that the viscosity is constant in space, together with the result (2.20) and the incompressibility constraint

(2.3), we can deduce that:

$$\nabla \cdot (2\mu \mathrm{D}(\mathbf{u})) = 2\mu \nabla \cdot \left(\frac{\nabla \mathbf{u} + \nabla^T \mathbf{u}}{2} \right) = \mu \Delta \mathbf{u}. \tag{2.21}$$

Finally, to incorporate the constitutive relation into the linear momentum equation (2.13), we apply the divergence operator to (2.19) while considering the identity (2.21):

$$\nabla \cdot \mathrm{T} = -\nabla p + \nabla \cdot (2\mu \mathrm{D}(\mathbf{u})) = -\nabla p + \mu \Delta \mathbf{u}. \tag{2.22}$$

2.6. The incompressible Navier-Stokes equations

Consider the incompressibility hypothesis (2.3), the momentum equation in the non-conservative form (2.13), and the constitutive model for Newtonian fluids (2.22). Taking into account the negligible temperature-dependence of fluid properties, the continuity equation and momentum equation are decoupled from the energy equation [14]. Consequently, in this study, the analysis and modelling of the energy equation are excluded, focusing primarily on the continuity and momentum equations.

To analyze the fluid dynamics in a fixed domain, consider an Eulerian approach, where $\Omega \subset D_t,\ \forall t \in [0, +\infty)$ is a stationary domain filled by the fluid at every time instant, with a Lipschitz boundary $\partial\Omega$. To define the boundary conditions consider a partition of the boundary such that $\partial\Omega = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$. Then, let $\mathbf{g}(\mathbf{x}, t) : \Gamma_D \times [0, +\infty) \to \mathbb{R}^d$ be the Dirichlet boundary condition, $\boldsymbol{\psi}(\mathbf{x}, t) : \Gamma_N \times [0, +\infty) \to \mathbb{R}^d$ be the Neumann boundary condition, and $\mathbf{n}$ the outward normal vector.

Ultimately, by dividing the linear momentum equation by ρ, we obtain a well-defined system of partial differential equations describing the differential form of the incompressible Navier-Stokes equations:

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} - \nu \Delta \mathbf{u} + \dfrac{1}{\rho} \nabla p = \mathbf{f}, \quad \text{in } \Omega,\ t > 0 \\ \nabla \cdot \mathbf{u} = 0, \quad \text{in } \Omega,\ t > 0 \\ \mathbf{u}(x, 0) = \mathbf{u}_0(\mathbf{x}), \quad \text{in } \Omega,\ t = 0 \\ \mathbf{u} = \mathbf{g}, \quad \text{on } \Gamma_D,\ t > 0 \\ \mu \dfrac{\partial \mathbf{u}}{\partial \mathbf{n}} - p\mathbf{n} = \boldsymbol{\psi}, \quad \text{on } \Gamma_N,\ t > 0 \end{cases} \tag{2.23}$$

where $\nu(\mathbf{x}, t) : \Omega \times [0, +\infty) \to \mathbb{R}^+$ is the kinematic viscosity defined as $\nu = \mu/\rho$.

2.7. Immersed Boundary Method

To advance the mathematical modelling of the differential problem concerning fish-like swimming, it is imperative to establish a cohesive relationship between the fluid and the structure [36]. In the present study, we focus on formulating the coupling, disregarding the development of the elasto-dynamics partial differential equations that govern the structure. We make this decision based on the premise that the displacement of the structure is determined by a pre-defined analytical function.

The Immersed Boundary Method is an efficient technique for Fluid-Structure Interaction. Introduced by Peskin [34] in the field of biological fluid dynamics, it is an approach for managing boundary conditions at different interfaces, including fluid-solid boundaries. This method is particularly valuable when dealing with non-body-conforming meshes, as it offers an effective solution by maintaining a structured and static background mesh throughout the simulation. It can effectively handle complex displacement scenarios, as it eliminates the challenges associated with complex geometries and distorted mesh arrangements often encountered in adaptive mesh conforming methods [40].

Let $\Omega \subset \mathbb{R}^d$ be the domain encompassing the entire physical scenario. We can partition the entire domain into two separate and non-overlapping subdomains at time t: $\Omega = \Omega_f^t \cup \Omega_s^t$, where Ω_f^t represents the fluid domain and Ω_s^t corresponds to the solid domain. Additionally, we define $\Gamma^t = \Omega_f^t \cap \Omega_s^t$ as the interface between these two domains. To enhance clarity within our analysis, we adopt the notations $\widehat{\Omega}_f = \Omega_f^0$ and $\widehat{\Omega}_s = \Omega_s^0$ to represent the reference configurations of the fluid and solid domains, respectively, at the initial time.

In the context of the present study, we consider a prescribed motion for the structure, in which a Lagrangian map to the reference state $\mathbf{X}(\hat{\mathbf{x}}, t) : \widehat{\Omega}_s \times [0, +\infty) \to \Omega_s^t$ is defined. Notice that the structure displacement is also prescribed, given that $\mathbf{d}(\mathbf{X}, t) = \mathbf{X}(\hat{\mathbf{x}}, t) - \hat{\mathbf{x}}$. Thus, considering the non-slip condition, we derive the kinematic compatibility condition, which imposes that the velocity of the fluid at the interface must coincide with the velocity of the corresponding point at the structure surface:

$$\frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t} = \mathbf{u}(\mathbf{X}, t), \quad \text{in } \Gamma^t, \ t > 0.$$

To establish the dynamic interface condition, we apply Newton's third law, which states that forces at the interface must be equal in magnitude and opposite. Therefore, we consider an exchange of forces, denoted as $\mathbf{f}_{\Gamma^t}(\mathbf{x}, t) : \Omega_f^t \times [0, +\infty) \to \mathbb{R}^d$, occurring only at the interface between the fluid and solid. It is important to note that, for the sake of

the assumptions of prescribed displacement of the structure, we will not delve into the detailed nature of these forces.

Hence, by considering these factors, we establish a representation of the fluid dynamic equations in the context of incompressible Navier-Stokes flow interacting with an immersed structure with prescribed motion:

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} - \nu\Delta\mathbf{u} + \dfrac{1}{\rho}\nabla p = \mathbf{f} + \delta_{\Gamma^t}\mathbf{f}_{\Gamma^t}, \quad \text{in } \Omega_f^t,\ t>0 \\ \nabla\cdot\mathbf{u} = 0, \quad \text{in } \Omega_f^t,\ t>0 \\ \mathbf{u} = \dfrac{\mathrm{d}\mathbf{X}}{\mathrm{d}t}, \quad \text{in } \Gamma^t,\ t>0 \\ + \text{Boundary Conditions on } \partial\Omega_f^t/\Gamma^t \\ + \text{Initial Conditions} \end{cases} \tag{2.24}$$

where δ_{Γ^t} is the Dirac delta function with support on Γ^t.

Regarding the fluid forces that act on the immersed body, let F_D represent the drag force and F_L represent the lift force. These forces are computed by evaluating a surface integral of the fluid stress tensor projected in the directions parallel and orthogonal to the freestream, as illustrated in Figure 2.2.

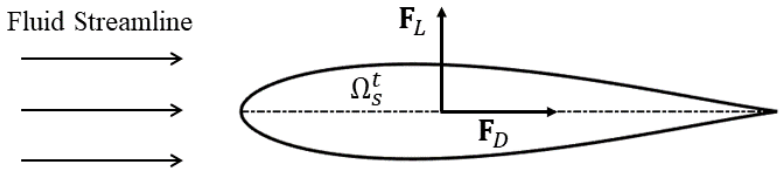


Figure 2.2: 2D representation of the drag and lift forces acting on the structure.

Consider $\mathbf{e}_x$ as the unit vector parallel to the freestream, and $\mathbf{e}_y$ as the vertical unit vector normal to the flow direction. The forces can then be calculated by applying the expressions:

$$F_D = \oint_{\Gamma^t} (p\mathbf{n} - 2\mu\mathrm{D}(\mathbf{u})\mathbf{n})\cdot\mathbf{e}_x \mathrm{d}S, \quad F_L = \oint_{\Gamma^t} (p\mathbf{n} - 2\mu\mathrm{D}(\mathbf{u})\mathbf{n})\cdot\mathbf{e}_y \mathrm{d}S. \tag{2.25}$$

Concerning a more detailed analysis of the drag force (2.25), it is possible to decompose it into two distinct components that allow for the isolation of the effects of the pressure and

velocity fields within the solid, such that $F_D = F_{DP} + F_{DF}$. These components are the pressure drag force, denoted as F_{DP}, and the viscous drag force, denoted as F_{DF}, given by:

$$F_{DP} = \oint_{\Gamma^t} (p\mathbf{n}) \cdot \mathbf{e}_x \mathrm{d}S, \quad F_{DF} = \oint_{\Gamma^t} (-2\mu \mathrm{D}(\mathbf{u})\mathbf{n}) \cdot \mathbf{e}_x \mathrm{d}S.$$

3 | Numerical approximation

This chapter introduces the numerical methods employed to solve the fluid dynamic model described in chapter 2. In section 3.1, the Finite Volume Method is defined, which consists of a numerical technique used to solve partial differential equations by partitioning a continuous problem into a set of finite volumes, allowing us to derive the algebraic counterpart of the incompressible Navier-Stokes system. Subsequently in section 3.2, a projection method is introduced to solve the system of linear equations obtained from the discrete problem.

Afterward in section 3.3, the immersed boundary numerical algorithm is introduced. This algorithm entails adapting the incompressible Navier-Stokes equations to account for the presence of an immersed structure using a non-conforming mesh. Ultimately, a modified projection method is described for solving the fluid-structure interaction problem.

3.1. Finite Volume Method

The Finite Volume Method (FVM) is a numerical technique that partitions the domain into a series of non-overlapping polyhedral elements. It enforces conservation principles within each finite-volume cell, allowing for the approximation of the solution to the partial differential equations as a set of linear algebraic equations. This method is commonly employed in various computational fluid dynamics (CFD) solver software [14, 27], including OpenFOAM, which will be used in this project.

To construct the discretized mesh, we employ a polyhedral tessellation denoted as $\mathcal{T}_h$ of the computational domain $\Omega \subset \mathbb{R}^d$. Each cell element within this tessellation is represented as C_i, with i indicating its index. These cells have the volume denoted as $|C_i|$ and centroids located at $\mathbf{c}_i$. Additionally, we define $\mathrm{NB}(C_i)$ as the set of adjacent cells to C_i. It is important to note that we ensure that these finite volumes occupy the entire domain, meaning that Ω can be represented as the union of all C_i, and these volumes are non-overlapping.

Let $\mathcal{F}_h$ represent the set of faces F_i within the tessellation $\mathcal{T}_h$, also consider the set of

faces of cell C_i as $\mathcal{F}_{C_i}$. Each face F_i is associated with a surface area denoted as $|F_i|$, and barycenter located at $\mathbf{f}_i$. These faces can be classified into two subsets: $\mathcal{F}_I$, comprising internal faces, and $\mathcal{F}_B$, containing faces situated on the boundary $\partial\Omega$. To describe an internal face shared by two distinct cells C_i and C_j, we use the notation $F_{ij} \in \mathcal{F}_I$.

First, to exemplify the key features and fundamental principles of the Finite Volume Method, we shall apply it to a simplified problem of steady advection-diffusion-reaction. The purpose of this exercise is to elucidate the core concepts and underlying principles that characterize this numerical approach. After gaining insights from the simplified problem, we will proceed to develop the Finite Volume Method for the incompressible Navier-Stokes equations.

3.1.1. FVM for the advection-diffusion-reaction equation

In the context of the advection-diffusion-reaction equation (ADR), let $\phi(\mathbf{x}) : \Omega \to \mathbb{R}$ denote an arbitrary scalar field, $\mu(\mathbf{x}) : \Omega \to \mathbb{R}$ stand for the diffusion coefficient, $\mathbf{b}(\mathbf{x}) : \Omega \to \mathbb{R}^d$ represent the divergence-free convective field, $c(\mathbf{x}) : \Omega \to \mathbb{R}$ refer to the reaction coefficient, and $s(\mathbf{x}) : \Omega \to \mathbb{R}$ be the source term. Then, the ADR differential equation is described as:

$$\begin{cases} -\nabla \cdot (\mu \nabla \phi - \mathbf{b}\phi) + c\phi = s, \quad \text{in } \Omega \\ + \text{Boundary Conditions on } \partial\Omega. \end{cases}$$

When deriving the boundary conditions in the context of the Finite Volume Method, various approaches are available, which are described in [14]. For the sake of simplicity, we will focus on formulating the method for cells that are internal to the domain, omitting the consideration of boundary cells in this study.

Let us consider the control volume of cell $C_i \in \mathcal{T}_h$, whose surface can be decomposed into a set faces $\mathcal{F}_{C_i}$. By integrating over the domain of cell C_i and applying the Divergence Theorem 2.1, we obtain the following formulation:

$$-\oint_{\partial C_i} \mu \nabla\phi \cdot \mathbf{n} \mathrm{d}S + \oint_{\partial C_i} \phi \mathbf{b} \cdot \mathbf{n} \mathrm{d}S + \int_{C_i} c\phi \mathrm{d}V = \int_{C_i} s \mathrm{d}V.$$

The main concept of numerical approximation involves the use of the one-point Gaussian quadrature method [39], which estimates the integrand by assuming that the average of this quantity over the domain corresponds to its centroid value. We adopt the notation of subscript $\square_E$ to denote the evaluation at the centroid $\mathbf{e}$ of element E.

To begin, we focus on the numerical approximation of the diffusion term. For calculating

the surface integral over each face, we employ the one-point Gaussian quadrature method:

$$\oint_{\partial C_i} \mu \nabla \phi \cdot \mathbf{n} \mathrm{d}S = \sum_{F \in \mathcal{F}_{C_i}} \int_F \mu \nabla \phi \cdot \mathbf{n} \mathrm{d}S \approx \sum_{F \in \mathcal{F}_{C_i}} (\mu \nabla \phi \cdot \mathbf{n})_F |F|. \tag{3.1}$$

Similarly, we derive the term related to the transport term by decomposing the surface of the control volume into its constituent faces and employing the one-point Gaussian method of numerical integration:

$$\oint_{\partial C_i} \phi(\mathbf{b} \cdot \mathbf{n}) \mathrm{d}S = \sum_{F \in \mathcal{F}_{C_i}} \int_F \phi(\mathbf{b} \cdot \mathbf{n}) \mathrm{d}S \approx \sum_{F \in \mathcal{F}_{C_i}} (\phi \mathbf{b} \cdot \mathbf{n})_F |F|. \tag{3.2}$$

To obtain the reaction term expressed in terms of an approximated term evaluated at the centroid of the cell, we utilize the one-point Gaussian method of numerical integration within the control volume:

$$\int_{C_i} c\phi \mathrm{d}V \approx (c\phi)_{C_i} |C_i|. \tag{3.3}$$

Finally, we can derive the approximation of the source term analogously using the quadrature method:

$$\int_{C_i} s \mathrm{d}V \approx s_{C_i} |C_i|. \tag{3.4}$$

Hence, by approximating all the terms within the advection-diffusion-reaction equation and assessing the continuous functions at a single point, we effectively discretize the problem within cell C_i based on the quantities evaluated at both the centroid and faces of the cell:

$$-\sum_{F \in \mathcal{F}_{C_i}} (\mu \nabla \phi \cdot \mathbf{n})_F |F| + \sum_{F \in \mathcal{F}_{C_i}} (\phi \mathbf{b} \cdot \mathbf{n})_F |F| + (c\phi)_{C_i} |C_i| = s_{C_i} |C_i|. \tag{3.5}$$

To establish a connection between the values at the centroids of faces and those at the centroids of cells, we utilize linear interpolation. Consider two adjacent cells, labeled as C_i and C_j, which share the face F_{ij}, as illustrated in Figure 3.1.

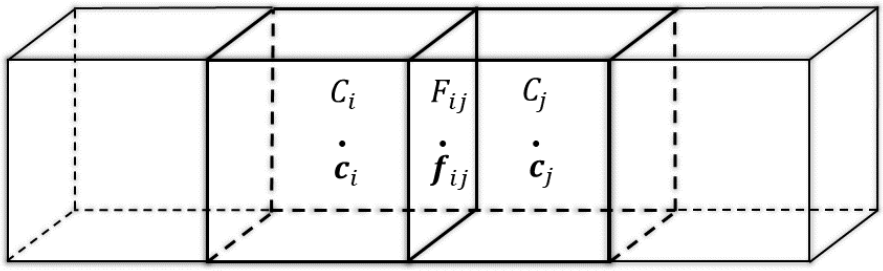


Figure 3.1: Illustration of the finite volume grid and the neighbors' computational cells C_i and C_j.

Then, by employing the average weight between the values of the scalar field ϕ at the centroids of adjacent cells, we can decompose the evaluation at the face centroid as

$$\phi_{F_{ij}} \approx \phi_{C_i}\lambda_{ij} + \phi_{C_j}(1-\lambda_{ij}), \quad \lambda_{ij} = \frac{|\mathbf{f}_{ij} - \mathbf{c}_i|}{|\mathbf{c}_j - \mathbf{c}_i|}.$$

In the context of a structured mesh, with the assumption of orthogonality between cell centers and assuming a linear profile between the centroids of the cells, we can approximate the gradient by employing a discretized approach using centered finite differences between two adjacent cell centroids:

$$(\nabla\phi \cdot \mathbf{n})_{F_{ij}} \approx \frac{\phi_{C_i} - \phi_{C_j}}{|\mathbf{c_i} - \mathbf{c_j}|}.$$

Subsequently, by utilizing interpolation for the values considered on the face and the method of centered finite differences for the gradient, it is possible to express the equation in terms of quantities evaluated at the centroid of cell C_i and its neighboring cells:

$$a_i\phi_{C_i} + \sum_{C_j \in \mathrm{NB}(C_i)} a_j\phi_{C_j} = b_i. \tag{3.6}$$

Ultimately, by generating (3.6) for each cell in the mesh, it is possible to formulate a system of linear equations that represents the entire simulation scenario. Let N_{dof} represent the number of degrees of freedom in the tessellation $\mathcal{T}_h$. Consider $\boldsymbol{\phi} \in \mathbb{R}^{N_{\mathrm{dof}}}$ as the vector of the scalar field ϕ evaluated at the centroids of the cells. Let $\mathrm{A} \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{dof}}}$ be the matrix of coefficients that accounts for the effects of advection, diffusion, reaction, and $\mathbf{b} \in \mathbb{R}^{N_{\mathrm{dof}}}$ be the vector representing the forcing terms. Thus, solving a high dimensional linear system we obtain a discretized approximation of the solution to the continuous partial

differential equation described by the ADR problem:

$$\mathrm{A}\boldsymbol{\phi} = \mathbf{b}.$$

3.1.2. FVM for the incompressible Navier-Stokes equations

To address the unsteady incompressible Navier-Stokes equations, we adopt a time discretization scheme. Consider a uniform partition of the time interval $[0, T]$, where T designates the terminal time of the designated scenario, and each discrete instant of time is denoted by $t^n = n\Delta t$, with Δt representing the time step. Furthermore, consider the superscript $\square^n$ to denote the approximation of a quantity evaluated at time t^n.

For the subsequent discourse in this chapter, consider the evaluation of the Navier-Stokes equations at the time instant t^{n+1}, taking into consideration the quantities from preceding time instances are known. To simplify the notation, we shall assume an implicit evaluation at t^{n+1} unless explicitly specified otherwise.

In addressing the time derivative, we consider the adoption of a second-order backward difference formula (BDF2), this choice leads to a second-order convergence in time:

$$\frac{\partial \mathbf{u}}{\partial t} \approx \frac{3\mathbf{u} - 4\mathbf{u}^n + \mathbf{u}^{n-1}}{2\Delta t}. \tag{3.7}$$

To spatially discretize and formulate an algebraic solution through a system of linear equations using the FVM, assuming that we are using a co-allocated mesh, we will employ techniques similar to those outlined in subsection 3.1.1.

To develop the spatial discretization of the finite volume formulation for the momentum equation, adopt the approximation of the time derivative term (3.7) and consider the momentum balance equation (2.23), integrating it over the control volume of the cell C_i:

$$\begin{aligned}\frac{3}{2\Delta t}\int_{C_i} \mathbf{u}\mathrm{d}V + \int_{C_i} (\mathbf{u}\cdot\nabla)\mathbf{u}\mathrm{d}V - \nu\int_{C_i}\Delta\mathbf{u}\mathrm{d}V \\ = \int_{C_i}\mathbf{f}\mathrm{d}V - \frac{1}{\rho}\int_{C_i}\nabla p\mathrm{d}V + \frac{1}{2\Delta t}\int_{C_i}(4\mathbf{u}^n - \mathbf{u}^{n-1})\mathrm{d}V.\end{aligned} \tag{3.8}$$

To develop a discrete formulation for the diffusion term, we will follow the methodology described in (3.1). This involves applying the Divergence Theorem 2.1 and using one-point

Gaussian quadrature to approximate the integration across each cell face:

$$\int_{C_i} \Delta \mathbf{u} \mathrm{d}V = \oint_{\partial C_i} \nabla \mathbf{u} \mathbf{n} \mathrm{d}S \approx \sum_{F \in \mathcal{F}_{C_i}} (\nabla \mathbf{u} \mathbf{n})_F |F|. \tag{3.9}$$

To formulate an equivalent expression for the transport term, as described in (3.2), we address the non-linearity by employing Picard's iteration method. This transforms the problem into an Oseen-type formulation, where we iteratively estimate the flux, denoted as $\widehat{\mathbf{u}}$. In this iterative process, we continuously update this estimate, aiming for convergence of the global solution, and ensuring that $(\widehat{\mathbf{u}} \cdot \nabla)\mathbf{u} \approx (\mathbf{u} \cdot \nabla)\mathbf{u}$.

Subsequently, we will proceed by employing the identities presented in (2.8) and (2.12), followed by the application of the Divergence Theorem 2.1. Finally, we shall employ the one-point quadrature over the cell's faces to complete the procedure:

$$\int_{C_i} (\widehat{\mathbf{u}} \cdot \nabla)\mathbf{u} \mathrm{d}V = \oint_{\partial C_i} (\widehat{\mathbf{u}} \cdot \mathbf{n})\mathbf{u} \mathrm{d}S \approx \sum_{F \in \mathcal{F}_{C_i}} ((\widehat{\mathbf{u}} \cdot \mathbf{n})\mathbf{u})_F |F|. \tag{3.10}$$

It is noteworthy that when considering the time derivative approximation, a reaction term emerges. This term will be addressed similarly to the approach detailed in (3.3):

$$\int_{C_i} \mathbf{u} \mathrm{d}V \approx \mathbf{u}_{C_i} |C_i|. \tag{3.11}$$

Evaluating the source term, which includes volume forces and velocity history, we utilize the integral approximation within the control volume, as demonstrated in (3.4):

$$\int_{C_i} \left(\mathbf{f} + \frac{1}{2\Delta t}(4\mathbf{u}^n - \mathbf{u}^{n-1}) \right) \mathrm{d}V \approx \left(\mathbf{f} + \frac{1}{2\Delta t}(4\mathbf{u}^n - \mathbf{u}^{n-1}) \right)_{C_i} |C_i|. \tag{3.12}$$

To estimate the integral of the pressure gradient, we use the Divergence Theorem 2.1. We then evaluate the approximation of this integration over the faces of the cell centroids:

$$\int_{C_i} \nabla p \mathrm{d}V = \oint_{\partial C_i} p \mathbf{n} \mathrm{d}S = \sum_{F \in \mathcal{F}_{C_i}} \int_F p \mathbf{n} \mathrm{d}S \approx \sum_{F \in \mathcal{F}_{C_i}} (p\mathbf{n})_F |F|. \tag{3.13}$$

Finally, for each computational cell $C_i \in \mathcal{T}_h$, analogous to the result in the ADR equation (3.5), we derive an approximation of the momentum equation (3.8). With this objective,

considering the results (3.9), (3.10), (3.11), (3.12), and (3.13), we obtain that:

$$\frac{3}{2\Delta t}\mathbf{u}_{C_i}|C_i| + \sum_{F\in\mathcal{F}_{C_i}}((\widehat{\mathbf{u}}\cdot\mathbf{n})\mathbf{u})_F|F| - \nu\sum_{F\in\mathcal{F}_{C_i}}(\nabla\mathbf{u}\mathbf{n})_F|F| + \frac{1}{\rho}\sum_{F\in\mathcal{F}_{C_i}}(p\mathbf{n})_F|F| = \left(\mathbf{f} + \frac{1}{2\Delta t}(4\mathbf{u}^n - \mathbf{u}^{n-1})\right)_{C_i}|C_i|. \quad (3.14)$$

To complete the system of differential equations of Navier-Stokes, we will discretize the continuity equation of the problem (2.23), which consists essentially of the divergence-free condition. Consider the control volume of cell C_i, and then apply the Divergence Theorem 2.1. Finally, approximate the integral using Gaussian quadrature:

$$\int_{C_i}\nabla\cdot\mathbf{u}\mathrm{d}V = \oint_{\partial C_i}\mathbf{u}\cdot\mathbf{n}\mathrm{d}S = \sum_{F\in\mathcal{F}_{C_i}}\int_F\mathbf{u}\cdot\mathbf{n}\mathrm{d}S \approx \sum_{F\in\mathcal{F}_{C_i}}(\mathbf{u}\cdot\mathbf{n})_F|F|.$$

As a result, considering the continuity equation of the system (2.23), we conclude that the total flux over the control volume of the cell must be zero:

$$\sum_{F\in\mathcal{F}_{C_i}}(\mathbf{u}\cdot\mathbf{n})_F|F| = 0. \quad (3.15)$$

Finally, we will adopt the algebraic counterpart of the problem. Let N_{dof} denote the number of degrees of freedom of the computational mesh $\mathcal{T}_h$, and consider $\mathbf{U}\in\mathbb{R}^{d\cdot N\text{dof}}$ and $\mathbf{P}\in\mathbb{R}^{N_{\text{dof}}}$ as the vectors that respectively hold the discretized components of velocity and pressure. Also, let $\mathbf{F}\in\mathbb{R}^{d\cdot N_{\text{dof}}}$ represent the source term in a vectorial formulation, that includes the volume force and velocity history contributions. Let $\mathrm{A}\in\mathbb{R}^{d\cdot N_{\text{dof}}\times d\cdot N_{\text{dof}}}$ represent the momentum coefficient matrix, and $\mathrm{B}\in\mathbb{R}^{N_{\text{dof}}\times d\cdot N_{\text{dof}}}$ be the divergence operator matrix. Thus, by considering the formulation (3.14) and (3.15) for all cells in the computational mesh, it is possible to write the system of equations in the matrix structure. In conclusion, we obtain a discretized approximation of the solution to the continuous partial differential equation described by the incompressible Navier-Stokes problem:

$$\begin{bmatrix}\mathrm{A} & \mathrm{B}^T\\ \mathrm{B} & 0\end{bmatrix}\begin{bmatrix}\mathbf{U}\\ \mathbf{P}\end{bmatrix} = \begin{bmatrix}\mathbf{F}\\ \mathbf{0}\end{bmatrix} \quad (3.16)$$

To solve the system described in equation (3.16), one can choose to use an iterative method due to its effectiveness in solving high-dimensional linear systems. However, a notable challenge arises from the absence of a pressure term in the continuity equation, since it

results in a block of zeros in the diagonal, generating a saddle-point problem structure. Consequently, this ill-conditioned system can lead to numerical instability and slowness at the solution [14]. To address this issue, projection methods are commonly employed, and their implementation in the current project will be elaborated in section 3.2.

3.2. PIMPLE projection method

Projection methods are algorithms that solve the incompressible Navier-Stokes equations in a segregated manner, which avoids the need for solving system (3.16). This approach involves obtaining an estimate of the velocity by solving the momentum equation, which may not necessarily satisfy continuity. Subsequently, the continuity equation is used to formulate an equation for the pressure field, incorporating the estimated velocity. Finally, the algorithm corrects the divergent velocity vector field to ensure it becomes non-divergent.

The PIMPLE algorithm [47] constitutes a synthesis of the SIMPLE algorithm (Semi-Implicit Method for Pressure Linked Equations) [33] with the PISO algorithm (Pressure-Implicit with Splitting of Operators) [45]. The PIMPLE method offers an advantage by combining the stability inherent in the SIMPLE method, with the computational efficiency of the PISO algorithm.

To briefly explain the PIMPLE method consider a partition of the incompressible Navier-Stokes problem into 3 steps. The primary phase entails solving the momentum equation for an intermediary velocity $\mathbf{u}^\star$, which is not necessarily constrained by the incompressibility condition. Subsequently, by introducing corrections to both pressure and velocity to uphold the continuity equation, it is possible to show that the pressure must satisfy a Poisson problem. Ultimately, the velocity components undergo a projection into a space that enforces the divergence-free condition, thereby yielding a new set of solutions for the velocity and pressure fields.

To express the differential form of the subproblems within the projection method at time t^{n+1}, we will use the superscript $\square^{(k)}$ to denote the iteration k. Thus, the PIMPLE method can be outlined as follows:

- **Momentum Prediction**:

$$\frac{3}{2\Delta t}\mathbf{u}^\star + (\mathbf{u}^{(k)} \cdot \nabla)\mathbf{u}^\star - \nu\Delta\mathbf{u}^\star = \mathbf{f} - \frac{1}{\rho}\nabla p^{(k)} + \frac{1}{2\Delta t}(4\mathbf{u}^n - \mathbf{u}^{n-1})$$

- **Pressure Equation**:

$$\nabla \cdot \left(\mathrm{D}^{-1}\nabla p^{(k+1)}\right) = \rho\nabla \cdot \mathbf{u}^\star$$

- **Velocity Correction**:

$$\rho \mathbf{u}^{(k+1)} = \rho \mathbf{u}^{\star} - \mathrm{D}^{-1} \nabla p^{(k+1)}$$

where D represents the diagonal of the momentum discretization matrix.

If the convergence tolerance is not achieved, the PIMPLE loop is restarted, updating the initial guess $(\mathbf{u}^{(k)}, p^{(k)})$ with the obtained result $(k \to k+1)$. On the other hand, if the results meet the convergence criterion, they are adopted as the solution, such that $(\mathbf{u}^{n+1}, p^{n+1}) = (\mathbf{u}^{(k+1)}, p^{(k+1)})$, and the algorithm proceed to the next time step.

The concept underlying the fusion of two approaches within the PIMPLE method is to integrate the SIMPLE outer loop with the PISO inner loop. The outer iterations encompass both the momentum and pressure correction equations, while the inner cycle begins iterating from the Poisson problem.

It is important to highlight that additional modifications to the method are possible, for example, the introduction of pressure relaxation between the Pressure Equation and Velocity Correction steps. This flexibility allows us to adapt the PIMPLE method to various problem settings, such as the Immersed Boundary Method, which will be explored in more detail in subsection 3.3.2.

3.3. Immersed Boundary numerical implementation

In the present work, we will adopt the formulation and numerical algorithm detailed in [30] to incorporate Fluid-Structure Interaction with an Immersed Boundary approach into the Navier-Stokes problem, as explained in section 2.7. Within the context of the Finite Volume discretization framework, as described in section 3.1, to adjust the cell value to the immersed boundary effect we will introduce the Immersed Boundary Interpolation Operator S_{IB} in subsection 3.3.1.

Subsequently, we formulate the projection algorithm, building upon the principles of the PIMPLE method outlined in section 3.2. This projection algorithm serves to couple the Fluid-Structure Interaction interface conditions into the fluid problem, leading to the development of the PIMPLE-IBM algorithm, which is detailed in subsection 3.3.2.

3.3.1. Immersed Boundary interpolator

To define the solid region Ω_s within the computational domain, we employ a closed surface triangulation Σ using a finite set of vertices $\mathcal{P}_{\text{IB}}$ as an approximation. In this project, the coordinates of the structural points were generated using MATLAB, resulting in the creation of a CSV file. This data was subsequently processed by Gmsh and ParaView, to generate an STL file that represents the geometry of the structure, encapsulating information about the triangulation and connectivity of the body surface. To illustrate the position, connectivity, and velocity of each vertex of a moving triangulated surface, refer to Figure 3.2.

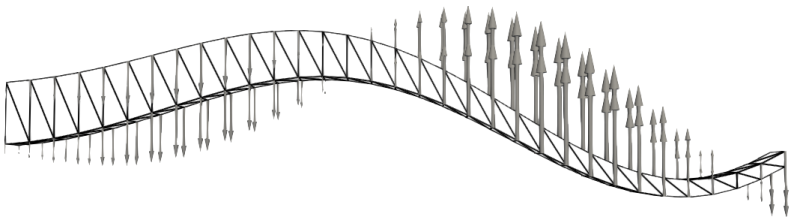


Figure 3.2: Representation of the triangulation Σ, and the velocity $\mathbf{u}_{\text{IB}}$ of the points.

Subsequently, it is necessary to categorize the cells comprising the computational mesh $\mathcal{T}_h$ into distinct subgroups. Referring to the mesh description within the Finite Volume Method, these cells are classified into three sub-categories: solid cells, fluid cells, and immersed boundary cells.

- **Solid Cells** ($\mathcal{C}_{\text{S}}$): Encompasses all cells with centroids situated within the solid surface region Σ. This region will symbolize the immersed object within the discrete space:

$$\mathcal{C}_{\text{S}} = \{C_i \in \mathcal{T}_h \mid \mathbf{c}_i \in \Sigma\}.$$

- **Immersed Boundary Cells** ($\mathcal{C}_{\text{IB}}$): Comprises cells whose centroids are not situated within the surface region and that share at least one face with the solid cells:

$$\mathcal{C}_{\text{IB}} = \{C_i \in \mathcal{T}_h \setminus \mathcal{C}_{\text{S}} \mid \ \text{NB}(C_i) \cap \mathcal{C}_{\text{S}} \neq \emptyset\}.$$

- **Fluid Cells** ($\mathcal{C}_{\text{F}}$): Covers all the remaining cells that are not included in the previous sets, this represents the fluid cells, where the solid and immersed boundary do not directly affect:

$$\mathcal{C}_{\text{F}} = \{C_i \in \mathcal{T}_h \setminus (\mathcal{C}_{\text{S}} \cup \mathcal{C}_{\text{IB}})\}.$$

To elucidate the criteria involved in mesh subdivision, Figure 3.3 presents a visual representation.

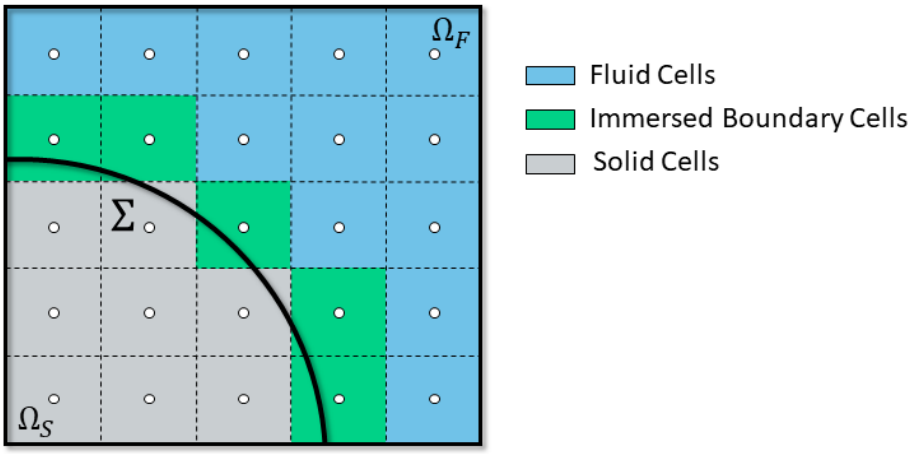


Figure 3.3: 2D representation of the mesh subdivision into solid cells $\mathcal{C}_{\mathrm{S}}$, fluid cells $\mathcal{C}_{\mathrm{F}}$, and immersed boundary cells $\mathcal{C}_{\mathrm{IB}}$.

Each immersed boundary cell, denoted as $C_i \in \mathcal{C}_{\mathrm{IB}}$, is associated with a set of extended stencil cells, represented as $\mathcal{S}_i$. This association becomes essential to select the relevant cells for the IB interpolation scheme.

$$\mathcal{S}_i = \{C_j \in \mathcal{T}_h \setminus \mathcal{C}_S \mid C_j \in \mathcal{S}_i^{\overline{c}} \cap \mathcal{S}_i^{\overline{d}} \cap \mathcal{S}_i^{\overline{\theta}}\}.$$

The set of cells of the fluid extended stencil is defined by the intersection of three geometric criteria that can be represented by $(\overline{c}, \overline{d}, \overline{\theta})$, which restrict the set based on the principles of connectivity, spatial distance, and field of view.

1. **Connectivity:** If a cell is included in the set related to a certain level of connectivity denoted as $\overline{c}$. The set $\mathcal{S}_i^{\overline{c}}$ is defined recursively using the *point-to-cell* stencil that includes all the cells that share at least one point:

$$\begin{aligned}
&\mathcal{S}_i^0 = \{C_i\}, \\
&\mathcal{S}_i^1 = \{C_j \in \mathcal{T}_h \mid C_j \cap \mathcal{S}_i^0 \neq \emptyset\} \cup \mathcal{S}_i^0, \\
&\quad \vdots \\
&\mathcal{S}_i^{\overline{c}} = \{C_j \in \mathcal{T}_h \mid C_j \cap \mathcal{S}_i^{\overline{c}-1} \neq \emptyset\} \cup \bigcup_{n=0}^{\overline{c}-1} \mathcal{S}_i^n.
\end{aligned}$$

2. **Spatial Distance:** If the distance between centroids of two cells is in a range of $\overline{d}$. This restriction can be interpreted as the cells whose centroids lie within a circle of radius $\overline{d}$ around $\mathbf{c}_i$:

$$\mathcal{S}_i^{\overline{d}} = \{C_j \in \mathcal{T}_h \mid |\mathbf{c}_i - \mathbf{c}_j| \leq \overline{d}\}.$$

3. **Field of View:** If the cell center is within the field of view defined by $\overline{\theta}$ with respect to the IB normal unit vector $\mathbf{n}_i$, which points in the direction of the projection of the IB cell centroid onto the surface Σ:

$$\mathcal{S}_i^{\overline{\theta}} = \left\{C_j \in \mathcal{T}_h \mid \mathbf{n}_i \cdot \frac{(\mathbf{c}_i - \mathbf{c}_j)}{|\mathbf{c}_i - \mathbf{c}_{\mathrm{IB},i}|} \leq \cos\overline{\theta}\right\}.$$

Illustrated in Figure 3.4 is an overview of the imposed restrictions used to determine the extended stencil cells related to the highlighted immersive boundary cell. Moreover, the picture also shows the IB point $\mathbf{c}_{\mathrm{IB}}$ defined by the projection of the IB cell centroid into the triangulated surface Σ.

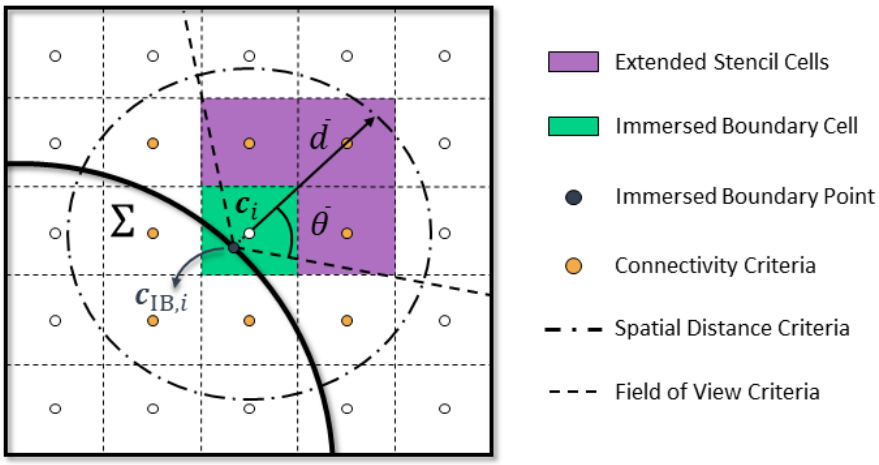


Figure 3.4: 2D representation of the extended stencil of an immersed boundary cell, considering connectivity criteria of level $\overline{c} = 1$, distance criteria of $\overline{d}$, and field of view criteria of $\overline{\theta}$.

Finally, to account for the effects imposed by the immersed structure, we will construct the IB interpolator denoted as $\mathrm{S}_{\mathrm{IB}}(\mathbf{U}, \mathbf{g})$ that corrects the solution field at the solid and immersed boundary cells. Define the IB value $\mathbf{g}_{\mathrm{IB}} = [g_{\mathrm{IB},i}]$ as the vector of values of the immersed boundary condition at the IB point $\mathbf{c}_{\mathrm{IB},i}$ correspondent to the cell $C_i \in \mathcal{C}_{\mathrm{IB}}$. Then, let $\mathbf{U}$ be the vector of the Finite Volume Method solution, and let $\mathbf{g}$ be the vector

of immersed boundary and solid region datum defined in every cell of the mesh:

$$\mathbf{g} = \begin{cases} g_{\mathrm{IB}.i}, & \text{if } C_i \in \mathcal{C}_{\mathrm{IB}} \\ g_i, & \text{if } C_i \in \mathcal{C}_{\mathrm{S}} \\ 0, & \text{if } C_i \in \mathcal{C}_{\mathrm{F}} \end{cases}$$

If $C_i \in \mathcal{C}_{\mathrm{IB}}$, we perform a linear combination of values from the cells within the stencil $\mathcal{S}_i$ and the corresponding IB point. If $C_i \in \mathcal{C}_{\mathrm{S}}$, we impose the prescribed solid solution, whereas if $C_i \in \mathcal{C}_{\mathrm{F}}$, we maintain the FVM solution. This process defines the interpolation operator S_{IB} that corrects the FVM solution:

$$\mathbf{U}^{\mathrm{corr}} = \mathrm{S}_{\mathrm{IB}}(\mathbf{g}, \mathbf{U}) = \begin{cases} s_{\mathrm{IB},i} g_{\mathrm{IB},i} + \sum\limits_{C_j \in \mathcal{S}_i} s_j u_j, & \text{if } C_i \in \mathcal{C}_{\mathrm{IB}} \\ g_i, & \text{if } C_i \in \mathcal{C}_{\mathrm{S}} \\ u_i, & \text{if } C_i \in \mathcal{C}_{\mathrm{F}} \end{cases} \tag{3.17}$$

To determine the coefficients s of the linear combination of effects in equation (3.17), we will employ the weighted least squares (WLS) approach. For this purpose, we will consider a polynomial vector $\mathbb{P}_{\mathbf{x}}$ of order p defined for a point $\mathbf{x} = [x\ y\ z]^T$:

$$\mathbb{P}_{\mathbf{x}} = \begin{bmatrix} 1 & x & y & z & \cdots & x^p & y^p & z^p \end{bmatrix},$$

and let $\mathbf{b}$ be the vector of coefficients of the polynomial:

$$\mathbf{b} = \begin{bmatrix} \beta_0 & \beta_1 & \beta_2 & \beta_3 & \cdots & \beta_{3p} \end{bmatrix}^T.$$

In the WLS approach, we aim to find the vector of coefficients $\mathbf{b}^\star$, which balances the solution of the extended stencil cells with that of the IB point:

$$\mathbf{b}^\star = \underset{\mathbf{b}}{\operatorname{argmin}} \left(w_{\mathrm{IB},i}(u_{\mathrm{IB},i} - \mathbb{P}_{\mathbf{c}_{\mathrm{IB},i}}\mathbf{b})^2 + \sum_{C_j \in \mathcal{S}_i} w_j (u_j - \mathbb{P}_{\mathbf{c}_j}\mathbf{b})^2 \right),$$

where the weight $w_i \in [0, 1]$ in the WLS method is determined based on the relative distance between the point under consideration and the center of the IB cell, as well as on the Dirichlet or Neumann condition. For a detailed formulation refer to [30].

Therefore, for each point $\mathbf{x}$ we can define the approximation function as $f_{\mathrm{IB}}(\mathbf{x}) = \mathbb{P}_{\mathbf{x}}\mathbf{b}^\star$. This function corresponds to the coefficients of equation (3.17), where we use $\mathbf{x}$ as either

the centroid of the stencil cell to obtain $s_j = f_{\mathrm{IB}}(\mathbf{c}_j)$ or as the IB point to estimate $s_{\mathrm{IB},i} = f_{\mathrm{IB}}(\mathbf{c}_{\mathrm{IB},i})$.

3.3.2. PIMPLE-IBM

The PIMPLE-IBM method is an integrated approach that incorporates IBM into the PIMPLE projection algorithm, as discussed in section 3.2. Our approach begins by establishing the monolithic problem formulation and subsequently employing the projection method algorithm to address the immersed boundary effect.

Within the formulation presented in (3.16), we introduce the operator $\mathbf{b} \in \mathbb{R}^{d \cdot N_{\mathrm{dof}}}$, which, utilizing the IB interpolator S_{IB} as detailed in subsection 3.3.1, ensures the non-conforming mesh correction in both solid and fluid cells:

$$\mathbf{b} = (1 - \chi_{\mathcal{C}_F})(-\mathrm{A}\mathbf{U} + \mathrm{B}^T\mathbf{P} + \mathbf{U} - \mathrm{S}_{\mathrm{IB}}(\mathbf{g}, \mathbf{U})),$$

where $\chi_{\mathcal{C}_F}$ is the indicator function of the discretized fluid domain.

Considering the system of equations as described in (3.16) with the addition of the Rhie-Chow stabilizer [31] denoted as $\mathrm{C} \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{dof}}}$, and introducing the non-conforming mesh corrector $\mathbf{b}$, we derive the algebraic counterpart of the immersed boundary problem (2.24):

$$\begin{bmatrix} \mathrm{A} & \mathrm{B}^T \\ \mathrm{B} & -\mathrm{C} \end{bmatrix} \begin{bmatrix} \mathbf{U} \\ \mathbf{P} \end{bmatrix} = \begin{bmatrix} \mathbf{F} + \mathbf{b} \\ \mathbf{0} \end{bmatrix}.$$

To formulate the PIMPLE-IBM algorithm, we start by decomposing the momentum discretization matrix $\mathrm{A} = \mathrm{D} - \mathrm{H}$ into two components: D as a diagonal matrix and $-\mathrm{H}$ as the off-diagonal counterpart. Furthermore, since the interpolator operator defined in (3.17) consists in a linear combination of the values of $\mathbf{U}$ and $\mathbf{g}$, it is possible to write the operator as the following decomposition:

$$\mathrm{S}_{\mathrm{IB}}(\mathbf{g}, \mathbf{U}) = \mathrm{S}_g\mathbf{g} + \mathrm{S}\mathbf{U}. \tag{3.18}$$

Hence, similar to the PIMPLE decomposition of the problem, we can divide the projection method into five steps, as represented on the scheme Figure 3.5. To describe the algebraic form of the problem at time t^{n+1}, adopt superscript $\square^{(k)}$ to denote iteration k, where the fraction represents an intermediate value for the variable. Thus, the PIMPLE-IBM method can be outlined as follows:

1. **Momentum Prediction**: Considering the momentum equation we make a prediction of the corrected velocity field using an estimative of the pressure and velocity fields $(\mathbf{U}^{(k)}, \mathbf{P}^{(k)})$, obtaining the first auxiliary velocity field $\mathbf{U}^{(k+1/3)}$.

$$\mathrm{A}\mathbf{U}^{(k+1/3)} + \mathrm{B}^T\mathbf{P}^{(k)} = \mathbf{F} + \mathbf{b}$$

2. **Velocity Corrector**: We apply a correction to the first predicted velocity field, yielding a refined estimate of the velocity field denoted as $\mathbf{U}^{(k+2/3)}$.

$$\mathbf{U}^{(k+2/3)} = \mathrm{D}^{-1}(\mathrm{H}\mathbf{U}^{(k+1/3)} + \mathbf{F})$$

   Additionally, it is essential to apply the IB interpolation in this procedure, ensuring that $\mathbf{U}^{(k+2/3)} = \mathrm{S}_{\mathrm{IB}}(\mathbf{g}, \mathbf{U}^{(k+2/3)})$ for the updated velocity field.

3. **Pressure Equation**: By combining the momentum and continuity equations, the pressure equation can be formulated. To achieve this, select the Rhie-Chow matrix as $\mathrm{C} = -\mathrm{S}\mathrm{D}^{-1}\mathrm{B}^T + \mathrm{R}(\mathrm{S}\mathrm{D}^{-1})$, where S is defined in (3.18). This strategic choice substitutes the pressure Schur Complement with the discretization of the pressure Laplacian operator via $\mathrm{R}(\mathrm{S}\mathrm{D}^{-1})$. Consequently, the resulting algebraic problem is solvable, yielding the pre-relaxation pressure field $\mathbf{P}^{(k+1/2)}$.

$$\mathrm{R}(\mathrm{S}\mathrm{D}^{-1})\mathbf{P}^{(k+1/2)} = \mathrm{B}\mathbf{U}^{(k+2/3)}$$

4. **Pressure relaxation**: Apply a pressure relaxation of a factor θ_P into the pressure field.

$$\mathbf{P}^{(k+1)} = \theta_P\mathbf{P}^{(k+1/2)} + (1 - \theta_P)\mathbf{P}^{(k)}$$

5. **Final Correction**: In the final step, the velocity field is projected into a divergence-free space, resulting in the new velocity field denoted as $\mathbf{U}^{(k+1)}$. This projection ensures consistency with the non-conforming condition.

$$\mathbf{U}^{(k+1)} = \mathbf{U}^{(k+2/3)} - \mathrm{S}\mathrm{D}^{-1}\mathrm{B}\mathbf{P}^{(k+1)}$$

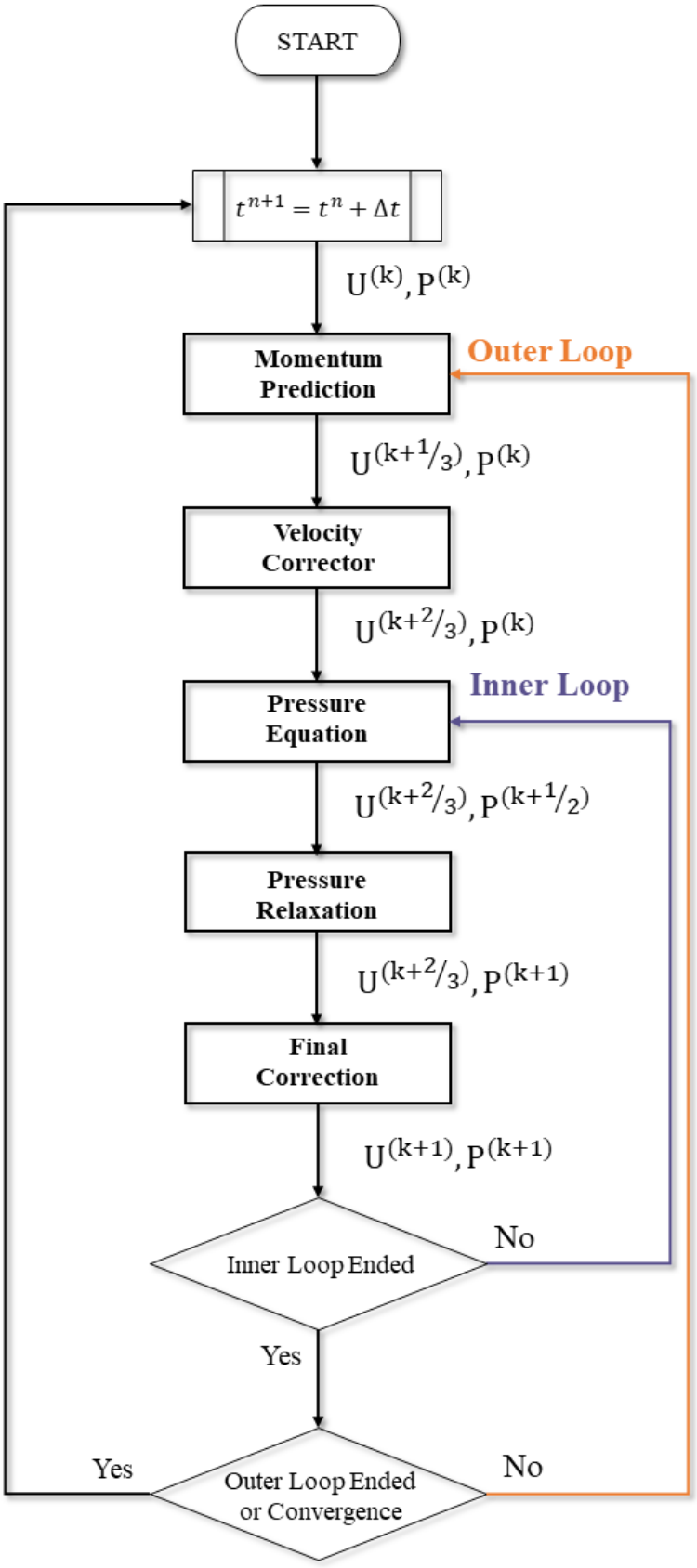


Figure 3.5: PIMPLE algorithm flowchart.

If the convergence tolerance is not attained, the PIMPLE-IBM loop is reinitiated, updating the initial estimative of the solution $(\mathbf{U}^{(k)}, \mathbf{P}^{(k)})$ based on the results obtained during the iteration $(k \rightarrow k+1)$. Conversely, if the convergence criterion is met, the obtained results are accepted as the solution, resulting in $(\mathbf{U}^{n+1}, \mathbf{P}^{n+1}) = (\mathbf{U}^{(k+1)}, \mathbf{P}^{(k+1)})$, and the simulation progresses to the next time step.

4 | Modelling of fish-like swimming

This chapter aims to analyze and develop a mathematical model to represent the geometry and motion of the fish. The modelling is based on previous simulations of fish-like swimming, which, in turn, were derived from empirical observations [46].

First, we considered the model of the fish described in [3], which has a geometry defined by a conforming map, resulting in an airfoil-like shape, as explained in section 4.1. Next, in section 4.2, we focus on analyzing the middle line of the fish, referred to as the backbone, which requires solving an integral equation to satisfy the fixed length of the backbone constraint. Then, in section 4.3, we analyze the solid body structure and apply segment rotations along the motion to ensure mass conservation. In the concluding section 4.4, the implementation of the OpenFOAM code related to the constructed algorithm is discussed.

4.1. Modelling of the swimmer steady-shape

To generate the reference state of the structure, we will consider the Karman-Trefftz airfoil, whose geometry can be described using a conformal map method. It is worth mentioning that the model is a common choice in the literature since it generates an airfoil which has several studies on potential flows and their theoretical solution [5, 43].

Consider the function $\zeta : [0, 2\pi] \to \mathbb{C}$, which is a parametrization of a circumference in the complex plane with center at the coordinate $(\eta_c, 0)$ and radius $(1 + |\eta_c|)$, defined as:

$$\zeta(\varphi) = (1 + |\eta_c|)e^{\varphi i} + \eta_c. \tag{4.1}$$

Let $\xi : \mathbb{C} \to \mathbb{C}$ be the parametrization of the airfoil shape with triangle-like cusp edge of trim angle α. The choice of the center of the circle ζ to lie on the real axis results in a shape with horizontal symmetry for the structure ξ. To produce the desired geometry,

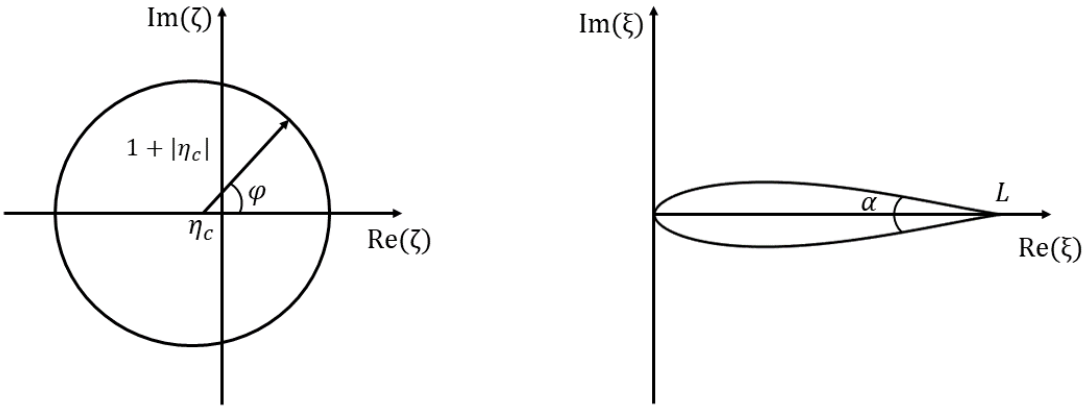
the Karman-Trefftz transformation will be applied:

$$\xi(\zeta) = n\frac{\left(1+\frac{1}{\zeta}\right)^n + \left(1+\frac{1}{\zeta}\right)^n}{\left(1+\frac{1}{\zeta}\right)^n - \left(1+\frac{1}{\zeta}\right)^n}, \tag{4.2}$$

where n is a function of the trim angle α:

$$n = 2 - \frac{\alpha}{\pi}.$$

Furthermore, a contraction or expansion of the resulting shape ξ was performed to fix it with a length of L. Finally, a translation of the structure was performed so its leading edge is at the origin of the complex plane, as illustrated in Figure 4.1.



(a) Circle parametrization ζ using equation (4.1). (b) Airfoil parametrization ξ using equation (4.2).

Figure 4.1: Example of Karman-Trefftz transformation.

Thus, for the modelling of the body when the backbone is completely horizontal, which is denoted as the reference configuration, 3 parameters define the structure:

- L: Backbone length.
- η_c: Circumference center.
- α: Trim angle of the airfoil.

4.2. Modelling of the fish backbone

To describe the motion of the backbone, it will be necessary to adopt two different coordinate systems. The first is the Cartesian coordinate system, which defines the points in the plane. The second consists of a curvilinear coordinate attached to the backbone, which demarcates the position over the middle line of the fish and varies between 0 and L.

4.2.1. Analytical description of the 2D motion

To simulate the backbone undulation of the swimming fish, let the function that describes the y Cartesian coordinate of the backbone be $y(x,t) : [0, x_L(t)] \times [0, +\infty) \to \mathbb{R}$, where $x_L(t) \in [0, L]$ is the x coordinate in the Cartesian plane such that the backbone has length L. A possible approach to simulate the movement of the backbone is the travelling wave undulation [3], which consists of a sinusoidal expression for the function $y(x,t)$:

$$y(x,t) = A_m(x) \sin\left(2\pi\left(\frac{x}{\lambda} - ft\right)\right), \tag{4.3}$$

where λ is the wavelength of the oscillations, f is the frequency of the oscillations, and A_m is the amplitude envelope defined by the quadratic polynomial:

$$A_m(x) = A_0 + A_1 x + A_2 x^2,$$

in which A_0, A_1, and A_2 are the coefficients of the polynomial.

The map between the curvilinear coordinate $s \in [0, L]$ and the Cartesian abscissa x is given by:

$$s(x,t) = \int_0^x \sqrt{1 + \left(\frac{\partial y(u,t)}{\partial u}\right)^2}\, \mathrm{d}u,$$

where for the derivative of the lateral motion function we have that:

$$\frac{\partial y(x,t)}{\partial x} = (A_1 + 2A_2 x) \sin\left(2\pi\left(\frac{x}{\lambda} - ft\right)\right) + \frac{2\pi}{\lambda}(A_0 + A_1 x + A_2 x^2) \cos\left(2\pi\left(\frac{x}{\lambda} - ft\right)\right). \tag{4.4}$$

To obtain the cartesian abscissa given a curvilinear coordinate, consider the inverse relation given by $x(s,t) = s^{-1}(x,t)$. It is worth noting that the quantity $x_L(t)$, which defines the domain of the travelling wave function (4.3), is obtained by $x_L(t) = x(L,t)$.

Another relevant information to be extracted from the equation of motion of the backbone

is the tangential angle of each point on the curve, which will be used afterward to model the motion of the swimmer surface. To obtain this quantity, compute the arc tangent of the derivative of the function in space:

$$\theta(x,t) = \arctan\left(\frac{\partial y(x,t)}{\partial x}\right). \tag{4.5}$$

4.2.2. Analytical description of the 3D motion

To generalize the motion of the swimmer backbone to a three-dimensional scenario, consider the coordinate $x \in [0, x_L(t)]$, such that the Cartesian coordinates of the backbone are described by $y(x,t) : [0, x_L(t)] \times [0, +\infty) \to \mathbb{R}$, and $z(x,t) : [0, x_L(t)] \times [0, +\infty) \to \mathbb{R}$. To replicate a similar movement, inspired by the 2D motion described in equation (4.3), consider the following formulation:

$$\begin{aligned} y(x,t) &= (A_{0y} + A_{1y}x + A_{2y}x^2)\sin\left(2\pi\left(\frac{x}{\lambda_y} - f_y t\right)\right), \\ z(x,t) &= (A_{0z} + A_{1z}x + A_{2z}x^2)\sin\left(2\pi\left(\frac{x}{\lambda_z} - f_z t\right)\right), \end{aligned} \tag{4.6}$$

where $A_{0y}, A_{0z}, A_{1y}, A_{1z}, A_{2y}, A_{2z}, \lambda_y, \lambda_z, f_y, f_z$ have an analogous meaning of its correspondent parameters described in (4.3).

The map between the curvilinear coordinate $s \in [0, L]$ and the Cartesian abscissa x is given by:

$$s(x,t) = \int_0^x \sqrt{1 + \left(\frac{\partial y(u,t)}{\partial u}\right)^2 + \left(\frac{\partial z(u,t)}{\partial u}\right)^2}\,\mathrm{d}u. \tag{4.7}$$

Finally, for the motion of the structure surface, it is necessary to obtain the direction of the tangential normalized vector of each point of the backbone. For the present project, the Euler Angles will be used, which have a wide application in aerodynamic studies [32], besides providing the motion of the structure one more degree of freedom, which corresponds to the rotation around the backbone's axis, referred as roll.

To obtain the angles that describe this direction, the set of Euler's angles $Y - z_1 - x_2$ will be applied [25]. Consider Figure 4.2 and the visual representation of the Euler's angles of yaw (ψ), pitch (θ) and roll (φ).

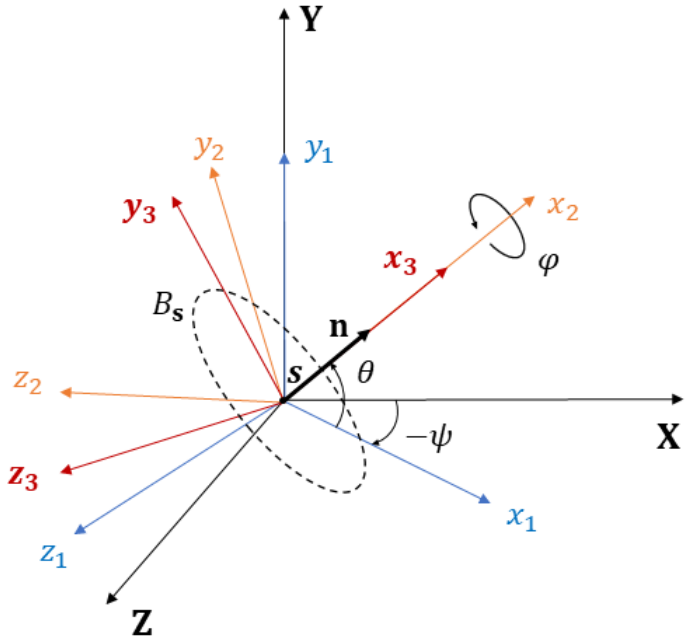


Figure 4.2: Euler's angles considering the rotation $Y - z_1 - x_2$.

The objective of using Euler's angles for orientation is to achieve a transformation of the coordinate system in such a way that the normal vector $\mathbf{n}$ of the segment $B_\mathbf{s}$ associated with the backbone point $\mathbf{s}$ remains parallel to the x-axis, as in the reference state. To accomplish this, the angles ψ and θ are determined, and their analytical expressions are as follows:

$$\psi(x,t) = -\arctan\left(\frac{\partial z(x,t)}{\partial x}\right), \quad \theta(x,t) = \arctan\left(\frac{{}^{\partial y(x,t)}/_{\partial x}}{\sqrt{\left({}^{\partial z(x,t)}/_{\partial x}\right)^2 + 1}}\right). \tag{4.8}$$

As a new degree of freedom was added for the rotation around the x_2 axis, the roll angle φ is free to choose for each point of the backbone. In this case, it can be selected considering the structure as a whole with the same roll angle:

$$\varphi(x,t) = 2\pi t f_\varphi \tag{4.9}$$

where f_φ is the roll frequency.

4.2.3. Backbone numerical algorithm

To describe the motion of the backbone points in space, consider a finite set of curvilinear coordinates denoted as $s = [s_i]$, which are provided in the discretized data of the structure. Without loss of generality consider that the points are uniformly distributed within the interval $[0, L]$. Considering the mapping between the curvilinear backbone coordinate s and x described in (4.7), each point s_i can be associated with a parameter x_i:

$$s_i = \int_0^{x_i} \sqrt{1 + \left(\frac{\partial y(u,t)}{\partial u}\right)^2 + \left(\frac{\partial z(u,t)}{\partial u}\right)^2}\,\mathrm{d}u. \tag{4.10}$$

Note that by adopting the origin of the Cartesian coordinate system as the initial backbone point, it follows that when $x_0 = 0$, then $s_0 = 0$.

After obtaining the mapping function, we can determine the coordinates of the corresponding backbone point at time t, prescribed by $\mathbf{s}_i = (x_i, y(x_i, t), z(x_i, t))$. Due to the complexity of the integral described in (4.10), obtaining an analytical solution for the parameter x_i is challenging. In this work, two algorithms have been developed for this purpose. The first algorithm utilizes the Newton Method to perform the inversion, while the second algorithm focuses on analyzing physical quantities related to distance and length.

Algebraic algorithm

The first algorithm involves directly solving the integral using Gaussian quadrature and estimating the parameter x_i through an iterative Newton Method. Define the auxiliary function $F(x, t) : [0, x_L] \times [0, +\infty) \to \mathbb{R}^+$, such that:

$$F(x,t) = \sqrt{1 + \left(\frac{\partial y(x,t)}{\partial x}\right)^2 + \left(\frac{\partial z(x,t)}{\partial x}\right)^2}. \tag{4.11}$$

Next, we define the function $G(x, t) : [0, x_L] \times [0, +\infty) \to \mathbb{R}$ such that its zeros correspond to the solutions of the curvilinear mapping equation (4.10), given by:

$$G(x,t) = \int_0^x F(u,t)\mathrm{d}u - s_i. \tag{4.12}$$

To outline Newton's method algorithm, we introduce the superscript $\square^{(k)}$ to denote iteration k of the scheme. The next iteration value depends on the evaluation of the functions

(4.11) and (4.12) using the previous iteration's data. Let $\epsilon \in \mathbb{R}^+$ be the convergence tolerance, the procedure is repeated until the convergence condition $|x_i^{(k+1)} - x_i^{(k)}| \leq \epsilon$ is satisfied. The iterative procedure is described by:

$$x_i^{(k+1)} = x_i^{(k)} - \frac{G(x_i^{(k)}, t)}{F(x_i^{(k)}, t)}.$$

It is worth mentioning that the initial guess for the parameter x_i is chosen by an extrapolation of the previous terms $x_i^{(0)} = 2x_{i-1} - x_{i-2}$.

The Gaussian quadrature will be employed to approximate the integral required for computing the function (4.12). We will use the notation w_q for the quadrature weight and ζ_q for the quadrature point with index q. To utilize the provided quadrature data that considers an integration in the interval $[-1, 1]$, we must adjust the integration interval, resulting in the approximation:

$$\int_0^x F(u,t)\mathrm{d}u = \frac{x}{2}\int_{-1}^{1} F\left(\frac{x}{2}u + \frac{x}{2}, t\right)\mathrm{d}u \approx \frac{x}{2}\sum_{q=1}^{n_{\text{quad}}} w_q F\left(\frac{x}{2}\zeta_q + \frac{x}{2}, t\right).$$

Geometric algorithm

The second algorithm involves creating a local discretization, calculating an approximation of the backbone's length at each local step, and then retrieving the corresponding x value when the desired curvilinear parameter is achieved.

Suppose that by a iterative process x_i , s_i and s_{i+1} are known and the objective is to estimate x_{i+1}. Notice that $x_{i+1} \in I_i = [x_i,\ x_i + (s_{i+1} - s_i)]$, this occurs once all points undergo a horizontal contraction when the object deforms from its initial flat position to a state at time t. Thus, it is expected that the distance of two consecutive points in x also undergo a relative contraction. Therefore, consider a new discretization of the interval I_i using uniformly distributed points denoted by the subscript $\square_{i,j}$, such that:

$$I_i = [x_i,\ x_i + (s_{i+1} - s_i)] = \bigcup_{j=0}^{M} [x_{i,j},\ x_{i,j+1}]. \tag{4.13}$$

The value x_{i+1} will occur when the string length is s_{i+1}. In this way, using the new discrete interval within I_i declared in (4.13), the length of the backbone at a given point $x_{i,j}$ will be calculated, until it approximately reaches the value s_{i+1} at the point $x_{i,J}$, in

the sense that $x_{i+1} \approx x_{i,J}$. To represent the concept, consider the expression:

$$s_{i+1} \approx s_i + \sum_{j=0}^{J} \sqrt{(x_{i,j+1} - x_{i,j})^2 + (y_{i,j+1} - y_{i,j})^2 + (z_{i,j+1} - z_{i,j})^2},$$

where J is the index such that $s_{i,J} \leq s_{i+1} \leq s_{i,J+1}$.

Figure 4.3 illustrates the backbone and local discretization in a simplified two-dimensional scenario used to numerically approximate the value of x_{i+1}.

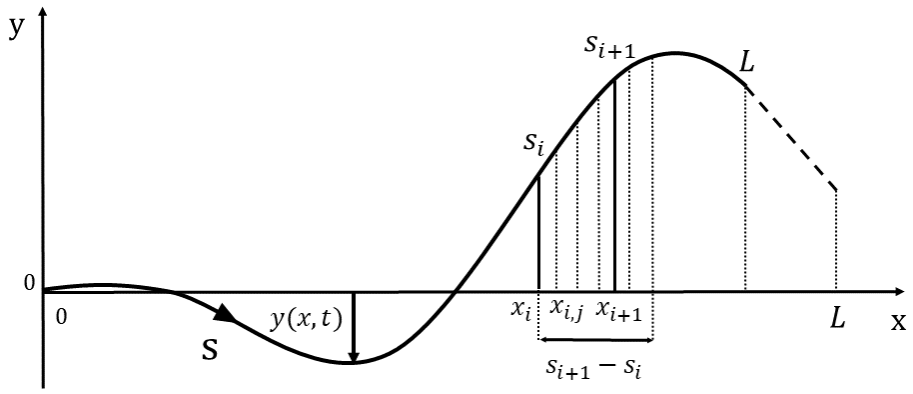


Figure 4.3: Example of the deformed backbone and the discretization procedure considering a 2D motion.

Comparison between the algorithms

The verification and comparative analysis of the algorithms will be conducted concerning a parabolic curve described by the quadratic polynomial $y(x) : \mathbb{R} \to \mathbb{R}$ and its associated length analytical function $s(x) : \mathbb{R}^+ \to \mathbb{R}^+$, described as:

$$y(x) = x^2, \quad s(x) = \frac{1}{2}\left((x\sqrt{(1+4x^2)} + \frac{1}{2}\log\left(2x + \sqrt{4x^2+1}\right)\right).$$

In this discussion, the objective is estimating the value of the x-axis coordinate, denoted as x_L, at which the curve achieves a unitary length, such that $s(x_L) = 1$.

In the algebraic method, a set of $n_{\text{quad}} = 8$ quadrature points is employed, and the efficiency of the algorithm is based on the number of iterations n_{iter} of the Newton method. To evaluate the geometric method it was employed a uniform partition with a step size of $\Delta x = 0.01$ within the interval $[0, 1]$. Between each subinterval, a local uniform discretization with n_{local} elements was adopted, as explained in equation (4.13). The results of the

algebraic and geometric algorithms are displayed in Table 4.1 and Table 4.2, respectively.

n_{iter}	x_L	$s(x_L)$
1	0.7858	1.0404
2	0.7641	1.0004
3	0.7639	1.0000
4	0.7639	1.0000

Table 4.1: Algebraic Method Analysis.

n_{local}	x_L	$s(x_L)$
10^1	0.7200	0.9214
10^2	0.7601	0.9930
10^3	0.7635	0.9993
10^4	0.7639	1.0000

Table 4.2: Geometric Method Analysis.

It is important to acknowledge that the geometric method introduces propagation of error within each subinterval, potentially requiring an increase in the number of local elements to attain a precise approximation. Therefore, in the present study, the algebraic algorithm will be employed to numerically map the Cartesian and curvilinear coordinates.

4.3. Modelling of the fish surface

To define the motion of the fish surface, let χ^* be the body in the reference configuration, $\chi(t)$ the body configuration at time t. Consider S^* the set of backbone points in the reference configuration and $S(t)$ the set of backbone points in the deformed configuration at time t.

4.3.1. Analytical description of the 2D motion

Regarding the movement of the fish, the conservation of mass should be respected, which considering the case of constant density would be equivalent to ensuring that the area of the structure remains constant over time. Therefore, for mass conservation [26], the following relation should hold:

$$\int_{\chi(t)} \mathrm{d}A = \int_{\chi^*} \mathrm{d}A^*, \quad \forall t \geq 0. \tag{4.14}$$

It is possible to associate each point $\mathbf{x}^* \in \chi^*$ a representative point in the backbone $\mathbf{s}^* = (s, 0) \in S^*$. For this purpose, define the function $\mathcal{S}^* : \chi^* \to S^*$ in the following way:

$$\mathcal{S}^*(\mathbf{x}^*) = \mathbf{s}^* \quad \text{such that} \quad \|\mathbf{x}^* - \mathbf{s}^*\| = \inf_{\mathbf{p} \in S^*} \|\mathbf{x}^* - \mathbf{p}\|. \tag{4.15}$$

For each point of the structure in the reference configuration $\mathbf{x}^\star$, we can associate a

curvilinear coordinate s. Using the mapping algorithms described in subsection 4.2.3, we can obtain the abscissa $x(s,t)$ and, as a consequence, the ordinate $y(x(s,t),t)$ of the backbone point. Those coordinates will be denoted as $\mathbf{s} = \mathbf{s}(s,t) = (x(s,t), y(x(s,t),t)) \in S(t)$, that defines the backbone point associated to $\mathbf{x}^\star$ at time t.

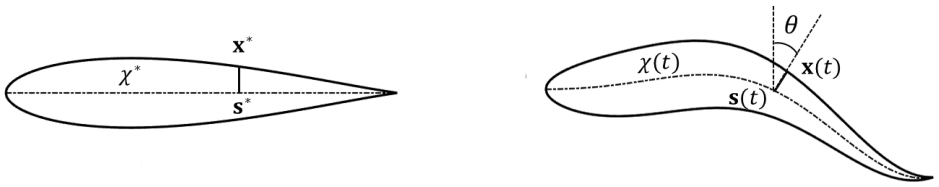
Furthermore, using the result of equation (4.5), it is possible to obtain the angle θ that generates the rotation matrix $\mathbf{R}$ associated to backbone point $\mathbf{s}$:

$$\mathbf{R}(\mathbf{s},t) = \begin{bmatrix} \cos\theta(x(s,t),t) & -\sin\theta(x(s,t),t) \\ \sin\theta(x(s,t),t) & \cos\theta(x(s,t),t) \end{bmatrix}.$$

Finally, the explicit map between the reference configuration point $\mathbf{x}^\star$ and the corresponding deformed configuration point $\mathbf{x} \in \chi(t)$ is described as follows:

$$\mathbf{x}(\mathbf{x}^*,t) = \mathbf{s} + \mathbf{R}(\mathbf{s},t)(\mathbf{x}^* - \mathbf{s}^*). \tag{4.16}$$

The visual representation of the strategy is performed in Figure 4.4.



(a) Reference configuration of the structure. (b) Deformed configuration of the structure at time t.

Figure 4.4: Illustration of the swimmer surface displacement.

4.3.2. Analytical description of the 3D motion

To describe the surface motion in a three-dimensional scenario, regarding the mass conservation of the body, and considering a constant density, the volume of the structure should remain constant over time. Therefore, similar to the area conservation defined in (4.14), we can define a volume conservation restriction represented as:

$$\int_{\chi(t)} \mathrm{d}V = \int_{\chi^*} \mathrm{d}V^*, \quad \forall t \geq 0.$$

Furthermore, in an analogous way of the two-dimensional case, using the mapping $\mathcal{S}^*$ defined in equation (4.15), it is possible to generate a relation between the reference structure point $\mathbf{x}^*$ to a reference backbone point $\mathbf{s}^* = (s, 0, 0)$. Then, using the mapping described in subsection 4.2.3 we can compute $\mathbf{s} = \mathbf{s}(s,t) = (x(s,t),\ y(x(s,t),t),\ z(x(s,t),t))$.

Subsequently, considering the expressions (4.8), and (4.9) it is possible to obtain the Euler's angles ψ, θ and φ that generates the rotation matrices $\mathbf{R}_\psi$, $\mathbf{R}_\theta$ and $\mathbf{R}_\varphi$ associated to a backbone point $\mathbf{s}$. The rotation matrices corresponding to each step of the change of basis can be obtained from the direction of cosine matrix transformation.

- Yaw (ψ): $O_{XYZ} \to O_{x_1y_1z_1}, \quad (Y = y_1)$

$$\mathbf{R}_\psi(\mathbf{s},t) = \begin{bmatrix} \cos\psi(x(s,t),t) & 0 & \sin\psi(x(s,t),t) \\ 0 & 1 & 0 \\ -\sin\psi(x(s,t),t) & 0 & \cos\psi(x(s,t),t) \end{bmatrix}$$

- Pitch (θ): $O_{x_1y_1z_1} \to O_{x_2y_2z_2}, \quad (z_1 = z_2)$

$$\mathbf{R}_\theta(\mathbf{s},t) = \begin{bmatrix} \cos\theta(x(s,t),t) & -\sin\theta(x(s,t),t) & 0 \\ \sin\theta(x(s,t),t) & \cos\theta(x(s,t),t) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Roll (φ): $O_{x_2y_2z_2} \to O_{x_3y_3z_3}, \quad (x_2 = x_3)$

$$\mathbf{R}_\varphi(\mathbf{s},t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi(x(s,t),t) & -\sin\varphi(x(s,t),t) \\ 0 & \sin\varphi(x(s,t),t) & \cos\varphi(x(s,t),t) \end{bmatrix}.$$

The transformation matrix, denoted as $\mathbf{T}(\mathbf{s},t)$, is obtained by multiplying the individual rotation matrices for each Euler angle, such that $\mathbf{T}(\mathbf{s},t) = \mathbf{R}_\psi(\mathbf{s},t)\mathbf{R}_\theta(\mathbf{s},t)\mathbf{R}_\varphi(\mathbf{s},t)$. The complete formulation of the 3D rotation matrix is given by:

$$\mathbf{T} = \begin{bmatrix} \cos\psi\cos\theta & -\cos\psi\sin\theta\cos\varphi + \sin\psi\sin\varphi & \cos\psi\sin\theta\sin\varphi + \sin\psi\cos\varphi \\ \sin\theta & \cos\theta\cos\varphi & -\cos\theta\sin\varphi \\ -\sin\psi\cos\theta & \sin\psi\sin\theta\cos\varphi + \cos\psi\sin\varphi & -\sin\psi\sin\theta\sin\varphi + \cos\psi\cos\varphi \end{bmatrix}. \quad (4.17)$$

Finally, the explicit map between the reference configuration point $\mathbf{x}^\star$ and the corresponding deformed configuration point $\mathbf{x}$ is described as follows:

$$\mathbf{x}(\mathbf{x}^*, t) = \mathbf{s} + \mathbf{T}(\mathbf{s}, t)(\mathbf{x}^* - \mathbf{s}^*). \tag{4.18}$$

4.3.3. Structure velocity field algorithm

Considering the formulations presented in equations (4.16) for two-dimensional scenarios and (4.18) for three-dimensional scenarios, a fundamental mapping is established. It connects a surface point in the reference configuration $\mathbf{x}^\star \in \chi^\star$, to its corresponding spatial location within the configuration at a specific time $\mathbf{x} \in \chi(t)$. For instance, when employing a time discretization scheme with time steps represented as $t^n = n\Delta t$, and utilizing the BDF2 approximation for the time derivative, the Lagrangian velocity $\mathbf{u}$ of the reference point $\mathbf{x}^\star$ is computed as:

$$\mathbf{u}(\mathbf{x}^\star, t^{n+1}) = \frac{3\mathbf{x}(\mathbf{x}^\star, t^{n+1}) - 4\mathbf{x}(\mathbf{x}^\star, t^{n+\frac{1}{2}}) + \mathbf{x}(\mathbf{x}^\star, t^n)}{\Delta t}. \tag{4.19}$$

To prescribe the velocity field for the solid cells, commonly referred to as *dead cells* within the OpenFOAM framework, it is necessary to reconstruct the Eulerian velocity field associated with the structure. Considering a three-dimensional scenario, this process involves associating a solid point $\hat{\mathbf{x}} = (\hat{x}, \hat{y}, \hat{z}) \in \chi(t)$ with a point $\mathbf{x}^\star = (x^\star, y^\star, z^\star) \in \chi^\star$, subsequently computing the Lagrangian velocity of $\mathbf{x}^\star$ at time t using (4.19).

By employing the mapping defined in (4.15), it creates a correspondence between each point $\hat{\mathbf{x}}$ of the deformed structure configuration to a point $\mathbf{s} = (x_s, y_s, z_s)$ on the deformed backbone, given by:

$$\hat{\mathbf{s}} = \underset{\mathbf{s} \in S(t)}{\operatorname{argmin}} \left(\sqrt{(x_s - \hat{x})^2 + (y_s - \hat{y})^2 + (z_s - \hat{z})^2} \right). \tag{4.20}$$

It can be demonstrated using the minimization of a function that the point $\hat{\mathbf{s}}$ satisfying (4.20) also complies with:

$$(x_s - \hat{x}) + (y(x_s, t) - \hat{y})\frac{\partial y(x_s, t)}{\partial x} + (z(x_s, t) - \hat{z})\frac{\partial z(x_s, t)}{\partial x} = 0, \tag{4.21}$$

where the functions $y(x, t)$ and $z(x, t)$ are defined at (4.6), the derivative $\frac{\partial y(x,t)}{\partial x}$ at (4.4), and the formulation of $\frac{\partial z(x,t)}{\partial x}$ is analogous to $\frac{\partial y(x,t)}{\partial x}$.

To solve the nonlinear problem represented by (4.21), the Newton method will be employed. In this context, adopt the superscript $\square^{(k)}$ to denote the k-th iteration of the

scheme, and initialize the process with $x_s^{(0)} = \hat{x}$ as the initial guess. Let $\epsilon \in \mathbb{R}^+$ be the convergence tolerance, and the algorithm is repeated until the convergence condition $|x_s^{(k+1)} - x_s^{(k)}| \leq \epsilon$ is satisfied. Then, the algorithm is prescribed as:

$$x_s^{(k+1)} = x_s^{(k)} - \frac{H(x_s^{(k)}, t)}{J(x_s^{(k)}, t)}.$$

The auxiliary functions $H(x,t) : [0, x_L(t)] \times [0, +\infty) \to \mathbb{R}$ and $J(x,t) : [0, x_L(t)] \times [0, +\infty) \to \mathbb{R}$ of the Newton method algorithm are defined as:

$$H(x,t) = (x - x^\star) + (y(x,t) - y^\star)\frac{\partial y(x,t)}{\partial x} + (z(x,t) - z^\star)\frac{\partial z(x,t)}{\partial x},$$
$$J(x,t) = 1 + \left(\frac{\partial y(x,t)}{\partial x}\right)^2 + (y(x,t) - y^\star)\frac{\partial^2 y(x,t)}{\partial x^2} + \left(\frac{\partial z(x,t)}{\partial x}\right)^2 + (z(x,t) - z^\star)\frac{\partial^2 z(x,t)}{\partial x^2},$$

where the second derivative of the lateral motion $\frac{\partial^2 y(x,t)}{\partial x^2}$ function is prescribed by

$$\frac{\partial^2 y(x,t)}{\partial x^2} = \left(2A_2 - \frac{4\pi^2}{\lambda^2}(A_0 + A_1 x + A_2 x^2)\right)\sin\left(2\pi\left(\frac{x}{\lambda} - ft\right)\right) + \frac{4\pi}{\lambda}(A_1 + 2A_2 x)\cos\left(2\pi\left(\frac{x}{\lambda} - ft\right)\right),$$

and $\frac{\partial^2 z(x,t)}{\partial x^2}$ posses an analogous formulation to $\frac{\partial^2 y(x,t)}{\partial x^2}$.

Thus, by solving (4.20), the point $\hat{\mathbf{s}} = (x_s, y_s, z_s)$ is obtained. This enables the reconstruction of the reference configuration point $\mathbf{x}^\star = (x^\star, y^\star, z^\star)$ through the use of the inverse formulation with respect to (4.18).

To retrieve the coordinates $x^\star$, $y^\star$ and $z^\star$, the mapping (4.10) and an inverse rotation are applied. Consider the angles (ψ, θ, φ) associated with the coordinate x_s, as described by the formulations (4.8) and (4.9). Then, define the inverse matrix $\mathbf{T}^{-1} = \mathbf{R}_\varphi^{-1}\mathbf{R}_\theta^{-1}\mathbf{R}_\psi^{-1}$ using the following formulation:

$$\mathbf{T}^{-1} = \begin{bmatrix} \cos\psi\cos\theta & \sin\theta & -\sin\psi\cos\theta \\ -\cos\psi\sin\theta\cos\varphi + \sin\psi\sin\varphi & \cos\theta\cos\varphi & \sin\psi\sin\theta\cos\varphi + \cos\psi\sin\varphi \\ \cos\psi\sin\theta\sin\varphi + \sin\psi\cos\varphi & -\cos\theta\sin\varphi & -\sin\psi\sin\theta\sin\varphi + \cos\psi\cos\varphi \end{bmatrix}. \tag{4.22}$$

Finally, the coordinates of the point $\mathbf{x}^\star$ of the reference configuration that on time t is

located at $\hat{\mathbf{x}}$ are given by the following expression:

$$\begin{bmatrix} x^\star \\ y^\star \\ z^\star \end{bmatrix} = \begin{bmatrix} x^{\star\star} \\ 0 \\ 0 \end{bmatrix} + \mathbf{T}^{-1} \begin{bmatrix} \hat{x} - x_s \\ \hat{y} - y_s \\ \hat{z} - z_s \end{bmatrix}, \tag{4.23}$$

where the coordinate $x^{\star\star}$ is given by:

$$x^{\star\star} = \int_0^{x_s} \sqrt{1 + \left(\frac{\partial y(u,t)}{\partial u}\right)^2 + \left(\frac{\partial z(u,t)}{\partial u}\right)^2}\, \mathrm{d}u.$$

Hence, a bijection has been established between the spatial coordinates of a solid point and its corresponding point in the reference configuration, denoted as $\hat{\mathbf{x}} \leftrightarrow \mathbf{x}^\star$. This enables the reconstruction of the Eulerian velocity field using the mapping defined by (4.16) for each point $\hat{\mathbf{x}} \in \chi(t)$, calculating the corresponding $\mathbf{x}^\star$ and applying the time derivative approximation (4.19).

4.4. Fish-like swimming implementation

The numerical simulations have been performed using the *open-source* software OpenFOAM, complemented by the incorporation of the Immersed Boundary Method library [30]. The code functionality, initially restricted to rigid body motions, was expanded in this thesis to encompass deformable bodies, incorporating the kinematic description of fish-like swimming.

Within the directory *src/immersedBoundary/immersedBoundaryDynamicFvMesh*, a folder labeled *flexibleIb* was created containing the source code *flexibleIb.C* and its header *flexibleIb.H*. This adds the option of *ibFlexibleBodyMotionFvMesh* into the parameter dynamicFvMesh of *constant/dynamicMeshDict*.

The C++ code implements Algorithm 4.1, which reads the STL file stored in the directory *constant/triSurface* at the simulation case, generating the deformed configuration of the solid at time t. Additionally, it calculates the structure surface velocity, as elaborated in section 4.2 and section 4.3. Considering that two-dimensional motion is a particular case of three-dimensional motion, the description will concentrate on the latter.

Algorithm 4.1 Flexible Motion Algorithm

1: **Input Parameters:** Reference coordinates $(x^\star, y^\star, z^\star)$ and time t
2: **for** All points of the reference STL **do**
3: Project the reference structure point onto the backbone to obtain the curvilinear coordinate s
4: Calculate $x(s,t)$ solving (4.10) using the algorithm described in subsection 4.2.3
5: Compute $y(x,t)$ and $z(x,t)$ using the expressions (4.6)
6: Compute the Euler angles $\psi(x,t)$, $\theta(x,t)$ and $\varphi(x,t)$ using (4.8) and (4.9)
7: Generate the transformation matrix $\mathbf{T}(x,t)$ described in (4.17)
8: Update the reference point position using the map (4.18)
9: Recalculate the position at the time $t - {}^{\Delta t}/_{2}$
10: Recalculate the position at the time $t - \Delta t$
11: **end for**
12: **Output Result:** Using the BDF2 approximation in (4.19), return the Lagrangian velocity field of the structure.

For prescription of the Eulerian velocity field within the solid cells, commonly referred to as *dead cells*, Algorithm 4.2 is applied. A dedicated folder named *nonUniformDeadValueIb* is created within the directory *src/immersedBoundary/immersedBoundary/fvPatchFields/derived*. This folder contains the source code *nonUniformDeadValueIbFvPatchVectorField.C* along with its corresponding header *nonUniformDeadValueIbFvPatchVectorField.H*. The code implements the *nonUniformDeadValueIb* type within the IBCYLINDER of *0.orig/U*.

Algorithm 4.2 Non Uniform Dead Cell Algorithm

1: **Input Parameters:** Dead cell coordinates $(\hat{x}, \hat{y}, \hat{z})$ and time t
2: **for** All dead cells **do**
3: Project the dead cell onto the backbone to obtain the x-coordinate x_s solving the minimization problem (4.21)
4: Compute $y(x_s,t)$ and $z(x_s,t)$ using the expressions (4.6)
5: Compute the Euler angles $\psi(x_s,t)$, $\theta(x_s,t)$ and $\varphi(x_s,t)$ using (4.8) and (4.9)
6: Generate the inverse transformation matrix $\mathbf{T}^{-1}(x_s,t)$ described in (4.22)
7: Calculate the corresponding point $(x^\star, y^\star, z^\star)$ in the reference configuration using the map (4.23)
8: Use the algorithm 4.1 to obtain the velocity of $(x^\star, y^\star, z^\star)$ at time t.
9: **end for**
10: **Output Result:** Eulerian velocity field of the structure.

To input the kinematic parameters, a dictionary named *flexibleMotionProperties* is established within the *constant* folder of the simulation setup. Within this dictionary, the user has the flexibility to make several selections:

- **Dimension:** The available options include two-dimensional motion, which can be selected using the *motion2D* option, or a simplified motion without considering inextensibility and mass conservation constraints, which can be chosen by selecting *simplified2DMotion*. In the absence of any selection, the complete 3D algorithm will be applied.
- **Kinematic parameters:** Specify various parameters of the wave function (4.6), including $A_{0y}, A_{0z}, A_{1y}, A_{1z}, A_{2y}, A_{2z}, \lambda_y, \lambda_z, f_y, f_z$, and also the roll frequency f_φ. Notice that in the 2D motion, it is assumed that only the y-direction is considered.
- **Transient time:** Specify the transient period T from the initial resting state. The characterization of the transient state is detailed in section 6.1.
- **Initial phase:** Set the phase ϕ_y and ϕ_z of the wave functions.
- **Newton Method Parameters:** Customize parameters for the Newton method, providing control over tolerance and the maximum number of iterations, to enable the effective solution of the algorithms outlined in both subsection 4.2.3 and subsection 4.3.3.

In the context of post-processing and force calculations, the implementation includes two folders, namely *ibForces* and *ibForceCoeffs*, situated within the directory *src/functionObjects*. These folders collectively generate the data file *ibForceCoeffs.dat*, located within the *postProcessing/forces/0* folder of the simulation case. This file computes the drag and lift coefficients, adhering to the instructions provided in the *forces* dictionary found in the *system* folder of the simulation setup.

5 | Numerical verification

In this chapter, we conducted a series of experiments to validate the immersed boundary algorithm and the force calculations using benchmark cases. Initially, in section 5.1, we presented results from simulations of a two-dimensional external flow over a circular cylinder, varying the Reynolds number to analyze different scenarios. These tests encompass a verification of the solution in a steady-state scenario, followed by a simulation of an unsteady case to confirm the consistency of the time-marching algorithm. Additionally, computations are performed to verify an immersed body with rigid body motion.

In section 5.2, we extended our investigations to scenarios involving immersed bodies with flexible motion. The first scenario involved a flat plate with an undulating motion, while the second case featured a fish-body airfoil exhibiting an undulating motion inspired by the kinematics of the animal. It is important to note that this representation simplifies the motion and does not consider constraints related to inextensibility and mass conservation.

In all the cases described in this section, we performed a dynamic local refinement of the background mesh when constructing the computational domain around the immersed body using a three-layer refinement. For instance, Figure 5.1 illustrates the refined mesh around a circular cylinder.

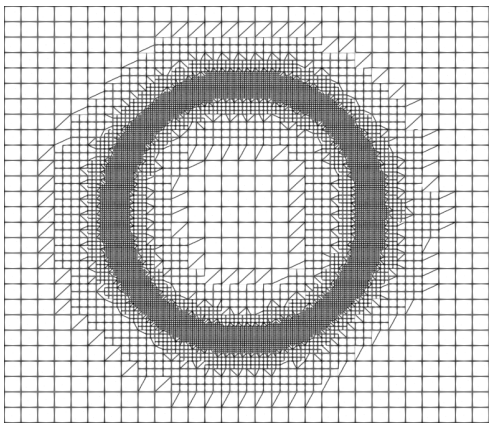


Figure 5.1: Illustration of a locally refined mesh around a circular cylinder.

5.1. Flow over a circular cylinder

To verify the simulation result using the immersed boundary algorithms and the forces computation, the benchmarking scenario of a two-dimensional flow around a circular cylinder is considered. Let the domain of the simulation be constructed such that $\Omega = [x_1, x_2] \times [y_1, y_2]$, and consider an immersed cylinder of diameter D centered at the coordinates (x_c, y_c) with a velocity of U_{IB}.

Regarding the boundary conditions applied in the rectangular domain, the left boundary features an inflow condition with a horizontally uniform velocity of U_∞, along with a zero gradient condition for pressure. The top and bottom boundaries are assigned slip conditions for velocity and zero gradient conditions for pressure. Finally, the right boundary features an outflow condition with a zero gradient for velocity and a uniform null value for pressure. The described scenario is illustrated in the Figure 5.2.

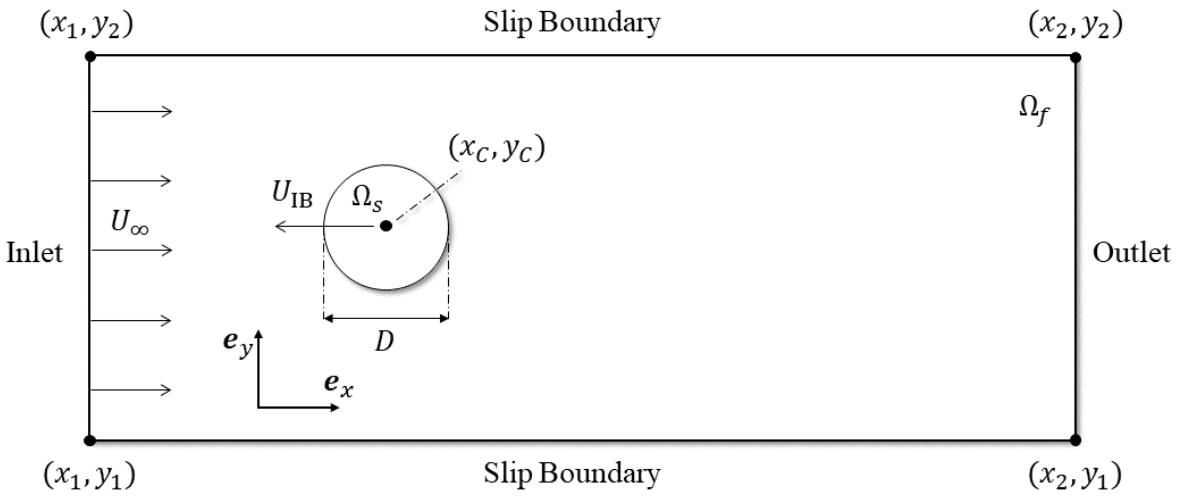


Figure 5.2: Geometric domain of the simulation cylinder.

To categorize and analyze the simulations, we will examine two dimensionless parameters, the Reynolds number (Re) and the Strouhal number (St), defined as:

$$Re = \frac{DU_r}{\nu}, \quad St = \frac{fD}{U_r},$$

where $U_r = |U_\infty - U_{\text{IB}}|$ is the relative velocity of the solid concerning the fluid, and f is the frequency of the lift force oscillation.

Furthermore, to evaluate the forces, it is common practice to adopt the dimensionless parameters of drag and lift coefficients, represented respectively as C_D and C_L. To compute

these parameters, we first calculate the forces as described in equation (2.25). Once these quantities are obtained, we calculate the dimensionless coefficients as:

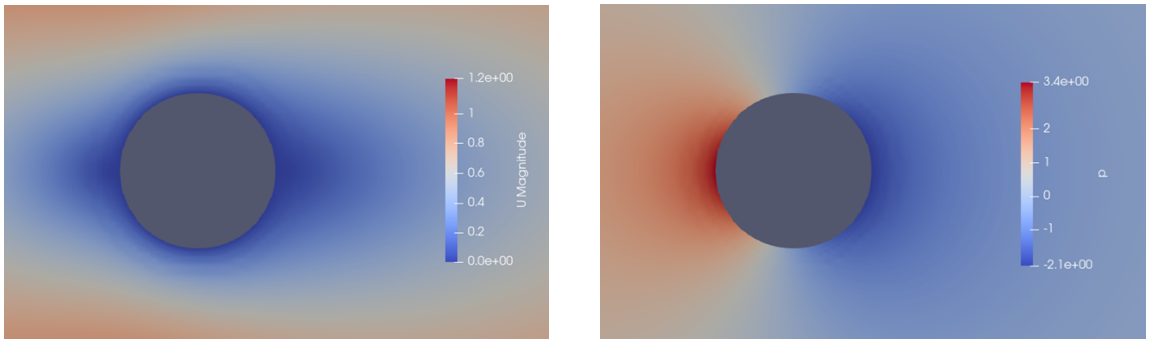
$$C_D = \frac{F_D}{\frac{1}{2}\rho U_r^2 D}, \quad C_L = \frac{F_L}{\frac{1}{2}\rho U_r^2 D}.$$

For the simulation setup of flow over a circular cylinder, we consider the same computational domain of $\Omega = [-6, 20] \times [-6, 6]$, utilizing anisotropic distributed nodes, with the number of points in the horizontal and vertical directions being $N_x = 600$ and $N_y = 300$, respectively. These parameters were selected through a series of experiments to guarantee grid independence and to ensure that the boundary conditions do not significantly influence the results in all scenarios.

5.1.1. Steady flow, $Re = 1$

The initial validation involves considering the creeping flow scenario of $Re = 1$, which characterizes a flow field predominantly influenced by viscosity. Assuming this state, the detachment of the boundary layer from a surface into a wake does not occur, which allows us to assume a steady-state hypothesis. Consequently, we employ the SIMPLE-IBM algorithm to solve the present scenario and verify the accuracy of the algorithm.

Concerning the solid surface, it consists of an immersed fixed cylinder $U_{\text{IB}} = 0$ m/s with a diameter of $D = 1$ m, the cylinder is centered at a slight displacement from the origin, at coordinates $x_c = 0.0$ m and $y_c = 0.1$ m. Finally, to achieve a Reynolds number of $Re = 1$, a uniform inflow with a fluid velocity of $U_\infty = 1$ m/s and a kinematic viscosity of $\nu = 1$ m^2/s were considered.



(a) Velocity field for Re = 1 (b) Pressure field for Re = 1

Figure 5.3: Results of the simulation of a flow over a cylinder considering Re = 1.

Analyzing the results presented in Figure 5.3, which occurs after 10^5 iterations of the

SIMPLE-IBM method, we observe the characteristic behavior of creeping flow. This includes a pressure gradient around the cylinder such that there is a drop at the wake of the cylinder and higher pressure at the front. Additionally, there is a region of low velocity near the solid surface, expected from the no-slip condition imposed at the solid-fluid interface.

The result obtained for the drag coefficient is approximately $C_d = 12.7$, which is consistent with the documented literature for this scenario as exhibited in Figure 5.4.

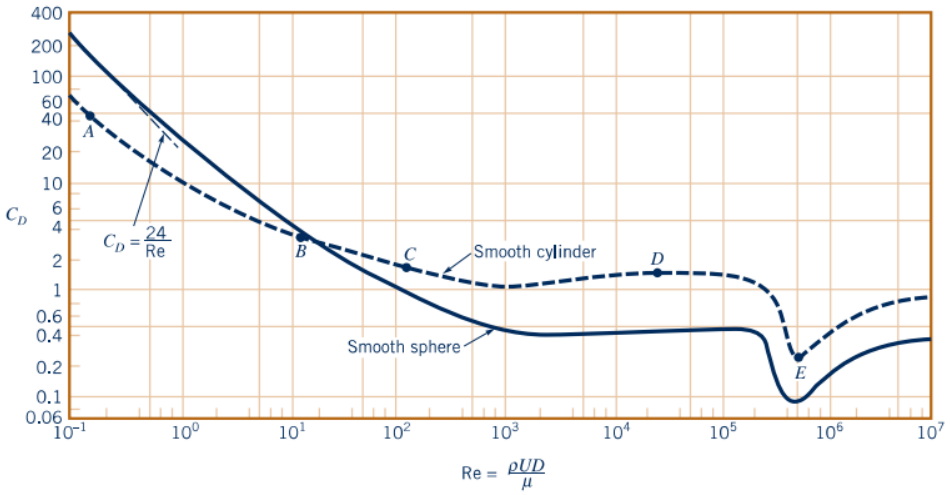


Figure 5.4: Drag coefficient as a function of Reynolds number for a smooth circular cylinder and sphere [28].

5.1.2. Unsteady flow, $Re = 200$

The second case focuses on a Von Kármán wake scenario obtained when $Re = 200$, where periodic vortex shedding is observed. It is important to highlight that due to the periodic nature of the dynamics, the simulation is unsteady, and the verification of the time-marching algorithm PIMPLE-IBM was performed.

Similar to the previous case, we considered a cylinder of diameter $D = 1$ m, centered at coordinates $x_c = 0.0$ m and $y_c = 0.1$ m. The slight elevation of the cylinder above the origin is made to introduce asymmetry and accelerate the development of the vortex-shedding pattern. To obtain the desired Reynolds number of $Re = 200$, a uniform inflow with a fluid velocity of $U_\infty = 1$ m/s and a kinematic viscosity of $\nu = 5 \times 10^{-3}$ m^2/s were considered.

In this evolutionary scenario, we employed a time step of $\Delta t = 5 \times 10^{-3}$ s, this parameter was determined through extensive testing, demonstrating its independence from the time

interval. The simulation was conducted until the capture and illustration of Von Kármán street patterns, as depicted in Figure 5.5. This figure illustrates the vorticity profile of the wake, defined as $\boldsymbol{\omega} = \nabla \times \mathbf{u}$.

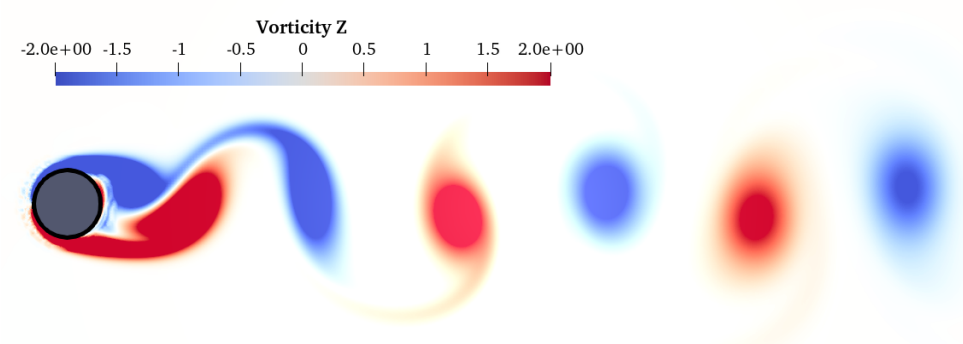


Figure 5.5: Von Kármán street vorticity patterns for an external flow over a cylinder at Re = 200.

After simulating a sufficient duration to capture the periodic behavior, the forces were calculated during the post-processing. Figure 5.6 illustrates the evolution of drag and lift coefficients, beginning from the transient state and progressing towards the periodic regime characterized by Von Kármán wake.

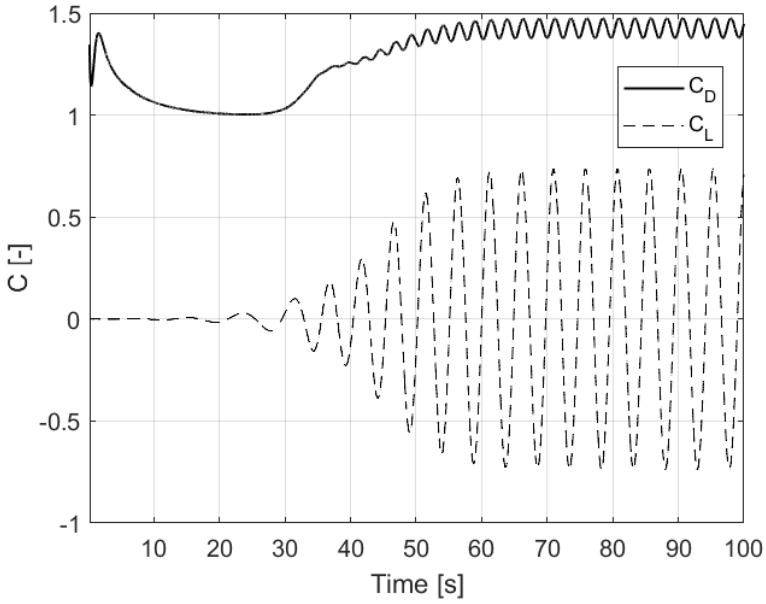


Figure 5.6: Drag and Lift coefficients at Re = 200.

To calculate the Strouhal Number, we utilized a Fast Fourier Transform algorithm to extract the main frequency of oscillation from the lift coefficient result, which corresponds

to the period of wake oscillation. The obtained results for the average drag coefficient and the Strouhal Number are consistent with findings in the literature, as presented in Table 5.1.

	$\mathbf{C_D}$	$\mathbf{S}_t$
Braza et al. [7]	1.4000	0.2000
Henderson [20]	1.3412	0.1971
He et al. [19]	1.3560	0.1978
Bergmann et al. [4]	1.3900	0.1999
Bergmann et al. [3]	1.3500	0.1980
Present Work	**1.4268**	**0.2050**

Table 5.1: C_D and S_t comparison for $Re = 200$.

5.1.3. Moving cylinder, $Re = 550$

In the present experiment, we aim to verify the scenarios with include the motion of the immersed body. Specifically, we will compare a simulation of external flow over a fixed cylinder at $Re = 550$ with a scenario that accounts for an inertial fluid and a moving cylinder to replicate the same conditions.

This experiment involves a comparison between two scenarios. In the first scenario, similar to previous cases, the cylinder remains fixed in space with a uniform inflow at a fluid velocity of $U_\infty = 1$ m/s. In the second scenario, we impose an impulsive rigid body movement of $U_{\text{IB}} = -1$ m/s on the cylinder, which is immersed in an inertial fluid, such that in the location of the inlet, it is adopted a second outlet boundary condition. To achieve a Reynolds number of $Re = 550$ in both scenarios, a kinematic viscosity of $\nu \approx 1.82 \times 10^{-3}$ m^2/s was employed. Notice that in this case the mesh is dynamically refined to encompass the current position of the cylinder in the motion case.

In this experiment, we employed a time step of $\Delta t = 5 \times 10^{-3}$ s, and the scenario was simulated for a shorter duration of 3 s. This shorter simulation time was sufficient to verify the adherence of the moving body to the expected curve of the drag coefficient obtained in the literature [38]. The result of both the fixed cylinder and moving cylinder is illustrated in Figure 5.7.

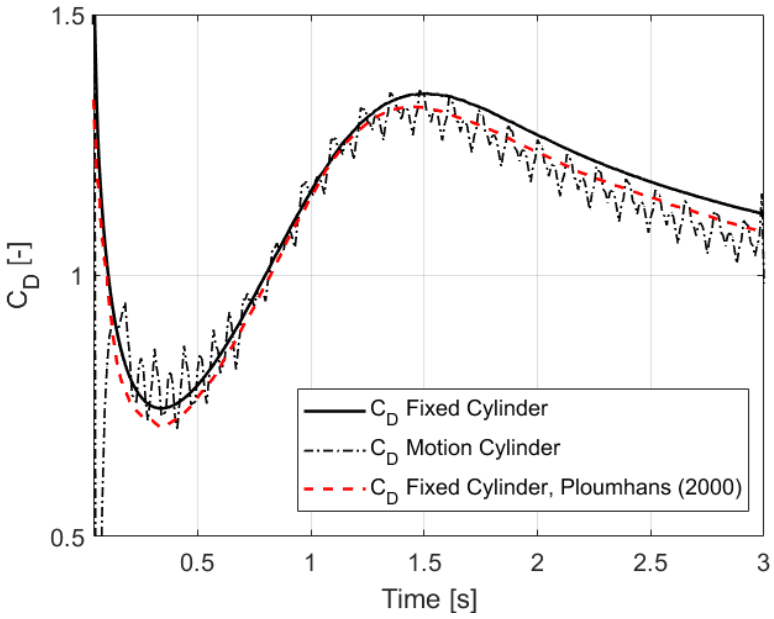


Figure 5.7: Comparison between the external flow over a fixed and motion circular cylinder at $Re = 550$, compared with values from Ploumhans & Winckelmans (2000) [38].

To generate Figure 5.7, a weighted moving average was applied to reduce noise in the case of the moving cylinder, eliminating high-amplitude outliers:

$$\widehat{x}_i = \frac{x_{i-2} + 2x_{i-1} + 3x_i + 2x_{i-1} + x_{i-2}}{9}.$$

It is important to note that when simulating the moving body, there are small amplitude oscillations around the expected curve of the fixed cylinder case. Similar differences were also observed in related experiments using the Immersed Boundary Method [3]. It was verified in the reference study that this phenomenon occurs because the discrete-time derivative is taken over a support that varies in time.

5.2. Travelling wavy foils

In section 5.1, we have verified that the immersed boundary algorithm accurately solves scenarios involving external flow over a cylinder. This extends to the verification of simulations involving the unsteady incompressible Navier-Stokes equations with an immersed body, whether it remains stationary or undergoes rigid body motion.

In this section, scenarios involving immersed bodies with non-uniform deformations, representing general cases of flexible bodies, are investigated. Benchmark cases are considered: In subsection 5.2.1 the wavy flexible plate is analyzed, and in subsection 5.2.2 a simplified

modelling of fish-like swimming that does not adhere to the constraints of fixed backbone length and mass conservation is performed.

Consider a simulation domain, denoted as $\Omega = [x_1, x_2] \times [y_1, y_2]$, which encompasses an immersed flexible body with arbitrary geometry, possessing a total horizontal length of L, where one of its ends is positioned at the origin of the Cartesian coordinate system. Regarding the boundary conditions, the same conditions applied to the cylinder case are considered. These include a uniform inflow velocity of U_∞, slip conditions on the bottom and top walls, and an outlet condition on the right wall. For illustrative purposes, refer to Figure 5.8.

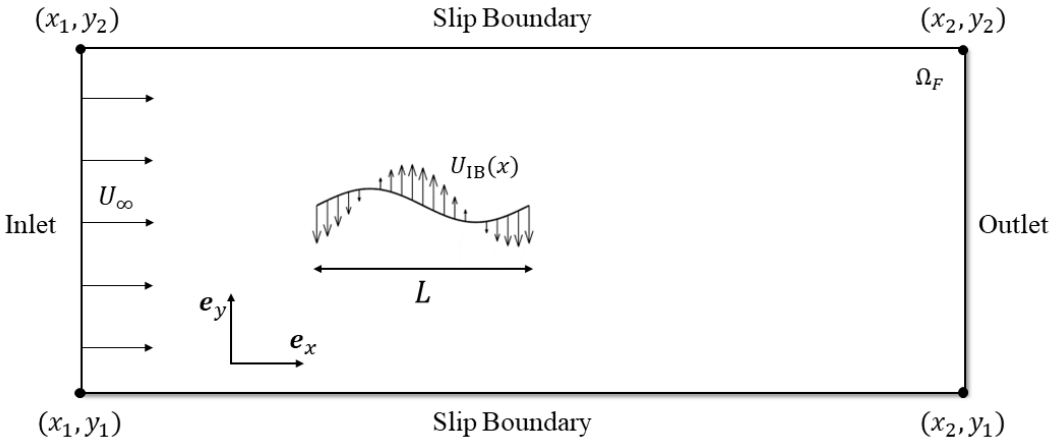


Figure 5.8: Geometric domain of the simulation of a flexible body.

The simulation setup for these scenarios remains consistent with the previous cases of flow over a circular cylinder. For both cases, it was considered a domain with dimensions of $\Omega = [-6, 20] \times [-6, 6]$, and an anisotropic mesh comprising $N_x = 600$ points in the horizontal direction and $N_y = 300$ points in the vertical direction, with dynamic mesh refinement applied at the immersed boundary location. For each case, the computational setup was verified to ensure grid independence and minimize the influence of boundary conditions.

The dimensionless parameters evaluated in the simulation of the flexible body include the Reynolds number (Re) and the Strouhal number (St):

$$Re = \frac{LU_\infty}{\nu}, \quad St = \frac{fL}{U_\infty},$$

where f is the imposed frequency of the travelling wave.

The results of this study will be based on the total drag coefficient (C_D), pressure drag co-

efficient (C_{DP}), viscous drag coefficient (C_{DF}), and lift coefficient (C_L). These parameters for the flexible body case are defined as:

$$C_D = \frac{F_D}{\frac{1}{2}\rho U_\infty^2 L}, \quad C_{DP} = \frac{F_{DP}}{\frac{1}{2}\rho U_\infty^2 L}, \quad C_{DF} = \frac{F_{DF}}{\frac{1}{2}\rho U_\infty^2 L}, \quad C_L = \frac{F_L}{\frac{1}{2}\rho U_\infty^2 L}.$$

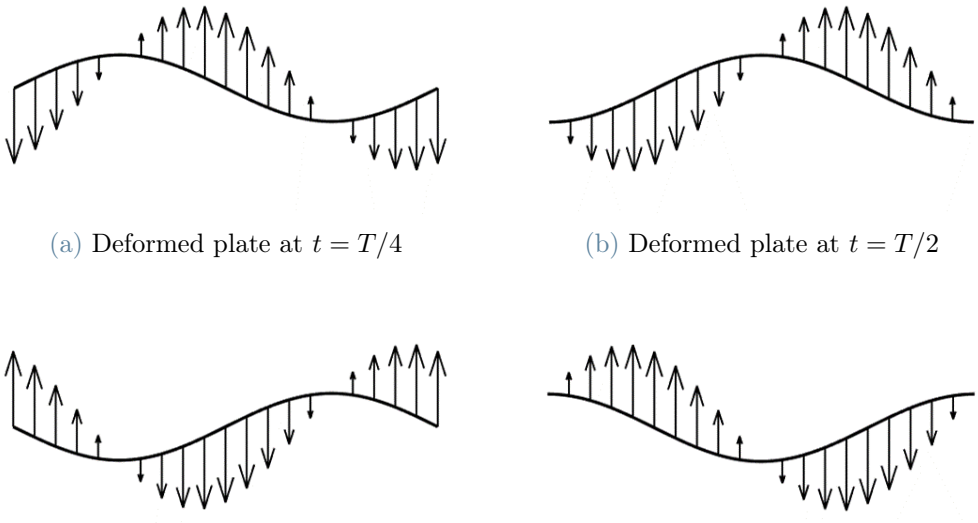
5.2.1. Travelling wave plate

The objective of modelling the wavy plate case is to validate the motion of a flexible immersed body and the calculation of forces for a simplified geometry. It is important to note that the study of this structure is not the primary focus of the work, rather, it serves as an initial verification and validation of a generic body motion.

Introducing the scenario for a travelling wave plate in a two-dimensional case, we consider a plate geometry with a length of $L = 1$ m and a small thickness of $e = 0.01$ m, subjected to an undulated motion of uniform amplitude $A_m = 0.1$ m. Then, we define the function $y(x,t) : [0, L] \times [0, +\infty) \to \mathbb{R}$ to represent the backward undulating motion of the plate:

$$y(x,t) = A_m \sin\left(2\pi\left(\frac{x}{L} - ft\right)\right), \quad 0 \leq x \leq L.$$

To visualize one period of the prescribed motion, refer to Figure 5.9, which presents the undulation motion at various time instances within a period T of the oscillation cycle.



(a) Deformed plate at $t = T/4$

(b) Deformed plate at $t = T/2$

(c) Deformed plate at $t = 3T/4$

(d) Deformed plate at $t = T$

Figure 5.9: Wavy plate motion in one period T.

To configure the simulation scenario, an intermediate Reynolds number of $Re = 500$ was selected. This setup involved employing a uniform inflow with a fluid velocity of $U_\infty = 1$ m/s and a kinematic viscosity of $\nu = 2 \times 10^{-3}$ m^2/s. Additionally, a time step of $\Delta t = 2.5 \times 10^{-3}$ s was employed. This choice was made to ensure an accurate representation of the oscillatory motion imposed on the flexible plate, considering a range of oscillation frequencies from $f = 0.5$ Hz to $f = 2.5$ Hz.

The drag force exerted on the wavy plate is crucial for comprehending the locomotion dynamics since it reflects the power required to propel. For each frequency of oscillation, we performed a simulation to determine the mean drag coefficient, which is then compared to values obtained in the literature, as presented in Figure 5.10.

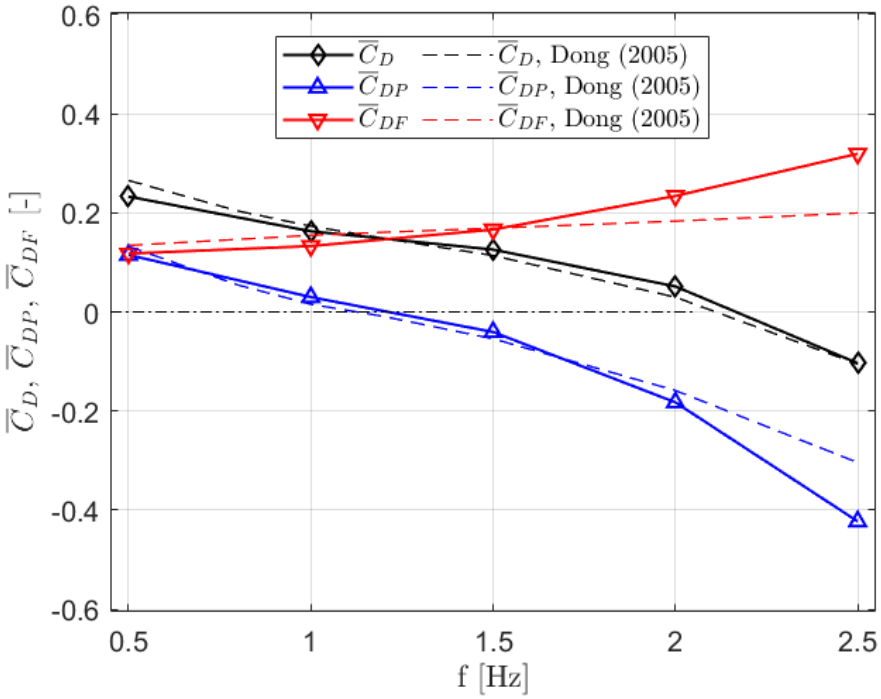


Figure 5.10: Time average drag coefficient for the travelling wavy plate as a function of the frequency of oscillation, compared with values from Dong & Lu (2005) [12].

It is possible to notice the trend of the mean drag coefficient, which decreases until it reaches a negative value approximately at frequencies $f > 2.0$ Hz, which implies the drag force acting as a thrust force. This phenomenon will be further investigated in the fish-like swimming simulation, where we delve into the propulsion force that enhances locomotion efficiency.

5.2.2. Travelling wavy fish-body like

In this section, we investigate a simplified two-dimensional kinematic model of fish-like swimming, which has been previously studied in [13]. This model offers insights into important aspects of swimming profile dynamics, including the average drag coefficient and motion efficiency. However, it has limitations as it does not impose fixed values for the length and volume of the fish, nor does it account for self-propulsion due to force feedback.

To replicate the fish-like shape in this set of simulations, we utilized the NACA0012 airfoil with a zero flap angle to define the geometry of the structure. The original formulation of the airfoil is scaled by a factor of $r = 1.0089$, ensuring a total length of $L = 1$, following the approach outlined in [1]. Thus, consider the function $Y_{\text{NACA}}(x) : [0, 1] \to \mathbb{R}$, which characterizes the body surface defined as:

$$Y_{\text{NACA}}(x) = \pm Y_0(Y_1\sqrt{x} + Y_2 x + Y_3 x^2 + Y_4 x^3 + Y_5 x^4), \quad 0 \leq x \leq 1$$

where the coefficients are $Y_0 = 0.5947$, $Y_1 = 0.2982$, $Y_2 = -0.1271$, $Y_3 = -0.3579$, $Y_4 = 0.2920$, $Y_5 = -0.1052$.

Based on empirical kinematic data acquired from observations of steady swimming saithe, particularly a carangiform swimmer [46], we define the function $y(x, t) : [0, L] \times [0, +\infty) \to \mathbb{R}$ to represent the backward undulating motion of the middle line. It is important to note that the backbone is defined over the interval $x \in [0, L]$, resulting in a curved middle-line in a deformed configuration with a length that exceeds the value of L:

$$y(x, t) = A_m(x) \sin\left(2\pi\left(\frac{x}{L} - ft\right)\right), \quad 0 \leq x \leq L$$

where f is the oscillation frequency, and $A_m(x)$ is the amplitude envelope described using a quadratic curve:

$$A_m(x) = A_0 + A_1 x + A_2 x^2.$$

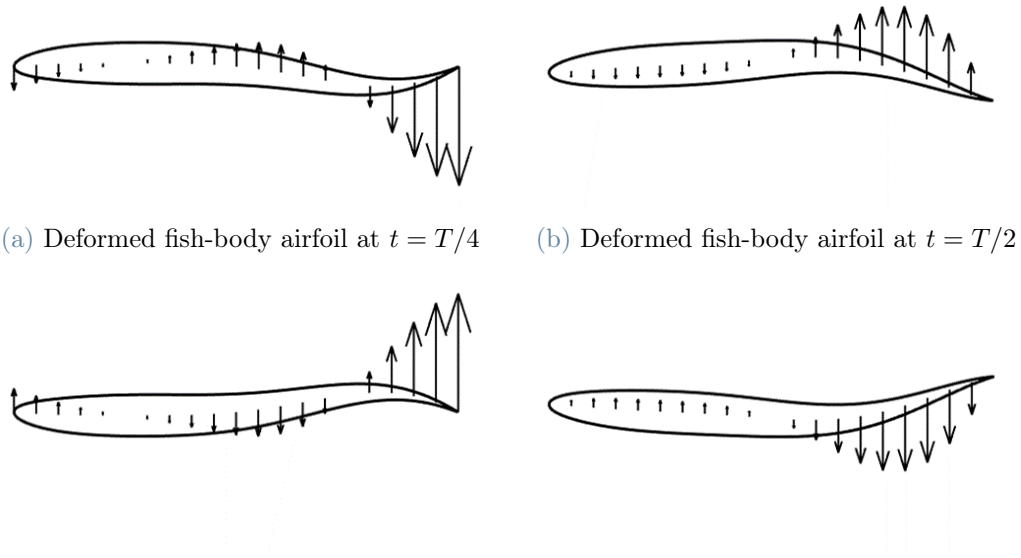
The parameters of the amplitude envelop that represents a swimming motion are chosen such that $A_m(0) = 0.02$, $A_m(0.2) = 0.01$, and $A_m(1) = 0.10$. These conditions results in the set of coefficients: $A_0 = 0.02$, $A_1 = -0.0825$, and $A_2 = 0.1625$.

Within the framework of the notation introduced in section 4.3, we establish the map between the reference and deformed state being a translation of the segment according to the backbone motion:

$$\mathbf{x}(\mathbf{x}^\star, t) = \mathbf{s} + (\mathbf{x}^\star - \mathbf{s}^\star).$$

Notice that in this case $\mathbf{s} = (x^\star, y(x^\star, t))$, neglecting the non-linear mapping.

To visualize the motion of the travelling wave fish-body like within one complete period, refer to Figure 5.11, where the arrows indicate the velocity of the backbone points.



(a) Deformed fish-body airfoil at $t = T/4$ (b) Deformed fish-body airfoil at $t = T/2$

(c) Deformed fish-body airfoil at $t = 3T/4$ (d) Deformed fish-body airfoil at $t = T$

Figure 5.11: Wavy fish-body airfoil motion in one period T.

Finally, to simulate the motion of an aquatic animal, a Reynolds number of $Re = 5000$ was selected. This Reynolds number denotes an intermediate regime, indicating that the flow does not exhibit purely laminar characteristics. It is important to note that the IBM solver utilized in this study does not support a turbulent model, which may consequently lead to a decrease in result accuracy.

This setup involved employing a uniform inflow with a fluid velocity of $U_\infty = 1$ m/s and a kinematic viscosity of $\nu = 2 \times 10^{-4}$ m^2/s. To perform the time discretization a step of $\Delta t = 2.5 \times 10^{-3}$ s was employed, such that it is sufficient to capture the oscillatory motion and the vortices of the wake.

Analyzing the effect of increasing the frequency of oscillation, it is possible to notice that the average mean coefficients decrease. Until it gets negative for frequencies $f > 1.5$. As illustrated in the Figure 5.12, which compares the result with the simulations presented in [13].

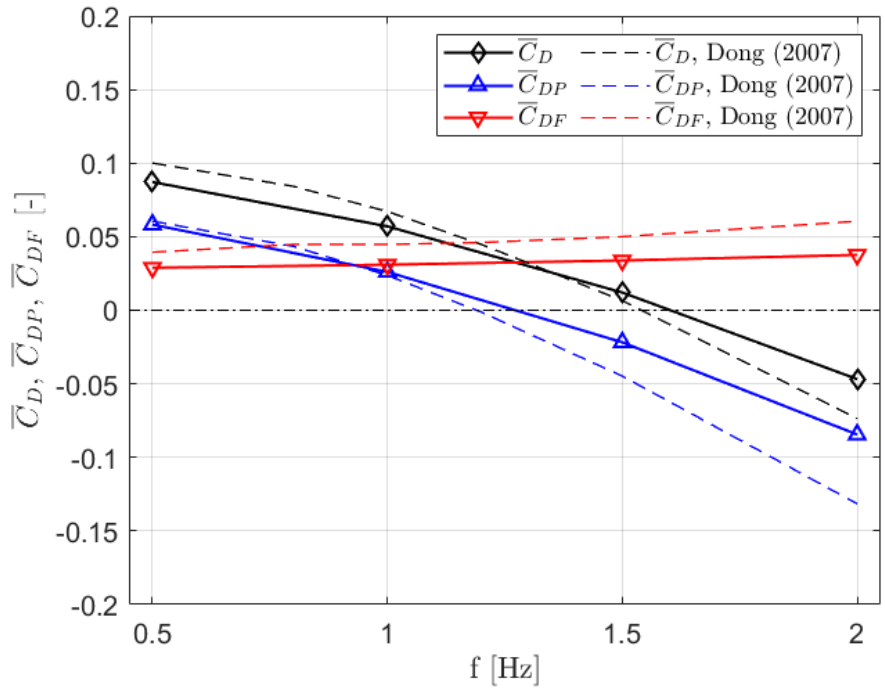


Figure 5.12: Time average drag coefficient for an undulation NACA0012 as a function of frequency of oscillation, compared with values from Dong & Lu (2007) [13].

6 | Fish-like swimming results

In this chapter, several analyses were carried out on the fish-like swimming model and simulations. First, in section 6.1, a discussion revolves around kinematics while considering the constraints of inextensibility and mass conservation. Then, in section 6.2, a series of experiments were conducted within a two-dimensional simulation. These experiments included variations in the frequency of oscillation, amplitude of motion, and form of movement. The purpose of these analyses was to assess their impact on thrust force and the patterns of the eddies on the wake.

Finally, in section 6.3, a more realistic three-dimensional scenario was employed. This involved utilizing a fish geometry resembling a mackerel and applying undulating wave motion to the backbone to replicate the swimming movement. This motion generated vortices with distinct topologies, which were influenced by the frequency of oscillation.

6.1. Fish-like swimming kinematics

To characterize the lateral undulation of the middle line of the swimmer within a two-dimensional framework, consider the backward wave function $y(x,t) : [0, x_L(t)] \times [0, +\infty) \to \mathbb{R}$. The expression encompasses the inclusion of a parameter $\alpha \in \mathbb{R}$ to modulate the amplitude of the undulate waveform, and the time-dependent function $\mathrm{Tr}(t) : [0, +\infty) \to [0, 1]$ to dictate the transient from the initial flat state:

$$y(x,t) = \alpha \mathrm{Tr}(t) A_m(x) \sin\left(2\pi\left(\frac{x}{\lambda} - ft\right) + \phi\right), \tag{6.1}$$

where f is the oscillation frequency, λ is the wavelength, ϕ the phase of the wave, and $A_m(x)$ is the amplitude envelope described by the quadratic curve:

$$A_m(x) = A_0 + A_1 x + A_2 x^2.$$

To ensure the continuity of the motion from the rest configuration, a smooth step function characterized by a sinusoidal waveform was adopted in the transient state:

$$\mathrm{Tr}(t) = \begin{cases} \frac{1}{2} - \frac{1}{2}\cos\left(\frac{t}{T}\pi\right), & \text{if } t \leq T \\ 1, & \text{if } t > T. \end{cases}$$

where $T \in [0, +\infty)$ represents the transient period.

In terms of the geometry of the structure, the parameters outlined in Table 6.1 are employed to characterize the conformal mapping that generates the fish surface, as detailed in section 4.1. As for the kinematics, the parameters governing the undulating wave motion are specified in Table 6.2. These values have been selected to emulate the carangiform fish-like swimming motion, while also accounting for an appropriate transient period.

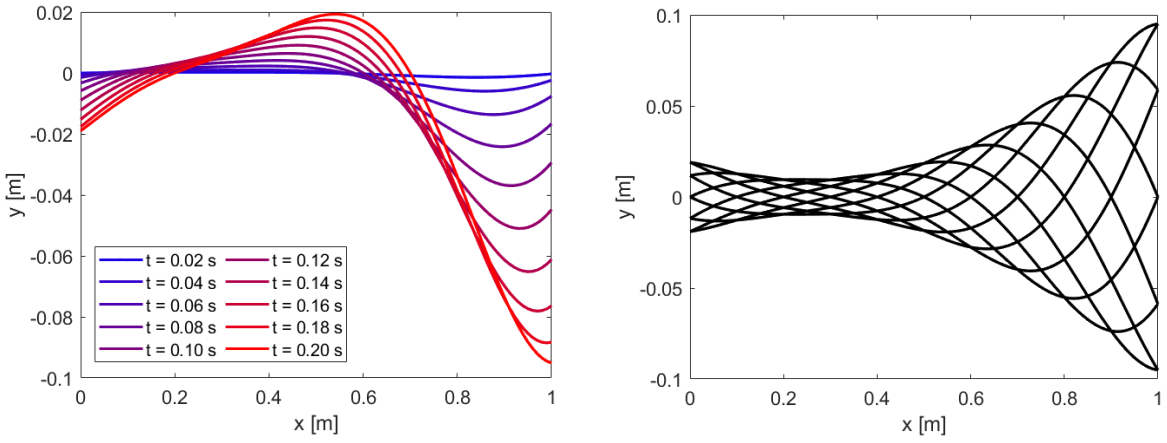
L	η_c	θ
1	-0.04	5°

Table 6.1: Geometric Parameters.

α	A_0	A_1	A_2	λ	ϕ	T
0.1	0.200	-0.825	1.625	1.00	0.0	0.2

Table 6.2: Kinematic Parameters.

For a visual representation of the chosen parameters governing the kinematics of the backbone, refer to Figure 6.1. This figure illustrates the transitional phase of the midline motion from its horizontal initial state and provides insight into the configuration of the backbone throughout a complete cycle.



(a) Transition state

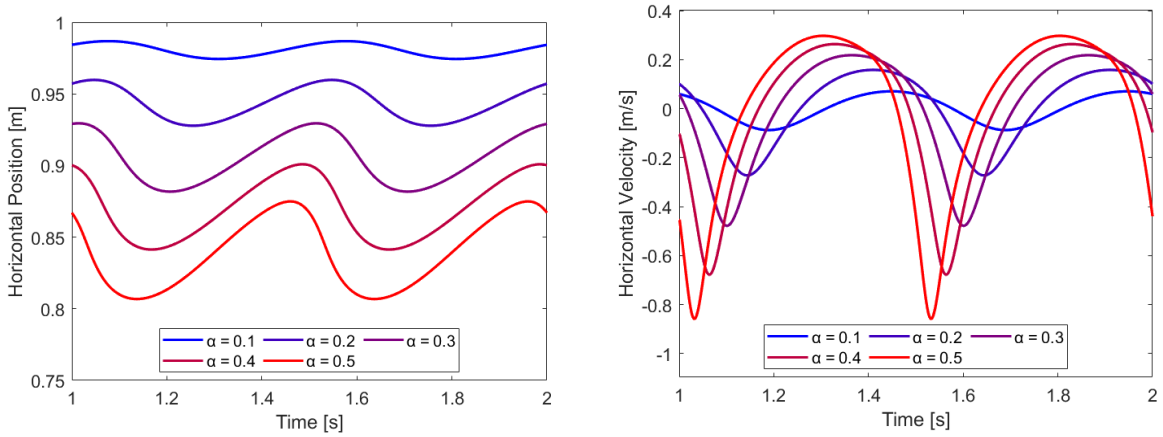
(b) Complete cycle

Figure 6.1: Backbone motion in the transitional state and a cycle of motion.

6.1.1. Swimmer backbone analysis

One of the consequences of the inextensibility constraint is the necessity for the middle line points to experience a horizontal displacement to ensure the maintenance of the backbone at a constant length L. In particular, the analysis will concentrate on the tail extremity point, although it applies to all points along the curve.

As discussed in subsection 4.2.3, the algebraic numerical algorithm is considered for generating the map between Cartesian and curvilinear coordinates. Thus, considering the travelling wave function and the parameters described in section 6.1, it is possible to examine the x-axis coordinate and the horizontal velocity of the backbone points. The results for the tail point in a full cycle of the undulated motion are depicted in Figure 6.2.



(a) Horizontal position of the tail

(b) Horizontal velocity of the tail

Figure 6.2: Tail horizontal position and velocity in a cycle of motion.

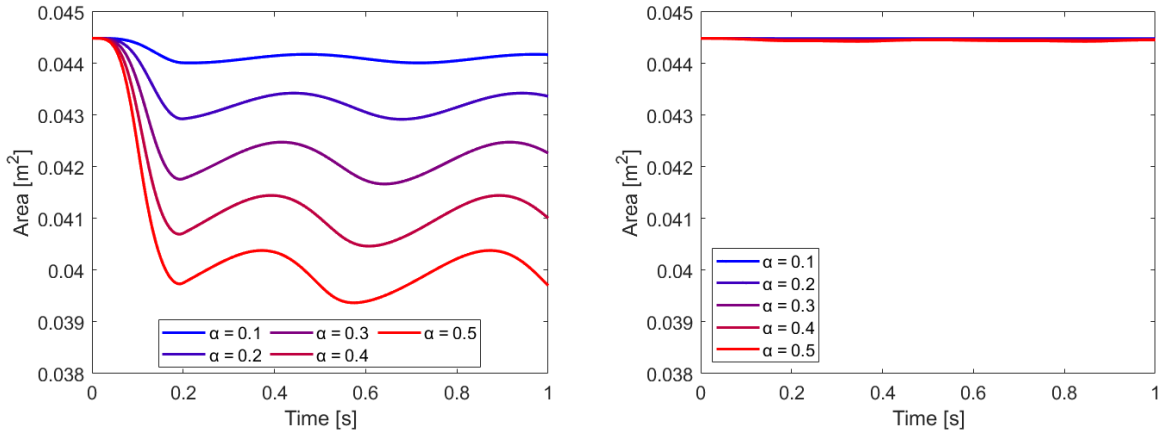
Due to the inextensibility constraint, there is a decrease in the average abscissa of the tail, and this effect becomes more pronounced as the amplitude of motion increases. This phenomenon occurs because, in the flat state, the backbone is at its maximum extensibility, and when the backbone starts to curve, all points undergo a horizontal contraction. Furthermore, note that the horizontal velocity exhibits significant peaks, that will have an impact on the immersed boundary kinematic conditions transmitted by the structure.

6.1.2. Swimmer surface analysis

Considering the geometric and kinematic parameters specified in section 6.1, it becomes possible to define the motion and area of the structure, which in its flat configuration is

approximated as $A = 0.0445$ m^2. Assuming a constant density for the structure, it is essential to ensure the preservation of this area throughout the motion to satisfy the mass conservation constraint.

Due to the inextensibility and the resultant horizontal contraction of the backbone, it is necessary to implement segment rotations to satisfy the mass conservation, as elaborated upon in section 4.3. Through the application of numerical integration, the area of the structure over time can be computed and it is present in Figure 6.3, starting from the flat state and comparing scenarios with and without segment rotations.

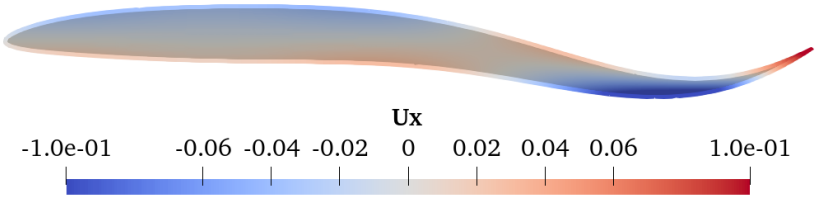


(a) Without segment rotation

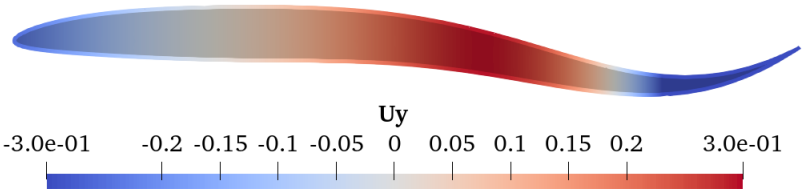
(b) With segment rotation

Figure 6.3: Surface variation in a cycle of motion.

Concerning the structure's velocity field at an arbitrary time instant of the simulation, refer to Figure 6.4. This figure illustrates the horizontal and vertical velocity field components on both the immersed boundary surface and solid cells. At the boundary, the velocity is determined using the direct algorithm of the Lagrangian map, as explained in subsection 4.3.1. In the interior cells, the velocity field is generated using the reconstructed Eulerian field using the algorithm described in subsection 4.3.3.



(a) Horizontal velocity field component



(b) Vertical velocity field component

Figure 6.4: Velocity field components in the immersed boundary surface and solid cells.

Note that the solid cells and the structure's velocity fields are compatible and continuous. It is important to highlight that further verification on a 3D structure with the generalized three-dimensional motion was performed, and the results of the internal cells also adhere to the structure velocity field. This serves as verification of the Eulerian velocity field reconstruction, ensuring the correct prescription of the solid cells values in the immersed boundary method.

6.2. Fish-like swimming 2D simulations

Consider a two-dimensional domain denoted as $\Omega = [x_1, x_2] \times [y_1, y_2]$, which encompasses an immersed flexible body with a fish-like shape and a length of L, with the head positioned at the origin. In the simulation scenario, the boundary conditions include a uniform inflow velocity of U_∞ applied to the left wall, slip conditions on the bottom and top walls, and an outlet condition on the right wall, as depicted in Figure 6.5.

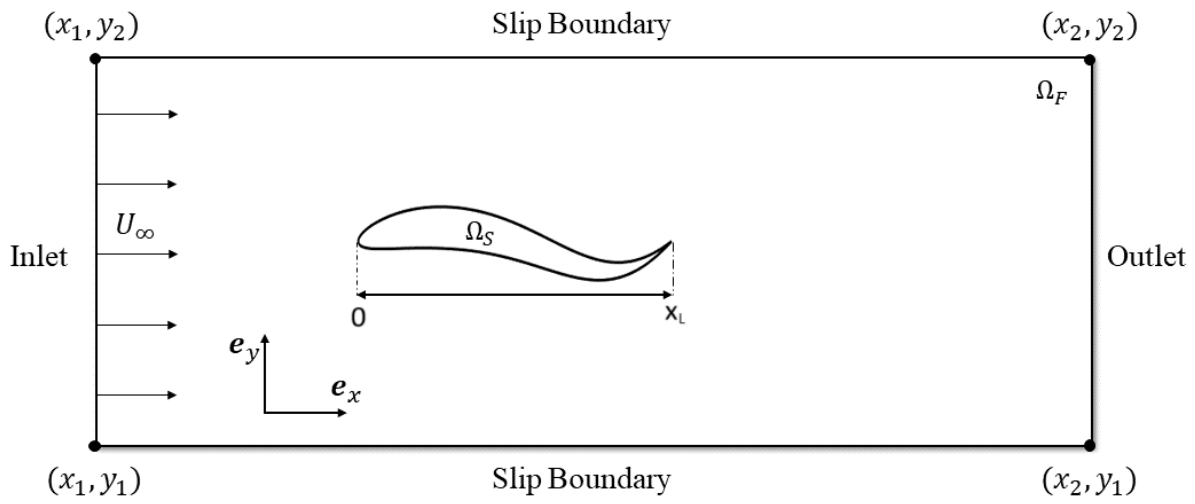


Figure 6.5: Illustration of the two-dimensional domain for fish-like swimming simulation.

An intermediate Reynolds number of $Re = 5000$ was chosen to represent the motion of the aquatic animal. This setup involved using a unit body length $L = 1$, a uniform inflow with fluid velocity $U_\infty = 1$ m/s, and a kinematic viscosity of $\nu = 2 \times 10^{-4}$ m^2/s.

To assess numerical convergence and determine the appropriate mesh size, time step, and domain dimensions, a series of experiments were conducted to ensure the results remained invariant with these parameters.

For the independence analysis, an intermediate setup was defined to resemble the one outlined in subsection 5.2.2. Subsequently, a coarse scenario was generated by increasing the mesh size and time step while using a more compact domain. In contrast, the refined scenario was developed by taking the opposite approach. The numerical values of the discussed parameters are presented in Table 6.3.

Setup	h	Δt	Ω
Coarse	4.0×10^{-2}	5.0×10^{-3}	$[-1, 8] \times [-2, 2]$
Intermediary	2.0×10^{-2}	2.5×10^{-3}	$[-1, 12] \times [-4, 4]$
Refined	1.5×10^{-2}	1.0×10^{-3}	$[-1, 16] \times [-6, 6]$

Table 6.3: Numerical setup configurations for mesh size, time step, and domain dimensions.

With the numerical setup of the simulations established, experiments were conducted with an oscillation frequency of $f = 2.0$ Hz. In all scenarios, three layers of dynamic local mesh

refinement were applied in the vicinity of the fish. Regarding the PIMPLE-IBM method parameters, a maximum of 10 iterations for the outer loop and 3 iterations for the inner loop were defined, without incorporating relaxation for either pressure or velocity.

Regarding the immersed boundary extended stencils parameters, it was considered the *point-to-cell* approach, a maximum connectivity level of $\overline{c} = 3$, a maximum spatial distance as a multiple of IB cell span of $\overline{d} = 2.5$ and a maximum field of view angle of $\overline{\theta} = 90°$.

The results for the drag coefficient in a cycle of motion across different setups are illustrated in Figure 6.6.

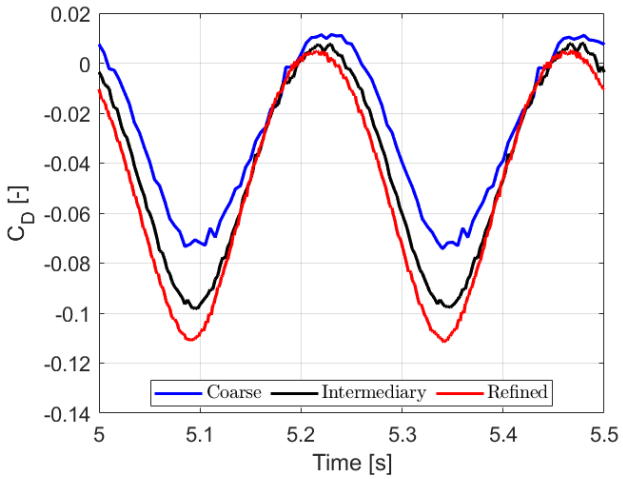
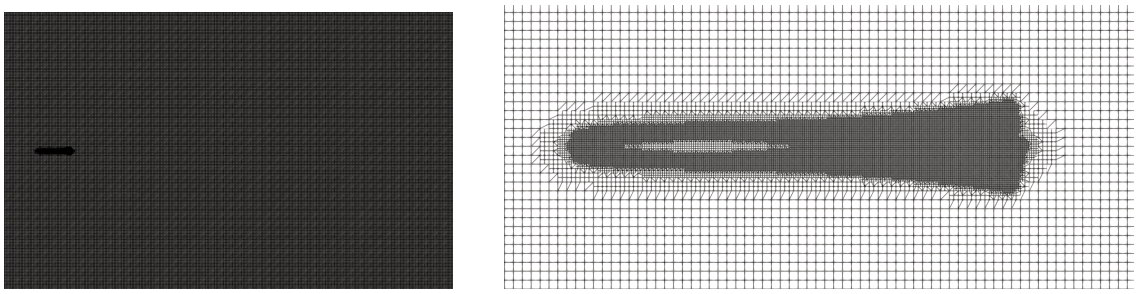


Figure 6.6: Comparison of drag coefficient for different setup configurations.

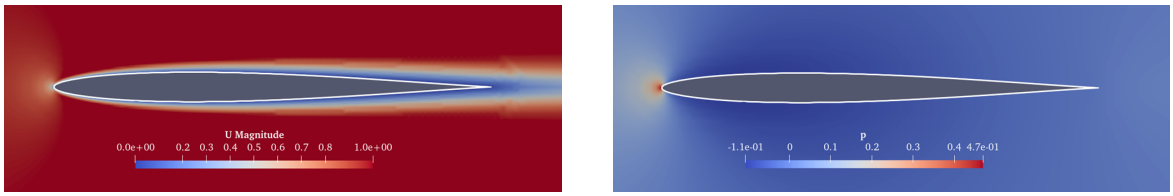
Analyzing the time-dependent drag coefficient results, it is apparent that the solution tends to converge with increasing refinement in the setup. Thus, to achieve an optimal trade-off between computational accuracy and efficiency, the intermediary setup illustrated in Figure 6.7 has been selected for the execution of the two-dimensional analysis in the project.



(a) Grid of the complete domain (b) Grid near fish

Figure 6.7: Mesh of the intermediary setup.

The initial conditions of the simulation are derived from the steady-state results obtained using the SIMPLE-IBM algorithm for the structure in its flat configuration. This outcome dictates the initial velocity and pressure fields in the vicinity of the fish in the flat state. Figure 6.8 provides the steady-state solution of the initial conditions of pressure and velocity fields set for the two-dimensional simulation.



(a) Velocity field (b) Pressure field

Figure 6.8: Initial configuration of the velocity and pressure.

6.2.1. Oscillation frequency analysis

Considering the kinematics and geometry parameters specified in section 6.1 and the simulation setup outlined in section 6.2, a series of simulations were conducted. In this scenario, the oscillation frequency was systematically varied from $f = 0.5 - 2.0$ Hz. The outcomes of these investigations are presented in Figure 6.9, illustrating the mean drag coefficient and its constituent viscous and pressure components.

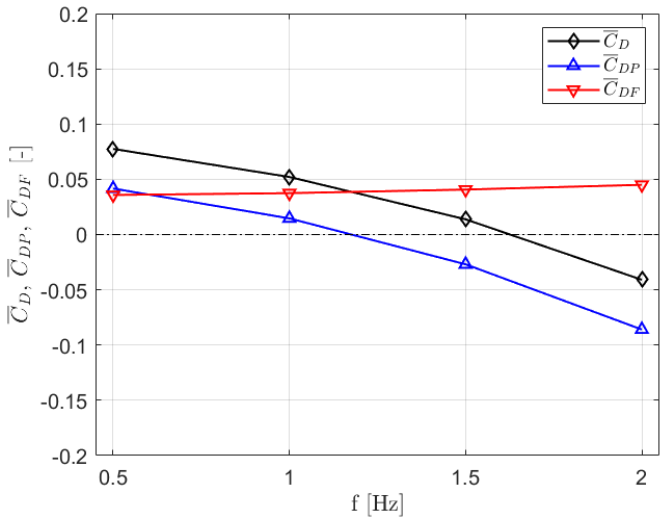
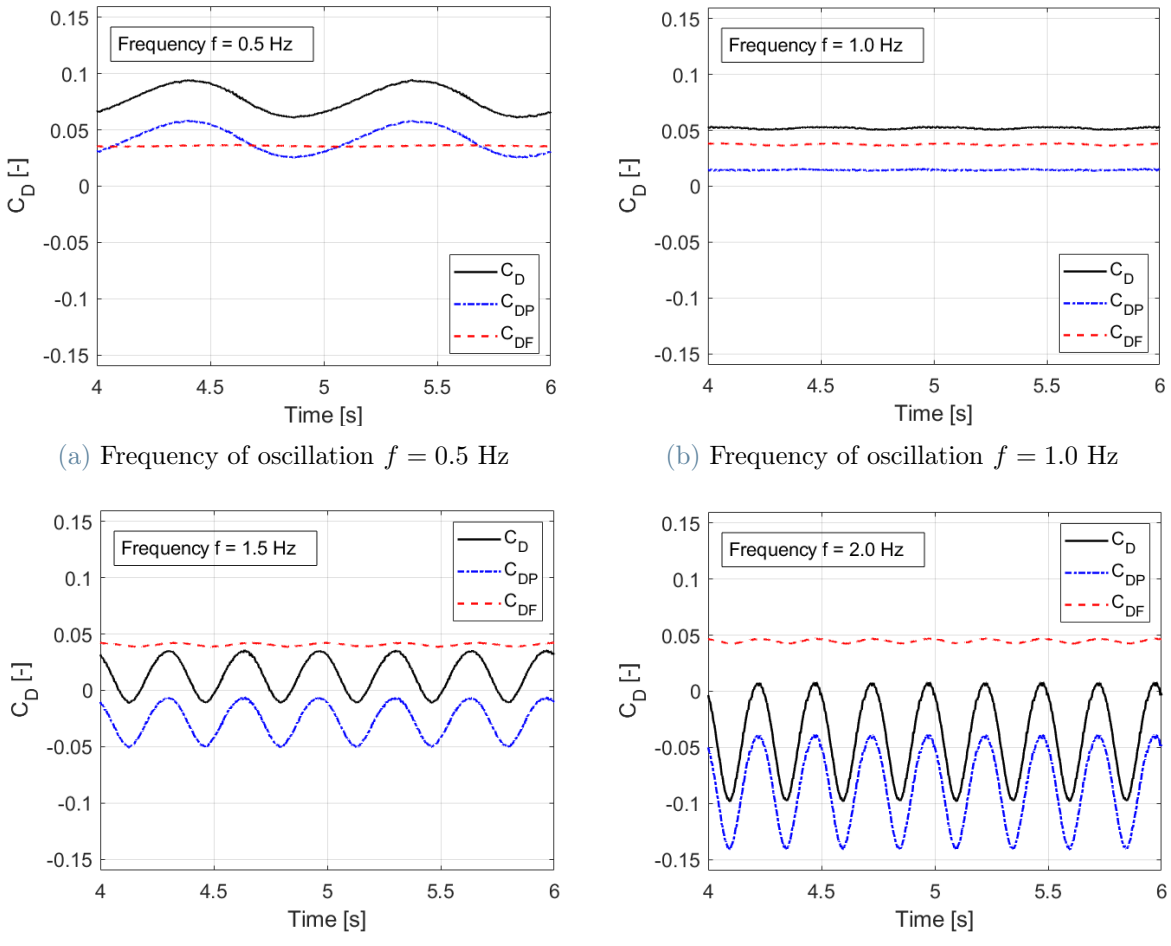


Figure 6.9: Time average drag coefficient for the fish-like swimming as a function of frequency of oscillation.

For a motion amplitude of $\alpha = 0.1$, the incorporation of inextensibility and mass conser-

vation constraints is not expected to significantly impact the simulation when compared to the simplified travelling wave model with null horizontal displacement as outlined in subsection 5.2.2. This observation is supported by the data presented in Figure 6.2, which indicates minimal changes in the motion of the backbone.

The results indicate a decreasing mean pressure drag coefficient $\overline{C}_{DP}$ with an increasing trend in the mean viscous drag coefficient $\overline{C}_{DF}$ as the frequency increases. In general, a decreasing trend is observed in the total drag coefficient $\overline{C}_D$, reaching a negative value at approximately $\overline{C}_D = 1.6$, implying that the drag force change direction and starts to act as a thrust force. For enhanced visualization of the results, Figure 6.10 presents the total drag coefficient and its components for different frequencies over a period sufficient to observe a periodic cycle.



(a) Frequency of oscillation $f = 0.5$ Hz

(b) Frequency of oscillation $f = 1.0$ Hz

(c) Frequency of oscillation $f = 1.5$ Hz

(d) Frequency of oscillation $f = 2.0$ Hz

Figure 6.10: Fish-swimming time-dependent drag force coefficients for the frequencies of oscillation $f = 0.5, 1.0, 1.5, 2.0$ Hz.

It is possible to notice that the viscous force contributions are almost constant over time, which reveals that the undulating profile of the drag coefficient is related to the pressure. In addition, there is a noticeable trend in the amplitude of the total drag coefficient, which possesses an almost constant value at $f = 1.0$ Hz and starts to increase with higher frequencies.

Further analysis of the simulation patterns can be performed. In Figure 6.11, the results for the lift coefficient throughout multiple cycles of motion at different frequencies are depicted. The lift coefficient exhibits a sinusoidal wave pattern with an amplitude that increases in tandem with the swimmer's oscillation.

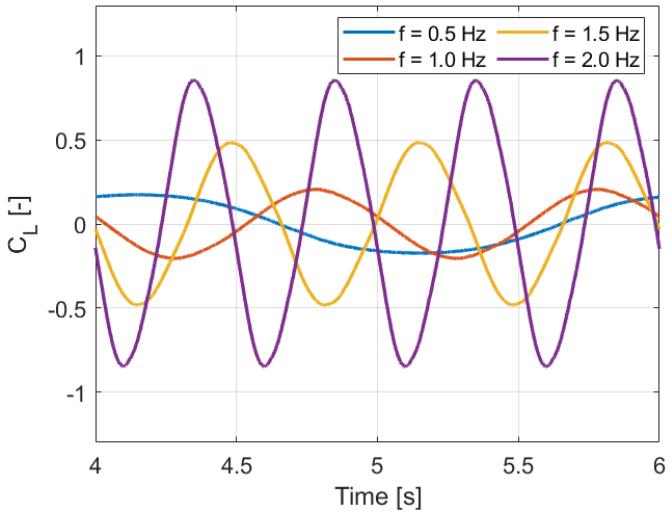
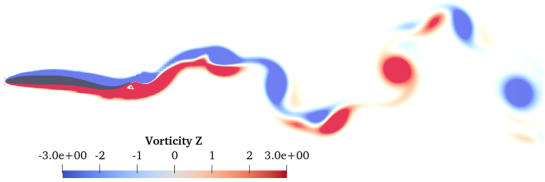


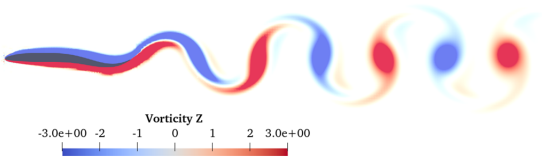
Figure 6.11: Fish-swimming time-dependent lift force coefficients for the frequencies of oscillation $f = 0.5, 1.0, 1.5, 2.0$ Hz.

Notably, the amplitude of the lift increases as the frequency of the motion rises. Coupled with the higher velocity at elevated frequencies, this necessitates additional internal power to execute the oscillation. Consequently, evaluating the efficiency of the motion requires a balance between the power associated with thrust and the energy required for the lateral oscillation.

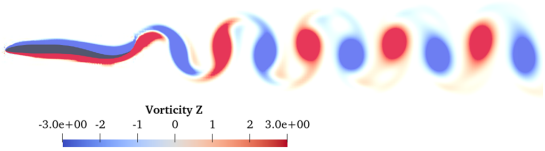
Finally, an essential characteristic of the oscillation frequency is its impact on the vorticity patterns in the wake of the swimmer. As depicted in Figure 6.11, the visualization of fluid vorticity shows distinct patterns of swirling structures with opposite directions. At lower frequencies, the vortices exhibit periodicity with significant spacing, whereas higher oscillation frequencies result in more pronounced and closely spaced wake patterns.



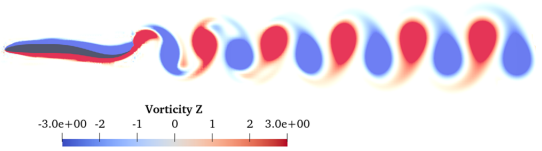
(a) Vorticity profile at a frequency $f = 0.5$ Hz



(b) Vorticity profile at a frequency $f = 1.0$ Hz



(c) Vorticity profile at a frequency $f = 1.5$ Hz



(d) Vorticity profile at a frequency $f = 2.0$ Hz

Figure 6.12: Vorticity profile of the wavy body-fish airfoil at different frequencies.

6.2.2. Oscillation amplitude analysis

The amplitude of the undulate waveform has a notable impact on the motion, inducing alterations in the dynamics, one example is the horizontal contraction of the backbone points, as depicted in Figure 6.2. To explore this parameter, a series of experiments were conducted, varying the amplitude, while keeping all other simulation settings constant. The results of the mean average drag coefficient for various frequencies and amplitudes are illustrated in Figure 6.13.

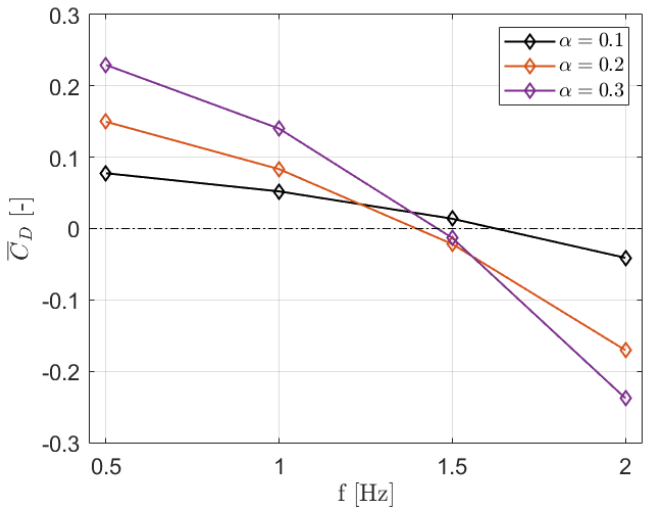
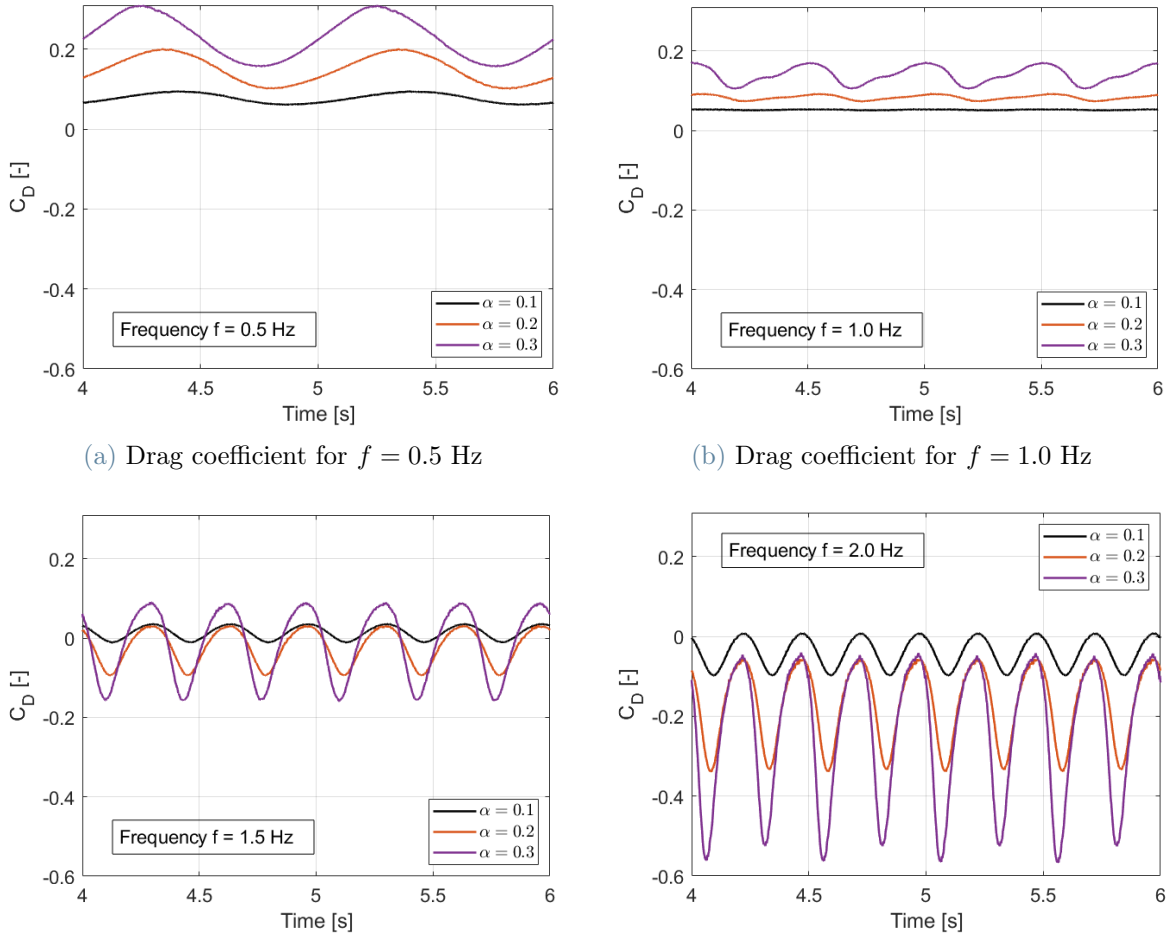


Figure 6.13: Comparison of the time average drag coefficient for the fish-like swimming for different amplitude as a function of frequency of oscillation.

The results show that an increase in amplitude amplifies the mean effect of the drag force, leading to increased drag forces at low frequencies, which negatively impacts swimming. Conversely, at higher frequencies it is noticeable that the thrust forces increase, propelling the fish and generating positive feedback for the swim. For a more refined analysis, Figure 6.14 provides a visual comparison of the drag coefficient throughout a complete cycle of motion.



(a) Drag coefficient for $f = 0.5$ Hz

(b) Drag coefficient for $f = 1.0$ Hz

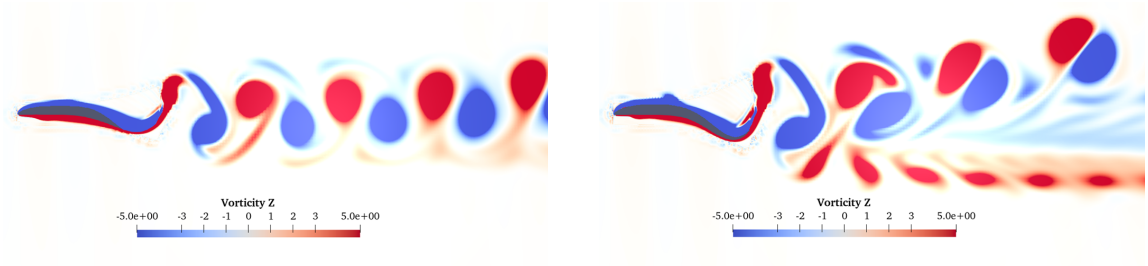
(c) Drag coefficient for $f = 1.5$ Hz

(d) Drag coefficient for $f = 2.0$ Hz

Figure 6.14: Time-dependant drag coefficient for different amplitudes for the frequencies of $f = 0.5, 1.0, 1.5, 2.0$ Hz.

In the analysis of the time-dependent drag coefficient, it becomes evident that, in addition to the trend of the mean drag coefficient, there is an increase in the amplitude of the oscillations. Furthermore, as the amplitude of motion increases, the sinusoidal shape of the curve of the drag coefficient value becomes less pronounced, reflecting the influence of the complex motion introduced by the effects of inextensibility and mass conservation.

Finally, to investigate the dynamic changes induced by varying the amplitude, Figure 6.15 displays the vorticity profile in the wake of the swimmer with increasing amplitudes.



(a) Vorticity Profile at $\alpha = 0.2$ (b) Vorticity Profile at $\alpha = 0.3$

Figure 6.15: Vorticity profile different amplitudes of the fish-like swimming motion at a frequency of $f = 2.0$ Hz.

Notably, when considering an amplitude of $\alpha = 0.2$, the vortices resemble the profile observed with an amplitude of $\alpha = 0.1$. In contrast, with a higher amplitude of $\alpha = 0.3$, the wake undergoes significant modification, in which the eddies are pushed upward, and the emergence of lines of vortices with the same direction indicates a distinct flow pattern.

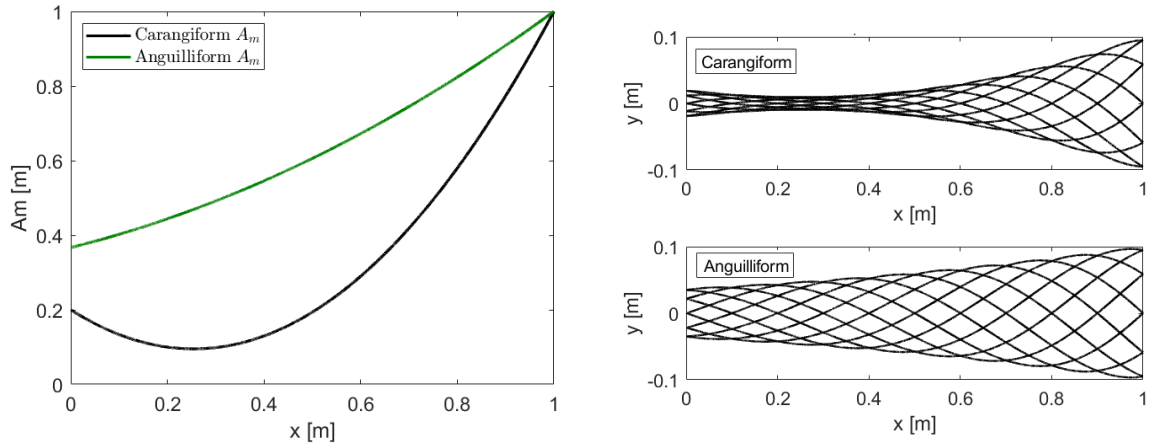
6.2.3. Swimming motion form analysis

The preceding results were based on a specific motion characterization known as carangiform. However, in the study of fish-like swimming, various motion configurations are considered, leading to the categorization of carangiform and anguilliform groups, as detailed in [24]. The differentiation between these motion types lies in the parameters defining the amplitude envelope of the travelling wave equation, as presented in Table 6.4.

Motion	α	A_0	A_1	A_2	λ
Carangiform	0.1	0.200	-0.825	1.625	1
Anguilliform	0.1	0.367	0.323	0.310	1

Table 6.4: Kinematic parameters of the carangiform and anguilliform motion.

In Figure 6.16, the difference in the amplitude envelope and its impact on the backbone motion is illustrated. It is important to note that, while the movements are similar in the tail region, the anguilliform motion exhibits a more pronounced amplitude at the head and in the middle.



(a) Amplitude envelope

(b) Cycle of the travelling-wave

Figure 6.16: Backbone motion of the carangiform and anguilliform.

Defined the kinematics for both types of motion, simulations were conducted to gain insights into the dynamics of each distinct swimmer. The results, depicted in Figure 6.17, express the mean average drag coefficient for different frequencies and motion configurations.

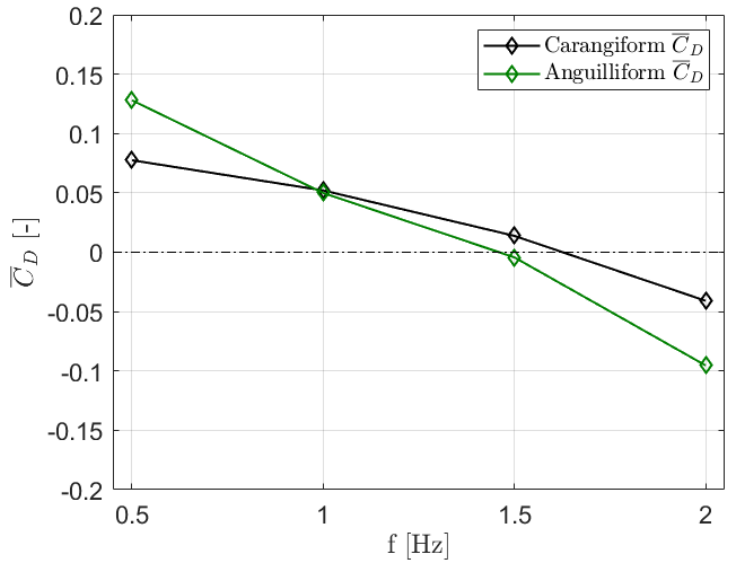
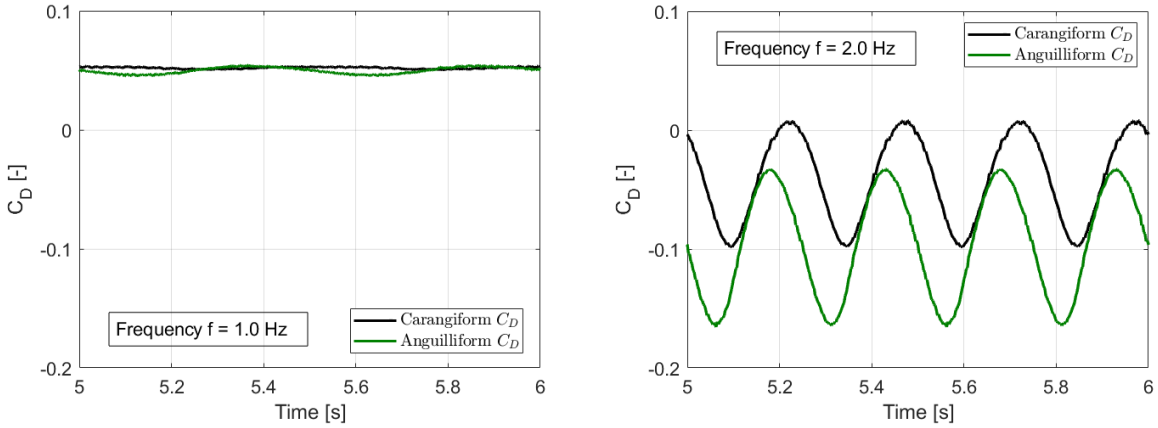


Figure 6.17: Comparison of the time average drag coefficient for the fish-like swimming carangiform and anguilliform as a function of frequency of oscillation.

The results demonstrate that at lower frequencies, the carangiform motion reduces drag force in comparison to the anguilliform, thereby improving swimming efficiency. On the other hand, at higher frequencies, the anguilliform motion outperforms by providing a

greater thrust force, propelling the fish, and facilitating its motion. For a more refined analysis, Figure 6.18 provides a visual representation of the drag coefficient throughout a complete cycle of motion.

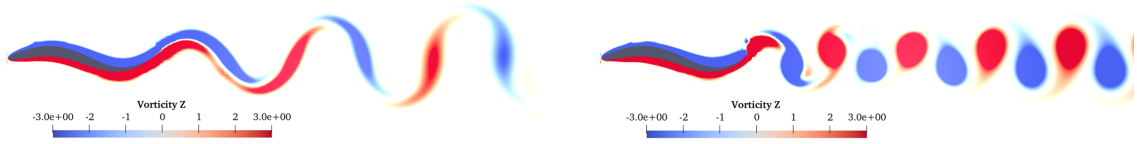


(a) Drag coefficient for $f = 1.0$ Hz (b) Drag coefficient for $f = 2.0$ Hz

Figure 6.18: Time-dependant drag coefficient for the carangiform and anguilliform motion for the frequencies of $f = 1.0$ Hz and $f = 2.0$ Hz.

Notably, at a frequency of $f = 1.0$ Hz, the dynamic response of the fluid yields a nearly identical constant drag coefficient for both cases. On the other hand, as the frequency rises to $f = 2.0$ Hz, the anguilliform motion exhibits a drag coefficient oscillation similar in amplitude to the carangiform, yet notably lower in mean value, signifying a more substantial thrust force, which also possesses a different phase.

Finally, regarding the vorticity profile of the anguilliform motion, as depicted in Figure 6.19, it does not exhibit significant differences from the carangiform form illustrated in Figure 6.12.



(a) Vorticity Profile at a frequency $f = 1.0$ Hz (b) Vorticity Profile at a frequency $f = 2.0$ Hz

Figure 6.19: Vorticity profile of anguilliform fish-like swimming motion.

6.3. Fish-like swimming 3D simulations

In this section, the objective is to simulate fish-like swimming within a three-dimensional scenario, striving for more realistic results. It is important to note that 3D simulations are more computationally expensive compared to two-dimensional scenarios. Therefore, considering the computational resources available, a limited number of cases will be analyzed.

To define a realistic geometry for the fish in the reference configuration, the 3D model referenced in [22] was adopted. The details, including the fins, were removed from the model, and its size and position were adjusted to achieve a unitary length, positioning the head at the origin. Figure 6.20 illustrates the geometry and size of the 3D model that will be incorporated in the simulations.

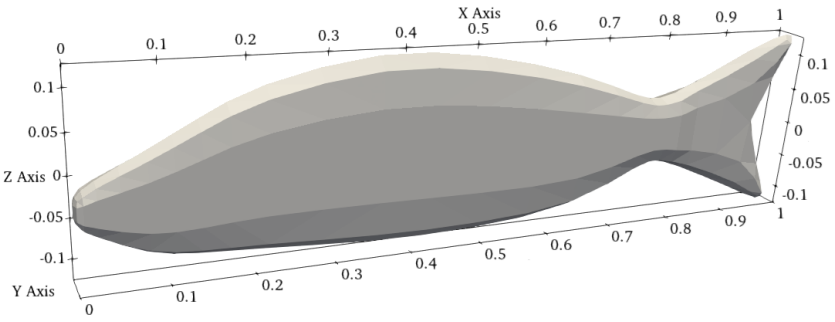


Figure 6.20: 3D geometry of a mackerel-inspired fish.

Consider a three-dimensional domain denoted as $\Omega = [x_1, x_2] \times [y_1, y_2] \times [z_1, z_2]$, and the subdomains Ω_1, Ω_2 and Ω_3 that define the regions of local refinement of the mesh, such that $\Omega \supset \Omega_1 \supset \Omega_2 \supset \Omega_3$. The decision to employ prescribed subdomains for refinement in 3D simulations is motivated by the necessity to maintain better control of the scheme employing a fixed number of elements and to enforce particular emphasis on the wake region.

Regarding the boundary conditions, these consist of a uniform inflow velocity of U_∞ applied to the left wall, slip conditions on the bottom, top, front, and back walls, and an outlet condition on the right wall. Figure 6.21 illustrates the described scenario for the simulation of fish-like swimming in three dimensions.

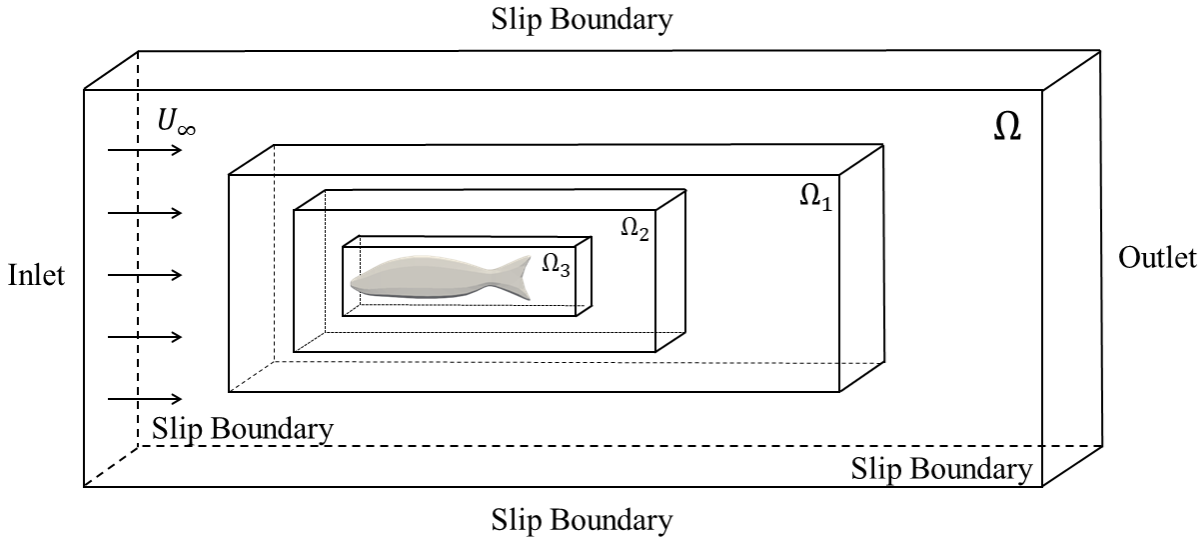


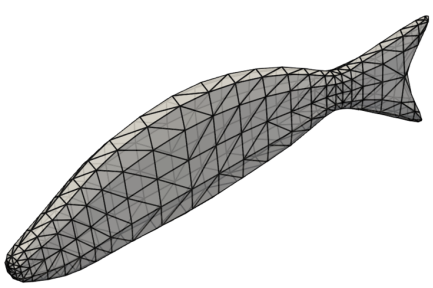
Figure 6.21: Illustration of the three-dimensional domain with local refinement for the fish-like swimming simulation.

The specific numerical values used to define the boundaries of the domain and the size h of the isotropic mesh element are outlined in Table 6.5. These specifications were chosen after a preliminary examination of the wake, focusing on identifying areas requiring refinement. Subsequently, the subdomains were delineated as follows: Ω_1 encompasses the entire wake region to be analyzed, Ω_2 targets the examination of eddies near the tail, and Ω_3 captures the fish's movement with a higher degree of refinement.

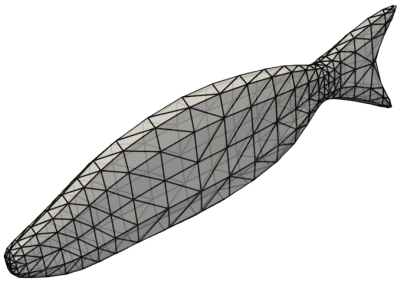
Domain	$[x_1, x_2]$	$[y_1, y_2]$	$[z_1, z_2]$	h
Ω	$[-1.0, 10]$	$[-2.0, 2.0]$	$[-2.0, 2.0]$	0.04
Ω_1	$[-0.5, 5.0]$	$[-1.0, 1.0]$	$[-0.4, 0.4]$	0.02
Ω_2	$[-0.2, 3.0]$	$[-0.6, 0.6]$	$[-0.3, 0.3]$	0.01
Ω_3	$[-0.1, 1.1]$	$[-0.3, 0.3]$	$[-0.2, 0.2]$	0.005

Table 6.5: Domain configuration of the local refinement.

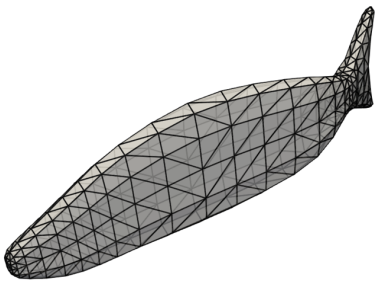
Concerning the imposed kinematics of the fish, the model of inextensibility and mass conservation described in subsection 4.3.2 will be considered. The backward wave function for $y(x,t)$ as indicated in (6.1) will be used, considering the parameters of the carangiform indicated in table 6.2. In addition, a null displacement of the backbone at $z(x,t) = 0$ and $\varphi(x,t) = 0$, will be taken into consideration. The chosen parameters produce a kinematic profile that resembles that of a fish, as illustrated in Figure 6.22.



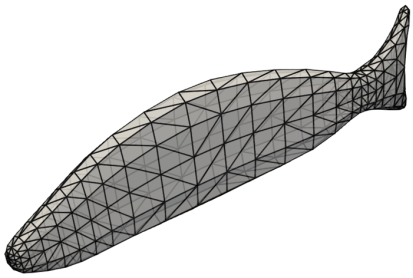
(a) Swimming 3D Fish at $t = T/4$



(b) Swimming 3D Fish at $t = T/2$



(c) Swimming 3D Fish at $t = 3T/4$



(d) Swimming 3D Fish at $t = T$

Figure 6.22: Swimming 3D Fish in one period T.

The specific set of parameters chosen was applied to represent the oscillatory caudal movement of a fish. However, the implemented code is generalized to consider $z(x,t)$ and $\varphi(x,t)$, which implies that it could be modified to represent the helicoidal movement of various bodies, such as sperm.

Regarding the time discretization, a time step of $\Delta t = 0.005$ s was selected, and all simulations were carried out until reaching a time of 5 seconds. The maximum Courant number at the highest frequency of the simulation was $\max(Co) = 2.0$. Thus, to achieve convergence, a total of 10 outer loops and 3 inner loops were executed in each iteration.

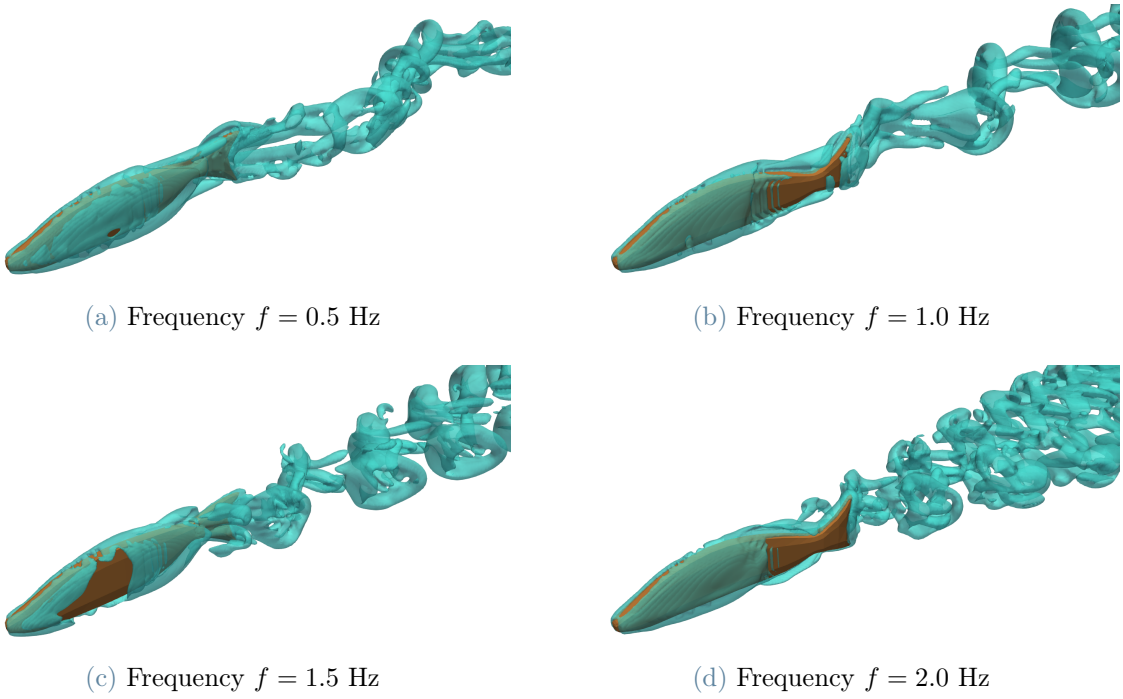
In terms of the scenario setup, an intermediate Reynolds number of $Re = 5000$ was selected to portray the motion of the aquatic animal. This involved employing a unit body length of $L = 1$, establishing a uniform inflow with a fluid velocity of $U_\infty = 1$ m/s, and utilizing a kinematic viscosity of $\nu = 2 \times 10^{-4}$ m^2/s.

The Q-Criterion method, detailed in [48], is employed for vortex identification in the flowfield. It involves comparing the magnitudes of vorticity and the strain rate, through the scalar:

$$Q = \frac{1}{2}(\|\mathbf{\Omega}\| - \|\mathbf{S}\|),$$

where $\mathbf{\Omega} = \text{Sym}(\nabla \mathbf{u})$ is the vorticity tensor, $\mathbf{S} = \text{Skw}(\nabla \mathbf{u})$ is the strain-rate tensor, and $\|\mathbf{H}\| = (\text{tr}(\mathbf{H}\mathbf{H}^T))^{1/2}$ is the Frobenius tensor norm.

The simulations were performed on the High-Performance Computing (HPC) system at the Modelling and Scientific Computing department of Politecnico di Milano, using 20 cores, which solved each simulation in approximately 23 hours. Conducting three-dimensional simulations of fish-like swimming and applying a Q-criterion threshold of 0.1, the resulting wake eddies are depicted in Figure 6.23.



(a) Frequency $f = 0.5$ Hz

(b) Frequency $f = 1.0$ Hz

(c) Frequency $f = 1.5$ Hz

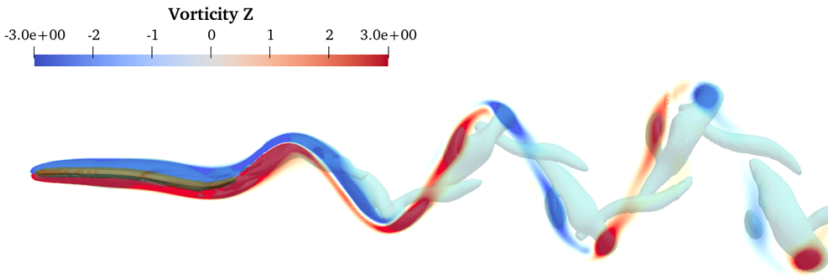
(d) Frequency $f = 2.0$ Hz

Figure 6.23: Three-dimensional vortical structures using the Q-criterion of $Q = 0.1$ for the frequencies of oscillation $f = 0.5, 1.0, 1.5, 2.0$ Hz.

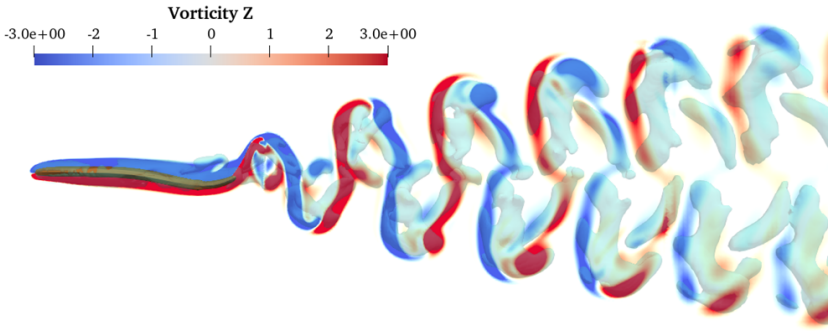
In the analysis of the wake's topology, it is observed that at lower frequencies, a single trail of eddies emerges, oscillating over the center line of the fish. Conversely, at higher frequencies, these eddies bifurcate into two distinct lines that expand laterally. This observation is consistent with the results reported in the numerical simulations conducted by [6]. This phenomenon can be attributed to the fact that at higher frequencies, the fish's tail exhibits greater lateral velocity, leading to the lateral dispersion of the fluid eddies.

To gain deeper insights into the vorticity pattern, horizontal (z-direction) and vertical (y-direction) cross-sections were analyzed. Figure 6.24 depicts a horizontal cross-section

through the middle of the fish, offering a perspective similar to the 2D simulation previously conducted.



(a) Frequency $f = 1.0$ Hz

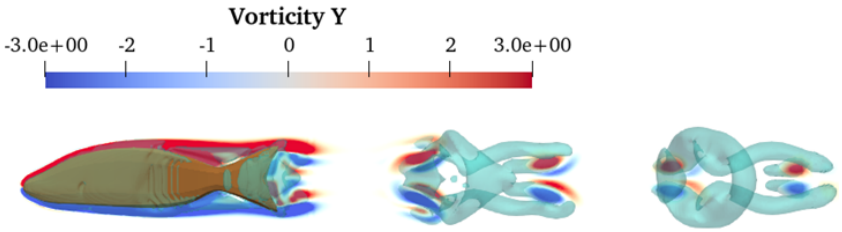


(b) Frequency $f = 2.0$ Hz

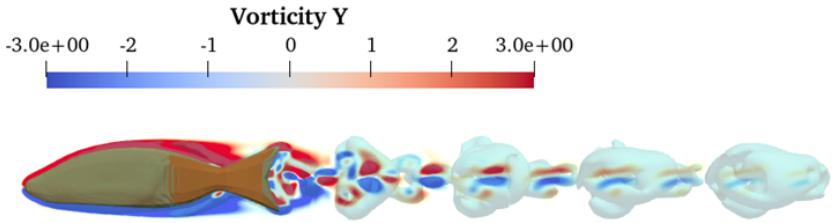
Figure 6.24: Z-plane cross-section vorticity profile in three-dimensional fish-like swimming compared with the Q-criterion profile of $Q = 0.1$.

When comparing the results with the 2D simulation depicted in Figure 6.12, there is a noticeable similarity between the wake structures. Particularly, at a frequency of $f = 1.0$ Hz, patterns observed closely resemble those observed in the 2D simulation. At a frequency of $f = 2.0$ Hz in the 3D simulation, deeper insights emerge as a lateral separation into two distinct wakes becomes evident. This results in a high-frequency alternating pattern in the vorticity direction, resembling the 2D simulation, albeit with a higher amplitude.

Furthermore, in the 3D simulations, it is possible to analyze the perspective perpendicular to the fish. Figure 6.25 offers a lateral view of the structure, highlighting the vorticity in the Y direction and providing additional insights into the Q-criterion results.



(a) Frequency $f = 1.5$ Hz



(b) Frequency $f = 2.0$ Hz

Figure 6.25: Y-plane cross-section vorticity profile in three-dimensional fish-like swimming compared with the Q-criterion profile of $Q = 0.1$.

By evaluating a lateral perspective of the swimming, it becomes apparent that at the lower frequency of $f = 1.0$ Hz, there is a region of null vorticity in the Y direction between the eddies. The pattern also reveals that the vorticity is consistently positive at the top and consistently negative at the bottom, which mirrors the behavior near the structure. As the frequency of the oscillation increases, the separation space between the eddies decreases, appearing more continuous, while the pattern of top and bottom vorticity remains unchanged.

7 | Conclusions and future developments

In this concluding remarks, we recall the key research findings regarding fish-like swimming simulations through the application of the Immersed Boundary Method. Additionally, we discuss the limitations of the study while presenting potential ideas for future research.

The proposed methodology employed a mathematical model based on the unsteady incompressible Navier-Stokes equations, incorporating the immersed body effect through a non-conforming mesh employing the Immersed Boundary Method. The *open-source* software OpenFOAM was utilized, which is based on the Finite Volume Method in conjunction with the PIMPLE-IBM algorithm to compute numerical solutions. The accuracy and reliability of the employed methodology were confirmed through the verification process involving benchmark case simulations.

To develop a kinematic model for fish-like swimming, an algorithm was developed that encompasses both two-dimensional and three-dimensional simulations. Based on the travelling-wave undulation, it modifies the standard model by incorporating the constraints of inextensibility and mass conservation. Hence, the fish-like swimming results were generated through the establishment of a computational setup involving domain and time discretization, followed by post-processing to analyze the desired results.

The study developed algorithms in OpenFOAM using the C++ language, built upon the previously elaborated kinematic model for the rigid body. This allowed for the efficient generation of a variety of tests, with different frequencies, amplitudes, and types of motion. This approach facilitated an investigation into how these parameters influence the drag coefficient, thus offering valuable insights into the propulsive force of swimming and the resultant vorticity patterns in the wake.

The study has certain limitations, including the absence of the self-propelling mode of swimming, which accounts for feedback mechanisms that both rotate and translate the center of mass within the structure. Additionally, the simulation setup did not incorporate a turbulence model; instead, it assumed laminar flow conditions. Finally, the scope of 3D

simulations was constrained due to limitations in available cluster resources and processing time, which in turn affected both spatial and temporal discretization to generate a more refined result.

Hence, a compelling direction for further research lies in leveraging the implemented model to advance the understanding of the fluid dynamics associated with swimming. By incorporating feedback mechanisms, simulations could be performed to identify acceleration, velocity, and power exertion in swimming, potentially leading to the discovery of optimal swimming mechanisms. Furthermore, investigations into scenarios involving multiple fish would explore the effects of wake interactions within a school of fish. To enhance the accuracy of results, the research can extend into 3D simulations using a more refined setup and explore a broader range of motions. Finally, to validate the findings, empirical experiments can be conducted for a comparison with the numerical data using a realistic case.

In conclusion, this project developed an algorithm for simulating deformable bodies within OpenFOAM, employing the Immersed Boundary Method. The customization of structure and motion parameters allowed for a comprehensive exploration of fish-like swimming scenarios, leading to valuable insights into swimming dynamics through the analysis of the generated forces and vorticity profiles.

Bibliography

[1] T. Ahmed, M. T. Amin, S. R. Islam, and S. Ahmed. Computational study of flow around a NACA 0012 wing flapped at different flap angles with varying mach numbers. *Glob J Res Eng*, 13(4):4–16, 2014.

[2] D. Barrett, M. Triantafyllou, D. Yue, M. Grosenbaugh, and M. Wolfgang. Drag reduction in fish-like locomotion. *Journal of Fluid Mechanics*, 392:183–212, 1999.

[3] M. Bergmann and A. Iollo. Modeling and simulation of fish-like swimming. *Journal of Computational Physics*, 230(2):329–348, 2011.

[4] M. Bergmann, L. Cordier, and J.-P. Brancher. Optimal rotary control of the cylinder wake using proper orthogonal decomposition reduced-order model. *Physics of fluids*, 17(9), 2005.

[5] J. Blom. Some characteristic quantities of Karman-Trefftz profiles. Technical report, NASA, 1983.

[6] I. Borazjani and F. Sotiropoulos. Numerical investigation of the hydrodynamics of carangiform swimming in the transitional and inertial flow regimes. *Journal of experimental biology*, 211(10):1541–1558, 2008.

[7] M. Braza, P. Chassaing, and H. H. Minh. Numerical study and physical analysis of the pressure and velocity fields in the near wake of a circular cylinder. *Journal of fluid mechanics*, 165:79–130, 1986.

[8] H.-B. Deng, Y.-Q. Xu, D.-D. Chen, H. Dai, J. Wu, and F.-B. Tian. On numerical modeling of animal swimming and flight. *Computational Mechanics*, 52:1221–1242, 2013.

[9] J. Deng, X.-M. Shao, and Z.-S. Yu. Hydrodynamic studies on two traveling wavy foils in tandem arrangement. *Physics of fluids*, 19(11), 2007.

[10] J. Donea, S. Giuliani, and J.-P. Halleux. An arbitrary Lagrangian–Eulerian finite element method for transient dynamic fluid-structure interactions. *Computer methods in applied mechanics and engineering*, 33(1-3):689–723, 1982.

[11] J. Donea, A. Huerta, J.-P. Ponthot, and A. Rodríguez-Ferran. Arbitrary Lagrangian–Eulerian methods. *Encyclopedia of computational mechanics*, 2004.

[12] G.-J. Dong and X.-Y. Lu. Numerical analysis on the propulsive performance and vortex shedding of fish-like travelling wavy plate. *International Journal for Numerical Methods in Fluids*, 48(12):1351–1373, 2005.

[13] G.-J. Dong and X.-Y. Lu. Characteristics of flow over traveling wavy foils in a side-by-side arrangement. *Physics of fluids*, 19(5), 2007.

[14] J. H. Ferziger, M. Perić, and R. L. Street. *Computational methods for fluid dynamics*, volume 3. Springer, 2002.

[15] G. B. Folland. *Real analysis: modern techniques and their applications*, volume 40. John Wiley & Sons, 1999.

[16] L. Formaggia, F. Nobile, et al. A stability analysis for the arbitrary Lagrangian Eulerian formulation with finite elements. *EAST-WEST JOURNAL OF MATHEMATICS*, 7:105–132, 1999.

[17] S. Forte, L. Preziosi, M. Vianello, et al. *Meccanica dei continui*. Springer, 2019.

[18] J. Gray. Studies in animal locomotion: I. the movement of fish with special reference to the eel. *Journal of experimental biology*, 10(1):88–104, 1933.

[19] J.-W. He, R. Glowinski, R. Metcalfe, A. Nordlander, and J. Periaux. Active control and drag optimization for flow past a circular cylinder: I. oscillatory cylinder rotation. *Journal of Computational Physics*, 163(1):83–117, 2000.

[20] R. D. Henderson. Nonlinear dynamics and pattern formation in turbulent wake transition. *Journal of fluid mechanics*, 352:65–112, 1997.

[21] G. Iaccarino and R. Verzicco. Immersed boundary technique for turbulent flow simulations. *Appl. Mech. Rev.*, 56(3):331–347, 2003.

[22] Kaangvl. Fish (Low Poly) 3D Model. `https://free3d.com/3d-model/3d-fish-model-low-poly-63627.html`, 2019. [Online; accessed 01-November-2023].

[23] S. Li, C. Li, L. Xu, W. Yang, and X. Chen. Numerical simulation and analysis of fish-like robots swarm. *Applied Sciences*, 9(8):1652, 2019.

[24] A. P. Maertens, A. Gao, and M. S. Triantafyllou. Optimal undulatory swimming for a single fish-like body and for a pair of interacting swimmers. *Journal of Fluid Mechanics*, 813:301–345, 2017.

[25] F. L. Markley and J. L. Crassidis. *Fundamentals of spacecraft attitude determination and control*, volume 1286. Springer, 2014.

[26] J. S. MARTíN, J.-F. Scheid, T. Takahashi, and M. Tucsnak. An initial and boundary value problem modeling of fish-like swimming. *Archive for rational mechanics and analysis*, 188:429–455, 2008.

[27] F. Moukalled, L. Mangani, M. Darwish, F. Moukalled, L. Mangani, and M. Darwish. *The finite volume method*. Springer, 2016.

[28] B. Munson, A. Rothmayer, T. Okiishi, and W. Huebsch. *Fundamentals of Fluid Mechanics*. Wiley, 2012. ISBN 9781118116135. URL `https://books.google.it/books?id=BSEpygEACAAJ`.

[29] U. K. Müller, B. L. E. Van Den Heuvel, E. J. Stamhuis, and J. J. Videler. Fish Foot Prints: Morphology and Energetics of the Wake Behind a Continuously Swimming Mullet (Chelon Labrosus Risso). *Journal of Experimental Biology*, 200(22):2893–2906, 11 1997. ISSN 0022-0949. doi: 10.1242/jeb.200.22.2893. URL `https://doi.org/10.1242/jeb.200.22.2893`.

[30] G. Negrini. Non-conforming methods for the simulation of industrial polymer mixing processes. *Ph.D. thesis, Politecnico di Milano*, 2023.

[31] G. Negrini, N. Parolini, and M. Verani. The Rhie-Chow stabilized box method for the stokes problem. *arXiv preprint arXiv:2308.01059*, 2023.

[32] Y. Ohkami. Spacecraft dynamics. In R. A. Meyers, editor, *Encyclopedia of Physical Science and Technology (Third Edition)*, pages 431–448. Academic Press, New York, third edition edition, 2003. ISBN 978-0-12-227410-7. doi: https://doi.org/10.1016/B0-12-227410-5/00898-X. URL `https://www.sciencedirect.com/science/article/pii/B012227410500898X`.

[33] S. Patankar. Numerical heat transfer and fluid flow hemisphere publishing corporation. *New York*, 1980.

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

[35] C. S. Peskin. Numerical analysis of blood flow in the heart. *Journal of computational physics*, 25(3):220–252, 1977.

[36] C. S. Peskin. The immersed boundary method. *Acta numerica*, 11:479–517, 2002.

[37] F. Picano, W.-P. Breugem, and L. Brandt. Turbulent channel flow of dense sus-

pensions of neutrally buoyant spheres. *Journal of Fluid Mechanics*, 764:463–487, 2015.

[38] P. Ploumhans and G. Winckelmans. Vortex methods for high-resolution simulations of viscous flow past bluff bodies of general geometry. *Journal of Computational Physics*, 165(2):354–406, 2000. ISSN 0021-9991. doi: https://doi.org/10.1006/jcph.2000.6614. URL `https://www.sciencedirect.com/science/article/pii/S0021999100966142`.

[39] A. Quarteroni. *Numerical models for differential problems*, volume 2. Springer, 2009.

[40] P. Saksono, W. Dettmer, and D. Perić. An adaptive remeshing strategy for flows with moving boundaries and fluid–structure interaction. *International Journal for Numerical Methods in Engineering*, 71(9):1009–1050, 2007.

[41] S. Schwarz, T. Kempe, and J. Fröhlich. An immersed boundary method for the simulation of bubbles with varying shape. *Journal of Computational Physics*, 315: 124–149, 2016.

[42] F. Sotiropoulos, T. B. Le, and A. Gilmanov. Fluid mechanics of heart valves and their replacements. *Annual Review of Fluid Mechanics*, 48:259–283, 2016.

[43] F. C. Thames, J. F. Thompson, C. Wayne Mastin, and R. L. Walker. Numerical solutions for viscous and potential flow about arbitrary two-dimensional bodies using body-fitted coordinate systems. *Journal of Computational Physics*, 24(3):245–273, 1977. ISSN 0021-9991. doi: https://doi.org/10.1016/0021-9991(77)90037-7. URL `https://www.sciencedirect.com/science/article/pii/0021999177900377`.

[44] E. D. Tytell and G. V. Lauder. The hydrodynamics of eel swimming: I. wake structure. *Journal of Experimental Biology*, 207(11):1825–1841, 2004.

[45] H. K. Versteeg and W. Malalasekera. *An introduction to computational fluid dynamics: the finite volume method*. Pearson education, 2007.

[46] J. Videler and F. Hess. Fast continuous swimming of two pelagic predators, saithe (pollachius virens) and mackerel (scomber scombrus): a kinematic analysis. *Journal of experimental biology*, 109(1):209–228, 1984.

[47] S. Ye, Y. Lin, L. Xu, and J. Wu. Improving initial guess for the iterative solution of linear equation systems in incompressible flow. *Mathematics*, 8(1):119, 2020.

[48] Y.-n. Zhang, X.-y. Wang, Y.-n. Zhang, and C. Liu. Comparisons and analyses of vor-

tex identification between omega method and q criterion. *Journal of Hydrodynamics*, 31:224–230, 2019.

A | 2D structure generation

To generate the STL (Standard Tessellation Language) representation of the structure, a MATLAB code was developed to create a script compatible with the GMSH software, which subsequently generates the tessellation. The transformation into a two-dimensional structure, achieved by removing the lateral components of the fish structure, is performed using the ParaView software in a later step.

Considering the geometrical description of the 2D swimmer steady-shape outlined in section 4.1, the MATLAB code receives as an input the parameter n_θ, which determines the number of discrete points used to parameterize the reference circumference. As for the shape parameters, the user can specify the horizontal coordinate of the center of the circumference, denoted as η_C, and the trailing edge angle, denoted as α.

Therefore, the code generates the IBFISH.geo file, which serves as input for GMSH to create the IBFISH.stl file. Moreover, for verification, the MATLAB script displays Figure 4.1 using the specified parameter set. The MATLAB code is provided as follows:

```
% Fish_Generator (Reference state)
clear;
close all;

% Circle Parametrization:
ntheta = 100;
theta = linspace(0, 2*pi, ntheta);

% Fish Parameters:
eta_c = -0.04;
alpha_degree = 5;

alpha = alpha_degree*(pi/180);
n = 2 - alpha/pi;

```

```
for j = 1:length(theta) - 1
    zeta(j) = (1 + abs(eta_c))*exp(1i*theta(j)) + eta_c;

    num = (1 + 1/zeta(j))^n + (1 - 1/zeta(j))^n;
    den = (1 + 1/zeta(j))^n - (1 - 1/zeta(j))^n;
    z = n*num/den;

    x(j) = real(z);
    y(j) = imag(z);
end

% Rescale:
rescale = abs(max(x) - min(x));
for i = 1:length(x)
    x(i) = x(i)/rescale;
    y(i) = y(i)/rescale;
end

% Translation:
translation = abs(min(x));
for i = 1:length(x)
    x(i) = x(i) + translation;
end

% Visual Plot of the result
figure(1)
subplot(1,2,1)
plot(real(zeta), imag(zeta),'r');
grid on;
title("Parametrized Circunference")

subplot(1,2,2)
plot(x,y,'black');
grid on;
xlim([0 1]);
ylim([-0.5 0.5]);
title("Fish Shape")
```

```
53
54 % Verification of the trailing edge angle:
55 alpha_airfoil = 2*atan(abs(y(end) - y(end - 1))/abs(x(end) -
      x(end - 1)))*(180/pi);
56
57 %% Save in Gmesh
58 clearvars -except x y
59
60 z = zeros(1,length(x)) - 0.5;
61 characteristicLength = z + 1.0;
62
63 points = [x',y',z', characteristicLength'];
64
65 lines = [];
66 for i = 0:length(x) - 2
67     lines = [lines; [i , i+1]];
68 end
69 lines = [lines; [lines(end), 0]];
70
71 % Writing in the output file.
72
73 % Open the output file for writing
74 fid = fopen('IBFISH.geo', 'w');
75
76 fprintf(fid, '// Generate the stl with the following command\
      n');
77 fprintf(fid, '//gmsh -2 -format stl IBFISH.geo\n');
78 fprintf(fid, '\n');
79 fprintf(fid, '// Shape points\n');
80
81 % Write the shape points
82 for i = 0:size(points, 1)-1
83     fprintf(fid, 'Point(%d) = {%f, %f, %f, %f};\n', i, points
         (i+1,:));
84 end
85
86 fprintf(fid, '\n');
```

```
87
88 % Write the lines
89 for i = 10:size(lines, 1)+9
90     fprintf(fid, 'Line(%d) = {%d, %d};\n', i, lines(i-9,:));
91 end
92
93 fprintf(fid, '\n');
94
95 fprintf(fid, 'Curve Loop(100) = {10:%d};\n', length(lines) +
       9);
96
97 fprintf(fid, '\n');
98
99 fprintf(fid, 'Plane Surface(1000) = {100};\n');
100
101 fprintf(fid, '\n');
102
103 fprintf(fid, 'extr[] = Extrude {0, 0, 1.0} {\n');
104 fprintf(fid, '  Surface {1000};\n');
105 fprintf(fid, '  Layers  {1};\n');
106 fprintf(fid, '  Recombine;\n');
107 fprintf(fid, '};\n');
108
109 fprintf(fid, '\n');
110
111 fprintf(fid, 'Volume("internal") = extr[1];\n');
112 % Close the file
113 fclose(fid);
```

List of Figures

List of Tables

Acknowledgements

I would like to express my gratitude to Professor Nicola Parolini, who played a vital role in shaping the content and direction of this thesis. He offered valuable insights and provided me with all the support I needed throughout my development as a student. Furthermore, I would like to express my appreciation to Doctor Giorgio Negrini. His help was indispensable throughout the entire journey using OpenFOAM. He provided me with essential codes to improve the results, corrected misconceptions, and helped me whenever I needed it. I extend my thanks to Professor Bruno Carmo, who gave me valuable feedback during the project and constructive criticism, which significantly improved the quality of this thesis.

I would also like to express my gratitude to my family, Hélio Murakami and Roseli Murakami, who, even on the other side of the world, have offered me encouragement and support, making this exchange program possible. Finally, I want to acknowledge all my friends who have been by my side throughout this incredible journey, with special mention to Carolina Iplinsky, Felipe Pascutti, Gustavo Torrico, Julia Rangel, Juliana Maines, and Pedro Gianjoppe.