Escola de Camins
Escola Tècnica Superior d'Enginyeria de Camins, Canals i Ports
UPC BARCELONATECH

MASTER'S DEGREE THESIS

Validation of two embedded computational fluid dynamics formulations in Kratos Multiphysics

Thesis written by:
Chiluba Isaiah Nsofu

Tutored by:
Dr. Riccardo Rossi
Rubén Zorrilla Martinez M.Sc.

Master in:
Erasmus Mundus MSc Computational Mechanics

Barcelona, June 15, 2018

Department of Civil & Environmental Engineering

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Validation of two embedded computational fluid dynamics formulations in Kratos Multiphysics

Submitted in the partial fulfillment

for the Erasmus Mundus Master

in

Computational Mechanics

Centre Internacional de Métodes Numérics a l'Enginyeria (CIMNE)

By:

Chiluba Isaiah Nsofu

M.Sc Computational Mechanics

Universitat Politécnica de Catalunya,

BarcelonaTech (UPC)

Supervised by:

Dr. Riccardo Rossi

Rubén Zorrilla Martínez M.Sc.

Universitat Politécnica de Catalunya,

BarcelonaTech (UPC), CIMNE

June 15, 2018

Swansea University
Prifysgol Abertawe

Abstract

One of the most difficult tasks in the numerical simulation of fluid flow is the generation of a grid around the object being modeled. Two types of grids are most commonly used in computational fluid dynamics: body-fitted and embedded grids. Therefore the work carried out in this thesis deals with the validation of two computational fluid dynamics formulation resulting from the use of these grids. These formulations commonly referred to in this work as symbolic and embedded formulations, are implemented in the fluid dynamics application of the multiphysics solver Kratos. The main advantage of embedded formulations in front of body-fitted ones is to get rid of the Arbitrary Lagrangian Eulerian mesh problem and its complexities. Besides, they suppress the model clean up tasks, since the problem is not solved for the real geometry but for a distance function that represents it. To have a firm fundamental background the two formulation, the frameworks on which these formulations have been developed are presented. These includes the basics of computational fluid dynamics, variational multiscale methods and the finite element method. The validation of this formulations takes into consideration benchmark problems involving watertight bodies. The purpose of the validation process is to solve 2D and 3D cases to prove the correctness and also to check the formulations capabilities in some complex examples formulations. The results obtained from benchmark tests have shown that both formulations are capable of giving the expected results.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisors Rubén Zorrilla Martínez MSc. and Dr Riccardo Rossi. I am so grateful for giving me a chance of doing my masters thesis under your supervision. Your patience, motivation, enthusiasm, and immense knowledge provided me a great platform to complete the thesis. I am also indebted to the European Union and the entire CIMNE staff for awarding me the Erasmus Mundus scholarship, I will forever be grateful for this amazing opportunity. I also express my gratitude to all my friends especially my colleagues Ahmed S, Ankit J, Nikhil D, Nikhil F and Sunag R.

This research work would not have been possible without the amazing professors who have taught me for the past two years. For that, I am so grateful to all the faculty members of both Swansea University, Wales and Universitat Politécnica de Catalunya, Barcelona. Many thanks to Ms. Lelia Zienloka, master Secretariat of the programme for being very supportive even before the beginning of the programme. Finally, I would like to thank my siblings and my parents Sandie B Nsofu and Violet C Chabatama for their continued support and encouragement.

Contents

Chapter 1

Introduction

Computational fluid dynamics (CFD) is the use of applied mathematics, physics and computational software to simulate how a gas or liquid flows, as well as how the gas or liquid affects objects as it flows past. A common approach in CFD is to solve the Navier-Stokes equations. These equations describe how the velocity, pressure, temperature, and density of a moving fluid are related. One of the most difficult tasks in the numerical simulation of fluid flow is the generation of a grid around the object being modeled [1]. This has always involved a large amount of time-consuming user interaction. If the object is moving and/or deforming, then the necessary regridding is an even greater and computationally expensive problem. After regridding, the solution variables must be transferred (i.e. interpolated) from the old to the new grid. This interpolation is a source of error and may also destroy the conservative properties of numerical schemes [19]. When dealing with CFD simulations two types of grids are most commonly used: body-fitted grids and embedded grids [27]. In the case of body-fitted grids, the external mesh faces match up with the body surfaces and external boundary faces of the domain. On the other hand in the embedded approach, the bodies are immersed inside a large mesh, most of the time a Cartesian mesh, and special treatments of the elements close to the body surfaces are performed. When considering general cases of moving or deforming surfaces along with topological changes, both approaches have complementary strengths and weaknesses. The two approaches are illustrated in the figure below.

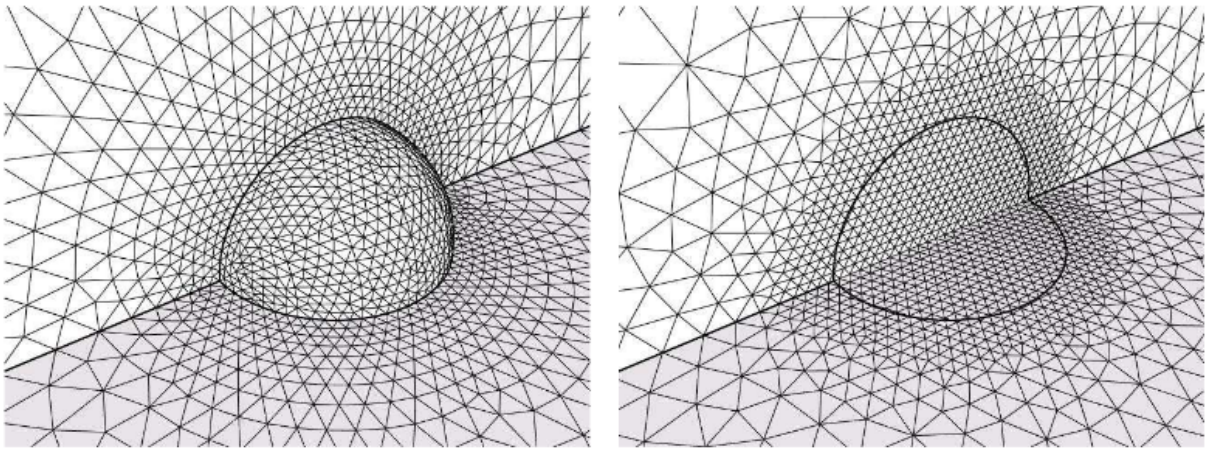


Figure 1.1: Body-fitted and Embedded formulations of a sphere mesh (Adopted from [19])

The objective of this thesis is the validation of one body-fitted (symbolic) and one (the no-slip) embedded computational fluid dynamics formulations in the multi-physics solver Kratos. Each of these formulations solves problems that can be described using Partial Differential Equations (PDE). These PDE's are capable of describing the physics of a problem which can then be discretized to get the algebraic system of equations to be solved. The process of getting the systems of equations requires the use of mathematical computations such as differentiation of the resulting PDE, hence for a given formulation this process can be done manually as it in the case of the embedded formulation or it can be done automatically as in the case of the symbolic formulation. For the symbolic case the process involves the use of techniques such as Automatic Differentiation (AD) which can be used together with a the Computer Algebra System (CAS) such as the Python module *Sympy*. Therefore, in this work these formulations will be validated against the results of the already validated body-fitted formulation (where the differentiation is done manually) together with the wind tunnel and/or experimental results. All the formulations in this work have been developed using the variational multiscale method (VMS) by the Kratos research team at the International Center for Numerical Methods in Engineering (CIMNE).

Based on all the aforementioned objectives, the present work is organized as follows: In the first section (Chapter 2) the fundamental theory of computational fluid dynamics is presented. This includes the derivation of the governing Navier-stokes equations and the use of the finite element method. To give a comprehensive review of the embedded formulations, the derivation and implementation of the embedded solver is also presented within this section.

The second section (Chapter 3) deals with methodology where all the softwares packages that were used in this work are described. This includes the opensource software Kratos Multi-physics and pre and post processing software GiD. As an overview, the continuous distance function that is used for the purpose of tracking the interface in the scope of the embedded formulation is also presented in this section. Further more an overview of the of parallelization techniques used in kratos is also presented.

Having discussed the fundamental concepts of both the body-fitted and embedded formulations the third section(Chapter 4) deals with the validation of the new body-fitted(symbolic) as well as the embedded formulation. Here the results of four benchmark problems in CFD are presented and analyzed. The first three problems are as a result of the work done under the Priority Research Programme [26] and the fourth one is the well known Silsoe cube problem [24]. Many papers have been published on these benchmark problems which therefore opens up the possibility to validate different data like drag , lift, pressure and velocity profiles.

Finally, in the fourth section (Chapter 5) we summarize the work and present its main conclusions, as well as proposing future lines of research.

Chapter 2

Mathematical and Computational Fluid Dynamics Background

2.1 Continuum Mechanics Basics

The continuum approach enables the neglect of the molecular structure of solids and fluids rather regarding them as being without empty spaces, i.e. there are always neighbor particles. Mathematically, one is left with continuous functions valid at all points in space and time. This applies to the derivatives of these functions as well. Therefore throughout this work we will consider the continuum approach.

2.1.1 Description of motion in Continuum Mechanics

The use of numerical simulations to solve complex problems in both fluid dynamics and nonlinear solid mechanics often requires coping with strong distortions of the continuum under consideration while allowing for a clear delineation of free surfaces and fluidfluid, solidsolid, or fluidstructure interfaces [12]. Hence it is fundamentally important to consider an appropriate choice for the kinematic description of the continuum. In fact, such a choice determines the relationship between the deforming continuum and the finite grid or mesh of computing zones, and thus conditions the ability of the numerical method to deal with large distortions and provide an accurate resolution of material interfaces. The most established approaches for describing motion in continuum mechanics are the Lagrangian, Eulerian and Arbitrary Lagrangian Eulerian formulations(ALE). The Lagrangian approach is mostly used to describe solids while the Eulerian approach used to describe fluids. The ALE formulations is widely used in fluid structure interactions (FSI). All three different approaches will be introduced in this chapter although more information on this topic can be found in the classical textbook given in [12].

Spatial and Material coordinates

Spatial point: A point which is fixed in space (its position does not change). We denote spatial coordinates $\mathbf{x}$ as:

$$\mathbf{x} = x_1\hat{\mathbf{e}}_1 + x_2\hat{\mathbf{e}}_2 + x_3\hat{\mathbf{e}}_3 \tag{2.1}$$

where $\mathbf{x}$ is a vector and $\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3$ are unit vectors.

Material point: A particle. It can occupy several positions (spatial points) in time. We identify a particle by its position at the initial configuration t_0 :

$$\mathbf{X} = X_1\hat{\mathbf{e}}_\mathbf{1} + X_2\hat{\mathbf{e}}_\mathbf{2} + X_3\hat{\mathbf{e}}_\mathbf{2} \tag{2.2}$$

Equation of movement: This equation allows us to identify, for a given particle $\mathbf{X}$, its spatial position $\mathbf{x}$ for each time instant t:

$$\mathbf{x} = \varphi(\mathbf{X}, t) = \mathbf{x}(\mathbf{X}, t) \tag{2.3}$$

Taking a property with values varying in space and time, we can choose to describe it as a function of spatial as $\gamma(\mathbf{x}, t)$ or material coordinates as $\Gamma(\mathbf{X}, t)$. Therefore, the two descriptions are related through the equations of movement as:

$$\mathbf{X} \ = \varphi^{-1}(\mathbf{x}, t) = \mathbf{X}(\mathbf{x}, t) \tag{2.4}$$

$$\Gamma(\mathbf{X}, t) = \Gamma(\mathbf{X}(\mathbf{x}, t)) = \gamma(\mathbf{x}, t) \tag{2.5}$$

Material Derivative

Since we have several description of the continuum (material, spatial), there are several definitions for the temporal derivatives. The Local time derivative which is the sensitivity of a property with respect to time in a point which is fixed in space. Using the spatial description of a property, the local time derivative is defined as

$$\frac{\partial \gamma(\mathbf{x}, t)}{\partial t} \tag{2.6}$$

On the other hand, the material time derivative is defined as the sensitivity of a property with respect to time following a specific particle (material point) of the continuum. This property is expressed using the material description as:

$$\frac{D}{Dt}\Gamma = \frac{\partial \Gamma(\mathbf{X}, t)}{\partial t} \tag{2.7}$$

Note that the two properties (2.6) and (2.7) are different and have different meanings. In the first one we are focusing in the variation in time of a variable at a fixed point, in the second one, we move around the domain following a particle. By replacing (2.5) into (2.7) and performing the expansion we get;

$$\frac{D}{Dt}\gamma(\mathbf{x}(\mathbf{X}, t)) = \frac{\partial \gamma(\mathbf{x}, t)}{\partial t} + \frac{\partial \gamma(\mathbf{x}, t)}{\partial x_i}\frac{x_i}{\partial t} = \frac{\partial \gamma(\mathbf{x}, t)}{\partial t} + \frac{\partial \gamma}{\partial \mathbf{x}}\frac{\mathbf{x}}{\partial t} = \frac{\partial \gamma(\mathbf{x}, t)}{\partial t} + v \cdot \nabla\gamma \tag{2.8}$$

The above equation represents the material, local and convective derivatives where velocity is defined as the derivative of the equations of movement with respect to time:

$$\mathbf{v} = \frac{\partial \mathbf{x}(\mathbf{X}, t)}{\partial t} \tag{2.9}$$

2.1.2 Lagrangian approach

The basic description of motion using the Lagrangian frameworks begins with the consideration of the referential description. Within the referential description the motion is described with respect to a

reference configuration in which a certain particle X of the continuum is located on the position $\mathbf{X}$ at time t.Therefore, if at time $t{=}o$ the reference configuration coincides with the initial configuration we refer to that as Lagrangian description. Usually in solid mechanics the unstressed state is chosen as the initial configuration. One can notice that using the Lagrangian framework we can keep track of the motion concerning a certain material particle by recording the position $\mathbf{x}$ of the particle X at a time t linking to its material coordinates to the spatial coordinates using a function φ as expressed in (2.3)

$$\mathbf{x} = \varphi(\mathbf{X}, t)$$

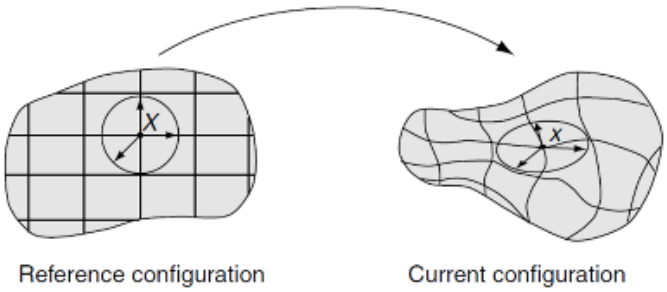


Figure 2.1: Lagrangian framework - The computational grid follows the material particles in the course of their motion (Adopted from [13]).

The figure (2.1) shows that within this framework the nodes in the discretized domain are permanently connected to their corresponding material particles at any point in time. This becomes a problem when carry out simulations such as those in the field of explicit dynamics where there are large deformations. Using this method for that kind of work results in large distortions of the mesh hence the method can fail as the distorted elements are directly connected with material particles. where the motion is described with respect to a reference configuration.

2.1.3 Eulerian approach

Instead of following an individual particle and deriving the needed equations based on the motion of that particle we can fix a control volume in space and see how the flow field is changing overtime by introducing field variables. Doing leads to another approach called the Eulerian description. In this frame of reference, we use local + convective to express the material derivative at a point fixed in space. Within this framework there is no connection of the spatial coordinates and the material coordinates hence a field variable such as velocity can be described using field function $\mathbf{g}$ such as

$$\mathbf{u} = \mathbf{g}(\mathbf{g}, t) \tag{2.10}$$

Using the material derivative we can link both the Eulerian and Lagrangian descriptions. This is demonstrated by taking pressure for example as the intensive property which leads to;

$$\frac{DP}{Dt} = \frac{dP}{dt} + (\mathbf{v} \cdot \nabla)P \tag{2.11}$$

where the first part on right hand side (R.H.S) describes the local rate of the change which is often called the local term and the second part is the convective term which describes the rate of change of the

particle when it moves to a region with a different pressure or velocity.

Table 2.1: Comparison of Lagrangian and Eulerian descriptions

Lagrangian description	Eulerian description
Mostly used in structure mechanics, free surface flow and simulations incorporating moving interfaces between different materials	Mostly used fluid mechanics for example in the simulation of vortices.
Allows to keep track of the material behavior and to move from the current configuration to the initial configuration of any material particle.	Does not permit to conclude from the current configuration to the initial configuration and the material coordinates X.
Following the motion of the particles might lead to excessive distortions of the finite elements when no remeshing is applied what in turn can cause numerical problems during simulation.	Due to the fixed computational mesh there are no distortions of finite elements such that large motion and deformation in the continuum can be analyzed.

2.1.4 Arbitrary Lagrangian Eulerian

The ALE method combines the advantages of both the Lagrangian and Eulerian methods as shown in table 2.1. In this method the computational system is not a prior fixed in space as it in the case of when using the Eulerian method neither is it attached to material like in the Lagrangian-based formulations. Thus, in the ALE description, the nodes of the computational mesh may be moved with the continuum in normal Lagrangian fashion, or be held fixed in Eulerian manner, or be moved in some arbitrarily specified way to give a continuous rezoning capability. Therefore, when using ALE we can alleviate many of the drawbacks faced by both the Lagrangian-based and Eulerian-based formulations.In the ALE description of motion, neither the material configuration R_X nor the spatial configuration R_x is taken as the reference. Thus as shown in Figure 2.2 , a third domain is needed the referential configuration R_χ where reference coordinates χ are introduced to identify the grid points. A more detailed explanation of this description can be found in the research paper by Donea [13] who first applied to this method to FEM. To present an overview of this approach we introducing the mesh movement:

$$\mathbf{x} = \boldsymbol{\phi}(\boldsymbol{\chi}, t) = \mathbf{x}(\boldsymbol{\chi}, t) \tag{2.12}$$

The derivative of this equation defines the mesh velocity which is used to trace the movement of the mesh as follows.

$$\mathbf{v}_{Mesh} = \frac{\partial \mathbf{x}(\boldsymbol{\chi}, t)}{\partial t} \tag{2.13}$$

The mapping between ALE domain and the material domain is

$$\boldsymbol{X} = \boldsymbol{\Psi}(\boldsymbol{\chi}, t) = \boldsymbol{X}(\boldsymbol{\chi}, t) \tag{2.14}$$

with the inverse mapping

$$\boldsymbol{\chi} = \boldsymbol{\Psi}^{-1}(\boldsymbol{X}, t) = \boldsymbol{\chi}(\boldsymbol{X}, t) \tag{2.15}$$

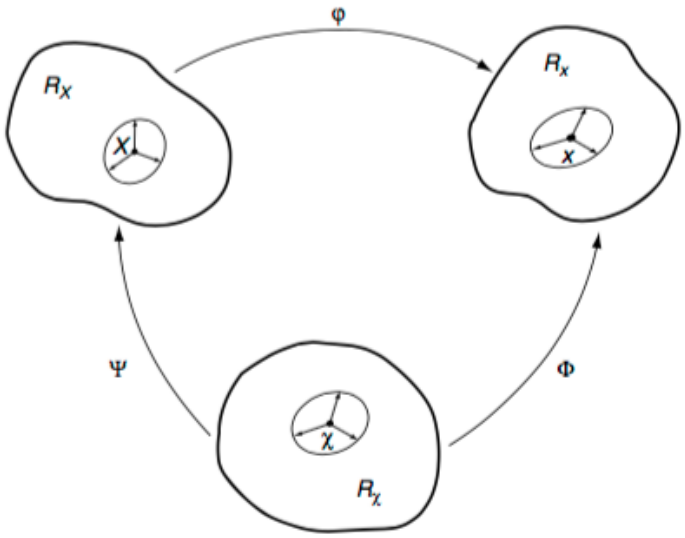


Figure 2.2: The motion of the ALE computational mesh is independent of the material motion (Adopted from [13]).

This inverse mapping transforms the material coordinates into coordinates in the reference system. Using this mapping we define velocity as

$$\mathbf{w} = \frac{\partial \boldsymbol{\chi}(\boldsymbol{X}, t)}{\partial t} = \frac{\partial \boldsymbol{\Psi}^{-1}(\boldsymbol{X}, t)}{\partial t} \tag{2.16}$$

The definition of any property γ in ALE takes the form of

$$\gamma_{ALE}(\boldsymbol{\chi}(\boldsymbol{X}, t), t) \tag{2.17}$$

and its material derivative is hence defined as

$$\frac{d}{dt}\gamma_{ALE}(\boldsymbol{\chi}(\boldsymbol{X}, t)) = \frac{\partial \Gamma(\boldsymbol{X}, t)}{\partial t} \tag{2.18}$$

which is further expanded as

$$\frac{d}{dt}\gamma_{ALE}(\boldsymbol{\chi}(\boldsymbol{X}, t)) = \frac{\partial \gamma_{ALE}(\boldsymbol{\chi}, t)}{\partial t} + \frac{\partial \gamma_{ALE}}{\partial \boldsymbol{\chi}} \cdot \boldsymbol{w} = \frac{\partial \gamma_{ALE}(\boldsymbol{\chi}, t)}{\partial t} + \frac{\partial \gamma}{\partial\, \mathbf{x}} \frac{\partial\, \mathbf{x}}{\partial \boldsymbol{\chi}} \cdot \boldsymbol{w} \tag{2.19}$$

$$= \frac{\partial \gamma_{ALE}(\boldsymbol{\chi}, t)}{\partial t} + (\mathbf{v} - \mathbf{v}_{Mesh}) \cdot \nabla \gamma(\mathbf{x}, t) \tag{2.20}$$

From this final form in Eq. (2.20) we see that the first terms is evaluated at the mesh points. The second term is evaluated at the spatial coordinates.

$$(\mathbf{v} - \mathbf{v}_{Mesh}) \cdot \nabla \gamma(\mathbf{x}, t) = \mathbf{c} \cdot \gamma(\mathbf{x}, t) \tag{2.21}$$

2.1.5 Strain rate and spin tensor

In fluids dynamics, the analysis of the relative motion of neighboring particles is done through the use of the rate of strain and rate of rotation. This is similar to what is done in solid mechanics where the strain and rotation of a solid are considered when finding the deformation of an elastic solid. A key variable in

the characterization of the fluid is the what is known as velocity gradient. This is a second order tensor which is defined in Cartesian coordinate system as follows;

$$\nabla \boldsymbol{u} = \begin{bmatrix} \frac{\partial u_1}{\partial x_1} & \frac{\partial u_1}{\partial x_2} & \frac{\partial u_1}{\partial x_3} \\ \frac{\partial u_2}{\partial x_1} & \frac{\partial u_2}{\partial x_2} & \frac{\partial u_2}{\partial x_3} \\ \frac{\partial u_3}{\partial x_1} & \frac{\partial u_3}{\partial x_2} & \frac{\partial u_3}{\partial x_3} \end{bmatrix} \tag{2.22}$$

where $\boldsymbol{u}$ is the velocity. Furthermore, this can be decomposed into its symmetric and skew-symmetric parts as

$$\nabla \boldsymbol{u} = \nabla^s \boldsymbol{u} + \nabla^w \boldsymbol{u} \tag{2.23}$$

Where in this case

$$\nabla^s = \frac{1}{2}\left(\nabla + \nabla^T\right) \text{ and } \nabla^w = \frac{1}{2}\left(\nabla - \nabla^T\right) \tag{2.24}$$

The symmetric tensor ∇^s is called the strain rate tensor while the skew -symmetric ∇^w is called the spin tensor.

2.2 The Navier-Stokes Equations

The well known Navier-Stokes Equations(NSE) were developed by Claude-Louis Navier and George Gabriel Stokes in 1822. These equations are very important in fluid dynamics as they are used to determine the velocity vector fields that applies to a fluid once a set of initial conditions have been given. They arise from the application of Newtons second law in combination with a fluid stress (due to viscosity) and a pressure term. For almost all real situations, they result in a system of nonlinear partial differential equations; however, with certain simplifications (such as 1-dimensional motion) they can sometimes be reduced to linear differential equations. Usually, however, they remain nonlinear, which makes them difficult or impossible to solve; this is what causes the turbulence and unpredictability in their results.

2.2.1 Derivation of the NSE

In this work the NSE will be derived using the Conservation form. Through this form the equations can be obtained by applying the underlying physical principles to a fluid element fixed in space. On the other hand it counter part, the Non-conservative forms can be obtained by considering fluid elements moving in the flow field. The link between these two forms of equations can be established using the material derivative as derived and elaborated in (2.8) and (2.11). Detailed information concerning the derivation of these equations can be found in the classicall text book on fluid dynamics by GK Batchelor [2]

2.2.2 Basic Theorems

Before deriving the conservation equations we need to establish two important theorems which will be used in the subsequent derivations. The derivation of these theorems will consider the concept of an intensive property L. This is defined as a property of a system that does not depend on the system size or the amount of material in the system(i.e Temperature). Throughout this work we will denote a control volume (domain) as Ω and its bounding surface area as $\partial\Omega$.

Reynolds transport theorem

This is an important theorem in fluid mechanics and is states that

$$\frac{d}{dt}\int_{\Omega} L\, dV = -\int_{\partial\Omega} L\mathbf{v}\cdot\boldsymbol{n}\, \mathrm{d}\Gamma - \int_{\Omega} Q \mathrm{d}\Omega \tag{2.25}$$

where the rate of change of the property L inside the volume is denoted on the left hand side (L.H.S) of the equation.On the other hand the R.H.S is the sum of two terms, a flux term $\int_{\partial\Omega} L\mathbf{v}\cdot\boldsymbol{n}\, \mathrm{d}\Gamma$ which basically indicates how much quantity of the property L is leaving the volume by flowing over the boundary and the sink term $\int_{\Omega} Q\, \mathrm{d}\Omega$ which describes how much of the property is leaving the volume due to sinks or sources inside the boundary.

Gauss's theorem

Also sometimes refereed to as divergence theorem, Gauss's theorem states that the integral of the outer normal component of a vector over a closed surface is equal to the integral of the divergence of the vector over the volume bounded by this closed surface. Therefore, with regards to equation (2.25) the divergence theorem allows the flux term of the equation to be expressed as a volume integral.

$$\int_{\partial\Omega} L\mathbf{v}\cdot\boldsymbol{n}\, \mathrm{d}\Gamma = \int_{\Omega} \nabla\cdot(L\mathbf{v})\, \mathrm{d}\Omega \tag{2.26}$$

Hence, (2.25) can be rewritten as

$$\frac{d}{dt}\int_{\Omega} L\, \mathrm{d}\Gamma = -\int_{\Omega} \nabla\cdot(L\mathbf{v}) + Q\, \mathrm{d}\Omega \tag{2.27}$$

To see the final resulting equation we can utilize the Leibniz's rule which states that

$$\frac{d}{dx}\int_a^b f(x,y)dy = \int_a^b \frac{d}{dx} f(x,y)dy \tag{2.28}$$

Therefore apply this to (2.27) we get the following

$$\int_{\Omega} \frac{d}{dt} L\, \mathrm{d}\Gamma + \nabla\cdot(L\mathbf{v}) + Q\, \mathrm{d}\Omega = 0 \tag{2.29}$$

Here it is noted that the e only way the above equality remains true for all control volumes is if the integrand itself is zero. Which leads to the general form of the continuity equation as follows

$$\frac{d}{dt} L + \nabla\cdot(L\mathbf{v}) + Q = 0 \tag{2.30}$$

2.2.3 Conservation of mass

The principle of conservation of mass declares that the mass contained in a volume Ω does not change with time. Therefore, this can mathematically be represented as

$$\frac{Dm}{Dt} = \frac{D}{Dt}\int_{\Omega} \rho(\mathbf{x},t)\ \mathrm{d}\Omega = 0 \tag{2.31}$$

where $\rho(\mathbf{x},t)$ represents the density of the continuum. Using the Reynolds theorem derived in (2.25) the right hand of (2.31) results in

$$\int_{\Omega} \frac{\partial\rho}{\partial t} + \int_{\partial\Omega} \rho(\mathbf{n}\cdot\boldsymbol{u})\ \mathrm{d}\Gamma = 0 \tag{2.32}$$

Applying the Gauss's theorem (2.26) the above equation becomes

$$\int_{\Omega} \left[\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u}) \right] \mathrm{d}\Omega = 0 \tag{2.33}$$

This equation should be valid for any arbitrarily volume Ω and hence, the integral can be dropped which results in

$$\frac{d\rho}{dt} + \nabla \cdot (\rho \boldsymbol{u}) = 0 \tag{2.34}$$

This is the equation of conversation of mass. For an incompressible fluid, the density is assumed to be constant which therefore leads to

$$\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0 \tag{2.35}$$

2.2.4 Conservation of momentum

One key component that is necessary for the derivation of the conservation of momentum is the concept of material derivative as described before in (2.7). The conservation of momentum states that material rate of change of linear momentum in a control volume Ω is equal to the rate of gain of momentum due to the body forces and surface stresses. This principle is stated as

$$\frac{\partial}{\partial t} = \int_{\Omega} \rho \, \boldsymbol{u} \, \mathrm{d}\Omega + \int_{\partial\Omega} \rho \, \mathbf{n} \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) \, \mathrm{d}\Gamma = \int_{\partial\Omega} \mathbf{n} \cdot \boldsymbol{\sigma} \, \mathrm{d}\Gamma \; + \int_{\Omega} \rho \, \mathbf{f} \, \mathrm{d}\Omega \tag{2.36}$$

where $\mathbf{f}$ represents the body forces, $\boldsymbol{\sigma}$ is the stress tensor, $\boldsymbol{u} \otimes \boldsymbol{u}$ is the dyadic product. Applying Gauss's theorem Eq.(2.26) to Eq.(2.36) yields

$$\int_{\Omega} \frac{\partial}{\partial t}(\rho \boldsymbol{u}) + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u}) \, \mathrm{d}\Omega = \int_{\Omega} \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} + \rho \, \mathbf{f} \, \mathrm{d}\Omega \tag{2.37}$$

The above equation is valid for any arbitrary volume which leads to

$$\frac{\partial}{\partial t}(\rho \boldsymbol{u}) + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u}) = \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} + \rho \, \mathbf{f} \tag{2.38}$$

using vector analysis we can further expand this equation as

$$\boldsymbol{u} \frac{\partial \rho}{\partial t} + \rho \frac{\partial \boldsymbol{u}}{\partial t} + \rho \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \boldsymbol{u} \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u}) = \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} + \rho \, \mathbf{f} \tag{2.39}$$

If we apply the continuity equation in Eq.(2.34) on the left side of Eq.(2.39) we obtain

$$\rho \frac{\partial \boldsymbol{u}}{\partial t} + \rho \boldsymbol{u} \cdot \nabla \boldsymbol{u} = \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} + \rho \, \mathbf{f} \tag{2.40}$$

This can be expressed in terms of the material derivative as

$$\rho \frac{D\boldsymbol{u}}{Dt} = \rho \, \mathbf{f} + \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} \tag{2.41}$$

Equations (2.40) and (2.41) are usually referred to as forms of the equation of motion. Since in this work we are interested in Newtonian fluids it important that we elaborate more on the $\boldsymbol{\sigma}$ term. For Newtonian fluids, the stress tensor is linearly related with the strain rate tensor. This relationship is expressed through the Cauchy stress tensor (σ_{ij}) which can be decomposed into the sum of the isotropic part $-p\delta_{ij}$ and the non isotropic part s_{ij} often referred to as the deviatoric tensor. That is:

$$\sigma_{ij} = -p\delta_{ij} + s_{ij} \tag{2.42}$$

Therefore, for a Newtonian fluid

$$\sigma_{ij} = -p\,\delta_{ij} + s_{ij} = -p\delta_{ij} + \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \lambda\frac{\partial u_k}{\partial x_k}\delta_{ij} \tag{2.43}$$

where μ is the dynamic viscosity and λ is the second coefficient of viscosity. Although we have not shown here, using vector notation and performing further manipulations on Eq. (2.43) leads to

$$\boldsymbol{\sigma} = -p\,\mathbf{I} + \boldsymbol{\tau} \Rightarrow -p\,\mathbf{I} + \mathbb{C}\nabla^s\boldsymbol{u} \Rightarrow -p\,\mathbf{I} + 2\mu\left(\nabla^s\boldsymbol{u} - \frac{1}{3}(\nabla\cdot\boldsymbol{u})\right) \tag{2.44}$$

where $\boldsymbol{\tau}$ is the shear-stress and $\mathbb{C}$ represents a fourth order tensor. As expressed in Eq. (2.35) for incompressible flow $\nabla\cdot\boldsymbol{u} = 0$ which then reduces Eq.(2.44) to

$$\boldsymbol{\sigma} = -p\,\mathbf{I} + 2\mu\nabla^s\boldsymbol{u} \tag{2.45}$$

Using Eq.(2.45) the equation of motion Eq.(2.40) can be expressed in a more convenient form

$$\frac{\partial\boldsymbol{u}}{\partial t} + (\mathbf{u}\ \cdot\boldsymbol{\nabla})\ \mathbf{u} - 2\nu\boldsymbol{\nabla}\cdot\boldsymbol{\nabla}^s\boldsymbol{u} + \boldsymbol{\nabla}p = \mathbf{f} \tag{2.46}$$

where in this case $\nu = \mu/\rho$ is kinematic viscosity and p is referred to as the kinematic pressure, that is pressure divided by the density. Further more Eq.(2.46) can be written as

$$\frac{\partial\boldsymbol{u}}{\partial t} + (\mathbf{u}\ \cdot\boldsymbol{\nabla})\ \mathbf{u} - \nu\boldsymbol{\nabla}^2\boldsymbol{u} - \nu\boldsymbol{\nabla}(\boldsymbol{\nabla}\cdot\boldsymbol{u}) + \boldsymbol{\nabla}p = \mathbf{f} \tag{2.47}$$

This form of the equation is referred to as the velocity−pressure stress−divergence form. Applying the incompressible condition Eq.(2.35) to Eq.(2.47) leads to an equation that represents the incompressible form of the momentum equation, that is

$$\frac{\partial\boldsymbol{u}}{\partial t} + (\mathbf{u}\ \cdot\boldsymbol{\nabla})\ \mathbf{u} - \nu\boldsymbol{\nabla}^2\boldsymbol{u} + \boldsymbol{\nabla}p = \mathbf{f} \tag{2.48}$$

2.2.5 Conservation of Energy

The conservation of energy in the control volume means that the rate of accumulation of energy together with the flux of energy through the boundary of the control volume is equal to the flux of heat through the boundary plus the rate of gain of energy due to surface stresses.

$$\frac{\partial}{\partial t}\int_\Omega \rho\,E\ \ \mathrm{d}\Omega + \int_{\partial\Omega}\rho\,E(\mathbf{n}\cdot\boldsymbol{u})\ \mathrm{d}\Gamma = -\int_{\partial\Omega}(\mathbf{n}\cdot\mathbf{q})\ \mathrm{d}\Gamma + \int_{\partial\Omega}\mathbf{n}\cdot(\boldsymbol{\sigma}\cdot\boldsymbol{u})\ \mathrm{d}\Gamma \tag{2.49}$$

here $\mathbf{q}$ is the heat flux vector E is the total specific energy given by

$$E = e + \frac{1}{2}\boldsymbol{u}^2 - \mathbf{f}\cdot\boldsymbol{u} \tag{2.50}$$

where e is the specific internal energy $\frac{1}{2}\boldsymbol{u}^2$ the specific kinetic energy , and $\mathbf{f}\cdot\boldsymbol{u}$ the specific potential energy. By applying Gauss's divergence theorem we have

$$\int_\Omega\left[\frac{\partial}{\partial t}(\rho E) + \nabla\cdot(\rho E\boldsymbol{u})\right]\ \mathrm{d}\Omega = \int_\Omega\left[-\nabla\cdot\mathbf{q} + \nabla\cdot(\boldsymbol{\sigma}\cdot\boldsymbol{u})\right]\mathrm{d}\Omega \tag{2.51}$$

since the control volume is arbitrarily the integrals can be dropped leading to

$$\frac{\partial}{\partial t}(\rho E) + \nabla\cdot(\rho E\boldsymbol{u}) = -\nabla\cdot\mathbf{q} + \nabla\cdot(\boldsymbol{\sigma}\cdot\boldsymbol{u}) \tag{2.52}$$

Equation (2.52) can further be expanded using vector and tensor analysis in order to obtain an expression of the first law of thermodynamics as shown below.

$$\rho \frac{De}{Dt} = -\nabla \cdot \mathbf{q} - p\nabla \cdot \boldsymbol{u} + \boldsymbol{\tau} : \mathbf{q} \tag{2.53}$$

where **:** represents the trace of two tensors. Further discussions on (2.52) can be found in the classical text books of fluid dynamics [2].

In this work we are interested in the case where the fluid flow is considered to be incompressible hence the energy equation will not be mentioned after this chapter. As mentioned before this conclusion is based on the fact that in incompressible flows the density is assumed to be a known constant value therefore there is no need to consider the equation of state which is needed for compressible flows to relate density, pressure and temperature.

2.2.6 Modified Navier-Stokes Equations for Embedded formulation

Before introducing the embedded formulation which uses the variational multiscale method, it is important to state here that although this formulation is used for incompressible flows problems, it has been implemented using pseudo-compressible Navier-stokes equations. This has been motivated by the need to bound pressure in complex geometries that may have isolated fluid cavities. Figure 2.3 shows a problem that may arise when using the embedded formulation, here we have a structure that has a fluid cavity inside which is represented by a circular hole (which in 3D we may assume to be a cylinder). Since the embedded formulation uses a level set distance function (d) to distinguish between the nodes inside the fluid ($d > 0$) and those inside structure ($d < 0$), the nodes representing the fluid cavity inside the structure will also be denoted as ($d > 0$). This causes a problem for the formulation as the pressure inside the hole is not bounded at any point which may cause the simulation to blow up. The remedy for this problem is the introduction of the pressure term inside the mass conservation equation. Introducing this term leads to the pseudo-compressible Navier-Stokes equations.

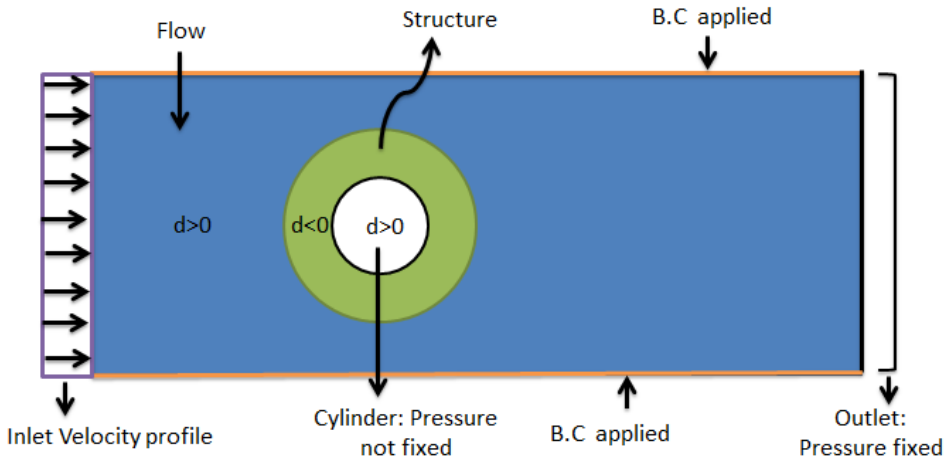


Figure 2.3: Complex geometry

We begin the process of deriving pseudo-compressible Navier-Stokes equations using the momentum equation from Eq.(2.40) where the definition of the Cauchy stress tensor is taken from Eqs.

(2.44) . As for the mass conservation we take Eq.(2.34) where we consider the addition of the material time derivative of the density as shown below

$$\rho\frac{\partial \boldsymbol{u}}{\partial t} - \nabla\cdot(-p\mathbf{I}+\boldsymbol{\tau}) + \rho\,\mathbf{a}\cdot\nabla\boldsymbol{u} = \rho\mathbf{f} \tag{2.54}$$

$$\frac{D\rho}{Dt} - \rho\nabla\cdot\boldsymbol{u} = 0 \tag{2.55}$$

Note that in equation 2.54 we have modified the convective term $\boldsymbol{u}\cdot\nabla\boldsymbol{u}$ to introduce a convective velocity $\mathbf{a}$ hence we have $\mathbf{a}\cdot\nabla\boldsymbol{u}$. This is done for the purpose of convienience, which helps if we want to linearize the problem. Furthermore, changes are made to the continuity equation using the material derivative expression and the sound velocity state equation as elaborated below.

$$\frac{D(\cdot)}{Dt} = \frac{\partial(\cdot)}{\partial t}\mathbf{a}\cdot\nabla(\cdot) \Rightarrow \frac{\partial\rho}{\partial t} + \mathbf{a}\cdot\nabla\rho + \rho\nabla\cdot\boldsymbol{u} = 0 \tag{2.56}$$

Assuming small space variation then $\mathbf{a}\cdot\nabla\rho \approx 0$ hence the convective term in the density material time derivative Eq. (2.56) is dropped. Therefore, the density derivative term is evaluated using the chain rule as

$$\frac{\partial p}{\partial t} = \frac{\partial p}{\partial\rho}\frac{\partial\rho}{\partial t} \Rightarrow \frac{\partial\rho}{\partial t} = \left(\frac{\partial p}{\partial\rho}\right)^{-1}\frac{\partial p}{\partial t} \tag{2.57}$$

and applying the sound velocity state equation we have

$$\frac{\partial p}{\partial\rho} = c^2 \Rightarrow \left(\frac{\partial p}{\partial\rho}\right)^{-1} = \frac{1}{c^2} \tag{2.58}$$

This is replaced in (2.57) which leads to the expression of the mass conservation equation (2.55) as

$$\frac{1}{c^2}\frac{\partial p}{\partial t} + \rho\boldsymbol{\nabla}\cdot\boldsymbol{u} = 0 \tag{2.59}$$

Therefore this results in the following formulations of the momentum and mass conservation laws

$$\rho\frac{\partial \boldsymbol{u}}{\partial t} - \nabla\cdot(-p\mathbf{I}+\boldsymbol{\tau}) + \rho\,\mathbf{a}\cdot\nabla\boldsymbol{u} = \rho\mathbf{f} \tag{2.60}$$

$$\frac{1}{c^2}\frac{\partial p}{\partial t} + \rho\boldsymbol{\nabla}\cdot\boldsymbol{u} = 0 \tag{2.61}$$

It is important to note that in if $c \to \infty$ then we can regain the original incompressible Navier-Stokes equations since the pseudo compressible term will vanish on this condition.

2.3 Variational Multiscale Method

In this section we will review some aspects of Variational Multiscale Methods (VMS) framework as it is the backbone of the embedded finite element solver that will be validated in this thesis. The VMS provides a stabilized method for the simulation of the Navier-stokes method. A key framework of VMS is the separation of the solution into resolved and unresolved parts which can be achieved through the definition of the large scale and small scale solution spaces. A detailed analysis of this method and its application in Kratos multiphysics can be found in [9]. The formulations to be derived in this section are as a result of the research carried out by Rubén Zorrilla Martínez who implemented the embedded solver in Kratos multiphysics.

To form a well-posed initial boundary problem we take the pseudo-compressible Navier-Stokes equations (2.60) and (2.61) which needs to be completed with suitable initial and boundary conditions.

This is done by prescribing the value $\boldsymbol{u}_D$ of the velocity on the portion Γ_D of the boundary and also the boundary traction $\mathbf{t}$ on the complementary portion of Γ_N as follows;

$$\boldsymbol{u} = \boldsymbol{u}_D \quad \text{on } \Gamma_D \tag{2.62}$$

$$(-p\mathbf{I} + \boldsymbol{\tau}) \cdot \mathbf{n} = \mathbf{t} \quad \text{on } \Gamma_N \tag{2.63}$$

where vector $\mathbf{n}$ denotes the unit outer normal to the boundary. Before finding the weak form of the NSE we introduce the residual forms of these equations as follows;

$$R^m(\boldsymbol{u}, p) = \rho\,\mathbf{f} - \rho\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot (-p\,\mathbf{I} + \boldsymbol{\tau}) - \rho\,\mathbf{a} \cdot \boldsymbol{\nabla}\boldsymbol{u} \tag{2.64}$$

$$R^c(\boldsymbol{u}, p) = -\frac{1}{c^2}\frac{\partial p}{\partial t} - \rho\nabla \cdot \boldsymbol{u} \tag{2.65}$$

To obtain the weak form for the Navier-Stokes equations through the Galerkin method we need to multiply Eq.(2.64) by a velocity test function $\boldsymbol{w}$ which is defined to be zero on the Dirichlet boundary Γ_D. Consequently we also multiply Eq.(2.65) by the pressure test function q. Performing these procedures and defining the dot product volume integration as $(\bullet,\bullet)_\Omega$ to represent the integrals leads to

$$(\boldsymbol{w}, R^m(\boldsymbol{u}, p))_\Omega \Rightarrow (\boldsymbol{w}, \rho\boldsymbol{f})_\Omega - \left(\boldsymbol{w}, \rho\frac{\partial \boldsymbol{u}}{\partial t}\right)_\Omega - (\boldsymbol{w}, \nabla p)_\Omega + (\boldsymbol{w}, \nabla \cdot \boldsymbol{\tau})_\Omega - (\boldsymbol{w}, \rho\boldsymbol{a} \cdot \nabla\boldsymbol{u})_\Omega \tag{2.66}$$

$$(q, R^c(\boldsymbol{u}, p))_\Omega \Rightarrow -\left(q, \frac{1}{c^2}\frac{\partial p}{\partial t}\right)_\Omega - (q, \rho\nabla \cdot \boldsymbol{u})_\Omega \tag{2.67}$$

By using the Galerkin functional which states $(\boldsymbol{w}, R^m(\boldsymbol{u}, p))_\Omega + (q, R^c(\boldsymbol{u}, p))_\Omega = 0$ leads to the following weak form.

$$\begin{aligned}(\boldsymbol{w}, \rho\mathbf{f})_\Omega - \left(\boldsymbol{w}, \rho\frac{\partial \boldsymbol{u}}{\partial t}\right)_\Omega - (\boldsymbol{w}, \nabla p + (\boldsymbol{w}, \nabla \cdot \boldsymbol{\tau})_\Omega - (\boldsymbol{w}, \rho\boldsymbol{a} \cdot \nabla\boldsymbol{u})_\Omega \\ - \left(q, \frac{1}{c^2}\frac{\partial p}{\partial t}\right)_\Omega - (q, \rho\nabla \cdot \boldsymbol{u})_\Omega = 0\end{aligned} \tag{2.68}$$

The issue of functional spaces will be discussed further in section 2.4.4, but its important to mention here that once integration by parts has been performed on the respective terms in Eq.(2.68) we require that the space of functions where $\boldsymbol{w}$,$\boldsymbol{u}$,q,p and $\mathbf{f}$ be defined by

$$\boldsymbol{w} \in H^1_D = \left\{\boldsymbol{w} \in H^1(\Omega) | \boldsymbol{w} = \mathbf{0} \text{ in } \Gamma_D\right\} \tag{2.69}$$

$$\boldsymbol{u} \in H^1_D = \left\{\boldsymbol{u} \in H^1(\Omega) | \boldsymbol{u} = \boldsymbol{u}_D \text{ in } \Gamma_D\right\} \tag{2.70}$$

$$p, q \in L^2(\Omega) \tag{2.71}$$

$$\mathbf{f} \in H^{-1}(\Omega) \tag{2.72}$$

where H^1, L^2 and H^{-1} are the Sobolev spaces.

The convective term in Eq.(2.68) plays an important role in the the formulation and hence we need to elaborate more on this term. As longs as the velocity field is divergence-free the convective terms can be expressed in three identical form as follows

$$\boldsymbol{a} \cdot \nabla\boldsymbol{u} = \nabla \cdot (\boldsymbol{a} \otimes \boldsymbol{u}) = \frac{1}{2}\boldsymbol{a} \cdot \boldsymbol{\nabla}\boldsymbol{u} + \frac{1}{2}\nabla \cdot (\boldsymbol{a} \otimes \boldsymbol{u}) \tag{2.73}$$

The *non-conservative* form of the convective term is represented by the first expression in Eq.(2.73) while the *conservative* form by the second term. On the other hand the third term represents the *skew-symmetric* form. In this work we will use the *non-conservative* form. Interested readers on the use

of the *conservative* can consult [7] where they investigated extensively the implications of keeping the nonlinear convective terms. The use of the *skew-symmetric* has also been researched in [9] where they used it to gain further insight on turbulent flow simulations. Unfortunately the Galerkin weak form in Eq.(2.68) is not easy to solve using finite element since its discrete version is not numerically stable. This is mainly due to two reasons; firstly because of the presence of nonlinear and non-symmetric convective terms which leads to the increase in the Reynolds number hence resulting in convection- dominated flows [16]. Secondly the discrete spaces in which the solutions $\boldsymbol{u}$ and p are sought must satisfy the *inf-sub* or *Ladyženskaja-Babuška-Brezzi* (LBB) condition ,[12],[14], . This condition is given by

$$\inf_{q \neq 0\, q \in Q} \sup_{\boldsymbol{w} \neq 0\, \boldsymbol{w} \in V} \frac{\int_\Omega q \nabla \cdot \boldsymbol{w}\, \mathrm{d}\Omega}{\|q\|_Q \|\boldsymbol{w}\|_V} \geq \alpha \tag{2.74}$$

where V and Q are the spaces where the velocity and pressure solutions are defined and α is a positive constant that is independent of the mesh size. Usually fulfilling in the LBB condition requires the use of higher order interpolation for velocity than pressure. A good example of stable elements that satisfy the LBB condition are the Taylor-Hood elements [12] though it sholud be noted that generally it is not easy to prove if a given velocity -pressure pair satisfies the LBB compatibility condition.

To overcome these instabilities stabilized formulations such as the Strealine-Upwind Petrov-Galerkin (SUPG), Galerkin-Least square (GLS) among others are employed. Usually these stabilized formulation are derived in such a way that they result in a modified weak form that is not restricted by the LBB condition. Other alternatives lies within the VMS framework which is the main framework on which this work is based.

Separation of scales

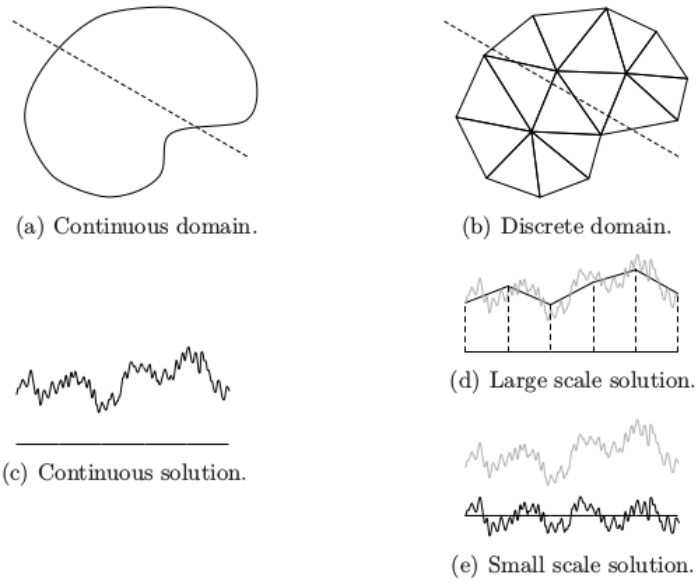
Since its introduction VMS methods have provided a framework for the development of stabilization techniques which are capable of overcoming the numerical difficulties encountered when using the standard Galerkin method. A general feature of this framework which is rather also the starting point of VMS is the decomposition of the solution variables into large scale $(\cdot)_h$ and the subscale (small scale) $(\cdot)_s$ which in the problem at hand corresponds to

$$\boldsymbol{u} = \boldsymbol{u}_h + \boldsymbol{u}_s \tag{2.75}$$

$$p = p_h + p_s \tag{2.76}$$

The way these are modeled defines the particular numerical approximation[7]. In this work will use the same finite element test functions for both that large and small scales however it should be noted that in some VMS framework the pressure and velocity test functions are also separated [9]. We can identify the FE components of the solution as the resolved scales, whereas the subscales are the unresolved scales. A graphical representation of the resolved and unresolved scales is illustrated Figure 2.4. Inserting Eq.2.75 and Eq.2.76 into Eq. 2.66 and Eq.2.67 leads to the following expression.

$$\begin{aligned}(\boldsymbol{w}, \rho\mathbf{f})_\Omega - \left(\boldsymbol{w}, \rho \frac{\partial(\boldsymbol{u}_h + \boldsymbol{u}_s)}{\partial t}\right)_\Omega - (\boldsymbol{w}, \nabla(p_h + p_s))_\Omega + (\boldsymbol{w}, \nabla \cdot \boldsymbol{\tau})_\Omega - (\boldsymbol{w}, \rho \boldsymbol{a} \cdot \nabla(\boldsymbol{u}_h + \boldsymbol{u}_s))_\Omega \\ - \left(q, \frac{1}{c^2} \frac{\partial(p_h + p_s)}{\partial t}\right)_\Omega - (q, \rho \nabla \cdot (\boldsymbol{u_h} + \boldsymbol{u_s}))_\Omega = 0\end{aligned} \tag{2.77}$$



(a) Continuous domain. (b) Discrete domain. (d) Large scale solution. (c) Continuous solution. (e) Small scale solution.

Figure 2.4: scale separation (Adopted from [9])

Where when the large and small scales are separated we get

$$\begin{aligned}(\boldsymbol{w},\rho\mathbf{f})_\Omega-\left(\rho\frac{\partial \boldsymbol{u}_h}{\partial t}\right)_\Omega-\left(\rho\frac{\partial \boldsymbol{u}_s}{\partial t}-\right)_\Omega+(\boldsymbol{w},\nabla p_h)_\Omega-(\boldsymbol{w},\nabla p_s)_\Omega+(\boldsymbol{w},\nabla\cdot\boldsymbol{\tau}_h)_\Omega+(\boldsymbol{w},\nabla\cdot\boldsymbol{\tau}_s)_\Omega\\ -(\boldsymbol{w},\rho\mathbf{a}\cdot\nabla\boldsymbol{u}_h)_\Omega-(\boldsymbol{w},\rho\mathbf{a}\cdot\nabla\ \mathbf{u}_s)_\Omega-\left(q,\frac{1}{c^2}\frac{\partial p_h}{\partial t}\right)_\Omega-\left(q,\frac{1}{c^2}\frac{\partial p_s}{\partial t}\right)_\Omega-(q,\rho\nabla\cdot\boldsymbol{u}_h)_\Omega\\ -(q,\rho\nabla\cdot\boldsymbol{u}_s)_\Omega=0\end{aligned}\tag{2.78}$$

As it can be seen in the above Eq.(2.78) we have terms that involve derivatives of the subscale variables, hence taking into account that linear FE are to be used, integration by parts is required to get rid of the second order terms and subscales derivatives.Note that the resulting boundary integrals will be denoted using the commonly used integration symbol;

$$(\boldsymbol{w},\nabla p_h)_\Omega=-(\nabla\cdot\boldsymbol{w},p_h)_\Omega+\int_\Gamma p_h\boldsymbol{w}\cdot\mathbf{n}\ d\Gamma\tag{2.79}$$

$$(\boldsymbol{w},\nabla p_s)_\Omega=-(\nabla\cdot\boldsymbol{w},p_s)_\Omega+\int_\Gamma p_s\boldsymbol{w}\cdot\mathbf{n}\ d\Gamma\tag{2.80}$$

$$(\boldsymbol{w},\nabla\cdot\boldsymbol{\tau}_h)_\Omega=-(\nabla^s\boldsymbol{w},\boldsymbol{\tau}_h)_\Omega+\int_\Gamma(\boldsymbol{w}\cdot\boldsymbol{\tau}_h)\mathbf{n}\ d\Gamma\tag{2.81}$$

$$(\boldsymbol{w},\nabla\cdot\boldsymbol{\tau}_s)_\Omega=-(\nabla^s\boldsymbol{w},\boldsymbol{\tau}_s)_\Omega+\int_\Gamma(\boldsymbol{w}\cdot\boldsymbol{\tau}_s)\mathbf{n}\ d\Gamma\tag{2.82}$$

$$(\boldsymbol{w},\rho\mathbf{a}\cdot\nabla\boldsymbol{u}_s)_\Omega=(\rho\mathbf{a}\otimes\boldsymbol{w},\nabla\boldsymbol{u}_s)_\Omega\tag{2.83}$$

$$=-(\rho\nabla\cdot(\mathbf{a}\otimes\boldsymbol{w}),\boldsymbol{u}_s)_\Omega+\int_\Gamma(\rho\nabla\cdot(\mathbf{a}\otimes\boldsymbol{w}),\boldsymbol{u}_s)\cdot\mathbf{n}\tag{2.84}$$

$$=-\left(\rho\frac{\partial}{\partial x_i}(a_iw_j)u_{s,j}\right)_\Omega+\int_\Gamma\rho(a_iw_j)u_{s,j}n_i\tag{2.85}$$

$$=-\left(\rho\left(\frac{\partial a_i}{\partial x_i}w_j+a_i\frac{\partial w_j}{\partial x_i}\right)u_{s,j}\right)_\Omega+\int_\Gamma\rho(a_iw_j)u_{s,j}n_i\tag{2.86}$$

$$=-(\rho((\nabla\cdot\mathbf{a})\boldsymbol{w}+\mathbf{a}\nabla\boldsymbol{w})\boldsymbol{u}_s)_\Omega+\int_\Gamma\rho(\mathbf{a}\otimes\boldsymbol{w})\boldsymbol{u}_s)\cdot\ \mathbf{n}\tag{2.87}$$

$$(\boldsymbol{w},\rho\mathbf{a}\cdot\nabla\boldsymbol{u}_s)_\Omega=-(\rho\,\mathbf{a}\nabla\boldsymbol{w},\boldsymbol{u}_s)_\Omega+\int_\Gamma(\mathbf{n}\cdot(\mathbf{a}\otimes\boldsymbol{w}))\boldsymbol{u}_s\ d\Gamma\tag{2.88}$$

$$(q,\rho\nabla\cdot\boldsymbol{u}_s)_\Omega=-(\nabla q,\rho\boldsymbol{u}_s)_\Omega+\int_\Gamma(\rho q\mathbf{n})\cdot\boldsymbol{u}_s\ d\Gamma\tag{2.89}$$

Note that the boundary terms in Eqs. 2.80 and 2.86 vanish over the internal boundaries if we assume that $p_s \approx 0$ and $\boldsymbol{u}_s \approx 0$. Inserting the results from Eq.(2.79) to Eq.(2.89) into Eq.(2.78) results in

$$\begin{aligned}(\boldsymbol{w}, \rho\mathbf{f})_\Omega - \left(\boldsymbol{w}, \rho\frac{\partial \boldsymbol{u}_h}{\partial t}\right)_\Omega - \left(\boldsymbol{w}, \rho\frac{\partial \boldsymbol{u}_s}{\partial t}\right)_\Omega + \left(\nabla\rho\frac{\partial \boldsymbol{u}_s}{\partial t}\right)_\Omega + (\nabla\cdot\mathbf{w}\ , p_h)_\Omega + (\nabla\cdot\mathbf{w}\ , p_s)_\Omega \\ - \int_\Gamma p_h \boldsymbol{w}\cdot\mathbf{n} - \int_\Gamma p_s \boldsymbol{w}\cdot\mathbf{n} - (\nabla^s\boldsymbol{w}, \boldsymbol{\tau}_h)_\Omega - (\nabla^s\boldsymbol{w}, \boldsymbol{\tau}_s)_\Omega + \int_{\Gamma_N}(\boldsymbol{w}\boldsymbol{\tau}_h)\cdot\mathbf{n} + \int_{\Gamma_N}(\boldsymbol{w}\boldsymbol{\tau}_s)\cdot\mathbf{n} \\ - (\boldsymbol{w}, \rho\,\mathbf{a}\cdot\nabla\boldsymbol{u}_h)_\Omega + (\rho\,\mathbf{a}\cdot\nabla\boldsymbol{w}, \boldsymbol{u}_s)_\Omega - \int_\Gamma(\mathbf{n}\cdot(\mathbf{a}\otimes\boldsymbol{w}))\boldsymbol{u}_h + (\rho(\nabla\cdot\mathbf{a})\boldsymbol{w}\boldsymbol{u}_s)_\Omega \\ - \left(q, \frac{1}{c^2}\frac{\partial p_h}{\partial t}\right)_\Omega - \left(q, \frac{1}{c^2}\frac{\partial p_s}{\partial t}\right)_\Omega - (q, \rho\nabla\cdot\boldsymbol{u}_h)_\Omega - (q, \rho\nabla\cdot\boldsymbol{u}_s)_\Omega - \int_\Gamma(\rho\,q\,\mathbf{n})\cdot\boldsymbol{u}_s = 0\end{aligned} \tag{2.90}$$

While Eq.(2.90) represents the complete derived model for the dynamic Navier-Stokes equations we need to define a model for the subscales. In terms of the time variation in the Quasi - static subscales we neglect the time variation of the velocity and pressure subscale model, that is to say:

$$\frac{\partial u_s}{\partial t} \approx \frac{\partial p_s}{\partial t} \approx 0 \tag{2.91}$$

On the other hand if the dynamic model is used then the time derivatives in 2.91 are kept. To define a model for the unknown subscales the algebraic sub-grid dynamic (ASGS) model is used. Hence using ASGS we define $\boldsymbol{u}_s$ and p_s which in the embedded formulation are defined as:

$$\begin{aligned}\boldsymbol{u}_s &= \tau_1 R^m(\boldsymbol{u}, p) = \tau_1(\rho\mathbf{f} - \rho\frac{\partial u_h}{\partial t} + \nabla\cdot(-p\mathbf{I} + \boldsymbol{\tau}) - \rho\mathbf{a}\nabla\boldsymbol{u}_h) \\ p_h &= \tau_2 R^c(\boldsymbol{u}, p) = \tau_2\left(\frac{1}{c^2}\frac{\partial p}{\partial t}\rho\nabla\cdot\boldsymbol{u}\right)\end{aligned}$$

where

$$\tau_1 = \left(\frac{\rho\,\tau_{Dyn}}{\Delta t} + \frac{c_1\mu}{h^2} + \frac{c_2\rho\,\|\boldsymbol{a}\|}{h}\right)^{-1} \quad \text{and} \quad \tau_2 = \frac{h^2}{c_1\tau_1} = \mu\frac{c_2\,\|a\|\,h}{c_1} \tag{2.92}$$

with h been the characteristic length of the element and c_1, c_1 been constants with $c_1 = 4$ and $c_1 = 2$ for linear finite elements. Likewise τ_{Dyn} is a dynamic tau (tuning parameter) which value ranges from 0$\sim$1. Once the subsscale model is defined we then the final form of the weak form becomes

$$\begin{aligned}(\boldsymbol{w}, \rho\mathbf{f})_\Omega - \left(\boldsymbol{w}, \rho\frac{\partial \boldsymbol{u}_h}{\partial t}\right)_\Omega + (\nabla\cdot\boldsymbol{w}, p_h)_\Omega - (\nabla^s\boldsymbol{w}, \boldsymbol{\tau}_h)_\Omega - (\boldsymbol{w}, \rho\,\mathbf{a}\cdot\nabla\boldsymbol{u}_h)_\Omega - \left(q, \frac{1}{c^2}\frac{\partial p_h}{\partial t}\right)_\Omega \\ -(q, \rho\nabla\cdot\boldsymbol{u}_h)_\Omega + \int_{\Gamma_N}\boldsymbol{w}\cdot\mathbf{t}\ \mathrm{d}\Gamma + (\nabla\cdot\boldsymbol{w}, p_s)_\Omega - (\nabla^s\boldsymbol{w}, \tau_s)_\Omega + (\rho\mathbf{a}\nabla\boldsymbol{w}, \boldsymbol{u}_s)_\Omega \\ -(\rho(\nabla\cdot\mathbf{a})\boldsymbol{w}, \boldsymbol{u}_s)_\Omega + (\nabla q, \rho\boldsymbol{u}_s)_\Omega = 0\end{aligned} \tag{2.93}$$

Note that as shown below, we have used the boundary integrals from Eqs. (2.79) and (2.81) together with the definition of the Neumann boundary condition Eq.(2.63) to insert the traction in Eq. (2.93).

$$-\int_\Gamma p_h\boldsymbol{w}\cdot\mathbf{n}\ d\Gamma + \int_\Gamma(\boldsymbol{w}\cdot\boldsymbol{\tau}_h)\mathbf{n}\ d\Gamma = \int_\Gamma\boldsymbol{w}(\boldsymbol{\tau_h} - p\mathbf{I})\cdot\mathbf{n}\,\mathrm{d}\Gamma = \int_{\Gamma_N}\boldsymbol{w}\cdot\mathbf{t}\ \mathrm{d}\Gamma \tag{2.94}$$

A key remark about this formulation is that in order to get a better conditioned matrix (especially for $\rho \gg 1$) one can divide the mass conservation matrix by "ρ". That is

$$\left(\frac{1}{c^2}\frac{\partial p_h}{\partial t}\right) + \rho\nabla\cdot\boldsymbol{u} = 0 \Rightarrow \left(\frac{1}{\rho c^2}\frac{\partial p_h}{\partial t}\right) + \nabla\cdot\boldsymbol{u} = 0 \tag{2.95}$$

2.4 Computational Fluid Dynamics using Finite Element Method (FEM)

This section presents a review of how the finite element method (FEM) is used to numerically solve problems involving incompressible flow. Throughout the section we will focus more on describing the most important points such linearization, space and time discretization and finally on the use of Newton's approximation method which will lead to a summary of the equations for the Body-fitted formulation (Monolithic Eulerian solver).

2.4.1 Numerical Simulation of Incompressible Flow using the Finite Element Method

For simple fluid dynamics problems with applied boundary conditions, it is not difficult to obtain analytical solutions but in reality most problems are complex hence the need to apply CFD. The development of the CFD solver that will be validated in this work is based on FEM hence it is discussed here. Other well known numerical methods for CFD includes the Finite Difference Method (FDM) and the Finite Volume Method (FVM). Unlike FDM and FVM which satisfies the governing equations in strong form the FEM satisfies these equations in weak form as it will be discussed in the following sections. For a more rigorous review of both FDM and FVM interested readers may refer to the classical textbook on CFD by Anderson [1]. Classical textbooks on the FEM in general includes [3],[28],[30] and with respect to fluid flow [12] is advised.

2.4.2 Linearization

To give a clear overview of the linearization and discretization process we start by focusing on the use of FEM to solve an incompressible flow problem described using the transient Navier-Stokes equations in a continuous domain (Ω). These equations are similar to those we derived in the previous sections.

$$\rho\frac{\partial \boldsymbol{u}}{\partial t} - \mu\Delta\boldsymbol{u} + \rho\boldsymbol{u}\cdot\nabla\boldsymbol{u} + \nabla P = \mathbf{f} \tag{2.96}$$

$$\rho\nabla\cdot\boldsymbol{u} = 0 \tag{2.97}$$

To complete the problem, we require suitable boundary conditions on either velocity or pressure, as well as an initial velocity:

$$\boldsymbol{u} = \bar{\boldsymbol{u}} \text{ on } \Gamma_u,\ t \in [0,T] \tag{2.98}$$

$$\mathrm{p} = \bar{\mathrm{p}} \text{ on } \Gamma_p,\ t \in [0,T] \tag{2.99}$$

$$\boldsymbol{u} = \boldsymbol{u}_0 \text{ on } \Omega,\ t = 0 \tag{2.100}$$

with $\Gamma_u \cup \Gamma_p = \partial\Omega, \Gamma_u \cap \Gamma_p = \emptyset$

Where of course equation (2.96). describes the balance of linear momentum, and the terms on the left hand side account, respectively, for the dynamic response of the system, diffusion, convection and the effect of pressure gradients on the flow, while the right hand side term describes the body forces (such as gravity) that act on the interior of the domain. Equation (2.97). represents mass balance, which in our case is equivalent to ensuring the incompressibility of the fluid in the domain.

The major problem with the above equations is that they describe a coupled system of non-linear equation involving velocity, and the pressure. Hence, in order to solve this kind of problem using FEM we would want to linearize the convective term (note that one can solve the non-linear ones depending on the implementation though it is less robust but if it convergence then the converges is faster) and afterwards we need to carry out the discretization process of the equations in time. The linearization process is carried out by replacing the direct solution defined by Eqs.(2.96) and (2.97) by an iterative solution which results in the Ossen equation [8]. as follows

$$\rho\frac{\partial \boldsymbol{u}}{\partial t} - \mu\Delta\boldsymbol{u} + \rho\boldsymbol{a}\cdot\nabla\boldsymbol{u} + +\nabla P = \mathbf{f} \tag{2.101}$$

$$\rho\nabla\cdot\boldsymbol{u} = 0 \tag{2.102}$$

where the convective velocity $\mathbf{a} = \boldsymbol{u}^{(i-1)}$ is obtained in the previous iteration. Note that, this is only valid as long as the forces behave linearly, otherwise the forces need to be linearized as well. Note that in the upcoming sections we will denote $\frac{\partial \boldsymbol{u}}{\partial t}$ as $\partial_t \boldsymbol{u}$ for simplification purposes.

2.4.3 Time discretization

As it will be discussed later there are two principal formulations in the Kratos Multiphysics fluid dynamics application that will be used in this thesis, these been the body-fitted and embedded formulations. The body-fitted formulation uses the monolithic and fractional step solvers. Since in this thesis we will validate the embedded formulation using the body-fitted formulation for which we will use only the monolithic solver, it is therefore important that we review the monolithic part of the Body-fitted formulation. Both the monolithic and fractional step solvers have been validated in [9],[15] and in terms of the algorithm for the fractional step solver readers may refer to [25].

As a time iteration scheme, both Body-fitted uses the Bossak time iteration scheme based on velocities wile the embedded formulation uses the BDF2. The Bossak scheme [8] is an α-modified version of the Newmark family which has second order accuracy, is stable and damps high frequencies. Using this method leads to the time-discrete scheme

$$\rho((1-\alpha_b)\frac{\partial \boldsymbol{u}_{n+1}}{\partial t} + \alpha_b\frac{\partial \boldsymbol{u}_n}{\partial t}) - \mu\Delta\boldsymbol{u}_{n+1} + \rho\boldsymbol{a}\cdot\boldsymbol{\nabla}\boldsymbol{u}_{n+1} + \nabla P_{n+1} = \mathbf{f}_{n+1} \tag{2.103}$$

$$\rho\nabla\cdot\boldsymbol{u}_{n+1} = 0 \tag{2.104}$$

Where we can notice that

$$\partial_t\boldsymbol{u}_{n+1} = (1-\alpha_b)\frac{\partial \boldsymbol{u}_{n+1}}{\partial t} + \alpha_b\frac{\partial \boldsymbol{u}_n}{\partial t} \tag{2.105}$$

Therefore, the acceleration is obtained using Newmarks relation as

$$\frac{\partial \boldsymbol{u}_{n+1}}{\partial t} = \frac{1}{\gamma\Delta t}(\boldsymbol{u}_{n+1} - \boldsymbol{u}_n) - \frac{1-\gamma}{\gamma}\frac{\partial \boldsymbol{u}_n}{\partial t} \tag{2.106}$$

where

$$\alpha_b \leqslant 0 \; and \; \gamma = 0.25(1-\alpha_b)^2 \tag{2.107}$$

For simplification purposes we can use the definition of $\partial t\boldsymbol{u}_{n+1}$ in (2.105) and replace it in (2.103) to obtain the following

$$\rho\partial_t\boldsymbol{u}_{n+1} - \mu\Delta\boldsymbol{u}_{n+1} + \rho\boldsymbol{a}\cdot\nabla\boldsymbol{u}_{n+1} + \nabla P_{n+1} = \mathbf{f}_{n+1} \tag{2.108}$$

$$\rho\nabla\cdot\boldsymbol{u}_{n+1} = 0 \tag{2.109}$$

2.4.4 Finite element discretization

To solve the above NS equations (2.108) and (2.109) for the values of the velocities $\boldsymbol{u}_{n+1}$ and for the pressure P_{n+1} using FEM, need to derive the respective weak forms using the Galerkin method. This method employs the use of test functions. Therefore, for the momentum equation the test functions to be used is $\boldsymbol{w}$ and for the mass conservation the test function is q, which leads to the following integral equations;

$$\int_\Omega \boldsymbol{w}(\rho\partial t\boldsymbol{u}_{n+1} - \mu\Delta\boldsymbol{u}_{n+1} + \rho\boldsymbol{a}\cdot\boldsymbol{\nabla}\boldsymbol{u}_{n+1} + \nabla P_{n+1})\ \mathrm{d}\Omega = \int_\Omega \boldsymbol{w}\mathbf{f}_{n+1}\ \mathrm{d}\Omega \tag{2.110}$$

$$\int_\Omega q(\rho\nabla\cdot\boldsymbol{u}_{n+1})\ \mathrm{d}\Omega = 0 \tag{2.111}$$

As we are expecting boundary integrals to appear in these equations, certain terms in Eqs. (2.110) and (2.110) can be integrated by using integration by parts and also through the use of the divergence theorem and the given boundary conditions some of the terms can be written as integrals over the boundary which leads to the following weak from:

$$\begin{aligned}\int_\Omega \boldsymbol{w}\rho\partial t\boldsymbol{u}_{n+1} + \int_\Omega \mu\nabla\boldsymbol{w}\nabla\boldsymbol{u}_{n+1} + \int_\Omega \boldsymbol{w}\rho\boldsymbol{a}\cdot\nabla\boldsymbol{u}_{n+1} + \int_\Omega \boldsymbol{w}\nabla P_{n+1}\,\mathrm{d}\Omega = \int_\Omega \boldsymbol{w}\mathbf{f}_{n+1}\mathrm{d}\Omega \\ + \int_\Gamma \boldsymbol{w}(\mu\Delta\boldsymbol{u}_{n+1} - P_{n+1})\mathbf{n}\,\mathrm{d}\Gamma\end{aligned} \tag{2.112}$$

$$\int_\Omega q(\rho\nabla\cdot\boldsymbol{u}_{n+1})\,\mathrm{d}\Omega = 0 \tag{2.113}$$

On the right hand side (RHS) of (2.112) we ca set the boundary terms $(\mu\Delta\boldsymbol{u}_{n+1} - P_{n+1})\mathbf{n}$ as $\mathbf{t}_n$ which describes stresses on the boundary. Therefore, the weak forms of the equations becomes.

$$\begin{aligned}\int_\Omega \boldsymbol{w}\rho\partial_t\boldsymbol{u}_{n+1} + \int_\Omega \mu\nabla\boldsymbol{w}\nabla\boldsymbol{u}_{n+1} + \int_\Omega \boldsymbol{w}\rho\boldsymbol{a}\cdot\boldsymbol{\nabla}\boldsymbol{u}_{n+1} + \int_\Omega \nabla\cdot\boldsymbol{w}P_{n+1}\,\mathrm{d}\Omega = \int_\Omega \boldsymbol{w}\mathbf{f}_{n+1}\mathrm{d}\Omega \\ + \int_\Gamma \boldsymbol{w}\,\mathbf{t}_n\,\mathrm{d}\Gamma\end{aligned} \tag{2.114}$$

$$\int_\Omega q(\rho\nabla\cdot\boldsymbol{u}_{n+1})\,\mathrm{d}\Omega = 0 \tag{2.115}$$

Before going any further, it is cardinal that we look into the space of function where $\boldsymbol{w}$,$\boldsymbol{u}$,q,p and $\mathbf{f}$ are defined. This is due to the fact that the integrals that appear Eqs.(2.114) and (2.115) must be bounded which then imposes regularity requirements on the solution, test functions as well as the problem data. Therefore the formulation and subsequent discretization of these integral forms requires the definition of some function spaces and associated norms. To define these space we will use the Sobolev spaces and a more detailed explanation on this types of spaces is given in [12]. We begin the discussion with the L^2 space. This denotes the space of functions that are square integrable over the domain Ω. That is for any two given functions such as f, g it is important to make that

$$\int_\Omega fg\ \mathrm{d}\Omega < \infty \tag{2.116}$$

where once again the L^2 norms of a function is stated as

$$\|f\|_{L^2(\Omega)} = \left(\int_\Omega f^2\ \mathrm{d}\Omega\right)^{1/2} \tag{2.117}$$

Furthermore, we define the well known Hilbert spaces. These spaces are used to define spaces of square integrable and derivatives. Therefore following the nomenclature in [12] we define the H^1 Sobolev space

as

$$H^1(\Omega) = \left\{ v \in L^2(\Omega) | \frac{\partial v}{\partial x_i} \in L^2(\Omega) i = 1, ...n_{sd} \right\} \tag{2.118}$$

This space has square integrable functions of which the derivatives are of order also square integrable (Note that $H^0(\Omega) = L^2(\Omega)$). Also it important to state the subsequent subspace of $H^1(\Omega)$

$$H_0^1(\Omega) = \left\{ v \in H^1(\Omega) | v = 0 \text{ on } \Gamma \right\} \tag{2.119}$$

This used to define spaces which posses a square integrable first order derivative over the domain Ω and vanish on its boundary Γ. To complete the discussion on space function will also need to define the functions for the test function and the trial solution that we are using. For the test function we define class of function V such over a computational domain (Ω) this class consists of all functions that are square integrable and have square integrable first derivatives over the domain. This is defined as follows:

$$V = \left\{ w \in H^1(\Omega) | w = 0 \text{ on } \Gamma \right\} \equiv H^1_{\Gamma_D} \tag{2.120}$$

For the trial function we define a class of function S such that trial solutions satisfy the Dirichlet conditions as follows

$$S = \left\{ u \in H^1(\Omega) | u = u_D \text{ on } \Gamma_D \right\} \equiv V + \bar{u}_D \tag{2.121}$$

where in this case $\bar{u}_D$ is any function in $H_1(\Omega)$ such that $\bar{u}_D = u_D$ on Γ_D. Not that in cases where $u_D = 0$ both $V = S = H_0^1(\Omega)$. For a discretized finite domain with an element size of h, we can further define the subsets (interpolation spaces) of both Eqs.(2.120) and (2.121) in a single element as.

$$V^h = \left\{ w \in H^1(\Omega) | w|_{\Omega^e} \in P_m(\Omega^e) \forall e \text{ and } w = 0 \text{ on } \Gamma_D \right\} \tag{2.122}$$

$$S^h = \left\{ u \in H^1(\Omega) | u|_{\Omega^e} \in P_m(\Omega^e) \forall e \text{ and } u = u_D \text{ on } \Gamma_D \right\} \tag{2.123}$$

where P_m represents the finite element interpolation space. Hence based on Eqs.(2.123) and (2.123) we can state that $\boldsymbol{u} \in H^1_D$ and $\boldsymbol{w} \in H^1_0$. On the other hand the mass conservation test function q and the pressure solution p are required to be square integrable since their spatial derivatives do not appear in Eqs. (2.114) and (2.115) we conclude that both q and p belongs to $L^2(\Omega)$ space of functions. Likewise, the external forces $\mathbf{f}$ belong to the dual space of H^1 (since $\boldsymbol{w} \in H^1$) which is denoted as $H^{-1}(\Omega)$ as discussed in [9].

In preparation of using Newton-Raphson method to solve Eqs. (2.114) and (2.115) we introduce the residuals formulations of these equations. This is an important step especially when performing the linearlization step with Newton-Raphson method.

$$\begin{aligned} \mathbf{R}^m_{n+1}(\boldsymbol{u}, p) = &\int_\Omega \boldsymbol{w} \mathbf{f}_{n+1} \mathrm{d}\Omega - \int_\Omega \boldsymbol{w} \rho \partial_t \boldsymbol{u}_{n+1} \mathrm{d}\Omega - \int_\Omega \mu \nabla \boldsymbol{w} \nabla \boldsymbol{u}_{n+1} \, \mathrm{d}\Omega - \int_\Omega \boldsymbol{w} \rho \mathbf{a} \cdot \boldsymbol{\nabla} \boldsymbol{u}_{n+1} \mathrm{d}\Omega \\ &+ \int_\Omega \nabla \cdot \boldsymbol{w} P_{n+1} \, \mathrm{d}\Omega \end{aligned} \tag{2.124}$$

$$R^c_{n+1}(\boldsymbol{u}) = -\int_\Omega \rho q \nabla \cdot \boldsymbol{u}_{n+1} \, \mathrm{d}\Omega \tag{2.125}$$

As we have discussed in the previous section, there are key issues that arises when using the Galerkin method, this leads to the use of stabilization methods. Therefore the body-fitted formulation also uses the VMS framework were the velocity field is expressed as $\boldsymbol{u} = \boldsymbol{u}_h + \boldsymbol{u}_s$, where $\boldsymbol{u}_h$ is part of the

continuous solution that can be replaced by the finite element mesh can and $\boldsymbol{u}_s$ are the subscale models that can be modeled using the ASGS model [5]. This leads to the following derivation of the weak form with the stabilization terms (Note we have dropped the integrals for simplification purposes).

$$\begin{aligned}\mathbf{R}^m_{n+1}(\boldsymbol{u},p) &= \boldsymbol{w}f_{n+1} - \rho\boldsymbol{w}\partial t\boldsymbol{u}_{n+1} - \mu\nabla\boldsymbol{w}\nabla\boldsymbol{u}_{n+1} - \rho\boldsymbol{w}\mathbf{a}\cdot\boldsymbol{\nabla}\boldsymbol{u}_{n+1}\nabla\cdot\boldsymbol{w}P_{n+1} \\ &+ (\mathbf{a}\cdot\boldsymbol{\nabla}\boldsymbol{w})\tau_1(f_{n+1} - \rho\partial_t\boldsymbol{u}_n - \rho\mathbf{u}^{(i-1)}\cdot\nabla\boldsymbol{u}_{n+1} - \nabla P_{n+1}) - (\nabla\cdot\boldsymbol{w})\tau_2(\rho\nabla\cdot\boldsymbol{u}_{n+1})\end{aligned} \tag{2.126}$$

$$R^c_{n+1}(\boldsymbol{u},p) = -\rho q\nabla\cdot\boldsymbol{u}_{n+1} + (\nabla q)\tau_1(\boldsymbol{f}_{n+1} - \rho\partial_t\boldsymbol{u}_n - \rho\mathbf{a}\cdot\nabla\boldsymbol{u}_{n+1} - \nabla p_{n+1}) \tag{2.127}$$

where the stabilization coefficients for the monolithic solver are defined as (from [9])

$$\tau_1 = \frac{1}{\frac{\beta}{dt} + \frac{\nu}{h^2} + \frac{u}{h}} \quad , \quad \tau_2 = \frac{4h^4}{\frac{\nu}{h^2} + \frac{u}{h}} \tag{2.128}$$

with ν been the viscosity, $\mathbf{u}$ the velocity, β the mass stabilization coefficient h the element size and dt the time step.

2.4.5 Newton raphson method

To complete this discussion, we will write the linear system required to iteratively solve Eqs. (2.126) and (2.127) in matrix form. Using the classical Newton-Raphson scheme, the following expression is obtained (from [15]).

$$-\begin{bmatrix} \frac{\partial \mathbf{R}^m_{n+1}}{\partial \boldsymbol{u}_{n+1}} & \frac{\partial \mathbf{R}^m_{n+1}}{\partial p_{n+1}} \\ \frac{\partial R^c_{n+1}}{\partial \boldsymbol{u}_{n+1}} & \frac{\partial R^c_{n+1}}{\partial p_{n+1}} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{u}_{n+1} \\ \delta p_{n+1} \end{bmatrix} = \begin{bmatrix} \mathbf{R}^m_{n+1} \\ R^c_{n+1} \end{bmatrix}$$

2.4.6 Spatial discretization

To perform the spatial discretization we can rewrite the formulation in the previous section by replacing the continuous variables with the nodal values as follows.

$$\begin{bmatrix} \mathbf{K} & \mathbf{G} \\ \mathbf{D} & \mathbf{L} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{u}_{n+1} \\ \delta\mathbf{p}_{n+1} \end{bmatrix} = \begin{bmatrix} \mathbf{R}^{mo}_{n+1} \\ \mathbf{R}^c_{n+1} \end{bmatrix}$$

with

$$\mathbf{K} := -\frac{\partial \mathbf{R}^m_{n+1}}{\partial \boldsymbol{u}_{n+1}} \; ; \; \mathbf{G} := -\frac{\partial \mathbf{R}^m_{n+1}}{\partial \mathbf{p}_{n+1}} \tag{2.129}$$

$$\mathbf{D} := -\frac{\partial \mathbf{R}^c_{n+1}}{\partial \boldsymbol{u}_{n+1}} \; ; \; \mathbf{L} := -\frac{\partial \mathbf{R}^c_{n+1}}{\partial \mathbf{p}_{n+1}} \tag{2.130}$$

where finally the components of the matrices are expressed as [8]

$$\begin{aligned}\mathbf{K}_{ij} &= \int \rho\frac{1-\alpha_b}{\gamma\Delta t}N^u_i N^u_j + \mu\nabla N^u_i\nabla N^u_j + \rho N^u_i \boldsymbol{a}_k\frac{\partial N^u_j}{\partial x_k}\,\mathrm{d}\Omega \\ &+ \sum_e^{n_{elem}}\int\left(\boldsymbol{a}_k\frac{\partial N^u_i}{\partial x_k}\right)\tau_1\left(\rho\frac{1-\alpha_b}{\gamma\Delta t}N^u_j + \rho\boldsymbol{a}_k\frac{\partial N^u_j}{\partial x_k}\right) + \frac{\partial \mathbf{N}^u_i}{\partial x_i}\tau_2\frac{\partial \mathbf{N}^u_j}{\partial x_j}\,\mathrm{d}\Omega_\mathrm{e}\end{aligned} \tag{2.131}$$

$$\mathbf{G}_{ij} = -\int\frac{\partial N^u_i}{\partial x_i}N^p\,\mathrm{d}\Omega + \sum_e^{n_elem}\int\left(\boldsymbol{a}_k\frac{\partial N^u_i}{\partial x_k}\right)\tau_1\int\frac{\partial N^p}{\partial x_j}\,\mathrm{d}\Omega_\mathrm{e} \tag{2.132}$$

$$\mathbf{D}_{ij} = \int\rho N^p\frac{\partial N^u_j}{\partial x_j}\,\mathrm{d}\Omega + \sum_e^{n_elem}\int\frac{\partial N^p}{\partial x_i}\tau_1\left(\rho\frac{1-\alpha_b}{\gamma\Delta t}N^u_j + \rho\boldsymbol{a}_k\frac{\partial N^u_j}{\partial x_k}\right) \tag{2.133}$$

$$\mathbf{L}_{ij} = \sum_e^{n_elem}\int\frac{\partial N^p}{\partial x_i}\tau_1\frac{\partial N^p}{\partial x_j}\mathrm{d}\Omega_\mathrm{e} \tag{2.134}$$

here once again Ω represents the full domain, Ω_e a single element and N_i^u and N_p represent the shape functions corresponding to the i-th velocity component and the pressure, respectively. Therefore this summarizes the equations for the monolithic solver. As a conclusion on this solver we can deduce that the resulting linear equation system is solved directly for the pressure and velocity field in one big equation system which makes this solver very stable and very accurate. In terms of the time step, this solver allows the use of It is bigger time steps compared with its counter part the fractional step methods. The drawback is that the system is poorly conditioned which leads to high computational costs.

2.4.7 Boundary conditions

In order to solve the Navier-Stokes equations using CFD appropriate initial conditions and boundary conditions need to be applied. Therefore, boundary conditions select the solutions compatible with the environmental constraints. The choice of boundary conditions depends both the environment and the inherent nature of the flow. It is therefore not surprising that they are the most critical factor in the development of robust and efficient CFD methods. The most common boundary conditions in CFD are;

- **Inlet boundary condition** : For this type of boundary conditions the distribution of all the flow variable needs to be specified at the inlet boundary mainly the flow velocity. Intended for incompressible flows. The total (stagnation) properties of flow are not fixed. Stagnation properties vary to accommodate prescribed velocity distribution. Using in compressible flows can lead to non-physical results. In cases where neither the flow rate nor the velocity are known(e.g) buoyancy flows pressure can be used as an inlet boundary condition. In this case the one will need to define the total gauge pressure as follows;

$$p_{total} = p_{static} + \frac{1}{2}\rho v^2 \qquad for\, incompressible\, flows$$

  Where k is the ratio of the specific heats (c_p/c_v) and M is the Mach number.

- **No-slip wall boundary condition** : Used to bound fluid and solid regions. For example in viscous flows, the no-slip is enforced at walls where the tangential fluid velocity is set to be equal to the wall velocity. Also the velocity component is set to be zero.

- **Slip wall boundary condition**: This is similar to the no slip boundary condition except that in this case the velocity of the fluid does not change in accordance with the wall in the tangential direction. The velocity in the normal direction is set to zero.

- **Outlet boundary condition** Here the distribution of all flow variables needs to be specified at outlet boundaries. For example the use of the Zero pressure boundary condition to define the oulet flow on the outlet surface.

2.5 Body-fiited vs Embedded fomulations

Moving and deforming boundaries are a common occurrence in many engineering applications. The effects such interactions have on the flow can be significant and can determine the problem solution [17]. Therefore, when dealing with fluid dynamics related problems it is important to make sure that the

material motion of the fluid is well formulated. This problem becomes even more critical when dealing with Fluid Structure Interaction (FSI) problems as both the fluid and the structure fields have to be well formulated. In this work two formulations are used, the body fitted formulation and embedded formulation. A detailed description of these methods in the context of FSI is given in [4]. To elaborate more on these two formulations we take a look at the rigid body motion $\boldsymbol{u}$ of a structure within a discretized fluid domain as depicted in Figure 2.5.

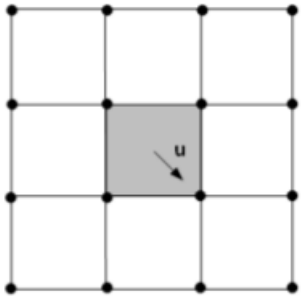


Figure 2.5: Rigid body motion of a structure (grey) in a fluid domain (Adopted from [4]).

2.5.1 Body-fitted formulation

The body-fitted formulation is usually the most commonly used approach in CFD for complex geometries. The type of body-fitted formulation used in this work utilizes the Eulerian description. Applying this approach to the description of the rigid motion $\boldsymbol{u}$ in Figure 2.5 allows the fluid nodes at the interface between the structure and the fluid to be forced to follow the movement of the structure as depicted in Figure 2.7. Although this formulation is widely used in the CFD community the definition of a suitable grid for a complex engineering geometry can be a laborious task and frequently requires a high degree of expertise and experience [18].

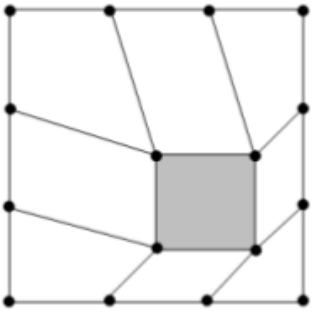


Figure 2.6: Body-fitted solution approach (Adopted from [4]).

2.5.2 Embedded formulation

The embedded formulation is another approach used to solve complex "dirty" geometries or moving boundary scenarios on arbitrary meshes. It is a non-body-fitted or fixed-mesh method where the boundary of the computational domain does not coincide with the boundary of the mesh. A key advantage of this formulation in comparison with the Body-fitted is the ability to deal with dirty geometries as it uses a distance algorithm that this capable to track the required interface of the geometry.

For the currently implemented embedded formulation the Eulerian description is utilized and hence when applied to the rigid motion problem in Figure 2.5, the structure moves independently from the fluid nodes locations as shown in Figure 2.6. Therefore, in general this non-boundary conforming technique allows to completely decouple the movement of the physical domain from the movement of the mesh. This allows the solutions to be computed without the need for complicated, inaccurate and computationally expensive mesh regeneration. Although there is no mesh distortion this convenience comes at a cost. By not having the boundaries coincide with the grid points, the Dirichlet boundary conditions are not easily enforced. An additional system is required to overcome this difficulty,which introduces some additional complexity and computational overhead to the underlying numerical scheme. The formulation used to tackle the issue of applying Dirichlet boundary condition will be reviewed further thereby introducing the non-slip embedded formulation that will be validated in this work. Note that there also other methods such as Other such as Penalty method, Nitsche's method, Lagrange multipliers that can be used to impose Dirichlet boundary conditions although for each method there are advantages and disadvantages.

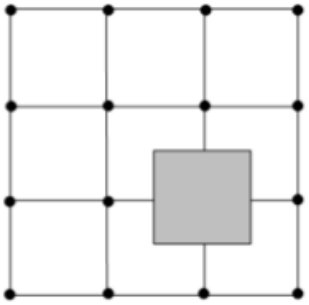


Figure 2.7: Embedded approach - The fluid and the structure mesh are completely decoupled from each other (Adopted from [4]).

2.6 Implementation of the Embedded No-Slip Boundary Condition Formulation in Kratos using the Modified Nitsche's Method

The imposition of Dirichlet boundary conditions in Embedded CFD methods has been well researched by Codina and Baiges [6]. In this paper they analyzed several possibilities of prescribing boundary conditions in the context of embedded methods. One key possibility that was proposed is the use of the Modified Nitsche's. This is also the method that has been implemented in Kratos hence refereed to as the Embedded No-Slip Boundary Condition Formulation in this thesis. This formulation is developed for flow problems (incompressible flow) and in particular for a problem involving the the scalar convection-diffusion-reaction equation. In this thesis we only present some key parts of the derivation as understood by the author, for a more vigorous explanation interested readers may refer to the above cited paper. The essential idea behind the derivation of this method is to use the degrees of freedom of certain nodes of the finite element mesh to minimize the difference between the exact and the approximated boundary condition. In accordance with [6] a problem described by the scalar convection−diffusion−reaction equation is used

to demonstrate the derivation process of the method. This equation and the boundary condition are

$$L(u) = -k\Delta u + \mathbf{a} \cdot \nabla u + su = f \quad \text{in } \Omega \tag{2.135}$$

$$u = \bar{u} \quad \text{on } \Gamma \tag{2.136}$$

where $k > o$, $\mathbf{a}$ is the advection velocity,f is the force, and $\bar{u}$ is the Dirichlet boundary condition. We

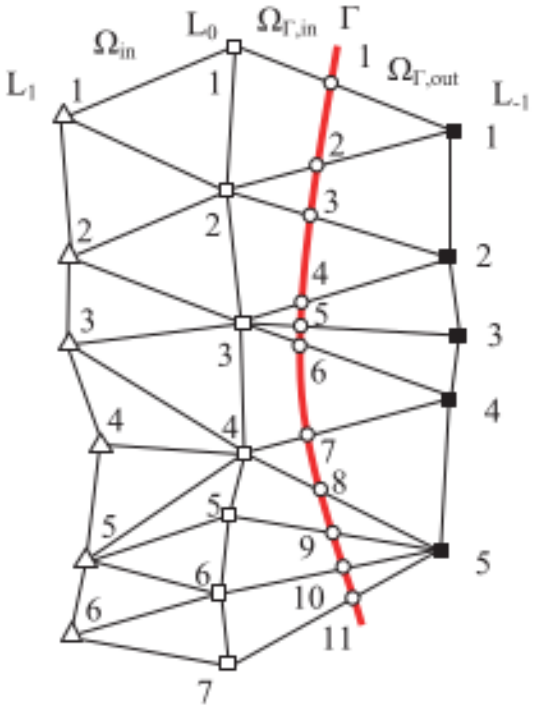


Figure 2.8: Domain setting (Adopted from [6])

consider the problem to be solved as depicted in Figure 2.8. Here the domain $\Omega \subset \mathbb{R}^d, d = 2, 3$ with boundary $\Gamma = \partial\Omega$ (thick curve in Figure 2.8) is covered by a mesh that occupies a domain $\Omega_h = \Omega_{\text{in}} \cup \Omega_\Gamma$ where $\Omega_{\text{in}} \subset \Omega$ is formed by the elements interior to Ω and Ω_Γ is formed by a set of elements cut by Γ. In turn let us split $\Omega_\Gamma = \Omega_{\Gamma,\text{in}} \cup \Omega_{\Gamma,\text{out}}$, where $\Omega_{\Gamma,\text{in}} = \Omega \cap \Omega_{\Gamma,\text{in}}$ and $\Omega_{\Gamma,\text{out}}$ is the interior of $\Omega_\Gamma \backslash \Omega_{\Gamma,\text{in}}$ Note that $\Omega = \Omega_{\text{in}} \cup \Omega_{\Gamma,\text{in}}$. Taking the finite element space V_h, Nitsches method applied to the problem defined by Eqs. (2.135) and (2.135) reads as: find $u_h \in V_h$ as

$$\begin{aligned} B(u_h, v_h) - k < \partial_n u_h, v_h >_\Gamma - k < u_h, \partial_n v_h >_\Gamma + \frac{\alpha k^*}{h} < u_h, v_h >_\Gamma \\ = < f, v_h >_\Omega - k < \bar{u}, \partial_n v_h > + \frac{\alpha k^*}{h} < \bar{u}, v_h > \end{aligned} \tag{2.137}$$

where the bilinear form is given below. Note that here we are using the compact notation to represent the integrals, hence for volume integrals we will use $(\bullet, \bullet)_\Omega$ and for boundary integrals $< \bullet, \bullet >_\Gamma$.

$$B(u_h, v_h) = k(\nabla u_h, \nabla v_h) + (\mathbf{a} \cdot \nabla u_h, v_h) + s(u_h, v_h) \tag{2.138}$$

The proposed method involves the rederivation of Nitsches method by considering the splitting of the test function and the solution as follows $v_h = v_{h,0} + v_{h,\Gamma}$ and $u_h = u_{h,0} + u_{h,\Gamma}$ then taking the test functions $v_h = v_{h,0}$ and $v_h = v_{h,\Gamma}$ and multiplying them with the differential equation (Eq.(2.135)). After that the

resulting set of equations can be added together to get Eq.2.137. These equations are given as:

$$B(u_{h,0}, v_{h,0}) - k < \partial_n u_{h,0}, v_{h,0} >_\Gamma + B(u_{h,\Gamma}, v_{h,0}) - k < \partial_n u_{h,\Gamma}, v_{h,0} >_\Gamma = < f, v_{h,0} >_\Omega \tag{2.139}$$

$$B(u_{h,0}, v_{h,\Gamma}) - k < \partial_n u_{h,0}, v_{h,\Gamma} >_\Gamma + B(u_{h,\Gamma}, v_{h,0}) - k < \partial_n u_{h,\Gamma}, v_{h,\Gamma} >_\Gamma = < f, v_{h,\Gamma} >_\Omega \tag{2.140}$$

$$-k < \partial_n u_{h,0}, v_{h,0} >_\Gamma - k < \partial_n u_{h,\Gamma}, v_{h,0} >_\Gamma = k < \bar{u}, v_{h,0} >_\Gamma \tag{2.141}$$

$$-k < \partial_n u_{h,0}, v_{h,\Gamma} >_\Gamma - k < \partial_n u_{h,\Gamma}, v_{h,\Gamma} >_\Gamma = k < \bar{u}, v_{h,\Gamma} >_\Gamma \tag{2.142}$$

$$\frac{\alpha k^*}{h} < u_{h,0}, v_{h,0} >_\Gamma + \frac{\alpha k^*}{h} < u_{h,\Gamma}, v_{h,0} >_\Gamma = \frac{\alpha k^*}{h} < \bar{u}, v_{h,0} >_\Gamma \tag{2.143}$$

$$\frac{\alpha k^*}{h} < u_{h,0}, v_{h,\Gamma} >_\Gamma + \frac{\alpha k^*}{h} < u_{h,\Gamma}, v_{h,\Gamma} >_\Gamma = \frac{\alpha k^*}{h} < \bar{u}, v_{h,\Gamma} >_\Gamma \tag{2.144}$$

where Eqs. 2.139 and 2.140 can be obtained by using integration by parts once the deferential equation is multiplied by $v_{h,0}$ and $v_{h,\Gamma}$ respectively. Considering the boundary condition $u_h = \bar{u}$ and taking the test functions as $k\partial_n v_{h,0}$ and $k\partial_n v_{h,\Gamma}$ we can obtain Eqs. 2.141 and 2.142. On the other hand Eqs. 2.141 and 2.142 are a weak form of the boundary condition $u_h = \bar{u}$ where the weighting functions are $v_{h,0}$ and $v_{h,\Gamma}$ respectively. Using the above equations Codina and Baiges [6] proposed a method that involves the use of only Eqs.(2.139) and (2.144). Furthermore with respect to stability they concluded that it is convenient to subtract Eq.(2.141) from Eq.(2.139). Thus the problem to be solved is : find $u_{h,0} \in V_{h,0}$ and $u_{h,\Gamma} \in V_{h,\Gamma}$ such that

$$\begin{aligned} &B(u_{h,0}, v_{h,0}) + B(u_{h,\Gamma}, v_{h,0}) - k < \partial_n u_{h,0}, v_{h,0} >_\Gamma - k < \partial_n u_{h,\Gamma}, v_{h,0} >_\Gamma \\ &+k < \partial_n v_{h,0}, u_{h,0} >_\Gamma + k < \partial_n v_{h,0}, u_{h,\Gamma} >_\Gamma = < f, v_{h,0} >_\Omega + k < \partial_n v_{h,0}, \bar{u} >_\Gamma \end{aligned} \tag{2.145}$$

$$\frac{\alpha k^*}{h} < u_{h,0}, v_{h,\Gamma} >_\Gamma + \frac{\alpha k^*}{h} < u_{h,\Gamma}, v_{h,\Gamma} >_\Gamma = \frac{\alpha k^*}{h} < \bar{u}, v_{h,\Gamma} >_\Gamma \tag{2.146}$$

To easy the process of implementation the J_2 functional is defined according to [6] as

$$J_2(u_{h,0}, u_{h,\Gamma}) = (\alpha k^*/h) \left\| u_{h,0} + u_{h,\Gamma} - \bar{u} \right\|^2_{L^2(\Omega)} \tag{2.147}$$

Hence for all $v_{h,0} \in V_{h,0}$ and $v_{h,\Gamma} \in V_{h,\Gamma}$ Eq. (2.146) can be equivalently written as

$$\delta_{(0,v_{h,\Gamma})} J_2(u_{h,0}, u_{h,\Gamma}) \tag{2.148}$$

This equation clearly shows that the component $u_{h,\Gamma}$ of the unknown is determined from the condition that the distance between $u_{h,0} + u_{h,\Gamma}$ and $\bar{u}$ is minimized by the norm of $L^2(\Gamma)$. Eqs. (2.145) and (2.146) form what we is refereed to in this thesis as the Modified Nitsche's method. It can be noted that Eq. (2.146) is simply the application of the penalty term on the nodes along L_{-1} in Figure 2.8. Consequently, the parameter $(\alpha k^*/h)$ here unnecessary as it has been introduced only to compare the resulting method with (2.137). The major differences between this Modified Nitsches method with respect to Nitsche (2.137) are elaborated in [6] as follows

- When Γ coincides with $\partial\Omega_h$ the boudary is imposed exactly provided that $\bar{u}$ is a finite element function
- There are no parameters to be tuned as $(\alpha k^*/h)$ can be canceled out in Eq.(2.146)
- The method is non-symmetric, even if B is symmetric.
- . The method is not well defined when Γ coincides with $\partial\Omega_{\text{in}}$

Implementation aspect

In order to implement this method Eqs.(2.145) and (2.146) needs to be expressed as a matrix form. To this end we suppose that the unknown u_h is interpolated as

$$u_h(\mathbf{x}) = \sum_{a=1}^{n_{\text{in}}} I^a_{\text{in}}(\mathbf{x}) U^a_{\text{in}} + \sum_{b=1}^{n_{\text{out}}} I^b_{\text{out}}(\mathbf{x}) U^b_{\text{out}} \tag{2.149}$$

$$= \mathbf{I}_{\text{in}}(\mathbf{x})\mathbf{U}_{\text{in}} + \mathbf{I}_{\text{out}}(\mathbf{x})\mathbf{U}_{\text{out}} \tag{2.150}$$

where I^a_{in} and I^b_{out} are the standard interpolation functions , n_{in} is the number of nodes in Ω_{in} including layer L_0 and n_{out} is the number of nodes in L_{-1} as sown in Figure 2.8. The main objective is the computation of U_{out}, that is done minimizing the functional that is

$$J_2(U_{\text{in}}, U_{\text{out}}) = \int_\Gamma (u_h(\mathbf{x}) - \bar{u}\mathbf{x})^2 = \int_\Gamma (\mathbf{I}_{\text{in}}(\mathbf{x})\mathbf{U}_{\text{in}} + \mathbf{I}_{\text{out}}(\mathbf{x})\mathbf{U}_{\text{out}} - \bar{u}(\mathbf{x}))^2 \tag{2.151}$$

it then considered that

$$\frac{\partial J_2}{\partial \mathbf{U}_{\text{out}}} = 0 \Rightarrow \mathbf{M}_\Gamma \mathbf{U}_{\text{out}} = \mathbf{f}_\Gamma - \mathbf{N}_\Gamma \mathbf{U}_{\text{in}} \tag{2.152}$$

where

$$\mathbf{M}_\Gamma = \int_\Gamma \mathbf{I}^{\text{t}}_{\text{out}}(\mathbf{x})\mathbf{I}_{\text{out}}(\mathbf{x}), \quad \mathbf{f}_\Gamma = \int_\Gamma \mathbf{I}^{\text{t}}_{\text{out}}(\mathbf{x})\bar{u}(\mathbf{x}), \quad \mathbf{N}_\Gamma = \int_\Gamma \mathbf{I}^{\text{t}}_{\text{out}}(\mathbf{x})\mathbf{I}_{\text{in}}(\mathbf{x}) \tag{2.153}$$

Now suppose the matrix from of Eq.(2.145) is

$$\mathbf{K}_{\text{in,in}}\mathbf{U}_{\text{in}} + \mathbf{K}_{\text{in,out}}\mathbf{U}_{\text{out}} = \mathbf{F}_{\text{in}} \tag{2.154}$$

where the matrices $\mathbf{K}_{\text{in,in}}$ and $\mathbf{K}_{\text{in,out}}$ extend only over Ω_{in}. Also $\mathbf{U}_{\text{out}}$ in this case is used to interpolate u_h in the subdomain Ω_{in}. With that, inserting Eq.(2.153) into Eq.(2.154) results in

$$(\mathbf{K}_{\text{in,in}} - \mathbf{K}_{\text{in,out}}\mathbf{M}_\Gamma^{-1}\mathbf{N}_\Gamma)\mathbf{U}_{\text{in}} = \mathbf{F}_{\text{in}} - \mathbf{K}_{\text{in,out}}\mathbf{M}_\Gamma^{-1}\mathbf{f}_\Gamma \tag{2.155}$$

which would be the ideal system to solve but as noted by Codina and Baiges [6] the matrix $\mathbf{M}_\Gamma$ is not diagonal hence they proposed an iterative scheme of the form

$$\mathbf{K}_{\text{in,in}}\mathbf{U}^{\text{k}}_{\text{in}} = \mathbf{F}_{\text{in}} - \mathbf{K}_{\text{in,out}}\mathbf{U}^{k-1}_{\text{out}} \tag{2.156}$$

$$\mathbf{M}_\Gamma \mathbf{U}^k_{\text{out}} = \mathbf{f}_\Gamma - \mathbf{N}_\Gamma \mathbf{U}^k_{\text{in}} \tag{2.157}$$

where k is an iteration counter. Therefore, the matrix form Eqs. (2.154) and (2.153) (represents the problem (2.145) - (2.146)) can solved in a coupled way as.

$$\begin{bmatrix} \mathbf{K}_{\text{in,in}} & \mathbf{K}_{\text{in,out}} \\ \mathbf{N}_\Gamma & \mathbf{M}_\Gamma \end{bmatrix} \begin{bmatrix} \mathbf{U}_{\text{in}} \\ \mathbf{U}_{\text{out}} \end{bmatrix} = \begin{bmatrix} \mathbf{F}_{\text{in}} \\ \mathbf{f}_\Gamma \end{bmatrix} \tag{2.158}$$

This represent the implementation of the method. Furthermore, although not shown here, Codina and Baiges [6] noted that the use of a diagonal matrix for $\mathbf{M}_\Gamma$ leads to a scalar equation

$$\left(\int_{\Gamma^b_{\text{out}}} I^b_{\text{out}}(\mathbf{x}) I^b_{\text{out}}(\mathbf{x}) \right) U^b_{\text{out}} = \int_{\Gamma^b_{\text{out}}} I^b_{\text{out}}(\mathbf{x})\bar{u}(\mathbf{x}) - \int_{\Gamma^b_{\text{out}}} I^b_{\text{out}}(\mathbf{x})\mathbf{I}_{\text{in}}(\mathbf{x})\mathbf{U}_{\text{in}} \tag{2.159}$$

for which they concluded that the implementation of the method can be done as elaborated above but in this case considering the diagonal matrix of $\mathbf{M}_\Gamma$.

2.7 Laminar and Turbulent parameters

There are four fundamental benchmark problems that will be used for validation in this study. The flow characteristics of the models are divided into laminar and turbulent flows (see Figure 2.9). The key difference between the two is that in laminar flow each particle of the fluid follows a streamline and this streamline does not interfere with other streamlines, but this is not the case in turbulent flows. One result of laminar flow is that the velocity of the fluid is constant at any point in the fluid while in turbulent flow velocity components randomly fluctuate. Some of the parameters that will be used to interpret the results produced by these types flows are discussed in this section.

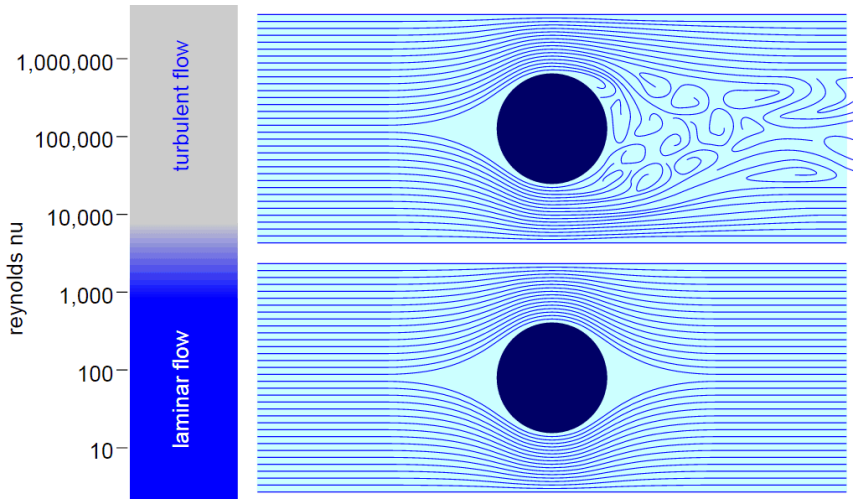


Figure 2.9: Flow-Regime

2.7.1 Reynolds number

This number is named after Osborne Reynolds who discovered in 1883 in 1883. It is a dimensionless number that predicts whether the fluid flow would be laminar or turbulent based on static and dynamic properties such as velocity, length, dynamic viscosity, and also characteristics of the flow. Its given by

$$Re = \frac{\rho U L}{\mu} \tag{2.160}$$

where ρ is the fluid's density and μ its dynamic viscosity and U is a characteristic velocity of the flow. This number is also widely used on Navier stokes equations to truncate mathematical models. When $Re \to \infty$ The terms in the Navier-stokes equations are drooped since the viscous effects are presumed to be negligible. The resulting simplified form of the NSE are called the Euler equations. Further more when $Re \ll 1y$ the inertial effects are presumed negligible which results in the formation the stokes flow.

2.7.2 Mean Value

Within the turbulence regime the transport variables such as velocity,temperature,pressure (u,v,w,T,p) always vary with time. A good example of this phenomena is difference that is observed in the magnitude and direction of the time-averaged instantaneous velocity vector with that of the time averaged velocity. While the time averaged velocity can be depend on time or not (see Figure 2.10) the instantaneous velocity in a turbulent flow is always time dependent. Therefore for any given location or point(x,y,z)

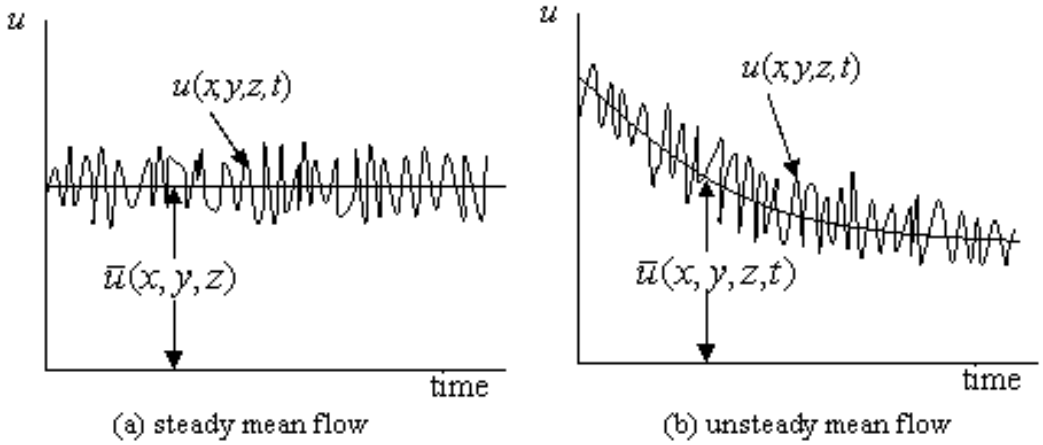


Figure 2.10: Time dependence of velocity component in the x-direction in a turbulent flow

at time t the local instantaneous velocity can be summed up as a summation of its mean value and fluctuations. That is

$$u = \bar{u}(x, y, z, t) + \acute{u}(x, y, z, t) \tag{2.161}$$

where

$$\bar{u}(x, y, z, t) = \frac{1}{\Delta t} \int_{t}^{t+\Delta t} u(x, y, z, t) dt \tag{2.162}$$

represents the time-averaged velocity at point (x,y,z). Note that of the case of steady flow the fluctuations are considered to be zero. In the same way the velocity in v, w directions can be evaluated. This is also similar with other parameters such pressure.

2.7.3 Mean Pressure coefficient

The pressure coefficient is a dimensionless number which is used to describe the relative pressures throughout a flow field in fluid dynamics. For each point in a fluid flow field there exists a unique pressure coefficient, C_p which is given as

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho_\infty U_\infty^2} \tag{2.163}$$

where

p is the static pressure at the point where the pressure coefficient is being evaluated

p_∞ is the static pressure in the freestream

ρ_∞ is the freestream fluid density

U_∞ is the freestream velocity of the fluid

2.7.4 Drag and Lift Coefficient

These two parameters are widely used engineering and hence it is vital to understand them. The drag coefficient is used to quantify the resistance of a body in a fluid environment. Likewise, lift coefficient relates the lift generated by a lifting body to the fluid density around the body, the fluid velocity and an associated reference area. Both of these quantities can be derived by considering a flow past a cylinder

as shown in Figure 2.11. Let us consider the derivation of the drag coefficient. From fundamental fluid dynamics, it is known that when the flow comes into contact with cylinder it exerts a resultant force on the cylinder. This force is a combination of the pressure force and the friction force [29] and its referred to as the mean drag.

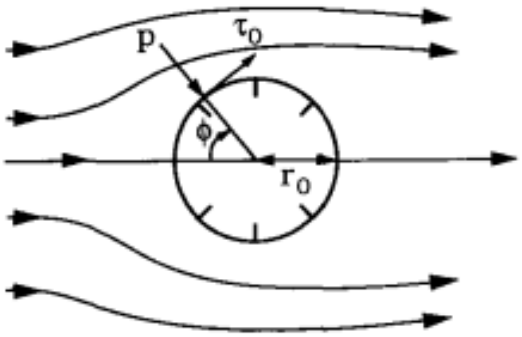


Figure 2.11: Components of the force acting on the cylinder (Adopted from [29])

The pressure force F_p and the friction force F_f can be expressed inform an integral based on the above figure as

$$F_p = \int_o^{2\pi} P\cos(\phi) r_0 d\phi \qquad \text{and} \qquad F_f = \int_o^{2\pi} \tau_0 \sin(\phi) r_0 \, d\phi \tag{2.164}$$

here as depicted in the figure 2.11 p is the pressure and τ_0 is the wall stress acting on the cylinder surface. Note that these are considered to be time-average values. With that said, the total mead drag read as

$$F_D = F_p + F_f \quad \Rightarrow F_D = \int_o^{2\pi} \left(P\cos(\phi) + \tau_0 \sin(\phi)\right) r_0 \, d\phi \tag{2.165}$$

Now, although it is not fully elaborated here, it can be shown according to Sumer en al [29] that the pressure contribution from potential flow theory is given as

$$p - p_0 = \frac{1}{2}\rho U^2 (1 - 4sin^2\phi) \tag{2.166}$$

where p_0 is the hydrostatic pressure, hence Eqs. 2.165 can be written as

$$\frac{F_D}{\frac{1}{2}\rho D U^2} = \int_o^{2\pi} \left[\frac{p - p_0}{\rho U^2}\cos(\phi) + \left(\frac{\tau_0}{\rho U^2}\right)\sin(\phi)\right] d\phi \tag{2.167}$$

Where D is the diameter of the cylinder. In the literature the right-hand-side of this equation is written inform of the drag coefficient C_D. The is due to the fact, for a smooth cylinder both the pressure term and the wall shear stress term are functions of the Re number hence the right hand side of Eq. 2.167 is a function of the Re number as well. Hence Eq. 2.167 can rewritten using C_D to get the mean drag coefficient as

$$\frac{F_D}{\frac{1}{2}\rho D U^2} = C_D \tag{2.168}$$

which can be expressed together with the mean lift coefficient as

$$C_D = \frac{2F_D}{\rho U_\infty^2 A_D} \quad , \quad C_L = \frac{2F_L}{\rho U_\infty^2 A_L} \tag{2.169}$$

where F_D and F_L represents the drag and lift forces respectively, A_D and A_L represent the drag and lift areas. Likewise, U_∞ is considered as the free-stream velocity of the fluid and ρ represents the the free-stream fluid density.

Chapter 3

Methodology

In this chapter we present the softwares and formulations that have been used in this work. For simulating the validation problems, two softwares have been used in a cooperative manner, these have been Kratos and Pre-Post processor GiD softwares.Furthermore, to understand the formulation used to track the interface in the embedded formulation the continuous distance function is also presented. Furthermore we will also discuss the parallelization techniques that have been used in solving the validation problems.

3.1 Open source Kratos Multi-physics

As the need to solve multifaceted physics problems continues to increase both in engineering and science, the need for powerful multi-physics codes increases as well. Currently there are many conventional finite elements codes that are been used both in the academic and industrial sectors. The only drawback of these codes is that most of them are optimized for solving single field problems. Multi-physics softwares such as Kratos have been developed to solve this problem by been able to externally manage different multiphysics problems.

Kratos has been developed through the use of object oriented programming. The main goal of an object-oriented structure is to split the whole problem into several objects and to define their interfaces. These objects are defined using the general finite element approach [11]. This approach was selected because of the originally intended idea of Kratos, which was to create a finite element environment for multidisciplinary problems [10]. The entire code has been developed mainly through the use of C++ and Python programming languages by a group of researchers at CIMNE and the Technical University of Munich. Kratos is made up of different applications as shown in Figure 3.1. All these applications are used to solve different types of problems. For example our application of interest which is the fluid dynamics application provides the capability of solving incompressible fluids related problems.

Most of these applications in Figure 3.1 have the finite element core in common which handles

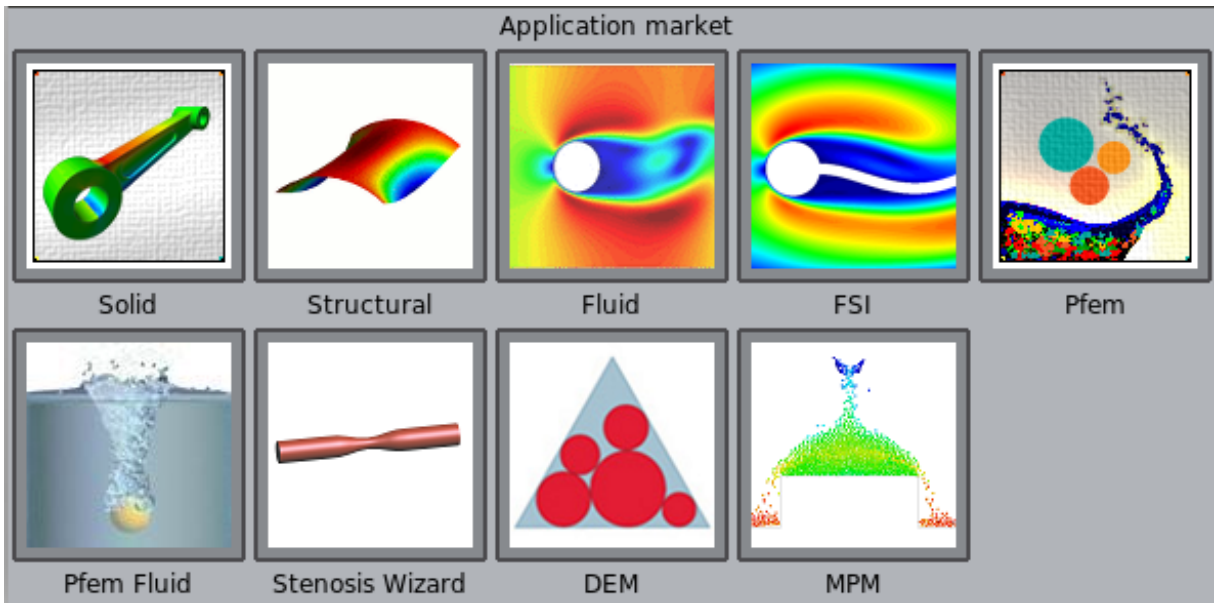


Figure 3.1: Kratos applications

the solution process, discretization and the numerical description on element level . Therefore in order to insure flexibility between the applications, the communication between these applications is managed by an interface which is also controlled by the Kratos kernel [11]. These relations are illustrated in Figure 3.2 which shows the main Kratos class structure. More in information on Kratos can be found on its GitHub account[1]

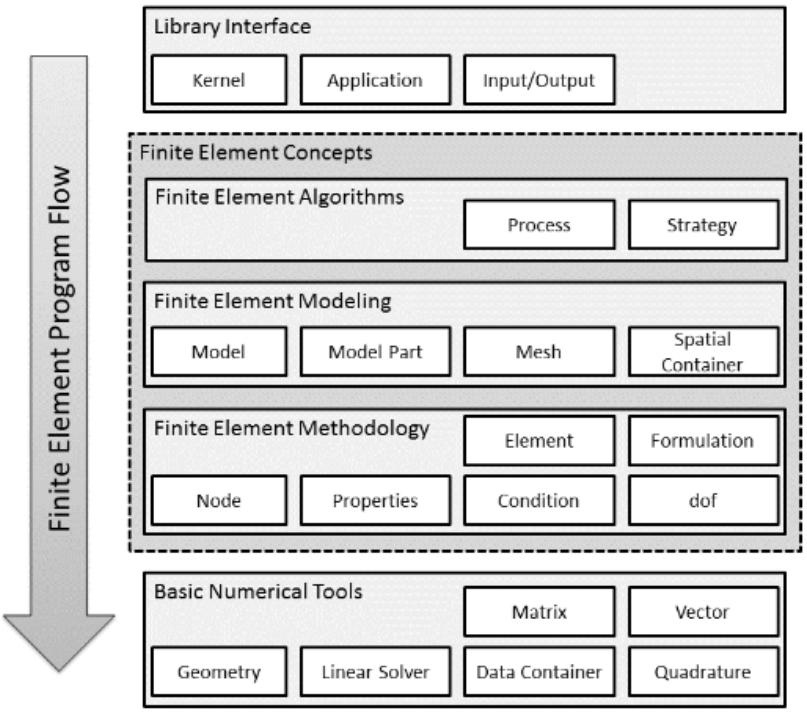


Figure 3.2: Essential classes and their main objects in Kratos Adopted from [4]

[1] https://github.com/KratosMultiphysics/Kratos

3.1.1 Fluid dynamics application

All the validation problems in this work were simulated using the fluid dynamics application. Within this application is the Body-fitted, Embedded and potential fluid applications as shown in Figure 3.3. For our work we only used the 2D and 3D formulations of both the Body-fitted and Embedded applications.

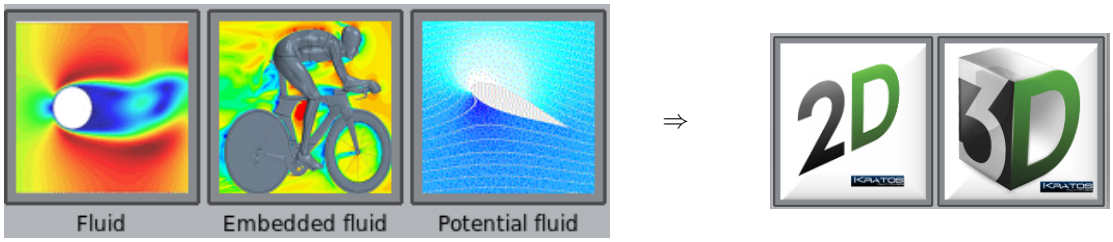


Figure 3.3: Kratos Fluid dynamic application

3.2 Personal Pre-Post processor GiD software

This is an interactive graphical user interface software developed by CIMNE for pre and post-processing of numerical simulations. Apart from this it is also widely used to generate mesh files suitable for several numerical methods such as finite element, finite volume or finite difference, particle based or meshless methods among others. With regards to the output files GiD is capable of writing output information for a given simulation in many popular file formats. As GiD is used for general purposes it has been designed to accommodate different types of solvers. This allows for the definitions of configurations depending on the analysis and and other specifications of each solver. The connection between the solver and GiD are depicted in Figure 3.4 In order to use GiD for a particular simulation, various types of problem

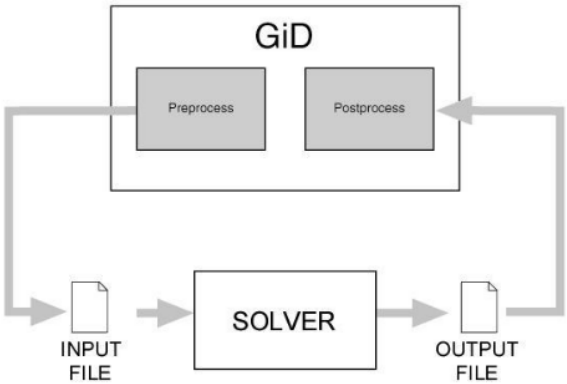


Figure 3.4: Essential connections between GiD and a solver

related data such as the definition of the geometry, materials, conditions, solution information and other parameters needs to be defined in GiD. To accomplish this task all the information files related to the analysis are defined as a `Problem Type`. Therefore, Kratos is one of the `Problem Type` defined in GiD and the essential connections between Kratos and GiD are shown in Figure 3.5. This is also the procedure followed in the study. More elaborative information on Gid can be found on [2].

[2] www.gidhome.com

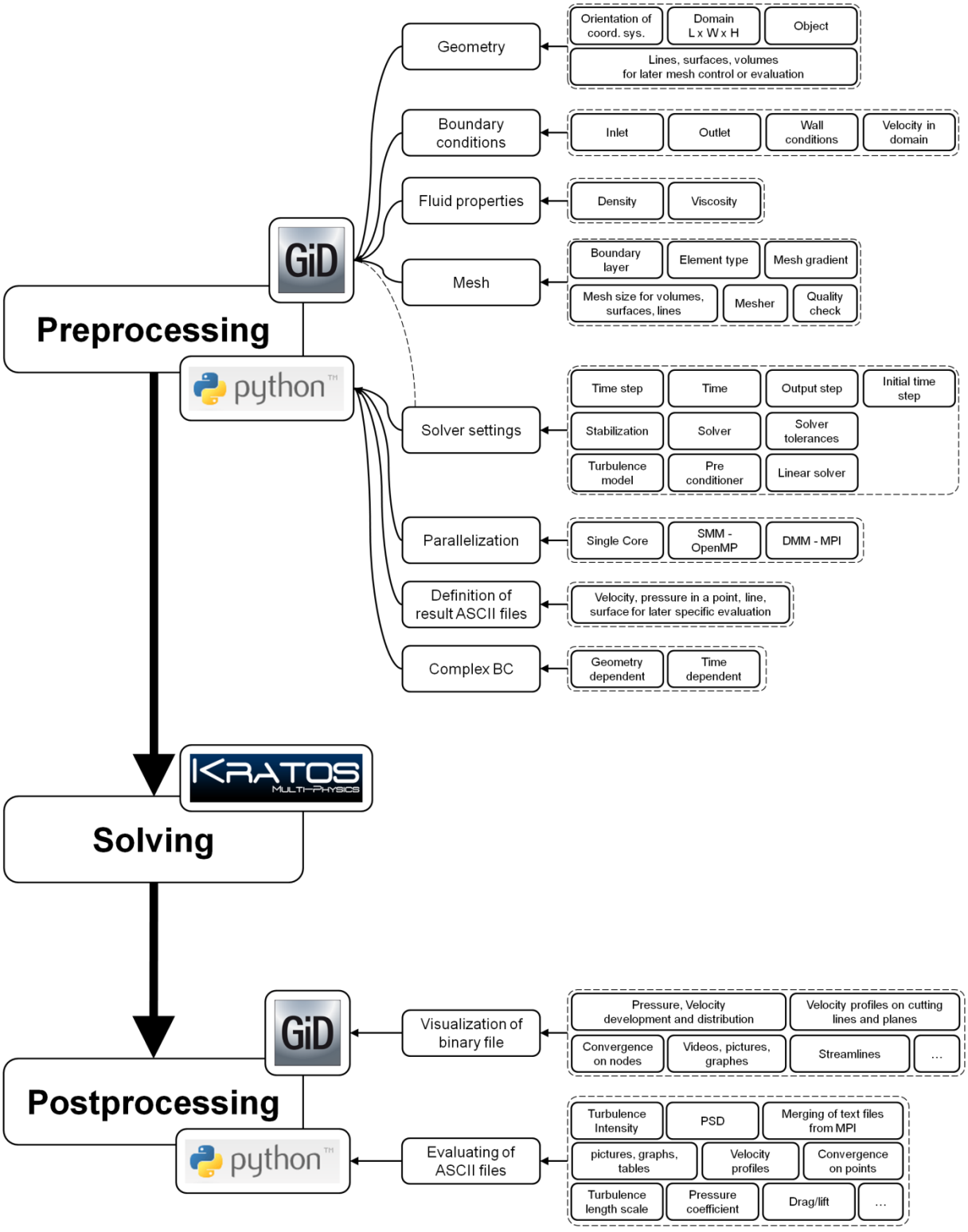
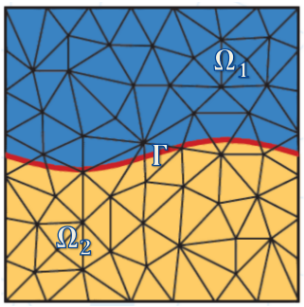


Figure 3.5: Essential connections between Kratos and GiD (Adopted from [15])

3.3 Interface tracking

3.3.1 Continuous Distance Function

Given a body-fitted computational mesh, the embedded formulation provides a more robust approach when dealing with moving structures or obstacles of which the exact location in the numerical domain is unknown. To perform this task, a distance function (e.i the level set function) is employed. Level-set methods are used for example in Extended finite element method(XFEM) to describe material interfaces in problems that may involve two or more different kinds of materials, voids, free-surfaces, etc. In most of these applications the definition of the level set function depends on the type of problem that needs to be solved, the required information on the interface and the various types of operations that ought to be carried out on the domain using the information from the interface. As an example the level set function ($\phi(x)$) for a problem involving a divided domain in Figure 3.6 is given below as follows;



$$\phi(x) = \begin{cases} > 0, & \text{if } x \in \Omega_1 \\ 0, & \text{if } x \in \Gamma \\ < 0, & \text{if } x \in \Omega_2 \end{cases} \tag{3.1}$$

Figure 3.6: Divided domain

The continuous distance function ($d(\mathbf{x})$) used in this work, operates just like the level set function, hence given a body-fitted mesh this function allows to classify fluid nodes as either inside ($d < 0$) or outside ($d > 0$) the embedded structure. The distance ($d(\mathbf{x})$ along the structure is taken to be zero. The continuous distance function is used here since the No-slip Embedded formulation which is used in this work has been implemented for bodies that have a well-defined internal volume (e.g. any aerofoil geometry). On the other hand another distance function, often referred to as the *discontinuous distance function*, is used for bodies without internal volume (e.g. boat sails). The discontinuous distance functions will not be discussed here but further information can be found in [4]. The continuous distance function is elaborated figuratively in Figure 3.7 where the distance of each node from the embedded structure is calculated and assigned a sign in accordance with the normal orientation of the structure.

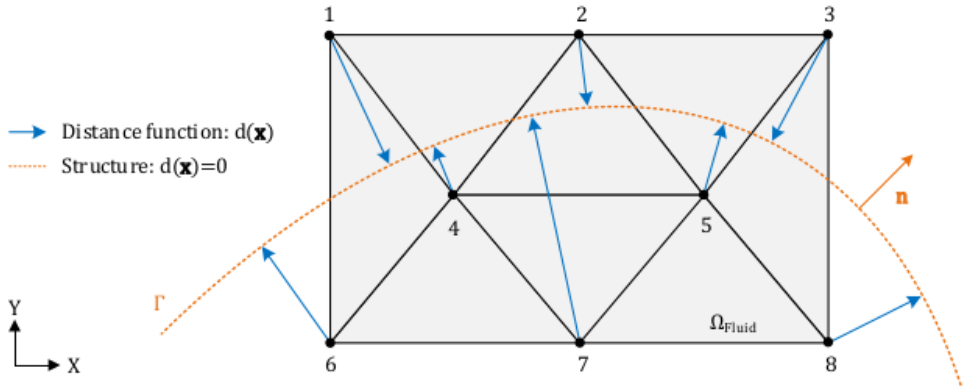


Figure 3.7: Concept of a distance function within a level set approach Adopted from [4]

To demonstrate this process we look at tracking the interface of a 2D circular structure (cylinder) embedded inside a domain as shown in Figure 3.8

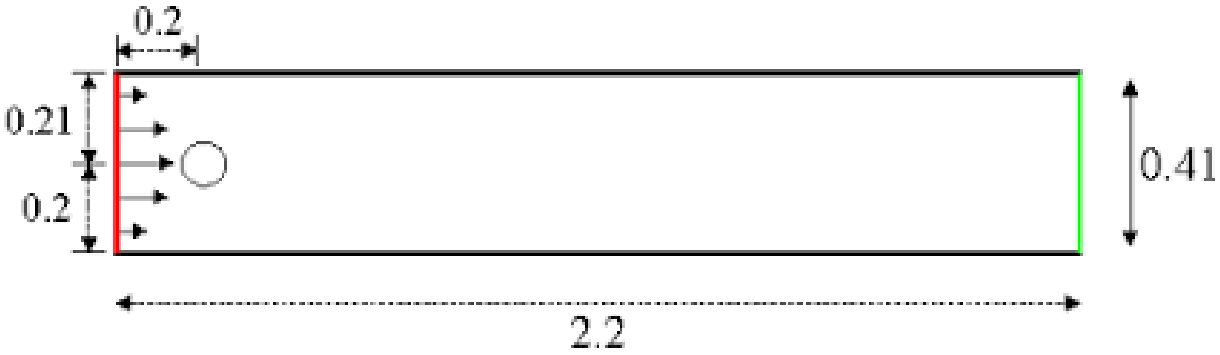


Figure 3.8: 2D problem of a flow over a cylinder

This process basically requires that the entire rectangular domain in Figure 3.8 be fully meshed and then a distance function as described below is applied to track the physical interface of the cylinder. Note with this distance function we assume that the center of the cylinder is at (0.0,0.0) in Cartesian coordinates and also the radius of the cylinder is taken to be 0.05 . For this a distance function can be written as

$$\phi(r) = \sqrt{(x^2 + y^2)} - \mathrm{r}$$

Where this equation is set as a loop to go through all the nodes in the mesh. The resulting distance variable ϕ can be printed as shown in Figure 3.9.

Figure 3.9: Cylinder interface tracking using a continuous distance function

The colormap in the above figure corresponds to the specific value of the distance function. As it can be seen the negative values which are depicted by the dark blue color, represents the nodes inside the physical structure (circle). The structure itself is represented with a distance value of zero. Likewise the fluid nodes outside the structure are all positive hence the distance value keeps on increasing as shown in the figure.

3.3.2 Distance modification algorithm

As the Embedded formulation depends on the interface cuts, problems might arise if there bad intersections in the mesh. This will directly affect the distance function which may eventually compromise the convergence of the embedded solution. To avoid this issue a distance modification algorithm is employed. As shown in Figure 3.10 when distance value is less than a relative tolerance for which such a relative tolerance is computed as a constant (typically 10^{-4} -10^{-5}) times the element size. Therefore the distance field is corrected by slightly modifying its position(from the dark line to the dotted line). This is done by setting minus the threshold value, that is to say we deactivate the almost empty elements.

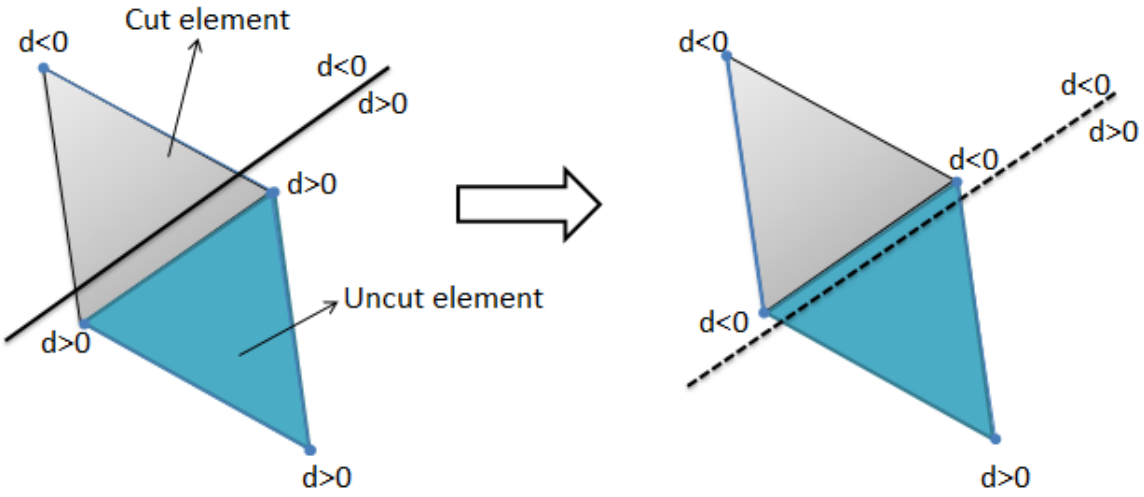


Figure 3.10: Interface tracking using a continuous distance function

3.4 Calculation of drag inside the embedded element

Here we review the drag calculation utility that was implemented in Kratos by the author. In the embedded formulation the process of calculating the drag around cut elements is fundamentally based on equation (3.2). This process begins with the computation of drag within each cut element in the mesh. Here the process includes getting the positive interface Gauss points and then computing the drag (3.3). This process involves the use of the constitutive law on $\boldsymbol{\tau}$ and the nodal interpolation of the pressure values p$\mathbf{I}$ on the Gauss points. The integration process used is the well known Gauss quadrature integration. Then afterwards the summation of all the elemental drag values is carried out.

$$Drag = \int_{\Gamma} \boldsymbol{\sigma} \cdot \mathbf{n} \ \ d\Gamma \tag{3.2}$$

where as discussed in chapter 2

$$\boldsymbol{\sigma} = -p\mathbf{I} + \boldsymbol{\tau} \tag{3.3}$$

This process of calculating drag is demonstrated analytically below using a single cut element as shown in Figure (3.11). Here N_0, N_1 and N_2 are the polynomial based shape functions located at the three nodes of the triangular element. Using the standard 2 points Gauss−Legendre quadrature rule, the coordinates

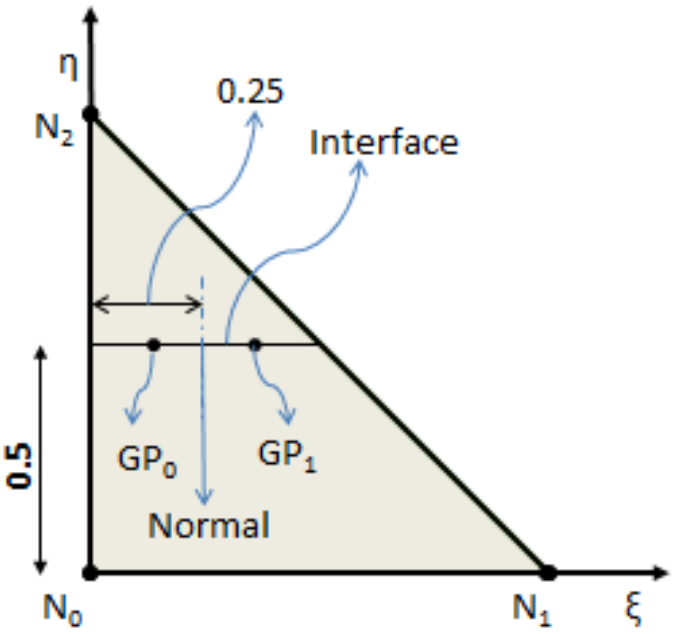


Figure 3.11: 2-D triangular cut element

weights of the two Gauss points along the interface are calculated as

$$GP_o \Rightarrow \xi^0 = 0.25\left(\frac{\sqrt{3}-1}{\sqrt{3}}\right) = 0.10566 \ , \eta^0 = 0.5 \quad w_0 = 0.25 \tag{3.4}$$

$$GP_1 \Rightarrow \xi^1 = 0.25\left(\frac{\sqrt{3}+1}{\sqrt{3}}\right) = 0.39433 \ , \eta^1 = 0.5 \quad w_1 = 0.25 \tag{3.5}$$

To perform the analytical calculation, the pressure and velocity values at the three nodes (labeled using Shape function positions) are assumed to be

$$N_0 \Rightarrow P_0 = 1, V_0 = (1.0, 1.0, 0.0) \tag{3.6}$$

$$N_1 \Rightarrow P_1 = 2, V_1 = (2.0, 2.0, 0.0) \tag{3.7}$$

$$N_2 \Rightarrow P_2 = 3, V_2 = (3.0, 3.0, 0.0) \tag{3.8}$$

Using (3.2) and expanding it for an incompressible Newtonian fluid by replacing σ with (3.3) gives

$$\sigma = \int_{\partial\Omega} \left(-p\mathbf{I} + 2\mu\left(\frac{1}{2}\left(\boldsymbol{\nabla u} + (\boldsymbol{\nabla u})^T\right)\right)\right)\mathbf{n}\, d\Gamma \tag{3.9}$$

The integration of this equation is performed by first considering the pressure contribution and then afterwards the contribution coming from the velocity terms is considered. The pressure contribution is calculated by integrating (3.10) at each Gauss point. This process is performed by interpolating each of the pressure values at the three nodes using the standard shape functions given in (3.11)

$$\sigma_p = \int_{\partial\Omega} (-p\mathbf{I}) \cdot \ \mathbf{n}\, d\Gamma \tag{3.10}$$

The polynomial shape functions at the three nodes

$$N_0 = 1 - \xi - \eta \quad , \quad N_1 = \xi \quad , \quad N_2 = \eta \tag{3.11}$$

Which leads to the expansion of the pressure contribution terms at the two Gauss point as

$$P^{GP_0} = N_0(x^{GP_0})p_0 + N_1(x^{GP_0})p_1 + N_2(x^{GP_0})p_2 \tag{3.12}$$

$$P^{GP_1} = N_0(x^{GP_1})p_0 + N_1(x^{GP_1})p_1 + N_2(x^{GP_1})p_2 \tag{3.13}$$

Using the above equations and the information at each Gauss point given in (3.4) and (3.4) together with pressure values from (3.6),(3.7) and (3.8) we see that the total pressure contribution is calculated as

$$
\begin{aligned}
&At\ \ GP_0\ \ (N_0 = 0.39434, N_1 = 0.10566, N_2 = 0.5) \\
&\quad \therefore P^{GP_0} = [-0.25 * (0.39434 * 1 + 0.10566 * 2 + 0.5 * 3)] * [0, -1, 0]^T = 0.526415 \\
&At\ \ GP_1\ \ (N_0 = 0.10566, N_1 = 0.39434, N_2 = 0.5) \\
&\quad \therefore P^{GP_0} = [-0.25 * (0.10566 * 1 + 0.39434 * 2 + 0.5 * 3)] * [0, -1, 0]^T = 0.598585
\end{aligned}
$$

which leads to

$$
\sigma_p = \int_{\partial\Omega} (-p\mathbf{I}) \cdot \ \mathbf{n}\ d\Gamma \Rightarrow 0.526415 + 0.598585 = [0, 1, 125, 0]^T \tag{3.14}
$$

$$
\sigma_\tau = \int_{\partial\Omega} 2\mu \left(\frac{1}{2} \left(\boldsymbol{\nabla u} + (\boldsymbol{\nabla u})^T \right) \right)\ \mathbf{n}\ d\Gamma \tag{3.15}
$$

With the pressure contribution already calculated, the contribution of the velocity terms are calculated by integrating (3.15). here we see that

$$
\left(\boldsymbol{\nabla u} + (\boldsymbol{\nabla u})^T \right) = \begin{bmatrix}
\dfrac{\partial u_1}{\partial x} & \dfrac{1}{2}\left(\dfrac{\partial u_2}{\partial x} + \dfrac{\partial u_1}{\partial y} \right) & \dfrac{1}{2}\left(\dfrac{\partial u_3}{\partial x} + \dfrac{\partial u_1}{\partial z} \right) \\
\dfrac{1}{2}\left(\dfrac{\partial u_1}{\partial y} + \dfrac{\partial u_2}{\partial x} \right) & \dfrac{\partial u_2}{\partial y} & \dfrac{1}{2}\left(\dfrac{\partial u_3}{\partial y} + \frac{\partial u_2}{\partial z} \right) \\
\dfrac{1}{2}\left(\dfrac{\partial u_1}{\partial z} + \dfrac{\partial u_3}{\partial x} \right) & \dfrac{1}{2}\left(\dfrac{\partial u_2}{\partial z} + \dfrac{\partial u_3}{\partial y} \right) & \dfrac{\partial u_3}{\partial z}
\end{bmatrix} \tag{3.16}
$$

Notice that we are dealing with a 2D element hence all the velocity components in the $z-$direction are zero, which reduces the size of (3.16) to a 2x2 matrix.This leads to the following expansion of each term inside the new matrix as follows.

$$
\frac{\partial u_x}{\partial x} = \frac{\partial N_0}{\partial x} u_{0,x} + \frac{\partial N_1}{\partial x} u_{1,x} + \frac{\partial N_2}{\partial x} u_{2,x} \tag{3.17}
$$

$$
\frac{\partial u_x}{\partial y} = \frac{\partial N_0}{\partial y} u_{0,x} + \frac{\partial N_1}{\partial y} u_{1,x} + \frac{\partial N_2}{\partial y} u_{2,x} \tag{3.18}
$$

where once again $N_0, N_1, and N_2$ are the polynomial based shape functions. The derivatives of these shapes functions are equal at both Gauss points.

$$
\begin{bmatrix}
\dfrac{\partial N_0}{\partial x} & \dfrac{\partial N_0}{\partial y} \\
\dfrac{\partial N_1}{\partial x} & \dfrac{\partial N_1}{\partial y} \\
\dfrac{\partial N_3}{\partial x} & \dfrac{\partial N_3}{\partial y}
\end{bmatrix} = \begin{bmatrix}
-1 & -1 \\
1 & 0 \\
0 & 1
\end{bmatrix} \tag{3.19}
$$

Using the values of velocity from (3.6),(3.7) and (3.8) for all the three nodes gives

$$
\frac{\partial u_1}{\partial x} = \frac{\partial N_0}{\partial x} u_{0,1} + \frac{\partial N_1}{\partial x} u_{1,1} + \frac{\partial N_2}{\partial x} u_{2,1} = 1 \tag{3.20}
$$

$$
\frac{\partial u_2}{\partial x} = \frac{\partial N_0}{\partial x} u_{0,2} + \frac{\partial N_1}{\partial x} u_{1,2} + \frac{\partial N_2}{\partial x} u_{2,2} = 1 \tag{3.21}
$$

$$
\frac{\partial u_1}{\partial y} = \frac{\partial N_0}{\partial y} u_{0,1} + \frac{\partial N_1}{\partial y} u_{1,1} + \frac{\partial N_2}{\partial y} u_{2,1} = 2 \tag{3.22}
$$

$$
\frac{\partial u_2}{\partial y} = \frac{\partial N_0}{\partial y} u_{0,2} + \frac{\partial N_1}{\partial y} u_{1,2} + \frac{\partial N_2}{\partial y} u_{2,2} = 2 \tag{3.23}
$$

Using the values from (3.23) and then (3.16) becomes

$$\begin{bmatrix} -1 & \frac{3}{2} \\ \frac{3}{2} & 1 \end{bmatrix} \tag{3.24}$$

$$\left(\boldsymbol{\nabla u} + (\boldsymbol{\nabla u})^T\right) = \begin{bmatrix} -1 & 3 \\ 3 & 1 \end{bmatrix} \text{ since from the constitutive law } \boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} & 2\sigma_{xy} \\ 2\sigma_{yx} & 2\sigma_{yy} \end{bmatrix} \tag{3.25}$$

inserting these matrix into (3.9) results in

$$\sigma = \int_{\partial\Omega} -p\mathbf{I} + 2\mu \begin{bmatrix} -1 & 3 \\ 3 & 1 \end{bmatrix} \mathbf{n}\, d\Gamma \tag{3.26}$$

By taking the value of $\nu = 0.001$ the integration of the stress distribution at each Gauss point is fist for the pressure contribution the velocity distribution.

At Gauss point 1 weight = 0.25

$$\sigma = \int_{\partial\Omega} +2\mu \begin{bmatrix} -1 & 3 \\ 3 & 1 \end{bmatrix} \mathbf{n}\, d\Gamma = 0.25 * 0.002 * \begin{bmatrix} \frac{1}{2} & \frac{3}{2} \\ \frac{3}{2} & 1 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \end{bmatrix} = \begin{bmatrix} -0.00075 \\ -0.0005 \end{bmatrix} \tag{3.27}$$

At Gauss point 2 weight = 0.25

$$\sigma = \int_{\partial\Omega} +2\mu \begin{bmatrix} -1 & 3 \\ 3 & 1 \end{bmatrix} \mathbf{n}\, d\Gamma = 0.25 * 0.002 * \begin{bmatrix} \frac{1}{2} & \frac{3}{2} \\ \frac{3}{2} & 1 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \end{bmatrix} = \begin{bmatrix} -0.00075 \\ -0.0005 \end{bmatrix} \tag{3.28}$$

$$\text{Total contribution at } GP_0 \text{and } GP_1 = \begin{bmatrix} -0.00015 \\ -0.0001 \end{bmatrix} \tag{3.29}$$

Therefore the total drag in the element is calculated to be

$$\text{Total contribution at GP}_0 \text{ and GP}_1 = \begin{bmatrix} 0 \\ 1.125 \\ 0 \end{bmatrix} + \begin{bmatrix} -0.00015 \\ -0.0001 \\ 0 \end{bmatrix} = \begin{bmatrix} -0.00015 \\ 1.124 \\ 0 \end{bmatrix} \tag{3.30}$$

3.5 Parallelization techniques in Kratos Multiphysics

Time consuming simulations that requires at least a day or more to run are mostly solved using parallelization techniques. This simulations result in large equation systems which are solved using distributed memory models where the original computational domain is split into independent parts and processed by different processors. Currently in Kratos there are two types of parallelization techniques that are used, MPI(message-passing interface) and OpenMP(open multi-processing). MPI is the most commonly used standard application programming interface in supercomputers and clusters for managing the communication between processors in distributed memory systems [4]. Although it a complex process, with MPI it is possible to transfer local data from one processor to another due to the interconnection network built between the processors. On the other hand if we consider shared memory programming

OpenMP is often used. Key advantages of using OpenMP are its flexibility during the implementation process also ability to support portability between different platforms.

All the 3D cases in this research were solved on the Acuario cluster using MPI and the the 2D cases using a potable laptop running with OpenMP. As the total number of elements for the first two cases was not very big(1.5e6-2e6 elements) we only used 1 or 2 processors. On the hand the silsoe cube which had at least 7.2 million elements a larger number of processors was utilized. With respect to the partition between processors in Kratos, an external library `METIS` is used. The use of this library is shown in Figure 3.12 where the domain of one of the validation problems has been partitioned using 16 processors.

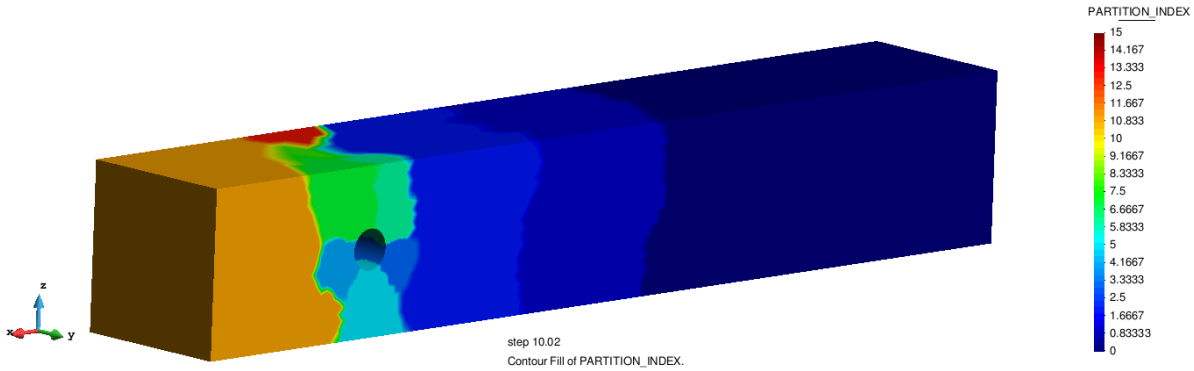


Figure 3.12: Domain partitioning of a problem decomposed into 16 subdomains

Chapter 4

Results and Analysis

This chapter collects all the tests carried out in order to validate the Symbolic and Embedded CFD formulations. The validation process will be done by validating the embedded element against the symbolic element which will be validated against the body-fitted element. The Body-fitted formulation which was implemented in Kratos by Cotela [9] has already been extensively validated in the past by a number of researchers [4], [9],[15] . Furthermore, whenever possible, the results of all the validation problems will be compared with either experimental or wind tunnel. Regarding the developed simulations, four different problems are presented. The first three benchmark problems are concerned with the computation of laminar flow over a cylinder [26]. In the first case we consider a 2D problem while in the other two, 3D cases are considered. Finally, a preliminary 3D case of the Silsoe cube problem is presented. Note that in all the validation tests will utilizes unstructured mesh.

4.1 Simulation of 2D Laminar flow around a cylinder

This problem has been intensively investigated, mainly because of its fundamental importance and its relevance for practical applications. Therefore, it has been selected due to the use of the circular shape of the cylinder, which means that the location of flow separation is only influenced by the flow regime or the upstream conditions, hence the accuracy is a challenge for numerical study.

Problem description

An incompressible Newtonian fluid for which the conservation equations of mass and momentum have already been discussed in Chapter 2 is considered. The kinematic velocity is defined as $\nu = 10^{-3} m^2/s$ and the fluid density as $\rho = 1.0\, kg/m^3$. The underlying geometry and boundary conditions are depicted in figure 4.1. where $H = 0.41$m is the channel height and $D = 0.1$ is the cylinder diameter. The Reynolds numbers is defined as Re=$\bar{U}D/\nu$ with the mean velocity $\bar{U}(t) = 2U(0, H/2, t)/3$. On the inlet boundary a parabolic inflow profile is prescribed,

$$U(0, y, t) = \big(4U_m y(H - y)H^2, V = 0\big) \tag{4.1}$$

with U_m =1.5 m/s yielding a Reynolds number $Re = 100$.

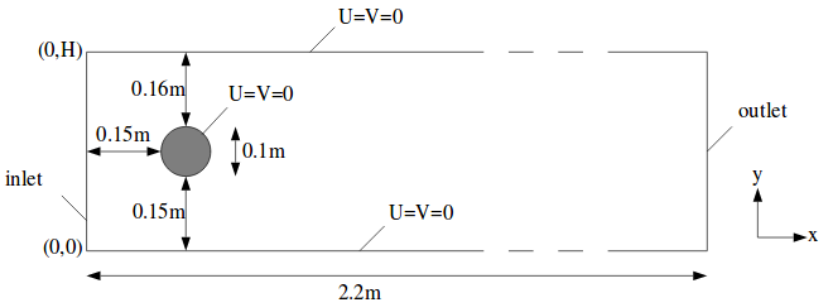


Figure 4.1: Geometry of 2D test case with boundary conditions

Mesh setup

In order to mesh the entire domain appropriately, the geometry is partitioned into different surfaces as shown in Figure 4.2, where a different element size (h) is applied on each surface. Since the Drag and Lift coefficients are computed around the cylinder boundary we need a fine mesh around this part. Therefore the applied surface elements sizes are shown in Figure 4.3. The element size for the boundary lines around the three surfaces was also taken to be the same as their respective surface element sizes. Note that since for the Embedded approach a distance function is used to track the physical location of the cylinder we need to mesh the entire geometry. Therefore the Embedded approach uses more elements (Nodes: 66,834 ; elements: 129,303) as compared with the Body-fitted approach (Nodes: 54,595 ; elements: 109,190). The resulting mesh in the case of the body-fitted approach is shown in Figure 4.4

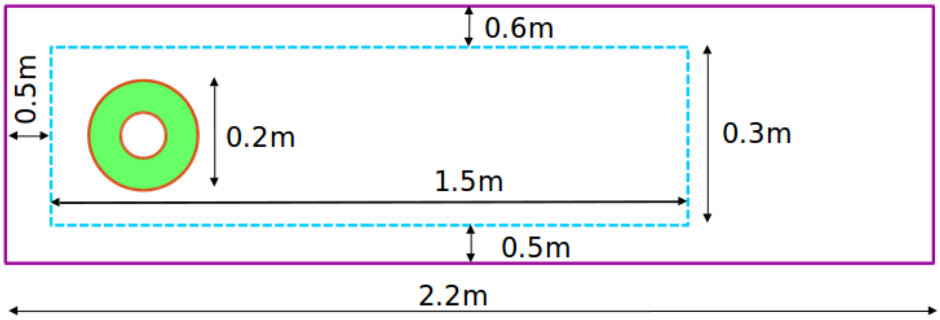


Figure 4.2: Geometry of 2D test case with boundary conditions (Not drawn to scale)

Figure 4.3: Geometry of 2D test case with boundary conditions

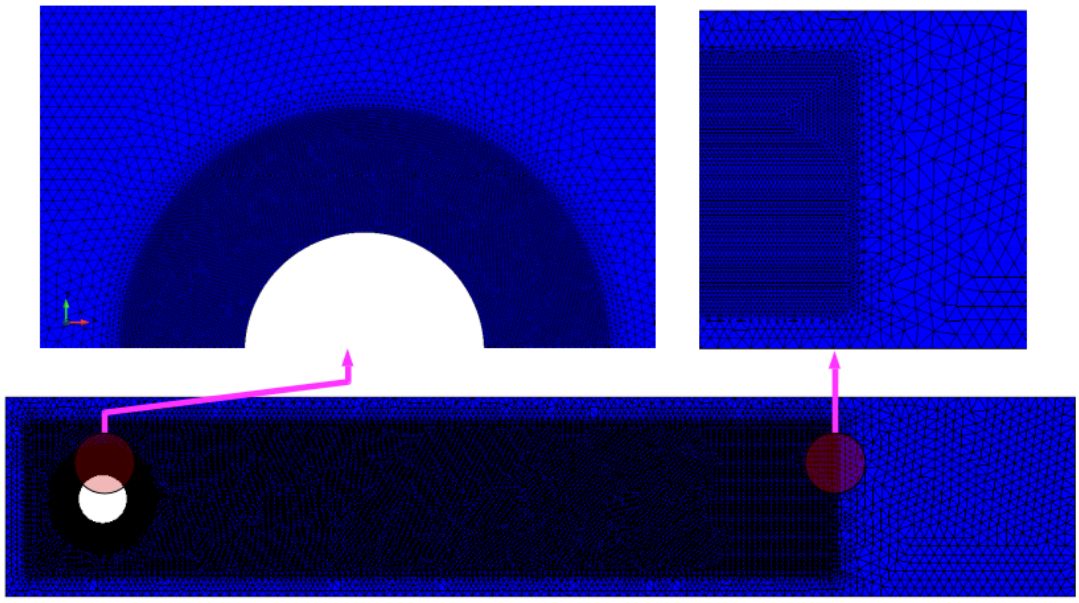


Figure 4.4: Body-fitted mesh

Simulation settings

This simulation was carried out for a total of 5s using a time-step of 0.01s. In terms of convergence, a tolerance of 10^{-5} was set for both pressure and velocity.

Results and Discussion

The drag and lift values are calculated using the coefficients as discussed in chapter 2 and hence th resulting values are shown in the table below.

Table 4.1: Comparison of maximum and minimum drag and lift coefficients

		Drag			**Lift**		
		Reference	Obtained	Rel. err[%]	Reference	Obtained	Rel. err[%]
Body-fitted	Max	3.2400	3.3169	2.37	1.0100	1.008	-0.20
	Min	3.2200	3.2352	0.47	0.9900	-1.0126	-2.28
Symbolic	Max	3.2400	3.3032	1.95	1.0100	1.0327	2.25
	Min	3.2200	3.2257	0.18	0.9900	-0.9721	-1.81
Embedded	Max	3.2400	3.2923	1.61	1.0100	1.0302	2.0
	Min	3.2200	3.2146	0.17	0.9900	-0.9718	-1.83

As it will be further discussed both the symbolic and embedded produced results that are within the reasonable range of both drag and lift coefficients. This can be observed in table where the relative errors for each formulation are calculated. The biggest relative error in terms of the drag is 2.37% while in terms of the lifts its 2.25 %. These values are within an acceptable range.

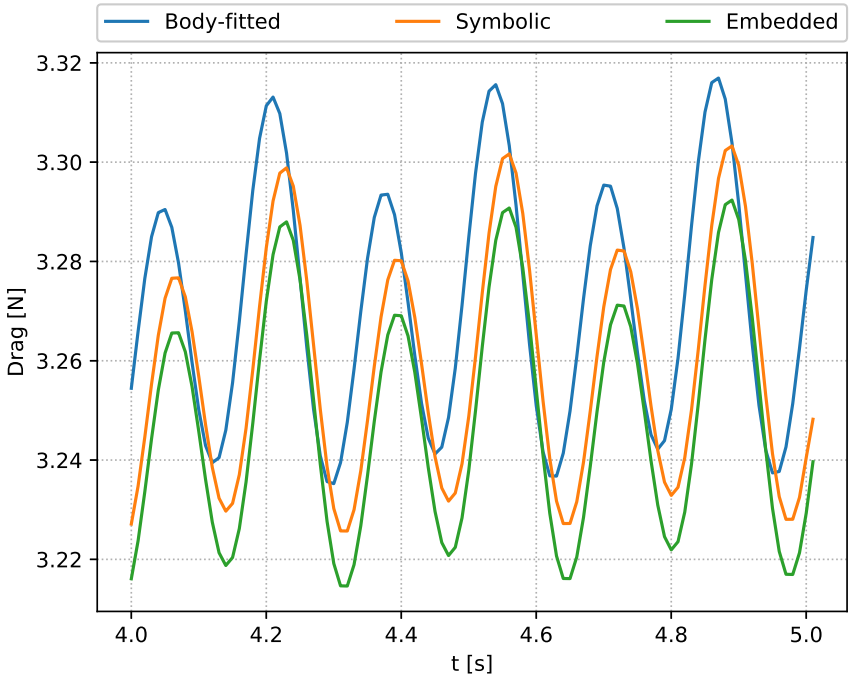


Figure 4.5: Comparison of Drag plots

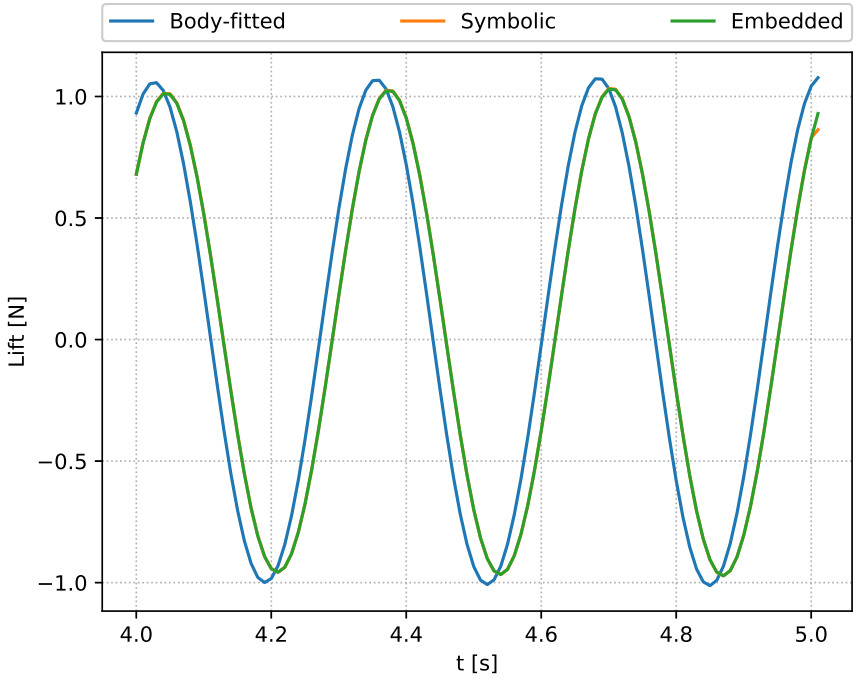


Figure 4.6: Comparison of lift plots

As we have already introduced the body-fitted formulation in Kratos Multiphysics has already been validated in the past. Therefore, for both the symbolic and embedded formulations will use the upper and lower limits of the body-fitted colormap for both pressure and velocity distributions. Doing so allows us to effectively compare the results of the three formulations using the same colormap.

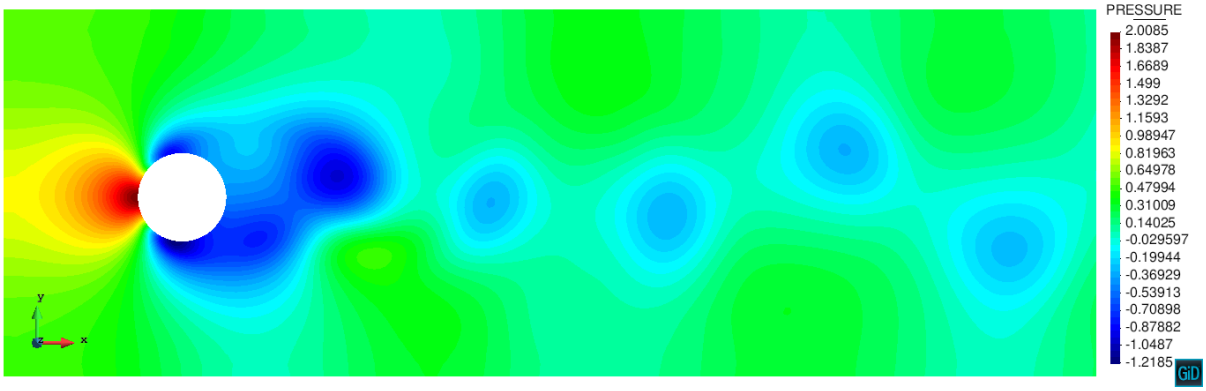
As it will be further elaborated, this is a non-stationary flow which results in a repetition

of periods, hence the performance of each formulation can be assessed by looking at the fundamental characteristics of the flow occurring within a single period in the flow. Hence for each formulation a single period constitutes of about five snapshots taken at different time steps within each period. Note that to avoid the use of many figures we will only show three snapshots (first, middle and final) for both the pressure and velocity distributions. For the pressure distribution these figures are shown in Figure 4.7, 4.8, and 4.9 and for the velocity distribution; Figure 4.10, 4.11, and 4.12.

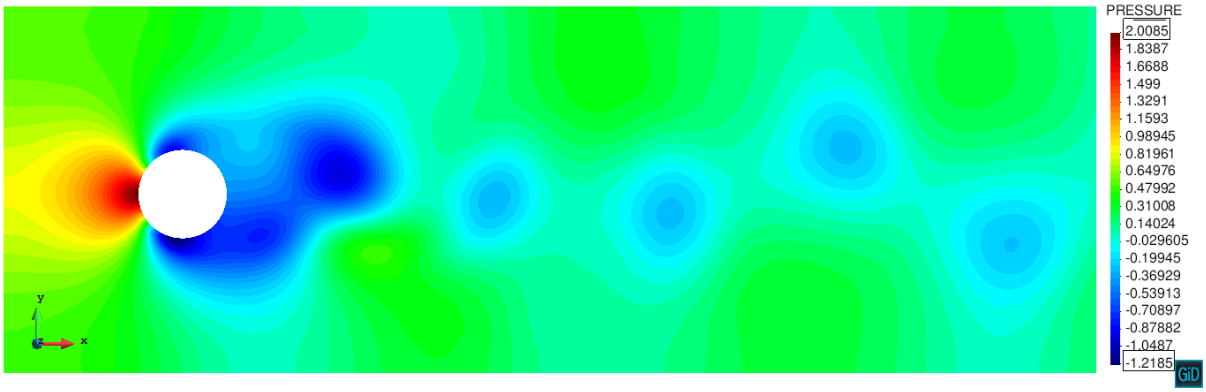
The flow characteristics are controlled by the Reynolds number, hence from fundamental fluid dynamics we know that an increase in this number (for example Re = 100) causes the development of the laminar vortex street in the downstream of the cylinder [2]. The development of this phenomena is of great importance in this benchmark test. This phenomenon can be understood by looking at what happens in two different regions of the flow, namely the wake and the boundary layer.

The geometrical configurations which constitutes of the use of a circular cylinder imposes adverse pressure gradient which leads to the separation of the boundary layer over the cylinder surface. This is well elaborated by sumer and Fredsoe [29] who stated that the pressure gradient are imposed by the divergent geometry of the flow environment at the rear side of the cylinder. This separation leads to a formation of the a shear layer. Considering the effect of the applied no-slip boundary condition around the cylinder, it is noted that the vorticity is created along the cylinder boundary layer due to this condition. The boundary layer that is formed from the cylinder surface contains a large amount of vorticity which in turn is fed in the shear layer formed downstream of the separation point. This entirely causes the shear layer to roll up into a vortex. As it can be seen for example in the figures representing pressure distribution, this process happens simultaneously on both sides of the cylinder. Eventually at Re=100 the wake is expected to become unstable leading to the so called vortex shedding in which vortices are shed alternatively at either side of the cylinder at a certain frequency. This then leads to the appearance of a vortex street as shown in from Figure 4.7 to Figure 4.12. This phenomena is observed in all the three formulations, which tells us that both the symbolic and embedded formulation are working as in a similar manner.

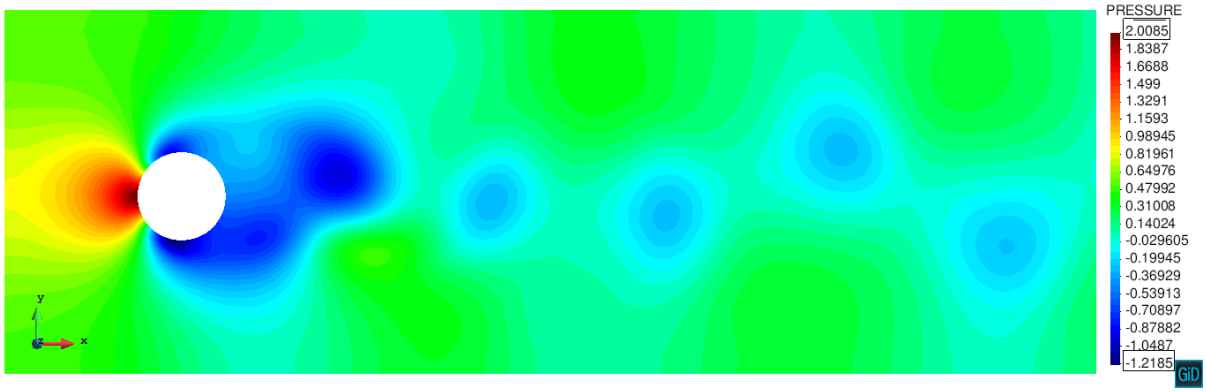
Furthermore, as we have discussed in section 2.7.4 the mean drag which comes about due to the result of the impact between the flow and the cylinder is a combination of the pressure force and the friction force. The pressure contribution is affected by the above discussed vortex-shedding phenomenon, due to the fact that when vortex shedding process progresses the pressure distribution around the cylinder undergoes a periodic change. This eventually results in a periodic variation in the force components (drag and lift forces) on the cylinder. Considering the vortex-shedding period (T_v), the drag force which acts in the in-line direction oscillates around the mean drag at $\frac{1}{2}T_v$. The mean drag force for each formulation can be calculated from the table 4.1 as (body-fitted = 3.2760 N, symbolic = 3.2644 N and embedded = 3.2534 N). Hence we see from Figure 4.5 that the drag plot for each of the three formulations oscillates around its respective mean drag force. This explains the difference in the Amplitude value of the drag plot in this figure. On the other hand the mean lift force is zero because of the complete symmetric nature of the incoming flow with respect to the cylinder axis. Nevertheless, as noted by sumer and Fredsoe [29] there exits a lift force on the cylinder in the transverse direction which also varies periodically with time. This lift force oscillates around the mean lift force at T_v, hence as shown in Figure 4.6 the corresponding plot for each formulation is depicted in this figure.



(a) Body-fitted

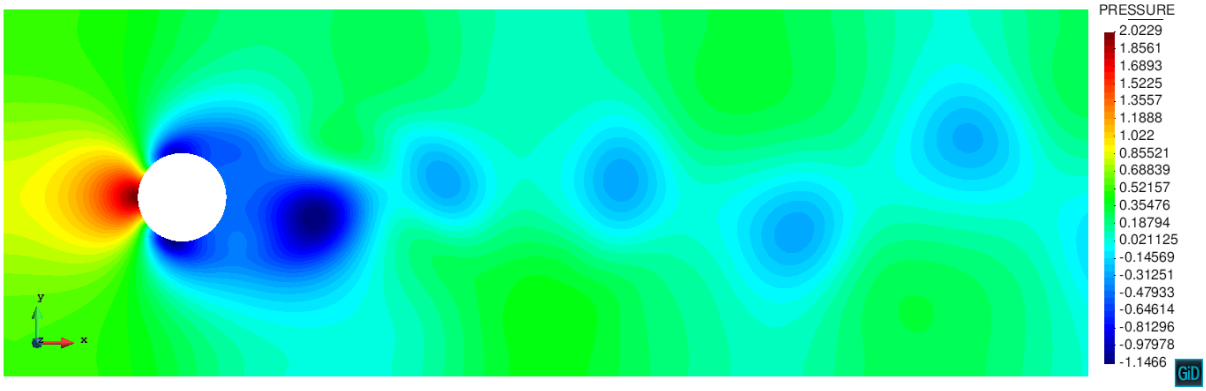


(b) Symbolic

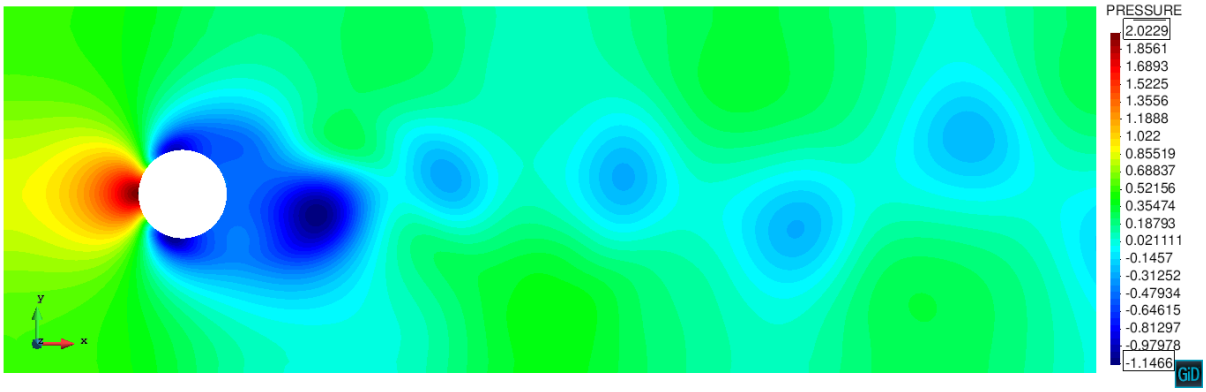


(c) Embedded

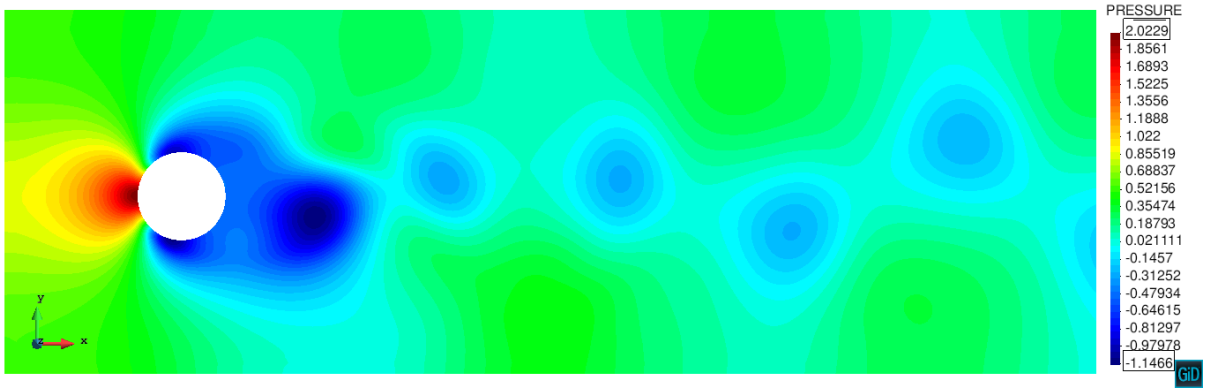
Figure 4.7: Pressure distribution at the start of a single period ($t_0 = 4.13$s for body-fitted, $t_0 = 4.15$s for symbolic, $t_0 = 4.15$s for embedded)



(a) Body-fitted

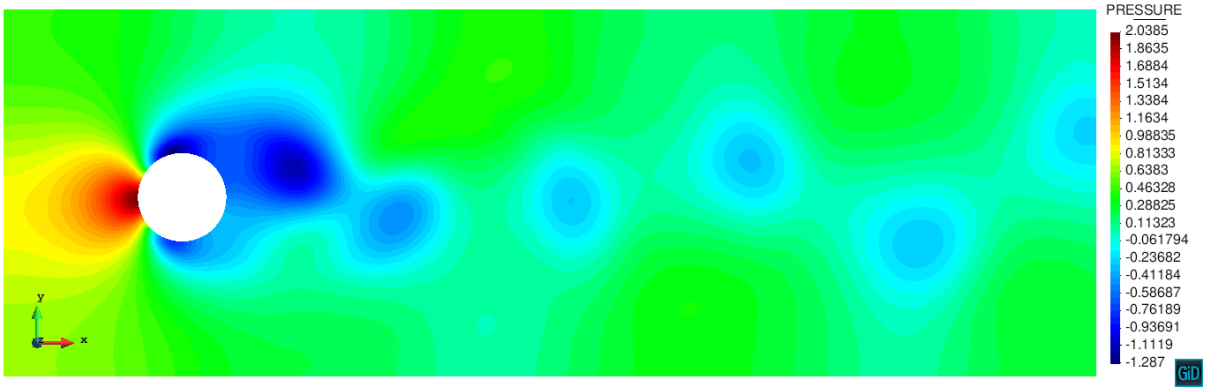


(b) Symbolic

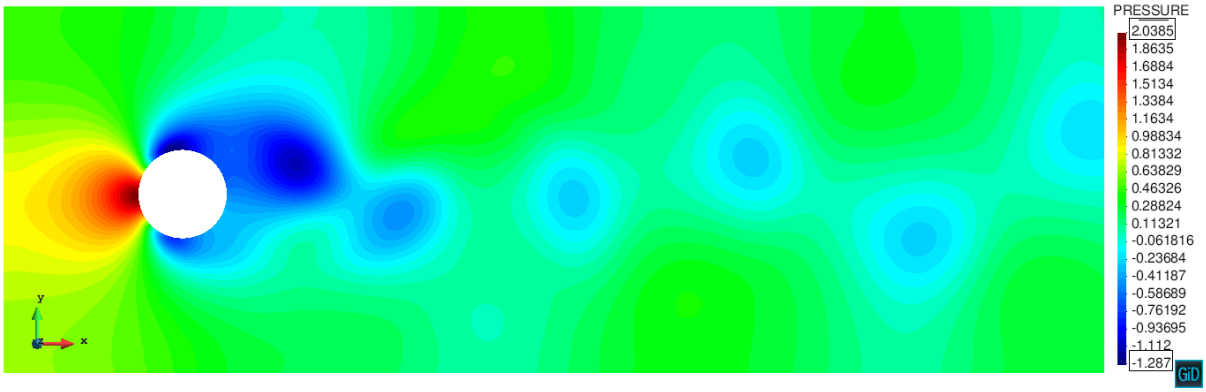


(c) Embedded

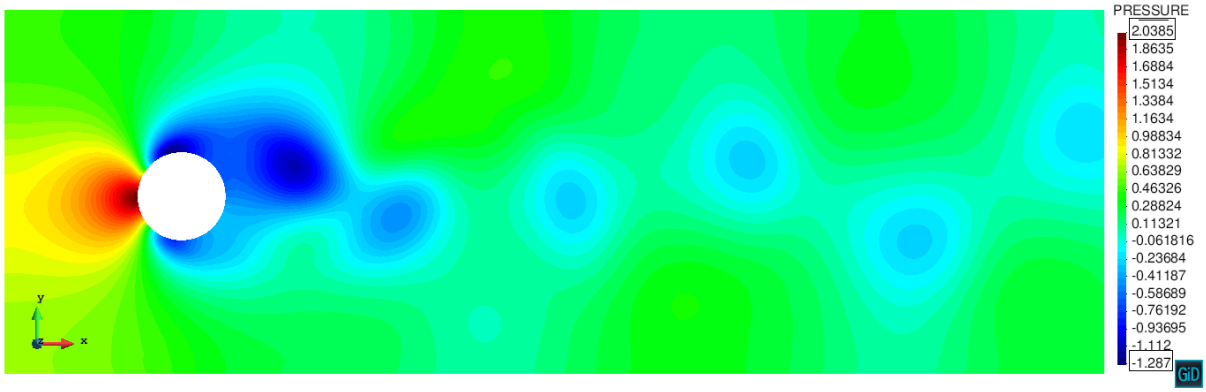
Figure 4.8: Pressure distribution at the middle part of a single period taken at time $t_0 + 0.12$s for each formulation



(a) Body-fitted

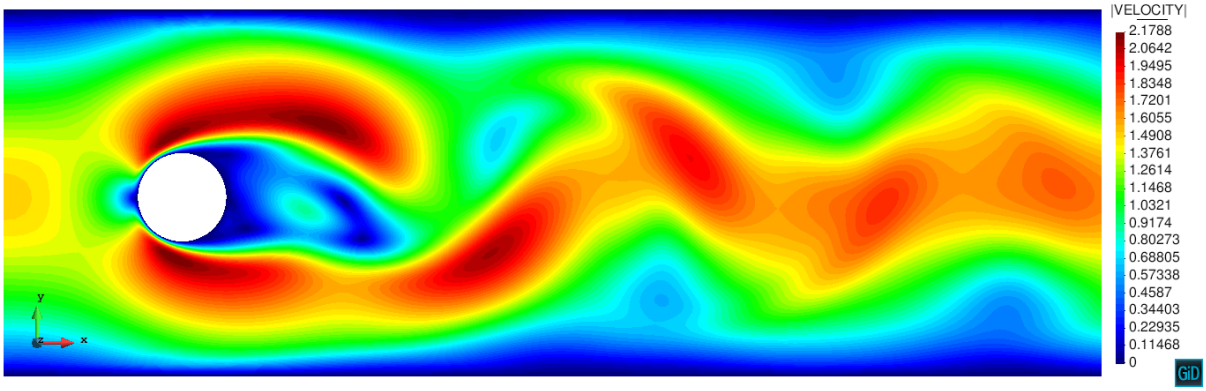


(b) Symbolic

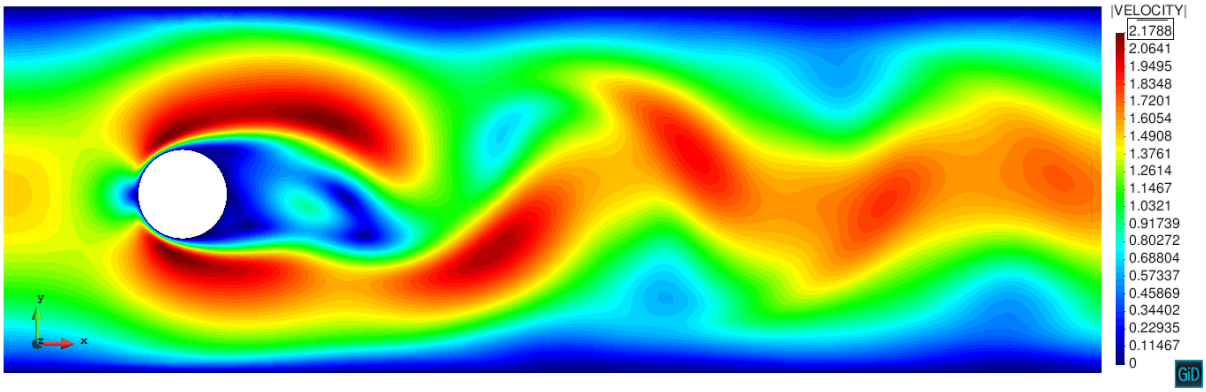


(c) Embedded

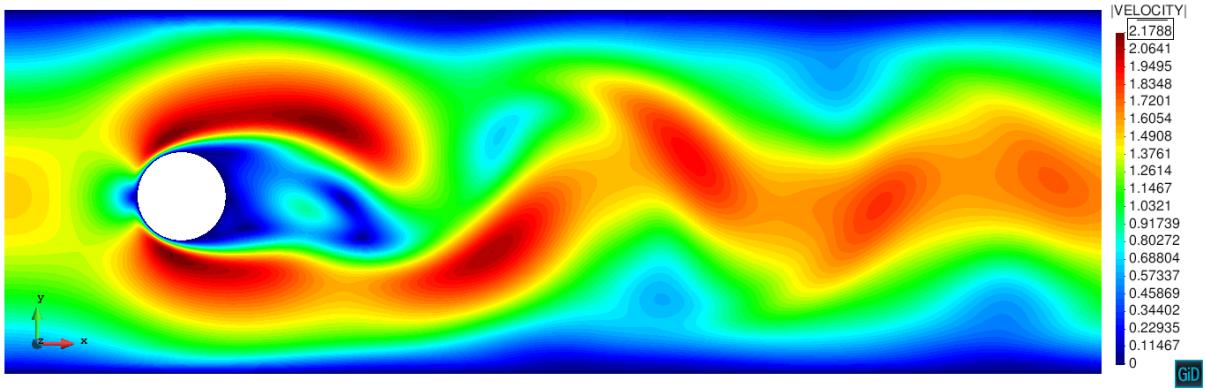
Figure 4.9: Pressure distribution at the end of a single period taken at time $t_0 + 0.24$s for each formulation



(a) Body-fitted

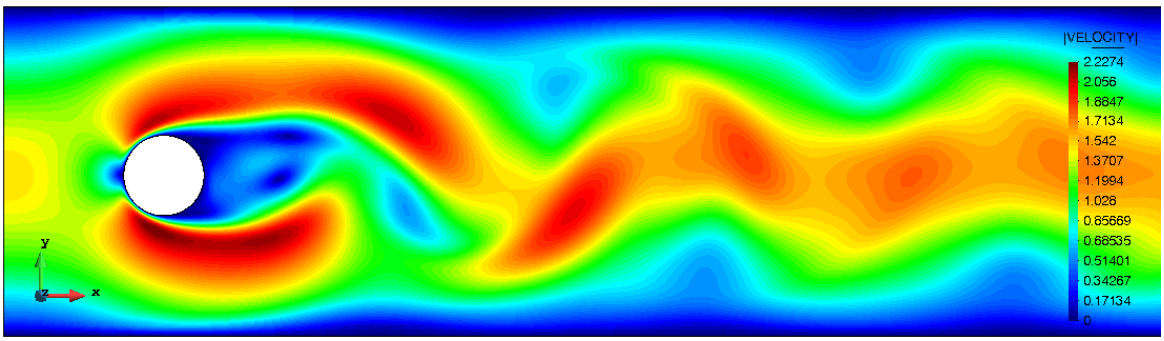


(b) Symbolic

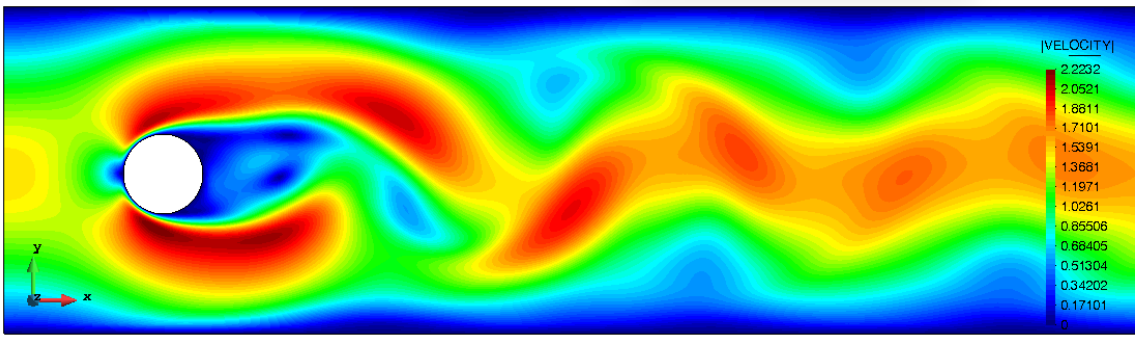


(c) Embedded

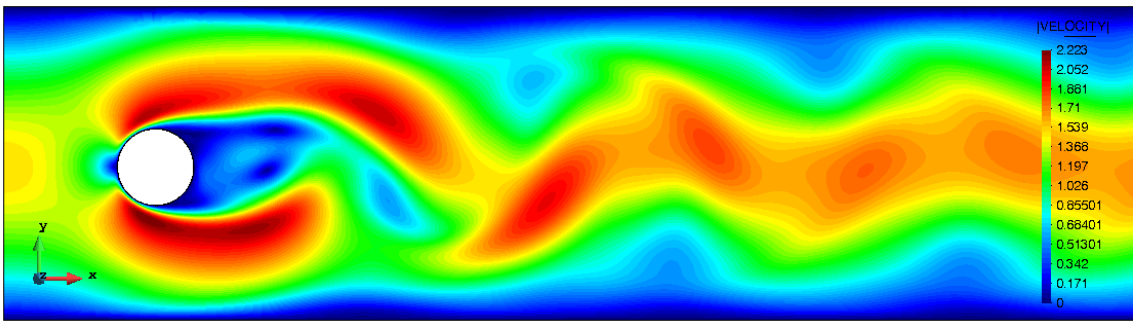
Figure 4.10: Velocity distribution at the start of a single period ($t_0 = 4.13$s for body-fitted, $t_0 = 4.15$s for symbolic, $t_0 = 4.15$s for embedded)



(a) Body-fitted

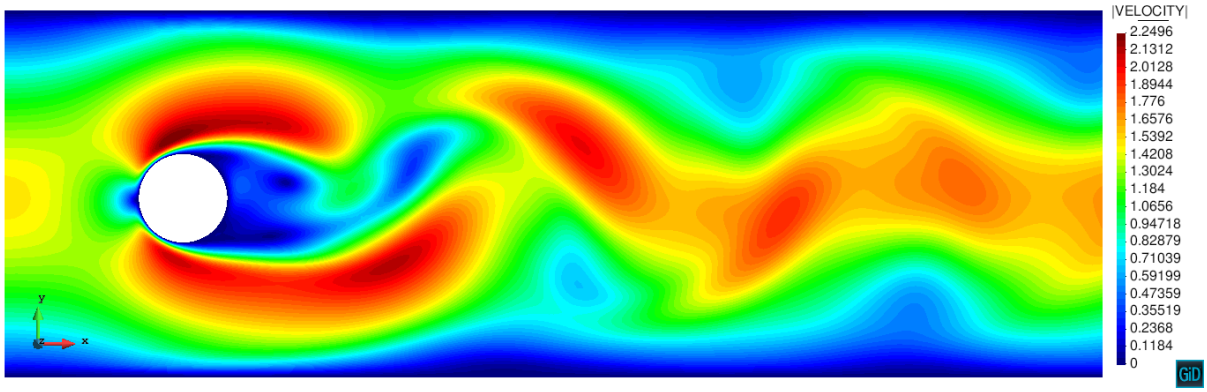


(b) Symbolic

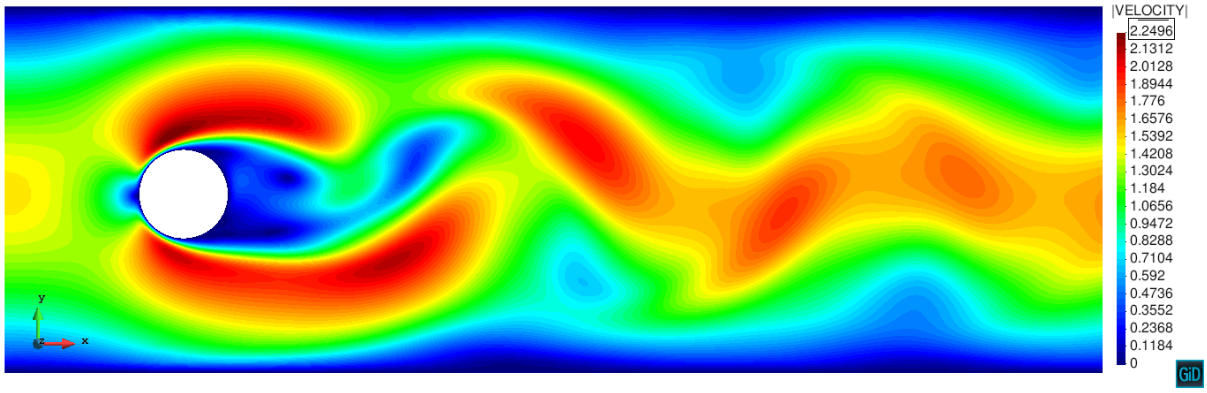


(c) Embedded

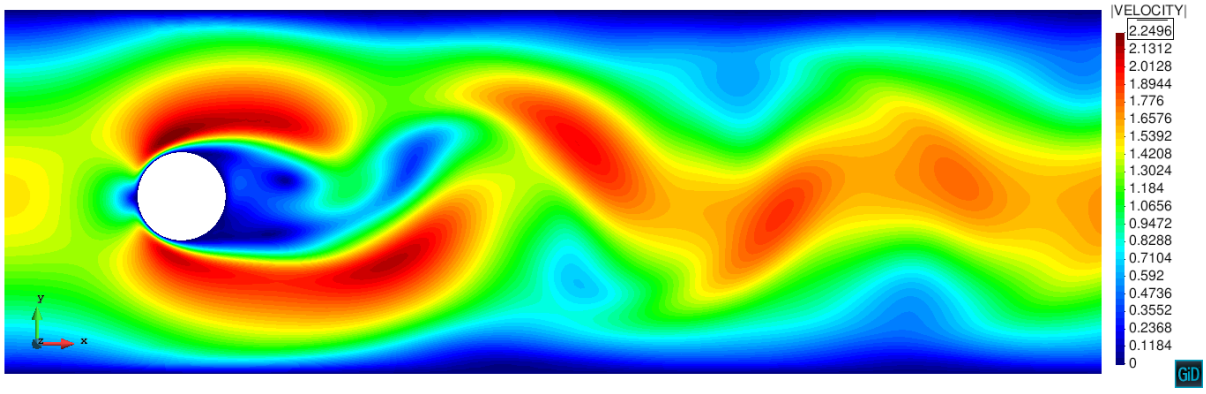
Figure 4.11: Velocity distribution at the middle part of a single period taken at time $t_0 + 0.12$s for each formulation



(a) Body-fitted



(b) Symbolic



(c) Embedded

Figure 4.12: Velocity distribution at the end of a single period taken at time $t_0+0.24$s for each formulation

4.2 3D cases of Laminar Flow around a Cylinder

Numerically challenging and comprehensive 3 dimensional benchmark cases are of great importance in the field of CFD. These cases provides a framework to quantitatively explore the limits of computational tools and to validate them as well. In this section we explore two cases of laminar flow over a cylinder; the first case been a cylinder with a circular cross-section area and the second case involving a cylinder with a square cross-section area. Both of these cases are defined within the DFG (German Research Foundation) priority research program "Flow simulation on high performance computers" by Schäfer and Turek [26].

4.2.1 Case 1: Cylinder with a circular cross-section area

Problem description

The configuration and boundary conditions for this case are shown in Figure 4.13. Here the problem at hand is considered to be unsteady with the fluid kinematic velocity $\nu = 10^{-3} m^2/s$ and the fluid density $\rho = 1.0\, kg/m^3$. The prescribed boundary condition on the channel walls is the no-slip condition. This condition is also proscribed on the surface of the cylinder. On another hand, the outer plane is prescribed with a pressure boundary with a zero magnitude value. The inflow inlet boundary condition is given by a parabolic velocity profile

$$U(0, y, z, t) = 16 U_m yz(H - y)(H - z)/H^4, V = W = 0 \tag{4.2}$$

With $U_m = 2.25$ m/s, yielding the Reynolds number Re = 100.

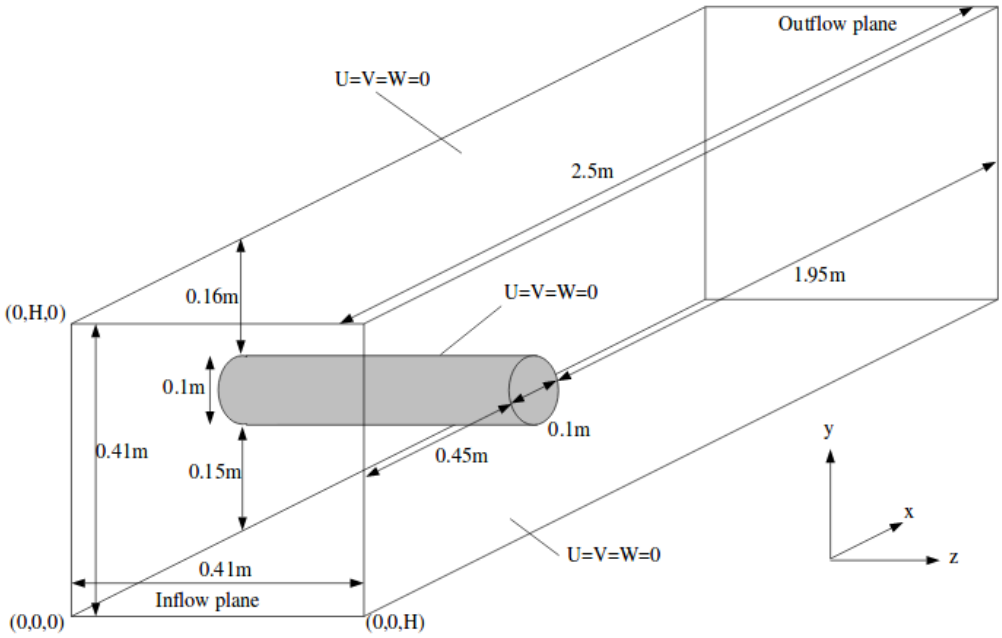


Figure 4.13: Configuration and boundary conditions of flow over cylinder with circular cross-section area

Mesh setup

The mesh settings are carried out in accordance with Figure 4.14, note that although this is given in 2D it represents the front plane of the geometry and hence these settings are protruded to the rest of the domain, as depicted in Figure 4.15. Similarly as before, important regions like the region surrounding the cylinder are set with a finer mesh. The respective mesh sizes for the volumes are given in Figure 4.15, this is also the same mesh size that is also used surfaces surrounding the volumes. As noted before, for a fair comparison the same mesh settings for all the three formulations. The resulting computational mesh is shown in Figure 4.26, No. of elements (body-fitted = symbolic = 1.2x10^6 , embedded = 1.5 x10^6).

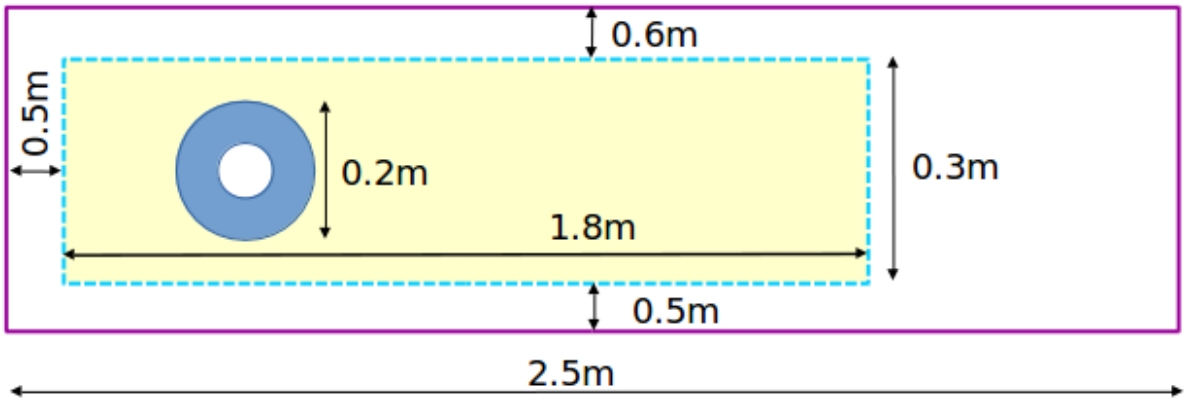


Figure 4.14: Mesh setup

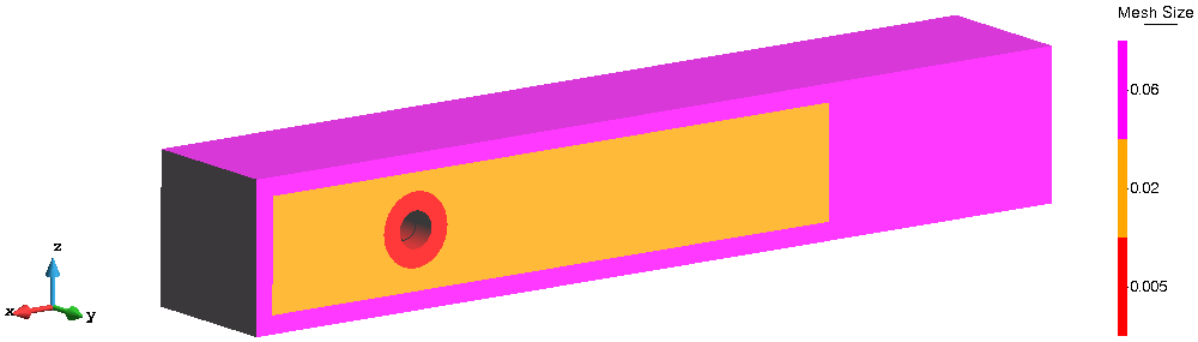


Figure 4.15: Mesh sizes on the volumes

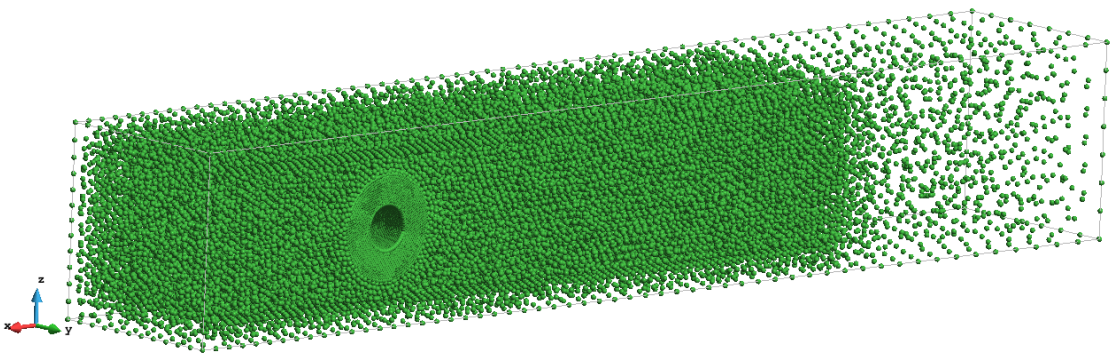


Figure 4.16: Resulting mesh with the each point representing each node in the mesh]

Simulation settings and result analysis

The simulation was carried out for a total of 10s using a time-step of 0.02s. The linear system was solved using the MultiLevelSolver with a tolerance of 10^{-6}. On the other hand the convergence tolerance for both pressure and velocity was set to 10^{-5}. As for the results, the drag and lift coefficients are calculated and their respective relative errors are shown in the table below. Furthermore, the pressure and velocity distributions are also presented.

Table 4.2: Comparison of maximum and minimum drag and lift coefficients for the three formulation

		Drag			**Lift**		
		Reference	Obtained	Rel. err[%]	Reference	Obtained	Rel. err[%]
Body-fitted	Max	3.3100	3.5200	6.34	-0.0110	0.0045	0.01*
	Min	3.2900	3.5197	6.98	-0.0080	-0.031	0.02*
Symbolic	Max	3.3100	3.5363	6.83	-0.0110	0.0044	0.01*
	Min	3.2900	3.5316	7.34	-0.0080	-0.027	0.02*
Embedded	Max	3.3100	3.1576	4.60	-0.0110	0.006	0.01*
	Min	3.2900	3.1548	4.11	-0.0080	-0.019	0.01*

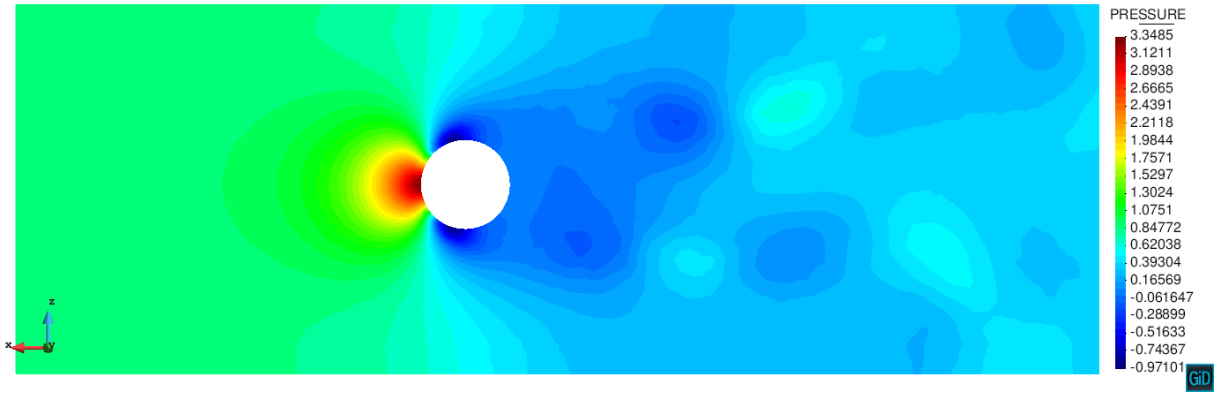
(Note that in the table * represents the absolute error since the values are close to zero)

It is important to understand that this problem is not a symmetric version of the previously discussed 2D problem. This is because the boundary conditions in all the four side walls of channel are prescribed with a No-slip boundary condition. Hence, this makes the flow to be constrained on the middle of the channel which therefore imposes a change in the behavior of the solution as it be further discussed.

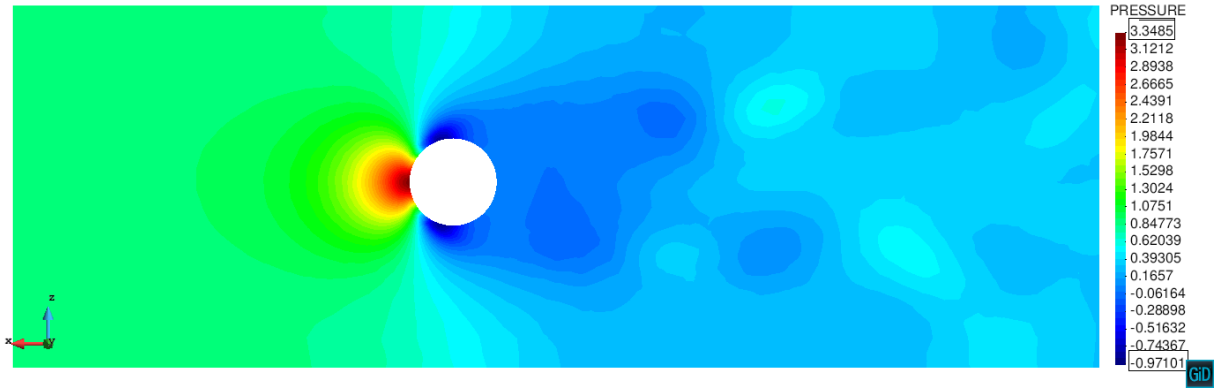
From table 4.2 we see that the values of the maximum and minimum drag and lift coefficients of both the body-fitted and symbolic are very high compared with the values of the embedded formulation. Also, all the results have higher relative errors due to the fact that we used a coarse to perform the simulations. This was done as a compromise between the required computational power and the accuracy of the results. Hence having a coarse mesh means that we have few elements between the upper and lower walls of the channel and the surface of the cylinder which eventually inhibits both the body-fitted and symbolic formulation to accurately compute the drag and lift forces in the boundary layer. As for the embedded, a coarse mesh will lead to a poorly defined structure of the cylinder which will definitely lead to low drag and lift coefficient values.

However, the flow characteristics can still be captured even this mesh. As this is a non-stationary problem, the resulting solutions is expected to be periodic. Hence once again as it is vital to look at a single period in order to validate our formulations. A single period in each of the formulations is completely represented by three snapshots as shown in Figures 4.17,4.18 and 4.19 for pressure and Figures 4.20,4.21 and 4.22 for velocity. By looking at these figures, for example the ones involving pressure, we can notice that as the flow progresses vortices begins to appear from each side of the cylinder leading to vortex shedding. This eventually develops into the vortex street with the repetition of the periods. Now here we will not explain the manner in which these vortices appear as this is similar to what has been explained

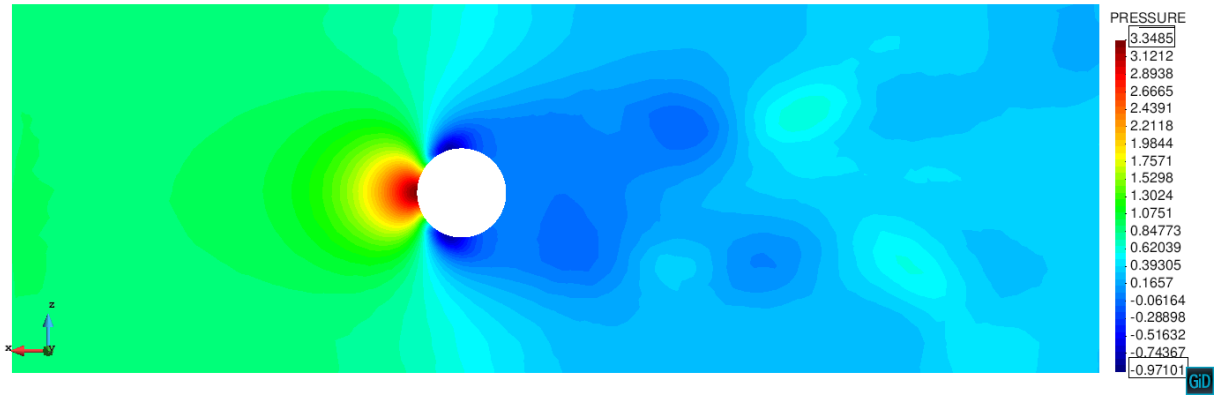
in the of the previous test. However the interesting aspect of these test is the way the vortex street takes place. Instead of the zig-zag movement of the vortices as in the case of the previous test here due to the fact that the flow is constrained in the middle of the channel each vortice that is produced from each side of the cylinder seem to maintain or rather follow a specific path such that this path does not coincide with the part of the other vortice coming from the other side of the cylinder. This behavior is what is expected in this benchmark test, which we have eventfully observed in all the three formulations.



(a) Body-fitted

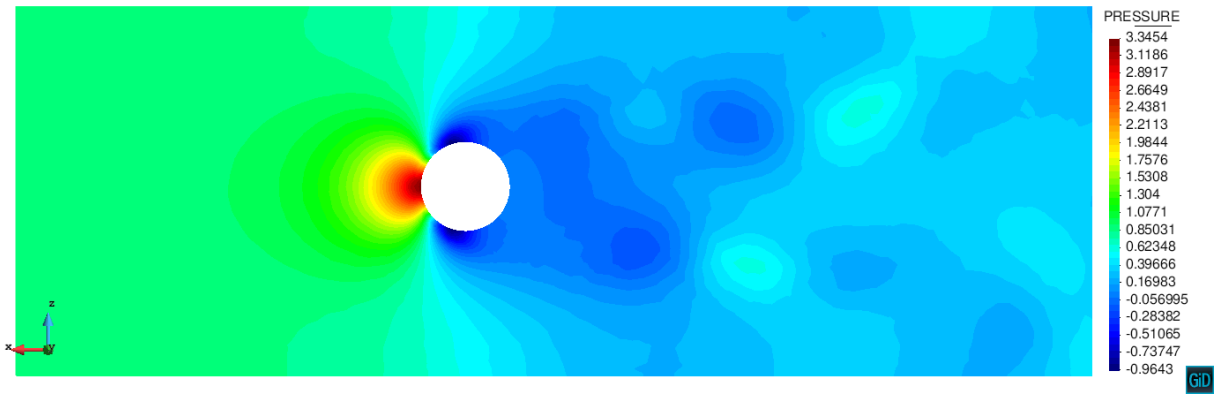


(b) Symbolic

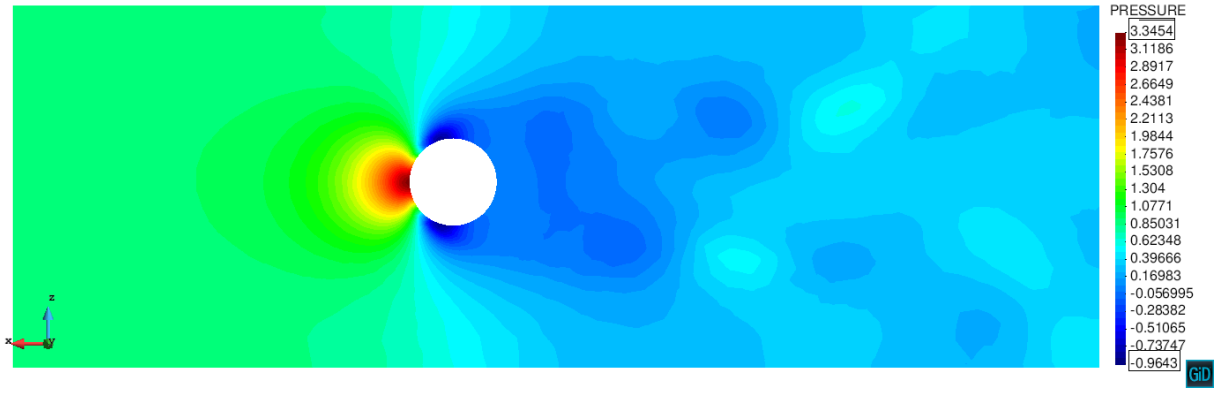


(c) Embedded

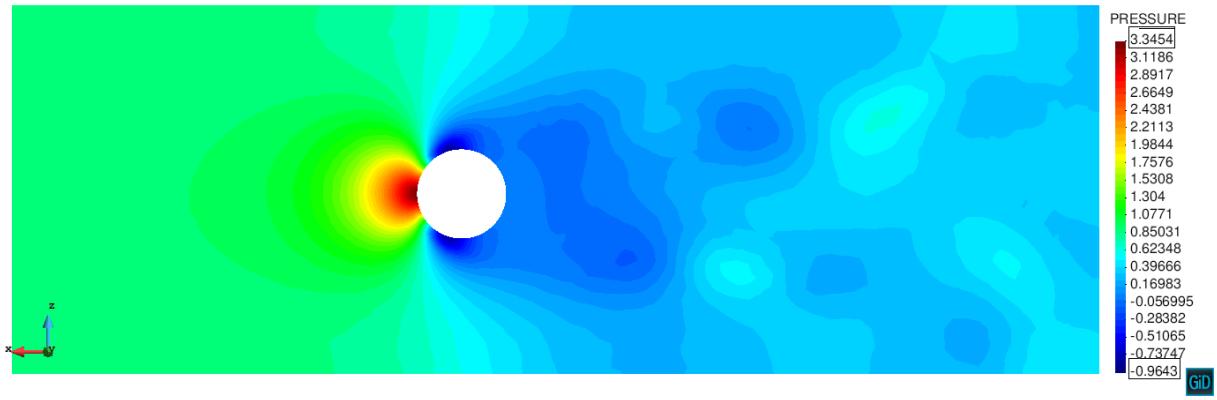
Figure 4.17: Pressure distribution at the start of a single period ($t = 4.42$s for body-fitted, $t = 3.3$s for symbolic, $t = 4.7$s for embedded)



(a) Body-fitted

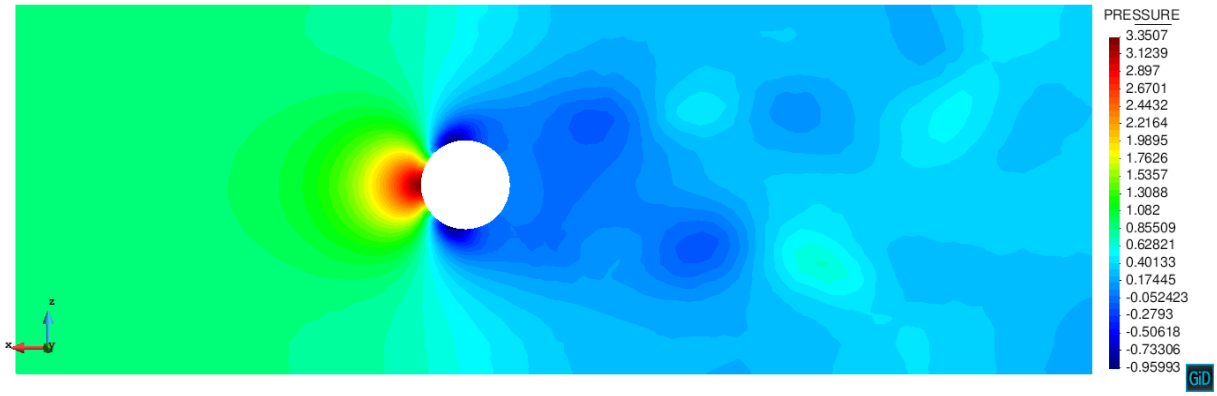


(b) Symbolic

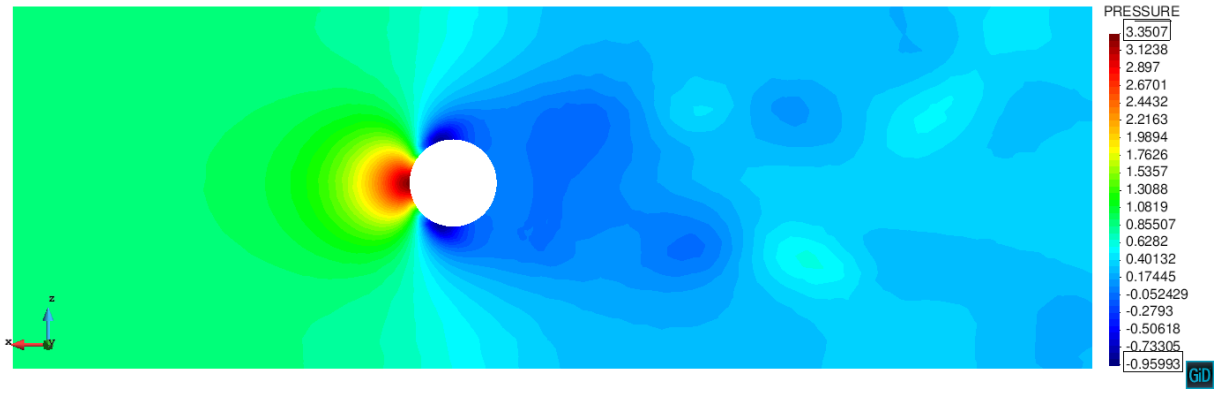


(c) Embedded

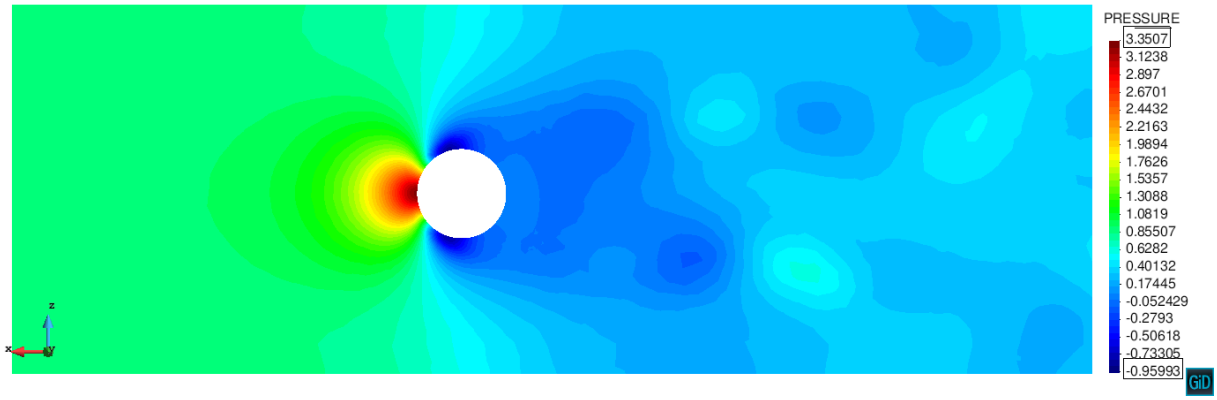
Figure 4.18: Pressure distribution at time ($t = 4.52$ s for body-fitted, $t = 3.4$s for symbolic, $t = 4.8$)



(a) Body-fitted

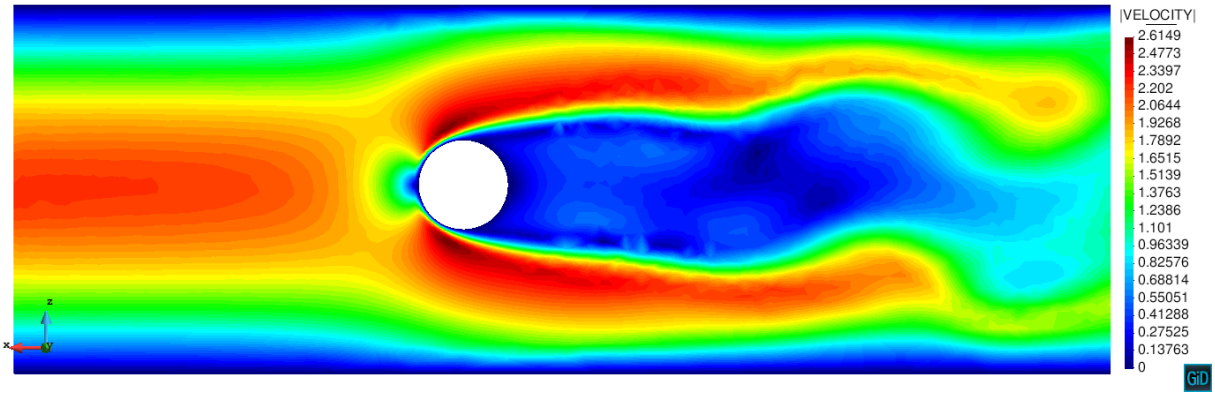


(b) Symbolic

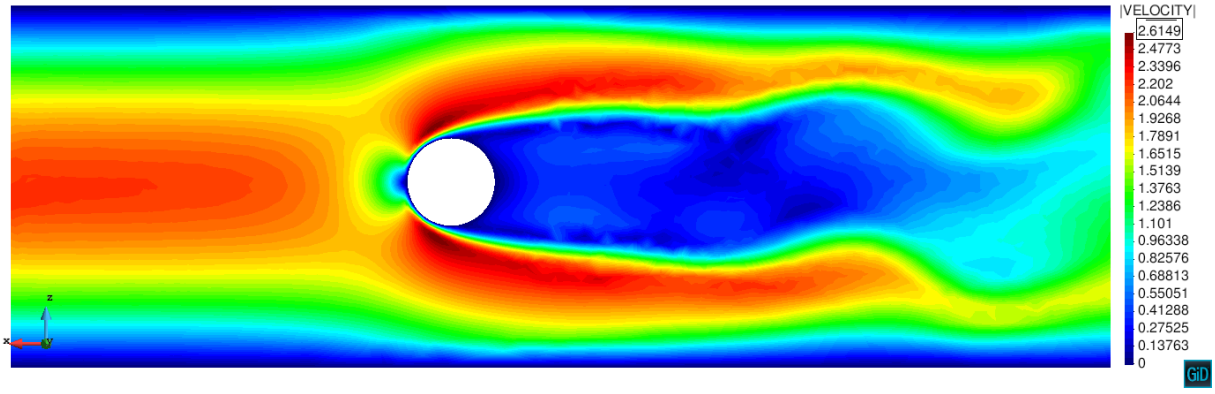


(c) Embedded

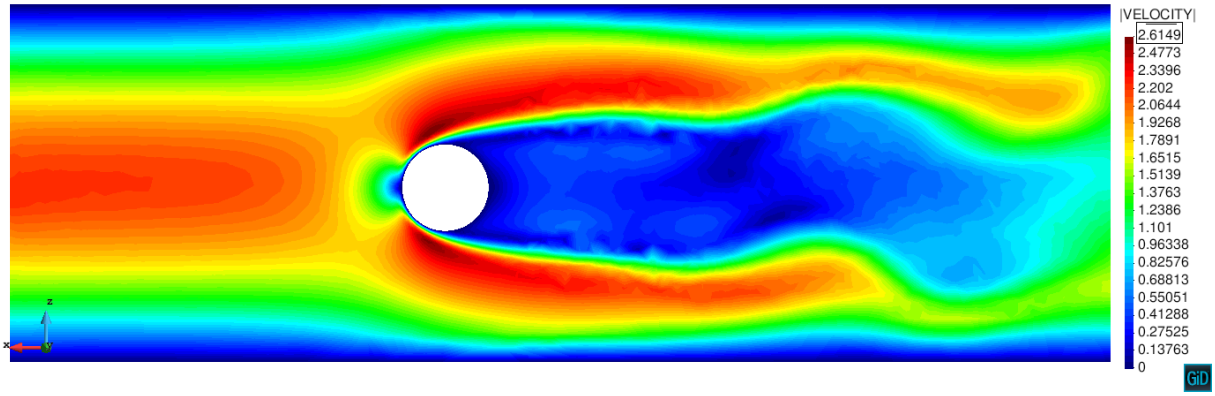
Figure 4.19: Pressure distribution at time ($t = 4.62$ s for body-fitted, $t = 3.5$s for symbolic, $t = 4.9$)



(a) Body-fitted

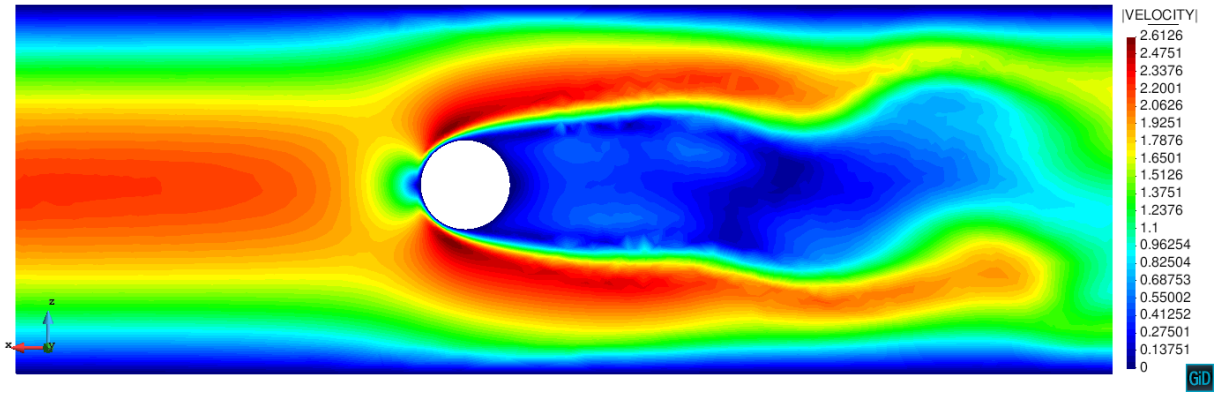


(b) Symbolic

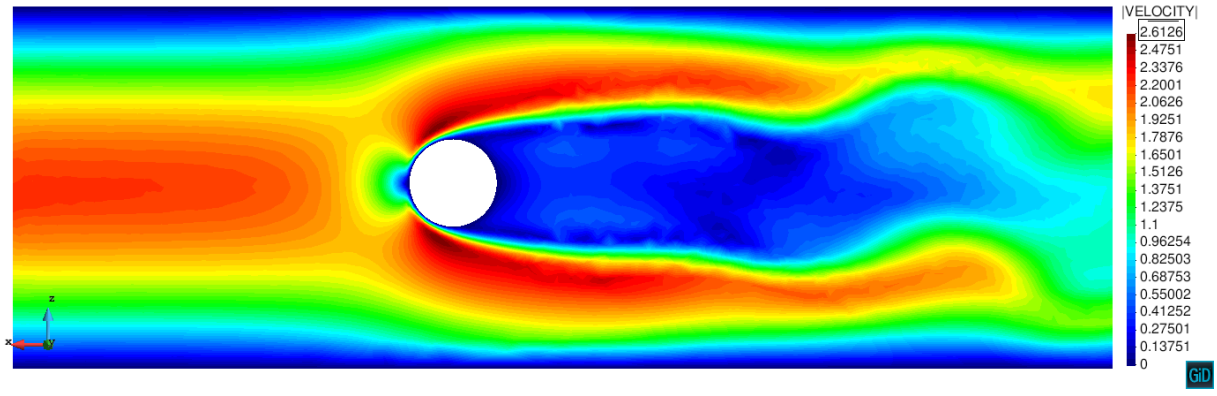


(c) Embedded

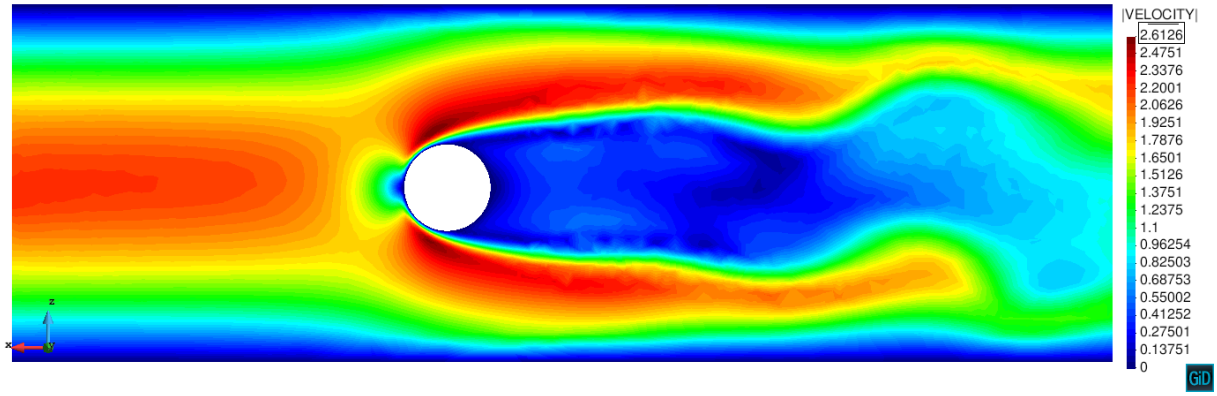
Figure 4.20: Velocity distribution at the start of a single period ($t = 4.42$s for body-fitted, $t = 3.3$s for symbolic, $t = 4.7$s for embedded)



(a) Body-fitted

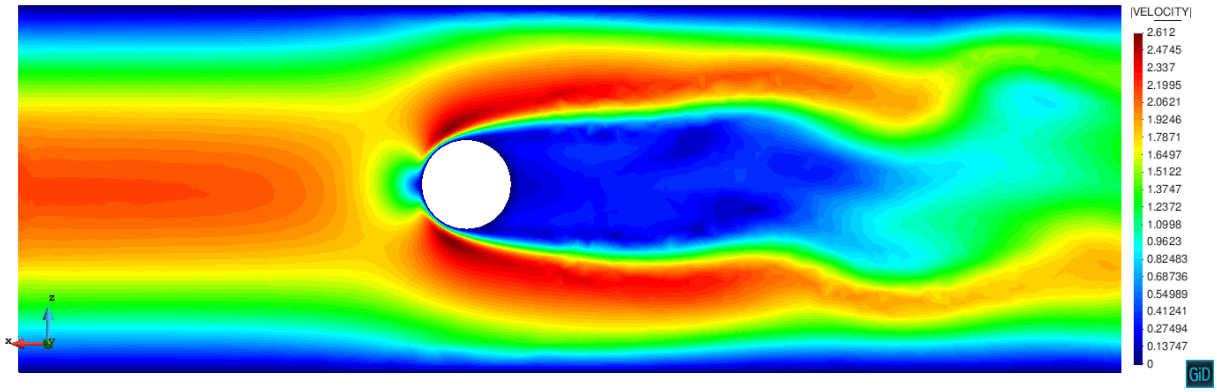


(b) Symbolic

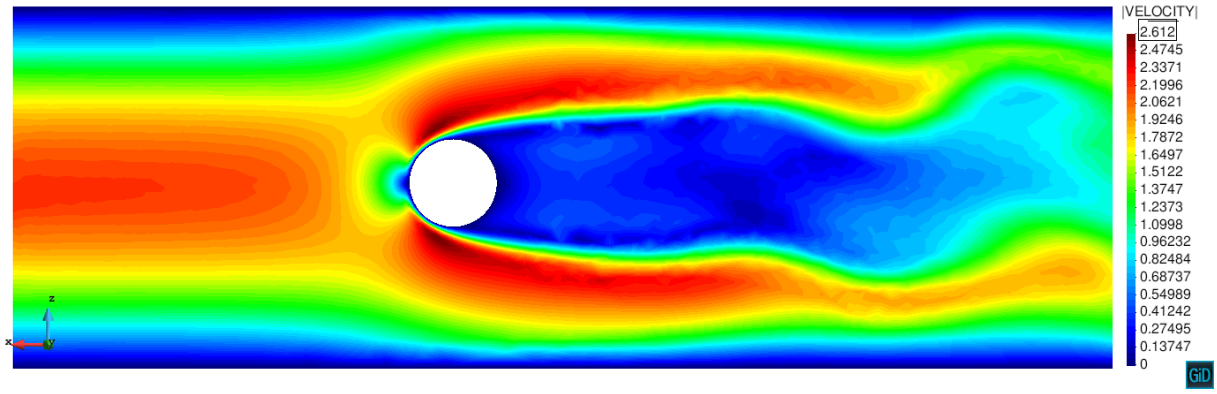


(c) Embedded

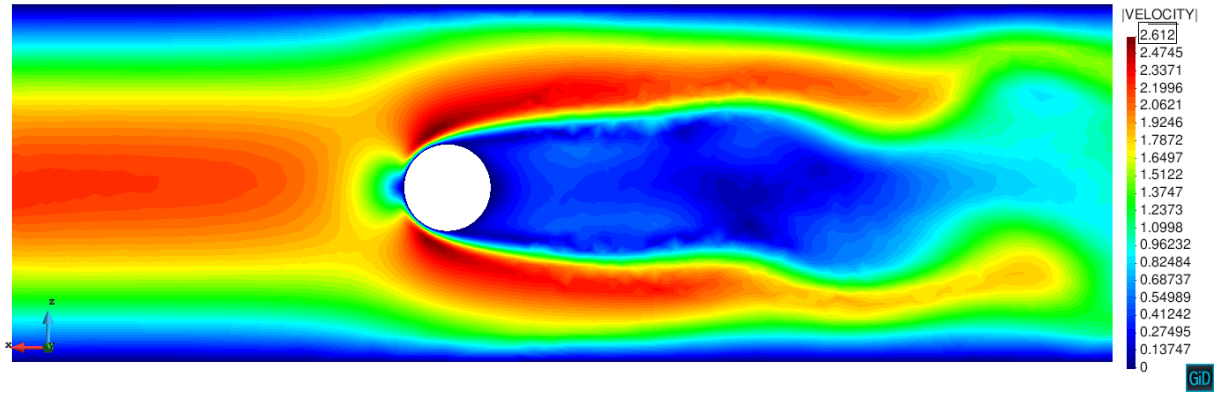
Figure 4.21: Velocity distribution at time ($t = 4.52$ s for body-fitted, $t = 3.4$s for symbolic, $t = 4.8$)



(a) Body-fitted



(b) Symbolic



(c) Embedded

Figure 4.22: Velocity distribution at time ($t = 4.62$ s for body-fitted, $t = 3.5$s for symbolic, $t = 4.9$)

4.2.2 Case 2: Cylinder with a square cross-section area

Problem description

The problem description and boundary conditions for this problem and the previous one almost the same. The key difference is that here the cylinder has a square cross-section area. With that, the configuration and boundary conditions for this case are shown in Figure 4.23. Here as well the problem is considered to be unsteady with kinematic velocity $\nu = 10^{-3} m^2/s$ and the fluid density $\rho = 1.0\, kg/m^3$. All the the four walls surrounding the cylinder are prescribed with a No-slip boundary conditions hence here as well the flow is constrained on th middle of the channel. On the outlet boundary pressure with a zero magnitude value is prescribed. The inflow inlet condition is given by a parabolic velocity profile

$$U(0, y, z, t) = 16 U_m yz(H - y)(H - z)/H^4, V = W = 0 \tag{4.3}$$

With $U_m = 2.25$ m/s, yielding the Reynolds number $Re = 100$.

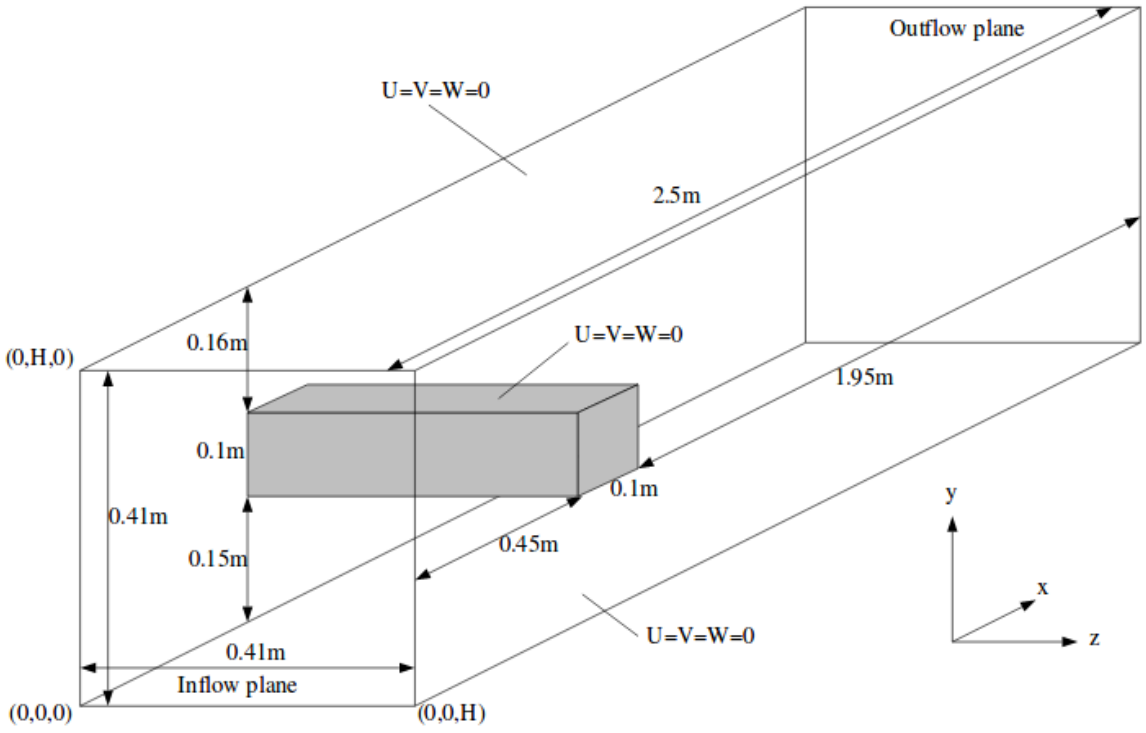


Figure 4.23: Configuration and boundary conditions of flow over cylinder with circular cross-section area

Mesh setup

The mesh setup was performed in a similar way as the previous test, here as well the entire geometry is divided into three different sections as shown in Figure 4.25. The most important part geometry is the region surrounding the cylinder, hence interms the mesh setup this region needs to be finer than the other two regions. Note that the same mesh size are also used for the respective surfaces pertaining to each the three volumes. Furthermore, the setting shown in this figure are for the embedded formulation since the region occupied by the cylinder needs to be mesh as well. Hence, for a fair comparison the same mesh settings are used for the body-fitted as well as the symbolic formulations. The resulting computational

mesh is shown in Figure 4.26. Notice the difference in the mesh sizes. The body-fitted and symbolic had 1739889 elements in each and the embedded had 2.6e6 elements.

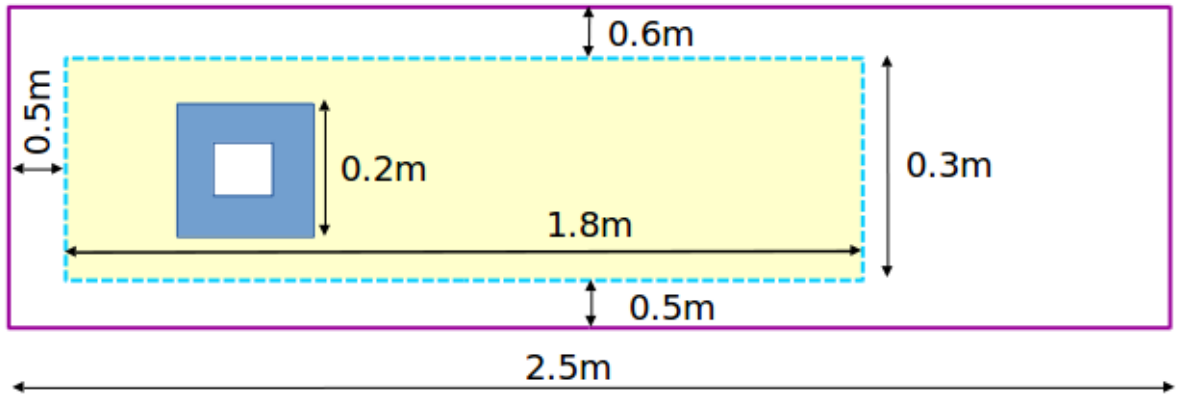


Figure 4.24: Mesh setup (Not drawn to scale)

Figure 4.25: Mesh sizes on the volumes

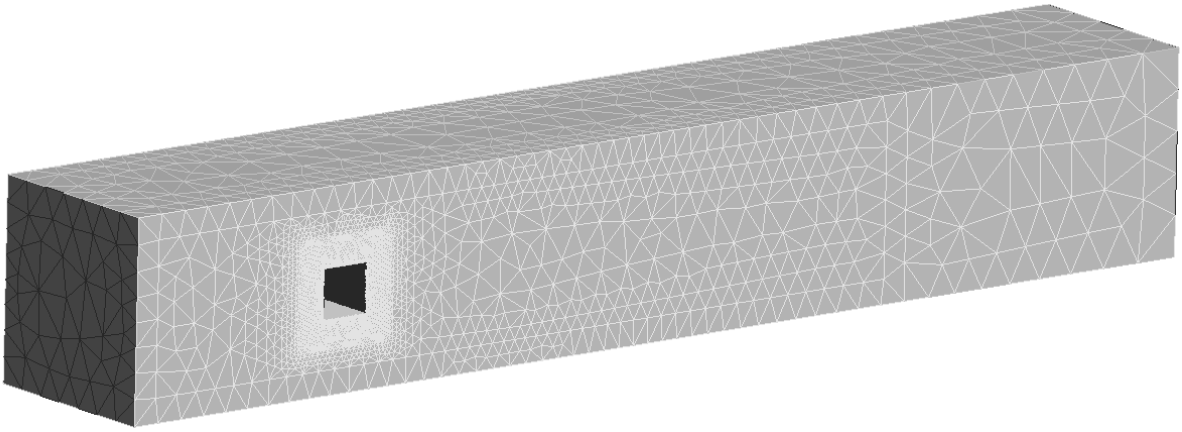


Figure 4.26: Resulting Mesh for cylinder with square cross-section area

Simulation settings

This simulation was carried out for a total of 20s using a time-step of 0.02s. The MultiLevelSolver with a tolerance of 10^{-6} was also utilized in this simulation to solver the resulting linear system of equation. In terms of convergence, a tolerance of 10^{-5} was set for both pressure and velocity. For the results the drag and lift coefficient are also calculated as well as the pressure and velocity distributions are elaborated on.

Results and analysis

First and foremost it must be noted that for this benchmark test, the reference data was not presented in the paper by Schäfer and Turek [26]. In fact it was stated that the problem turned out to be harder than expected and hence it is still considered to be an open question. The quest for answers motivated the inclusion of this problem among the validation tests. Nevertheless, this problem has been studied by Olshanskii en al [21] and hence for this test will use the reference data provided in this paper. Note however that even Olshanskii en al have stated that their reference data may not be very accurate as well.

Table 4.3: Case2: Comparison of maximum and minimum drag and lift coefficients

		Drag			**Lift**		
		Reference	Obtained	Rel. err[%]	Reference	Obtained	Rel. err[%]
Body-fitted	Max	4.67	4.51	-3.42	0.05	- 0.21	3.2
	Min	4.32	4.51	4.40	0.015	- 0.21	0.19*
Symbolic	Max	4.67	4.54	-2.78	0.05	-0.16	0.11*
	Min	4.32	4.54	5.09	0.015	-0.16	0.15*
Embedded	Max	4.67	4.42	-5.78	0.05	0.09	0.04*
	Min	4.32	4.42	2.31	0.015	0.09	0.07*

(Note that in the table * represents the absolute error since the values are close to zero)

Considering the calculation of the solution it was noted by Olshanskii en al [21] that is the singularity of geometry which is due to the edges of a square cylinder is likely to destroy the regularity of the solution. Furthermore, Schäfer and Turek stated that although a higher Reynolds number (although we have Re=100) is required for nonstationary 3D problems such as the one at hand, the results showed that the flow tends to be stationary. These observations makes the accurate numerical prediction of the lift and drag coefficients a difficult task. These issues are also observed in this work.

From table 4.3 we can see that the drag and lift values for our formulations are constant which means that the flow in each simulation became stationary after a period of time. Hence its non periodic. On the other hand, a closer look at relative errors coming from the embedded formulation shows that the errors are very high as compared with the body-fitted and symbolic formulations. These problem has to do with the sharp edges of the cylinder. This sharp edges are very difficult for the embedded formulation to predict. This can be observed by looking at Figure 4.28 where we can see that instead of having a

right angled corner the continuous distance function makes a slight approximation at a slight angle. This affects flow prediction at the separation edge which is located almost at the upper edges of the front face as it can be observed in the Figure 4.30 and 4.31. Note that figure 4.28 is just a small section of Figure 4.27.

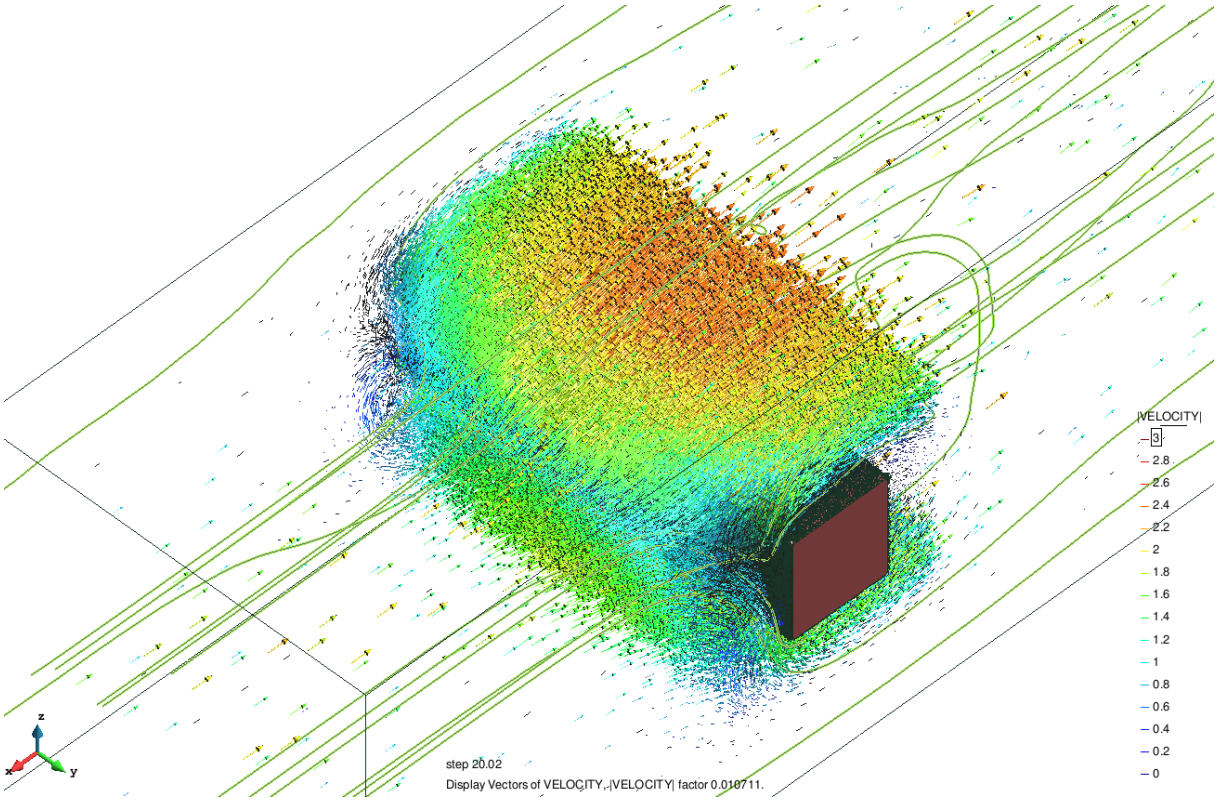


Figure 4.27: Streamlines of velocity using the Embedded formulation

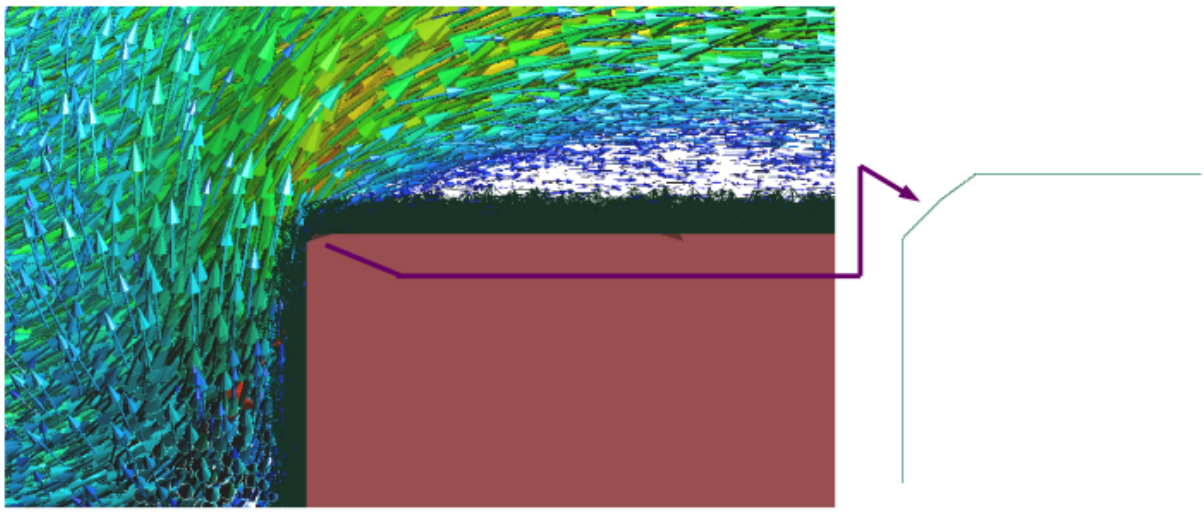
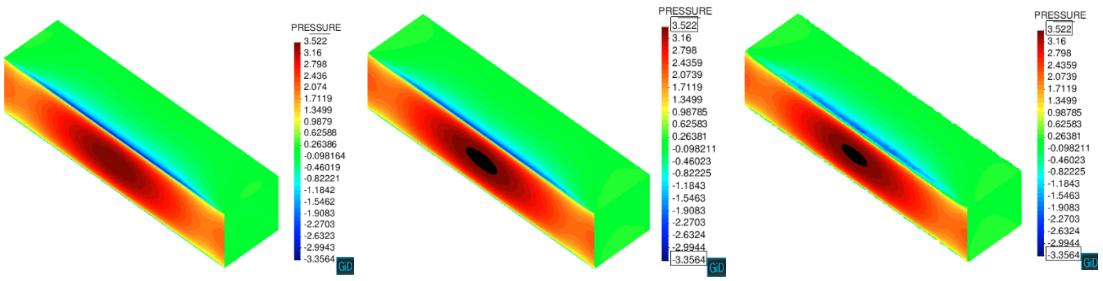
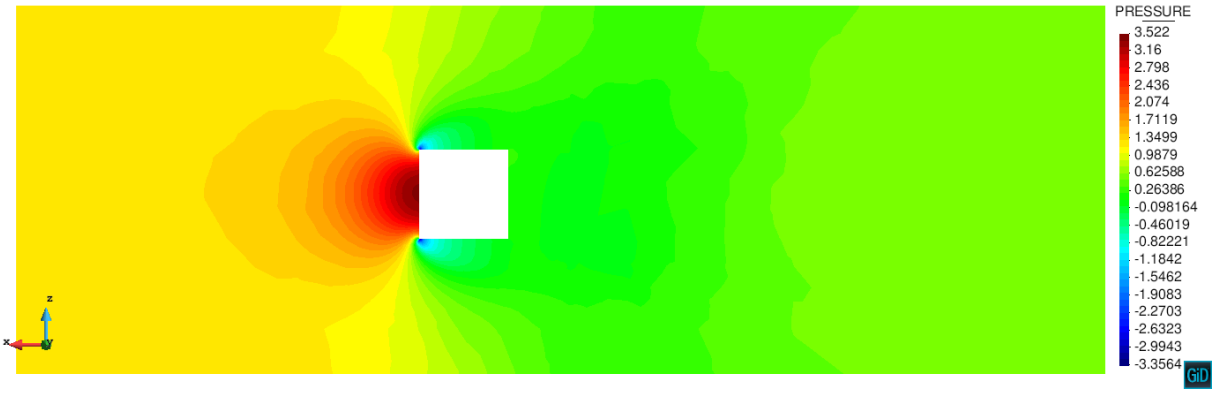


Figure 4.28: Embedded formulation sharp corner problem

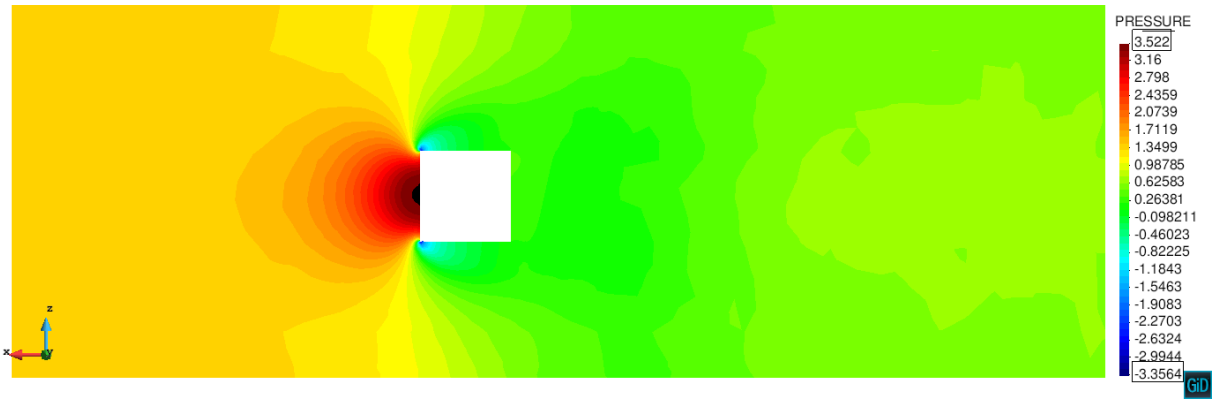


(a) Body-fitted (b) Symbolic (c) Embedded

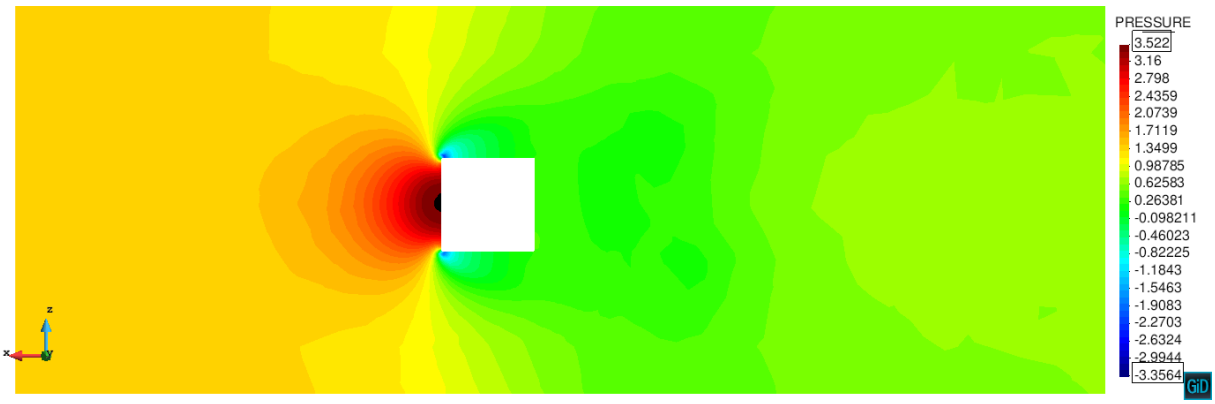
Figure 4.29: Pressure distribution across the cylinder for the three formulations at $t = 15$s



(a) Body-fitted

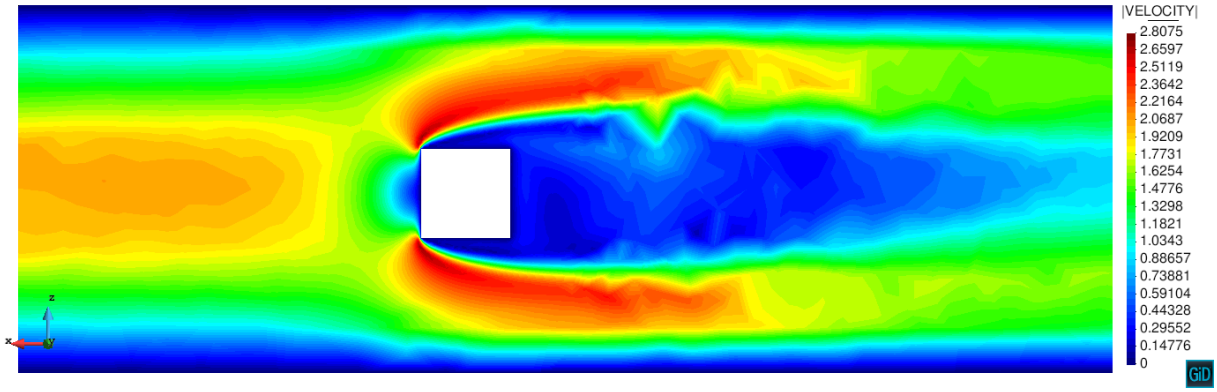


(b) Symbolic

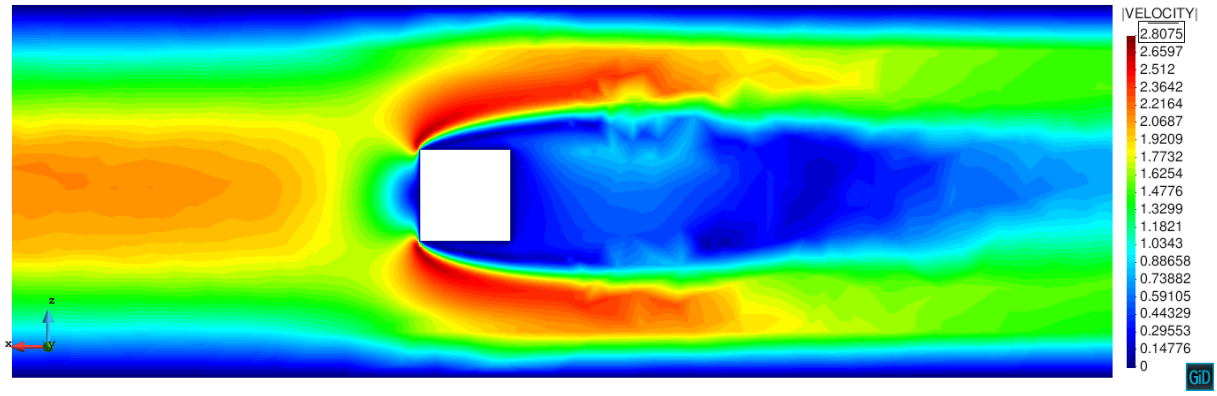


(c) Embedded

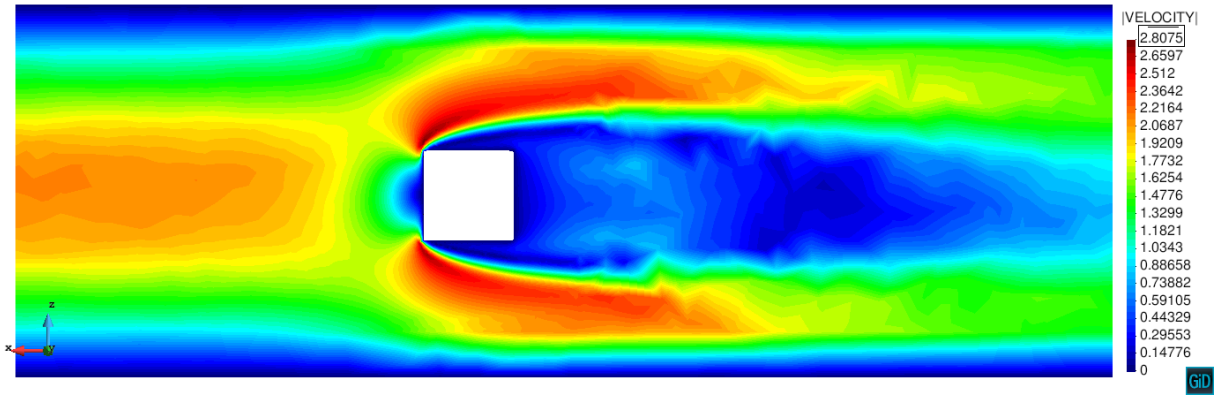
Figure 4.30: Pressure distribution for the three formulations at $t = 15$s



(a) Body-fitted



(b) Symbolic



(c) Embedded

Figure 4.31: Velocity at time $t = 15$s for the three formulations

4.3 CFD simulation of the Silsoe cube

Modelling a flow around low objects of non-aerodynamic shapes using both numerical and physical simulations brings about many problems. The aim of this section is to present the results and analysis of a far more challenging benchmark test in CFD. This analysis will deal with the numerical modelling of an air-flow around a 6.m cube. It represents the so called Silsoe cube which is a standardized experimental element in the field of building aerodynamics [20]. Based on the previous tests we have validated that both the symbolic and embedded formulations are capable of achieving good results. Hence for the current benchmark test we will only present results from the symbolic and embedded formulations.

The Silsoe cube experiment was carried out at the Silsoe research institute in the UK. The purpose of this experiment was to study wind flow around structures in full scale. Due to the cubic shape of the cube it was possible to carry out many tests as the cube has multiple planes of symmetry and hence it can be rotated to allow the measurement of pressures from different wind angles. The cube was equipped with sensors, which recorded the dynamic wind pressure with all three velocity components and upstream of the cube, the undisturbed flow parameters have been collected.

For the problem at hand will focus on the one where the wind direction is considered to be perpendicular to the front surface of the cube. Much data exists for this particular case and it has been used in many wind tunnel experiments and CFD simulations for validation. In our analysis will use the experimental data corresponding to the papers by Richards [23] [24]. On the other hand for the CFD simulations the model set up has been done using the parameters provided by Gerhard Steber [15] who validated the body fitted formulation in Kratos against the experimental results of the Silsoe benchmark. With respect to the embedded formulation, this example is suited to validate the embedded approach in particular, as the terrain around the cube is completely flat and no disturbance in the boundary layer profile by terrain effects are to be considered [22].

Model set up and boundary conditions

The set up of the entire model is shown in Figure 4.32 with the cube placed at the inside a model of a wind tunnel. The distance between the inlet surface and the cube is long is enough to allow the flow to fully develop.

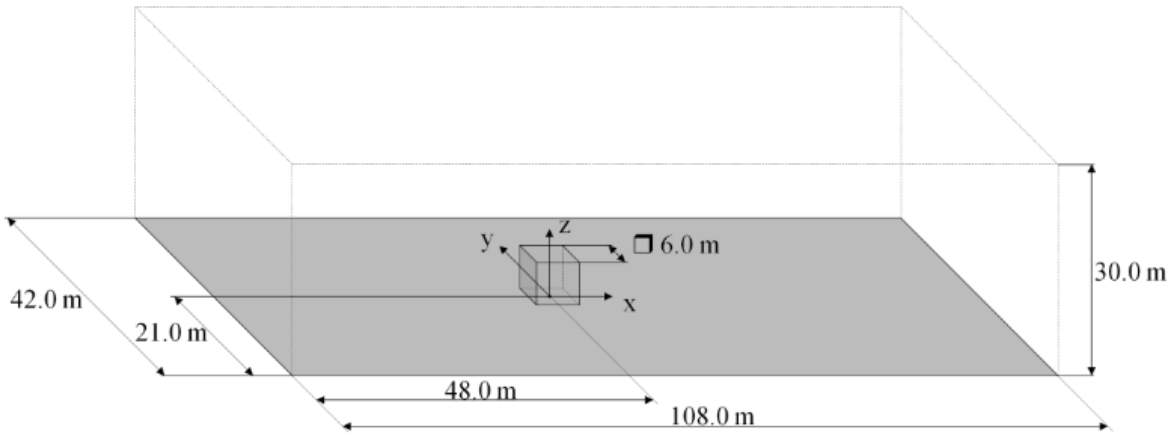


Figure 4.32: Dimensions of the Silsoe cube and the surrounding domain(Adopted from [15])

The boundary conditions of the entire model are shown in Figure 4.33. Here the top and side walls which accounts for the open area in the real experiment are prescribed with a slip boundary condition, this is done in order to avoid any shear stresses which influence the velocity profile alone these surfaces. Both the bottom surface and surfaces around the cube are prescribed with a no-slip boundary condition order to create the shear stresses and the correct velocity profiles. Note that in the case of the embedded formulation this boundary condition is imposed weakly using the distance function and the Embedded no-slip formulation which we have already discussed in the previous chapters. On the other hand a pressure of zero is imposed on the outlet surface in order to control the flow direction.

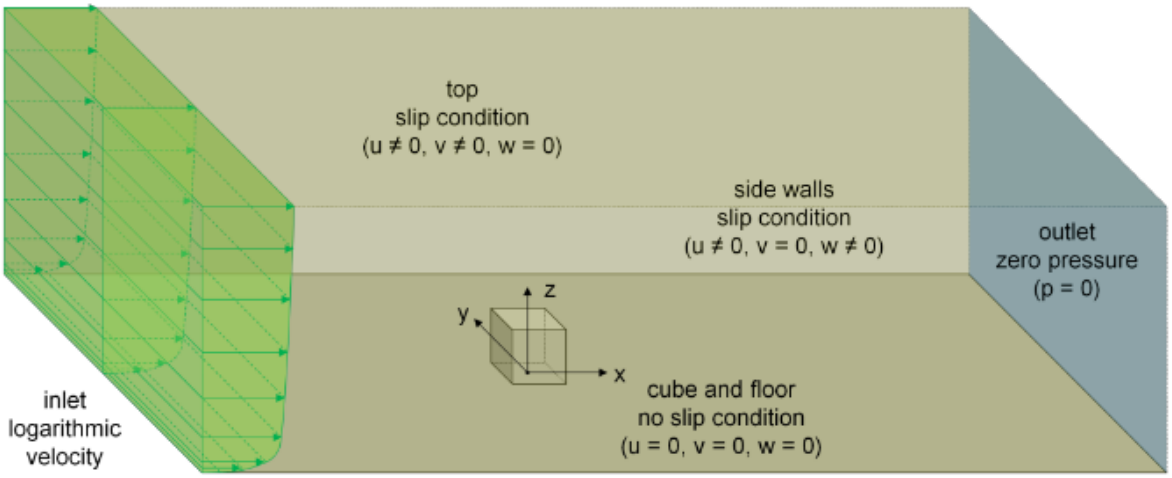


Figure 4.33: Boundary conditions in the entire domain (Adopted from [15])

Considering the inlet surface, a logarithmic inlet profile is imposed, this profile is not time dependent and was derived by Gerhard [15] to fit the data given in [24], hence resulting in the following equation:

$$u(z) = 0.272\left(\frac{1}{0.41}log\left(\frac{0.272}{1.51\cdot 10^{-5}}z\right)+5.2\right) \tag{4.4}$$

This inlet profile also provides data which is similar to the experiment as shown in the table below. The

Table 4.4: Silsoe cube velocity profile

Z [m]	**Real Experiment Velocity** [m/s]	**Numerical Model Velocity** [m/s]
1	6.97	7.92
3	8.65	8.65
6	9.52	9.11
10	10.13	9.45

fluid is considered to be air with density $\rho = 1.21\frac{kg}{m^3}$ and dynamic viscosity $\mu = 1.85\cdot 10^{-5}\frac{kg}{ms}$ which leads to a Reynolds number of $Re = 4.05\cdot 10^{-6}$.

Mesh parameters

In order to avoid the use of so much computational cost the entire fluid domain has been divided into different regions with each region having its own mesh parameters as shown in Figure 4.34. The most

important part of the velocity profile is the lower section as it is the one that comes into contact with the cube. Hence, the expected path of this section needs to be well meshed which has led to the use of different mesh-boxes. Using these mesh settings leads to a total of 7.2e6 elements. Note that all the fluid elements inside the embedded structure are deactivated during the solving process. As it can been seen

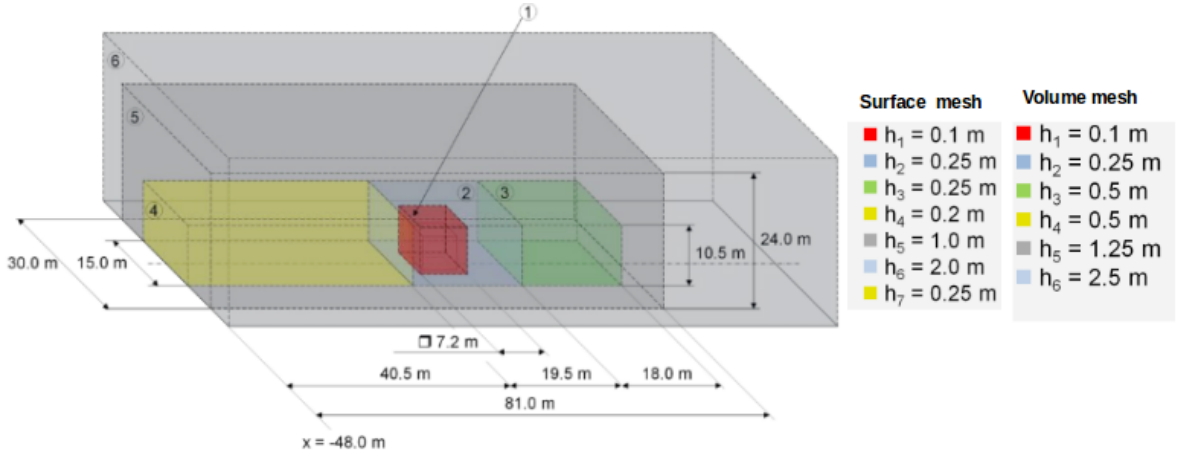


Figure 4.34: Domain setting for efficient meshing using fine mesh data (Adopted from [15])

in Figure 4.35 the region around the cube has a much finer mesh than the outer domain in order to firstly be able to resolve the flow details close to the cube. Secondly its the region used by the distance function to approximate the cube geometry in the embedded formulation. Note that the green part in Figure 4.35

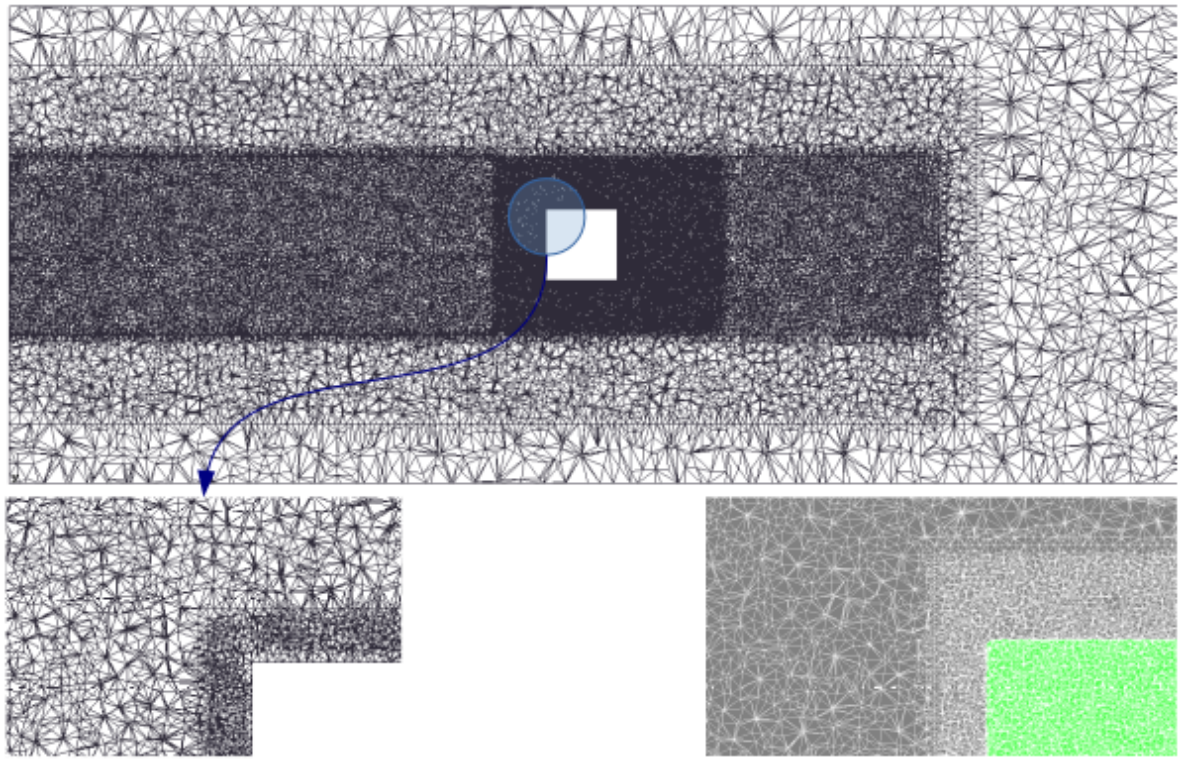


Figure 4.35: Cutting plane of fluid mesh across the Z=3 plane

indicates the embedded cube. The full cube approximated by the continuous distance function is shown in the figure below. Likewise it important to remember that the distance value is slightly changed at certain locations using the distance modification process (see chapter 3), where the embedded structure

would cut an element in a way, that would lead to a very distorted sub-element. These changes are vital as they enhance the stability of the method.

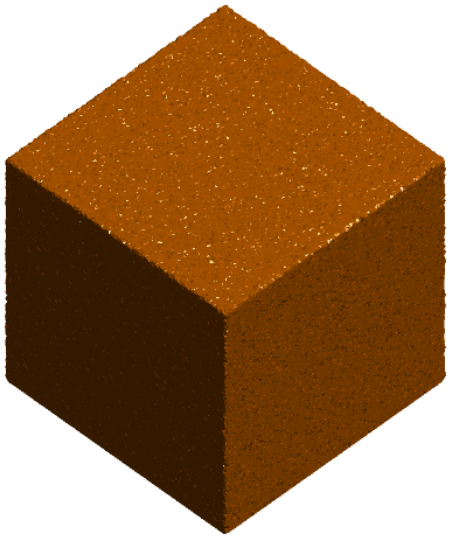


Figure 4.36: Approximated cube suing the continous distance function.

Simulation setting

The simulation time was set to 80 s with a time step of 0.1 s in order to allow enough time for the full development of the flow. The resulting linear system of equations was solved with a linear solver tolerance of 10^{-6}. The convergence criteria were set to 10^{-5} for both velocity and to pressure. To compare the pressure results of the simulation to that of the wind tunnel and full scale experimental model, the mean pressure coefficient c_p is introduced and calculated as discussed in section 2.7.3. Using this coefficient allows us to scale the pressure to a generally comparable dimensionless number. The free stream variable to bused when calculating 2.7.3 were collected at point (-23.4m; 6.24m; 6.0m). This point was suggested by Richards [24] and utilized also by Gerhard [15]. Furthermore, Richards also suggested that the evaluation lines along which the pressure field is supposed to be captured be in accordance with the figure below.

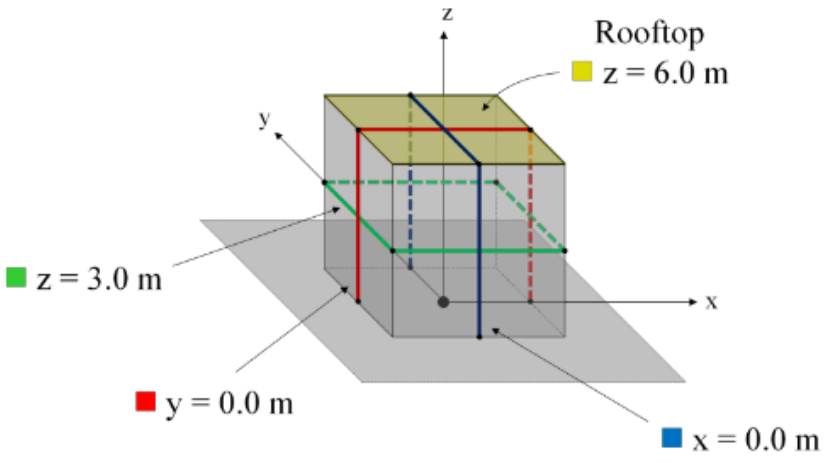
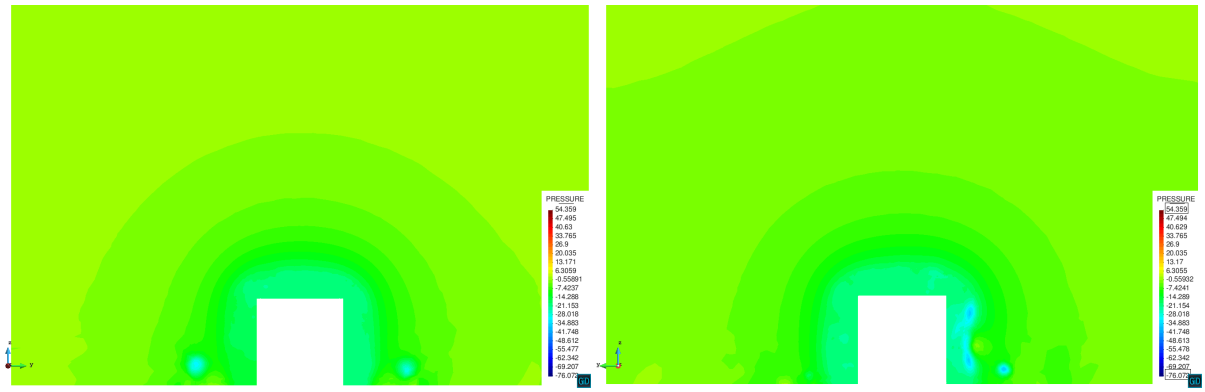


Figure 4.37: Evaluation lines for the pressure field. (Adopted from [15])

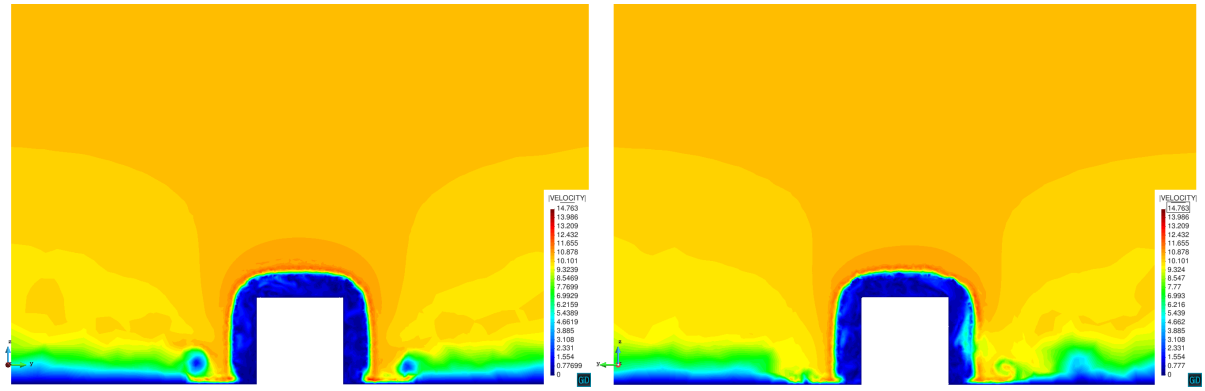
Result analysis along the x = 0.0 m Cutting plane

The pressure and velocity distribution for both the symbolic and embedded formulations are shown in Figure 4.38 and 4.39 respectively. For a fair comparison between the two formulation we have used the same colormap. Hence, as the flow is perpendicular to this plane we expect a symmetric distribution of both pressure and velocity. Although both formulations have shown very good distribution in both pressure and velocity, as expected the symbolic formulations shows better distribution. This can be noted in Figure 4.39 where there is a slight change in the velocity distribution around the right surface of the cube. This area should have the lowest value in velocity due to the applied boundary condition.



(a) Symbolic (b) Embedded

Figure 4.38: Pressure profile across the x-axis



(a) Symbolic (b) Embedded

Figure 4.39: Velocity profile across the x-axis

To compare the pressure results with both the experimental and wind tunnel data, a c_p comparison plot is shown in Figure 4.40. Here we see that the behavior of both the symbolic and embedded is very similar. As we can see the results of both formulations lies within the corridors of the wind tunnel results, although compared with the experimental results, they are not as expected. This is especially shown by this diagram as at positions 1 and 2 where the plots are almost constant. Notice that this part of the plane is behind the separation edge hence the results are directly affected with how the each formulations handles the flow around the separation region.

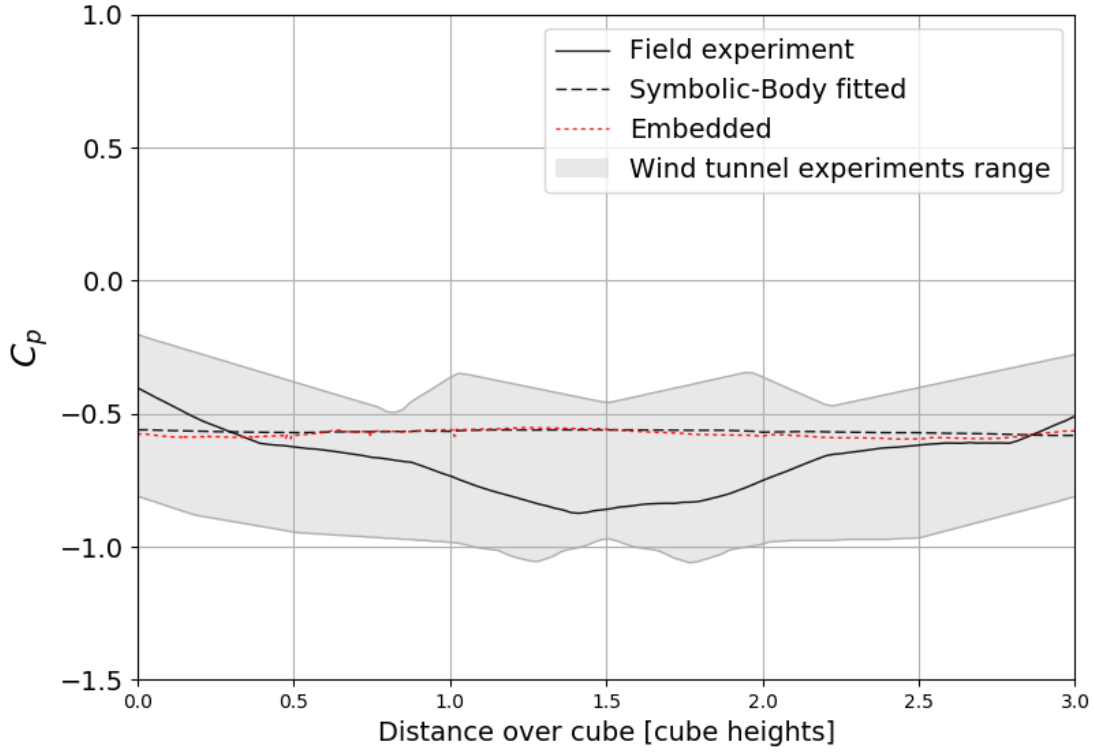


Figure 4.40: Comparison of mean pressure coefficient along the x = 0m cutting plane

Result analysis along the y = 0.0m Cutting plane

The fundamental characteristics of the flow around cube can further be observed by looking at the pressure and velocity distribution along the y = 0.0m cutting plane. These distributions are shown in Figures 4.41, 4.42 and 4.44, 4.43 for both simulations. By looking at the pressure results in Figure 4.41 and 4.42 it can be noted that as the flow approaches the front part of the cube it begins to be stagnate hence forming a stagnation pressure zone. Furthermore we can see that the applied no-slip boundary condition on the ground which results in the formation of shear stresses and the implied large local pressure gradient contributes to the development of the horseshoe-vortex [4].

The distance between the cubes front face and the location of the horseshoe vortex is a good determining factor of the ability of a given formulation to predict better pressure distribution along the front face. For the two formulation this distance is almost the same. The effect of this distance can be seen by looking at the c_p plot in Figure 4.45, here we can see that along the stagnation zone which is the position 0.0 - 1.0 cube heights, both the symbolic as well as the embedded CFD results tend to slightly over-estimate the pressure, especially at the beginning (between positions 0.0 and 0.5). Nevertheless, as we go up the prediction of both formulation is almost aligned with the experimental (see position 0.5 to 1.0).

The separation region can observed along the edge separating the front face of the cube and the rooftop. This is a very complex and sensitive region. In both the pressure (Figures 4.41, 4.42) and velocity (Figures 4.44, 4.43) results its clear that once the flow passes this region it begins to accelerate at a faster rate which leads to high turbulence and henceforth resulting in the generation of vortices

above and behind the cube. The pressure and velocity distribution in both simulations seem to be the same.

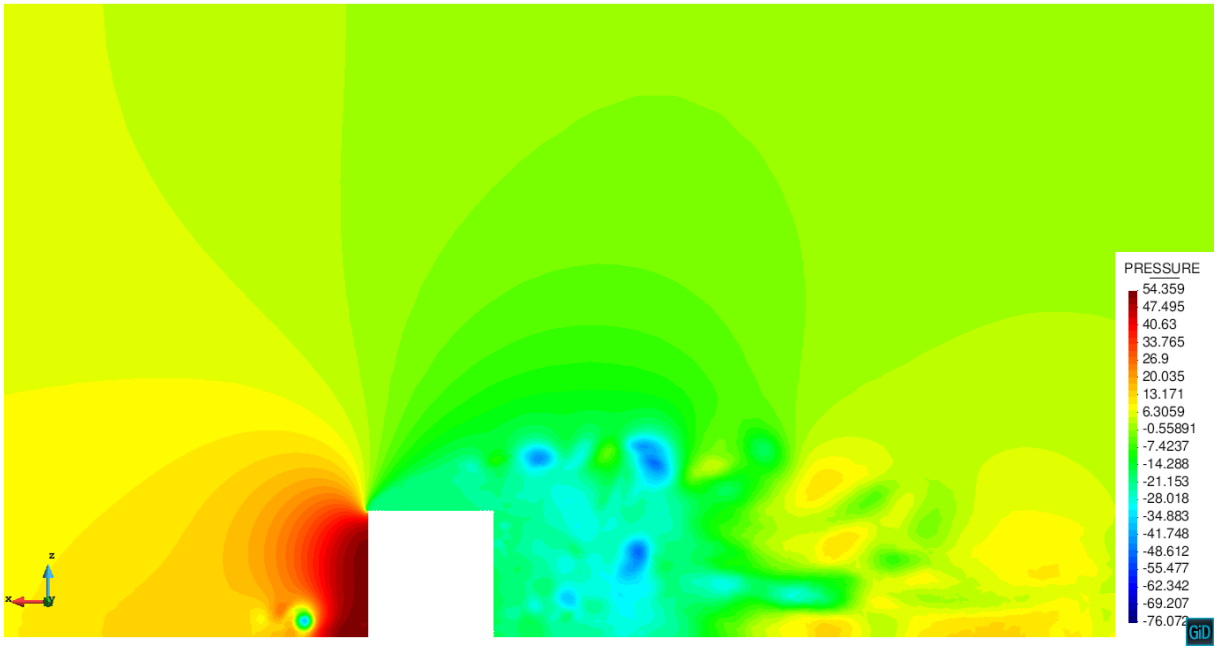


Figure 4.41: Pressure distribution in the y = 0.0m cutting plane using Symbolic

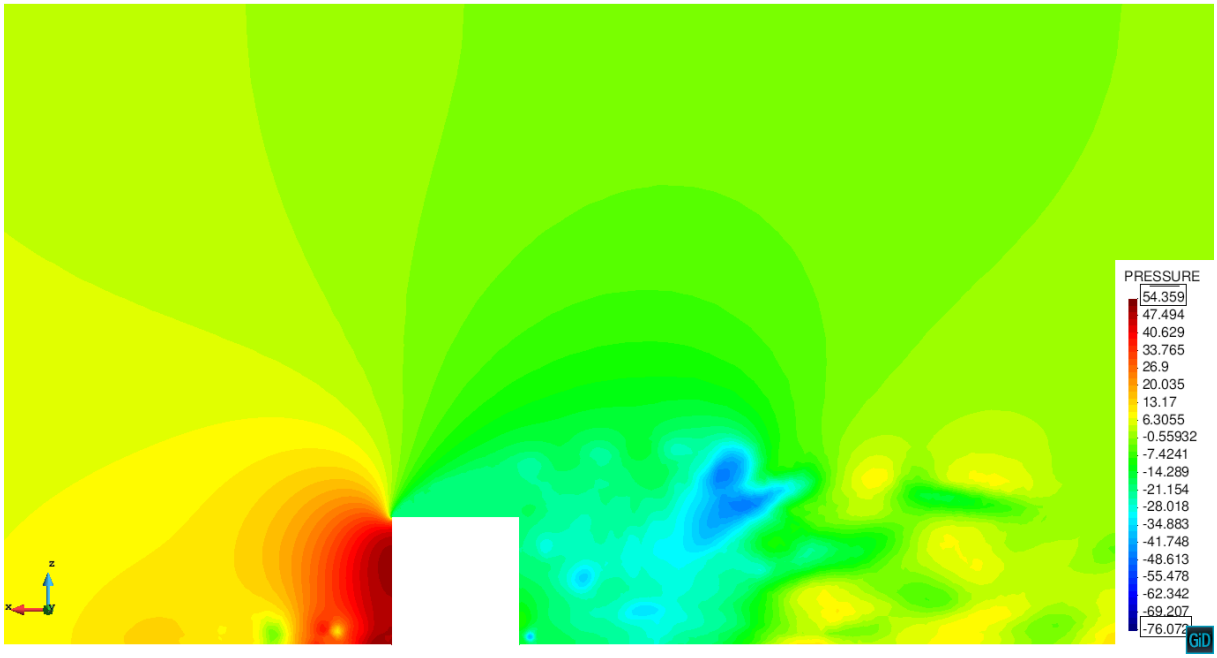


Figure 4.42: Pressure distribution in the y = 0.0m cutting plane using Embedded

As the flow becomes more turbulent large deviations in the pressure are observed as it can be seen between the positions 1 and 2 in Figure 4.45. Here both the symbolic and embedded formulations struggles to correctly represent the pressure field behind the separation edge. Further observations show that the results are not even with the range of the wind tunnel results. This behavior is also observed in the back face of the cube represented by the the positions 2 and 3. The similarity in the behavior of the

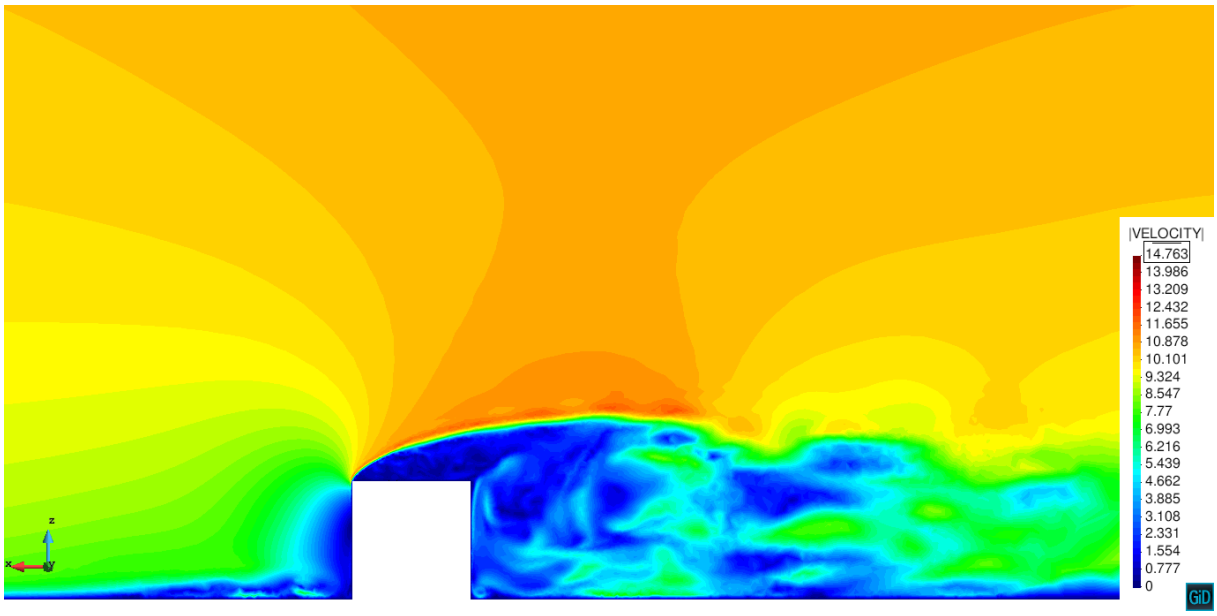


Figure 4.43: Velocity distribution along the y = 0.0m cutting plane using Embedded

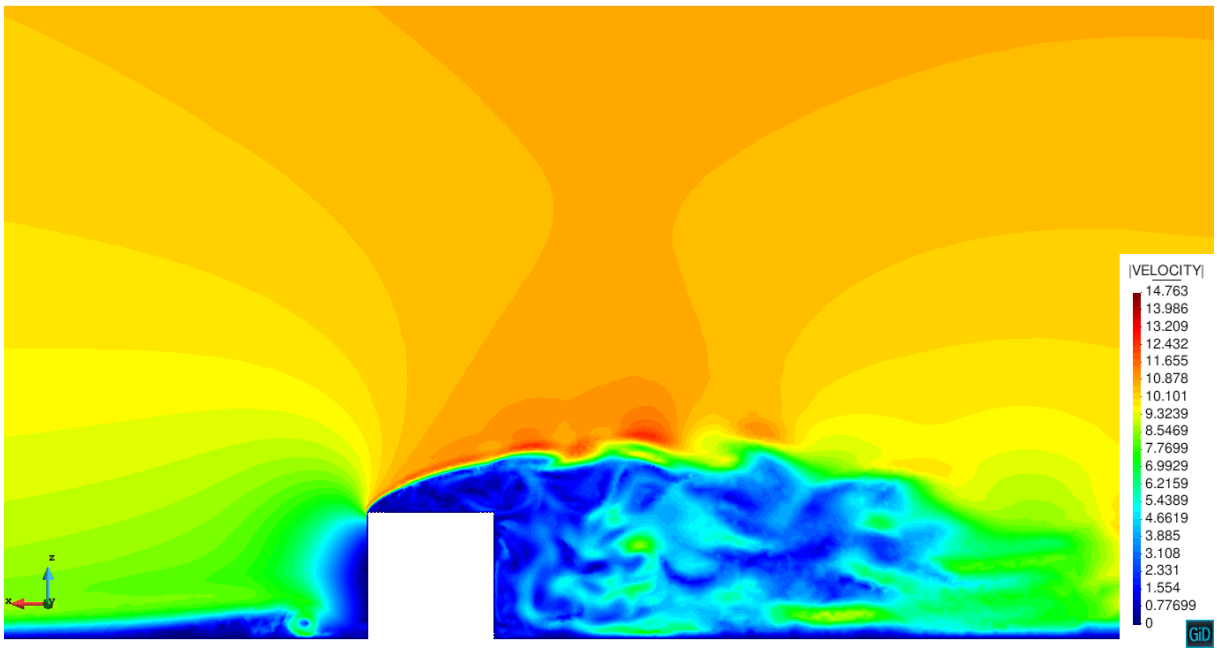


Figure 4.44: Velocity distribution along y = 0.0m cutting plane using Symbolic

methods shows that the with a finer mesh and the distance modification process the embedded method is not very much affected by the use of the continuous distance function. Notice however that if this will not be the case when using a coarse mesh.

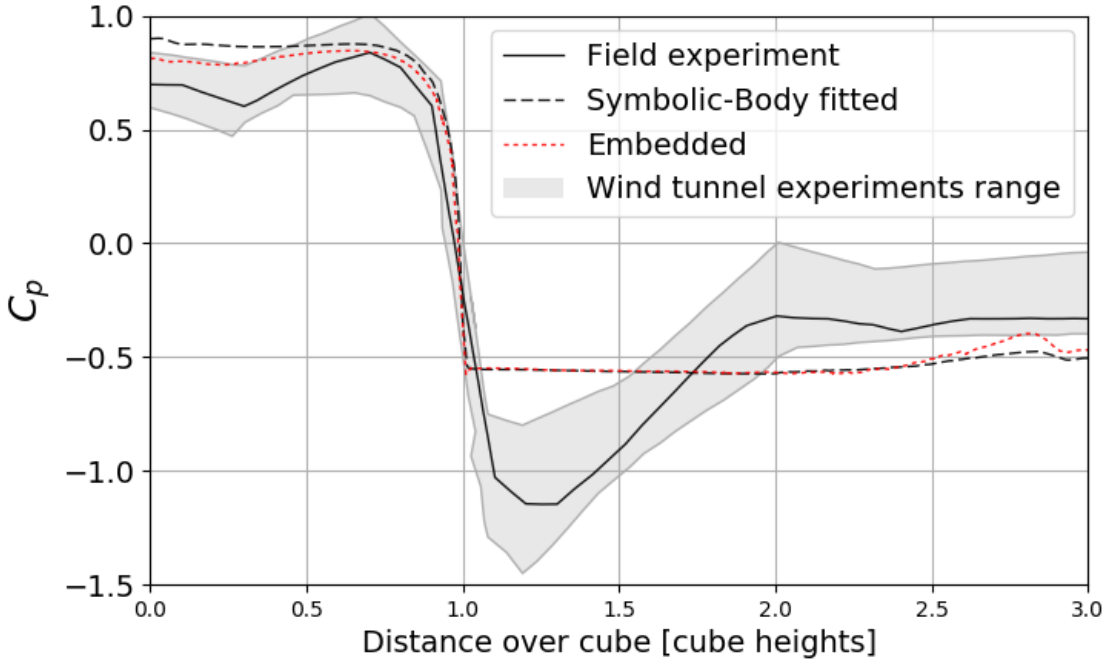


Figure 4.45: Comparison of mean pressure coefficient along the y = 0m plane

Result analysis along the z = 3.0m Cutting plane

The results obtained from both formulations shows a similar pattern behavior as we have discussed previously. At the front face, the results are very much aligned with that of the wind tunnel and experimental. However behind the point of flow detachment the two formulations do not show very good results, as observed at positions between 2 and 3.0 in Figure 4.48.

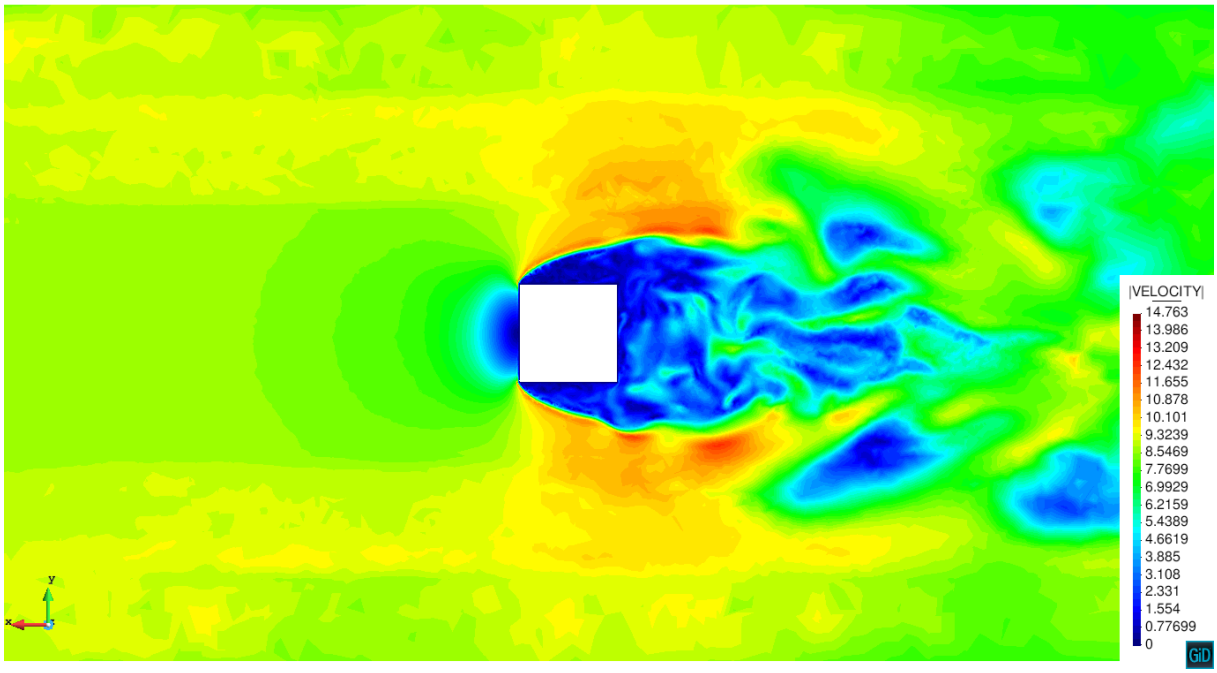


Figure 4.46: Velocity distribution in the Z -axis using Symbolic

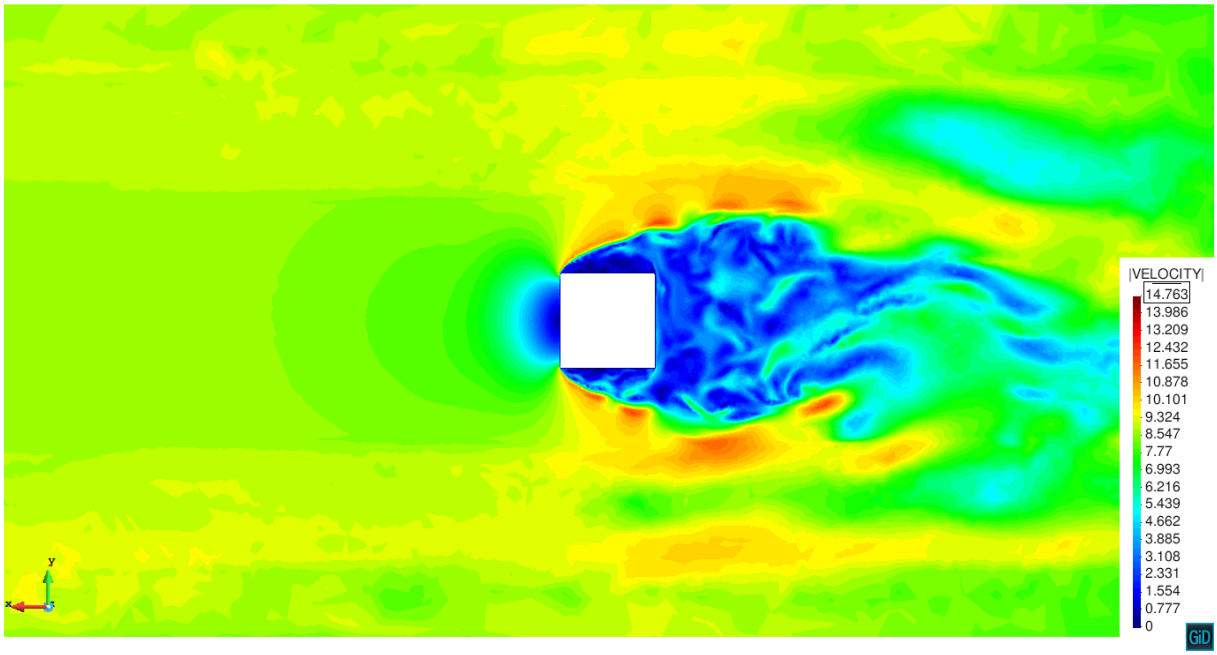


Figure 4.47: Velocity distribution in the Z -axis using Embedded

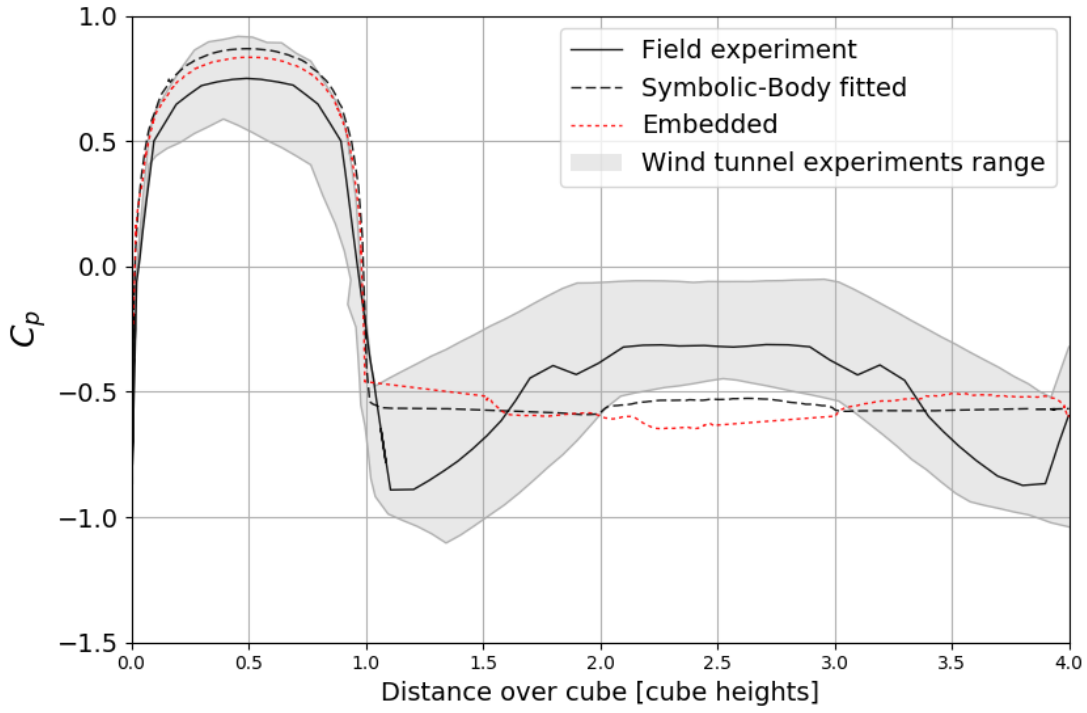


Figure 4.48: Comparison of mean pressure coefficient along the z = 3.0 m plane

Another inlet profile

It is clear from the results we have obtained in the previous analysis that the inlet velocity profile condition has a very big influence on the result produced by the two formulation. Hence in this section we have decided to use an type of inlet function in order to see the effect of the inlet profile by comparing the results with the previous results. As we can see in Figure 4.49 the key parts of the velocity profile is the bottom section .

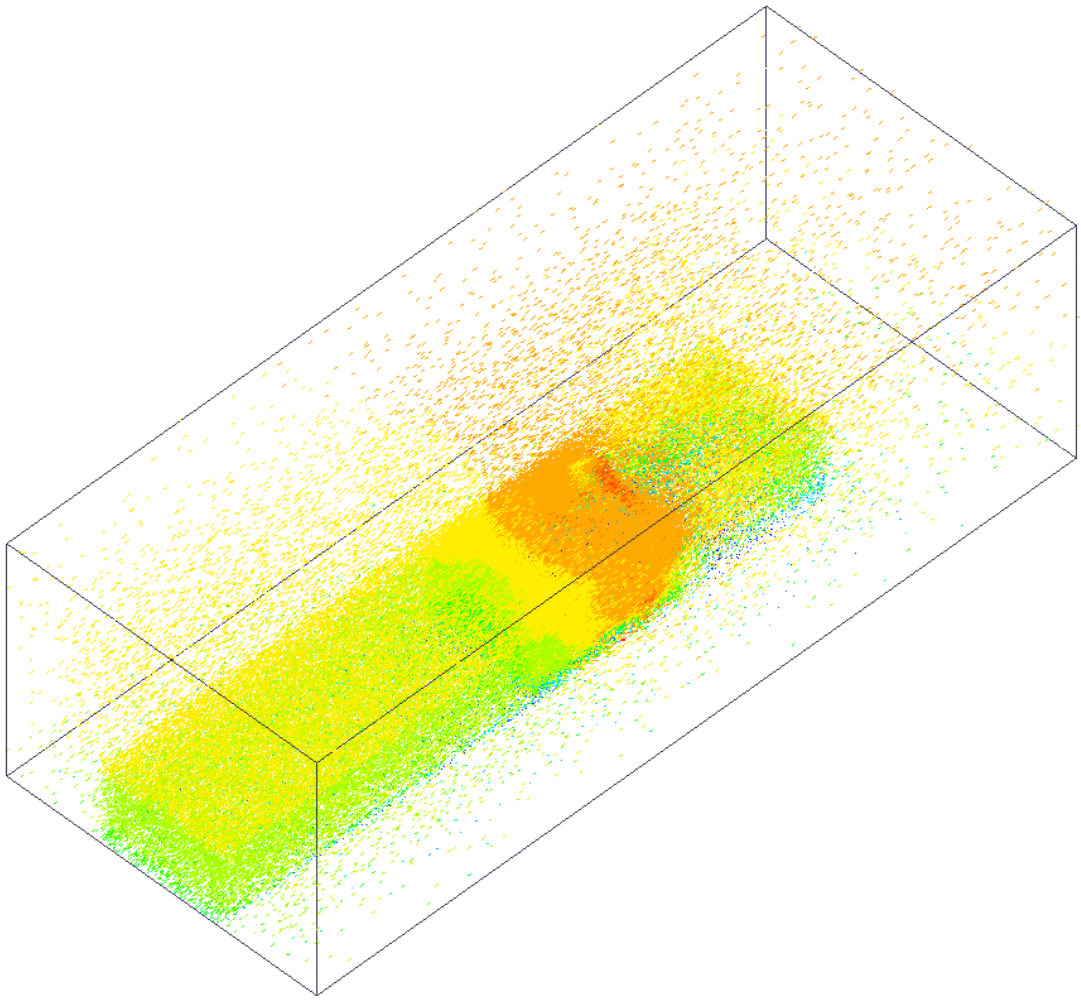


Figure 4.49: Concentration of the velocity isolines

The second inlet function is take to be

$$14.7\left(\frac{\log((z+0.01)/0.01)}{\log((25+0.01)/0.01)}\right)\text{if}(z>0.01)\text{else}1.3 \tag{4.5}$$

To compare both velocity inlet profiles the plot for each profile are shown in Figure 4.50. The y- axis for the plot represents the height of the geometrical domain(model of the wind tunnel). From the graph we can the difference in the two inlet profiles. Hence it is expected that the two profiles will certainly produce different results.

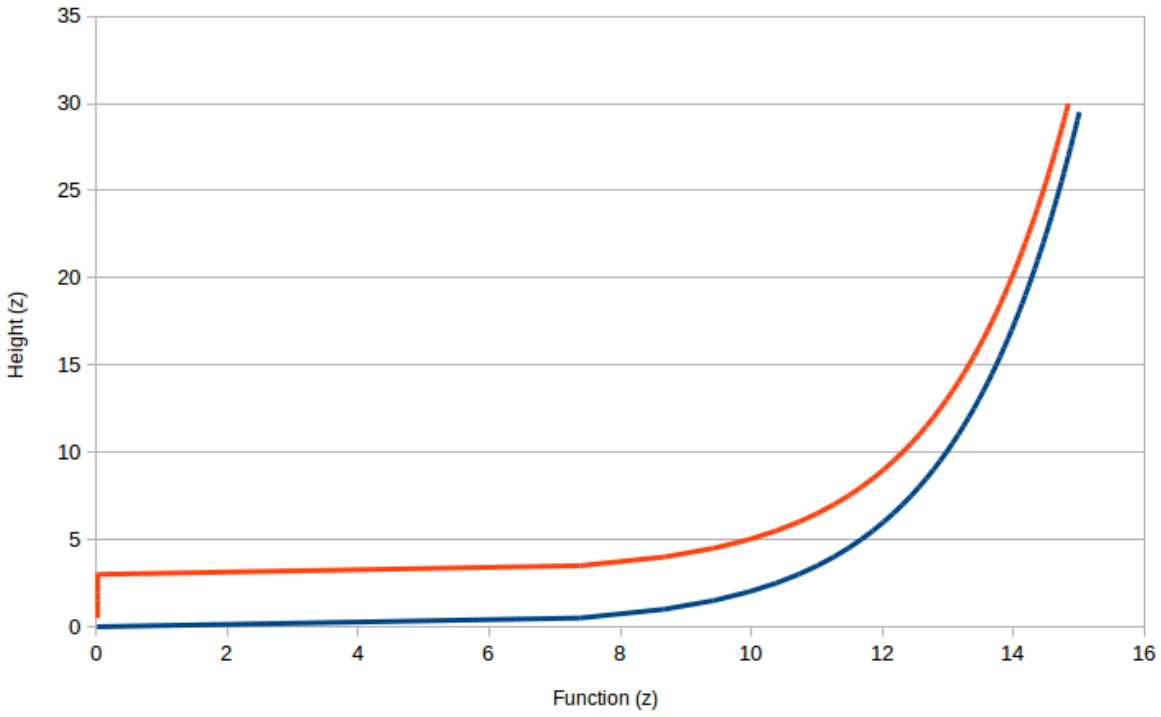


Figure 4.50: Comparison of inlet profile *Orange = First inlet , *Blue = Second inlet profile

Results and Analysis

The plots for the coefficient of pressure comparison are shown in both Figure 4.51 and Figure 4.52. By looking at the c_p comparison along the x=0.0 m cutting plane in Figure 4.51, we can see that the pressure values produced by the second inlet profile are very high, hence resulting in very low values of c_p. This graph also shows that the plot tend to go over the corridors of the wind tunnel range especially at the left and right side edges of the cube.

Furthermore, Considering the behavior of both inlet conditions along the cutting plane y = 0.0m In Figure 4.52, we see that within the stagnation zone (between position 0 to 1) the second inlet profile over predicts the pressure values leading to the graph been over the expected range of c_p. On the other range at the separation edge this profile seem to be closer to the experimental values although as we can see that the results produced in the region behind the separation edge are not as expected.

In terms of the pressure distribution along the y= 0.0m cutting plane we can see in Figure 4.54 that the location of the horseshoe vortex is far from the leading face of the cube when compared with the corresponding results in Figure 4.53. This further contribute to the unexpected result that we ave observed with this profile. Further more we see side there are no vortices produced by the second inlet profile in the regions behind the separation edge. This behavior is also depicted in along along the z = 3.0m cutting plane as shown in Figures 4.55 and 4.56 respectively.

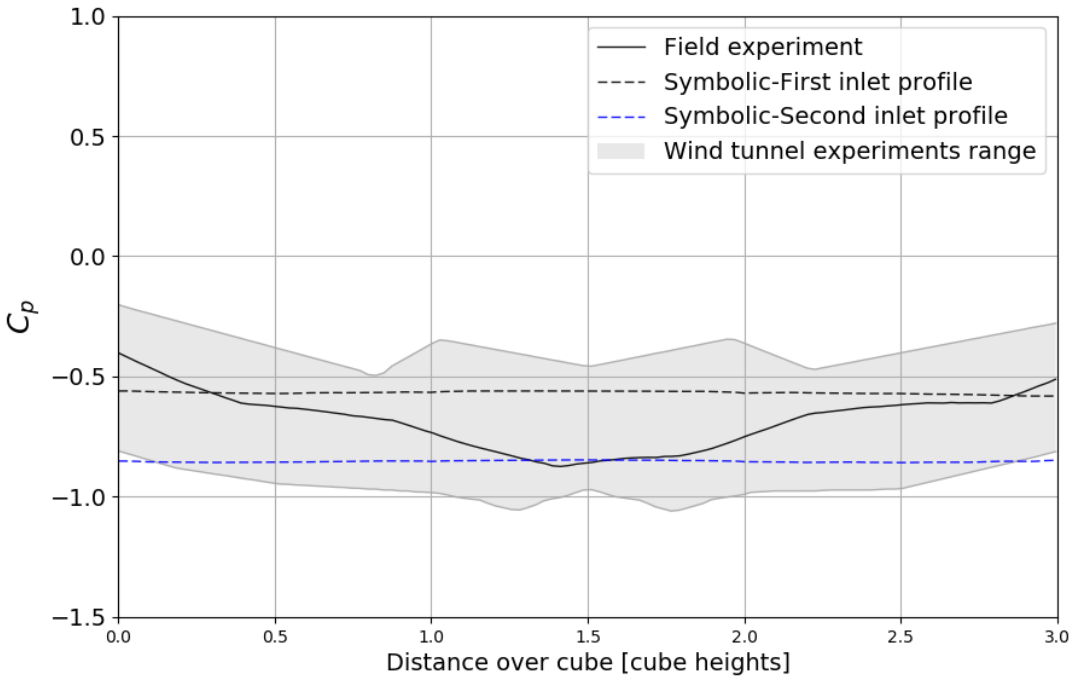


Figure 4.51: Velocity distribution in the y -axis Embedded

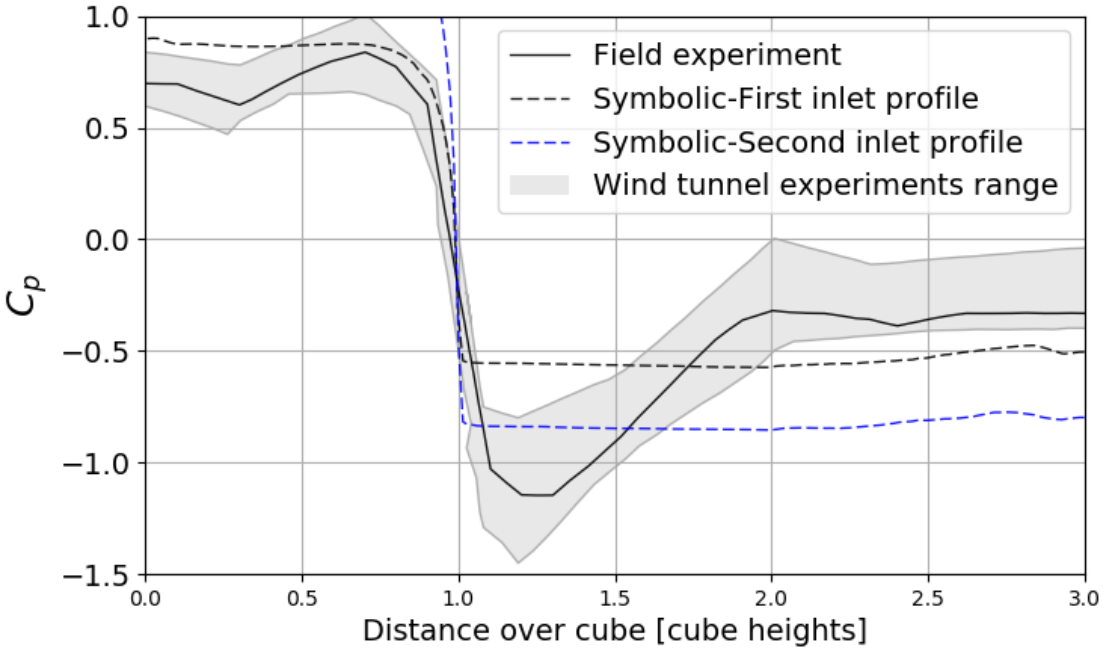


Figure 4.52: Velocity distribution in the y -axis Embedded

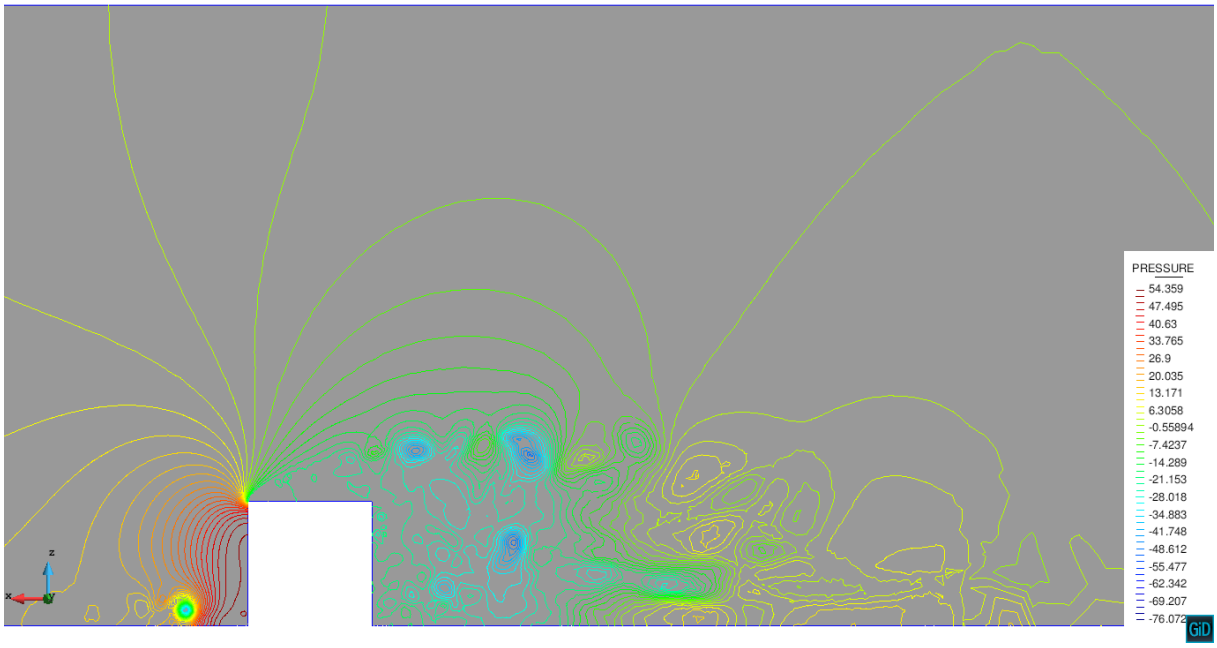


Figure 4.53: Pressure distribution in the y = 0.0m cutting plane using the first inlet profile

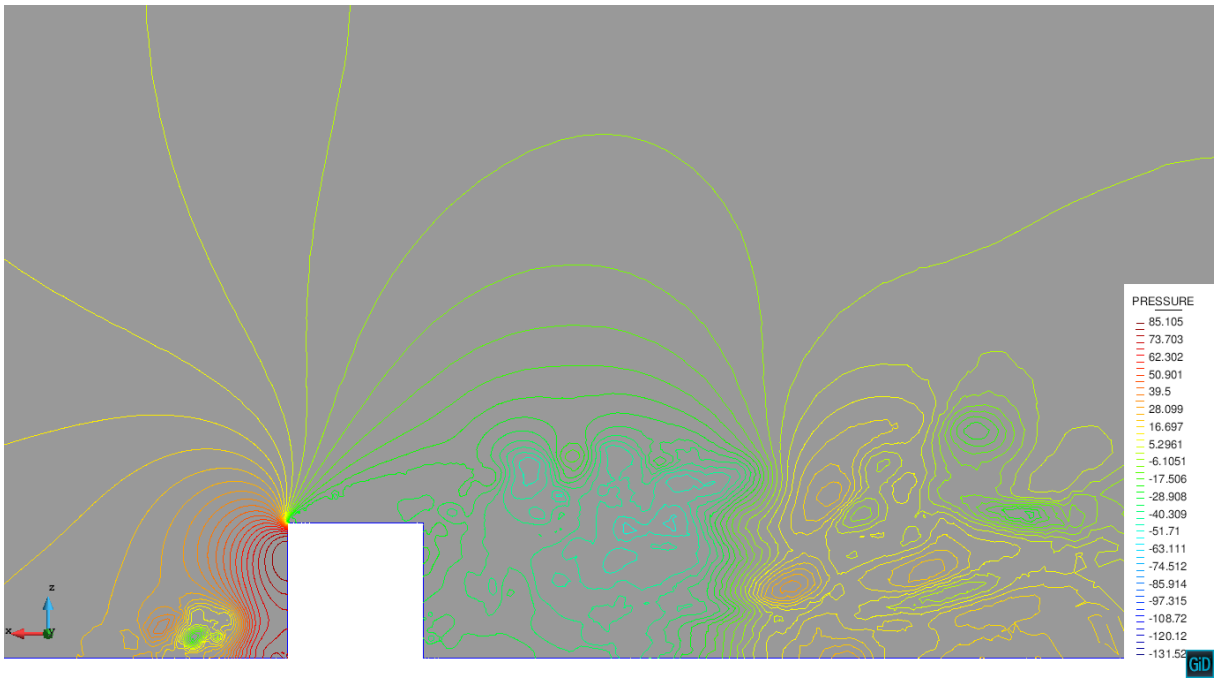


Figure 4.54: Pressure distribution in the y = 0.0m cutting plane using the second inlet profile

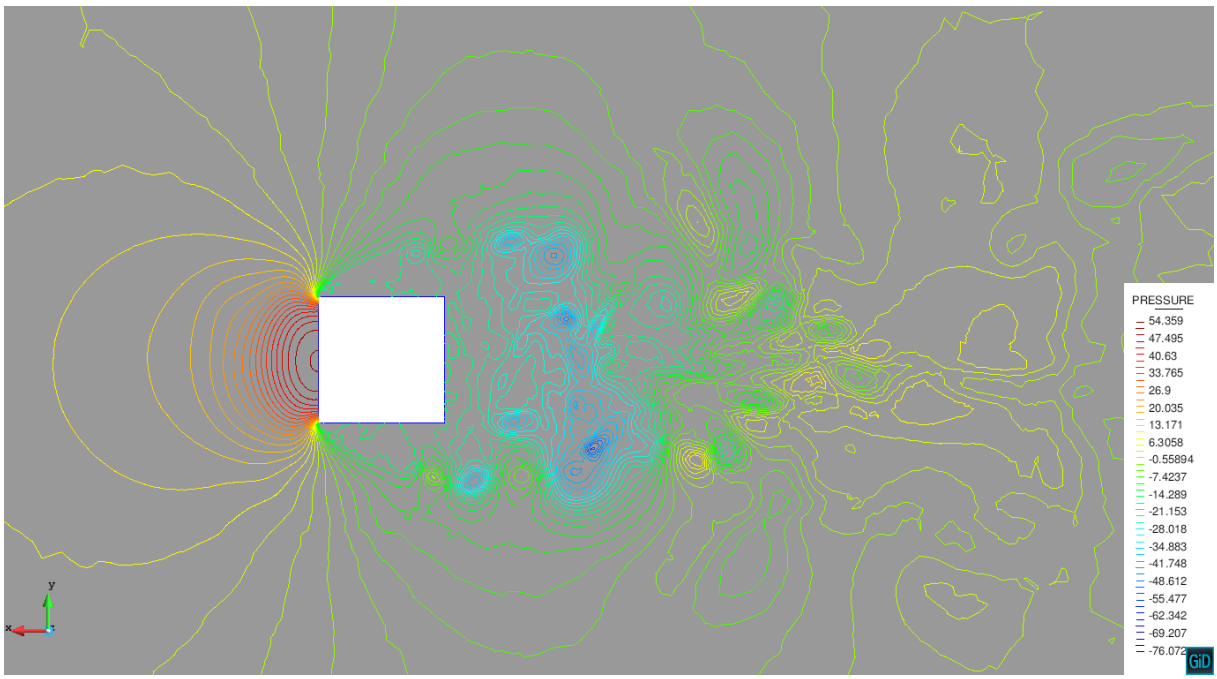


Figure 4.55: Pressure distribution in the z = 3.0m cutting plane using the first inlet profile

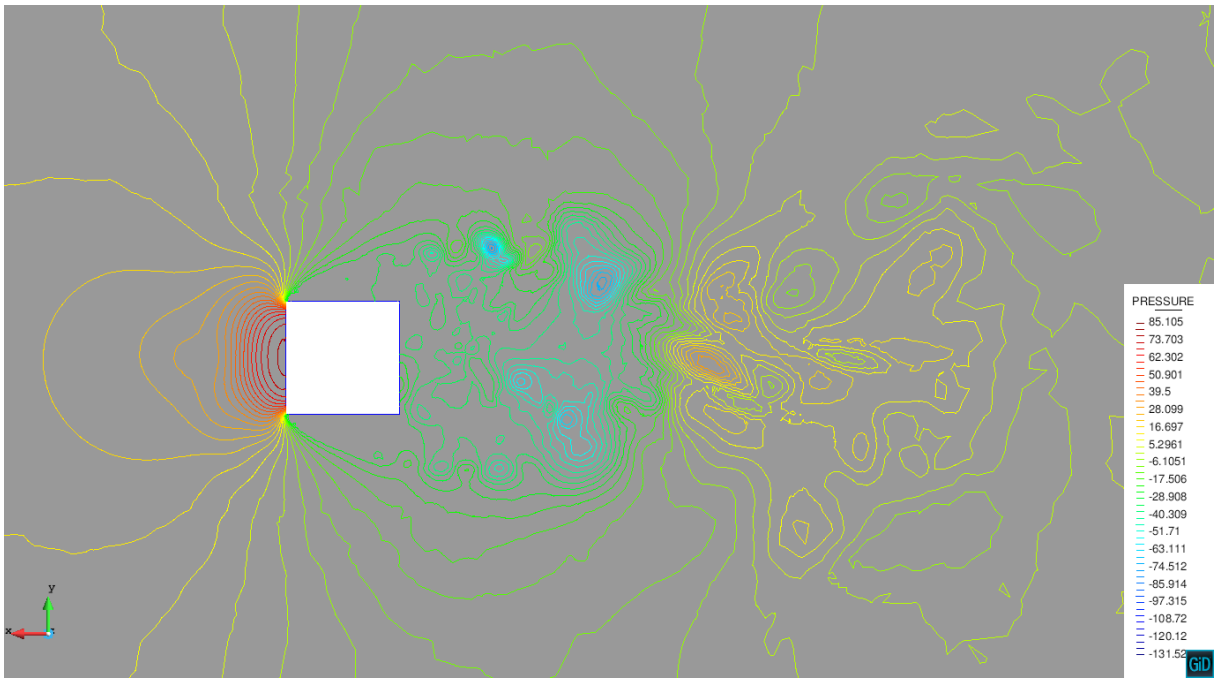


Figure 4.56: Pressure distribution in the z = 3.0m cutting plane using the second inlet profile

Conclusion on silsoe cube simulation

The solution of this problem proved that a flow around an object of non-aerodynamic shape is a complex action for modeling. This is due to a high Reynolds number which results into turbulence flow. Also the boundary layer is very thin which requires a very fine mesh around cube surrounding. Furthermore the experimental result comes from the experimental data which includes things like atmospheric effects. This clearly poses a big question on what type of inlet profile is needed, maybe we need to use a periodic one, otherwise the issue of having a good solution does not lies in the formulation rather the type of inlet profile in use.

Chapter 5

Conclusion

The aim of this study was to validate two computational fluid dynamics formulations, these been the symbolic body-fitted and the embedded formulations. These formulations have been implemented in the fluid dynamic application of the multi-physics solver Kratos. The key difference between the two formulations is that the symbolic formulations uses a body fitting mesh while for the embedded formulation the structure is immersed in the computational mesh. The purpose of the validation process was to solve 2D and 3D cases to prove the correctness and also to check the formulations capabilities in some complex examples. The entire validation process was carried out against the results of the already validated body-fitted formulation and the results from wind tunnel and/or experimental results.

To have a a firm fundamental background of the two formulations validated in this work, the basics of computational fluid dynamics, variational multiscale methods and the finite element method were discussed. Furthermore the key difference between the body-fitted and the embedded formulations was discussed as well. With respect to the implementation of the embedded method in Kratos multiphysics , the method used to weakly impose the Dirichlet boundary is discussed. Furthermore, the continuous distance function which is used for the purpose of interface tracking was elaborated together with the distance modification algorithm. In terms of the pre and post processing of the validation problems, the personal and post processor GiD that was used in a cooperative manner with Kratos is presented as well. It is suffice to say that with respect to the modeling and post processing of the results using GiD we did not run into any problems for all the cases that were tested.

The formulations were first of all tested to compute flow parameters such as drag and lift forces. In these simulations the use of the already validated body-fitted formulation together with the experimental and wind tunnel results proved to be indispensable to the success of both the symbolic and embedded formulations. This was observed especially at points where the two formulation were in line with the experimental result. On the other hand, certain decisions that were made during the setting up of the simulations, for example the use of a coarse mesh caused some minor errors in the solution provided by the formulation. However, dispute these small errors the most important flow characteristics were captured by the formulations. This is evident by the pressure and velocity distributions that have been obtained in the simulations. Furthermore with respect to the use of the available computational cost, partitioning of the geometries proved to be vital in that the most important parts of the simulation were captured using less computational cost. All in all the formulation have shown that they have been

well implemented and if more accuracy is required a very fine mesh may be used.

.

The challenging Silsoe cube test led to the validation of both the symbolic and embedded approaches in dealing with complex simulations. This problem presented an effective platform to validate the methods especially in regions of flows separations. as we know this is where complex flow phenomena come into play and hence it is modeling of this parts is a vital part of the validation process. The results observed in this regions showed that the formulations produced almost the same results which shows that the problem encountered in the results did not come from the formulations rather than it may have been due to the way the entire model was setup. This supported by the fact that the results obtained were almost similar with the one obtained Gerhard [15] who hand originally derived the first inlet profile used in this work.

5.1 Future work

A number of interesting problems were encountered during the thesis, although we did not have time to solve them. The first issue was concerned with the embedded formulation where we needed to know how much error was been introduced into the solution through the use of the distance modification algorithm. This is important as it can allow us to effectively distinguish between the modeling errors and formulation based errors. Actually this can be tested by considering a case study such as the modeling a sphere where the analytical volume is known. Note that with respect to the distance modification process there are also other methods in the literature there such as the Ghost point method, but due to the complexity of dealing with the parallelization techniques currently been used in Kratos this method have not been implemented.

Secondly prediction of wind loads on bodies such as the silsoe cube is quiet challenging and as we did not have enough time to deal with the test, it would be wonderful to spend more time on this problem. A key suggestion is to try and use a periodic inlet profile at least in order to be able to capture the effects of factors such as the atmospheric effect. In response to the required mesh, the solution could be improved by the increasing of the mesh density in the vicinity of the cube. This, however, will lead to higher demands on the computational time.

Thirdly, as all the benchmark problems in this work were modeled in GiD it is vital that the formulations be tested on real external geometries coming from a computer aided design (CAD) software (i.e a model of a car). This present a chance of validating the two formulations on issues such as the amount of time and work that goes into the process of cleaning the geometry and also apart from this, we are aware that given an efficient mesh refinement algorithm, the embedded approach is very suitable for rapid prototyping of structures hence it will be great to see the efficiency of using the embedded method distance algorithm in predicting the geometry.

Finally theres is also another embedded formulation in Kratos multiphysics that would be interesting to validate, this is the slip boundary embedded method. This formulation can be tested on the various benchmark test from the literature.

Bibliography

[1] J. D. Anderson and J. Wendt. *Computational fluid dynamics*, volume 206. Springer, 1995.

[2] G. K. Batchelor. *An introduction to fluid dynamics*. Cambridge university press, 2000.

[3] K.-J. Bathe. *Finite element method*. Wiley Online Library, 2008.

[4] D. Baumgartner, J. Wolf, R. Rossi, R. Wüchner, and P. Dadvand. *Contribution to the fluid-structure interaction analysis of ultra-lightweight structures using an embedded approach*. 2015.

[5] R. Codina. A stabilized finite element method for generalized stationary incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 190(20-21):2681–2706, 2001.

[6] R. Codina and J. Baiges. Approximate imposition of boundary conditions in immersed boundary methods. *International Journal for Numerical Methods in Engineering*, 80(11):1379–1405, 2009.

[7] R. Codina, S. Badia, J. Baiges, and J. Principe. Variational multiscale methods in computational fluid dynamics. *Encyclopedia of Computational Mechanics Second Edition*, 2017.

[8] J. Cotela, E. Oñate, and R. Rossi. A comparison of different parallel techniques applied to the solution of the navier-stokes equations. 2011.

[9] J. Cotela, E. Oñate, and R. Rossi. Applications of turbulence modeling in civil engineering. *Monographs of the International Centre for Numerical Methods in Engineering (CIMNE)*, 2016.

[10] P. Dadvand. *A framework for developing finite element codes for multi-disciplinary applications*. Universitat Politècnica de Catalunya, 2007.

[11] P. Dadvand, R. Rossi, and E. Oñate. An object-oriented environment for developing finite element codes for multi-disciplinary applications. *Archives of computational methods in engineering*, 17(3): 253–297, 2010.

[12] J. Donea and A. Huerta. *Finite element methods for flow problems*. John Wiley & Sons, 2003.

[13] J. Donea, A. Huerta, J.-P. Ponthort, and A. Rodriguez-Ferran. Arbitrary lagrangian eulerian methods. *Encyclopedia of Computational Mechanics Second Edition*, 2004.

[14] T. Gelhard, G. Lube, M. A. Olshanskii, and J.-H. Starcke. Stabilized finite element schemes with lbb-stable elements for incompressible flows. *Journal of Computational and Applied Mathematics*, 177(2):243–267, 2005.

[15] S. Gerhard. Msc thesis: Evaluation of the finite element method for turbulent flows with the open source software kratos. 2012.

[16] E. Hachem, B. Rivaux, T. Kloczko, H. Digonnet, and T. Coupez. Stabilized finite element method for incompressible flows with high reynolds number. *Journal of computational physics*, 229(23): 8643–8665, 2010.

[17] D. R. Hillier, K. Ryan, and G. J. Sheard. Implementation of an immersed boundary method in spectral-element software. *iIB*, 1(2):2, 2009.

[18] M. W. Johnson. A novel cartesian cfd cut cell approach. *Computers & Fluids*, 79:105–119, 2013.

[19] R. Löhner, S. Appanaboyina, and J. R. Cebral. Comparison of body-fitted, embedded and immersed solutions of low reynolds-number 3-d incompressible flows. *International journal for numerical methods in fluids*, 57(1):13–30, 2008.

[20] V. Michalcová, S. Kuznětsov, and S. Pospíšil. Models of load on buldings from the effects of the flow field. *Transactions of the VŠB–Technical University of Ostrava, Civil Engineering Series*, 13 (2):91–97, 2013.

[21] M. A. Olshanskii, K. M. Terekhov, and Y. V. Vassilevski. An octree-based solver for the incompressible navier–stokes equations with enhanced stability and low dissipation. *Computers & Fluids*, 84:231–246, 2013.

[22] A. Quirinh. Msc thesis: Simulating wind fields over hcomplex terrain. technical university of munich. 2017.

[23] P. Richards, R. Hoxey, and L. Short. Wind pressures on a 6 m cube. *Journal of Wind Engineering and Industrial Aerodynamics*, 89(14-15):1553–1564, 2001.

[24] P. Richards, R. Hoxey, B. Connell, and D. Lander. Wind-tunnel modelling of the silsoe cube. *Journal of Wind Engineering and Industrial Aerodynamics*, 95(9-11):1384–1399, 2007.

[25] R. Rossi, J. Cotela, N. M. Lafontaine, P. Dadvand, and S. R. Idelsohn. Parallel adaptive mesh refinement for incompressible flow problems. *Computers & Fluids*, 80:342–355, 2013.

[26] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher. Benchmark computations of laminar flow around a cylinder. In *Flow simulation with high-performance computers II*, pages 547–566. Springer, 1996.

[27] J. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, and D. Mavriplis. Cfd vision 2030 study: a path to revolutionary computational aerosciences. 2014.

[28] G. Strang and G. J. Fix. *An analysis of the finite element method*, volume 212. Prentice-hall Englewood Cliffs, NJ, 1973.

[29] B. M. Sumer et al. *Hydrodynamics around cylindrical strucures*, volume 26. World scientific, 2006.

[30] O. C. Zienkiewicz and R. L. Taylor. *The finite element method for solid and structural mechanics*. Elsevier, 2005.