

UNIVERSITÀ DEGLI STUDI DELLA TUSCIA

Dipartimento di Economia, Ingegneria, Società e
Impresa

Corso di Dottorato di Ricerca in
Engineering for Energy and Environment
Curriculum Energy and Engineering Systems
XXXIV Ciclo

Tesi di Dottorato

**Advanced methods for fluid-structure
interaction in case of deformable bodies**

Settore scientifico disciplinare Ing-Ind/14

Tesi di dottorato di

Chiara Stefanini

Coordinatore del corso

Prof. Andrea L. Facci

Tutore

Prof. Pierluigi Fanelli

Co-tutore

Prof. Andrea L. Facci

A.A. 2020/2021

*Alla mia Famiglia.
A chi mi ha accompagnato lungo il percorso.*

Contents

Abstract	2
Introduction	5
1 Numerical general approach for FSI problems	11
1.0.1 Mathematical definition of the derivation operation in different domains	13
1.0.2 Fluid domain conservation equations	17
Mass conservation	17
Momentum conservation	18
Energy conservation	18
1.0.3 Solid domain conservation equation	21
1.0.4 Resolution method of the equations system	25
2 Fluid-dynamics solution implementation	29
2.1 The Finite Volume Method	29
2.1.1 Discretization of the Solution Domain	31
2.1.2 Discretization of the Transport Equation	33
Space Integration of ϕ	34
Convective terms	35
Diffusion terms	37
Time integration of ϕ	40
Discretised transport equation	41
2.1.3 Discretization of Navier-Stokes	42
Boundary Conditions	46
3 Solid-mechanics solution implementation	49
3.1 The Finite Volume Method	49
3.1.1 Discretization of Momentum Conservation Law	49

	Solution procedure	51
	Boundary Conditions	53
3.2	The Beam Deflection Method	54
3.2.1	Solving algorithm	55
3.3	The Modal Superposition Method	62
3.3.1	Numerical algorithm	67
4	<i>modSFsiFoam</i> solver code	69
4.0.1	Object-orientation and C++	71
4.0.2	Solver codes	71
4.0.3	OpenFOAM FSI case structure	72
4.0.4	Solver structure	73
5	Applications	79
5.1	Fluid-structure interaction problem of an inverted flag in a wind tunnel	82
5.2	Cylinder-lamina system in a laminar incompressible chan- nel flow	89
	Conclusions	95
	List of Figures	100
	List of Tables	102
	Bibliography	103

Abstract

Fluid-structure interaction (FSI) problems regard multiphysics systems where structures interact with fluid flows. The structure is subjected to flow-related loads; as a consequence, structural displacements and oscillations influence the fluid motion field. Many engineering fields require the resolution of FSI problems, yet a comprehensive study of such problems remains a challenge due to their strong non-linearity and multidisciplinary nature. To investigate the physics involved in this complex interaction, numerical simulations may be employed.

The research activity here proposed aimed at giving an improvement to the scientific knowledge about fluid-structure interaction through the development of *modSFsiFoam* (*MODalSuperpositionFsiFoam*) solver in OpenFOAM, an opensource software developed for Linux distributions and released under a GPL license with the C++ source code.

The partitioned approach has allowed treating the multi-physics phenomenon as two different computational fields, with their respective mesh discretization and algorithms. It has enabled the application of two different methods for fluid and solid solutions. In particular, the solver code is based on a theoretical approach, the modal superposition principle, for the structural solution; this method provides a certain result for the linear structural field. The FVM (Finite Volume Method), by far the most widely used method for fluid-dynamics and gas-dynamics problems, has been used instead to solve the Navier-Stokes equations for an incompressible laminar flow of Newtonian fluid. The interfacial conditions have been used to pass information between the domains through a coupling algorithm.

Satisfactory results were obtained from the application of the solver to two FSI problems known in the scientific literature.

Keywords: Fluid-structure interaction problems, OpenFOAM, *modSF-siFoam*, FVM, modal superposition principle.

Introduction

In fluid-structure interaction (FSI) problems, one or more solid structures interact with an internal or surrounding fluid flow. FSI problems play prominent roles in many scientific and engineering fields, yet a comprehensive study of such problems remains a challenge due to their strong non-linearity and multidisciplinary nature. There are many engineering application fields for Fluid-Structure Interaction. The main fields of interest regards marine, aeronautical, aerospace, biomedical and energy engineering. Considering maritime FSI systems, structures should be divided into rigid structures, which are widely described in scientific literature, and deformable structures, which are still under-researched. All structures in the field of offshore and ocean engineering may deform under the action of water and air. However, the effects of deformation are sometimes negligible, but in some cases, their evaluation is essential for a proper analysis of the problem. This is not the only differentiation in FSI maritime systems; different characteristics could lead to a set of classifications of systems themselves, based both on experimental and numerical points of view. FSI problems can be classified into a group that involves the free surface and another group where this influence is neglected. Furthermore, marine structures can be classified as moving or fixed structures. Fixed structures do not undergo rigid motions, while moving structures do the opposite, though both may have structural deformation. In addition, is possible to determine linear or nonlinear interaction. There are many factors influencing this property. Depending on the nature of fluid and structural dynamics, the effects of the factors can be superimposed. In this case, the FSI problem is linear. Comparatively, in non-linear FSI problems, the effect of one factor depends on others. The solutions are not equal to the superimposition of the decoupled solutions [41]. Over the years, a wide range of experimental campaigns allowed the investigation of all the above-mentioned aspects, which covered different kinds of naval systems

subjected to the interaction problem. These tests ranged from simple experiments on the structural response of ships under varying sailing conditions, to investigations on the behaviour of simple geometry specimens in case of interaction with water, in order to simulate specific components of the structure. Concerning aircrafts and spacecrafts design, most relevant FSI challenges are the optimisation of aerodynamic properties of the structure and the improvement of bodies performances during water impacts. In particular, water impacts represent an important research theme in aerospace field. Until the development of NASA space program, most of the research works focused on this problem were based on Von Karman's theory, in particular on Von Karman's principle of added mass and conservation of momentum to determine the impact forces acting upon rigid two-dimensional bodies undergoing vertical impacts. In the early years, the experimentation regarded only the rigid components of spacecrafts; however, during the Apollo programme, elastic materials were used and the failure of the spacecraft was investigated experimentally. With recent advances of computer technology, numerical simulations allowed a more complete analysis of spacecraft components [58][53]. In the aerospace field, the fluid-structure interaction is also simulated in air to study the effects of buckling of the panels on the wings as a consequence of the bending state to which they are subjected [10][42][38]. Bioengineering mostly involves the study of the response of biological systems to mechanical loading or stimulus, for example cardiovascular diseases where hemodynamic forces, i.e. wall shear stress, are known to play a key role in Atherosclerosis. Several studies have become key to understand the role of hemodynamic forces in cardiovascular disease, through a deep focus on the interaction between flowing blood and deforming vascular wall, which is a typical example of FSI [23][34][21][40]. Remarkable are also FSI applications in energetic field. In this regard, several studies are carried out on wind turbine, turbomachines, reactors and systems able to capture and to stock ambient energy in order to convert it into a useful form, such as electricity (energy harvesting) [59][8][35][70]. A wide range of laboratory tests allow a deep analysis of FSI systems in the above-mentioned engineering fields; the most common classification provides for a division of tests themselves into continuous flow tests, which are based on the velocity and the flow-rate of the fluid, and non-continuous flow tests. These allow to simulate different conditions of FSI, such as in water impact, dynamic and static fluid tests; dynamic fluid tests allow a proper simulation of slamming and

sloshing phenomena. On the other hand, water impacts on static fluid can simulate the slamming phenomena in particular conditions, such as bodies ditching. All these tests can be carried out through a single experimental setup known as water tunnel. There are also specific tests which require particular experimental setup, such as towing tank and ice tank [46][64][25][56][18]. During these tests, data acquisition can be focused on both fluid and structure behaviour, but also on one of two elements of the system. The acquisition of data on both fluid and structure requires the use of sensors suited for direct measurements and of non-invasive techniques, such as image analysis techniques (PIV and PIBV) [63][51][17][50].

For most FSI problems, analytical solutions to the model equations are impossible to obtain, whereas laboratory experiments are limited in scope; thus to investigate the fundamental physics involved in the complex interaction between fluids and solids, numerical simulations may be employed. In the past three decades, the Finite Element Method (FEM) has enjoyed an undivided popularity as the method for solid body stress analysis. On the other hand, the Finite Volume Method (FVM) has been established as a very efficient way of solving fluid flow problems. The fact that solid body mechanics and fluid mechanics share the same governing equations, and differ in constitutive relations only, has meant that, in the last years, attempts to apply FEM to thermofluidodynamic analyses and FVM to structural analyses have been made. Although they are inherently similar, the two methods have their advantages and disadvantages, which make them better suited for certain classes of problems. However, the situation is not as clear-cut as it might seem. In fact, we cannot tell in advance whether the block solution associated with the FEM gives an a priori advantage over the segregated FVM solver even for a simple linear elastic problem. This is a question of the trade-off between the high expense of the direct solver for a large matrix and the cheaper iterative solvers with the necessary iteration over the explicit cross-component coupling. Although the FV discretization may be thought to be inferior to the FEM in linear elasticity, it is still compelling to examine its qualities. The reason for this may be the fact that the FVM is inherently good at treating complicated, coupled and non-linear differential equations, widely present in fluid flows. By extension, as the mathematical model becomes more complex, as in the case of FSI problem, the FVM should become a more interesting alternative to the FEM [33].

The numerical procedures to solve these FSI problems may be broadly classified into two approaches: the monolithic approach and the partitioned approach. It is understood that researchers from different fields may view the distinction between the monolithic and partitioned approaches differently. The monolithic approach treats the fluid and structure dynamics in the same mathematical framework to form a single system of equations for the entire problem, which is solved simultaneously by a unified algorithm. The interfacial conditions are implicit in the solution procedure. This approach can potentially achieve better accuracy for a multidisciplinary problem, but it may require substantially more resources and expertise to develop and maintain such a specialized code. In contrast, the partitioned approach treats the fluid and the structure as two computational fields, which can be solved separately with their respective mesh discretization and numerical algorithm. The interfacial conditions are used explicitly to communicate information between the fluid and structure solutions. A motivation of the latter approach is to integrate available disciplinary (i.e., fluid and structural) algorithms and reduce the code development time by taking advantage of the legacy codes or numerical algorithms that have been validated and used for solving many complicated fluid or structural problems. As a result, a successful partitioned method can solve a FSI problem with sophisticated fluid and structural physics. The challenge of this approach is, however, to coordinate the disciplinary algorithms to achieve accurate and efficient fluid-structure interaction solution with minimal code modification. In the partitioned approaches, the structural solution can be obtained through FEM or FVM, which is the same method used to pursue the fluid solution. In addition to this, it is possible to consider other analyses to obtain structural displacement and strain fields, such as modal analyses [26][27].

In this scenario, the activity carried out during the Ph.D., is placed. It has allowed to develop a partitioned numerical code suitable for solving FSI problems in the case of deformable bodies. In particular, by an in-depth theoretical study of the most effective methods for FSI simulations, to identify the main criticalities of each FSI system and, consequently, to improve the quality of the design based on the simulation results, through a partitioned approach, *ModSfsiFoam* (*MODalSuperpositionFsiFoam*) solver has been developed. It allows the application of two different methods for fluid and solid solutions. In particular, it is based on a theoretical approach, the modal superposition principle, for the structural

solution; this method provides a certain result for the linear structural field. The FVM (*Finite Volume Method*), by far the most widely used method for fluid-dynamics and gas-dynamics problems, has been used instead to solve the Navier-Stokes equations for an incompressible laminar flow of Newtonian fluid. The interfacial conditions have been used to pass information between the domains through a coupling algorithm. Implemented in OpenFOAM, a development environment consisting of libraries and applications for Linux distributions, written in the C++ source code, *ModSFsifoam* solver has been applied with satisfactory results to typical fluid-structure interaction cases found in literature and reported in the following chapters, the results of which are known for both fluid-dynamics and structural fields.

With the aim of validating these results also through experimental campaigns, an experimental setup has been designed and it is available in the Tuscia University laboratories.

Chapter 1

Numerical general approach for FSI problems

The numerical simulation of multidimensional problems in fluid dynamics and nonlinear solid mechanics often requires to deal with strong distortions of the continuum. The choice of an appropriate kinematical description of the continuum itself is essential to face this kind of problems. Such a choice determines the relationship between the deforming continuum and the finite grid or mesh of computing zones. This conditions the ability of the numerical method in dealing with large distortions and providing an accurate resolution of material interfaces and mobile boundaries.

The algorithms of continuum mechanics usually make use of two classical descriptions of motion:

- *Lagrangian*, in which each individual node of the computational mesh follows the associated material particle during motion are mainly used in structural mechanics. The Lagrangian description allows an easy tracking of free surfaces and interfaces between different materials. It also facilitates the treatment of materials with history-dependent constitutive relations. Its weakness is its inability to follow large distortions of the computational domain without recourse to frequent remeshing operations.
- *Eulerian*, which are widely used in fluid dynamics. Here the computational mesh is fixed and the continuum moves with respect to

the grid. In the Eulerian description, large distortions in the continuum motion can be handled with relative ease, but generally at the expense of precise interface definition and the resolution of flow details.

In order to combine the advantages of the above classical kinematical descriptions, and to minimize their respective drawbacks, the *Arbitrary Lagrangian-Eulerian* (ALE) description has been developed [60].

In the ALE description, the nodes of the computational mesh may be moved with the continuum in normal Lagrangian approach, or may be held fixed in Eulerian manner, or may be moved in some arbitrarily specified way to give a continuous rezoning capability. Because of this freedom in moving the computational mesh offered by the ALE description, available in OpenFOAM, it is allowed to deal with greater distortions of the continuum than in a purely Lagrangian method, with more resolution than in a purely Eulerian approach [15][11][14].

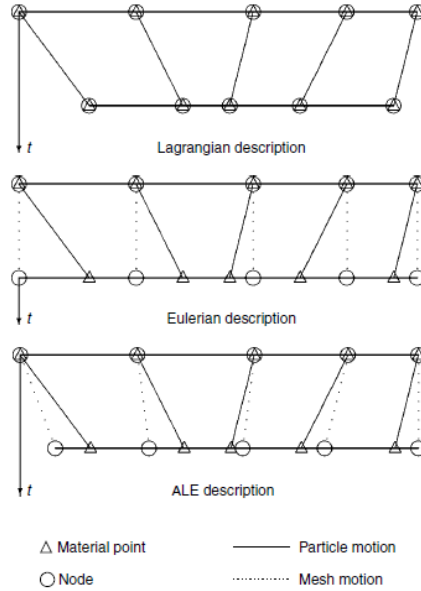


Figure 1.1: One-dimensional example of Lagrangian, Eulerian and ALE mesh and particle motion [15].

1.0.1 Mathematical definition of the derivation operation in different domains

For a comprehensive analysis of the above-mentioned themes, it is necessary to define three different domains:

- *Material domain* $R_{\mathbf{X}}$, which is constituted of the particles the analysed body is made of, and is described by material coordinates $\mathbf{X}$.
- *Spatial domain* $R_{\mathbf{x}}$, which is made of the spatial point chosen to describe the body motion, and is described by spatial coordinates $\mathbf{x}$.
- *Reference domain* R_{χ} , which is constituted of the grid points, and is described by reference coordinates χ .

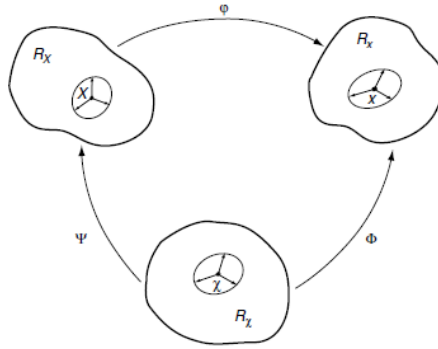


Figure 1.2: The motion of the ALE computational mesh is independent of the material motion [15].

The above-described domains are related to each other by the $\phi(\chi, t)$, $\varphi(X, t)$ and $\psi(\chi, t)$ functions, which allow to express displacement and velocity in all the mentioned domains, by referring to the time coordinate t . Considering that $\varphi = \Phi \circ \psi^{-1}$, by deriving all the terms with respect to their spatial and temporal coordinates.

$$\frac{\partial \varphi}{\partial (\mathbf{X}, t)} (\mathbf{X}, t) = \frac{\partial \phi}{\partial (\chi, t)} (\psi^{-1} (\mathbf{X}, t)) \frac{\partial \psi^{-1}}{\partial (\mathbf{X}, t)} (\mathbf{X}, t) = \frac{\partial \phi}{\partial (\chi, t)} (\chi, t) \frac{\partial \psi^{-1}}{\partial (\mathbf{X}, t)} (\mathbf{X}, t) \quad (1.1)$$

By using the matrix formulation, it can be obtained:

$$\begin{bmatrix} \frac{\partial \mathbf{x}}{\partial \mathbf{X}} & \mathbf{v} \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathbf{x}}{\partial \boldsymbol{\chi}} & \hat{\mathbf{v}} \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} \frac{\partial \boldsymbol{\chi}}{\partial \mathbf{X}} & \mathbf{w} \\ 0^T & 1 \end{bmatrix} \quad (1.2)$$

where:

- the material velocity $\mathbf{v}(\mathbf{X}, t) = \frac{\partial \mathbf{x}}{\partial t}|_{\mathbf{X}}$
- the mesh velocity $\hat{\mathbf{v}}(\boldsymbol{\chi}, t) = \frac{\partial \mathbf{x}}{\partial t}|_{\boldsymbol{\chi}}$
- the particle velocity $\mathbf{w}(\boldsymbol{\chi}, t) = \frac{\partial \mathbf{x}}{\partial t}|_{\mathbf{x}}$

which yields, after block multiplication

$$\mathbf{v} = \hat{\mathbf{v}} + \mathbf{w} \frac{\partial \mathbf{x}}{\partial \boldsymbol{\chi}} \quad (1.3)$$

From the equation 1.3, it can be seen how the velocity of each particle of the material domain equals the mesh motion velocity with a term related to the relative motion with respect to the reference volume added. The following relation allows the definition of the convective velocity:

$$\mathbf{c} = \mathbf{v} - \hat{\mathbf{v}} = \mathbf{w} \frac{\partial \mathbf{x}}{\partial \boldsymbol{\chi}} \quad (1.4)$$

The convective velocity $\mathbf{c}$ should not be confused with $\mathbf{w}$ (the mesh motion velocity), considering that $\mathbf{w}$ is the particle velocity as seen from the referential domain $R_{\boldsymbol{\chi}}$, whereas $\mathbf{c}$ is the particle velocity relative to the mesh as seen from the spatial domain R_x . In fact, equation 1.4 implies that $\mathbf{c} = \mathbf{w}$ if and only if $\frac{\partial \mathbf{x}}{\partial \boldsymbol{\chi}} = [I]$ (where $[I]$ is the identity tensor), that is, when the mesh motion is purely translational, without rotations or deformations. From the equation 1.4 it is possible to exploit the lagrangian approach, in which $\mathbf{w}=0$ because there is not a relative motion between the material and spatial domain, or to exploit the eulerian approach, in which the grid is static and $\hat{\mathbf{v}}=0$. In order to relate the time derivative in the material, spatial, and referential domains, let a scalar physical quantity be described by $f(\mathbf{x}, t)$, $f^*(\boldsymbol{\chi}, t)$, and $f^{**}(\mathbf{X}, t)$ in the spatial, referential and material domains respectively. Considering that the physical variable value must

be the same in each domain, it can be stated that:

$$f^{**}(\mathbf{X}, t) = f(\mathbf{x}, t) = f(\varphi(\mathbf{X}, t)) = f \circ \varphi \quad (1.5)$$

The gradient of this expression can be easily computed as

$$\left(\frac{\partial f^{**}}{\partial \mathbf{X}} \frac{\partial f^{**}}{\partial t} \right) = \left(\frac{\partial f}{\partial \mathbf{x}} \frac{\partial f}{\partial t} \right) \begin{bmatrix} \frac{\partial \mathbf{x}}{\partial \mathbf{X}} & \mathbf{v} \\ 0^T & 1 \end{bmatrix} \quad (1.6)$$

which renders, after block multiplication, the expression $\frac{df}{dt} = \frac{\partial f}{\partial t} + \mathbf{v} \frac{\partial f}{\partial \mathbf{x}}$, which is the well-known equation that relates the material and the spatial time derivatives. Dropping the stars to ease the notation, this relation is finally cast as the eulerian total derivative:

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \mathbf{v} * \nabla f \quad (1.7)$$

which can be interpreted in this way: the variation of a physical quantity for a given physical variable is the local variation in time plus a convective term taking into account the relative motion between the material and spatial domains.

The transformation from the referential description $f^*(\boldsymbol{\chi}, t)$ of the scalar physical quantity to the material description $f^{**}(\mathbf{X}, t)$ can be written as

$$f^{**}(\mathbf{X}, t) = f^*(\boldsymbol{\chi}, t) = f(\psi^{-1}(\mathbf{X}, t)) = f \circ \psi^{-1} \quad (1.8)$$

$$\left(\frac{\partial f^{**}}{\partial \mathbf{X}} \frac{\partial f^{**}}{\partial t} \right) = \left(\frac{\partial f^*}{\partial \boldsymbol{\chi}} \frac{\partial f^*}{\partial t} \right) \begin{bmatrix} \frac{\partial \boldsymbol{\chi}}{\partial \mathbf{X}} & \mathbf{w} \\ 0^T & 1 \end{bmatrix} \quad (1.9)$$

$$\frac{\partial f^{**}}{\partial t} = \frac{\partial f^*}{\partial t} + \mathbf{w} \frac{\partial f^*}{\partial \boldsymbol{\chi}} \quad (1.10)$$

Note that the last equation requires the evaluation of the gradient of the considered quantity in the referential domain for the definition of the time derivative in the material domain. This can be done, but in computational mechanics it is usually easier to work in the spatial (or material) domain.

Thus, using the definition of $\mathbf{w}$ given in equation 1.4, the previous equation may be rearranged into

$$\frac{\partial f^{**}}{\partial t} = \frac{\partial f^*}{\partial t} + \mathbf{c} \frac{\partial f^*}{\partial \mathbf{x}} \quad (1.11)$$

The fundamental ALE relation between material time derivatives, referential time derivatives, and spatial gradient is finally cast as (stars dropped)

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \mathbf{c} * \nabla f \quad (1.12)$$

which shows that the time derivative of the physical quantity f for a given particle χ , that is its material derivative, is its local derivative (with the reference coordinate χ held fixed) plus a convective term taking into account the relative velocity $\mathbf{c}$ between the material and the reference system.

In order to establish the integral form of the basic conservation laws for mass, momentum, and energy, it is also necessary to consider the rate of change of integrals of scalar and vector functions over a moving volume occupied by fluid. With this aim in view, let V_t be the material volume, i.e. a volume that permanently contains the same particles of the continuum under consideration, is continuum and bounded by a smooth closed surface S_t . On the other hand, V_c is the control volume, which is fixed in the space bounded by S_c and coincides with V_t at the first instant, t . By referring to the Reynolds transport theorem, the following expressions can be stated:

$$\begin{aligned} \frac{d}{dt} \int_{V_t} f(\mathbf{x}, t) dV &= \int_{V_c=V_t} \frac{\partial f(\mathbf{x}, t)}{\partial t} dV + \int_{S_c=S_t} f(\mathbf{x}, t) \mathbf{v} \cdot \mathbf{n} dS = \\ &= \int_{V_c} \left(\frac{\partial f(\mathbf{x}, t)}{\partial t} + \nabla \cdot (f \mathbf{v}) \right) dV \end{aligned} \quad (1.13)$$

in which the last term has been obtained through the Gauss theorem, which allows the transposition of a surface integral in a volume integral. In order to switch from the classic formulation to the ALE formulation, it is sufficient to replace the $\mathbf{v}$ term of equation 1.13 with the convective velocity $\mathbf{c}$.

1.0.2 Fluid domain conservation equations

It is now possible to write the conservation equations for mass, momentum and energy from the corresponding well-known Eulerian forms, which will be modified in order to take into account the possibility of a mesh motion.

Mass conservation

Considering a generic volume V_t in a generic timestep, there is a certain mass quantity contained in the volume, which can be expressed as:

$$m = \int_{V_t} \rho dV \quad (1.14)$$

The conservation equation states that the mass variation in V_t , which is related to local variations of density in time, can be offset by the total mass flux through the S_t surface:

$$\int_{V_t} \frac{\partial \rho}{\partial t} dV + \int_{S_t} \rho \mathbf{v} \cdot \mathbf{n} dS = 0 \quad (1.15)$$

which can be modified, using the Gauss Theorem, as follows:

$$\int_{V_t} \left(\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) \right) dV = 0 \quad (1.16)$$

which is the integral expression of the mass conservation. Considering that the above-stated equation must be suited for each V_t volume, the equation 1.16 must be recast as:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \quad (1.17)$$

which is the differential form. In order to switch to the ALE formulation, it is sufficient to replace the $\mathbf{v}$ term with the convective velocity $\mathbf{c}$, with the aim of taking into account any possible grid motion.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{c}) = 0 \quad (1.18)$$

Momentum conservation

The momentum conservation is closely related to the second law of motion, which states that:

$$\mathbf{F} = m\mathbf{a} = m \frac{d\mathbf{v}}{dt} = \frac{d(m\mathbf{v})}{dt} = \frac{d\mathbf{Q}}{dt} \quad (1.19)$$

This equations concerns both the solid and the fluid domain, for which can be written as:

$$\int_{V_t} \frac{\partial(\rho\mathbf{v})}{\partial t} dV + \int_{S_t} \rho\mathbf{v}(\mathbf{v} \cdot \mathbf{n}) dS = \int_{V_t} \mathbf{f} dV + \int_{S_t} (\boldsymbol{\tau} - \mathbf{p} \cdot \mathbf{n}) dS \quad (1.20)$$

The first term of the equation is the addition of the local variation of momentum in the V_t volume and the total flux through the S_t surface of the domain; on the other hand, the second term is made of the volume forces $\mathbf{f}$ (applied to the fluid volume) and the addition of the surface forces applied on the S_t surface, which can be both viscous and hydrostatic. By considering that the equation must be valid for each chosen volume, the application of the Gauss theorem allows to obtain the differential expression of the momentum conservation:

$$\frac{\partial(\rho\mathbf{v})}{\partial t} + \nabla \cdot (\rho\mathbf{v}\mathbf{v}) = \rho\mathbf{g} + \nabla \left(p + \frac{2}{3}\mu\nabla \cdot \mathbf{v} \right) + \nabla \cdot (\mu(\nabla\mathbf{v} + \nabla\mathbf{v}^T)) \quad (1.21)$$

In order to switch to the ALE formulation, it is necessary to modify the convective term of the equation, with the aim of taking into account any possible grid motion.

$$\frac{\partial(\varphi\mathbf{v})}{\partial t} + \nabla \cdot (\rho\mathbf{c}\mathbf{v}) = \rho\mathbf{g} + \nabla \left(p + \frac{2}{3}\mu\nabla \cdot \mathbf{v} \right) + \nabla \cdot (\mu(\nabla\mathbf{v} + \nabla\mathbf{v}^T)) \quad (1.22)$$

Energy conservation

The enery conservation principle states that the total energy variation in fluids equals the variation of mechanical and thermal power in the analysed V_t volume. Let h_t be the total enthalpy ($h_t = e_0 + \frac{p}{\rho} + \frac{1}{2}v^2$), the integral expression of the conservation itself can be written as:

$$\int_{V_t} \frac{\partial(\rho h_T)}{\partial t} dV + \int_{S_t} \rho h_T(\mathbf{v} \cdot \mathbf{n}) dS = \int_{V_t} \frac{\partial p}{\partial t} dV + \int_{V_t} (\mathbf{f} \cdot \mathbf{v}) dV +$$

$$+ \int_{S_t} ((\boldsymbol{\tau} \cdot \mathbf{n}) \cdot \mathbf{v}) dS - \int_{S_t} ((p \cdot \mathbf{n}) \cdot \mathbf{v}) dS + \int_{S_t} k (\nabla T \cdot \mathbf{n}) dS + \int_{S_t} (-q \cdot \mathbf{n}) dS \quad (1.23)$$

That can be also expressed following the eulerian approach:

$$\begin{aligned} & \frac{\partial(\rho h_T)}{\partial t} + \nabla \cdot (\rho h_T \mathbf{v}) = -\frac{\partial p}{\partial t} + \rho(\mathbf{g} \cdot \mathbf{v}) - \nabla \cdot (p \cdot \mathbf{v}) + \\ & + \nabla \cdot \left(\frac{2}{3} \mu (\nabla \cdot \mathbf{v}) \cdot \mathbf{v} \right) + \nabla \cdot (\mu (\nabla \mathbf{v} + \nabla \mathbf{v}^T) \cdot \mathbf{v}) + \nabla \cdot (k \nabla T) + \rho Q \end{aligned} \quad (1.24)$$

or the ALE approach:

$$\begin{aligned} & \frac{\partial(\varphi h_T)}{\partial t} + \nabla \cdot (\rho h_T \mathbf{c}) = -\frac{\partial p}{\partial t} + \rho(\mathbf{g} \cdot \mathbf{v}) - \nabla \cdot (p \cdot \mathbf{v}) + \\ & + \nabla \cdot \left(\frac{2}{3} \mu (\nabla \cdot \mathbf{v}) \cdot \mathbf{v} \right) + \nabla \cdot (\mu (\nabla \mathbf{v} + \nabla \mathbf{v}^T) \cdot \mathbf{v}) + \nabla \cdot (k \nabla T) + \rho Q \end{aligned} \quad (1.25)$$

The first term of the equation is made of the non-stationary variation and the enery flux through the domain surfaces; otherwise, the second part introduces the following terms:

- $-\frac{\partial p}{\partial t}$ represents the non-stationary term, which is related to the need of considering the total enthalpy as a variable.
- $\rho(\mathbf{g} \cdot \mathbf{v})$ expresses the work made by the gravity force.
- $-\nabla \cdot (p \cdot \mathbf{v})$ represents the work which is done or absorbed during the interaction between a fluid and structural elements such as turbine blades.
- $\nabla \cdot \left(\frac{2}{3} \mu (\nabla \cdot \mathbf{v}) \cdot \mathbf{v} \right) + \nabla \cdot (\mu (\nabla \mathbf{v} + \nabla \mathbf{v}^T) \cdot \mathbf{v})$ is the term related to the energy dissipated because of the viscous forces.
- $\nabla \cdot (k \nabla T)$ is the term related to the energy exchanged between the fluid and the environment beacuse of the thermal gradient.
- ρQ represents a generic energy source within the domain.

In order to give a full description of the problem, it is necessary to introduce the state equations to describe the relation between the thermodynamic variables.

$$p = p(\rho, T) \quad (1.26)$$

$$e = e(T) \quad (1.27)$$

$$h = h(T) \quad (1.28)$$

The above-expressed equations solution will allow the determination of the fluid and solid domain behaviour. The fluid-structure interface conditions will be described by the displacements congruence and the equality of the stresses:

$$u_{solid} = u_{fluid} \quad (1.29)$$

$$[\sigma_{solid}] = [\sigma_{fluid}] \quad (1.30)$$

1.0.3 Solid domain conservation equation

The equation (1.21) describes the Newton's second law for a fluid domain, where the interested variable is the velocity $\mathbf{v}$. In order to apply it to the solid domain, the above-mentioned equation has to be modified to use the displacement as variable.

In this study, the structure is assumed to be elastic and isothermal. The dynamic behaviour of such a continuum is usually described by considering only the linear momentum conservation law in Lagrangian formulation [31]:

$$\frac{D}{Dt} \int_{V_t} \rho \mathbf{v} dV = \int_{S_t} \mathbf{n} \cdot \boldsymbol{\sigma} dS + \int_{V_t} \rho \mathbf{f} dV \quad (1.31)$$

where the operator $\frac{D}{Dt}$ represents the material derivative.

The conservation law can be written with respect to the initial (undeformed) configuration as:

$$\int_{V_{t0}} \rho_0 \frac{\delta \mathbf{v}}{\delta t} dV_0 = \int_{S_{t0}} \mathbf{n}_0 \cdot (\boldsymbol{\Sigma} \cdot \mathbf{F}^T) dS_0 + \int_{V_{t0}} \rho_0 \mathbf{f} dV_0 \quad (1.32)$$

where the subscripts 0 represent the quantities related to the initial configuration, $\mathbf{v}$ is the velocity vector, $\mathbf{F}$ is the deformation gradient tensor defined as:

$$\mathbf{F} = \mathbf{I} + (\nabla \mathbf{U})^T \quad (1.33)$$

and $\boldsymbol{\Sigma}$ is the second Piola-Kirchhoff stress tensor, which is related to the Cauchy stress tensor by the following expression:

$$\boldsymbol{\sigma} = \frac{1}{\det \mathbf{F}} \mathbf{F} \cdot \boldsymbol{\Sigma} \cdot \mathbf{F}^T \quad (1.34)$$

From the integral form of the equation (1.32), where an infinitesimal values increment is considered, it is possible to obtain an incremental form, with a small but finite increment. This approach, known as updated-lagrangian method, was the first numerically implemented. In fact, the need to store only one configuration for every time step was well suited to the limited

computational resources.

The other approach, known as total-lagrangian method, was implemented in the following years since it requires a greater memory space. In fact, for every single time step, at least two solid structure's configurations need to be remembered. However, it is able to reconstruct the material behaviour.

$$\int_{V_{i0}} \rho_0 \frac{\partial \delta \mathbf{v}}{\partial t} dV_0 = \int_{S_{i0}} \mathbf{n}_0 \cdot (\delta \boldsymbol{\Sigma} \cdot \mathbf{F}^T + \boldsymbol{\Sigma} \cdot \delta \mathbf{F}^T + \delta \boldsymbol{\Sigma} \cdot \delta \mathbf{F}^T) dS_0 + \int_{V_{i0}} \rho_0 \delta \mathbf{f} dV_0 \quad (1.35)$$

At this stage, the constitutive equation for a St. Venant-Kirchhoff material is used as the relation between the second Piola-Kirchhoff stress tensor and the Green-Lagrangian strain tensor:

$$\boldsymbol{\Sigma} = 2\mu \mathbf{E} + \lambda \text{tr}(\mathbf{E}) \mathbf{I} \quad (1.36)$$

where μ and λ are the Lamè's coefficients and $\mathbf{E}$ is the Green-Lagrangian strain tensor defined as:

$$\mathbf{E} = \frac{1}{2} \left[\nabla \mathbf{U} + (\nabla \mathbf{U})^T + \nabla \mathbf{U} \cdot (\nabla \mathbf{U})^T \right] \quad (1.37)$$

Finally the last calculated configuration is considered the reference configuration.

By replacing the equation (1.37) in the equation (1.36) and passing on the incremental expression of $\boldsymbol{\Sigma}$, Piola-Kirchhoff tensor can be described through the only $\mathbf{U}$ grid-point displacement variable:

$$\delta \boldsymbol{\Sigma} = \mu \left[\nabla \delta \mathbf{U} + (\nabla \delta \mathbf{U})^T \right] + \lambda \text{tr}(\nabla \delta \mathbf{U}) \mathbf{I} + \mu \nabla \delta \mathbf{U} \cdot (\nabla \delta \mathbf{U})^T + \frac{1}{2} \lambda (\nabla \delta \mathbf{U} : \nabla \delta \mathbf{U}) \mathbf{I} \quad (1.38)$$

By replacing the equation (1.38) in the equation (1.35), the linear momentum conservation equation for an elastic solid in the updated Lagrangian description, with the displacement vector $\mathbf{u}$ as the primitive variable, is obtained:

$$\int_{V_t} \rho \frac{\partial \delta \mathbf{v}}{\partial t} dV - \int_{S_t} \mathbf{n} \cdot (2\mu + \lambda) \nabla \delta \mathbf{U} dS_0 = \int_{S_t} \mathbf{n} \cdot \mathbf{q} dS + \int_{V_t} \rho \delta \mathbf{f} dV \quad (1.39)$$

where:

$$\begin{aligned} \mathbf{q} = & \mu(\nabla \delta \mathbf{U})^{\mathbf{T}} + \lambda tr(\delta \mathbf{U}) \mathbf{I} - (\mu + \lambda) \nabla \delta \mathbf{U} + \mu \nabla \delta \mathbf{U} \cdot (\nabla \delta \mathbf{U})^{\mathbf{T}} + \\ & + \frac{1}{2} \lambda (\nabla \delta \mathbf{U} : \nabla \delta \mathbf{U}) \mathbf{I} + \boldsymbol{\Sigma} \cdot \delta \mathbf{F}^{\mathbf{T}} + \delta \boldsymbol{\Sigma} \cdot \delta \mathbf{F}^{\mathbf{T}} \end{aligned} \quad (1.40)$$

It can be noted as the first side of the equation (1.39) contains terms linearly dependent on the displacement, while $\mathbf{q}$ consists of nonlinear and coupling terms.

The specification of the problem is completed with the definition of a boundary condition, considering the pressure and the shear stresses externally applied. It allows to describe the displacement increment $\delta \mathbf{u}$ through these values and to maintain only one integration variable.

After mathematical passages, here not included but available in [31], this expression is obtained:

$$\begin{aligned} \mathbf{n}_0 \cdot \nabla \delta \mathbf{U} = & \left((\delta \mathbf{t} - \mathbf{p} \mathbf{n}) \frac{d\mathbf{S}}{d\mathbf{S}_0} - \mathbf{n}_0 \cdot \boldsymbol{\Sigma}_0 \cdot \delta \mathbf{F}_0^{\mathbf{T}} \right) \cdot \frac{(\mathbf{F}_0^{-1})^{\mathbf{T}}}{(2\mu + \lambda)} + \\ & - \mathbf{n}_0 \cdot \frac{\mu(\nabla \delta \mathbf{U})^{\mathbf{T}} + \lambda tr(\delta \mathbf{U}) \mathbf{I} - (\mu + \lambda) \nabla \delta \mathbf{U} + \mu \nabla \delta \mathbf{U} \cdot (\nabla \delta \mathbf{U})^{\mathbf{T}} + \frac{1}{2} \lambda (\nabla \delta \mathbf{U} : \nabla \delta \mathbf{U}) \mathbf{I}}{(2\mu + \lambda)} \end{aligned} \quad (1.41)$$

At this stage, all the elements needed to define the solid structure domain are available. In fact:

- By combining the expressions (1.39) and (1.41), the displacement increment of every grid-point $\delta \mathbf{u}$ and the updated displacement $\mathbf{u} =$

$\mathbf{u}_0 + \delta \mathbf{u}$ are determinable.

- Knowing the displacement increment, the Piola-Kirchhoff tensor elements variation (1.38) and overall values $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}_0 + \delta \boldsymbol{\Sigma}$ are determinable.
- Known the Piola-Kirchhoff tensor and calculated the $\mathbf{F}$ tensor, the Cauchy stress tensor for every domain point is determinable.

After the definition of the expressions which describe the fluid domain and the solid domain behaviour, the approach for the equation system solution needs to be determined.

1.0.4 Resolution method of the equations system

As explained in this thesis introduction, there are two methods for the coupling of the solid and the fluid domains:

- Monolithic approach
- Partitioned approach

In the monolithic approach, the fluid-solid domain is considered as a whole and all the above-mentioned equations are solved through an only system [28][36]. This method provides for the definition of a single computational grid for the two domains; in these conditions, the problem of the informations exchange at the domains interface is solved by removing the interface itself. The monolithic approach allows an high-precision resolution, because of the absence of a real separation between the two domains, with the consequent need for the interpolation of the interface values, which is one of the principal approximation factors of the solution. Besides, the monolithic method is energy conservative; this implies that the choice of the timestep is related to the accuracy, but it can not influence the stability of the simulation [44]. In fact, they usually allow for the use of bigger time-steps during the discretization of the problem in time [47]. For these reasons, the monolithic approach requires dedicated and high-efficient codes, with the aim of optimizing the performances and minimizing the solution time. Additionally, only one method can be used for the two fisical domains resolution.



Figure 1.3: Graphical scheme of the monolithic approach. In this approach, the fluid domain equations and the solid domain equations are solved in each timestep in a single system.

In the partitioned approach, the fluid and the solid domain are treated

separately, referring to their respective computational grids and solution algorithms. In this case, the solid and the fluid domains communicate through the conditions imposed at the interface; this said, the detection of the contact zone (which is not known beforehand) is important for the variables interpolation from the fluid to the solid interface and vice versa. The partitioned approach is not energy conservative; this implies that the system could become instable in case of large timestep [44]. In fact, under some physical and geometrical conditions, partitioned algorithms are unstable; this situation may occur when the physical domain has a slender shape or when the fluid density is close to the solid density: this is almost always the case in hemodynamics applications, where the density of the blood is quite close to the density of the walls of the vessel. This instability occurs because of the well-known added mass effect: the fluid acts similar to an added mass to the solid, thus changing its natural behavior. With a partitioned procedure, we can give rise to a variety of different algorithms according to the strategy used to impose the coupling conditions at the fluid-structure interface:

- Explicit algorithms : after time discretization, the coupling conditions are treated explicitly at every time-step. These algorithms, also known as weakly or loosely coupled algorithms [7], are successfully applied in aerodynamics applications [20][54], but some studies [9][22][39] showed that they are unstable under some physical and geometrical conditions due to the added mass effect, as we previously mentioned.
- Implicit algorithms: in these algorithms, also known as strongly coupled algorithms, the coupling conditions are treated implicitly at every time-step; see for example [67][68]. This implicit coupling represents a way to circumvent the instability problems due to the added mass effect; nevertheless, an implicit treatment of the coupling conditions leads to algorithms that are more expensive in terms of computational time.
- Semi-implicit algorithms: in these algorithms [2][3][4], the continuity of the displacement is treated explicitly whereas the other coupling conditions are treated implicitly. This alternative represents a trade-off between the computational cost of the algorithm and its stability in relation to the physical and geometrical properties of the problem.

However, the use of a partitioned method is recommended in case of complex geometry systems, because of the reduction of the computation time based on the optimization of the algorithms dedicated to both the fluid dynamics and the solid mechanics. In addition, the main goal is to be able to use two different methods for finding fluid-dynamic and structural solutions, thus using the most suitable algorithms for both physical fields. In relation to the here-described solver, this feature allowed to use the FVM for the fluid-dynamics solution and a theoretical method, the modal superposition method, for the structural solution, a method able to provide a certain solution in according to solid-mechanics theory. Moreover, this method, making use also of dynamics reduction strategies [57], allows to face complex structures with high frequency behaviour.



Figure 1.4: *Graphical scheme of the partitioned approach.* In this approach, the fluid domain equations and the solid domain equations are solved separately one or more times in each timestep and the informations about the congruence are exchanged through the two regions interface.

Chapter 2

Fluid-dynamics solution implementation

2.1 The Finite Volume Method

Many phenomena of engineering interest are governed by differential equations, that can be represented by a general equation for the variable ϕ . Finding analytical solution, when possible, could result too onerous. For this reason, the research of an approximate solution is pursued. Although this result is not exact, it provides informations that advances the understanding of the scientific phenomena [52].

A numerical solution of a differential equation consists of a set of numbers from which the distribution of the dependent variable ϕ can be constructed. For a given differential equation, the required discretization equations can be derived in many ways. In the FVM, the calculation domain is divided into a number of nonoverlapping control volumes such that there is one control volume surrounding each grid point. The differential equation is integrated over each control volume.

Piecewise profiles expressing the variation of ϕ between the grid points are used to evaluate the required integrals. The result is the discretization equation containing the values of ϕ for a group of grid points. The discretization equation obtained in this manner expresses the conservation principle for ϕ for the finite control volume, just as the differential equation expresses it for an infinitesimal control volume.

The most attractive feature of the control volume formulation is that the resulting solution would imply that the integral conservation of quantities such as mass, momentum, and energy is exactly satisfied over any group of control volumes and, of course, over the whole calculation domain. This characteristic exists for any number of grid points, not just in a limiting sense when the number of grid points becomes large. Thus, even the coarse-grid solution exhibits exact integral balances.

When the discretization equations are solved to obtain the grid point values of the dependent variable, the result can be viewed in two different ways. In the finite element method and in most weighted residual methods, the assumed variation of ϕ consisting of the grid point values and the interpolation functions, or profiles, between the grid points is taken as the approximate solution. In the finite difference method, however, only the grid point values of ϕ are considered to constitute the solution, without any explicit reference as to how ϕ varies between the grid points. In the control volume approach, it is possible to seek the solution in the form of the grid point values only. The interpolation formulas or the profile will be regarded as auxiliary relations needed to evaluate the required integrals in the formulation. Once the discretization equations are derived, the profile assumptions can be forgotten. This viewpoint permits complete freedom of choice in employing different profile assumptions for integrating different terms in the differential equation.

The finite Volume discretization is based on the integral form of the governing equation. This approach is made of three steps: the solution domain discretization, the equation discretization and the temporal discretization.

The discretization of the solution domain produces a numerical description of the computational domain, including the positions of points in which the solution is sought and the description of the boundary. The space is divided into a finite number of discrete regions, called control volumes or cells. For transient simulations, the time interval is also split into a finite number of time-steps. Equation discretization gives an appropriate transformation of terms of governing equations into algebraic expressions. In fact, the purpose of any discretization practice is to transform one or more partial differential equations into a corresponding system of algebraic equations.

2.1.1 Discretization of the Solution Domain

Discretization of the solution domain produces a computational mesh on which the governing equations are subsequently solved. It also determines the positions of points in space and time where the solution is sought. The procedure can be split into two parts: discretization of time and space. Since time is a parabolic coordinate, the solution is obtained by marching in time from the prescribed initial condition. For the discretization of time, it is therefore sufficient to prescribe the size of the time-step that will be used during the calculation.

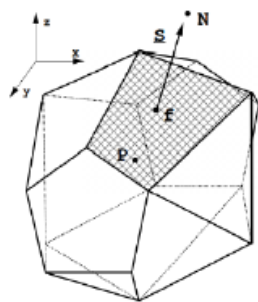
The discretization of space for the Finite Volume method used in this study requires a subdivision of the domain into Control Volumes (CV). This procedure makes a mesh of the volume, that can be classified as:

- **Structured:** this is a mesh in which all interior vertices are topologically alike. In graphtheoretic terms, a structured mesh is an induced subgraph of an infinite periodic graph such as a grid.
- **Unstructured:** this is a mesh in which vertices may have arbitrarily varying local neighborhoods.
- **Hybrid:** it is a mesh formed by a number of small structured meshes combined in an overall unstructured pattern.

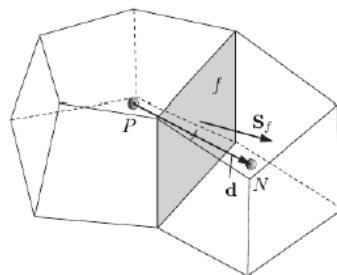
In general, structured meshes offer simplicity and easy data access, while unstructured meshes offer more convenient mesh adaptivity refinement based on an initial solution) and a better fit to complicated domains. High-quality hybrid meshes enjoy the advantages of both approaches, but hybrid meshing is not yet fully automatic.

The division between structured and unstructured meshes usually extends to the shape of the elements: two-dimensional structured meshes typically use quadrilaterals, while unstructured meshes use triangles. In three dimensions the analogous element shapes are hexahedra, meaning topological cubes, and tetrahedra. There is, however, no essential reason for structured and unstructured meshes to use different element shapes. In fact it is possible to subdivide elements in order to convert between triangles and quadrilaterals and between tetrahedra and hexahedra [5].

The topology of the control volume is not important, it is a general polyhedron. However, the positioning of the cell impose some limitations: two adjacent cells can not overlap, they are positioned in order to avoid any discontinuity and they can only share one entire face [16].



(a) Generic Control Volume



(b) Connection between two cells

Figure 2.1: Control Volume of FVM mesh.

The cell faces in the mesh can be divided into two groups internal faces, between two control volumes, and boundary faces, which coincide with the boundaries of the domain. The face area vector, $\mathbf{S}$, (figure 2.1), is constructed for each face in such a way that it points outwards from the cell with the lower label, is normal to the face and has the magnitude equal to the area of the face [29]. Points $\mathbf{P}$ and $\mathbf{N}$ are nodes of two neighboring cells. They are positioned in the center of owner cell, in order to meet the following condition:

$$\int_{V_P} (\mathbf{x} - \mathbf{x}_P) \, dV = 0 \quad (2.1)$$

Where $\mathbf{x}_P$ is the position of the computational node P, $\mathbf{x}$ is a generic locator vector and V_P is the volume of the cell.

2.1.2 Discretization of the Transport Equation

The evolution of a generic physical quantity inside a control volume can be described through the transport equation, which is stated below:

$$\frac{\partial \rho \phi}{\partial t} + \nabla \cdot (\rho \phi \mathbf{u}) + \nabla \cdot (\Gamma \nabla \phi) = 0 \quad (2.2)$$

This is a second order differential equation which can be solved using the Finite Volume Method; in particular, the above-mentioned control volume corresponds to a mesh cell typical of a FVM method. By ignoring the source term, the transport equation is made up by the following terms [37]:

- $\partial \phi / \partial t$, which is the variation over time of the variable ϕ .
- $\nabla \cdot (\phi \mathbf{u})$, which represents the specific flux of the variable ϕ through the cell walls.
- $\nabla \cdot (\Gamma \nabla \phi)$, which represents the variation of ϕ related to the diffusion, represented by the Γ term.

The solution of the transport equation and the reconstruction of the distribution of the ϕ variable are allowed by the integration of the equation itself in space and time, as stated below:

$$\int_t^{t+\Delta t} \int_{V_P} \left[\frac{\partial \rho \phi}{\partial t} + \nabla \cdot (\rho \phi u) + \nabla \cdot (\Gamma \rho \nabla \phi) \right] dV dt = 0 \quad (2.3)$$

where V_P is the cell volume and Δt is the time step.

Space Integration of ϕ

The above-mentioned $\phi(x, t)$ variable depends both on the position x and on time t . As a first step, ϕ can be considered only as a function of space. This said, $\phi(x)$ can be expanded using the Taylor's theorem, expressed in the following relation [45]:

$$F(a) = \sum_{n=0}^k \frac{f^n(a)}{n!} (x - a)^n \quad (2.4)$$

The application of this theorem makes possible to approximate a generic function f around a point a ; the k term, which enestablishes the truncation order of the function f , also represents the precision of this approximation. By applying the Taylor theorem around the computational node $\mathbf{P}$, $\phi(x)$ can be expressed as follows:

$$\phi(x) = \phi_P + (\mathbf{x} - \mathbf{x}_P) \nabla \phi + \frac{1}{2} (\mathbf{x} - \mathbf{x}_P)^2 \nabla^2 \phi + \dots + \frac{1}{k!} (\mathbf{x} - \mathbf{x}_P)^k \nabla^k \phi \quad (2.5)$$

The series above is truncated at the second order, in this analysis. Considering this, $k = 2$ and the function $\phi(x)$ becomes:

$$\phi(x) = \phi_P + (\mathbf{x} - \mathbf{x}_P) \nabla \phi \quad (2.6)$$

The integration of the equation 2.6 into the cell volume V_P , allows to find the value of $\phi(x)$ in the analysed cell. This integral can be splitted into two parts, in order to take out ϕ_P and $\nabla \phi$:

$$\int_{V_P} \phi(\mathbf{x}) dV = \int_{V_P} [\phi_P + (\mathbf{x} - \mathbf{x}_P) \nabla \phi] dV = \phi_P \int_{V_P} dV + \nabla \phi \int_{V_P} (\mathbf{x} - \mathbf{x}_P) dV \quad (2.7)$$

In eq. 2.7, the second integral of the third member coincide with the definition of centroid, which means that it equals to zero. Considering the above-mentione steps, the integral of ϕ over the cell volume can be considered equal to the product of the variable value in the computational node and of the cell volume:

$$\int_{V_P} \phi(\mathbf{x}) dV = V_P \phi_P \quad (2.8)$$

Convective terms

The Gauss-Ostrogradsky Theorem allows to turn the integral over V_P into an integral over the V_P surface. Following this theorem, the outward flux of a vector field through a closed surface is equal to the volume integral of the divergence over the region inside the surface [69]. This is expressed by the following formula:

$$\int_V \nabla \cdot \mathbf{F} dV = \oint_{\partial V} \mathbf{F} \cdot \mathbf{n} dS \quad (2.9)$$

where $\mathbf{F}$ is a generic vector field and V represents a volume, encompassed by the ∂V surface. By applying the Gauss-Ostrogradsky Theorem to the integral of the transport convective term, the following expression can be obtained:

$$\int_V \nabla \cdot (\rho \phi \mathbf{u}) dV = \oint_{\partial V} (\rho \phi \mathbf{u}) \cdot \mathbf{n} dS = \sum_{i=1}^N S_i (\rho \mathbf{u})_i \phi_i \quad (2.10)$$

Considering that the cell is a polyhedron, with a finite number of faces, in the equation the integral is replaced by a summation, in which ϕ_i represents the value of the ϕ variable in the i -surface centroid. This value is calculated by referring to the ϕ values in the compute nodes of the cells which share the i -surface. This requires an interpolation of ϕ_i , which can be realised by following two basic methods: the central differencing and the upwind schemes. The first one firstly considers a generic f -surface, shared by the $\mathbf{P}$ and $\mathbf{N}$ cells; the method consists in finding the ϕ_f value by applying a lineal interpolation between $\mathbf{P}$ and $\mathbf{N}$ as follows:

$$\phi_f = \phi_P + (\phi_P - \phi_N) \frac{d}{PN} \quad (2.11)$$

where d is the distance between the f -surface and the node $\mathbf{P}$ and PN is the distance between the above-mentioned nodes. The central differencing method guarantees a good accuracy (second order), but is unstable. Furthermore, the upwind scheme is less accurate (first-order), but shows better stability.

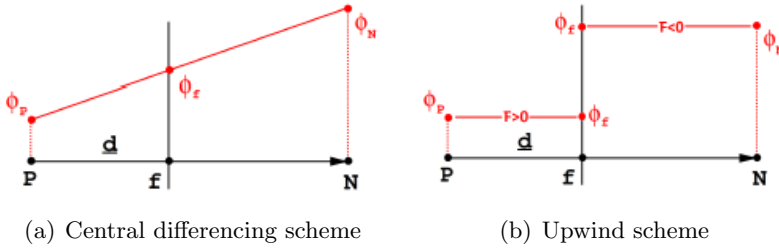


Figure 2.2: *Interpolation schemes.*

In this scheme, ϕ_f is given by the function F , which represents the convective flux direction (i.e. the product of the vector $\mathbf{S}_f$ and the velocity field on the surface $(\rho \mathbf{u})_f$). Therefore, the upwind scheme can be expressed as follows:

$$\phi_f = \begin{cases} \phi_P & \text{if } F \geq 0 \\ \phi_N & \text{if } F < 0 \end{cases} \quad (2.12)$$

Diffusion terms

The discretization methods applied for the convective terms, can also be extended to the diffusive terms of the transport equation. By applying the above-mentioned Gauss-Ostrogradsky Theorem, the diffusive integral can be written as follows:

$$\int_V \nabla \cdot (\Gamma \rho \nabla \phi) dV = \oint_{\partial V} (\Gamma \rho \nabla \phi) \cdot \mathbf{n} dS = \sum_{i=1}^N \mathbf{S}_i (\Gamma \rho \nabla \phi)_i \quad (2.13)$$

In this case, it is necessary to find the $\nabla \phi$ on cell faces. By referring to the literature, the most common method is based on the definition of the gradient of ϕ considering its value on the computational nodes of cells which share the i -face [29]:

$$\mathbf{S}_f (\nabla \phi)_f = |\mathbf{S}_f| \frac{\phi_N - \phi_P}{PN} \quad (2.14)$$

Alternatively, the central differencing scheme can be used for the gradient of ϕ . The gradient itself can be calculated in the computational node as follows:

$$\nabla \phi_P = \frac{1}{V_P} \sum_{i=1}^N \mathbf{S}_i \phi_i \quad (2.15)$$

By considering the eq. 2.11 and eq. 2.15, the gradient of ϕ on the f -face can be calculated as:

$$(\nabla\phi)_f = (\nabla\phi)_P + ((\nabla\phi)_P - (\nabla\phi)_N) \frac{d}{PN} \quad (2.16)$$

Both the methods have a second order accuracy, but are suited only for orthogonal meshes cases (i.e. the vectors $\mathbf{S}$ and PN are parallel).

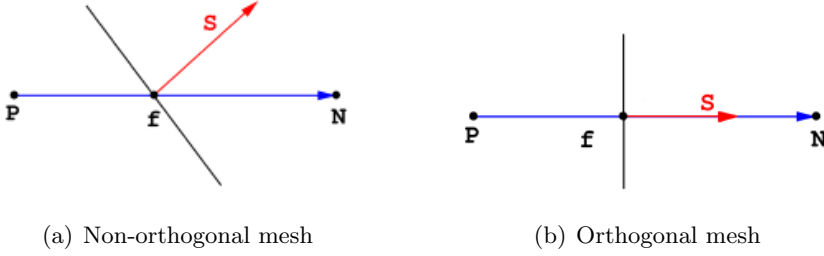


Figure 2.3: *Orthogonal and non-orthogonal mesh.*

Otherwise, in most cases meshes are not orthogonal and this requires to retouch a few terms. The product $\mathbf{S}_f(\nabla\phi)_f$ must be splitted into two parts:

$$\mathbf{S}_f(\nabla\phi_f) = \Delta(\nabla\phi_f) + \kappa(\nabla\phi_f) \quad (2.17)$$

where $\Delta(\nabla\phi_f)$ is the orthogonal contribution and $\kappa(\nabla\phi_f)$ is the non orthogonal correction. The above-mentioned terms must satisfy the following expression:

$$\mathbf{S} = \Delta + \kappa \quad (2.18)$$

Considering the availability of several methods for decomposition, in this work the most common three are described.

The first is the *Minimum correction approach* and it is conceived so as to keep the non-orthogonal correction of eq. 2.17 as small as possible, by making the Δ and the κ terms orthogonal. This said, Δ is expressed as:

$$\Delta = \frac{\mathbf{PN} \cdot \mathbf{S}}{\mathbf{PN} \cdot \mathbf{PN}} \mathbf{PN} \quad (2.19)$$

and κ is obtained from eq. 2.18:

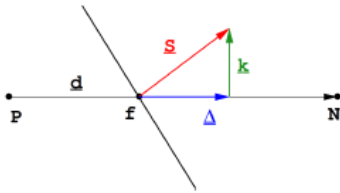
$$\kappa = \mathbf{S} - \frac{\mathbf{PN} \cdot \mathbf{S}}{\mathbf{PN} \cdot \mathbf{PN}} \mathbf{PN} \quad (2.20)$$

The second approach is the *Orthogonal correction approach*, in which the contribution from ϕ_P and ϕ_N are the same as on the orthogonal mesh. The orthogonal term is obtained as follows:

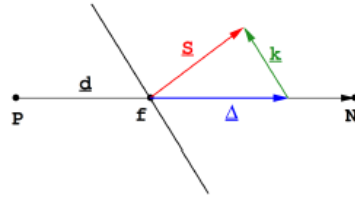
$$\Delta = \frac{\mathbf{PN}}{|\mathbf{PN}|} |\mathbf{S}| \quad (2.21)$$

The κ -term is obtained in the same manner as in the previous method. The third approach is the *Over-relaxed approach*; in this method, the importance of ϕ_P and ϕ_N is related to the non-orthogonality of the mesh. This said, The Δ -term is obtained as follows:

$$\Delta = \frac{\mathbf{PN}}{\mathbf{PN} \cdot \mathbf{S}} |\mathbf{S}|^2 \quad (2.22)$$



(a) Minimum correction approach



(b) Orthogonal correction approach

Figure 2.4: Correction methods for a non-orthogonal mesh.

Time integration of ϕ

The last term to be treated is the derivation of ϕ in time. The discretization of this term can be obtained through several methods, such as the *Explicit* and *Implicit Euler Method* (also known as *Backward* and *Forward Euler Methods*), and the *Crank-Nicholson Method*. Considering the diffusion of ϕ , a generic physical quantity in a monodimensional system, the transport equation can be expressed as follows:

$$\frac{\partial \phi}{\partial t} = D \frac{d^2 \phi}{dx^2} \quad (2.23)$$

By using the derivative definition, each term of the equation can be rewritten as:

$$\frac{\phi_P^t - \phi_P^{t-\Delta t}}{\Delta t} = \theta D \frac{\phi_{P-\Delta x}^{t-\Delta t} - 2\phi_P^{t-\Delta t} + \phi_{P+\Delta x}^{t-\Delta t}}{\Delta x^2} + (1-\theta) D \frac{\phi_{P-\Delta x}^t - 2\phi_P^t + \phi_{P+\Delta x}^t}{\Delta x^2} \quad (2.24)$$

Where the θ -value is related to the dependence of ϕ from its future value, while $(1-\theta)$ expresses the dependence from the past values. By grouping the past terms ($t - \Delta t$) and the future terms (t), the equation can be written as follows:

$$\begin{aligned} \phi_{P-\Delta x}^t \left[-\frac{D\Delta t(1-\theta)}{\Delta x^2} \right] + \phi_P^t \left[1 + \frac{2D\Delta t(1-\theta)}{\Delta x^2} \right] + \phi_{P+\Delta x}^t \left[-\frac{D\Delta t(1-\theta)}{\Delta x^2} \right] = \\ = \phi_{P-\Delta x}^{t-\Delta t} \left[\frac{D\Delta t(\theta)}{\Delta x^2} \right] + \phi_P^{t-\Delta t} \left[1 - \frac{2D\Delta t(1-\theta)}{\Delta x^2} \right] + \phi_{P+\Delta x}^{t-\Delta t} \left[\frac{D\Delta t(\theta)}{\Delta x^2} \right] \end{aligned} \quad (2.25)$$

Once the term θ has been chosen, the distribution of ϕ can be obtained as:

$$[\mathbf{A}] [\phi^t] = [\mathbf{B}] [\phi^{t-\Delta t}] \quad (2.26)$$

$$[\phi^t] = [\mathbf{B}] [\phi^{t-\Delta t}] [\mathbf{A}]^{-1} \quad (2.27)$$

The chosen method depends on the θ -term:

- $\theta=1$ Backward Euler Method.
- $\theta=1/2$ Crank-Nicholson
- $\theta=0$ Forward Euler Method

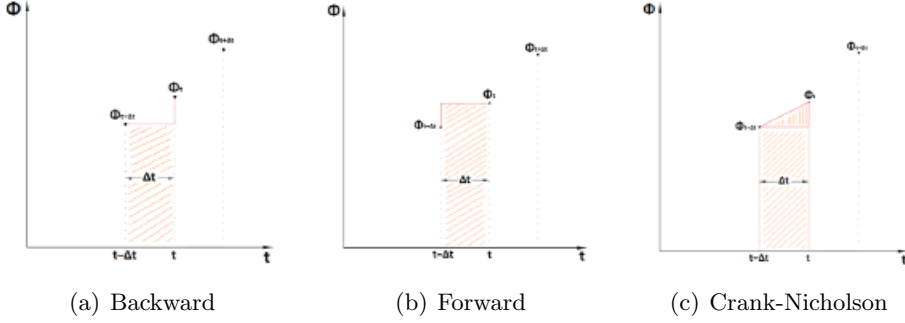


Figure 2.5: Time integration methods.

Discretised transport equation

In previous section, the terms of the transport equation have been discretised. Considering the initial form (equation 2.3), using the spatial discretization it has been obtained:

$$\int_t^{t+\Delta t} \left[\frac{\partial \rho \phi}{\partial t} + \sum_{i=1}^N \mathbf{S}_i (\Gamma \rho u)_i \phi_i + \sum_{i=1}^N \mathbf{S}_i \cdot (\Gamma \rho \nabla \phi)_i \right] dt = 0 \quad (2.28)$$

Moreover, by applying time discretization, the following expression can be obtained:

$$[\mathbf{A}] [\phi] = [\mathbf{R}] \quad (2.29)$$

where $[\mathbf{A}]$ is a sparse matrix, in which the elements of the main diagonal are coefficients referring to the $\mathbf{P}$ - node. The matrix $[\mathbf{A}]$ elements may depend on ϕ values, while $[\phi]$ is made of ϕ -values on the analysed mesh. $[\mathbf{R}]$ is the matrix of source terms.

2.1.3 Discretization of Navier-Stokes

Considering an incompressible fluid (whose viscosity is not dependent from position), the Navier-Stokes and continuity equations can be written as follows:

$$\frac{D\mathbf{u}}{Dt} = \mathbf{g} - \frac{\nabla p}{\rho} - \nu \nabla^2 \mathbf{u} \quad (2.30)$$

In solving these equations, two main problems arise: the non-linearity of the convective term included in the material derivative of the velocity field (i.e. the first member of the above equation) and the coupling of velocity and pressure. The first problem can be handled referring to the paragraph about the discretization of the transport equation:

$$\int_V \nabla \cdot (\mathbf{u}\mathbf{u}) dV = \oint_{\partial V} (\mathbf{u}\mathbf{u}) \cdot \mathbf{n} d\mathbf{S} = \sum_{i=1}^N \mathbf{S}_i (\mathbf{u})_i (\mathbf{u})_i \quad (2.31)$$

By applying one of the above-mentioned methods for the definition of the $\mathbf{u}$ -value on the N -faces, the convective term can be written as:

$$\int_V \nabla \cdot (\mathbf{u}\mathbf{u}) dV = \mathbf{a}_P \mathbf{U}_P + \sum_{i=1}^N \mathbf{a}_{Ni} \mathbf{U}_{Ni} \quad (2.32)$$

where $\mathbf{a}_P$ is the coefficient related to the investigated cells (cells with the computational node $\mathbf{P}$ and N number of faces) and $\mathbf{a}_{N_i}$ are the values yielded by the neighboring cells (indicated by N_i).

On the other hand, $\mathbf{U}_P$ and $\mathbf{U}_{N_i}$ are the values of the variable $\mathbf{u}$ at the nodes $\mathbf{P}$ and $\mathbf{N}_i$.

In this case, the non-linearity arises out the dependence of $\mathbf{a}_P$ and $\mathbf{a}_{N_i}$ from velocity. Another problem is given by the fact that fluxes must satisfy both the continuity equation, which implies that both the continuity equation itself and the Navier-Stokes equation must be solved at the same time, thus creating a larger non-linear system. In order to solve this conundrum, it is normally applied a linearization of the convective term; this said, a flux that satisfies the continuity equation is deployed to define the $\mathbf{a}_P$ and $\mathbf{a}_{N_i}$ terms. In steady-state flux simulations, this method do not affect results, while in transient flow simulations an interaction is needed.

Considering the second mentioned problem, with the aim of deriving the pressure equation a semi-discretised form of the momentum equation is used:

$$\mathbf{a}_P \mathbf{U}_P = \mathbf{H}(\mathbf{u}) - \nabla p \quad (2.33)$$

where the $\mathbf{H}(\mathbf{u})$ matrix is made of the terms yielded by the nearby cells and the source terms of velocity. The terms relates to the $\mathbf{P}$ -node are put in the term $\mathbf{a}_P$, which means that each term in the $\mathbf{H}(\mathbf{u})$ main diagonal equals zero. This said, $\mathbf{H}(\mathbf{u})$ is expressed by the following relation:

$$\mathbf{H}(\mathbf{u}) = - \sum_{i=1}^N a_{N_i} \mathbf{U}_{N_i} + \frac{U_0}{\Delta t} \quad (2.34)$$

A proper combination of the discretised form of the continuity equation and the semi-discretised form of Navier-Stokes solved for $\mathbf{U}_f$ is required to obtain a completely discretized form of Navier-Stokes equation:

$$\int_V \nabla \cdot \mathbf{U} dV = \sum_{i=1}^N \mathbf{S}_i \cdot (\mathbf{u})_i \quad (2.35)$$

$$\mathbf{U}_P = \frac{\mathbf{H}(\mathbf{u})}{\mathbf{a}_P} - \frac{\nabla p}{\mathbf{a}_P} \quad (2.36)$$

$$\mathbf{U}_i = \left(\frac{\mathbf{H}(\mathbf{u})}{\mathbf{a}_P} \right)_i - \left(\frac{\nabla p}{\mathbf{a}_P} \right)_i \quad (2.37)$$

By substituting the relations above in the discretised form of the continuity equation, the completely discretised form of Navier-Stokes equation is obtained:

$$\mathbf{a}_P \mathbf{U}_P = \mathbf{H}(\mathbf{u}) - \sum_{i=1}^N \mathbf{S}_i \cdot (p)_i \quad (2.38)$$

$$\sum_{i=1}^N \mathbf{S}_i \cdot \left[\left(\frac{1}{\mathbf{a}_P} \right)_i (\nabla p)_i \right] = \sum_{i=1}^N \mathbf{S}_i \cdot \left(\frac{\mathbf{H}(\mathbf{U})}{\mathbf{a}_P} \right)_i \quad (2.39)$$

Because of the presence of the pressure-velocity coupling, which is clear considering the above equations, it is necessary to use co-working algorithms for the solution of the system for the whole domain. This procedure is suited for a small number of computational nodes, but it is rare. In other cases, instead, it is chosen to work with segregated approaches. Among these approaches, the most used are PISO and SIMPLE algorithms; they are based on the assignment of an arbitrary initial condition, corrected by the algorithm in each iteration. The SIMPLE algorithm is used per stationary problems solution, while PISO is used for non-stationary problems.

The SIMPLE algorithm (Semi-Implicit Method for Pressure-Linked Equations) follows these stages [29]:

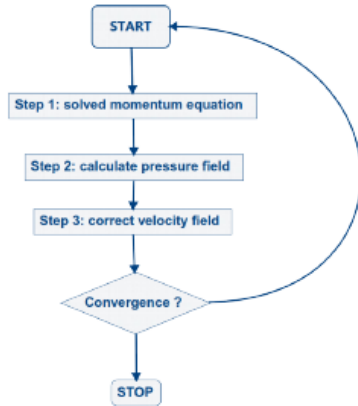
1. the momentum equation is solved using the pressure field calculated in previous time steps. The equation is implicitly under-relaxed and an additional term called velocity under-relaxation factor α_U is provided.
2. the new pressure field is calculated.

3. the new velocity field is calculated and new matrix $\mathbf{H}(\mathbf{u})$ is assembled.
4. in order to take into account the velocity error, the resulting pressure solution should be under-relaxed:

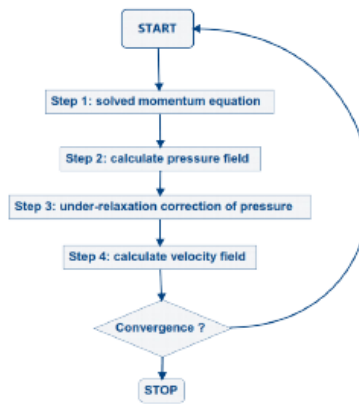
$$p_{New} = p_{Old} + \alpha_P(p_P - p_{Old}) \quad (2.40)$$

where p_{New} is the new pressure field, which is later used to calculate the new velocity field, p_{Old} and p_P represent respectively the pressure field at point 1 and point 2.

5. The new pressure obtained permits the calculation of the new velocity field.
6. The loop repeats itself to point 1 using the new velocity; the cycle is iterated until the tolerance set is reached.



(a) PISO



(b) SIMPLE

Figure 2.6: PISO and SIMPLE algorithms.

This said, the steps which are fundamental to solve the Navier-Stokes equation are listed below:

1. setting up the initial conditions for the variables (e.g. velocity and pressure).
2. starting the calculation of the new timestep value.
3. assembling and solving the momentum predictor equation, based on the available face fluxes.
4. using PISO or SIMPLE algorithm (depends on the nature of the problem) to solve the pressure-velocity coupling.
5. using the continuity equation to solve all the other equations of the system.
6. if the final time step is not achieved, the loop has to start all over again from step 2.

Boundary Conditions

The solution of the discretised equations over a cell requires the definition of the variable value on the cell faces; at mesh boundary, this value can not be yielded by the neighbouring cells and must be imposed arbitrarily. These imposed values are known as “*boundary conditions*” and can be related to the physical boundary of the domain where the fluid-dynamics event takes place.

These conditions can be divided in numerical and physical conditions. Numerical conditions can be further divided into two groups [29]:

- **Dirichlet conditions**, which are imposed on the cell faces. Considering that the fluid-dynamics equations need a specific value at the boundary:
 - the convective term needs the flux values to be discretized at the boundary

$$\mathbf{S}_b(\rho \mathbf{u})_b \phi_b = F_b \phi_b \quad (2.41)$$

where b refers to the boundary faces and F_b to the flux through the b -face.

- The diffusion term requires the following term to be discretized on the boundary:

$$\mathbf{S}_b(\nabla \phi_b) = |\mathbf{S}_b| \frac{\phi_b - \phi_P}{|\mathbf{d}|} \quad (2.42)$$

where $\mathbf{d}$ is the perpendicular distance between the face and the computational nodes.

- **Von Neumann conditions**, in which the value is not directly assigned to the variable, because they are derived by an arbitrarily assigned gradient
 - Convective terms require the value on the boundary to be calculated by resorting to the following relationship:

$$\phi_b = \phi_P + \mathbf{d} \cdot (\nabla \phi)_b = \phi_P + |\mathbf{d}| g_b \quad (2.43)$$

where g_b is the gradient value.

- Diffusive terms require the same method:

$$\mathbf{S}_b(\nabla \phi)_b = \mathbf{S}_b g_b \quad (2.44)$$

The physical boundary conditions are the characteristics and behaviour of the domain on the boundary. In a CFD simulation, the most common physical conditions that must be reproduce are:

- **Inlet boundaries/Outlet boundaries**, which represents the inlet/outlet points of the domain. The velocity field values must be set and the pressure gradient zeroed.

- **Symmetry plane boundaries**, which imply that the component of the gradient orthogonal to the boundary equals zero. The components parallel to the gradient are projected onto the boundary face from inside the domain.
- **Impermeable no-slip wall boundaries**, in which the fluid velocity near the wall must be the same as the wall's and the gradient condition must equal zero.

Chapter 3

Solid-mechanics solution implementation

As research activity several methods, in order to find the most suitable for the structural solution, have been investigated. In particular, as an alternative to the FEM, the FVM, the beam deflection method and finally, the one chosen for the solution, the modal superposition method, have been investigated and tested. An excursus of these methods is reported below.

3.1 The Finite Volume Method

3.1.1 Discretization of Momentum Conservation Law

The spatially discretised form of equation (1.39) for the updated control volume V_{P_u} reads:

$$\begin{aligned} \rho_{p_u} V_{p_u} \frac{\partial \delta \mathbf{v}}{\partial t} dV_u - \sum_f (2\mu_f + \lambda_f) \mathbf{n}_{fu} \cdot (\nabla \delta \mathbf{u})_f S_{fu} = \\ = \sum_f \mathbf{n}_{fu} \cdot \mathbf{q}_f S_{fu} + \sum_{\mathbf{f}} \rho_u (\delta f_b)_P V_{p_u} \end{aligned} \quad (3.1)$$

where the subscripts P and f represent the cell and face values. The face values of all dependent variables are calculated by linear interpolation of the neighbouring cell values.

The face normal derivative of the displacement increment $\mathbf{n}_{fu} \cdot (\nabla \delta \mathbf{u})_f$ is

discretised as follows:

$$\mathbf{n}_{fu} \cdot (\nabla \delta \mathbf{u})_f = |\Delta_{fu}| \frac{\delta \mathbf{u}_n - \delta \mathbf{u}_P}{|\mathbf{d}_{fu}|} + (\mathbf{n}_{fu} - \Delta_{fu}) \cdot (\nabla \delta \mathbf{u})_f \quad (3.2)$$

where:

- $\Delta_f = \frac{\mathbf{d}_f}{\mathbf{d}_f \cdot \mathbf{n}_f}$,
- $|\Delta_{fu}| = \frac{\delta \mathbf{u}_N - \delta \mathbf{u}_P}{\mathbf{d}_{fu}}$ is the orthogonal contribution in the above equation and it is treated implicitly,
- $(\mathbf{n}_{fu} - \Delta_{fu}) \cdot (\nabla \delta \mathbf{u})_f$ is the non-orthogonal correction and it is explicit.

The cell gradient of the displacement increment vector used for the calculation of the tensor $\mathbf{q}$ and the non-orthogonal correction, is obtained by using the least-squares fit. This method produces a second-order accurate gradient irrespective of the mesh quality. The temporal discretisation of equation (3.1) is performed by using two implicit schemes: the first-order accurate Euler scheme and the second-order accurate three time level scheme referred to as the backward scheme. In both schemes, all terms of are evaluated at the new time instance $t^n = t^0 + \delta t$, and the velocity increment $\delta \mathbf{v}$ and its temporal derivative are discretised by using one old-time level in the Euler scheme:

$$\delta \mathbf{v}_P^n = \frac{\delta \mathbf{u}_P^n - \delta \mathbf{u}_P^0}{\delta t} \quad (3.3)$$

$$\left(\frac{\partial \delta \mathbf{v}_P}{\partial t} \right)^n = \frac{\delta \mathbf{v}_P^n - \delta \mathbf{v}_P^0}{\delta t} \quad (3.4)$$

and two old-time levels in the backward scheme:

$$\delta \mathbf{v}_P^n = \frac{3\delta \mathbf{u}_P^n - 4\delta \mathbf{u}_P^0 + \delta \mathbf{u}_P^{00}}{2\delta t} \quad (3.5)$$

$$\left(\frac{\partial \delta \mathbf{v}_P}{\partial t}\right)^n = \frac{3\delta \mathbf{v}_P^n - 4\delta \mathbf{v}_P^0 + \delta \mathbf{v}_P^{00}}{2\delta t} \quad (3.6)$$

where $\delta \mathbf{v}_P^n = \delta \mathbf{v}_P(t^0 + \delta t)$, $\delta \mathbf{v}_P^0 = \delta \mathbf{v}_P(t^0)$, and $\delta \mathbf{v}_P^{00} = \delta \mathbf{v}_P(t^0 - \delta t)$. When equations (3.2, 3.5, 3.6) are substituted into equation (3.1), the fully discretised form of the linear momentum conservation law (1.39) can be written in the form of linear algebraic equation, which for cell P reads:

$$a_P \delta \mathbf{u}_P^n + \sum_N a_N \delta \mathbf{u}_N^n = \mathbf{r}_P \quad (3.7)$$

where the diagonal coefficient a_P , the neighbour coefficient a_N and the source term $\mathbf{r}_P$ are defined for the backward scheme by the following expressions:

$$a_P = \frac{9\rho_{P_u} V_{P_u}}{4\delta t} + \sum_f (2\mu_f^n + \lambda_f^n) \frac{|\Delta_{f_u}|}{\mathbf{d}_{f_u}} S_{f_u} \quad (3.8)$$

$$a_N = - \sum_f (2\mu_f^n + \lambda_f^n) \frac{|\Delta_{f_u}|}{\mathbf{d}_{f_u}} S_{f_u} \quad (3.9)$$

$$\begin{aligned} \mathbf{r}_P = & \rho_{P_u} V_{P_u} \left(\frac{3\delta \mathbf{u}_P^0}{\delta t^2} - \frac{3\delta \mathbf{u}_P^{00}}{4\delta t^2} + \frac{3\delta \mathbf{v}_P^0}{\delta t} + \frac{\delta \mathbf{v}_P^{00}}{2\delta t} \right) + \\ & + \sum_f (2\mu_f + \lambda_f) (\mathbf{n}_{f_u} - \Delta_{f_u}) \cdot (\nabla \delta \mathbf{u})_f^n S_{f_u} + \\ & + \sum_f \mathbf{n}_{f_u} \cdot \mathbf{q}_f^n S_{f_u} + \sum_f \rho_{P_u} (\delta \mathbf{f}_b)_P^n V_{P_u} \end{aligned} \quad (3.10)$$

Solution procedure

Equation (3.7) is assembled for all CVs in the computational mesh, resulting in the following system of linear algebraic equations:

$$[A] \{\delta \mathbf{u}\} = \{\mathbf{r}\} \quad (3.11)$$

where $[A]$ is the sparse matrix, with coefficients a_P on the diagonal and a_N off the diagonal, $\delta \mathbf{u}$ is the vector consisting of displacement increments $\delta \mathbf{u}_P$ for all CVs and $\mathbf{r}$ is the righthand side vector consisting of source terms r_P for all CVs.

The system of equations (3.11) is solved by using a segregated algorithm, where the three components of the displacement vector increment are solved separately from each other. Since nonlinear and coupling terms depending on the unknown displacement vector increment are placed in the right-hand side vector, the system is solved in the iterative manner where, each outer iteration the right hand-side vector $\mathbf{r}$ is updated at the beginning of each outer iteration by using the displacement vector increment from the previous iteration. When the solution changes less than some pre-defined tolerance, the system is considered to be solved. This is done for every time-step of the transient simulation.

At the end of each time step, the total displacement vector and the total second Piola-Kirchhoff stress tensor are updated as follows:

$$\mathbf{u}^n = \mathbf{u}^0 + \delta \mathbf{u}^n \quad (3.12)$$

$$\boldsymbol{\Sigma}_u^n = \boldsymbol{\Sigma}_u^0 + \delta \boldsymbol{\Sigma}_u^n \quad (3.13)$$

where the second Piola-Kirchhoff stress tensor increment $\boldsymbol{\Sigma}_u^n$ is calculated according to equation (1.38) by using the newly calculated displacement vector increment $\delta \mathbf{u}^n$. When the mesh is moved at the beginning of the new time step by using the displacement increment from the previous time step, the second Piola-Kirchhoff stress tensor related to the old updated configuration must be transformed to the new updated configuration:

$$\boldsymbol{\Sigma}_u^0 = \frac{1}{\det \mathbf{F}_u^0} \mathbf{F}_u^0 \cdot \boldsymbol{\Sigma}_u^0 \cdot (\mathbf{F}_u^0)^T \quad (3.14)$$

Boundary Conditions

The specification of the problem is completed with, in addition to the definition of the solution domain in space and time, the characterization of the initial and boundary conditions. The initial condition consists of the specified distribution of $\mathbf{u}$ and $\delta\mathbf{u} = \delta t$ at time zero. The boundary conditions, either constant or time varying, can be of the following type:

- fixed displacement,
- planes of symmetry,
- fixed pressure,
- fixed traction,
- free surfaces (zero traction).

The Finite Volume Method applied to the structural solution has been tested using solvers included in the open-source software OpenFOAM, based on the Finite Volume Method for both the fluid dynamics and the structural solutions. However, the results obtained were not satisfactory. Considering the partitioned structure of the tested solver, it was therefore decided to replace, for the structural solution, the Finite Volume Method with a theory-based method, the Beam Deflection Method. This method represents a certain reference for the the displacements field evaluation in the case of simple linear elastic structures, in structural mechanics. This choice was based on the fact that the structural FVM has been validated in previous work on non-multiphysical systems; this does not imply possible problems in FSI systems applications. Furthermore, the substitution of this method allowed the possibility of developing a linear and computationally light algorithm for the structural solution.

3.2 The Beam Deflection Method

The basics of the Beam Deflection Method state in the assumption that the resolution of particular cases allows to find the solution of more complex problems. By considering the linearity of the elastic problem governing equations, the superposition of the effects is allowed: the behaviour of an elastic-linear body subjected to every kind of combination of external effects can be found by overlaying the effects related to every single action applied [12].

Another important tool for several applications is the *elastic equivalence principle*. In several cases, it allows to neglect the local distribution of the forces applied on a limited portion of space or volume, because, at a sufficient distance from the portion itself, the behaviour depends on the resultant loads.

This principle is usually known as *De Saint Venant principle*, was introduced by De Saint Venant itself by referring to cylindrical bodies subjected to loads applied on the external faces and extended by Boussinesq to more general cases. Moreover, this principle enables the resolution of specific cases; this is possible by considering, under the same resultant loads, the external actions distributions which makes the resolution simpler. This resolution can be considered valid, at a certain distance from the loaded area, also for the original problem.

The De Saint Venant principle has significant applicative aspects. Its solution represents the base of the beam theory. As a further consideration, it must be noted that the De Saint Venant principle allows the calculation, without further hypotheses, of the structural behaviour of a constant section beam, made of a linear elastic material, homogeneous and isotropic, which is loaded on the external sections.

In order to complete the description of the beam deflection method and its application for this work aims, the following paragraphs report a concrete example of the application of the method itself on a system with mechanical features similar to the system used for the solver test, with simplified

load conditions.

The test case consists in a cantilever beam subjected to distributed loads along its axis; the loads themselves represents the pressure field applied on the structure by the fluid flow. Referring to the beam deflection theory, the solid domain should be divided into a number of subdomains equal to the discontinuities of the applied load.

The aim of the method is the computation of the displacement of the border points for each subdomain. In each subdomain, the stiffness of each subdomain is stated, with the exception of the analysed one; considering this assumption, the displacement and the rotation of the border point (v and ϕ) are calculated, by considering the subdomain subjected to a load system which consists in the load distributed on the subdomain itself and the sum of the the distributed loads acting on the rigid subdomains, analysed as concentrated loads acting on the subdomain border point (by referring to the De Saint Venant Principle).

Since the structure is linear-elastic, the displacement of each subdomain border point can be computed through the effects superposition, i.e. the summatory of the point displacement in each subscheme. In each subscheme, the border point displacement is calculated by summing the displacement due to the distributed load, the concentrated force and the concentrated moment. The displacement function for each addend can be found by referring to the linear elasticity equations. This procedure is followed also for the computation of the rotation of the elastic subdomain border point, which allows the calculation of the rigid subdomains border points displacement, by considering their distance from the elastic subdomain.

3.2.1 Solving algorithm

The above-described procedure can be applied on the cantilver beam; for the sake of simplicity, the structure is subjected to a load system

applied onto the midline, by considering the loads themselves as the sum of the distributed loads above and below the beam, in case of equivalent abscissa. As a greater simplification, the structure has been subjected to four different distributed loads along its axis.

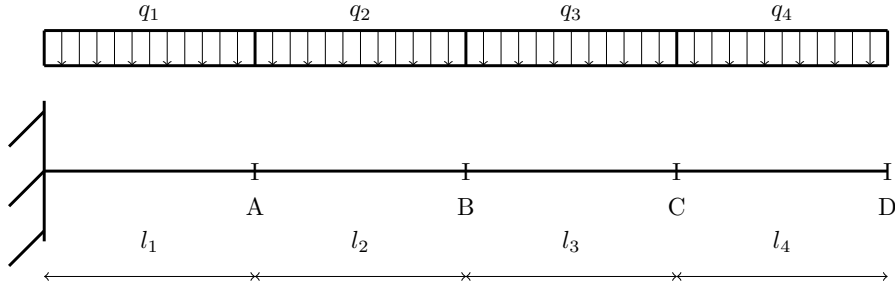


Figure 3.1: Graphical representation of the application example.

As it has been said in the previous paragraphs, the main aim of the beam deflection method is to calculate the displacement of the subdomain border points; in this case, they are v_A , v_B , v_C and v_D , considering that the solid domain has been divided into a number of subdomains equal to the number of distributed loads.

For each elastic subdomain, the displacement v and the rotation ϕ are calculated through the linear elasticity equations:

$$v_B = \frac{ql^4}{8EI} \quad \phi_B = -\frac{ql^3}{6EI}$$

Figure 3.2: Linear elasticity functions for a cantilever beam.

First sub-scheme

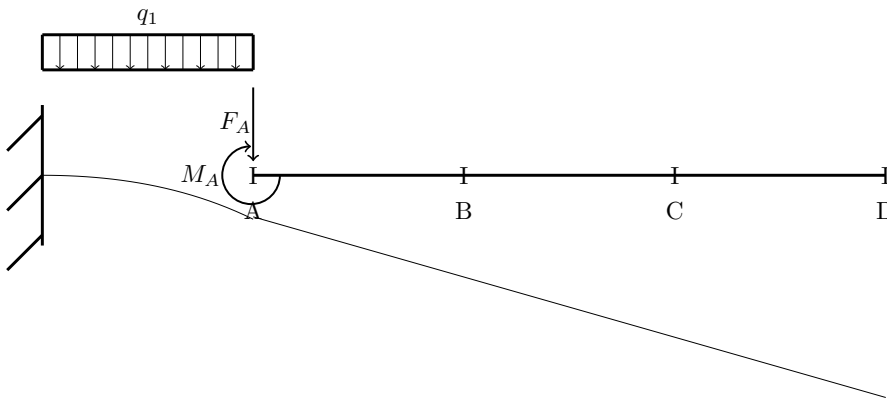


Figure 3.3: Graphical representation of the first sub-scheme of the application example.

In the first sub-scheme (figure 3.3), all the subdomains are rigid, with the exception of the first one, which is subjected to q_1 , F_A and M_A , which are equals to:

$$F_A = F_2 + F_3 + F_4 \quad (3.15)$$

$$M_A = M_2 + M_3 + M_4 \quad (3.16)$$

where:

$$F_2 = q_2 l_2$$

$$F_3 = q_3 l_3$$

$$F_4 = q_4 l_4$$

and

$$M_2 = F_2 l_2 / 2$$

$$M_3 = F_3 (l_3 / 2 + l_2)$$

$$M_4 = F_4 (l_4 / 2 + l_3 + l_2)$$

By applying the linear elasticity equations, it can be obtained:

$$v_A^I = \frac{q_1 l_1^4}{8EI} + \frac{F_A l_1^3}{3EI} + \frac{M_A l_1^2}{2EI} \quad (3.17)$$

$$\phi_A^I = -\frac{q_1 l_1^3}{6EI} - \frac{F_A l_1^2}{2EI} - \frac{M_A l_1}{EI} \quad (3.18)$$

By considering the rigid rotation of the beam, it can be obtained:

$$v_B^I = v_A^I + \phi_A^I l_2 \quad (3.19)$$

$$v_C^I = v_A^I + \phi_A^I (l_2 + l_3) \quad (3.20)$$

$$v_D^I = v_A^I + \phi_A^I (l_2 + l_3 + l_4) \quad (3.21)$$

Second sub-scheme

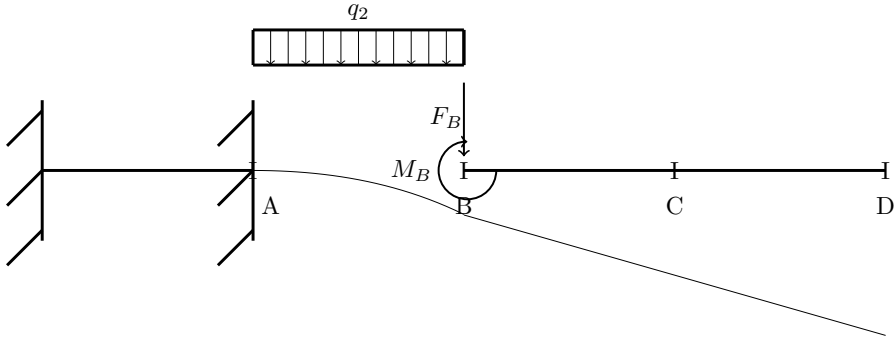


Figure 3.4: Graphical representation of the second sub-scheme of the application example.

In the second sub-scheme (figure 3.4), all the subdomains are rigid, with the exception of the second one, which is subjected to q_2 , F_B and M_B , which are equals to:

$$F_B = F_3 + F_4 \quad (3.22)$$

$$M_B = M_3 + M_4 \quad (3.23)$$

where:

$$F_3 = q_3 l_3$$

$$F_4 = q_4 l_4$$

and

$$M_3 = F_3(l_3/2)$$

$$M_4 = F_4(l_4/2 + l_3)$$

By applying the linear elasticity equations, it can be obtained:

$$v_B^{II} = \frac{q_2 l_2^4}{8EI} + \frac{F_B l_2^3}{3EI} + \frac{M_B l_2^2}{2EI} \quad (3.24)$$

$$\phi_B^{II} = -\frac{q_2 l_2^3}{6EI} - \frac{F_B l_2^2}{2EI} - \frac{M_B l_2}{EI} \quad (3.25)$$

By considering the rigid rotation of the beam, it can be obtained:

$$v_A^{II} = 0 \quad (3.26)$$

$$v_C^{II} = v_B^{II} + \phi_B^{II} l_3 \quad (3.27)$$

$$v_D^{II} = v_B^{II} + \phi_B^{II} (l_3 + l_4) \quad (3.28)$$

Third sub-scheme

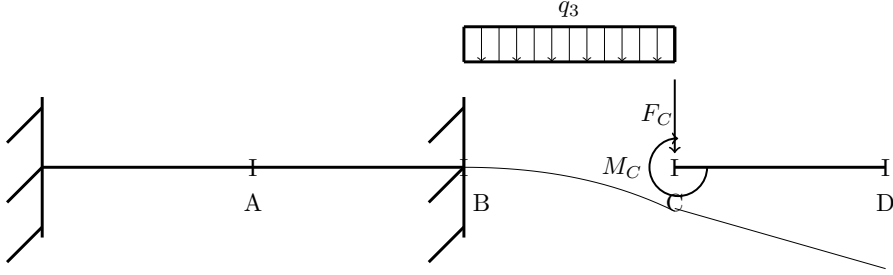


Figure 3.5: Graphical representation of the third sub-scheme of the application example.

In the third sub-scheme (figure 3.5), all the subdomains are rigid, with the exception of the third one, which is subjected to q_3 , F_C and M_C , which are equals to:

$$F_C = F_4 \quad (3.29)$$

$$M_C = M_4 \quad (3.30)$$

where:

$$F_4 = q_4 l_4$$

and

$$M_4 = F_4 l_4 / 2$$

By applying the linear elasticity equations, it can be obtained:

$$v_C^{III} = \frac{q_3 l_3^4}{8EI} + \frac{F_C l_3^3}{3EI} + \frac{M_C l_3^2}{2EI} \quad (3.31)$$

$$\phi_C^{III} = -\frac{q_3 l_3^3}{6EI} - \frac{F_C l_3^2}{2EI} - \frac{M_C l_3}{EI} \quad (3.32)$$

By considering the rigid rotation of the beam, it can be obtained:

$$v_A^{III} = 0 \quad (3.33)$$

$$v_B^{III} = 0 \quad (3.34)$$

$$v_D^{III} = v_C^{III} + \phi_C^{III} l_4 \quad (3.35)$$

Fourth sub-scheme

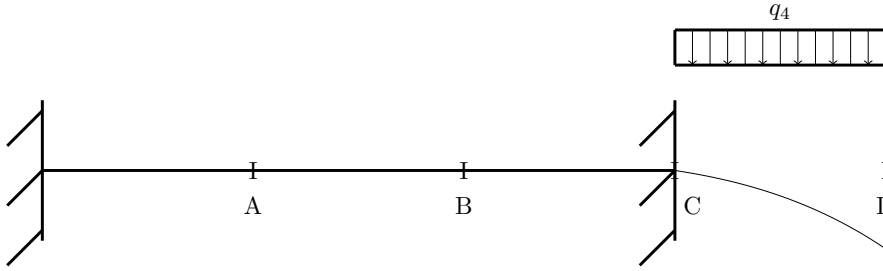


Figure 3.6: Graphical representation of the fourth sub-scheme of the application example.

In the fourth sub-scheme (figure 3.6), all the subdomains are rigid, with the exception of the fourth one, which is subjected to q_4 .

By applying the linear elasticity equations, it can be obtained:

$$v_D^{IV} = \frac{q_4 l_4^4}{8EI} \quad (3.36)$$

By considering the rigid rotation of the beam, it can be obtained:

$$v_A^{IV} = 0 \quad (3.37)$$

$$v_B^{IV} = 0 \quad (3.38)$$

$$v_C^{IV} = 0 \quad (3.39)$$

After the computation of each displacement value at each subdomain border point for each subscheme, the total displacement for each of these points is calculated as:

$$v_A = v_A^I + v_A^{II} + v_A^{III} + v_A^{IV} \quad (3.40)$$

$$v_B = v_B^I + v_B^{II} + v_B^{III} + v_B^{IV} \quad (3.41)$$

$$v_C = v_C^I + v_C^{II} + v_C^{III} + v_C^{IV} \quad (3.42)$$

$$v_D = v_D^I + v_D^{II} + v_D^{III} + v_D^{IV} \quad (3.43)$$

Although the Beam Deflection Method is an accurate and certain method for the structural solution, it has been chosen to replace this method which allows the displacement field calculation in the single time-step as a stationary solution. In fact, in order to extend the solver application field to transient systems, it has been necessary to consider the inertial effects, which play a predominant role in the time-varying analysis.

It was therefore implemented an algorithm from the dynamics of structures theory, known as Modal Superposition Method.

3.3 The Modal Superposition Method

A wide range of mechanical systems can be described using a finite number of degrees of freedom (DOF); this happens when there are elements with high elasticity and low mass and, at the same time, elements with high mass and high stiffness [1]. When, on the contrary, the system has an infinite number of "mass points" and/or has no "concentrations" of deformable zones or members, an infinite number of coordinates are required to specify its deformed configuration. Models having a finite number of degrees of freedom are called *lumped-parameter* or *discrete* systems, while those with an infinite number of degrees of freedom are called

distributed or continuous systems.

Often, continuous systems are approximated by discrete models; this makes it easier to obtain the solution to the dynamic problem. The choice of modeling a given system as discrete or continuous depends on the purpose of the analysis and the expected accuracy of the results. The motion of an n degree-of-freedom system is governed by a system of n coupled second-order ordinary differential equations. For a continuous system, the governing equation of motion is in the form of a partial differential equation. Since the solution of a set of ordinary differential equations is simple, it is relatively easy to find the response of a discrete system that is experiencing a specified excitation. On the other hand, solution of a partial differential equation is more involved, and closed-form solutions are available for only a few continuous systems that have a simple geometry and simple boundary conditions and excitations. For an n -degree-of-freedom system, there will be, at most, n distinct natural frequencies of vibration with a mode shape corresponding to each natural frequency. A continuous system, on the other hand, will have an infinite number of natural frequencies, with one mode shape corresponding to each natural frequency. A continuous system can be approximated as a discrete system, and its solution can be obtained in a simpler manner by increasing the DOF number.

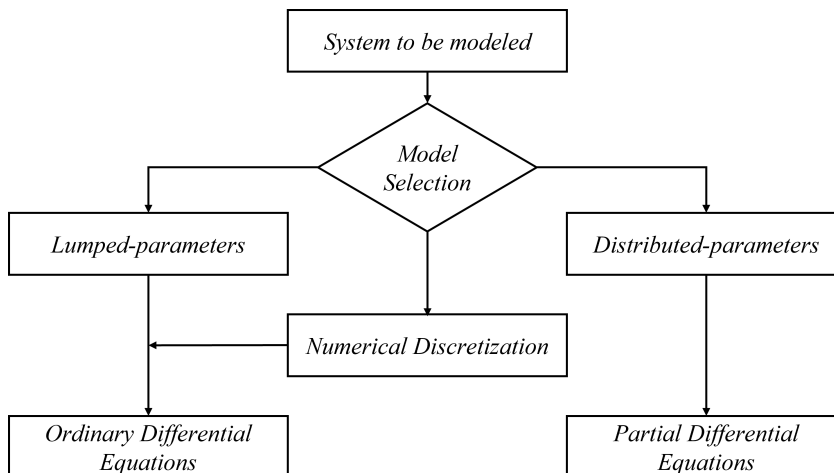


Figure 3.7: Modeling methodology.

The characteristics of a system influence its response to given perturba-

tions. To determine the response of such a system to these perturbations, i.e. for the differential equations of motion resolution, it is of fundamental importance whether the system is linear or non-linear. In particular, in a linear system, the response is directly proportional to the forces; if this is not the case, the system is non-linear. In a linear system, the solution research can be simplified by exploiting the superposition principle. This principle states that in a linear system the total effect can be seen as the sum of the individual effects caused by each excitation, individual effects which are combined linearly to obtain the final response, as shown in Fig. 3.8. There are many types of external forces (or moments) and determining the response involves different techniques for different types. Different types of excitations require different methods of solution [43].

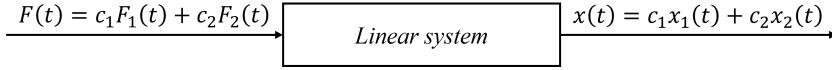


Figure 3.8: Excitation response for linear system.

The dynamic equation of motion, in case of a n-DOFs system, is:

$$[M] \{\ddot{y}\} + [C] \{\dot{y}\} + [K] \{y\} = \{g(t)\} \quad (3.44)$$

with $[M]$ the inertial matrix, $[K]$ the stiffness matrix, $[C]$ the damping matrix, y the displacement vector for the n-DOFs dynamic and $g(t)$ the force vector for the n-DOFs:

$$\{g(t)\}^T = \left\{ g_1(t) \quad g_2(t) \quad \dots g_n(t) \right\}^T \quad (3.45)$$

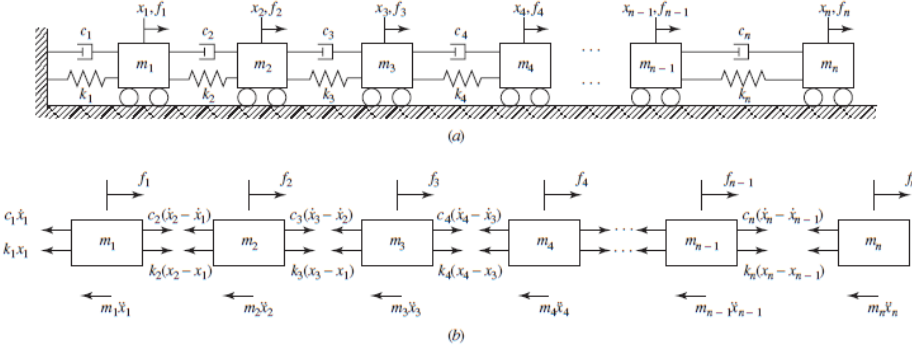


Figure 3.9: (a) An n DOF system; (b) Free-body diagrams of the masses [55].

The n -differential equations become independent of one another using properly the principal, or natural, coordinates. These latter do not represent real displacements, they are abstract quantities and their superposition allows the individual masses motion description.

Through the natural coordinates v :

$$[M] [\Phi] \{\ddot{v}\} + [C] [\Phi] \{\dot{v}\} + [K] [\Phi] \{v\} = \{g(t)\} \quad (3.46)$$

where the modal matrix $[\Phi]$ has many rows equal to the n -degrees of freedom and the considered vibration modes as columns.

By multiplying both the equation members of (3.46) by $[\Phi]^T$, it can be obtained:

$$[\Phi]^T [M] [\Phi] \{\ddot{v}\} + [\Phi]^T [C] [\Phi] \{\dot{v}\} + [\Phi]^T [K] [\Phi] \{v\} = [\Phi]^T \{g(t)\} \quad (3.47)$$

Under the proportional damping hypothesis, from (3.47) it states:

$$[M]_C \{\ddot{v}\} + [C]_C \{\dot{v}\} + [K]_C \{v\} = [\Phi]^T \{g(t)\} \quad (3.48)$$

where $[M]_C = [\Phi]^T [M] [\Phi]$, $[C]_C = [\Phi]^T [C] [\Phi]$ and $[K]_C = [\Phi]^T [K] [\Phi]$.

The subscript C denotes matrices obtained from the mass normalization process. The coefficient matrices, although symmetric, are fully populated since these are simultaneous equations. Their solutions can be only obtained by making the equations independent through the modal analysis. Thus, a mass normalization process is required. Through linear algebra, the solution of an algebraic eigenvalue problem and an orthogonal transformation using the modal matrix are made.

(3.48) represents a decoupled equations system where vector $[\Phi]^T g(t)$ members are the modal forces:

$$[\Phi]^T \{g(t)\} = \begin{bmatrix} T_{11}g_1(t) + T_{21}g_2(t) + \dots + T_{n1}g_n(t) \\ T_{12}g_1(t) + T_{22}g_2(t) + \dots + T_{n2}g_n(t) \\ T_{1n}g_1(t) + T_{2n}g_2(t) + \dots + T_{nn}g_n(t) \end{bmatrix} \quad (3.49)$$

In equation (3.49), T terms represent the modal matrix components. The natural coordinates and principal, or modal, coordinates are linked by the modal matrix as follows:

$$\{y(t)\} = [\Phi] \{v(t)\} \quad (3.50)$$

In the system of equations (3.44) real coordinates can be replaced with modal coordinates, in order to obtain the equations of motion.

Exploiting the orthogonality property of the vectors associated to the modes of vibration, the system of equations (3.44) can be reduced to decoupled equations of motion for each degree of freedom. Thus, the respective equation of motion for each i degree can be obtained as described in [6], denoted by ω the pulsation:

$$\{\ddot{v}_i(t)\} + 2\xi_i\omega_i \{\dot{v}_i(t)\} + \omega_i^2 \{v_i(t)\} = \frac{G_{C_i}}{M_{C_i}} \quad (3.51)$$

where:

$$G_{C_i} = \{\Phi\}_i^T \{g(t)\} \quad (3.52)$$

$$\xi_i = \frac{\alpha + \beta\omega_i^2}{2\omega_i} \quad (3.53)$$

In order to solve decoupled equations of motion, Duhamel convolution

integral can be used. Although it is possible to solve the equation of motion by finite differences method explicitly, with the Duhammel integral convergence is guaranteed. In fact, the explicit approach is strongly conditioned by the choice of Δt . The response of a linear system with constant coefficients can be seen as a superposition of different magnitudes impulse responses, applied at different times. This superposition is known as the convolution integral or the superposition integral. Through the Duhamel convolution integral on each dynamic degree of freedom, displacement response can be obtained:

$$v_i(t) = \frac{1}{M_{C_i}\omega_{C_i}} \int_0^t G_{C_i}(\tau) e^{-\xi_i\omega_{C_i}(t-\tau)} \sin \omega_{C_i}(t-\tau) d\tau \quad (3.54)$$

where:

$$\omega_{C_i} = \omega_i \sqrt{1 - \xi_i^2} \quad (3.55)$$

The n-dof system dynamics analysis is reduced to the calculation of the Duhamel convolution integral.

3.3.1 Numerical algorithm

In order to integrate numerically the Duhamel convolution integral, the trapezoids method can be used. It approximates, with a trapezoid area, the area delimited by the function $z : [a, b] \rightarrow R$. In order to approximate the integral of $z(x)$, the interval $[a, b]$ can be divided into two sub-ranges $[a, x_1]$ and $[x_1, b]$:

$$\begin{aligned} \int_a^b z(x) dx &= \int_a^{x_1} z(x) dx + \int_{x_1}^b z(x) dx \approx \frac{\delta}{2} [z(a) + z(x_1)] + \frac{\delta}{2} [z(b) + z(x_1)] = \\ &= \frac{\delta}{2} [z(a) + 2z(x_1) + z(b)] = \frac{b-a}{4} [z(a) + 2z(x_1) + z(b)] \end{aligned} \quad (3.56)$$

$[a, b]$ interval can be divided in n sub-ranges so that $\delta = (b-a)/n$ to reduce the integration error. The integral of $z(x)$ can be calculated as:

$$\int_a^b z(x) dx = \int_a^{x_1} z(x) dx + \dots + \int_{x_{n-1}}^b z(x) dx \approx$$

$$\begin{aligned}
& \approx \frac{\delta}{2} [z(a) + z(x_1)] + \dots \frac{\delta}{2} [z(b) + z(x_{n-1})] = \\
& = \frac{\delta}{2} [z(a) + 2z(x_1) + \dots + 2z(x_{n-1}) + z(b)] = \\
& = \frac{b-a}{2n} [z(a) + 2z(x_1) + \dots + 2z(x_{n-1}) + z(b)] \quad (3.57)
\end{aligned}$$

The numerical error linked to the method of trapezoids is related to the integration step δ . If this value decreases, it can be obtained a better estimate. In this work, the trapezoids method has been applied on the relation (3.54) for the determination of P and Q coefficients:

$$\begin{aligned}
v(t) &= \int_0^\tau \frac{G(\tau)}{m\omega} e^{-\xi\omega(t-\tau)} \sin \omega(t-\tau) d\tau = \\
&= \left(\int_0^\tau \frac{G(\tau)}{m\omega} e^{\xi\omega\tau} \cos \omega\tau d\tau \right) e^{-\xi\omega t} \sin \omega t + \\
&- \left(\int_0^\tau \frac{G(\tau)}{m\omega} e^{\xi\omega\tau} \sin \omega\tau d\tau \right) e^{-\xi\omega t} \cos \omega t = \\
&= P(t) e^{-\xi\omega t} \sin \omega t - Q(t) e^{-\xi\omega t} \cos \omega t \quad (3.58)
\end{aligned}$$

where:

$$P(t) = \int_0^\tau \frac{G(\tau)}{m\omega} e^{\xi\omega\tau} \cos \omega\tau d\tau \quad (3.59)$$

$$Q(t) = \int_0^\tau \frac{G(\tau)}{m\omega} e^{\xi\omega\tau} \sin \omega\tau d\tau \quad (3.60)$$

Chapter 4

modSFSiFoam solver code

The solver realized during the Ph.D. activity has been implemented in OpenFOAM. This is an open source software developed for Linux distributions. It is released with a free licence, together with the entire modifiable C++ source code which allows users contributions to the development of the software itself. FOAM states for *Field Operation And Manipulation*. This represents one of the most relevant strenghts of the software. It has not graphical interfaces; the OpenFOAM user has a deep knowledge of numerical analysis, programming and continuum mechanics and can modify the source code in order to customize and upgrade the performances of OpenFOAM itself [49].

In fact, this software has a complex structure. The documentation and user interface are not as comprehensive as those for commercial codes. However, OpenFOAM comes with pre-processing and post-processing environments, ensuring easy and consistent data management across all environments. Third-party software can be used at these stages, such as ParaView, used for data visualization and analysis, which communicates with OpenFOAM through the paraFoam utility. An overview is reported in Figure 4.1.

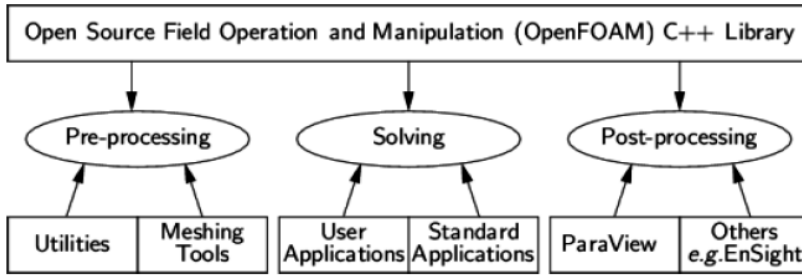


Figure 4.1: OpenFOAM structure

Therefore, OpenFOAM is not a monolithic code. OpenFOAM is a development environment in which several libraries and applications are collected. The applications are divided into two categories:

- *Solvers*, which are built to solve specific continuum mechanics problems.
- *Utilities*, which are developed for pre-processing and post-processing tasks, mainly involving data manipulation and algebraic calculations.

The OpenFOAM distribution collects several solvers and utilities which covers a wide range of problems; moreover, other solvers can be realised for the user needs. This software is capable in treating differential equations naturally, by exploiting the object-oriented nature of the C++ language.

By the end of the 1980s, OpenFOAM has been developed at the Imperial College of London by Henry Weller. Weller's aim was to develop a numerical calculation platform which could be more flexible than FORTRAN monolithic programs; nowadays, it is exploited for fluid dynamics and gas dynamics phenomena study. Considering that a wide range of physical systems can be treated using OpenFOAM, a large number of researchers (principally from Alojz Ivankovic research group) have been involved in the development of *utilities* and *solvers* for solid mechanics. The code has been assembled by Philip Cardiff, from the University College of Dublin, but also other researchers have contributed to its development, such as Aleksandar Karac, Zeljko Tukovic, Hrvoje Jasak, Declan Carolan,

Michael Leonard, Valentine Kanyanta, David McAuliffe, Declan McNamara and Tian Tang, which represent a reference for the research work here described [32][30][62].

4.0.1 Object-orientation and C++

Object-oriented programming languages, such as C++, provide the mechanism classes to declare types and operations that are part of the verbal and mathematical languages. E.g., the velocity is a vector field for which there should exist, in an object-oriented code, a *vectorField* class. The velocity field U would then be an instance, or object, of the *vectorField* class; hence the term object-oriented. The clarity of having objects in programming that represent physical objects and abstract entities should not be underestimated [48].

The class structure collects code development to contained regions of the code, i.e. the classes themselves, and makes the code easier to manage. New classes can be derived or inherit properties from other classes, e.g. the *vectorField* can be derived from a *vector* class and a *Field* class. C++ provides the mechanism of template classes such that the template class *FieldType* can represent a field of any *Type*, e.g. scalar, vector, tensor.

4.0.2 Solver codes

Solver codes are a close representation of solution algorithms and equations; this said, the procedural nature of algorithms and equations can be found in solver codes themselves. Users do not need a deep knowledge of object-orientation and C++ programming to write a solver; otherwise, a proper knowledge of the principles behind object-orientation and classes is recommended, in addition to a basic knowledge of some C++ code syntax [48].

An understanding of the underlying equations, models and solution method and algorithms is far more important. The essence of object-orientation is

that the user should not have to know the code of each class; the knowledge of the class functionality are sufficient to use the class itself.

4.0.3 OpenFOAM FSI case structure

A FSI case, suited for multiphysics simulations, can be built by the user, based on how the FSI solvers in OpenFOAM are designed, of two main folders: *Solid* and *Fluid*. *Solid* folder includes the structural specifications of the simulation, while *Fluid* folder collects the fluid ones. Both the folders are made of three main directories: *0*, *Constant* and *System*, in which the problem specifications are collected. All of the informations must be specified in editable run-time text files.

- The *0* folder collects the initial conditions and there is a dedicated file for all the variables. In solid mechanics applications, the only variable is the displacement, represented by $\mathbf{u}$. Concerning the fluid dynamic application, the variables are pressure p and velocity $\mathbf{u}$.
- The *Constant* folder collects, concerning the *Solid* folder, the *rheology Properties* file, in which the rheological properties of the material are indicated (i.e. Young modulus, Poisson ratio and density). Otherwise, considering the *Fluid* folder, it gathers the *transportProperties* file, in which the kinematic viscosity and fluid density are stated, and the *couplingProperties* file, which contains the fluid-structure coupling properties.

Both the *Constant* folders contain the *polyMesh* subfolder, concerning the domain discretization. It is made of two files:

- *BlockMeshDict*: it collects a list of point and vertices, which are necessary to characterize the domain. Moreover, it collects the controls which are fundamental for the domain division into control volumes and specify the characteristics of every domain

face. This file allows to generate a non-structured mesh.

- *Boundary*: it collects a list of the boundary conditions to be applied on every surface of the domain.

- The *System* folder collects data related to the numerical solution of the differential equations. It is divided into:
 - *controlDict*: in which are collected the initial and final timestep, the timestep dimension and the time precision.

 - *decomposeParDict*: in which are collected the decomposition informations for parallel computing.

 - *fvSchemes*: in which are specified the discretization schemes for each term of the differential equation.

 - *fvSolution*: in which are collected the solution informations, about algorithms and every iteration tolerances .

4.0.4 Solver structure

modSFsiFoam has been implemented inside the *Solver* application. Once the simulation parameters has been set in an appropriate *case*, specific for the resolution of the examined FSI problem, the solver, recalled from Linux terminal, acquires such data and begins to solve the problem through the algorithm reported below. This FSI solver is applied as transient solver for incompressible, laminar flow of Newtonian fluids with mesh motion, and for a linear elastic structure.

The solver refers to the structure of the *IcoFsiFoam* solver, which belongs to the *SolidMechanics* field and is part of the *Deprecated Solvers* folder because of the development of the upgraded *icoFsiElasticNonLinULSolidFoam*, which has been tested, with bad results obtained, in previous works. The structure of *IcoFsiFoam* solver has been taken up because of its simplicity and linearity, before the application of further modifications.

In particular, the applied modifications are based on the resolution of the structural domain through the above-mentioned theoretical-based approach (*SolveSolid.H*) and, consequently, on the chosen techniques for the pressure field transmission to the structure (*setPressure.H*) and for the communication of the structure displacement, in terms of fluid mesh movement (*setMotion.H*), which is the input of the FVM fluid dynamic solution (*solveFluid.H*). Boundary conditions have been imposed on the fluid-dynamic solution, through the imposition of the fluid velocity and pressure fields. The solver workflow is shown in Fig. 4.2.

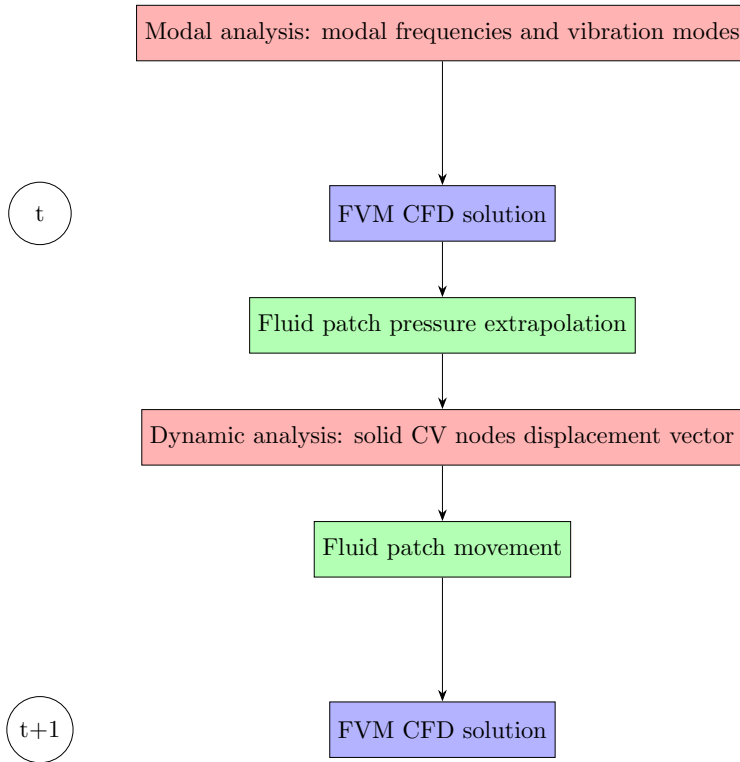


Figure 4.2: *modSFsiFoam* solver workflow.

The first command of the code is the declaration of the header *readPISOControls.H*, which reads the pressure-velocity coupling parameters and the mesh orthogonality conditions.

The *while (runTime.run ())* command does not require the timestep for each cycle iteration and the simulation is not interrupted before the end, with the exception of errors or user commands. *modSFsiFoam* calculates the timestep by referring to the Courant number (Co) value and exploits three libraries:

- *readTimeControls.H*, which reads the modifiable Courant number value, the maximum CO value and the maximum timestep specified by the user;

- *Courant.No.H*, which calculates the average value and the maximum value of Co along the domain;
- *setDeltaT.H*, which computes the timestep value required to satisfy the maximum CO value defined by the user.

setPressure.H

SetPressure.H is the header file, called by the *modSFsiFoam* main code, through whom the pressure field, obtained in the fluid dynamic solution, is passed to the solid domain.

For each timestep, the pressure acting on the fluid patch obtained through the fluid-dynamic solution is collected; the so obtained pressure field value acting on the control volume face is then multiplied for the area of the face itself, in order to calculate the force acting on each face center. Through these steps, the mechanical system forces vector is obtained.

With this aim, the starting point is the extrapolation of the position of the center points of each face of the patch interface between the fluid and the solid domain. The following step consists in the assembly of a matrix which contains, for each face center, the related pressure field value. In order to calculate the pressure value equal to the sum of the loads acting on points with the same abscissa and, therefore, acting on the same solid mesh nodes, it is necessary to reorder the same pressure values in function of the abscissa value of the loading point. This allows to describe the analysed system by referring to the lamina model subjected to n -loads acting above the midline.

solveSolid.H

SolveSolid.H is the header file, called by the *modSFsiFoam* main code, which is used to compute the displacement field of the structure due to the pressure field through the Modal Superposition Method.

With this aim, a modal analysis is done to obtain the pulsations of the structure for each vibration mode and, consequently, to collect the modal matrix of the mechanical system. The dynamic analysis of the system starts from the Duhamel convolution integral through the trapezoids method. For each vibrating mode, in each timestep, the above-mentioned P and Q

values are obtained, with the aim of calculate the v displacement vector in natural coordinates related to each vibrating mode. By considering the system modal matrix and the associated natural coordinates, the displacement field for each control volume can be obtained.

setMotion.H

SetMotion.H is the header file, called by the *modSFsiFoam* main code, through whom the nodes positions are modified by referring to the structure displacement due to the pressure field. With this aim, it has been developed an algorithm, through which the displacements are passed in the OpenFOAM order, i.e. the original order. The displacements obtained from the structural solution are passed to the fluid domain patch and the fluid mesh is then updated.

- *solveFluid.H*, taken from the previous solver, solves the fluid domain equations by using the PISO algorithm for the pressure-velocity coupling.

The operative scheme of *modSFsiFoam* solver is shown in the Fig. 4.3 below.

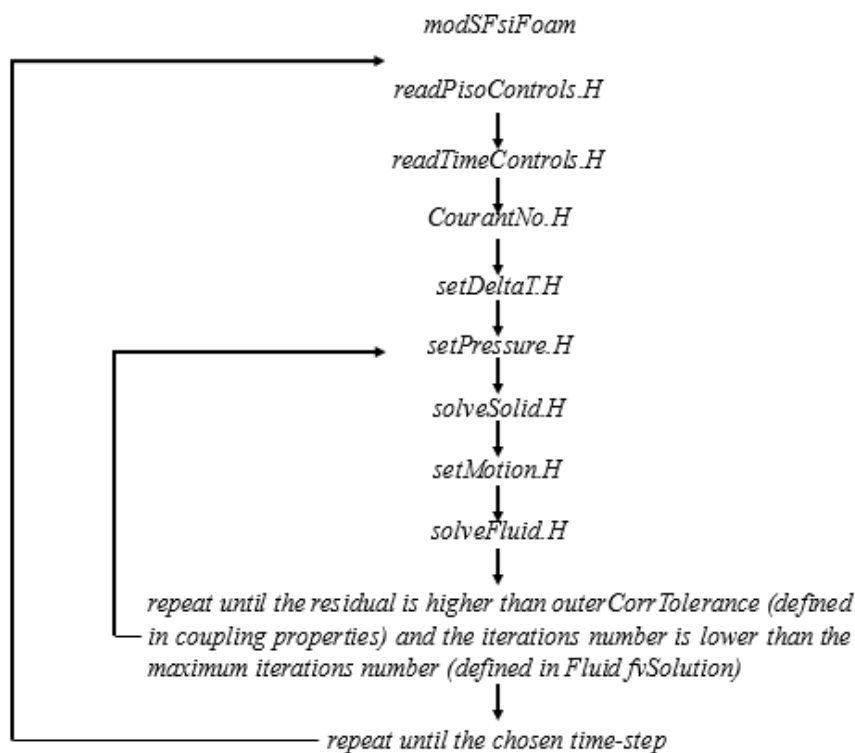


Figure 4.3: Operative scheme of *modSFsiFoam* solver.

Chapter 5

Applications

Fluid-structure interaction is a complex multi-physics issue with contiguous domains consisting generally on viscous fluid flow over elastic solid structures. The elastic structure deforms due to fluid action; mainly pressure and viscous stress [24]; the structure displacements, in turn, influence the motion field of the fluid itself, making the multiphysical phenomenon complex to study. Moreover, in the analysis of systems in which a fluid-structure interaction takes place, a proper focus on the characteristics of the constituent elements of the system (solid and fluid) is required. In particular, for a complete description of the interaction, the study of the coupling between fluid and structure is recommended.

The coupling between the two constituent elements is related to the ratio between the solid and the fluid density. Considering the value of this ratio, FSI systems can be classified as:

- *Weakly coupled*, if the ratio is greater than 50. In this case, the solid density is much higher than the fluid density and the velocity which characterizes the adaptation of the solid domain to the modifications of the fluid domain is negligible.
- *Strongly coupled*, if the ratio is lower than 10. In this kind of system, the solid and fluid densities have the same order of magnitude, which implies a strong influence of the deformation of the fluid domain on the deformation of the solid domain.

The physics of the problem is fundamental in view of the numerical solvers development, essential in many engineering fields. In fact, the character-

istics of the system, influence the choice of appropriate methods for the computational codes building. In fact, as presented in the previous chapters, two different numerical approaches can be distinguished: the monolithic approach and the partitioned approach.

In the monolithic approach, the complete non-linear system of fluid flow and solid displacement equations are solved simultaneously and are discretized in time and space in the same manner. This fully-coupled or direct approach is known to be highly robust and stable for very strong fluid-structure interaction including for example phase transformation in material processing, viscoplastic deformation, fracturing due to shocks or detonation. However, monolithic methods represent less modularity and require more coding than partitioned approach in which flow and structural equations are solved by using independent suitable algorithms and discretization methods, great strength of the partitioned approach, as outlined in this thesis.

Usually, an iterative algorithm must be used to handle the communication between both flow and structural solvers and to enforce the equilibrium on the fluid-structure interface. This means that the fluid flow and the structural deformation are solved successively within an iterating loop until that the difference between the flow and structural solutions, such as the interface displacement, is smaller than a given convergence criterion. This implies that, when the coupling between the fluid flow and the structural deformation is strong due to low solid stiffness or high fluid/structure density ratio, the iterations converge slowly and they even may diverge, depending on the relaxation parameter value choice.

Therefore, in an attempt to develop a solver for solving fluid-structural problems, as a first analysis we chose to focus on the study of weakly coupled problems. This is in order to test the current developed numerical code goodness, based on the methods chosen for fluid-structural solution. As second step, it has been chosen to focus on the extension to larger cases of strong coupling.

In particular, the satisfactory results obtained by the solver developed and described in the previous chapter, are reported in the following sections. In detail, the results obtained from the application of this code to two cases known in literature have been reported, in order to compare the output of the same with experimental and numerical cases validated

by the scientific community. Specifically, this transient solver has been tested on two well-known fluid-structure interaction problems. The applications have been published as: “*Fluid-structure interaction problem of a deformable lamina solved with an original OpenFOAM code*” [19] and “*Cylinder-lamina system fluid-structure interaction problem solved with an original OpenFOAM code*” [61].

5.1 Fluid-structure interaction problem of an inverted flag in a wind tunnel

With the aim of evaluating the effectiveness of the solver, it has been applied on a typical fluid-structure interaction case found in literature. The system is composed by an inverted flag, here modelled as a flexible lamina, clamped to a wind tunnel wall, in which an airflow is introduced in axial direction.

In the experimental test mentioned above, the polycarbonate plate was clamped between two bars at the trailing edge, leaving the leading edge free. The bars were held between two optical anglers allowing for precise alignment to the wind flow. Fig.5.1 depicts the inverted flag configuration at an angle of attack, θ . The orange block indicates the clamped trailing edge, h is the plate thickness and the plate height, w , is into the page. Plate position was recorded at 100 frames per second with a high speed camera positioned above a wind tunnel. The wind tunnel was composed of a 10 by 10 array of computer fans, with a 1.2 m by 1.2 m test section. Wind velocities were set between $2.5 - 8.5 \text{ ms}^{-1}$, and plate was allowed sufficient time to exit initial transient responses to the fluid flow. Different plate lengths, heights and angles of attack (θ) were used during experiments. Images were analyzed to determine the position of the leading edge as a function of time. This data allowed the amplitude to be calculated, as well as allowed the creation of stroboscopic images at several angles (θ) and non-dimensional flexural rigidities.

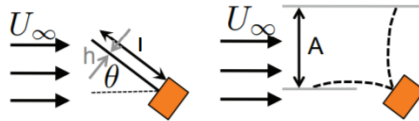


Figure 5.1: Inverted flag configuration

By considering experimental tests described in [13], for the numerical simulation it has been chosen an attack angle θ of 0 deg with respect to the wall.

Material characteristics (Young modulus E , Poisson ratio ν and density ρ_s) and geometrical parameters of the polycarbonate plate (length l , height w and thickness h) are given in the tables 5.1 and 5.2 below.

Geometrical parameters	Value [m]
Length l	0.255
Height w	0.51
Thickness h	0.8e-3

Table 5.1: *Structural geometrical parameters*

Parameters setting	Value
Solid density $\rho_s[kg/m^3]$	1200
Poisson coefficient ν_s	0.38
Young modulus E [GPa]	2.38

Table 5.2: *Material parameters setting*

The related flexural rigidity (R), defined in equation (5.1), is equal to 0.12.

$$R = \frac{Eh^3}{12(1 - \nu^2)} \quad (5.1)$$

For the air flow, instead, the values assumed for the density ρ_f and the kinematic viscosity ν_f are reported in the table 5.3.

Fluid flow parameters	Value
ρ_f [kg/m ³]	1.2
ν_f [m ² /s]	1.51e-5

Table 5.3: Fluid flow parameters

By referring to the dimensional wind velocity (Υ) as:

$$\Upsilon = \frac{\rho_f U^2 L^3}{R} \quad (5.2)$$

with the aim of comparing the numerical and the literature results, it has been chosen to work with a non-dimensional wind velocity of 1.5, which implies a flow velocity U of 3 m/s. The wind tunnel (figure (5.2)) in which experimental tests have been taken place has a 1.2 m length (L), 1.2 m height (H) and a 1.2 m width (W).

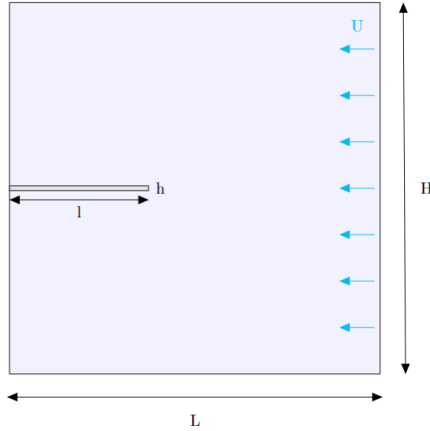


Figure 5.2: FSI case system

In reference to the initial conditions of the simulation, it has been imposed a zero *movingWallVelocity* value for the fluid patch with the aim of reproducing the analysed case. It has been set a *fixedVelocity* to the inlet, while the *zeroGradient* velocity condition has been chosen for the outlet. Concerning pressure initial conditions, it has been applied a null value to the outlet and *zeroGradient* condition for all the other patches.

Since the problem under investigation is bidimensional, the *empty* condition is applied to the *frontAndBack* faces, in order to avoid the solution computation along the *z* direction.

While the structural solution is calculated using the modal superposition algorithm, for the fluid-dynamic solution the FVM, called from the *solveFluid.H* header file, has been chosen. Therefore, the following numerical schemes for the discretization of the equations have been set in the OpenFOAM case, useful to the development of the simulation.

In detail, a Backward scheme has been chosen for the discretization of the time derivatives. As for the gradients instead, a linear Gauss scheme has been used, with a linear type interpolation between cell centers and face centers.

The same scheme is used for the Laplacians, for which it has been also adopted a correction method for the error introduced by the possible of the non-orthogonal mesh on the diffusive terms.

For the interpolations a linear method is adopted, while for the gradients normal to the surface a scheme with correction of non-orthogonality, for the same reason as the scheme adopted for the Laplacian terms, has been chosen. Finally, the schemes used for the discretization of the convective terms are of the Gauss linear type.

Mathematical terms	Numerical scheme
ddtSchemes	backward
gradSchemes	
grad(p)	Gauss linear
snGradCorr(U)	Gauss linear
grad(U)	Gauss linear
snGradCorr(p)	Gauss linear
snGradCorr(cellMotionU)	Gauss linear
divSchemes	
div(phi,U)	Gauss skewCorrected linear
laplacianSchemes	
laplacian(nu,U)	Gauss linear corrected
laplacian(diffusivity,cellMotionU)	Gauss linear corrected
interpolationSchemes	skewCorrected linear

Table 5.4: *Fluid-dynamic fvSchemes*

In order to evaluate simulation results, displacement values of the lamina free edge (last Control Volume node) have been extracted, with the aim of comparing vibrations trends and values.

The first 75ms of the experimental test have been simulated, because of the high computational burden of the process. In this time range, the lamina free edge, in according to the experimental reference values, oscillates around an equilibrium position by following a sinusoidal trend. Vibration peaks are in the order of $10^{-9}m$, which are comparable with the reference values (figure (5.3)).

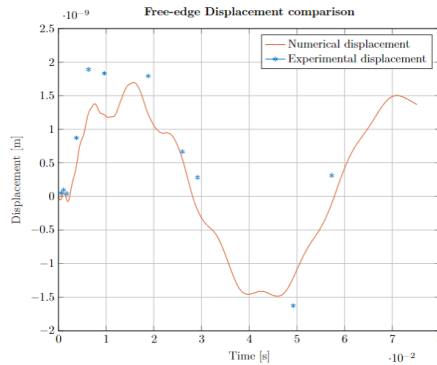


Figure 5.3: Comparison between simulated and experimental free-edge displacement.

By referring to the fluid domain, the velocity and pressure last timestep values are reported (figures (5.4) and (5.5)).

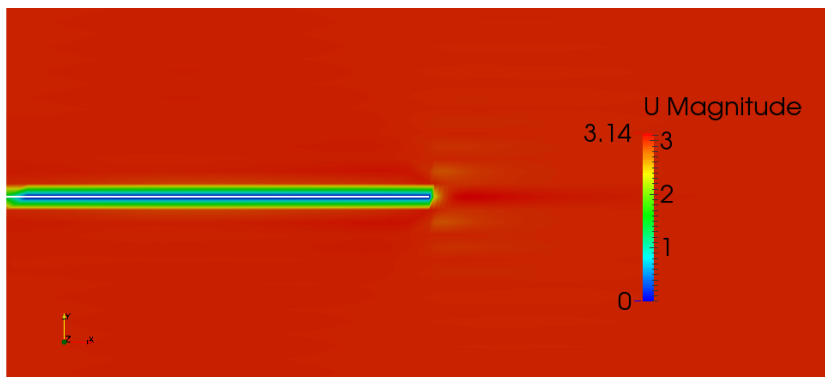


Figure 5.4: Air flow velocity in m/s.

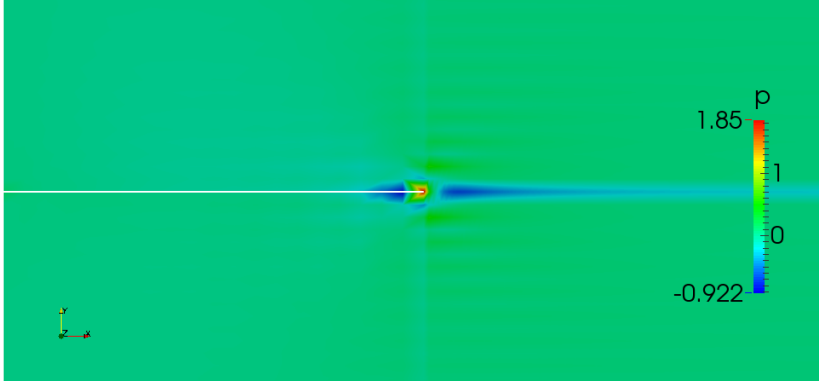


Figure 5.5: Air flow pressure in Pa.

As can be seen from the above results, the comparison between experimental case from scientific literature and numerical results obtained led to encouraging conclusions regarding the potential of the solver in FSI applications.

Referring to the good results obtained from the study of weakly coupled problems in order to test the goodness of the numerical code on the basis of the methods chosen for the fluid-dynamic and structural solutions, thus bypassing possible convergence problems due to strong coupling, it was chosen to test the developed solver on increasing complexity FSI cases. It was then decided to apply it to a case of strong interaction that reported as output also fluid-dynamic results, in order to evaluate the effectiveness in a complete way.

5.2 Cylinder-lamina system in a laminar incompressible channel flow

To evaluate the effectiveness of the solver, modSFsiFoam has been applied on a further typical fluid-structure interaction case [66][65]. The configuration consists of a laminar incompressible channel flow around a solid structure with an elastic part which results in self-induced oscillations. In particular, the domain is based on a 2D version of the benchmark above-mentioned. The circle center is located in a position of 0.2m from the left wall and 0.2m from the lower wall. Its radius is 0.05m. The elastic structure bar left end is fully attached to the fixed cylinder. The control point $A(t)$, fixed with the structure, is the end point along the midline. 1m/s velocity value is chosen at the left channel inflow. A zero-pressure value is selected instead for the right channel outflow. A no-slip condition is set for the fluid on the other boundary areas, such as top and bottom walls, circle, and fluid-structure interface.

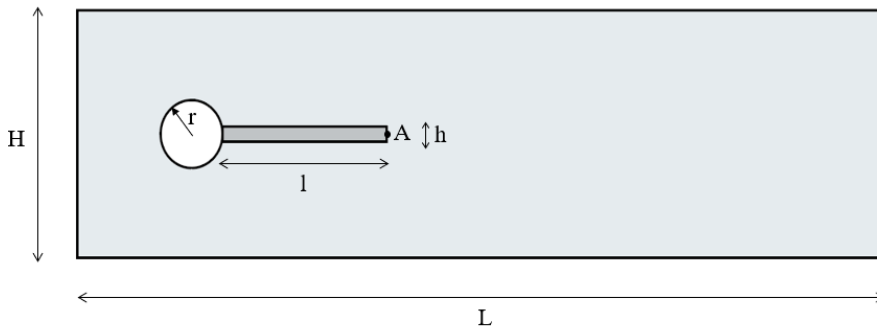


Figure 5.6: Computational domain.

The computational domain values are shown in the Table 5.5. The test case parameter setting are reported in Table 5.6.

Geometry parameters	Value [m]
Channel length L	2.5
Channel width H	0.41
Cylinder center position	(0.2,0.2)
Cylinder radius r	0.05
Elastic structure length l	0.35
Elastic structure thickness h	0.02
Reference point A (at $t = 0$)	(0.6,0.2)

Table 5.5: *Computational domain geometry values.*

Parameters setting	Value
Solid density ρ_s [$\frac{kg}{m^3}$]	10000
Poisson coefficient ν_s	0.4
Young modulus E [MPa]	1.4
Fluid density ρ_f [$\frac{kg}{m^3}$]	1000
Kinematic viscosity ν_f [$\frac{m^2}{s}$]	10^{-6}
Inflow velocity u_f [$\frac{m}{s}$]	1
Reynolds number Re	100

Table 5.6: *Test case parameters setting*

As a simplification, the lamina has been discretized by a single cell in the y-direction. For each time-step and each cell of the lamina, the pressure on the top fluid patch and on the bottom fluid patch, obtained through the fluid-dynamic solution, is extracted; the so obtained pressure field value acting on the control volume face allows to calculate the force acting on each face center. The mechanical system forces vector is so achieved.

For the fluid-dynamic part solution the discretization schemes for each term of the differential equation have been specified in *fvSchemes* case file. In particular, for the temporal derivatives a backward scheme has been used. For the gradient, a Gauss linear scheme has been set. For the laplacian term, a Gauss skewCorrected linear scheme has been chosen while for the interpolations a linear method is adopted. A detailed overview of these schemes is given in the table 5.7 below.

Mathematical terms	Numerical scheme
ddtSchemes	backward
gradSchemes	
grad(p)	Gauss linear
snGradCorr(U)	Gauss linear
grad(U)	Gauss linear
snGradCorr(p)	Gauss linear
snGradCorr(cellMotionU)	Gauss linear
divSchemes	
div(phi,U)	Gauss skewCorrected linear
laplacianSchemes	
laplacian(nu,U)	Gauss linear corrected
laplacian(diffusivity,cellMotionU)	Gauss linear corrected
interpolationSchemes	skewCorrected linear

Table 5.7: *Fluid-dynamic fvSchemes*

The figures below show the results obtained through the numerical simulation. In particular, in Figure 5.7 y-displacement values of the lamina-free edge are reported. The lamina control point, according to the reference values, oscillates around an equilibrium position by following a sinusoidal trend reaching the regime conditions. As can be noted from the graph, only the y-displacement is computed since the 1-DOF mathematical model has been used.

The Figure 5.8 shows instead the y-displacements extrapolated from the one-period oscillation trend after the regime conditions have been reached. The obtained results are in good agreement with Hron-Turek benchmark results. The discrepancy in the results can be explained by the simplifications made by the solver: For each control volume, 1 DOF has been evaluated. Moreover, for the structural solution, the first four vibration modes have been considered.

y-displacement parameters	Obtained value	Reference value	Percentage error
Maximum value [mm]	84.56	81.83	3.34 %
Minimum value [mm]	-79.60	-79.37	0.29 %
Oscillation frequency [Hz]	1.70	1.90	10.50 %

Table 5.8: *y-displacements comparison.*

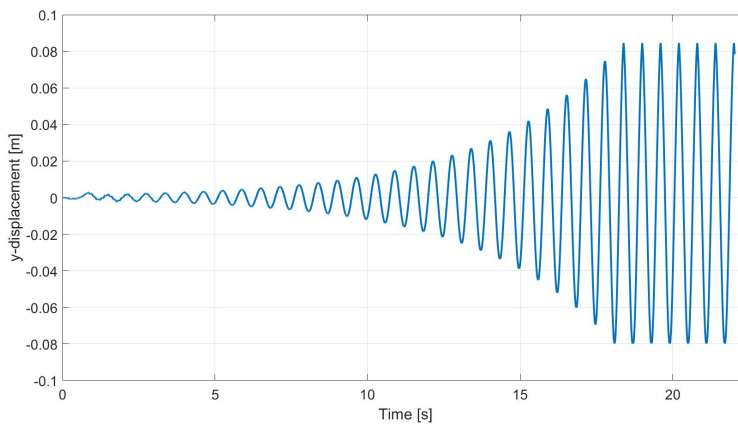


Figure 5.7: *y*-displacement of the lamina control point over time.

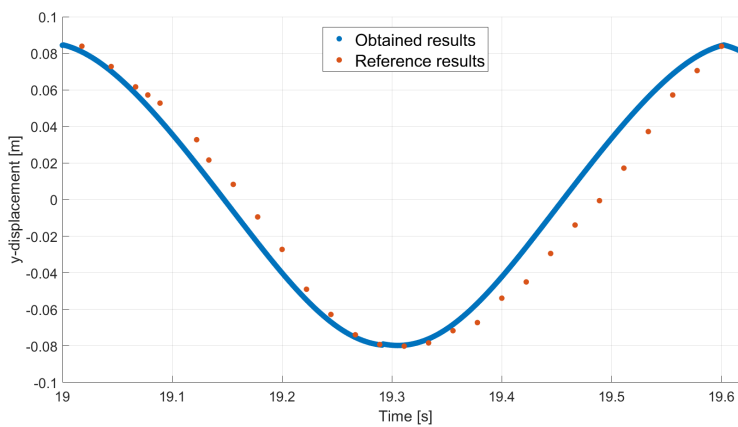


Figure 5.8: One-period *y*-displacement of the lamina control point.

For further comparison, the lift force acting on the cylinder-lamina system has been computed. The sinusoidal peaks and the periodic trend frequency value are shown in Table 5.9 below. Also, in this case, the obtained results can be considered comparable with the reference values.

Lift force parameters	Obtained value	Reference value	Percentage error
Maximum value $[N]$	235.30	235.08	0.09 %
Minimum value $[N]$	-216.70	-233.32	7.12 %
Oscillation frequency $[Hz]$	1.70	1.90	10.50 %

Table 5.9: *Lift force parameters comparison.*

The displacement obtained through OpenFOAM and that obtained through literature shows a phase difference that is constant over time. This could depend on whether only the first four modes were chosen to be used for the structural solution. As a future development, increasing the number of modes to investigate whether the solution obtained through developed solver converges to the one derived from literature can be considered. As mentioned in chapter 3 in reference to the previous work of [6], concerning damping, for these results a value of 0.05 was assumed as fraction of critical damping for the computation of the damped system pulsations.

From the above, good results were obtained from the solver outputs compared with an axially loaded cylinder-lamina system whose numerical results are known for both fluid-dynamics and structural fields, in terms of time-dependent characteristic quantities and corresponding plots, related to the sinusoidal trend of the free-edge displacement of the lamina and the lift force parameters.

As satisfactory results have been obtained, the complexity of the solver can be further increased to make it suitable for solving even more complex problems in terms of analyzed fluids, such as two-phase flow interacting with typical impact test structures. In fact, this FSI solver is applied as a transient solver for incompressible and laminar flow of Newtonian fluids with mesh motion.

Referring instead to the solid domain, the limitation is represented by the application on simple geometries, such as beams, in case of linear elasticity. In addition, 1DOF was considered for each control volume. As a future development, the solver could be applied to more complex solid geometries by numerically computing vibrational modes, bypassing the problem

of analytical determination. The effectiveness of the solver at solving FSI problems could be further increased by determining the stress and strain field of the solid structure from the displacement field of the structure. The results obtained will also be tested through future experimental campaigns.

Conclusions

The research activity here proposed aimed at giving an improvement to the scientific knowledge about Fluid-Structure Interaction (FSI). Specifically, the main focus was on multiphysics systems where a partially or totally deformable structure interacts with a fluid flow. The above-mentioned structure is subjected to a wide range of loads, especially dynamic loads, related to the presence of the flow; as a consequence, displacements and oscillations of the structure itself influence the fluid motion field. The purpose of this thesis work was the definition and validation of advanced methods for studying the behavior of FSI systems.

Generally, a wide range of laboratory tests allows a deep analysis of FSI systems for most engineering fields; the most common classification provide for a division of tests themselves into continuous flow tests, which are based on the velocity and the flow-rate of the fluid, and non-continuous flow tests. All these tests can be carried out through a single experimental setup known as water tunnel. Data acquisition can be focused on both fluid and structure behaviour through the use of sensors suited for direct measurements and of non-invasive techniques, such as image analysis techniques (PIV and PIBV), but also on one of two elements of the system.

However, for most FSI problems, analytical solutions to the model equations are impossible to obtain, whereas laboratory experiments are limited in scope; thus to investigate the fundamental physics involved in the complex interaction between fluids and solids, numerical simulations may be employed.

In this scenario, the activity carried out during the Ph.D. is placed. It has allowed to develop *ModSFsiFoam* (*MODalSuperpositionFsiFoam*), a partitioned numerical code suitable for solving FSI problems in the case of deformable bodies.

The solver implementation is made possible through the open source software OpenFOAM, which is developed for Linux operating systems and released under a GPL license with the C++ source code. This code can be modified by the user, who can contribute to its development by customizing and improving its features. OpenFOAM, which is mostly used for fluid dynamics and gas dynamics problems, can be used for a wide range of physical systems, such as the structures typical of solid mechanics.

The partitioned approach made it possible to use two different methods for finding fluid-dynamic and structural solutions, thus using the most suitable algorithms for both physical fields. In particular, the FVM has been used to solve the Navier-Stokes equations for an incompressible laminar flow of Newtonian fluid. A theoretical method, the modal superposition method, has been used instead for the structural solution, as a method able to provide a certain solution in according to solid-mechanics theory.

However, in a partitioned approach an iterative algorithm must be used to handle the communication between both flow and structural solvers and to enforce the equilibrium on the fluid-structure interface. This means that the fluid flow and the structural deformation are solved successively within an iterating loop until that the difference between the flow and structural solutions, such as the interface displacement, is smaller than a given convergence criterion. This implies that, when the coupling between the fluid flow and the structural deformation is strong due to low solid stiffness or high fluid/structure density ratio, the iterations converge slowly and they even may diverge. Therefore, as first analysis, in an attempt to develop a solver to resolve fluid-structure problems, it has been chosen to focus on the weakly coupled problems study, to test the actual numerical code developed goodness, on the basis of the methods chosen for the fluid-dynamic and structural solution, and to focus, as second step, on the extension to larger cases of strong coupling.

The solver was in fact applied to two cases known in the literature, of weak coupling and then strong coupling, in order to compare the output of the same with experimental and numerical cases validated by the scientific community.

Since satisfactory results have been obtained, the solver's complexity can be further increased to make it suitable for solving even more complex problems in terms of analyzed fluids, such as biphasic flux interacting with structures typical of impact tests. In fact, this FSI solver is applied as transient solver for incompressible, laminar flow of Newtonian fluids with mesh motion.

Referring to the solid domain instead, the limit is represented by the application on simple geometries, such as beams, in case of linear elasticity. Moreover, for each control volume 1DOF has been considered. As a future development, the solver could be applied to more complex solid geometries by calculating vibrational modes numerically, bypassing the problem of analytical determination.

Furthermore as a future work, the solver will be firstly tested through specific experimental campaigns, which will be carried out through an experimental setup, a water tunnel, designed and under construction at the Tuscia University laboratories.

List of Figures

1.1	One-dimensional example of Lagrangian, Eulerian and ALE mesh and particle motion [15].	12
1.2	The motion of the ALE computational mesh is independent of the material motion [15].	13
1.3	Graphical scheme of the monolithic approach. In this approach, the fluid domain equations and the solid domain equations are solved in each timestep in a single system. . .	25
1.4	Graphical scheme of the partitioned approach. In this approach, the fluid domain equations and the solid domain equations are solved separately one or more times in each timestep and the informations about the congruence are exchanged through the two regions interface.	27
2.1	Control Volume of FVM mesh.	32
2.2	Interpolation schemes.	36
2.3	Orthogonal and non-orthogonal mesh.	38
2.4	Correction methods for a non-orthogonal mesh.	39
2.5	Time integration methods.	41
2.6	PISO and SIMPLE alghoritms.	45
3.1	Graphical representation of the application example.	56
3.2	Linear elasticity functions for a cantilever beam.	56
3.3	Graphical representation of the first sub-scheme of the application example.	57
3.4	Graphical representation of the second sub-scheme of the application example.	58
3.5	Graphical representation of the third sub-scheme of the application example.	60

3.6	Graphical representation of the fourth sub-scheme of the application example.	61
3.7	Modeling methodology.	63
3.8	Excitation response for linear system.	64
3.9	(a) An nDOF system; (b) Free-body diagrams of the masses [55].	65
4.1	OpenFOAM structure	70
4.2	<i>modSFsiFoam</i> solver workflow.	75
4.3	Operative scheme of <i>modSFsiFoam</i> solver.	78
5.1	Inverted flag configuration	82
5.2	FSI case system	84
5.3	Comparison between simulated and experimental free-edge displacement.	87
5.4	Air flow velocity in m/s	87
5.5	Air flow pressure in Pa	88
5.6	Computational domain.	89
5.7	y-displacement of the lamina control point over time.	93
5.8	One-period y-displacement of the lamina control point.	93

List of Tables

5.1	Structural geometrical parameters	83
5.2	Material parameters setting	83
5.3	Fluid flow parameters	84
5.4	Fluid-dynamic fvSchemes	86
5.5	Computational domain geometry values.	90
5.6	Test case parameters setting	90
5.7	Fluid-dynamic fvSchemes	91
5.8	y-displacements comparison.	92
5.9	Lift force parameters comparison.	94

Bibliography

- [1] A., R.: *Meccanica e dinamica delle macchine*. 2019.
- [2] Badia, S., A. Quaini, and A. Quarteroni: *Modular vs. non-modular preconditioners for fluid–structure systems with large added-mass effect*. Computer Methods in Applied Mechanics and Engineering, 197(49-50):4216–4232, 2008.
- [3] Badia, S., A. Quaini, and A. Quarteroni: *Splitting methods based on algebraic factorization for fluid-structure interaction*. SIAM Journal on Scientific Computing, 30(4):1778–1805, 2008.
- [4] Basting, S., A. Quaini, S. Čanić, and R. Glowinski: *Extended ale method for fluid–structure interaction problems with large structural displacements*. Journal of Computational Physics, 331:312–336, 2017.
- [5] Bern, M.W. and P.E. Plassmann: *Mesh generation*. In *Handbook of Computational Geometry*, 2000.
- [6] Botis, M.: *Matlab program for the numerical solution of duhamel convolution integral*. 2012.
- [7] Bukač, M., S. Čanić, R. Glowinski, J. Tambača, and A. Quaini: *Fluid–structure interaction in blood flow capturing non-zero longitudinal structure displacement*. Journal of Computational Physics, 235:515–541, 2013.
- [8] Campbell, R. and E. Paterson: *Fluid–structure interaction analysis of flexible turbomachinery*. Journal of Fluids and Structures, 27(8):1376–1391, 2011.
- [9] Causin, P., J.F. Gerbeau, and F. Nobile: *Added-mass effect in the design of partitioned algorithms for fluid–structure problems*. Computer

- methods in applied mechanics and engineering, 194(42-44):4506–4527, 2005.
- [10] Cella, U., P. Della Vecchia, C. Groth, S. Porziani, A. Chiappa, F. Giorgetti, F. Nicolosi, and M.E. Biancolini: *Wind tunnel model design and aeroelastic measurements of the ribes wing*. Journal of Aerospace Engineering, 34(1):04020109, 2021.
- [11] Cetraro, G.: *High-fidelity flow simulations of electroactive membrane wings*. PhD thesis, University of Southampton, 2017.
- [12] Corradi Dell’Acqua, L.: *Meccanica delle strutture*. McGraw-Hill libri Italia., 2010.
- [13] Cossé, J., J. Sader, D. Kim, C.H. Cerdeira, and M. Gharib: *The effect of aspect ratio and angle of attack on the transition regions of the inverted flag instability*. In *Pressure Vessels and Piping Conference*, vol. 46018, p. V004T04A046. American Society of Mechanical Engineers, 2014.
- [14] Donea, J. and A. Huerta: *Finite element methods for flow problems*. John Wiley & Sons, 2003.
- [15] Donea, J., A. Huerta, J.P. Ponthot, and A. Rodriguez-Ferran: *Encyclopedia of computational mechanics vol. 1: Fundamentals., chapter 14: Arbitrary lagrangian-eulerian methods*, 2004.
- [16] Facci, A.L.: *Numerical simulation in openfoam*. 2015.
- [17] Facci, A.L., R. Panciroli, S. Ubertini, and M. Porfiri: *Assessment of piv-based analysis of water entry problems through synthetic numerical datasets*. Journal of Fluids and Structures, 55:484–500, 2015.
- [18] Faltinsen, O.M. and A.N. Timokha: *An inviscid analysis of the prandtl azimuthal mass transport during swirl-type sloshing*. Journal of Fluid Mechanics, 865:884–903, 2019.
- [19] Fanelli, P., C. Stefanini, A.L. Facci, and S. Ubertini: *Fluid-structure interaction problem of a deformable lamina solved with an original openfoam code*. Procedia Structural Integrity, 24:939–948, 2019.
- [20] Farhat, C., K.G. Van der Zee, and P. Geuzaine: *Provably second-order time-accurate loosely-coupled solution algorithms for transient*

- nonlinear computational aeroelasticity*. Computer methods in applied mechanics and engineering, 195(17-18):1973–2001, 2006.
- [21] Friedman, M.H., R. Krams, and K.B. Chandran: *Flow interactions with cells and tissues: cardiovascular flows and fluid–structure interactions*. Annals of biomedical engineering, 38(3):1178–1187, 2010.
- [22] Gerbeau, J.F. and M. Vidrascu: *A quasi-newton algorithm based on a reduced model for fluid-structure interaction problems in blood flows*. ESAIM: Mathematical Modelling and Numerical Analysis, 37(4):631–647, 2003.
- [23] Gerbeau, J.F., M. Vidrascu, and P. Frey: *Fluid–structure interaction in blood flows on geometries based on medical imaging*. Computers & Structures, 83(2-3):155–165, 2005.
- [24] Habchi, C., S. Russeil, D. Bougeard, J.L. Harion, T. Lemenand, A. Ghanem, D. Della Valle, and H. Peerhossaini: *Partitioned solver for strongly coupled fluid–structure interaction*. Computers & Fluids, 71:306–319, 2013.
- [25] Hak, M. Gad-el: *The water towing tank as an experimental facility*. Experiments in fluids, 5(5):289–297, 1987.
- [26] Heil, M., A.L. Hazel, and J. Boyle: *Solvers for large-displacement fluid-structure interaction problems: segregated versus monolithic approaches*. Computational Mechanics, 43(1):91–101, 2008.
- [27] Hou, G., J. Wang, and A. Layton: *Numerical methods for fluid-structure interaction-a review*. Communications in Computational Physics, 12(2):337–377, 2012.
- [28] Hron, J. and S. Turek: *A monolithic fem/multigrid solver for an ale formulation of fluid-structure interaction with applications in biomechanics*. In *Fluid-structure interaction*, pp. 146–170. Springer, 2006.
- [29] Jasak, H.: *Error analysis and estimation for the finite volume method with applications to fluid flows*. 1996.
- [30] Jasak, H.: *Multi-physics simulations in continuum mechanics*. In *Proceedings of 5th International Conference of Croatian Society of Mechanics, Trogir*, page, 2006.

- [31] Jasak, H.: *Updated lagrangian finite volume solver for large deformation dynamic response of elastic body*. Transactions of FAMENA, 31(1):55, 2007.
- [32] Jasak, H., A. Jemcov, Z. Tukovic, *et al.*: *Openfoam: A c++ library for complex physics simulations*. In *International workshop on coupled methods in numerical dynamics*, vol. 1000, pp. 1–20. IUC Dubrovnik, Croatia, 2007.
- [33] Jasak, H. and H. Weller: *Application of the finite volume method and unstructured meshes to linear elasticity*. International journal for numerical methods in engineering, 48(2):267–287, 2000.
- [34] Kanyanta, V., N. Quinn, S. Kelly, A. Ivankovic, and A. Karac: *Fluid-structure interaction (fsi) in bioengineering*. In *2nd Open FOAM workshop*. Dublin: University College Dublin, 2007.
- [35] Korobenko, A., M.C. Hsu, I. Akkerman, J. Tippmann, and Y. Bazilevs: *Structural mechanics modeling and fsi simulation of wind turbines*. Mathematical Models and Methods in Applied Sciences, 23(02):249–272, 2013.
- [36] Küttler, U. and W.A. Wall: *Fixed-point fluid–structure interaction solvers with dynamic relaxation*. Computational Mechanics, 43(1):61–72, 2008.
- [37] Kuzmin, D.: *A guide to numerical methods for transport equations*. University Erlangen-Nuremberg, 2010.
- [38] Lakshminarayan, V.K. and C. Farhat: *Nonlinear aeroelastic analysis of highly flexible flapping wings using an ale formulation of embedded boundary method*. In *52nd Aerospace Sciences Meeting*, p. 0221, 2014.
- [39] Le Tallec, P. and J. Mouro: *Fluid structure interaction with large structural displacements*. Computer methods in applied mechanics and engineering, 190(24-25):3039–3067, 2001.
- [40] Lee, S.H., S. Kang, N. Hur, and S.K. Jeong: *A fluid-structure interaction analysis on hemodynamics in carotid artery based on patient-specific clinical data*. Journal of mechanical science and technology, 26(12):3821–3831, 2012.

- [41] Ma, Q., V. Sriram, and S. Yan: *Fluid–structure interactions in marine engineering*. Encyclopedia of Maritime and Offshore Engineering, pp. 1–11, 2017.
- [42] MacPhee, D.W. and M. Luhar: *Asymmetric buckling of curved flapping wings*. In *2018 Fluid Dynamics Conference*, p. 3079, 2018.
- [43] Meirovitch, L.: *Fundamentals of vibrations*. Waveland Press, 2010.
- [44] Michler, C., S. Hulshoff, E. Van Brummelen, and R. De Borst: *A monolithic approach to fluid–structure interaction*. Computers & fluids, 33(5-6):839–848, 2004.
- [45] Moin, P.: *Fundamentals of engineering numerical analysis*. Cambridge University Press, 2010.
- [46] Newman, J.N.: *Marine hydrodynamics*. The MIT press, 2018.
- [47] Nonino, M., F. Ballarin, and G. Rozza: *A monolithic and a partitioned, reduced basis method for fluid–structure interaction problems*. Fluids, 6(6):229, 2021.
- [48] Open, C.: *Openfoam programmer’s guide*. OpenFOAM Foundation, 2(0), 2011.
- [49] OpenFOAM, F.: *User guide*. 2017.
- [50] Panciroli, R. and M. Porfiri: *Evaluation of the pressure field on a rigid body entering a quiescent fluid through particle image velocimetry*. Experiments in fluids, 54(12):1–13, 2013.
- [51] Panciroli, R., A. Shams, and M. Porfiri: *Experiments on the water entry of curved wedges: high speed imaging and particle image velocimetry*. Ocean Engineering, 94:213–222, 2015.
- [52] Patankar, S.: *Numerical heat transfer and fluid flow*. CRC press, 1980.
- [53] Patel, M., J.B. Mouillet, D. Burkhalter, A. Robert, and T. Schwertzig: *Ditching simulation of air and space vehicles*. In *13th European Conference on Spacecraft Structures, Materials & Environmental Testing*, vol. 727, p. 202, 2014.

- [54] Piperno, S. and C. Farhat: *Design of efficient partitioned procedures for the transient solution of aeroelastic problems*. Revue Européenne des Eléments Finis, 9(6-7):655–680, 2000.
- [55] Rao, S.S.: *Vibration of continuous systems*. John Wiley and Sons, 2019.
- [56] Ringsberg, J.W., S.E. Heggelund, P. Lara, B.S. Jang, and S.E. Hirdaris: *Structural response analysis of slamming impact on free fall lifeboats*. Marine Structures, 54:112–126, 2017.
- [57] Salvini, P. and F. Vivio: *Dynamic reduction strategies to extend modal analysis approach at higher frequencies*. Finite Elements in Analysis and Design, 43(11):931–940, 2007, ISSN 0168-874X.
- [58] Seddon, C. and M. Moatamedi: *Review of water entry with applications to aerospace structures*. International Journal of Impact Engineering, 32(7):1045–1067, 2006.
- [59] Singh, K., S. Michelin, and E. De Langre: *The effect of non-uniform damping on flutter in axial flow and energy-harvesting strategies*. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, 468(2147):3620–3635, 2012.
- [60] Spina, L.: *Analisi cfd con interazione fluido-struttura per la caratterizzazione acustica di silenziatori per motori a combustione interna*. 2014.
- [61] Stefanini, C., F. Giorgetti, A. Mercuri, A. Facci, and P. Fanelli: *Cylinder-lamina system fluid–structure interaction problem solved with an original openfoam code*. Journal of Computational Science, 54:101420, 2021.
- [62] Tang, T., F. Vesting, and J. Andric: *Implementation of solid body stress analysis in openfoam*. Report for the course CFD with OpenSource software held at Chalmers University of Technology, 2012.
- [63] Thielicke, W. and E. Stamhuis: *Pivlab—towards user-friendly, affordable and accurate digital particle image velocimetry in matlab*. Journal of open research software, 2(1), 2014.
- [64] Todd, F.: *The fundamentals of ship model testing*. SNAME, 1951.

- [65] Turek, S., H.J.R.M.W.H..S.M.: *Numerical benchmarking of fluid-structure interaction: A comparison of different discretization and solution approaches*. Fluid Structure Interaction II, 331:413–424, 2011.
- [66] Turek, S. and J. Hron: *Proposal for numerical benchmarking of fluid-structure interaction between an elastic object and laminar incompressible flow*. In *Fluid-structure interaction*, pp. 371–385. Springer, 2006.
- [67] Wang, W. and Y. Yan: *Strongly coupling of partitioned fluid–solid interaction solvers using reduced-order models*. Applied Mathematical Modelling, 34(12):3817–3830, 2010.
- [68] Wang, Y., A. Quaini, and S. Čanić: *A higher-order discontinuous galerkin/arbitrary lagrangian eulerian partitioned approach to solving fluid–structure interaction problems with incompressible, viscous fluids and elastic structures*. Journal of Scientific Computing, 76(1):481–520, 2018.
- [69] Weisstein, E.W.: *Divergence theorem*. 2002.
- [70] Yoon, K., J. Kim, K. Lee, Y. Lee, and H. Kim: *Control rod drop analysis by finite element method using fluid–structure interaction for a pressurized water reactor power plant*. Nuclear Engineering and Design, 239(10):1857–1861, 2009.