UNIVERSITY OF GENOVA

POLYTECHNIC SCHOOL

MASTER OF SCIENCE THESIS
IN
MECHANICAL ENGINEERING

CFD analysis for the investigation of an aeroelastic system for wind extraction

Supervisors:

Prof. Andrea Mazzino

Co-supervisor:

Ing. Stefano Olivieri

Candidate:

Alberto Rossi

March 2017

CFD analysis for the investigation of an aeroelastic system for wind energy extraction

Abstract

The development of Wireless Integrated Network Sensors (WINS) requires the installation of a large number of sensors around the world. These devices cannot be wired because they are positioned in places hard to reach. The solution to follow without batteries is the energy harvesting, which is a process that uses innovative devices to extract energy from the environment, supplying an endless power to the sensors. In this work we present a novel energy harvesting device able to capture the kinetic energy from a fluid in motion and transform it in electrical energy. This device, named FLEHAP (FLuttering Energy Harvesting for Autonomous Powering), is based on an aeroelastic effect named fluttering, in which a totally passive airfoil shows sustained motion depending on few parameters. The aim of the thesis is to study numerically the problem, through 3D simulations, using the open source code OpenFOAM, based on a finite volume approach. A campaign for different wind velocities has been made in order to characterize the device operation, comparing the numerical solutions with experimental and analytical results. Moreover, a particular focus is made on the case in which the fluid-structure interaction is maximized: by means of a Large Eddy Simulation approach, we study the correlation between the aerodynamic coefficients and the vortex dynamics, supplying a useful insight for future investigations.

Utilizzo di una strategia di tipo CFD per lo studio di un sistema aeroelastico per l'estrazione di energia del vento

Sommario

Lo sviluppo di reti di sensori integrati (WINS) richiede l'utilizzo di una grande quantità di sensori per il monitoraggio di un determinato ambiente. La loro posizione è normalmente difficile da raggiungere, dunque l'alimentazione dei sensori avviene normalmente tramite batterie. Per ovviare al problema della sostituzione di queste ultime, è conveniente utilizzare dispositivi capaci di estrarre energia dall'ambiente in cui i sensori sono inseriti. Questo processo è noto come *energy harvesting*. In questo lavoro ci occuperemo di un dispositvo innovativo, denominato FLEHAP (*FLuttering Energy Harvesting for Autonomous Power*), il quale sfrutta i principi dell'aeroelasticità per estrarre energia da una corrente fluida. Lo scopo della tesi sarà quello di studiare numericamente il problema, attraverso simulazioni 3D, utilizzando il codice *open source* OpenFOAM, il quale utilizza un approccio ai volumi finiti. Come prima analisi è stata condotta una campagna per diverse velocità del vento ai fini di caratterizzare il funzionamento del dispositvo, confrontando i risultati numerici con quelli sperimentali e con un modello analitico. Il fulcro del lavoro sarà comunque incentrato solo per una data velocità del flusso che massimizza l'interazione tra il fluido e la struttura. Attraverso una simulazione LES (*Large-Eddy Simulation*), studieremo infine la correlazione tra i coefficienti aerodinamici e la dinamica dei vortici, fornendo utili informazioni per studi successivi.

Ringraziamenti

Desidero ringraziare il Professor Mazzino per avermi concesso l'opportunità di lavorare su questa tesi che mi ha permesso di ampliare le mie conoscenze in materia. Un sentito grazie lo devo al mio correlatore Stefano che mi ha sempre seguito e consigliato durante la stesura di questo lavoro, mostrando una pazienza degna di nota; gli auguro inoltre di proseguire il suo dottorato al meglio. Un grazie a Gregorio che con la sua disponibilità e conoscenza mi ha permesso di capire meglio alcune sfaccettature del funzionamento del dispositivo studiato. Un sentito grazie alla mia famiglia che mi ha permesso di portare a termine gli studi supportandomi in ogni momento. Un grazie va a Camilla la quale mi è sempre stata vicina mostrandomi sempre affetto e amore. Un grazie a tutti coloro che ho incontrato in questi anni e che sono stati capaci di avermi fatto passare momenti felici.

Acknowledgements

I wish to thank Professor Mazzino for giving me the opportunity to work on this thesis which has allowed me to expand my knowledge. A heartfelt thanks I owe it to my co-supervisor Stefano who always followed me and recommended during the writing of this work, showing a patience worthy of note, also I wish him to pursue his Phd at best. Thanks to Gregorio that with his availability and knowledge has allowed me to better understand some aspects of the operation of the device studied. A big thanks to my family who allowed me to finish the studies supporting me at all times. Thanks go to Camilla who has always been close to ever showing me affection and love. Thanks to all those I have met over the years and have been capable of making me pass happy moments.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Energy harvesting

In recent times, the increase in global power generation needs has raised two major issues, namely the decreasing availability of fossil fuels for power generation and their impact on the global climate. These two issues are the main driving force behind the search for alternative methods to produce energy from renewable sources. *Energy harvesting* (EH), or energy scavenging, is a process that allows to recovery little quantity of energy from any environmental source around us (e.g. solar, wind, thermal, kinetic) using innovative and simple devices. The power is used to operate small devices which absorb power of the order of mW.

One of the goals of energy harvesting is to generate power for sensors or little devices that are difficult to connect to the electrical grid because of their remote positioning.

A typical application of energy harvesting are the *Wireless Integrated Network Sensors* (WINS) that monitor parameters such as the wind speed, the height of waves, the pollutant concentration in a city and so on. All the sensors transmit the information to a principal node named sink, which in turn transmits them to a central unit [35]. This evolution in the electronic world leverages on the concept of omnipresent connectivity among different sensors, that is expected to billions in 2020. These sensors cannot be wired, so they must work with their own energy source, usually a battery. Anyway, for a large number of application domains, the battery reveals critical issues in terms of installation and replacement costs, circuit complexity and long-time reliability. This is especially true when the sensor networks are intended to be used on very large areas or placed in difficult places to reach (e.g. buried or in harsh or hazardous locations).

In the EH perspective, the possibility for a certain network to stay reliable and economically viable on the long term (10 years or more) will depend on the node's capability to recover energy from the node's environment (en-

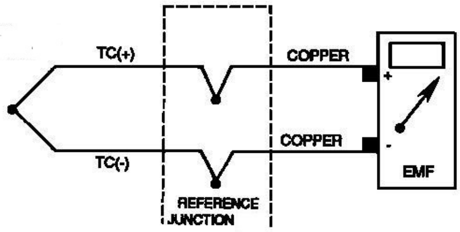


Figure 1.1: Example of thermocouple's principle of operation (taken by [33]).

ergy harvesting), either to prevent the battery from discharging or, much better, to get rid of the battery itself.

It is therefore necessary to design innovative energy harvesters, that allow the recovery of energy from the environment in which they are installed.

1.2 Types of energy harvesters

In the world of energy harvesting we can encounter all sorts of devices that recover power from many different sources of energy. To make the treatment clearer, we will refer to a classification according to the principle of operation. We will briefly review three possible ways, although there are many other ones:

- thermoelectrical;
- electromagnetic;
- mechanical.

1.2.1 Thermoelectrical energy harvesters

These devices, better known as thermocouples, work using the Seebeck's effect, which consist in the creation of a electric potential difference in two different metals when their junction points experience a temperature gradient (Fig. 1.1). The main relation is:

$$\varDelta U = \alpha \varDelta T \tag{1.1}$$

where α is the Seebeck's coefficient dependent from the material and of the order of μV/°C. Hence, for an efficient system it is better to put the highest possible number of thermocouples in series [33].

1.2.2 Electromagnetic energy harvesters

These devices are able to convert electromagnetic waves in electrical energy. The photovoltaic cells are one of the countless examples, where we obtain

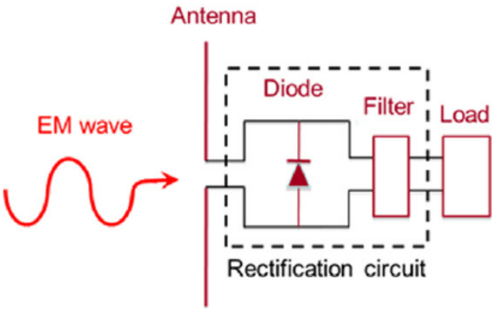


Figure 1.2: Example of Rectenna's principle of operation (taken by [25]).

voltage differences in a material upon exposure to light. Another example is represented by the so called *Rectenna* (rectifying antenna), recently proposed in literature [25]. A rectenna (Fig. 1.2) is a special type of antenna that is used to convert electromagnetic energy into direct current (DC). It can be used in wireless power transmission systems that transmit power by radio waves. A simple rectenna element consists of a dipole antenna with an RF diode connected across the dipole elements. The diode rectifies the AC current induced in the antenna by the microwaves to produce DC power, supplying a load connected across the diode. Schottky diodes are typically used because they have the lowest voltage drop and highest speed and therefore have the lowest power losses due to conduction and switching. Large rectennas consist of an array of many dipole elements of this kind.

1.2.3 Mechanical energy harvesters

These devices are able to convert directly the mechanical energy in electric energy mainly through piezoelectric effect and/or electromagnetic coupling (EMC). Only few materials present piezoelectric effect. It consists in the creation of an electrical potential difference as a consequence of a mechanical deformation. The EMC is based on the well known Faraday-Neumann's law from which it is possible to extract energy [17]:

$$\Delta U = -\frac{d\Phi}{dt} \tag{1.2}$$

indeed if the vibrations are such as to induce oscillation, a magnet fixed to the device can produce a variation of electromagnetic flux (Φ) on a fixed coil.

1.3 Aeroelastic energy harvesters

These devices belong to mechanical energy harvesters. Indeed, in the presence of a fluid current, self-sustained vibrations of the device may be induced

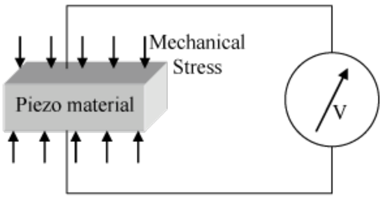


Figure 1.3: Diagram of operation of a piezoelectric element (taken by [17]).

from which it is possible to obtain electrical power [18]. Several aerolastic instabilities can be exploited such as vortex induced vibrations and/or galloping for bluff bodies, or flutter for airfoils and slender structures. In this work we will focus on the so called flapping energy harvesters, where moving wings are capable of harvesting energy from vortices, free-surface waves and uniform streams. The first case involves the foil's capacity of manipulating the ambient vorticity field as well as controlling its own vorticity shedding. At least two distinctive interaction modes have been identified: the constructive mode and the destructive mode. In the constructive mode, the vortices created by the foil and the incoming vortex street are in the same phase and reinforce each other. It leads to stronger reverse Kármán vortex streets and increased thrust. The destructive mode, on the other hand, is characterized by a phase difference of approximately 180° between the foil-generated vortices and the incoming ones. As a result the combined wake is weakened, while the propulsion efficiency is increased. Similar vorticity control mechanisms have been discovered in interactions between leading-edge vortices (LEV) and trailing-edge vortices in a flapping foil. The basic idea to use a foil to harvest energy from a uniform flow was proposed over 2 decades ago [20]. Indeed, the capacity of the foil to absorb flow energy is clearly demonstrated in flow-induced vibrations such as flutter.

These devices are inspired by nature, indeed aquatic animals, as well as insects and birds, which exploit periodic motions with fins or wings to achieve highly effective propelling and maneuvering, instead of rotational propellers. For example, tunas, dolphins and sharks exhibit excellent hydrodynamic performance with high cruising speed, high efficiency and low noise through the flapping motion of their caudal fins. Moreover, through these oscillatory motions it is possible to extract energy from the incoming vortices or unsteady flows. It has been numerically demonstrated that the caudal fin of a fish can absorb energy from vortices shed from dorsal/ventral fins to achieve higher propulsion efficiency [37].

Unlike conventional turbines, there are several prominent features of such bio-inspired energy converters:

- they are environmentally friendly in terms of noise generation due to their relatively low tip speed, thus reducing impact on the navigation of aquatic animals;

- without the centrifugal stress associated with rotating blades, the oscillatory devices are structurally robust;

- oscillating hydrofoil systems sweep a rectangular cross section of flow. The swept area for a single device can thus be wide and shallow, allowing large systems to be installed in shallow water. Subsequently, multi-megawatt devices can be envisaged for a wider range of tidal stream resource areas [8].

With the rapid development of such devices, the understanding of their underlying physics, including the fluid dynamics, the fluid-structure interactions, and the coupled dynamics of the non-linear system, is required in order to improve the existing devices efficiency and pave the way for the development of new systems that will be commercially profitable. In the past few years, the research on the dynamics of oscillating energy extraction devices based on flapping foils has attracted gradually more attention.

As shown by Ashraf *et al.* [2] by means of a numerical study, the combined oscillation occurs in such a manner that there is a 90 deg phase angle Φ between the pitch and the plunge (vertical) motion, then the aerodynamic lift is always in the same direction as the motion of the airfoil, as shown in Fig. 1.4a. In this case, the work is done by the the fluid on the airfoil throughout the complete cycle. In other words, the airfoil is extracting energy out of the air flow. On the other hand, if the phasing between pitch and plunge is zero, as shown in Fig. 1.4b, then during parts of the cycle the aerodynamic lift opposes the motion and no net work is done on the airfoil. The type of flutter shown in Fig. 1.4a can easily occur on airplane wings, for certain values of the bending and torsional stiffnesses of the wing. Clearly, this phenomenon can also be used for power generation.

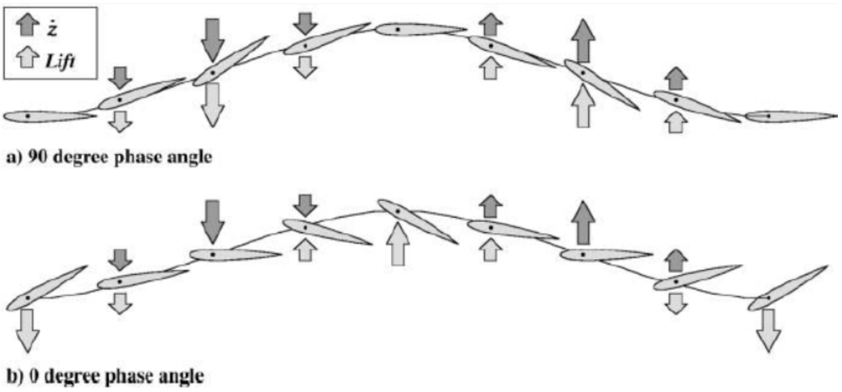


Figure 1.4: Trend of Lift force and body acceleration in a period varying with phase angle (taken by [2]).

Chapter 2

Object and aim of the work

2.1 Fluttering Energy Harvesting for Autonomous Power

In this work we will deal with the FLEHAP device (Fluttering Energy Harvesting for Autonomous Power) [3, 4, 22, 23] originated from a joint project on energy harvesting from fluid-structure interaction involving two departments of our University (DICCA and DIFI). The FLEHAP project concerns a novel energy harvester under active development based on an elastically bounded wing, exploiting the coupled-mode flutter aeroelastic instability. This device is developed to work for low wind speed (2-6 m/s) and for low Reynolds regime $O(10^3)$.

The harvester is sketched in Fig. 2.1. The system is composed by a wing W connected to a support S via 2 elastomers E. The wing can freely rotate around a rigid (pivot) axis A. The elastomers are fixed at the ends of the axis, they are parallel to the wind direction and movement is mainly confined along y direction.

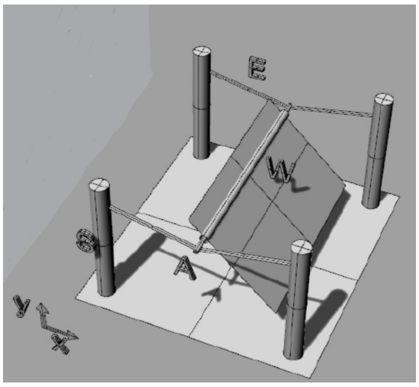


Figure 2.1: Sketch of energy harvester (taken by [3]).

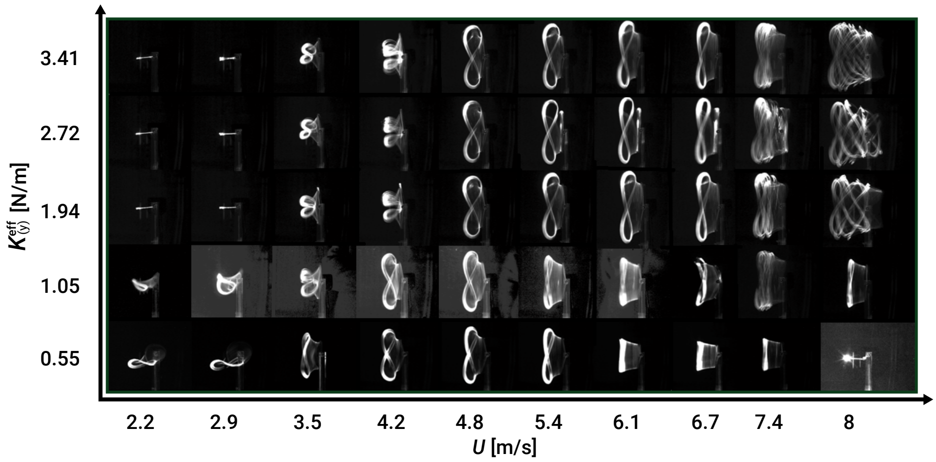


Figure 2.2: Wing motion regimes as a function of flow velocity and elastomer equivalent stiffness. The images are acquired by a digital camera with long time exposure (see Sec. 2.2.1) (taken by [22])

When a fluid in motion impinges on this elastic structure, an amount of kinetic energy is transferred to the system, inducing large amplitude oscillations if few mechanical parameters are correctly set (Fig. 2.2). In this situation, the system exhibits limit cycle oscillations (LCOs) whose characteristics depend on many parameters such as the wind velocity, geometrical and mechanical properties.

From the resulting self-sustained pitch and plunge movement, it is possible to extract energy using electromagnetic coupling (EMC) and/or elastomeric capacitors replacing the standard elastomers. The principle beneath the EMC is the Faraday-Neumann's law, e.g. by placing two magnetic elements at the ends of the pivot axis, and two fixed coils on pylons. When the foil flutters, the coils experience a variation of induced magnetic field, so it is possible to extract electric power proportional to the variation frequency of the magnetic field. At the present stage, the order of the generated power is around mW.

The wing dynamics is given by the combination of gravity, aerodynamics and elastic forces that influence the motion. The fluttering phenomenon appears when the body can not contrast bend and torque by a fluid flow, varying the wing angle of attack, thereby there is a variation of aerodynamic force. So if the damping is not sufficiently high the body starts to oscillate increasing the amplitude of motion. The fluttering condition depends on: the centre of mass position, the mass of the wing, the wing dimensions, elastomers stiffness and the value of resistive load.

Although the elastomeric elements do not follow the linear Hooke's law, by measuring the natural frequency of the structure (e.g. imposing a perturbation in quiescent fluid) it is possible to derive an equivalent stiffness along the vertical direction, i.e.: $\omega_{\mathrm{n}} = \sqrt{K_{(y)}^{\mathrm{eff}}/m}$, where m is system translational mass and $K_{(y)}^{\mathrm{eff}}$ is the effective elastomers stiffness along the vertical direction. It is also possible to give an approximation of this quantity as [22]:

$$K_{(y)}^{\mathrm{eff}} = 12GA_0\frac{\epsilon}{l_0^2}$$

where G is the shear modulus, A_0 is the initial elastomer cross section, ϵ is the elastomer pre-stretching and l_0 is the initial elastomer length.

Recalling Fig. 2.2, we show the trajectories of the pivot point (PP) and trailing edge (TE), in a configuration without energy extraction, in function of U and $K_{(y)}^{\mathrm{eff}}$.

Some observations can be made:

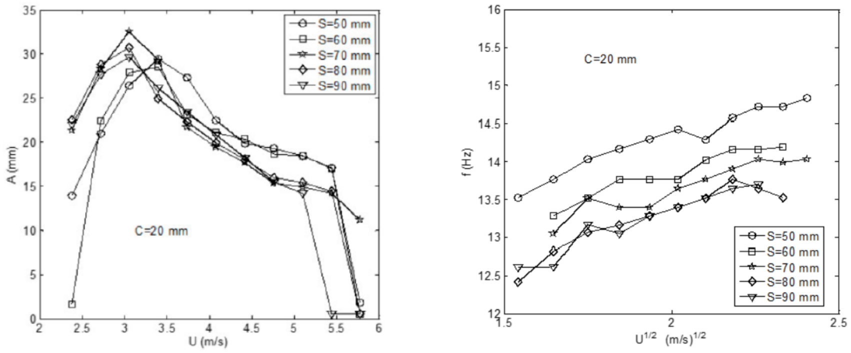
- The aeroelastic instability occurs only for flow velocities sufficiently high ($U > 2.9$ m/s) .
- Increasing U we reach pre-transition phase where the PP amplitude is greater than TE amplitude for $K_{(y)}^{\mathrm{eff}}$ ¿ 0.55 N/m.
- At speed of 4.8 and 5.4 m/s we have a substantial independence from the stiffness, indeed the trajectories do not vary changing the $K_{(y)}^{\mathrm{eff}}$ value.
- For higher speed we are in post-transition phase where the TE amplitude is greater than the PP amplitude. In this case decreasing the stiffness the wing stops to vibrate, aligning with the flow returning to a stable condition. On the contrary increasing the stiffness we have chaotic motion.

As an example, in Fig. 2.3 we report the trend of the flapping amplitude and frequency as a function of the flow velocity, for different wing spans [3]. For this configuration the maximum oscillation is at 3 m/s, while the angle it is around 50° at low velocity, reaches a maximum around 90° at velocity of 3.5 m/s, then decreases at higher speed. It has been observed that for $U >$ 6m/s the movement stops and the wing assumes a horizontal position.

In order to maximize the energy extracted from an electromagnetic coupling, the oscillation frequency needs to be maximized.

To calculate the power generated by the device we refer to $P_{plunge} = Lv_{pp}$ (mechanical power extracted by the device) and P_{betz} (maximum power extractable from the fluid flow). The first term is the product between the lift (L) force and vertical pivot point velocity(v_{pp}):

$$P_{plunge} = \frac{L}{v_{pp}} \qquad (2.1)$$



(a) (b)

Figure 2.3: (a) Amplitude and (b) frequency of motion as a function of the wind speed (taken by [3]).

while P_{betz} is equal to:

$$P_{Betz} = \frac{1}{2}\rho_f A U^3 \frac{d}{c} \tag{2.2}$$

it is possible to define Betz efficiency, that means how much energy it is possible to extract from the harvester compared to energy flow:

$$\eta_B = \frac{\bar{P}_{plunge}}{P_{Betz}} \tag{2.3}$$

where $\bar{P}_{plunge}$ is the power average in the time. It is also possible define other efficiency terms as the electro-mechanic efficiency (η_{EM}):

$$\eta_{EM} = \frac{\bar{P}}{P_{Betz}} \tag{2.4}$$

where $\bar{P}$ is the average electrical power absorbed by the load:

$$\bar{P} = \frac{1}{T}\int_t^{t+T} VIdt \tag{2.5}$$

where T is the period of a cycle, V is the voltage and I is the current. We can therefore the global efficiency (η_g) as:

$$\eta_g = \eta_B \eta_{EM} = \frac{\bar{P}}{P_{Betz}} \tag{2.6}$$

we can see from Fig. 2.4 that for each wind speed there is a best value of resistance load to adopt in order to maximize the power extracted. For U ¿ 4.5, the electrical power extracted assumes the highest values, but works for a limited range of resistance load. While for lower wind speed the power extracted is less but the device can work in a wider range of resistance load.

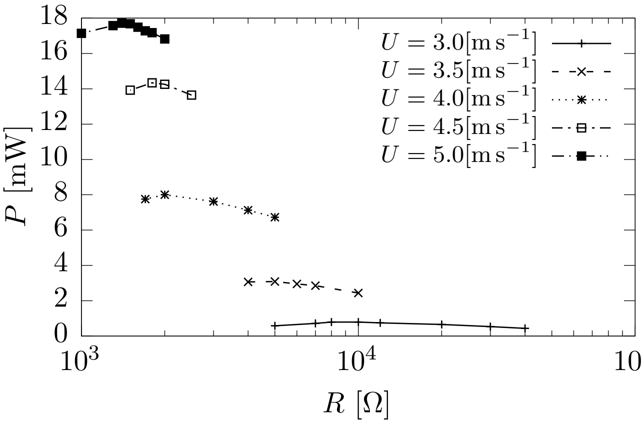


Figure 2.4: Power trend in function of resistive load. The wing has chord equal to 35 mm and span equal to 85mm (taken by [22]).

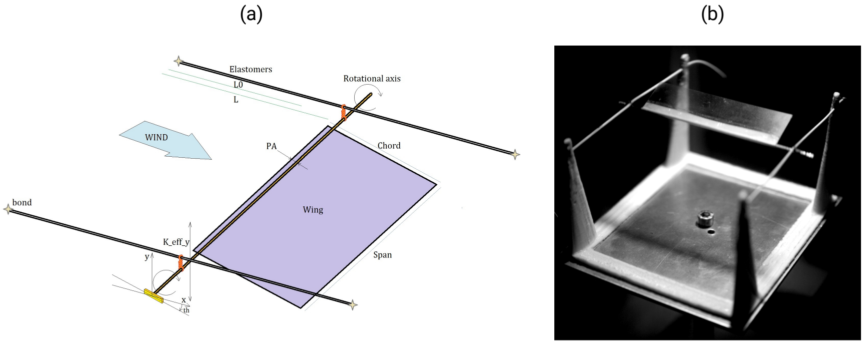


Figure 2.5: Sketch of device studied in this work (taken by [22]).

2.2 Object of the work

The device studied in this work is shown in the Fig. 2.5. The wing is composed by polymeric material and it is anchored by four elastomers through a cylindrical rod, perpendicular to the wind direction. The weight of the wing is of 0.845 g, the dimensions are 20 mm × 70 mm × 0.1 mm. The elastomers force exerted by elastomers follow this relation [27]:

$$\mathbf{F}_{el} = -GA_0 \left[\frac{|\mathbf{L}|}{L_0} - \left(\frac{L_0}{|\mathbf{L}|} \right)^2 \right] \frac{\mathbf{L}}{|\mathbf{L}|} \tag{2.7}$$

where G, is the shear modulus (in this work equal to 0.534 MPa), A_0 is the cross section at rest, $|\mathbf{L}|$ is the elastomer present length, L_0 is the initial length and $\mathbf{L}/|\mathbf{L}|$ is the dimensionless versor. The anchor points are at 0.05 mm from leading edge, this is an important parameter that influences the auto-excited flapping motion of the wing [23].

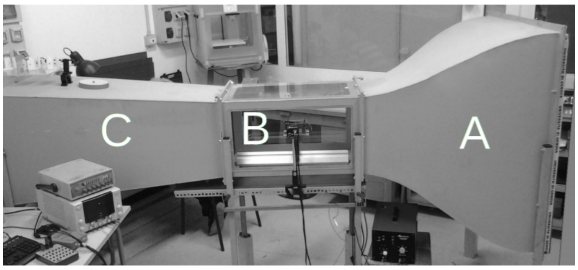


Figure 2.6: The employed wind tunnel where the real device is tested; (A) nozzle, (B) test chamber, (C) diffuser (taken by [22]).

2.2.1 Wind tunnel configuration

The wind tunnel used for the experimental measurements is a subsonic aspirating open circuit tunnel built at the Department of Physics of the University of Genoa (Italy). With a total length of 3 m, its shape is designed to obtain a stable and controlled laminar flow inside the test chamber, the 40 cm x 40 cm 70 cm middle part of tunnel (Fig. 2.6). By exploiting the Venturi effect, the air is aspirated through a honeycomb grid into the nozzle (A), conveyed to the test chamber (B) and through the diffuser (C), at the end of which a three-phase motor of the aspirating fan is collocated.

The tunnel calibration was performed using a Cobra probe sampling at 1250 Hz. A free-stream turbulence level of around 0.6% is ensured thanks to the use of the honeycomb grid, the appropriate curvature of the nozzle and the slope of the diffuser. By varying the fan rotational speed, the operational flow velocity in the test chamber ranges from 1.7 to 10.5 m/s with an absolute error of 0.15 m/s. Moreover, visualizations of the free-stream flow and of the flow-structure interaction have been possible using a home-made smoke generator apparatus (Fig. 2.7). The use of a high definition camera (500 fps) coupled to an appropriate illumination and video processing system allows to obtain a precise acquisition of the wing motion: a light scatterer placed on the rotational axis of the wing is illuminated by a line laser beam and a numerical elaboration of the movie returns the main quantities in time such as the pivot point position and the pitching angle.

2.3 Phenomenological model

It is also possible to give an analytical description of the system representing a complementary way to study the problem, based on quasi-steady phenomenological model, originally developed for investigating falling plates [1]. The model consists of a system of first order ODEs written in the frame of reference (x', y') co-rotating with the plate:

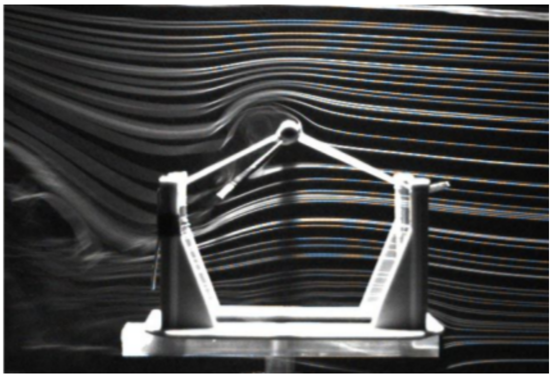


Figure 2.7: Flow visualizations using smoke generation (taken by [22]).

$$(m + m_{11})\dot{v}_{x'} = (m + m_{22})\dot{\theta} v_{y'} + F^{\mathrm{el}}_{x'} - \rho_{\mathrm{f}} \Gamma \tilde{v}_{y'} - F^{\nu}_{x'} \tag{2.8}$$

$$(m + m_{22})\dot{v}_{y'} = -(m + m_{11})\dot{\theta} v_{x'} + F^{\mathrm{el}}_{y'} + \rho_{\mathrm{f}} \Gamma \tilde{v}_{x'} - F^{\nu}_{y'} \tag{2.9}$$

$$(I_G + I_{\mathrm{a}})\dot{\omega} = (m_{11} - m_{22}) v_{x'} v_{y'} - crF^{\mathrm{el}}_{y'} - l_{\tau} \rho_{\mathrm{f}} \Gamma \tilde{v}_{x'} - \tau^{\nu} \tag{2.10}$$

$$\dot{x}'_G = v_{x'} + \dot{\theta} y'_G \tag{2.11}$$

$$\dot{y}'_G = v_{y'} - \dot{\theta} x'_G \tag{2.12}$$

$$\dot{\theta} = \omega \tag{2.13}$$

where $v_{x'}$ and $v_{y'}$ denote the center of mass velocities, $\tilde{v}_{x'} = (v_{x'} - U\cos\theta)$ and $\tilde{v}_{y'} = (v_{y'} + U\sin\theta)$ are those relative to the unperturbed flow, m is the plate mass, I_G is the moment of inertia with respect to the center of mass, r represents the distance between the center of mass and the pivot point normalized with the chord, x'_G and y'_G are the coordinates of the plate center of mass, θ is the pitching angle, $\mathbf{F}^{\mathrm{el}}$ is the elastic force and $l_{\tau} = c/4\cos\theta$ is the moment arm of the circulatory force. The added mass coefficients m_{11}, m_{22} and I_{a} are expressed for the plate of rectangular cross-section following [15]: $m_{11} = (3\pi/8)\rho_{\mathrm{f}}\delta^2$, $m_{22} = (3\pi/8)\rho_{\mathrm{f}}c^2$, $I_{\mathrm{a}} = (5\pi/256)\rho_{\mathrm{f}}(c^2 - \delta^2)^2$.

The circulation Γ, the viscous force $\mathbf{F}^{\nu}$ and the dissipative fluid torque τ^{ν} are expressed in a semi-empirical manner using several free parameters [1]:

$$\Gamma = -C_T c \frac{\tilde{v}_{x'}\tilde{v}_{y'}}{\sqrt{\tilde{v}^2_{x'} + \tilde{v}^2_{y'}}} + \frac{1}{2} C_R c^2 \dot{\theta} \tag{2.14}$$

$$\mathbf{F}^{\nu} = \frac{1}{2}\rho_{\mathrm{f}} c \left[C_D(0)\,\tilde{v}_{x'}^2 + C_D(\pi/2)\,\tilde{v}_{y'}^2\right] \frac{\sqrt{\tilde{v}_{x'}^2 + \tilde{v}_{y'}^2}}{\tilde{v}_{x'}^2 + \tilde{v}_{y'}^2} \left(\tilde{v}_{x'}, \tilde{v}_{y'}\right) \tag{2.15}$$

$$\tau^{\nu} = C_{\tau}\rho_{\mathrm{f}}\dot{\theta}\left(\frac{|\tilde{v}_{y'}|c^3}{24} + \frac{|\dot{\theta}|c^4}{64}\right) \tag{2.16}$$

where C_T and C_R are the translational and rotational lift coefficients, respectively, $C_D(0)$ and $C_D(\pi/2)$ are the drag coefficients while C_τ is the dissipative torque coefficient.

On the numerical side, the evolution of equations (2.8)–(2.13) is performed using a third order Adams-Bashfort integration:

$$\mathbf{X}^{n+1} = \mathbf{X}^n + c_{\mathrm{f1}}\,\mathbf{G}^n + c_{\mathrm{f2}}\,\mathbf{G}^{n-1} + c_{\mathrm{f3}}\,\mathbf{G}^{n-2} \tag{2.17}$$

where $\mathbf{X}$ is a vector containing the six unknown quantities and $\mathbf{G}$ collects the right-hand-side of the equations; the superscript denotes that the quantity is evaluated at a certain discrete time $t_n = n\Delta t$, while expressions of the coefficients are: $c_{\mathrm{f1}} = \frac{23}{12}\Delta t$, $c_{\mathrm{f2}} = -\frac{4}{3}\Delta t$ and $c_{\mathrm{f3}} = \frac{5}{12}\Delta t$.

2.4 Scope of the work

The purpose of the present work is to investigate the problem numerically, using the open source code OpenFOAM [10], written in C++. OpenFOAM is a general purpose code created to resolve an extended variety of problems and it is suitable for studying fluid-structure interactions problems.

We will set up 3D simulations corresponding to the configuration specified at the beginning of Sec. 2.2, aiming at characterizing the flapping dynamics within the design wind velocity range. Among the main results, we will monitor the frequency, the vertical amplitude of pivot point motion and the phase between the pitch and plunge motion degrees of freedom. Furthermore, we will compare the numerical findings with those from experimental wind tunnel investigation and the phenomenological model prediction.

Chapter 3

Fundamentals of fluid-structure interactions and turbulence modelling

In the introduction has not been deepened which are physical working principles of energy harvester, limiting ourselves to explain its the general operation. In this section will be shown physical principles that allow to get sustained motion, as the fluttering and dynamic stall phenomenon. We will refer to dynamic stall instead of static stall because the device deals with unsteady condition and its incidence varies rapidly.

3.1 Aeroelasticity

Flutter is a dangerous phenomenon encountered in flexible structures subjected to aerodynamic forces. This includes aircraft, buildings, telegraph wires, stop signs, and bridges [6]. Flutter occurs as a result of interactions between aerodynamics, stiffness, and inertial forces on a structure. In an aircraft, as the speed of the wind increases, there may be a point at which the structural damping is insufficient to damp out the motions which are increasing due to aerodynamic energy being added to the structure. These vibrations can cause structural failure and therefore considering flutter characteristics is an essential part of designing an aircraft. The basic type of flutter of aircraft wing is described here. Flutter may be initiated by a rotation of the airfoil (see t=0 in Fig. 3.1). As the increased force causes the airfoil to rise, the torsional stiffness of the structure returns the airfoil to zero rotation (t=T/4). The bending stiffness of the structure tries to return the airfoil to the neutral position, but now the airfoil rotates in a nose-down position (t=T/2). Again the increased force causes the airfoil to plunge and the torsional stiffness returns the airfoil to zero rotation (t=3T/4). The cycle is completed when the airfoil returns to the neutral position with a

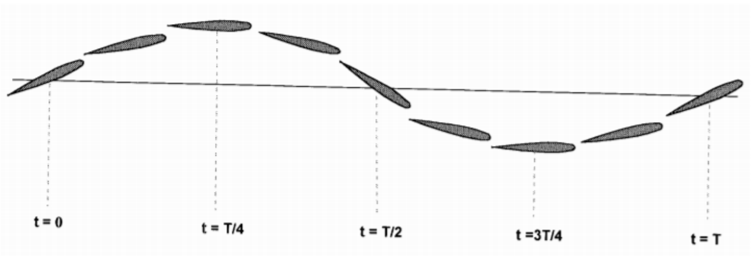


Figure 3.1: Rotation and Plunge Motion for an Airfoil Exhibiting Flutter (taken by [6]).

nose-up rotation. Notice that the maximum rotation leads the maximum rise or plunge by 90 degrees (T/4). As time increases, the plunge motion tends to damp out, but the rotation motion diverges. If the motion is allowed to continue, the forces due to the rotation will cause the rupture of the structure.

Anyway, in energy harvesting field fluttering can be viewed as a positive phenomenon as it may generated self sustained body fluctuations.

The flutter is caused by the coalescence of two structural modes pitch and plunge (or wing-bending) motion. We report the example of a wing which has two basic degrees of freedom, where pitch mode is rotational and the bending mode is a vertical up and down motion. As the airfoil flies at increasing speed, the frequencies of these modes coalesce or come together to create one mode at the flutter frequency and flutter condition. This is the flutter resonance [26].

Now we will have a deeper look showing the interaction fluid-solid equation. In this kind of problem the fluid and the solid equations are coupled through the boundary conditions. For the incompressible fluids the equations are:

$$\begin{cases} \nabla \cdot \mathbf{U} = 0 \\ \rho \dfrac{D\mathbf{U}}{Dt} + \rho g \mathbf{e}_z + \nabla p + \mu \Delta \mathbf{U} = 0 \end{cases} \tag{3.1}$$

with its relative boundary conditions, while for the solid we will refer to the single mode approximation:

$$\xi(\mathbf{x}, t) = q(t)\phi(\mathbf{x}) \tag{3.2}$$

where $\xi(\mathbf{x}, t)$ is the displacement function for the solid, $q(t)$ is the modal displacement that is the solution of an ordinary differential equation and ϕ is the modal shape, normally known, that depends by model type. In many cases $q(t)$ satisfies oscillator equation:

$$m\frac{d^2q}{dt} + kq = f \tag{3.3}$$

where m is the modal mass, k is the modal stiffness and f is the modal force, all this parameters are known.

The Navier's equation and the modal equations are coupled by the continuity equation at the interface: the kinematic condition (Eq.(3.4)) and the dynamic condition (Eq. (3.5)). They represent there is no exchange of mass at the interface and the force exerted by the fluid is equal and opposed by the solid force.

$$\mathbf{U} = \frac{d\xi}{dt} \tag{3.4}$$

$$\int_{\text{Interface}} \{[-pI + \mu(\nabla\mathbf{U} + \nabla^t\mathbf{U}) \cdot \mathbf{n}]\}\phi \quad dS = f \tag{3.5}$$

so Eq.(3.1)-(3.5) and the fluid and the solid boundary conditions, are the equations used to model the flutter system. The coupling problem can be simplified using a dimensionless form of the previous equations, depending on the reduced velocity number value: $U_R = U_{flow}/U_{solid}$.

$$U_R = \frac{U_{flow}}{U_{solid}} = \frac{U_0}{c} \tag{3.6}$$

where U_0 is the undisturbed flow velocity, and c is the speed wave propagation in the solid.

3.1.1 Flutter involving fast flow (high reduced velocity)

Here we show the theory for the cases that involves $U_R >> 1$. A possible example for viewing flutter instability is an airfoil immersed in a wind current at a specified speed flow (Fig. 3.2) with only just torsion freedom degree. The equation for this case is:

$$\ddot{\theta} + \left[1 - C_Y \frac{x}{L}\left(\frac{\partial C_l}{\partial \theta}\right)_0\right]\theta = 0 \tag{3.7}$$

the total stiffness vanished at a critical Cauchy number (C_Y) that is simply to compute, and so it is possible to obtain the critical velocity at which instability occurs:

$$C_Y^{Critical} = \frac{L}{2\pi x} \qquad \Rightarrow \qquad C_Y = \frac{\rho U^2 L^2}{2C} \qquad \Rightarrow \qquad U^{critical} = \sqrt{\frac{C}{\rho\pi L x}}$$

remember that C is rotation stiffness. For $U < U^{Critical}$ the airfoil just only oscillates without motion divergence, for $U > U^{Critical}$ the amplitude of

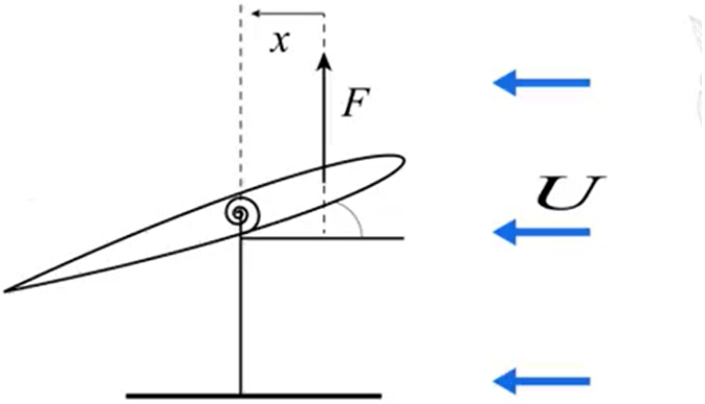


Figure 3.2: Airfoil invested by a wind current with velocity U, that exerts a lift force spaced x from centre of rotation. The system is bounded with a rotational spring which stiffness is C.

motion increases exponentially with no oscillation. For avoiding this problem it is necessary to adopt stiffer materials and move upward the point of axis rotation.

We can develop a more sophisticated model, named the two-modes approximation. The displacement ξ is a combination of two contributes, like as a two type of translation, or a translation and a rotation:

$$\xi = Dq_1(t)\phi_1(x) + Dq_2(t)\phi_2(x) \tag{3.8}$$

$$m_1\ddot{q_1} + k_1 q_1 = f^1_{FS} \tag{3.9}$$

$$m_2\ddot{q_2} + k_1 q_2 = f^2_{FS} \tag{3.10}$$

the fluid loading f_{FS} needs to be projected on the two modes q_1 and q_2 and can be expanded as in terms of displacement number, that is assumed to be small:

$$f^i_{FS} = C_Y F^0_i + DC_Y \left(\frac{\partial F_i}{\partial q_1}\right)_0 + DC_Y \left(\frac{\partial F_i}{\partial q_2}\right)_0 + ... \qquad i = 1;2 \tag{3.11}$$

naming

$$K_{ij} = \left(\frac{F_i}{\partial q_j}\right)_0$$

the fluid stiffness, that depends by flow velocity. We have now a set of two equations, coupled by the flow-induced stiffness term:

$$\begin{cases} m_1\ddot{q_1} + k_1 q_1 & = C_Y K_{11} q_1 + C_Y K_{12} q_2 \\ m_2\ddot{q_2} + k_2 q_2 & = C_Y K_{21} q_1 + C_Y K_{22} q_2 \end{cases} \tag{3.12}$$

the terms on the right vary with the velocity through the Cauchy number; the off diagonal terms induces coupling of two modes, while diagonal terms is referred to a single mode. For a certain U the two frequencies modes are coincident, the critical flow velocity depends by flow induced stiffness. We can consider two cases, the first when K_{12} and K_{21} are equal and the second when they are opposite. In both cases the non diagonal terms are littler than diagonal terms ($\epsilon << 1$):

$$\begin{cases} \ddot{q_1} + q_1 & = \epsilon q_2 \\ \ddot{q_2} + q_2 & = \epsilon q_1 \end{cases} \tag{3.13}$$

$$\begin{cases} \ddot{q_1} + q_1 & = \epsilon q_2 \\ \ddot{q_2} + q_2 & = -\epsilon q_1 \end{cases} \tag{3.14}$$

in case of the coupling is symmetric:

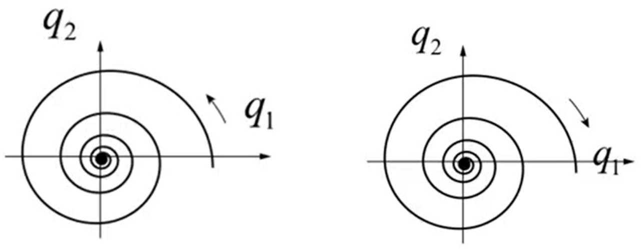
$$\omega \approx 1 \pm \frac{\epsilon}{2} \tag{3.15}$$

$$\begin{aligned} \omega_A &= 1 + \frac{\epsilon}{2} \\ \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}_A &= \begin{pmatrix} 1 \\ -1 \end{pmatrix} \end{aligned}$$

$$\begin{aligned} \omega_B &= 1 - \frac{\epsilon}{2} \\ \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}_B &= \begin{pmatrix} 1 \\ -1 \end{pmatrix} \end{aligned}$$

their motion is formed by the combination of the eigenvectors, while the eigenvalues are the frequencies of two modes, neither of two are unstable motion. The antisymmetric coupling has a big difference with the previous case, indeed the frequency has a illusionary part, the solution is the following:

$$\begin{aligned} \omega_A &= 1 + i\frac{\epsilon}{2} \\ \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}_A &= \begin{pmatrix} 1 \\ -i \end{pmatrix} \end{aligned}$$



(a) Stable mode. (b) Unstable modde.

Figure 3.3: Plot of q_1 and q_2 trend depending on mode coupling (taken by [30]).

$$\begin{aligned} \omega_B &= 1 - i\frac{\epsilon}{2} \\ \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}_B &= \begin{pmatrix} 1 \\ i \end{pmatrix} \end{aligned}$$

these two modes have complex frequencies and complex eigenvectors; in particular motion at frequency ω_A gives to life to a damped mode (Eq. (3.16)), while motion at frequency ω_B to an unstable mode (3.17), the effect are visible in Fig. 3.3.

$$\begin{pmatrix} q_1 \\ q_2 \end{pmatrix}_A = \begin{pmatrix} 1 \\ -i \end{pmatrix} = \begin{pmatrix} cos(t) \\ sin(t) \end{pmatrix} e^{-\epsilon t/2} \tag{3.16}$$

$$\begin{pmatrix} q_1 \\ q_2 \end{pmatrix}_B = \begin{pmatrix} 1 \\ -i \end{pmatrix} = \begin{pmatrix} cos(t) \\ -sin(t) \end{pmatrix} e^{\epsilon t/2} \tag{3.17}$$

unlike the previous example of one degree of freedom, this time the instability provides an exponential growth and at the same time an oscillation. To sum up, when the coupling is symmetric we have two neutral modes, either, when there is an antisymmetric coupling, we have a damped mode and unstable mode. In Fig. 3.4 we can see the most important theory result: the dynamic instability, also called mode-coupled flutter, occurs only when the frequency are equal and the coupling is non symmetric.

Here we report a final analysis of a plunge and torsion movement for an airfoil. Resolving the set of Eq.(3.12) we can compute and plot the real and imaginary part of ω, as we can see in Fig. 3.5. Until the Cauchy number is minor of 0.08, the instability does not occur, after this threshold, when the two real frequencies merges, the imaginary part is not zero any more and we can have a stable mode or flutter, if $W_{FS} = \int_{\tau} F\dot{y}dt > 0$.

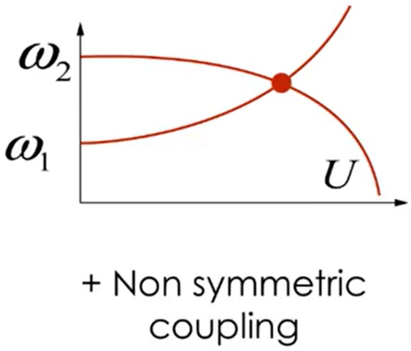


Figure 3.4: Flutter occurs when the two frequencies are merged and at the same time there is a non symmetric coupling (taken by [30]).

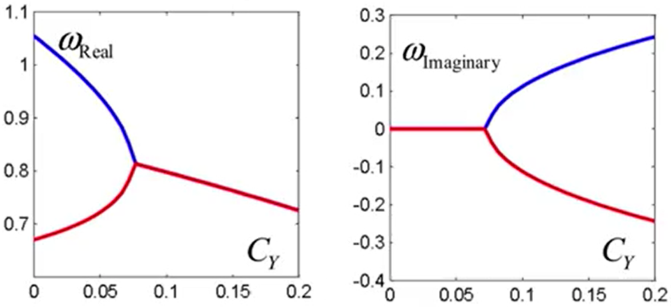


Figure 3.5: Plot of ω trends vary with Cauchy number, in blue we represent stable condition and in red unstable condition. Instability occurs only when $C_Y > 0.08$ (taken by [30]).

3.1.2 Flutter involving slow flow (low reduced velocity)

In this case the force induced by the fluid is different compared to the previous case, indeed if we develop the expansion of fluid force for low U_R, we obtain a stiffness force and a damping force. The solid body equation is:

$$\ddot{q} + c_F \dot{q} + (1 + k_F) q = 0, \qquad c_F = -\frac{C_Y}{U_R} \left(\frac{\partial F}{\partial \dot{q}} \right)_0, \quad k_F = -C_Y \left(\frac{\partial F}{\partial q} \right)_0 \tag{3.18}$$

If $c_F > 0$ the flow is going to damp the oscillation, conversely if it is negative every perturbation will be amplified in time. We have seen for fast flow that increasing C_Y, the flow brings frequencies of two modes together, and then, for slow flow, a combined mode, with negative damping, appears. The substantial difference to remark is that, for slow flow, the mode could be unstable itself as soon as the fluid force acts(Fig. 3.6). For the general case, with an initial damping c_0, we can predict the reduced velocity for which the instability occurs:

$$\ddot{q} + (c_0 + c_F)\dot{q} + (1 + k_F) q = 0 \qquad U_R^C = \frac{c_0}{M(\frac{\partial F}{\partial \dot{q}})_0} \tag{3.19}$$

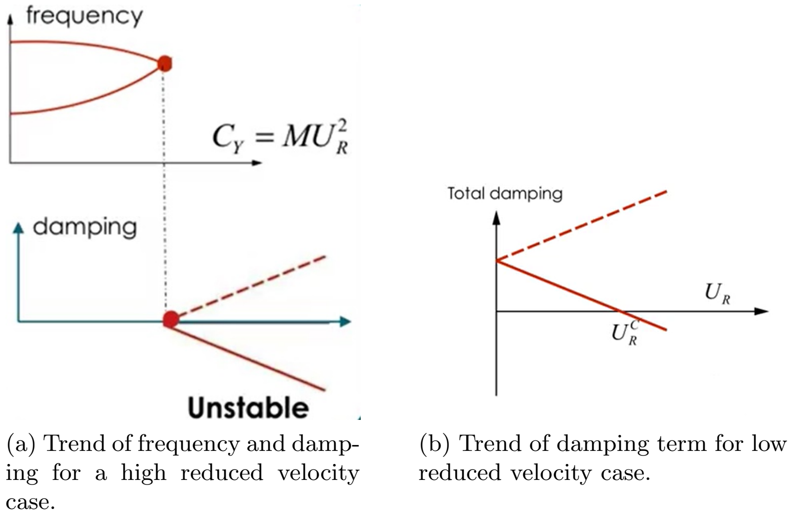
where M is the mass number. This mechanism is much more dangerous than the before case, because it does not require the coincidence of two modes, indeed every single mode could be unstable. We can make an example with an airfoil with only plunge as degree of freedom, developing the equation of motion for the system:

$$M\ddot{y} + \left[\frac{1}{2} \rho U L \left(\frac{\partial C_l}{\partial \theta} \right)_{\theta_0} \right] \dot{y} + K y = 0 \tag{3.20}$$

we can understand that if lift slope is positive there is a positive damping, so there is a stable condition, conversely for high angles of attack dynamic instability occurs. It is also called stall flutter because the lift slope is negative (fig. 3.7).

In stable case the airfoil moves upward, decreasing the apparent angle of attack, reducing in turn the lift; so the variation of the force is opposed to body's velocity, then it acts as a damping force. Conversely for the negative lift slope the variation of the force is in the same way of velocity, causing instability, in the latter region the fluid transfers energy to the solid, causing the stall flutter.

In conclusion, in fast flow, we can encounter the static instability in torsion, with no oscillation for bodies with only one degree of freedom, and dynamic instability, for bodies with two or more degree of freedom, also called coupled mode flutter, where the airfoil combines torsion and plunge



(a) Trend of frequency and damping for a high reduced velocity case.

(b) Trend of damping term for low reduced velocity case.

Figure 3.6: Trend of instability for high and low reduced velocity.

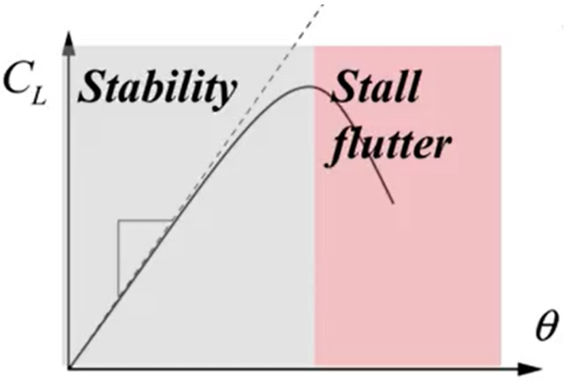


Figure 3.7: Lift slope for a NACA airfoil

motion, that could diverge for specific conditions. For slow flow, we have find out that an airfoil can be unstable in a pure plunge motion, called stall flutter.

3.2 Dynamic stall

Aerodynamic bodies subjected to pitch and plunge motions exhibit a different stalling behaviour from stalling phenomenon of fixed wing. In the case of a dynamically pitching body, such as an airfoil, the shear layer near the leading edge rolls up to form a leading-edge vortex (LEV) which provides additional suction over the upper airfoil surface as it convects downstream. This increases the suction region, leading to gain in lift and delaying the stall, compared to a fixed wing. However the LEV quickly becomes unstable and detaches from the airfoil [34]. The LEV detachment is accompanied by a dramatic decrease in lift and sudden increase in pitching moment. Dynamic stall is not a well-understood phenomenon despite its importance to the performance and operational limits of helicopters, flapping wings, and wind turbines. In fact, dynamic stall can lead to violent vibrations and dangerously high loads in these aerodynamic applications, leading to fatigue and structural failure. But, for energy harvesting's world, dynamic stall is a desirable process. Here we show a NASA report that treats about dynamic stall phenomenon [19].

In Fig. 3.8 we can see the different phases of dynamic stall. In the unseparated region, between points 1 and 2, the lift and pitching moment follow approximately the trends of unsteady linear thin airfoil theory. After point 2 it is found that is formed a vortex at leading edge (Leading Edge Vortex) and starts to move downstream at the trailing edge. The presence of this vortex distorts the pressure distribution and hence the overall lift, indeed it is the LDV vortex the cause of the extra lift, which increases until the vortex is attached to the body. After point 4 LDV is completely detached, it is responsible of lift decreasing. At same time there is the phenomenon of secondary vortices release, their effects can be seen at point 5. These vortex give a temporary increase of lift, but always less than of the first leading edge contribution.

If we refer to pitching moment coefficient we can see in advance the dynamic stall phenomenon, indeed C_m decrease rapidly after point 3.

The dynamic stall does not have a mirror trend and this brings to large hysteresis zones. It is so possible defines an aerodynamic damping coefficient:

$$\zeta = \oint_C c_m d\alpha$$

where ζ expresses the work done by aerodynamic force every period. If ζ

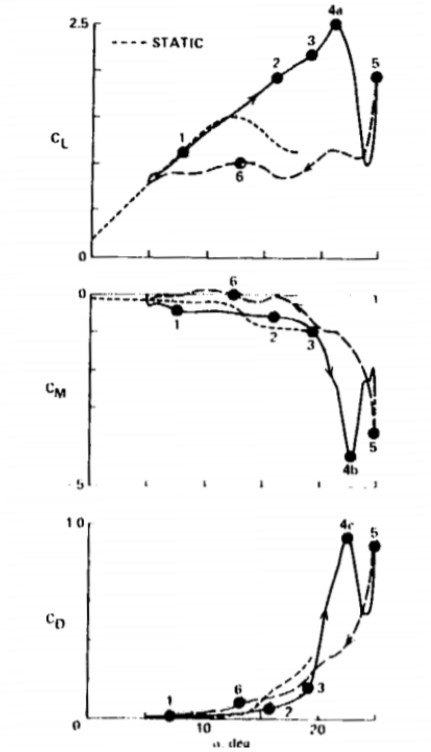


Figure 3.8: Graphs of lift, moment and drag coefficient varying with angle of attack.

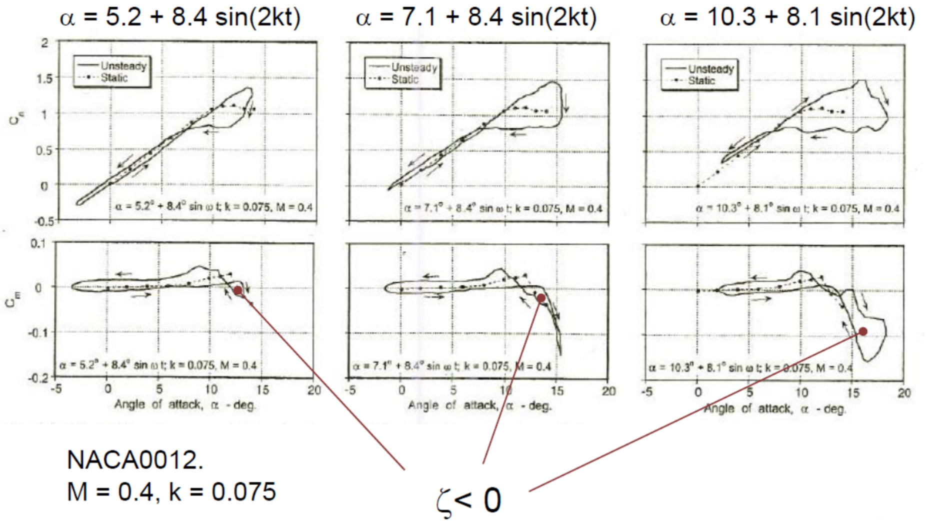


Figure 3.9: The effect of mean angle on lift and moment coefficients, for bigger average AOA ζ positive zone is larger.

is positive, the fluid receives energy from body motion, but if ζ is negative the fluid transfers energy to body and fluctuations tends to growth. From Fig. 3.9 we can note how the mean AOA influences damping coefficient on airfoil with sinusoidal motion.

It is also possible defines four stall regimes for a oscillating body:

1. *No stall*: We are in the case in which the oscillations are such as not cause separations. The profile behaviour is reproducible by linear theories.

2. *Stall onset*: The oscillation amplitude is such as to generate a small stall area. However it does not determine a lift fall.

3. *Light stall*: A portion of the cycle is in conditions of stalled flow. In the remaining part of the cycle the flow is attached and profile operation is linear.

4. *Deep stall*: The angular variation is such as to maintain the profile always in a stalled condition.

In Fig. 3.10 we can appreciate the increasing of hysteresis cycle for bigger body incidence. Some features of deep dynamical stall are intense vortex detachment and the viscous region is order of magnitude of chord length, while in light stall the viscous region is the order of profile's thickness. The geometry influences in particular the light stall, in fact thin leading edge leads to greater adverse pressure gradient, while in case of deep stall geometry does not affect the stall regime.

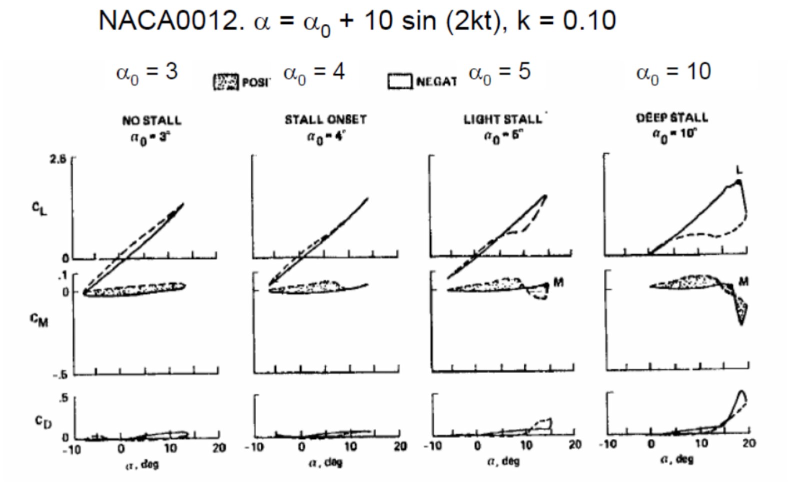


Figure 3.10: Operating regimes by increasing the average incidence.

3.3 Turbulence

We will have a deeper look to turbulence model because in this work we will face on it, modelling the features flow, because a Direct Numerical Simulation has execution time unacceptable. The simulations have a Reynolds regime not so low to allow to use a laminar model, so we will use first RANS approach (Reynolds Average Navier–Stokes) with $k - \omega$ SST model, and then LES approach (Large-Eddy Simulation). For turbulent fluid we mean a flow which has features on many different length scales, which all interact with each other, with no periodicity. The flow can be seen as a sum of averaged quantity plus stochastic fluctuations, for RANS, or a sum of filtered quantity plus high frequency modelled terms for LES.

3.3.1 RANS (Reynolds Averaged Navier–Stokes Equations)

The Reynolds-averaged Navier—Stokes equations (or RANS equations) are time-averaged equations of motion for fluid flow. The idea behind the equations is Reynolds decomposition, whereby an instantaneous quantity is decomposed into its time-averaged and fluctuating quantities, an idea first proposed by Reynolds:

$$\begin{aligned}\mathbf{u} &= \mathbf{u}' + \bar{\mathbf{u}} \\ p &= p' + \bar{p}\end{aligned}$$

these equations can be used with approximations based on knowledge of the properties of flow turbulence to give approximate time-averaged solutions to the Navier—Stokes equations. For a stationary, incompressible Newtonian fluid, these equations can be written in Einstein notation as:

$$\rho \bar{u}_j \frac{\partial \bar{u}_i}{\partial x_j} = \rho \bar{f}_i + \frac{\partial}{\partial x_j} \left[-\bar{p}\delta_{ij} + \mu \left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i} \right) - \rho \overline{u_i' u_j'} \right] \tag{3.21}$$

the continuity equation is unchanged respect to average operation. The fundamental nonlinear term to understand is $\overline{u_i' u_j'}$ named Reynolds stress, which has to be modelled in order to resolve the NS equations.

The problem can be closed introducing the concept of eddy viscosity, conceived by Boussinesq. He proposed relating turbulence stresses to the gradient of mean flow to close the system with a new constant ν_t:

$$-\overline{u_i' u_j'} = \nu_t \left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i} - \frac{2}{3} \frac{\partial \bar{u}_k}{\partial x_k} \right) - \frac{2}{3} K \delta_{ij} \tag{3.22}$$

where $K = \frac{1}{2}\overline{u_i' u_j'}$ is the turbulence kinetic energy. In this model, the additional turbulence stresses are given by augmenting the molecular viscosity with an eddy viscosity. The Boussinesq hypothesis is employed in the Spalart–Allmaras (S–A), k–ϵ (k–epsilon), and k–ω (k–omega) models and offers a relatively low cost computation for the turbulence viscosity ν_t. The S–A model uses only one additional equation to model turbulence viscosity transport, while the k models use two.

Prandtl introduced the additional concept of the mixing length, along with the idea of a boundary layer. For wall-bounded turbulent flows, the eddy viscosity must vary with distance from the wall, hence the addition of the concept of a 'mixing length'. In the simplest wall-bounded flow model, the eddy viscosity is given by the equation:

$$\nu_t = \left| \frac{\partial u}{\partial y} \right| l_m^2 \tag{3.23}$$

where $\frac{\partial u}{\partial y}$ is the velocity gradient on the wall and l_m is the mixing length. This simple model is the basis for the "law of the wall", which is a surprisingly accurate model for wall-bounded, attached (not separated) flow fields with small pressure gradients.

In final, Smagorinski, proposed a useful formula for the eddy viscosity in numerical models, based on the local derivatives of the velocity field and the local grid size:

$$\nu_t = c_s \Delta x \Delta y \sqrt{ \left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial y} \right)^2 + \frac{1}{2} \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right)^2 } \tag{3.24}$$

where c_s is an empirical value equal to 0.16.

k-ω SST

Here we present the turbulence model for RANS simulations used in this work.

k-ω SSTturbulence model combine the best of k-ω and SST (shear stress transport). The use of k-ω formulation in the inner parts of the boundary layer makes the model directly usable all the way down to the wall through the viscous sub-layer, hence the SST k-ω model can be used as a Low-Re turbulence model without any extra damping functions. The SST formulation also switches to a k-ϵ behaviour, converting it in a k-ω model, in the free-stream and thereby avoids the common k-ω problem that the model is too sensitive to the inlet free-stream turbulence properties. Authors who use the SST k-ω model often merit it for its good behaviour in adverse pressure gradients and separating flow. The SST k-ω model does produce a bit too large turbulence levels in regions with large normal strain, like stagnation regions and regions with strong acceleration. This tendency is much less pronounced than with a normal k-ϵ model though. There are necessary requirement to satisfy to set a right formulation of k-ω SST:

1. The standard k- ω model and the transformed k- ϵ model are both multiplied by a blending function and both models are added together. The blending function is designed to be one in the near-wall region, which activates the standard k- ω model, and zero away from the surface, which activates the transformed k- ϵ model.

2. The SST model incorporates a damped cross-diffusion derivative term in the ω equation.

3. The definition of the turbulent viscosity is modified to account for the transport of the turbulent shear stress.

4. The modeling constants are different.

These features make the SST k- ω model more accurate and reliable for a wider class of flows than the standard k- ω model. Other modifications include the addition of a cross-diffusion term in the ω equation and a blending function to ensure that the model equations behave appropriately in both the near-wall and far-field zones. The governing equations are:

$$\frac{\partial k}{\partial t} + U_j \frac{\partial k}{\partial x_j} = P_k - \beta^* k\omega + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_k \nu_t)\frac{\partial k}{\partial x_j}\right] \tag{3.25}$$

$$\frac{\partial \omega}{\partial t} + u_j \frac{\partial \omega}{\partial x_j} = \alpha S^2 - \beta\omega^2 + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_\omega \nu_t)\frac{\partial \omega}{\partial x_j}\right] + 2(1-F_1)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i} \tag{3.26}$$

where F_1 is the blending function, which values 0 near the wall and 1 in the free stream:

$$F_1 = tanh\{\{min\big[max\big(\frac{\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\big), \frac{4\sigma_{\omega 2}k}{CD_{k\omega}y^2}\big]\}^4\} \tag{3.27}$$

$$CD_{k\omega} = max\left(2\rho\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}, 10^{-10}\right) \tag{3.28}$$

eddy viscosity and second blending function are:

$$\nu_t = \frac{a_1 k}{max(a_1\omega, SF_2)} \tag{3.29}$$

$$F_2 = tanh\left[max\left(\frac{2\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right)\right]^2 \tag{3.30}$$

for avoiding excessive turbulent kinetic energy near stagnation point is introduced production limiter P_k:

$$P_k = min\left(\tau_{ij}\frac{\partial U_i}{\partial x_j}, 10\beta^*\omega k\right) \tag{3.31}$$

finally the boundary condition for a domain with scale length L are:

$$\frac{U_\infty}{L} < \omega < 10\frac{U_\infty}{L} \tag{3.32}$$

$$\frac{10^{-5}U_\infty^2}{Re_L} < k < 0.1\frac{10^{-5}U_\infty^2}{Re_L} \tag{3.33}$$

3.3.2 LES (Large-Eddy Simulation)

Large-Eddy Simulation (LES) is a mathematical model for turbulence used in computational fluid dynamics. It was initially proposed in 1963 by Joseph Smagorinsky and is currently applied in a wide variety of engineering applications, including combustion, acoustics, and simulations of the atmospheric boundary layer. The simulation of turbulent flows by numerically solving the Navier–Stokes equations requires resolving a very wide range of time and length scales, all of which affect the flow field. Such a resolution can be achieved with direct numerical simulation (DNS), but DNS is computationally expensive, and its cost prohibits simulation of practical engineering systems with complex geometry or flow configurations, such as turbulent jets, pumps, vehicles, and landing gear. Large eddy simulation uses a an implication of Kolmogorov's (1941) theory of self similarity, that is that the large eddies of the flow are dependant on the geometry while the smaller scales more universal. This feature allows one to explicitly solve for the

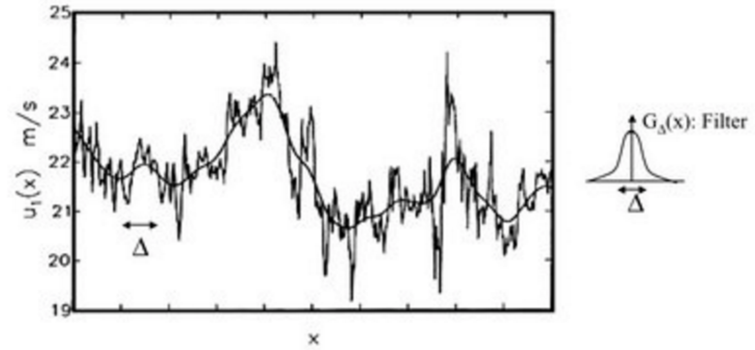


Figure 3.11: Difference between DNS simulation and LES simulation. LES does not contain fluctuation because equation has been filtered with a kernel filtering with a determined shape G and width Δ.

large eddies in a calculation and implicitly account for the small eddies by using a subgrid-scale model (SGS model). Subgrid-scale modeling refers to the representation of important small-scale physical processes that occur at length-scales that cannot be adequately resolved on a computational mesh. In Large-Eddy Simulation of turbulence, subgrid-scale modeling is used to represent the effects of unresolved small-scale fluid motions (small eddies, swirls, vortices) in the equations governing the large-scale motions that are resolved in computer models. The formulation of physically realistic SGS models requires understanding of the physics and the statistics of scale interactions in hydrodynamic turbulence, and is an open research question owing to the fact that turbulence remains an unsolved problem in classical physics.

In Large-Eddy Simulation of turbulence, the time and space dependence of the fluid motions are resolved down to some prescribed length-scale Δ, the motion is separated into small and large scales, often by spatially filtering the velocity field with a kernel $G_\Delta(\mathbf{x})$. The convolution kernel eliminates scales smaller than Δ . The eliminated scales are called subfilter or subgrid-scale motions (Fig. 3.11). The spectrally sharp filter, the Gaussian filter, and the box or top-hat filter are often used. The filtered velocity (denoted by a tilda), is thus obtained by convolution:

$$\tilde{u}_i(\mathbf{x}) = \int G(\mathbf{x} - \xi)u(\xi)d\xi \tag{3.34}$$

so velocity can be seen has a filtered part plus subgrid-scale part: $u_i = \tilde{u}_i + u_i'$.

The most common filter used are Box filter and Gaussian filter, we report the physical filter kernel ($G(\mathbf{x})$) and filtering kernel in Fourier-wavenumber space ($\widehat{G(k)}$):

$$
\begin{aligned}
G(\mathbf{x})_{\text{box}} &= \frac{1}{\Delta} H\left(\frac{1}{2}\Delta - |\mathbf{x}|\right) && (3.35) \\
G(\mathbf{x})_{\text{Gauss}} &= \left(\frac{6}{\pi\Delta^2}\right)^{1/2} e^{\frac{-6\mathbf{x}^2}{\Delta^2}} && (3.36) \\
\widehat{G(k)}_{\text{box}} &= \frac{sin(\frac{k\Delta}{2})}{\frac{k\Delta}{2}} && (3.37) \\
\widehat{G(k)}_{\text{Gauss}} &= e^{\frac{-k^2\Delta^2}{24}} && (3.38)
\end{aligned}
$$

the filtered Navier–Stokes equations become:

$$
\frac{\partial \tilde{\mathbf{u}}}{\partial t} + \tilde{\mathbf{u}} \cdot \nabla \tilde{\mathbf{u}} = -\frac{1}{\rho} \nabla \tilde{p} + \nu \nabla^2 \tilde{\mathbf{u}} - \nabla \cdot \tau, \qquad \nabla \cdot \tilde{\mathbf{u}} \tag{3.39}
$$

the SGS stress tensor τ is defined according to: $\tau_{ij} = \widetilde{u_i u_j} - \tilde{u}_i \tilde{u}_j$. The LES equations can be discretized numerically by employing a spatial resolution that is on the order of Δ, when Δ is chosen to be much larger than the Kolmogorov scale η_K, LES is typically far less expensive than Direct Numerical Simulation which requires resolutions near η_K. However the filtered equation are still unclosed for the presence of stress tensor, we need again additional equation. Physically, the SGS stress determines the dynamical coupling between large and small scales in turbulence. Dimensionally, it scales quadratically with turbulent velocity differences at scales of order Δ, Unlike Reynolds stresses, the SGS stress is a fluctuating turbulence quantity, i.e. a complicated time-dependent field $\tau(\mathbf{x}, t)$. Traditionally, the effects of SGS motions upon resolved scales are modeled in analogy with molecular degrees of freedom in kinetic theory of gases, in which the momentum fluxes are linearly dependent upon the rate of strain of the large scales. This is written as an eddy-viscosity closure according to:

$$
\tau_{ij} = -2\nu_{sgs} \hat{S}_{ij} \tag{3.40}
$$

the issue is model the eddy viscosity ν_{sgs} for every type of problem. A closure equation can be given by Smagorinski model:

$$
\tau_{ij}^{ev} = -2(c_s\Delta)^2 |\tilde{S}| \tilde{S}_{ij} \tag{3.41}
$$

where $|\tilde{S}|$ represents a Galilean invariant estimation of velocity differences over length-scales of order Δ, c_s is Smagorinski coefficient and $\tilde{S}_{ij}$ is filtered shear stress tensor. The Smagorinsky model is the oldest of LES SGS models, but because of its simplicity it is still widely used. It is not a particularly good choice for wall-bounded shear flows, but for flows far from solid boundaries it can be quite adequate. Using the requirement that modeled rate of energy transfer from large to small scales, $< \tau_{ij}\hat{S}_{ij} >$ is equal to overall rate dissipation of turbulent kinetic energy $< \epsilon >$, exploiting the insight provided by Kolmogorov theory to evaluate the moment of strain $< \hat{S}_{ij}\hat{S}_{ij} >$ and c_s derived empirically equal to 0.16, it is possible to find out the SGS stress tensor contributes. The Smagorinsky model assumes that an equilibrium exists between kinetic energy flux across scale and the large scales of turbulence. In many applications, such as free stream turbulence impinging on the leading edge of an airplane wing, turbulent water flow through a pipe with changing cross section, or turbulence with strong buoyancy, such equilibrium conditions are not established.

For that, in order to model the time development of the small-scale turbulence, which affects the eddy viscosity, so-called kinetic energy models use an eddy viscosity of the form $\nu_{sgs} = c_e \Delta \sqrt{e(\mathbf{x},t)}$, where $e(\mathbf{x},t)$ is the SGS kinetic energy. It is defined in terms of the trace of the SGS stress tensor according to $e(\mathbf{x},t) = \tau_{kk}/2$, and is used to quantify the local velocity scale. In order to determine $e(\mathbf{x},t)$, an additional scalar transport equation, derived from the trace of the transport equation for the SGS stress tensor, is solved. In this equation, diffusion and dissipation terms must be modeled:

$$\frac{\partial e}{\partial t} + \hat{u}_j \frac{\partial e}{\partial x_j} = c_e \Delta \sqrt{e} \hat{S}_i j \hat{S}_i j - C_\epsilon \frac{e^{3/2}}{\Delta} - \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{c_e}{\sigma_e}\Delta\sqrt{e}\right)\frac{\partial e}{\partial x_j}\right] \quad (3.42)$$

with the two additional empirical coefficients, the SGS Prandtl number σ_e and the dissipation parameter C_ϵ . This approach incorporates memory effects and has seen extensive applications especially in simulations of atmospheric flows [21].

All the above expressions are based on the basic eddy-viscosity closure assumption in Eq.(3.40). This expression can also be (partly) justified from the linearized evolution equation for the SGS fluctuating velocity $\partial u_i'/\partial t = -u_k' \partial \hat{u}_i/\partial x_k$, which includes several strong assumptions such as neglect of pressure and nonlinear effects. Coupled with further assumptions of separation of time-scales, the solution can be written in terms of matrix exponentials and short-time expansion to first order yields the eddy-viscosity closure for the deviatoric part of the SGS stress tensor. The need for the various assumptions illustrates the limitations of the eddy-viscosity closure from a fundamental viewpoint. Still, in practical applications of LES, the eddy-viscosity approach is very popular due to its robustness, general ease of implementation, and low computational cost.

Germano identity and dynamic model

The limitations of the Smagorinsky model have led to the formulation of more general subgrid-scale models. Perhaps the best known of these newer models is the dynamic subgrid-scale (DSGS) model of Germano [9]. The DSGS model may be viewed as a modification of the Smagorinsky model, as the dynamic model allows the Smagorinsky constant c_s to vary in space and time and is calculated locally in each timestep based upon two filterings of the flow variables, which we will denote by a $\tilde{}$that is grid filter, with Δ width, used by now and with a superscript t to indicate the test filter, which its width is assumed to be larger than grid filter ($\alpha\Delta$, $\alpha > 1$).

Remembering that the SGS stress tensor is:

$$\tau_{ij} = \widetilde{u_i u_j} - \tilde{u}_i \tilde{u}_j \tag{3.43}$$

now we filter again the latter equation with test filter:

$$T_{ij} = (\widetilde{u_i u_j})^t - (\tilde{u}_i \tilde{u}_j)^t \tag{3.44}$$

the two subgrid-scale stress terms are related by Germano identity:

$$\mathcal{L} = T_{ij} - \tau_{ij}^r = (\tilde{u}_i \tilde{u}_j)^t - \tilde{u}_i{}^t \tilde{u}_j{}^t \tag{3.45}$$

that is the resolved turbulent stress. The Germano identity is used to calculate dynamic local values for C_s by applying the Smagorinsky model to both T_{ij} and τ_{ij}.

Remarkably, all the terms in the latter equations are computable from the resolved field. $\mathcal{L}$ represents the 'band-pass filtered' contribution to the nonlinear term in the scale range between the grid and the test filter levels (fig. 3.12).

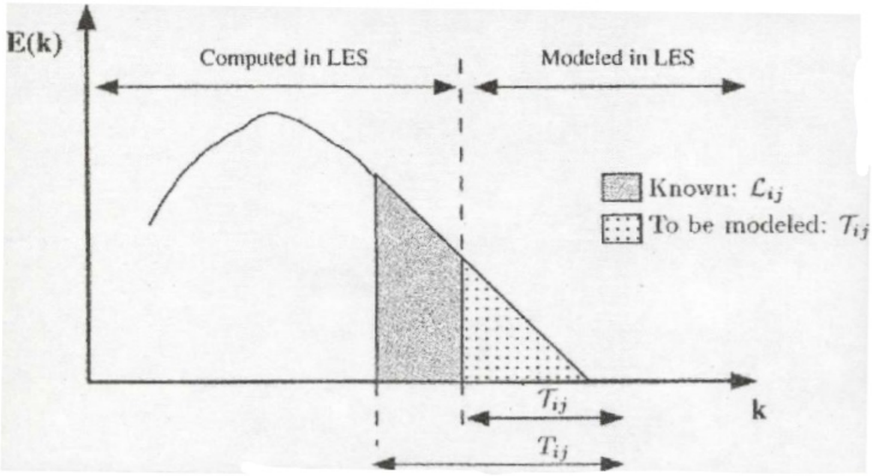

E(k)
Computed in LES
Modeled in LES
▨ Known: $\mathcal{L}_{ij}$
⁙ To be modeled: $\mathcal{T}_{ij}$
k
$\mathcal{T}_{ij}$
T_{ij}

Figure 3.12: Plot of energy spectrum of turbulence. If we do not use dynamic model we do not resolve non linear term in the inertial scale. If we use dynamic model in the region $\mathcal{L}$ we can resolve non linear terms. We can compute more non linear effect if the test filter width is greater.

Chapter 4

Mathematical model and numerical method

Here we will show the mathematical model adopted and the numerical method used to resolve the Navier-Stokes equation. OpenFOAM[10] is the leading free, open source software for computational fluid dynamics, owned by the OpenFOAM Foundation and distributed exclusively under the General Public Licence (GPL). The GPL gives users the freedom to modify and redistribute the software and a guarantee of continued free use, within the terms of licence.

4.1 Equation used in this work

4.1.1 Incompressible Navier-Stokes equation

We will deal with incompressible Navier-Stokes equation, because Mach number effect are negligible for our problem. The equation set is:

$$\begin{cases} \nabla \cdot \mathbf{u} & = 0 \\ \dfrac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \frac{\nabla p}{\rho} - \nu \nabla^2 \mathbf{u} - \mathbf{f} & = 0 \end{cases} \tag{4.1}$$

where f is volumetric force. These equations are used for resolving fluid field.

4.1.2 Rigid body equation

The wing presented in this thesis, assumed as a rigid body, has a pitch and plunge motion because of fluid-structure interaction, or rather through forces and torques exercised by the flow on the wing, and it is constrained to four elastomer. An important step of the work concerns the development of the *dynamicMeshDict*, in which it has been specified the solver used for rigid body motion (*sixDofRigidBodyMotion*), the body's properties (mass, moment of inertia,...) and type of constraint (four non-linear spring). The parameters for rigid body equations are:

m_b	mass of rigid body
$\mathbf{x}_{cm}(t)$	position of center of mass
$\mathbf{v}_{cm}(t)$	velocity of the center of mass
$\mathbf{a}_{cm}(t)$	acceleration of the center of mass
$\mathbf{F}(t)$	force on the body
$\mathbf{G}(t)$	torque on the body
$\mathbf{h}(t)$	angular momentum
$\mathbf{e_i}(t)$	principal axes of inertia, $i = 1, 2, 3$
$\omega(t)$	angular velocity
$E(t) \in \mathbb{R}^{3\times3}$	matrix with columns $\mathbf{e_i}$
$R(t) \in \mathbb{R}^{3\times3}$	rotation matrix
$A(t) \in \mathbb{R}^{3\times3}$	moment of inertial tensor
I_i	moments of inertia

the rigid body motion equation in a cartesian reference are:

$$\begin{aligned} \dot{\mathbf{x}}_{cm} &= \mathbf{v}_{cm} && (4.2) \\ m_b \dot{\mathbf{v}_{cm}} &= \mathbf{F} && (4.3) \\ \dot{\mathbf{h}} &= \mathbf{G} && (4.4) \end{aligned}$$

where:

$$\begin{aligned} \mathbf{F} &= \int_{\partial\Omega} \mathbf{f}_s ds + \mathbf{f}_b \qquad (\mathbf{f}_s = \text{surface force, } \mathbf{f}_b = \text{body force}) && (4.5) \\ \mathbf{G} &= \int_{\partial\Omega} (\mathbf{x} - \mathbf{x}_{cm}) \times \mathbf{f}_s ds + \mathbf{g}_b \qquad (\mathbf{g}_b = \text{body torque}) && (4.6) \end{aligned}$$

where $\partial\Omega$ is the surface of rigid body. The angular momentum $\mathbf{h}$ is given by:

$$\mathbf{h} = A(t)\omega \tag{4.7}$$

and $A(t)$ is defined by:

$$A(t) = \int_{\Omega} \rho(\mathbf{x})[\mathbf{y}^T\mathbf{y}\mathbf{I} - \mathbf{y}^T\mathbf{y}]d\mathbf{x}, \quad \mathbf{y} = \mathbf{x} - \mathbf{x}_{cm} \tag{4.8}$$

with $\mathbf{I}$ identity matrix.

The eigenvalue of $A(t)$ are principal inertia moment I_i and eigenvectors are $\mathbf{e}_i$ the principal inertia axes.

$$\begin{aligned}
A\mathbf{e}_i &= I_i\mathbf{e}_i && (4.9)\\
\Lambda &= diag(I1, I2, I3) && (4.10)\\
\mathbf{e}_i \cdot \mathbf{e}_j &= \delta_{ij} && (4.11)\\
E &= [\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3] && (4.12)\\
A &= E\Lambda E^T && (4.13)\\
E^{-1} &= E^T && (4.14)
\end{aligned}$$

from the definition of $\mathbf{h}$ as $\mathbf{h} = A(t)\omega$ it follows that the evolution for the angular velocity ω is:

$$A\dot{\omega} = -\Omega A\omega + \mathbf{G} \tag{4.15}$$

where Ω is the rotation matrix that is unknown.

So we have to solve the following set of ODEs to resolve a rigid body motion problem:

$$\begin{cases}
\dot{\mathbf{x}}_{cm} &= \mathbf{v}_{cm}\\
m_b\mathbf{v}_{cm} &= \mathbf{F}\\
A\dot{\omega} &= -\Omega A\omega + \mathbf{G}\\
\dot{\mathbf{e}}_i &= \omega \times \mathbf{e}_i
\end{cases} \tag{4.16}$$

4.2 OpenFOAM structure

OpenFOAM is a C++ library used primarily to create executables, known as applications, and to start a large set of precompiled applications has distributed to users, who have the freedom to create their own or modify existing ones. Applications are split into two main categories:

- **Solvers**: that are each designed to solve a specific problem in computational continuum mechanics;
- **Utilities**: that perform simple pre-and post-processing tasks, mainly involving data manipulation and algebraic calculations.

The fluid dynamic problem is developed through object orientation language, so that it is possible to declare classes,as well as to implement types and associated operations, furthermore from a general class it is possible to create a more specific class without re-writing all the code. In OpenFOAM a certain structure of the input files is expected. A case has to be set up in a predestined manner which contains a minimum of three directories. A *constant* folder and a *system* folder are needed. Also a time folder is

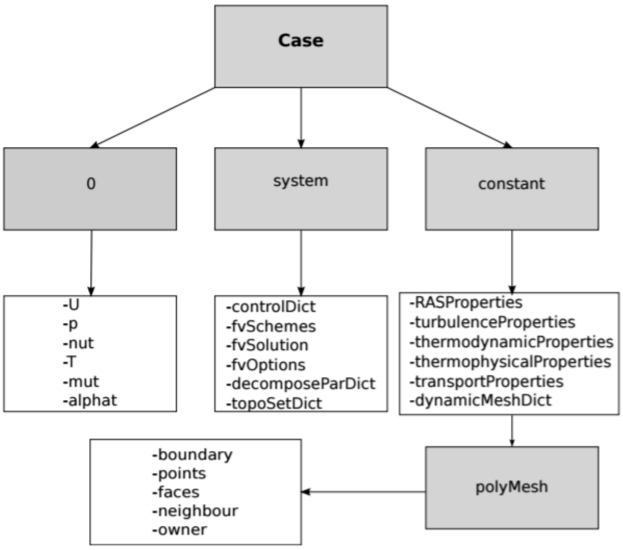


Figure 4.1: Case structure in OpenFOAM.

needed, which is usually named 0, but can be named differently if 0 is not the starting time. There are also subfolders and files that are contained in the mentioned folders, a few options are reviewed here. The structure of an OpenFOAM case can be seen in figure 4.1.

4.2.1 0 folder

This folder contains files with the initial conditions of the used variables. For laminar incompressible Navier-Stokes equation only files containing the initial conditions of U and p are needed. For turbulence models or compressible flow other variables will need to be added as well. Three entries have to be done for each variable file. The dimension of the variable is assigned through dimensions in the file (for instance m/s for the velocity). The internal field is assigned through internal Field and the boundary field is given through boundary Field.

4.2.2 Constant folder

This folder contains specifications for turbulence and fluid properties. Depending on the solver chosen, different files need to be specified. For all solvers which calculate the RANS equations, the file *RAS Properties* determines the turbulence model used, the same goes for LES equation. The type of turbulence model applied is determined in *turbulenceProperties* where either LES, RAS or laminar model can be chosen. For incompressible solvers the file *transportProperties* determines the behaviour of the kinematic viscosity ν. For the compressible solver *rhoCentralFoam* temperature dependency is determined in the files *thermodynamicProperties* and *thermophysicalProperties.* For dynamic mesh operations, the file *dynamicMeshDict* prescribes the conditions of constraints and restraints. To use dynamic mesh operations specific solvers have to be used such as *pimpleDyMFoam.*

PolyMesh folder

OpenFOAM uses a cell-centered control volume for its calculations. In the *polyMesh* folder files are contained describing the mesh. These files include points, which contain the points of the mesh. Faces, which contain the faces of the cells. Owner, that contains what faces belong to a cell and neighbour which contains the information about the connectivity between cells. Also the boundaries are given in the file boundary, where the boundaries are assigned names and also of what type they are, such as empty, wall or patch for instance. The mesh can build up with *blockMesh* or imported from another mesh software.

BlockMesh utility

The principle behind blockMesh is to decompose the domain geometry into a set of 1 or more three dimensional, hexahedral blocks. Edges of the blocks can be straight lines, arcs or splines. The mesh is ostensibly specified as a number of cells in each direction of the block, sufficient information for blockMesh to generate the mesh data. Each block of the geometry is defined by 8 vertices, one at each corner of a hexahedron. The vertices are written in a list so that each vertex can be accessed using its label, remembering that OpenFOAM always uses the C++ convention that the first element of the list has label 0. An example block is shown in figure 4.2 with each vertex numbered according to the list. The edge connecting vertices 1 and 5 is curved to remind the reader that curved edges can be specified in blockMesh.

To create a grid with blockMesh the steps are:

- Define position of every vertex.
- Define which vertex define each block.
- Define the type of edge of each edge (line, arc,...).
- Define faces to set up boundary conditions (wall, inlet, outlet,...).

4.2.3 System folder

This folder contains the specifications for the simulation. In the *decomposeParDict* file the mesh is decomposed into an assigned number of parts for parallel simulations. The *topoSetDict* creates sets in the mesh, which

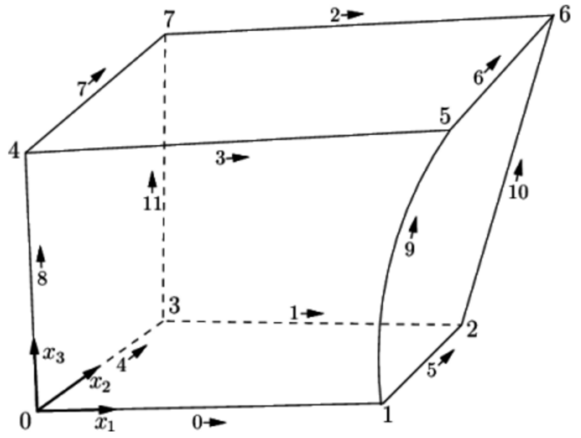


Figure 4.2: Sketch of single block, the face are oriented and edge can be arc.

can be used to define areas with extra source terms. For the *fvOptions* file extra source terms can be assigned to sets. In *controlDict* the frequency of solution file outputs, run time, time steps and Courant number are assigned.

4.3 Numerical Schemes and algorithm control

4.3.1 FvSchemes

We will take a deeper look to *fvschemes*, which sets the numerical schemes for terms, such as derivatives that are calculated during a simulation. The set of terms, for which numerical schemes must be specified, are subdivided within the *fvschemes* dictionary into the categories below:

- timeScheme: first and second time derivatives ($\partial/\partial t, \partial^2/\partial t^2$)
- gradSchemes: gradient ∇
- divSchemes: divergence $\nabla\cdot$
- laplacianSchemes: Laplacian ∇^2
- interpolationSchemes: cell to face interpolations of values.
- snGradSchemes: component of gradient normal to a cell face.
- wallDist: distance to wall calculation, where required.

Each keyword in represents the name of a sub-dictionary which contains terms of a particular type, such as ∇p, and must be specified how the terms have to be discretized like this:

```
ddtSchemes
{
    default         Euler;
}

gradSchemes
{
    default         Gauss linear;
}

divSchemes
{
    default         none;
    div(phi,U)      bounded Gauss linearUpwind grad(U);
    div(phi,k)      bounded Gauss upwind;
    div(phi,epsilon) bounded Gauss upwind;
    div(phi,R)      bounded Gauss upwind;
    div(R)          Gauss linear;
    div(phi,nuTilda) bounded Gauss upwind;
    div((nuEff*dev2(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}
```

The schemes that we are interested in are: time schemes $\partial/\partial t$), gradient schemes (∇), divergence schemes ($\nabla\cdot$) and Laplacian schemes (Δ). The user can specify through subdictionary of the previous dictionary, the right discretization which is needed. Starting from time schemes, we can choose between: steady state, which set time derivatives to zero. Euler resolves transient problem bounded, with first order precision. Backward resolves transient problem potentially unbounded, with second order precision. Crank-Nicolson for transient bounded problem with second order precision, and least one local Euler for pseudo transient for accelerating a solution to steady state.

Gradient schemes sub-dictionary contains gradient terms, the default discretization scheme is Gauss linear. The Gauss entry specifies the standard finite volume discretization of Gaussian integration which requires the interpolation of values from cell centres to face centres. The interpolation scheme is then given by the linear entry, meaning linear interpolation or central differencing. Another important scheme is cell limited Gauss linear 1, which limits the gradient such that when cell values are extrapolated to faces using the calculated gradient, the face values do not fall outside the bounds of values in surrounding cells. A limiting coefficient is specified after the underlying scheme for which 1 guarantees boundedness and 0 applies no limiting; 1 is invariably used.

Divergence schemes sub-dictionary contains only divergence terms, excluding Laplacian terms. The schemes are all based on Gauss integration, using the flux ϕ (volumetric flux of velocity on the cell faces for constant-density flows and the mass flux for compressible flows) and the advected field being interpolated to the cell faces by one of a selection of schemes, e.g. linear, linearUpwind, etc. There is a bounded variant of the discretization, discussed later. The interpolation schemes can be: linear for unbounded problem and second order precision, linear upwind for much less unbounded problem, Lust for 75% linear and 25% linear upwind, it requires a coefficient to tend towards to linear or linear upwind.

Laplacian schemes are useful to discretize diffusion term in momentum equation, the Gauss scheme is the only choice of discretization and requires a selection of both an interpolation scheme for the diffusion coefficient

4.3.2 fvSolution

The equation solvers, tolerances and algorithms are controlled from the *fvSolution* dictionary in the system directory. The first sub-dictionary in our example is solvers. It specifies each linear-solver that is used for each discretized equation; here, the term linear-solver refers to the method of number-crunching to solve a matrix equation, as opposed to an application solver, such as simpleFoam which describes the entire set of equations and algorithms to solve a particular problem. The term ‘linear-solver’ is abbreviated to ‘solver’ in much of what follows; hopefully the context of the term avoids any ambiguity. The syntax for each entry within solvers starts with a keyword that is of the variable being solved in the particular equation. For example, icoFoam solves equations for velocity **U** and pressure p, hence the entries for U and p. The keyword relates to a sub-dictionary containing the type of solver and the parameters that the solver uses. The solver is selected through the solver keyword: PCB/PBiCG p reconditioned (bi-)conjugate gradient, with PCG for symmetric matrices, PBiCG for asymmetric matrices, smoothSolver solver that uses a smoother, GAMG generalised geometric-algebraic multi-grid, diagonal: diagonal solver for explicit systems. The matrices are generally sparse, consequently the solvers are iterative, i.e. they are based on reducing the equation residual over successive solutions. The residual is ostensibly a measure of the error in the solution so that the smaller it is, the more accurate the solution. More precisely, the residual is evaluated by substituting the current solution into the equation and taking the magnitude of the difference between the left and right hand sides; it is also normalised to make it independent of the scale of the problem being analysed. Before solving an equation for a particular field, the initial residual is evaluated based on the current values of the field. After each solver iteration the residual is re-evaluated. The solver stops if any one of the following conditions are reached: the residual falls below the solver tolerance, the ratio of current to initial residuals falls below the solver relative tolerance, the number of iterations exceeds a maximum number of iterations. The solver tolerance should represent the level at which the residual is small enough that the solution can be deemed sufficiently accurate. The solver relative tolerance limits the relative improvement from the initial to the final solution. In transient simulations, it is usual to set the solver relative tolerance to 0 to force the solution to converge to the solver tolerance in each time step. Most fluid dynamics solver applications in OpenFOAM use either the pressure-implicit split-operator (PISO), the semi-implicit method for pressure-linked equations (SIMPLE) algorithms, or a combined PIMPLE algorithm, this latter will be used in this work. These algorithms are iterative procedures for coupling equations for momentum and mass conservation, PISO and PIMPLE being used for transient problems and SIMPLE for steady-state. Within time, or solution, step, both algorithms solve a pressure equation, to enforce mass conservation, with an explicit correction to velocity to satisfy momentum conservation. They optionally begin each step by solving the momentum equation — the so-called momentum predictor. While all the algorithms solve the same governing equations (albeit in different forms), the algorithms principally differ in how they loop over the equations. The looping is controlled by input parameters. They are set in a dictionary named after the algorithm. *Ncorrectors* used by PISO, and PIMPLE, sets the number of times the algorithm solves the pressure equation and momentum corrector in each step; typically set to 2 or 3. *NNonOrthogonalCorrectors* used by all algorithms, specifies repeated solutions of the pressure equation, used to update the explicit non-orthogonal correction, of the Laplacian term. *nOuterCorrectors* used by PIMPLE, it enables looping over the entire system of equations within the time step, representing the total number of times the system is solved. *MomentumPredictor* switch that controls solving of the momentum predictor.

The PIMPLE algorithm plans to calculate in order: momentum, pressure, recalculate momentum from new pressure and recalculate pressure from a new momentum. In formulas:

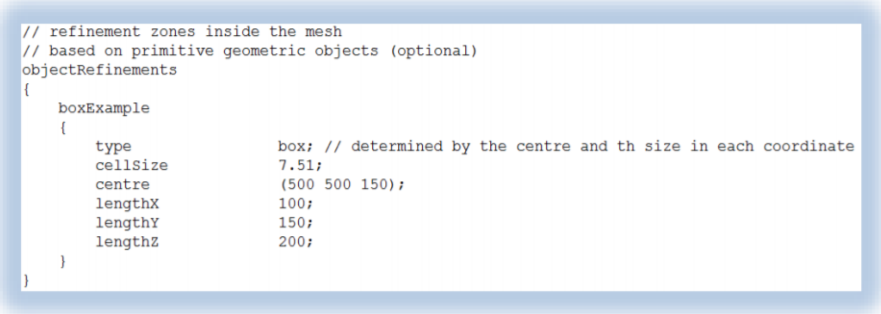
- $\dfrac{\partial \mathbf{U}}{\partial t} + \nabla \cdot (\mathbf{U}\mathbf{U}) + \nabla \cdot \mathbf{R} = -\nabla p$
- $\nabla^2 p = f(\mathbf{U}, \nabla p)$
- $\Rightarrow p_{new}$
- *Correct* $\mathbf{U}_{corrected}$ *with* $p_{new} \Rightarrow \quad \mathbf{U}_{corrected}$
- *Now you use the* $\boldsymbol{U}_{corrected}$ *and* p_{new} *for the second loop*
- $\dfrac{\partial \mathbf{U}}{\partial t} + \nabla \cdot (\mathbf{U}\mathbf{U}) + \nabla \cdot \mathbf{R} = -\nabla p$
- $\nabla^2 p = f(\mathbf{U}, \nabla p)$
- $\Rightarrow p_{new}$
- *Correct* $\mathbf{U}_{corrected}$ *with* $\quad p_{new} \Rightarrow \quad \mathbf{U}_{corrected}$

4.4 Mesh generator used in this work

In this work has been chosen to create the grid with an external program but fully compatible with OpenFOAM, named *cfMesh* [16].

4.4.1 cfMesh

For OpenFOAM cases the grid are created using *cfMesh*, which is a cross-platform library for automatic mesh generation that is built on top of OpenFOAM. It is licensed under GPL, and compatible with all recent versions of OpenFOAM and foam-extend. It supports various 3D and 2D workflows, built by using components from the main library, which are extensible and can be combined into various meshing workflows. The core library utilises the concept of mesh modifiers, which allows for efficient parallelisation using both shared memory parallelisation (SMP) and distributed memory parallelisation using MPI. In addition, special care has been taken on memory



```
// refinement zones inside the mesh
// based on primitive geometric objects (optional)
objectRefinements
{
    boxExample
    {
        type                box; // determined by the centre and th size in each coordinate
        cellSize            7.51;
        centre              (500 500 150);
        lengthX             100;
        lengthY             150;
        lengthZ             200;
    }
}
```

Figure 4.3: Example of box refinement.

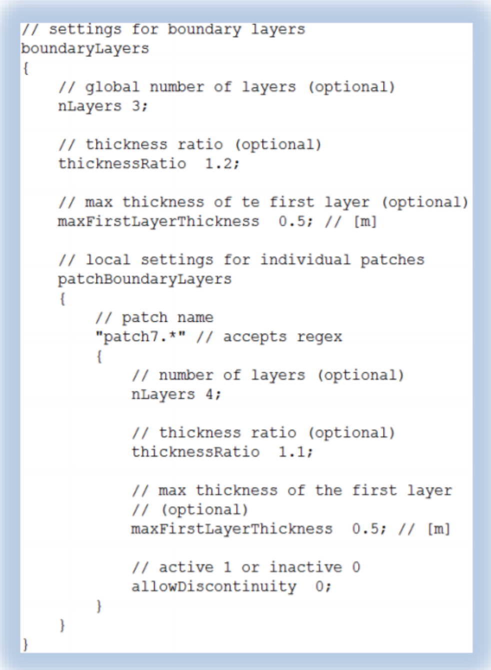
usage, which is kept low by implementing data containers (lists, graphs, etc.) that do not require many dynamic memory allocation operations during the meshing process. The meshing process is steered by the settings provided in a meshDict dictionary located in the *system* directory of the case, the resulting volume mesh is written in the *polyMesh* directory inside the constant directory. The settings available in meshDict will be handled in the remainder of this section.

meshDict

cfMesh requires only two mandatory settings to start a meshing process: *surfaceFile* points to a geometry file, and *maxCellSize* that represent the default cell size used for the meshing job. It is the maximum cell size generated in the domain.

If uniform cell size is not satisfactory, refinement setting must be specified. there are many options for local refinement sources in cfMesh. *BoundaryCellSize* option is used for refinement of cells at the boundary. It is a global option and the requested cell size is applied everywhere at the boundary. *BoundaryCellSizeRefinementThickness* specifies the distance from the boundary at which the *boundaryCellSize* is still applied. *MinCellSize* is a global option which activates automatic refinement of the mesh template. This option performs refinement in regions where the cells are larger than the estimated feature size. The scalar value provided with this setting, specifies the smallest cell size allowed by this procedure. This option is useful for quick simulation because it can generate meshes in complex geometry with low user effort.

LocalRefinement allows for local refinement regions at the boundary. It is a dictionary of dictionaries and each dictionary inside the main *LocalRefinement* dictionary is named by a patch or facet subset in the geometry that is used for refinement. The requested cell size for an entity is controlled by the *cellSize* keyword and a scalar value, or by specifying *additionalRefine-*



```
// settings for boundary layers
boundaryLayers
{
    // global number of layers (optional)
    nLayers 3;

    // thickness ratio (optional)
    thicknessRatio  1.2;

    // max thickness of te first layer (optional)
    maxFirstLayerThickness  0.5; // [m]

    // local settings for individual patches
    patchBoundaryLayers
    {
        // patch name
        "patch7.*" // accepts regex
        {
            // number of layers (optional)
            nLayers 4;

            // thickness ratio (optional)
            thicknessRatio  1.1;

            // max thickness of the first layer
            // (optional)
            maxFirstLayerThickness  0.5; // [m]

            // active 1 or inactive 0
            allowDiscontinuity  0;
        }
    }
}
```

Figure 4.4: Example of boundary layer subdictionary.

mentLevels keyword and the desired number of refinements relative to the maximum cell size. It is possible to specify patches via regular expressions. The thickness of the refinement zone can be specified by the *refinementThickness* option.

Boundary layers in cfMesh are extruded from the boundary faces of the volume mesh towards the interior, and cannot be extruded prior to the meshing process. In addition, their thickness is controlled by the cell size specified at the boundary and the mesher tends to produce layers of similar thickness to the cell size. Layers in cfMesh can span over multiple patches if they share concave edges or corners with valence greater than three. All boundary layer settings are provided inside a boundaryLayers dictionary, shown in figure 4.4. The options are: *nLayers* specifies the number of layers which will be generated in the mesh, *thicknessRatio* o is a ratio between the thickness of the two successive layer, *maxFirstLayerThickness* ensures that the thickness of the first boundary layer never exceeds the specified value.

4.5 Numerical schemes used in this work

Here we explain which are the numerical schemes used in this work. In particularly, Crank-Nicolson for time derivative and finite volume method for spatial derviatives.

4.5.1 Crank-Nicolson method

Crank-Nicolson is a finite difference method second order implicit in time, so unconditionally stable, allowing significantly larger steps. This method uses the average of values at timestep n and $n+1$, for example, for a variable u resolved by a differential equation:

$$\frac{u_i^{n+1} - u_1^n}{\Delta t} = \frac{1}{2}\left[F_i^{n+1}\left(x, t, u, \frac{\partial u}{\partial x}, ...\right) + F_i^n\left(x, t, u, \frac{\partial u}{\partial x}, ...\right)\right] \tag{4.17}$$

this scheme requires very little more computational effort per step than the first order implicit Euler scheme. Von Neumann stability analysis shows that the scheme is unconditionally stable, but oscillatory solutions (and even instability) are possible for large time steps.

4.5.2 Spatial derivative discretization

OpenFOAM is a code that uses Finite Volume Method (FVM) [7, 32]. The FVM is a method for representing and evaluating partial differential equations in the form of algebraic equations. Similar to finite difference or finite element method, values are calculated at discrete places on a meshed geometry. In this method, spatial derivatives terms are first integrated over a cell volume V and then converted to integrals over the cell surface bounding the volume using Gauss's theorem:

$$\int_V \nabla \cdot \psi dV = \int_S \psi \cdot d\mathbf{S} = \sum_f \psi_f \tag{4.18}$$

where V is volume cell, S is surface cell, ψ_f is the quantity value at face center and f inn number of faces. It is important to remark that ψ_f depends on values of adjacent cell and on type of function f will be chosen:

$$\psi_f = f(\psi_P, \psi_N) \tag{4.19}$$

One of the most important aspects of finite volume method is that is a conservative method, or rather the fluxes of quantity which cross a finite volume must be the same of the quantity generated in the same volume.

In base of function f the scheme will be upwind, linear upwind linear, limited linear or other (fig 4.5) So the approximations to the integrals require the values of variables at locations other than computational nodes (CV centers). The quantity ψ involves the product of several variables and/or variable gradients at those locations ($\psi = \rho\phi\mathbf{U} \cdot \mathbf{n}$, for the convective flux or $\psi = \Gamma\nabla\phi \cdot \mathbf{n}$, for diffusive flux). If we assume that the fluid properties are known at all locations, to calculate the convective and diffusive flux, the value of ϕ and its gradient normal to the cell face are needed.

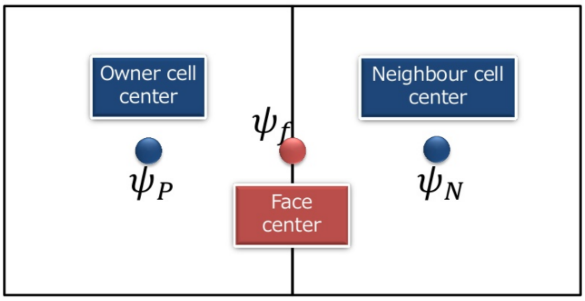


Figure 4.5: In base of function f the value of ψ_f can be ψ_P, ψ_N or an average between these two.

Upwind interpolation

One method is the upwind interpolation. Approximating ϕ_f by its value at the node upstream of f is equivalent to using a backward- or forward-difference approximation for the first derivative (depending on the flow direction), hence the name upwind differencing scheme (UDS) for this approximation. In UDS ϕ_f is approximated as:

$$\phi_f = \begin{cases} \phi_O \quad if \quad (\mathbf{U}\cdot\mathbf{n})_f > 0 \\ \phi_N \quad if \quad (\mathbf{U}\cdot\mathbf{n})_f > 0 \end{cases} \tag{4.20}$$

using this method, oscillation in solution are not present, but the defect is that, if the cell are not aligned with the flux, will be generated an extra numerical diffusive flux. The numerical diffusion is magnified in multidimensional problems if the flow is oblique to the grid; the truncation error then produces diffusion in the direction normal to the flow as well as in the streamwise direction, a particularly serious type of error. Peaks or rapid variations in the variables will be smeared out and, since the rate of error reduction is only first order, very fine grids are required to obtain accurate solutions [7].

Linear interpolation

Another straightforward method is linear interpolation between the two nearest nodes. At location f on a Cartesian grid we have:

$$\phi_f = \phi_N\lambda + \phi_O(1-\lambda) \tag{4.21}$$

where the linear interpolation factor λ is defined as:

$$\lambda = \frac{x_f - x_O}{x_N - x_O} \tag{4.22}$$

Equation 4.21 is second-order accurate as can be shown by using the Taylor series expansion of ϕ_N about point x_O to eliminate the first derivative:

$$\phi_f = \phi_N\lambda + \phi_O(1-\lambda) - \frac{(x_f - x_O)(x_N - x_f)}{2}\left(\frac{\partial^2\phi}{\partial x^2}\right) + H \qquad (4.23)$$

where H denotes higher order terms. The leading truncation error term is proportional to the square of the grid spacing, on uniform or non-uniform grids. As with all approximations of order higher than one, this scheme may produce oscillatory solutions. This is the simplest second-order scheme and is the one most widely used. The assumption of a linear profile between the O and N nodes also offers the simplest approximation of the gradient, which is needed for the evaluation of diffusive fluxes:

$$\left(\frac{\partial\phi}{\partial x}\right) \approx \frac{\phi_N - \phi_O}{x_N - x_O} \qquad (4.24)$$

By using Taylor series expansion around ϕ_N, one can show that truncation error of the above approximation is:

$$\epsilon_r = \frac{(x_f - x_O)^2 - (x_N - x_f)^2}{2(x_N - x_O)}\left(\frac{\partial^2\phi}{\partial x^2}\right) - \frac{(x_f - x_O)^3 + (x_N - x_f)^3}{6(x_N - x_O)}\left(\frac{\partial^3\phi}{\partial x^3}\right) + H \qquad (4.25)$$

When the location f is midway between O and N (for example on a uniform grid), the approximation is of second-order accuracy, since the first term on the right-hand side vanishes and the leading error term is then proportional to $(\Delta x)^2$. When the grid is non-uniform, the leading error term is proportional to the product of Δx and the grid expansion factor minus unity.

4.6 Numerical linear solver used in this work

After we have discrezitized the partial differential equations in space and time, we have to solve an algebraic system. Here we present the linear solvers used in this work. In particularly, we have used GAMG (Geometric Agglomerated Algebraic Multigrid Solver) for the pressure and smooth solver using Gauss-Seidel method for velocity, k and ω.

GAMG (Geometric Agglomerated Algebraic Multigrid Solver)

The basic idea behind multi-grid solvers is to use a coarse grid with fast solution times to smoothen out high frequency errors to generate a starting solution for the finer grid. This can either be done be a geometric coarsening of the grid (geometric multi-grid), or by applying the same principles dirctly to the matrix, regardless of the geometry (algebraic multi-grid). Inside GAMG the mesh is coarsed in steps and the coarsening or agglomeration

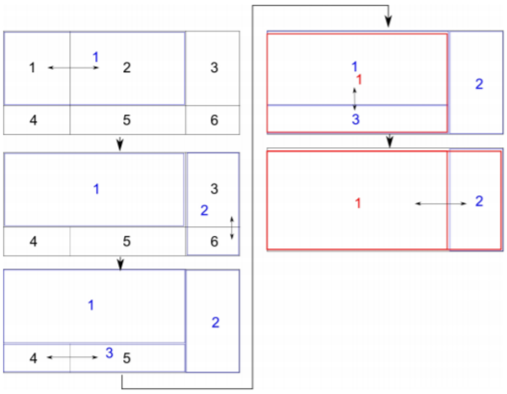


Figure 4.6: Example of geometric agglomeration process.

algorithm can be faceAreaPair (geometric) or algebraic pair. The loop for the solution is:

1. Get the finest level interfaces from the mesh.
2. Start agglomeration from the given faceWeights:
   - a) For each cell calculate faces, afterwards go through the faces and create groups or clusters.
     - Check faces to find ungrouped neighbour with largest face weight
     - Check if current cell is face owner or neighbour
     - When a match is found, pick up all the necessary data and generate a new group or cluster, elas find the best neighnbouring cluster and add the cell to it
   - b) Check that all cells are part of clusters, not create a single cell cluster for each
   - c) Reverse the map ordering to potentially improve the next level of agglomeration
3. Agglomerate the faceWeights field for the next level and continue unless the user specified approximate mesh size at the most coarse levelor maximum of grid levels is reached.

In figure 4.6 a simple geometric example for this process is shown. Starting with the black grid, consisting of six cells, the coarser mesh (blue) is built by joining two cells for each new cell. For the coarsest level (red) first cells 1 and 3 are joined. Cell 2 cannot find an ungrouped partner and will be joined with the neighbouring group.

4.6.1 Gauss-Seidel method

In numerical linear algebra, the Gauss–Seidel method [36], also known as the Liebmann method or the method of successive displacement, is an iterative method used to solve a linear system of equations. It can be applied to any matrix with non-zero elements on the diagonals, convergence is only guaranteed if the matrix is either diagonally dominant, or symmetric and positive definite. The Gauss–Seidel method is an iterative technique for solving a square system of n linear equations with unknown x:

$$A\mathbf{x} = \mathbf{b} \tag{4.26}$$

where:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

and $\mathbf{x}$ and $\mathbf{b}$ are vector of n dimension. It is defined by the iteration:

$$L_*\mathbf{x}^{k+1} = \mathbf{b} - U\mathbf{x}^k \tag{4.27}$$

where A is decomposed in A_* in a lower triangular component (L_*) and in a upper triangular component (U). The Gauss–Seidel method now solves the left hand side of this expression for $\mathbf{x}$, using previous value for $\mathbf{x}$ on the right hand side. Analytically, this may be written as:

$$\mathbf{x}^{k+1} = L_*^{-1}(\mathbf{b} - U\mathbf{x}^k) \tag{4.28}$$

Taking advantage of the triangular form of L_*, the elements of x^{k+1} can be computed sequentially using forward substitution:

$$x_i^{k+1} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum_{j=1+1}^{n} a_{ij}x_j^k\right), \qquad i = 1, 2, ..., n \tag{4.29}$$

the procedure is generally continued until the changes made by an iteration are below some tolerance, such as a sufficiently small residual.

4.7 Algorithm of mesh morphing

OpenFOAM uses SLERP (Spherical Linerar intERPolation) algorithm for the purpose of moving mesh. It refers to constant-speed motion along a unit radius circle arc, given the ends and an interpolation parameter between 0 and 1. SLERP has a geometric formula independent of the dimension of the space in which the arc is embedded. This formula is based on the fact that any point on the curve must be a linear combination of the ends:

$$Slerp(p_0, p_1, t) = \frac{\sin[(1-t)\Omega]}{\sin\Omega} p_0 + \frac{\sin(t\Omega)}{\sin\Omega} p_1 \tag{4.30}$$

let p_0 and p_1 be the first and last points of the arc, and let t the parameter from 0 to 1 and Ω is the angle subtended by the arc. We can see that in the limit of Ω tending to zero the formula corresponds to a linear interpolation. SLERP is a non-commutative method, or rather the order of how the vectors/matrices are passed will affect the result.

4.8 Q-criterion

In flapping energy harvester, it is very important to visualize the vortex released by the device, so using Paraview [31] tools it is possible to show Q-criterion [11]. Q-criterion proposes to not create dis-ambiguity in the definition of vortex, because this method exploit theory independent by reference system. Indeed, in this way vortex it is not intended as a region where vorticity is not zero or where streamlines are closed, which are affected by reference system, but a region where a determined quantity named Q is major of zero:

$$Q = \frac{1}{2}[||\mathbf{\Omega}||^2 - ||\mathbf{S}||^2] \tag{4.31}$$

where $\mathbf{S} = \frac{1}{2}[\nabla\mathbf{U} + (\nabla\mathbf{U})^T]$, is the rate of strain tensor, and $\mathbf{\Omega} = \frac{1}{2}[\nabla\mathbf{U} - (\nabla\mathbf{U})^T]$ is the vorticity tensor. So we have a vortex when the vorticity magnitude dominates the magnitude of rate of strain.

Chapter 5

Benchmark

In this chapter we will deal with the validation process of OpenFOAM, to verify its reliability. We have chosen first to simulate a bi-dimensional vortex induced vibration case. We have compared our results with referenced articles [29, 28], where a discrete vortex method is used, and with the Overture CFD code [12], that has been another candidate for studying FLEHAP device, on which preliminary studies have been performed. This code uses a fourth order difference method on overlapping grids [14].

On the other hand, a test has been performed regarding the main 3D simulation case when the fluid is at rest, in order to evaluate the natural frequency of the device, compared to the experiment carried out in wind tunnel.

5.1 Vortex Induced Vibration

The case that has been chosen to verify the reliability of the code is about the vortex induced vibration (VIV) phenomenon on a cylinder [24]. The flow induced vibration of an elastically constrained two-dimensional circular cylinder has become a canonical problem in the efforts to understand more general situations of fluid structure interactions. The situation in which vibrations are permitted only transversely to the freestream has received particular attention as a fundamental case. Relevant structural parameters are the system mass m (and associated density ρ_b), spring constant k and diameter D. Important fluid parameters include the fluid density ρ, kinematic viscosity ν, and constant freestream velocity U_∞. The cylinder is constrained to move only normal to the freestream; its transverse motion is determined by the lift generated from the flow. The vibrations occur when the body's natural frequency is close to the vortex shedding frequency. It is a phenomenon that does not verify for all flow conditions, but only for a given set of dimensionless parameters, this state termed "lock-in". It has been observed that there is VIV for a limited range of reduced velocity

$U_R = 0.6 \div 2$ ($U_R = U_0/c$, where U_0 is the undisturbed flow velocity and c is the propagation wave speed in the solid) and then disappears. The main dimensionless parameter for this study is the Strouhal number:

$$St = \frac{T_{fluid}}{T_{vortex}},$$

where

$$T_{Fluid} = \frac{L}{U_0}, \quad T_{vortex} = \frac{\Lambda}{U_0} \tag{5.1}$$

where L is the length scale and Λ is the vortex wave length. We can define Strouhal's number starting from definition of solid time scale and reduced velocity:

$$\frac{T_{Solid}}{T_{vortex}} = StU_R, \quad \text{if} \quad StU_R = 1 \quad \Rightarrow \quad T_{Solid} = T_{vortex}, \quad T_{Solid} = \frac{1}{2\pi}\sqrt{\frac{M}{k}}$$

where T_{Solid} is the ratio between the solid length scale and c (speed of wave propagation in the solid). It is important to remark that VIV occurs when $StU_R \approx 1 \div 2$. This latter relation is always valid for a fixed body, instead for a moving solid there will be a deviation from this linear law. The phenomenon can be explained easily: we have oscillations of the flow at the period T_{vortex}, these cause an oscillating force acting on surface cylinder at the same period, if the flow fluctuations are equal to natural frequencies of the solid, we have a resonance state; here is a simple relation of this effect for a body having length L:

$$F_{vortex}(t) = \frac{1}{2}\rho U^2 L C_l sin\left(2\pi \frac{t}{T_{vortex}}\right) \tag{5.2}$$

If the cylinder is a mass spring system allowed to move only in vertical direction, we can write down a forced oscillator equation:

$$m\ddot{y} + ky = \frac{1}{2}\rho U^2 L C_l \sin\left(2\pi \frac{t}{T_{vortex}}\right) \tag{5.3}$$

The response of Eq.(5.3) is analytically known and is equal to:

$$Y = \frac{MC_l}{2\pi^3}\frac{U_R^2}{(1 - St^2 U_2)}, \quad Y = \frac{y}{L}, \quad M = \frac{\rho_f \pi L^2}{4m} \tag{5.4}$$

Fig. 5.1 shows that the amplitude does not go to zero after the resonance, but stays on a plateau, because increasing the speed also the frequency and magnitude of the forces are increased.

In case of cylinder which is free to move the frequency of vortex shedding deviates from linear law of Strouhal's number, as we can see in Fig. 5.2.

if we increase flow velocity above the point where there is resonance, the frequency of shedding is said to lock on the frequency of oscillation of

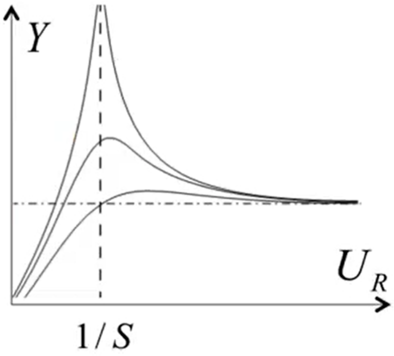


Figure 5.1: Dynamic response of a forced oscillator equation in dimensionless form.

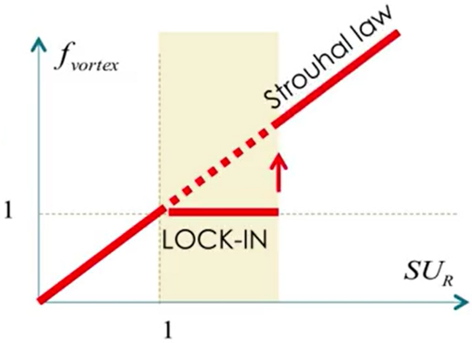


Figure 5.2: Lock in phenomenon and Strouhal law for cylinder free to move.

cylinder. It strongly deviates from Strouhal's law, being locked to cylinder fluctuations. Even for higher reduced velocity f_{vortex} jumps back to the Strouhal's law. The lock in phenomenon is like an extended resonance, the wake continues to excite the cylinder at its own frequency even at higher reduced velocity, so we have VIV for a wide range of U_R, even after the condition of pure resonance between the fluid motion and the solid.

5.2 Benchmark VIV case

The performed simulations aim to reproduce the VIV phenomena of a cylinder, comparing first the result obtained by Overture [5] with other two well referenced work of Shiels [29] and Shen [28]. Then we will compare OpenFOAM results with Overture and Shiels. For OpenFOAM case, the grid has been built up with *blockMesh*, and with *Ogen* [13], for Overture. The cell dimension has been built in agreement with the articles, mentioned above.

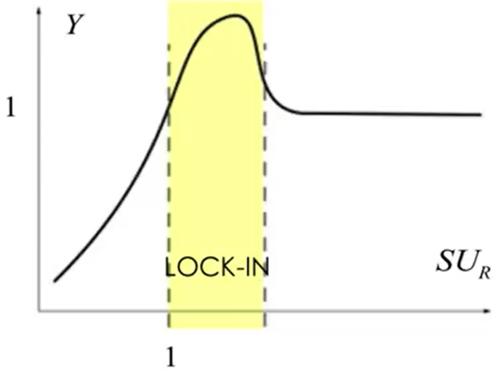


Figure 5.3: The plot shows that amplitude still increase after resonance condition.

We have tested coarser grids but did not give acceptable results. In Fig. 5.5 is shown the computational domain with its relative boundary conditions.

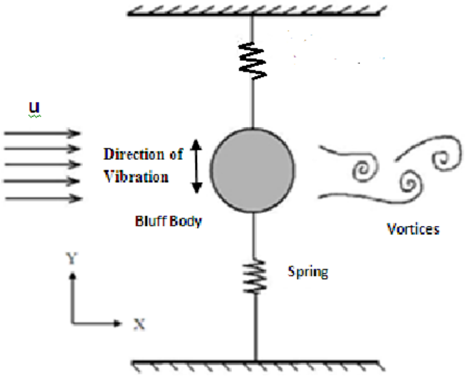


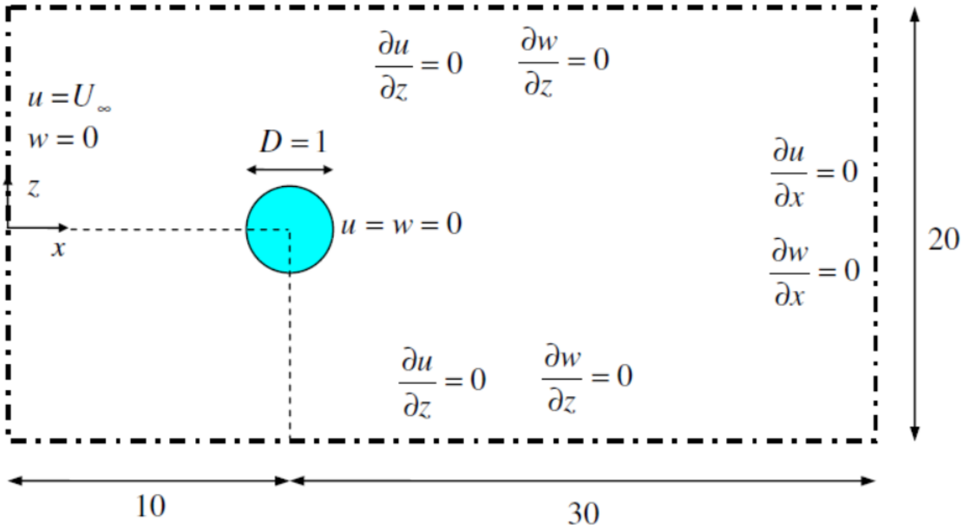
Figure 5.4: Model used to simulate VIV phenomenon on a cylinder, bonded with two springs of equal stiffness [28].

Transverse oscillation of a dynamically supported circular cylinder in a flow at $Re = 100$ has been numerically simulated using a direct numerical simulations, for two values of reduced velocity ($U_R = 0.71$ and $U_R = 1$). The dimensionless quantities used for this analysis are:

$$A^* = \frac{y}{D}, \qquad m^* = \frac{2m}{\rho D^2}, \qquad k^* = \frac{2k}{\rho U_\infty^2}, \qquad St = \frac{fD}{U_\infty}$$

where A^* is dimensionless amplitude, m^* dimensionless mass, k^* dimensionless stiffness ad St is Strouhal number.

We will report results for Strouhal number (St), the ratio between the vortex shedding frequency (f) and natural frequency of the system f_n and



$\frac{\partial u}{\partial z}=0$
$\frac{\partial w}{\partial z}=0$
$u=U_\infty$
$w=0$
$D=1$
z
x
$u=w=0$
$\frac{\partial u}{\partial x}=0$
$\frac{\partial w}{\partial x}=0$
20
$\frac{\partial u}{\partial z}=0$
$\frac{\partial w}{\partial z}=0$
10
30

Figure 5.5: Computational domain for the flow around a circular cylinder together with the associated parameters and boundary conditions (taken by [29]).

	U_R	St	f/f_n	A^*
Present work (Overture)	0.71	0.192	0.856	0.57
Shiels et al. [29]	0.71	0.196	0.874	0.58
Shen et al. [28]	0.71	0.19	0.847	0.57

Table 5.1: Table for Overture validation. Comparison of VIV results for $Re = 100$ at $U_r = 0.71$ $k^* = 4.96$ and $m^* = 2.5$

.

dimensionless transversal amplitude of motion (A^*). In Tab. 5.1 we have reported results for the same value of $U_R = 0.71$ for checking Overture reliability.

For concluding bi-dimensional benchmark analysis, we have conducted a final simulation at $U_R = 1$ and $m^* = 20$ to test OpenFOAM reliability. We know from theory that for reduced velocity equal to 1 the natural frequency is the same of vortex shedding frequency. From Tab. 5.2 we can see that all the quantities differ less than 4% of OpenFOAM values. In Fig. 5.6 we have reported Shiels and OpenFOAM result.

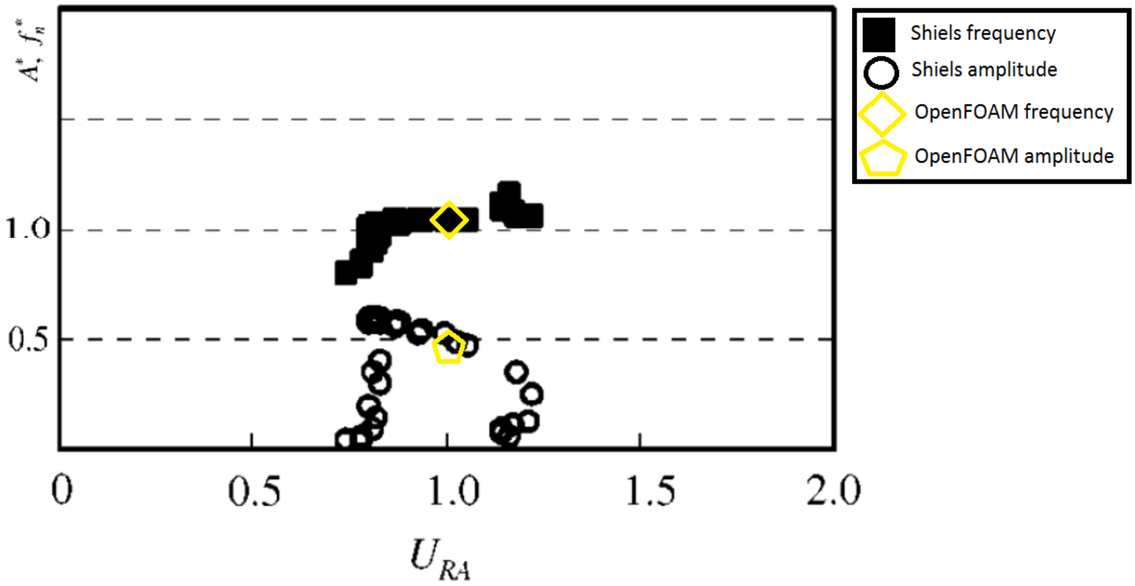


Figure 5.6: Benchmark between OpenFOAM and Shiels results of vortex shedding frequency on natural frequency (f_n^*) and dimensionless amplitude for $U_r = 1$(adapted by [29]).

	U_R	St	f/f_n	A^*
Present work (OpenFOAM)	1	0.159	1.002	0.44
Present work (Overture)	1	0.161	1.01	0.46

Table 5.2: Comparison between OpenFOAM and Overture. VIV result for $U_r = 1$, $k^* = 20$ and $m^* = 20$ at $Re = 100$.

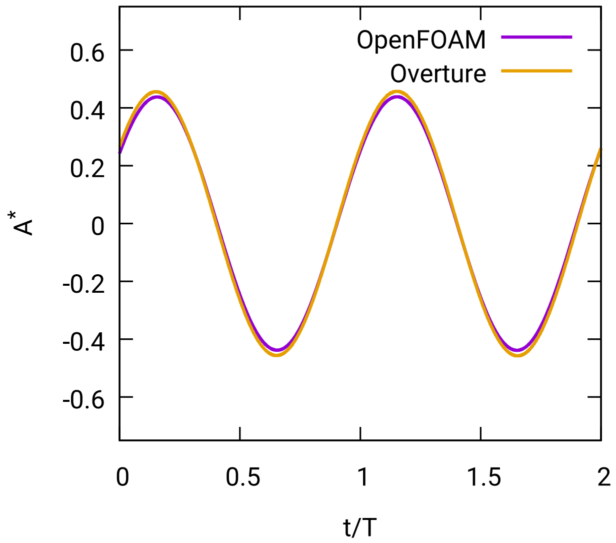


Figure 5.7: Benchmark between OpenFOAM dimensionless amplitude of motion and Overture. In blue line OpenFOAM and in orange line Overture.

	f_n $[Hz]$
Present work (OpenFOAM)	13.51
Experimental data	13.2

Table 5.3: Benchmark between values of natural frequency computed by OpenFOAM and experiment.

5.3 3D test

After this 2D validation, we will set up a 3D simulation with OpenFOAM. The test consist of simulating the wing used in this work, anchored by elastomers, oscillating in stationary flow, under initial vertical velocity perturbation of 1 m/s towards down and with only the vertical displacement as allowed degree of freedom. We will compare the experiment with numerical result, in particular we will investigate the natural frequency of the device. The elastomer damping is set equal to 0.2 Ns/m The natural frequency is $f_n = 1/2\pi\sqrt{K_{(y)}^{\text{eff}}/m}$ where $K_{(y)}^{\text{eff}}$ is equivalent elastomer stiffness and m is wing mass. In the Tab. 5.3 we report the frequency evaluated by experiment and our numerical result. As we can see, the difference between the two frequency is 2%. In Fig. 5.8 we have reported the vertical coordinate of pivot point during the numerical simulation, which is damped in time.

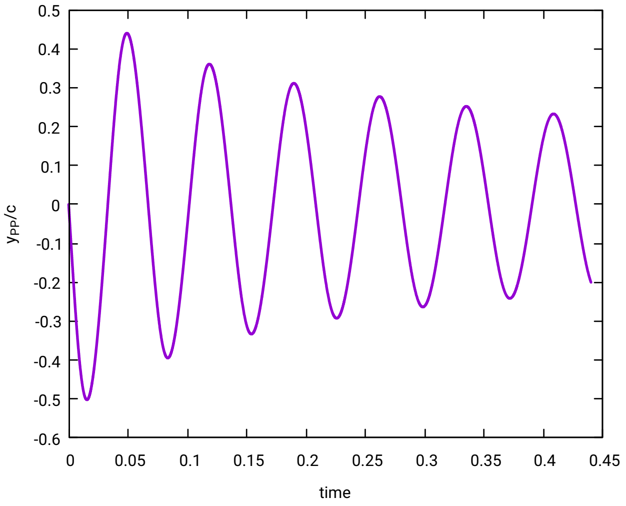


Figure 5.8: Trend of vertical coordinate of pivot point. After initial perturbation the wing start to oscillate at natural frequency. The amplitude is damped in time.

Chapter 6

Results

In this chapter we will show the numerical results obtained by OpenFOAM on flapping wing. First of all we have done preliminary simulations for wind velocities from 2.5 m/s to 5.5 m/s, proceeding to step of 0.5 m/s. We will analyse only the case for 4 m/s, for the which a numerical damping analysis and a grid dependency have been made, through RANS $k - \omega\ SST$ simulation. Then we have done a force analysis on the wing, which has been set up with LES simulation, using Smagorinski model.

6.1 Parametric study on the wind speed

In order to characterize the device operation, it has been investigated several wind velocities, monitoring the trajectory of pivot point (PP) and trailing edge (TE), the frequency and the phase between pitch and plunge motion. We have compared the numerical results with experimental data and phenomenological model solution. From the results obtained (Fig. 6.1 and 6.2), we can identify three distinct operational phase. The first is named the pre-transition phase, where the PP amplitude is greater than the TE amplitude. The second is named the transition phase where the two amplitudes are comparable and third one is named the post-transition phase where the TE amplitude is bigger than the PP amplitude. The operational phase can be identified looking at phase diagram, in particular if ϕ is under 20° the system is in the pre-transition, if ϕ is about 20° the system is in the transition, and if the phase is near to 90° the system is in post-transition. We can also noting that the phase increases abruptly after 4 m/s. The results show that the CFD simulations are in trouble in the transition region, indeed all the quantities monitored are far from experimental data, while for high and low wind speed velocity there is a substantial agreement. On the contrary the phenomenological model seems to be in agreement with experimental data, despite of its simplicity. A final remark can be made on trajectories. CFD and phenomenological model give results quite similar to experiment, however we can notice that CFD anticipates the transition phase, indeed the TE amplitude is bigger than the PP amplitude already at 4 m/s.

The most critical case is that for $U = 4$ m/s, indeed the frequency, the dimensionless amplitude and the phase between pitch and plunge differ most from experimental data. On the basis of these result, we have chosen to improve the numerical set up for the case $U = 4$ m/s, checking the influence of the elastomer damping and the grid.

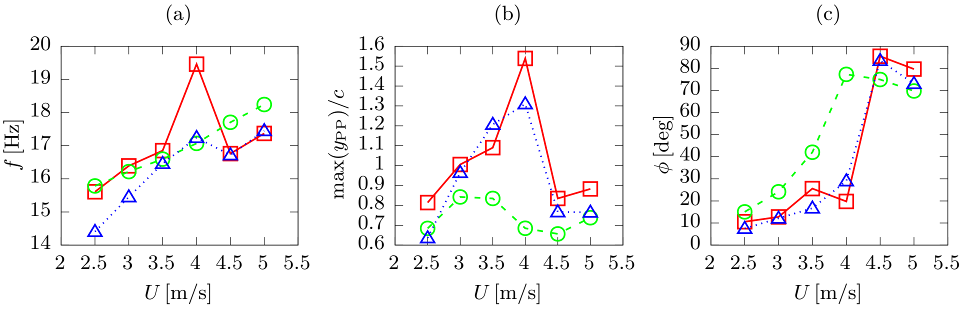


Figure 6.1: Flapping frequency (a), dimensionless amplitude (b) and phase between pitch and plunge (c), from experiments (squares filled line), simulations (circles dashed line) and phenomenological model (triangles dotted line), for wind speed U=(2.5,3.0,3.5,4.0,4.5,5.0,5.5) m/s.

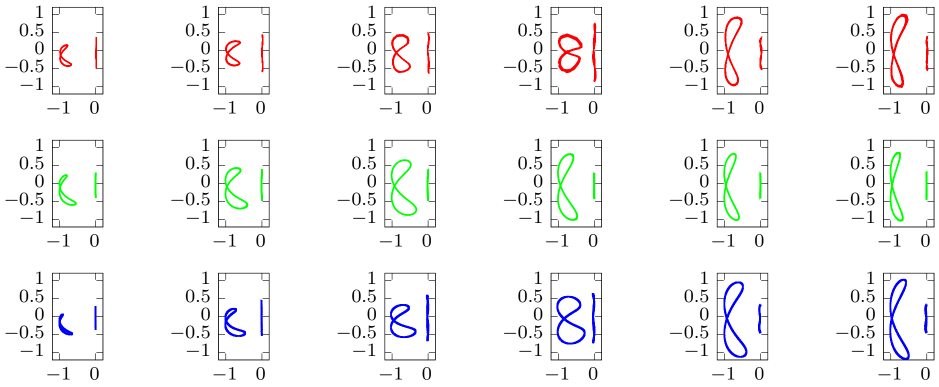


Figure 6.2: Pivot point and trailing edge trajectories in the $x - y$ plane from experiment (first row), CFD simulation (second row), and phenomenological model (third row), for wind speed U=(2.5,3.0,3.5,4.0,4.5,5.0,5.5) m/s (from left to right. Lengths are normalized with the chord).

	f [Hz]	A^*	ϕ [°]
$\zeta = 0.2$	17.96	0.61	78.68
$\zeta = 0.3$	16.32	0.57	78.12
$\zeta = 0.5$	16.24	0.52	77.52

Table 6.1: Numerical results for frequency, dimensionless amplitude and phase for each grid.

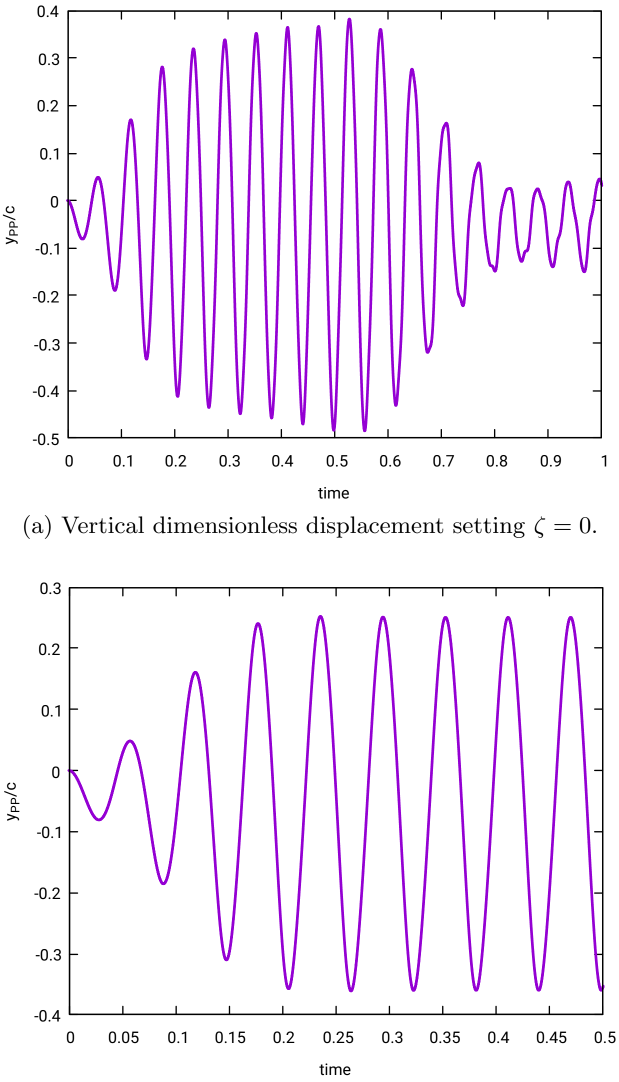
6.2 Focus on U=4 m/s

6.2.1 Sensitivity on numerical damping

We have done this analysis because it is emerged that resolving the equations using only the elastic force showed in Sec. 2.2, we did not obtain a physical solution, or rather something that does not present a limit cycle of oscillation. From Fig. 6.3a we can see the vertical pivot point coordinate trend in time, where damping term (ζ [Ns/m]) was set equal to zero and in Fig. 6.3b a case for $\zeta = 0.2$. The first one is a trend that has never been observed in the wind tunnel and the second one present a limit cycle of oscillation. From experiments, we know that after a period of transitory we have a sustained flapping regime, with constant amplitude for this parameters set, so if we do not get it the numerical result is unrealistic. The issue is find out the damping threshold value, which allows to obtain a limit cycle of oscillation. The ζ threshold has been found equals to 0.2. Therefore the damping setting needs a careful tuning to set up in the best way the simulation, and the value which has been chosen will not represent the real physical damping (evaluated around at 1.4×10^{-3} Ns/m) but will assume the meaning of numerical gimmick. We have tested three values of ζ equals to 0.2, 0.3, 0.5, to verify how the solution varies changing the damping value. We have done all these simulations on c_1 grid (see Sec. 6.2.2 for furthermore information on it) because it is the coarser mesh and guarantees the fastest time of simulation, without loss of accuracy. We will show how the amplitude motion, the phase between pitch and plunge, and the frequency vary as a function of damping value. We have reported the analysis results in Tab. 6.1, showing how the frequency, dimensionless plunge amplitude and phase between pitch and plunge movement vary with ζ. We can see that the frequency and the dimensionless amplitude decrease increasing damping, while the phase remains substantially unchanged.

6.2.2 Grid dependency

We have conducted a grid dependency analysis, as usual in CFD. We have tested three meshes with a progressive refinement level from the coarser (c_1) to the finest (c_4). The grid topology is composed by a big rectangu-



(a) Vertical dimensionless displacement setting $\zeta = 0$.

(b) Vertical dimensionless displacement setting $\zeta = 0.2$.

Figure 6.3: Plot of dimensionless vertical displacement of pivot point.

	Δ_{max}	Δ_{box_1}	Δ_{box_2}	Δ_{box_3}	Δ_{box_4}	$N°$cells
c_1	$2c$	c	$0.5c$	$0.25c$	$0.125c$	57 680
c_2	c	$0.5c$	$0.25c$	$0.125c$	$0.0625c$	409 575
c_4	$0.5c$	$0.25c$	$0.125c$	$0.0625c$	$0.03125c$	2 994 635

Table 6.2: Refinement of the meshes used for grid dependency. The cell dimension is referred to chord length c.

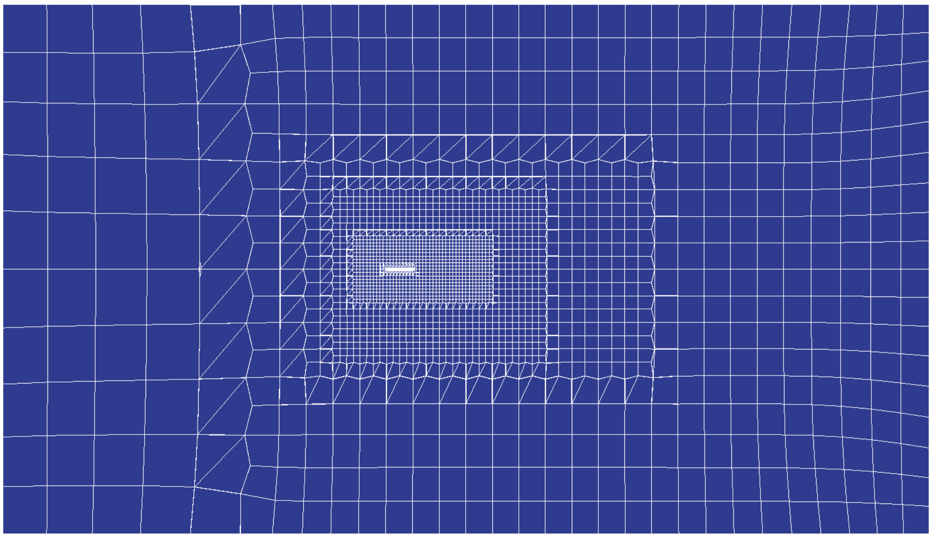
	f [Hz]	A^*	ϕ [°]
c_1	17.09	0.61	78.68
c_2	17.21	0.57	70.44
c_4	17.06	0.59	68.18
Experimental data	19.42	1.54	19.8

Table 6.3: Numerical results for frequency, dimensionless amplitude and phase for each grid.

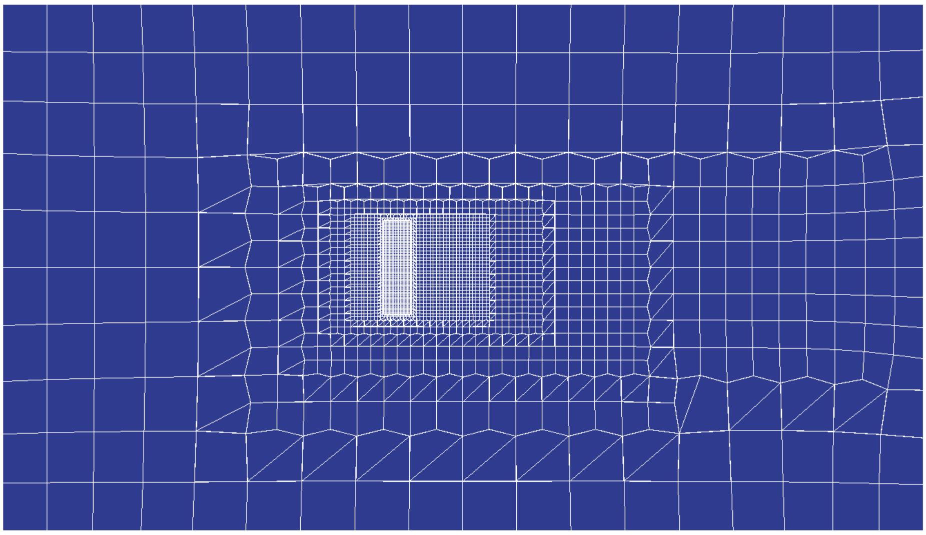
lar box, with coarser refinement, that represents wind tunnel of dimension $0.7m \times 0.4m \times 0.4m$, the device is positioned at $0.3m \times 0.2m \times 0.2m$. The computational domain is composed by a coarse base mesh, which has the dimension of wind tunnel, and refinement boxes which have following dimensions: box_1 has dimension $0.45 \times 0.25 \times 0.15$, centred in $(0.125, 0, 0)$ and half cell size of base mesh, box_2 has dimension $0.25 \times 0.15 \times 0.12$, centred in $(0.06, 0, 0)$ and half cell size of box_1, box_3 has dimension $0.15 \times 0.12 \times 0.10$, centred in $(0.02, 0, 0)$ and half cell size of box_2, box_4 has dimension $0.1 \times 0.05 \times 0.08$, centred in $(0.02, 0, 0)$ and half cell size of box_3. In tab 6.2 are summarized all mesh parameters and figure is shown the coarser grid.

The grid dependency analysis has been led out checking the frequency, the dimensionless plunge amplitude of pivot point and phase between pitch and plunge for each grid, for a fixed value of elastomer damping ($\zeta = 0.2$).

In Tab. 6.3 we can see the values carried out by the simulations, while in Fig. 6.5 we can appreciate the quantities trend as a function of the grid. The dimensionless amplitude has not a monotonous trend, but the value tends to stabilize increasing mesh refinement, indeed the difference between c_1 and c_2 is of 7%, while for c_2 and c_4 is 3.5%. The phase has a trend that decreases with the mesh refinement. The frequency is virtually the same for the c_1 and c_4 grid, while c_2 differs of 4% compared to the other two. However, neither setting the threshold value of nor adopting a finer mesh, we cannot reach the experimental data values. We can conclude that, for studying the case of 4 m/s, it is not sufficient set the threshold damping value and adopt a finer mesh, but it will necessary make changes in physical model. We will choose the c_4 grid to make a force and vortex analysis on



(a) $x - y$ plane at $z = 0$.



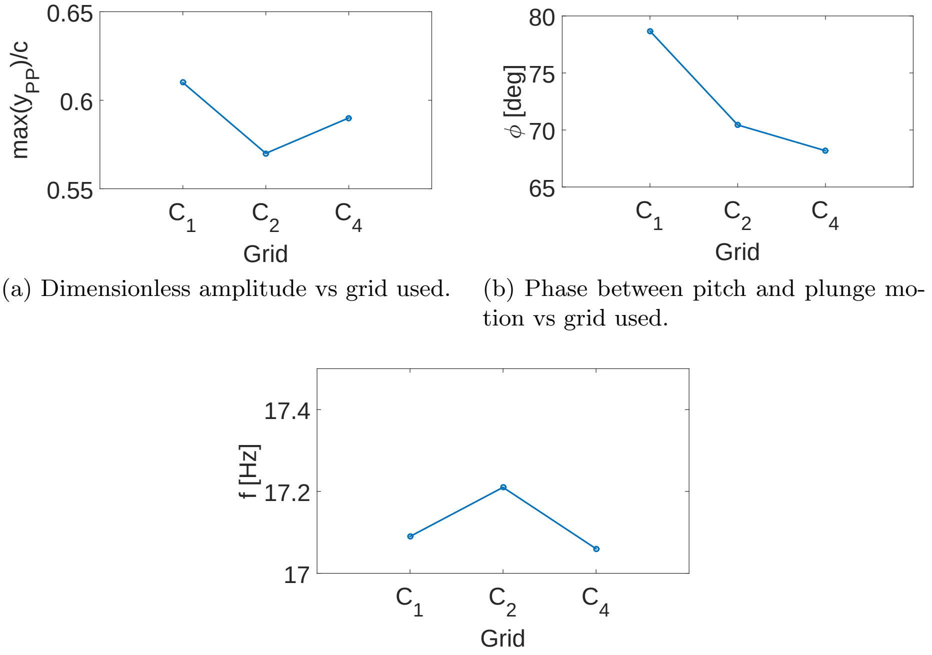
(b) $x - z$ plane at $y = 0$.

Figure 6.4: Representation of c_1 grid.

the wing, so as to better display the flow.

6.2.3 Aerodynamic coeffiecients

In this section we will analyse the force acting on the wing, using the lift, drag and pitching moment coefficient trend in a limit cycle oscillation (Fig.



(a) Dimensionless amplitude vs grid used.

(b) Phase between pitch and plunge motion vs grid used.

(c) Frequency vs grid used.

Figure 6.5: Graphs of dimensionless amplitude, phase between pitch and plunge motion and frequency as a function of the grid used.

6.6). We will also show the correlation between vortex dynamics computed by OpenFOAM, through z component of vorticity and pressure snapshot, with the frames flow, taken by a digital camera in wind tunnel. The images from experiment are referred to a previous wing configuration, with the chord equal to 15 mm instead of 20 mm. For set up problems, it has not been possible reproduce correctly the wind tunnel experiments with the wing chord equal to 20 mm. The comparison between the experiment and numerical simulation (Fig. 6.7 to Fig. 6.16) shows that the vortex movement are similar while the pitch angle is not the same in every frame, because of the different wing geometry. Qualitatively we can conclude that the flow captured by numerical simulation is in agreement with experimental results. The force analysis will be explained using Fig. 6.6, which represents a limit oscillation cycle, and Fig. 6.7 to 6.16, starting from pitch angle (α) equal to zero. Starting to analyse the lift coefficient (Fig. 6.6 purple line), we can see that the lift increases until t/T=0.1 (Fig. 6.7) because there is the leading edge counter clockwise vortex which is growing, creating a region of increasingly large suction. After this time the lift goes down due to vortex shedding, doing stalling the wing (Fig. 6.9). The lift has a little plateau at t/T=0.3 because of there is the presence of a secondary leading edge

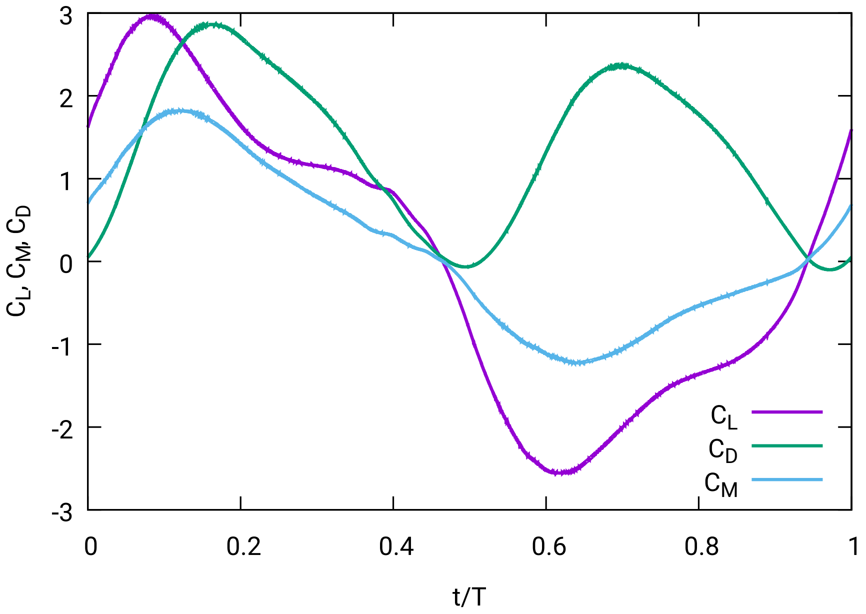
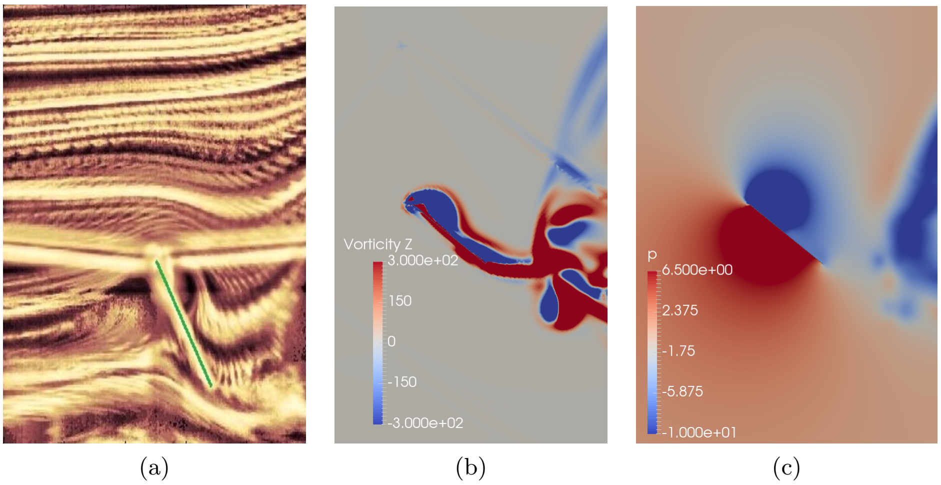


Figure 6.6: Trend of aerodynamic coefficients in a period.

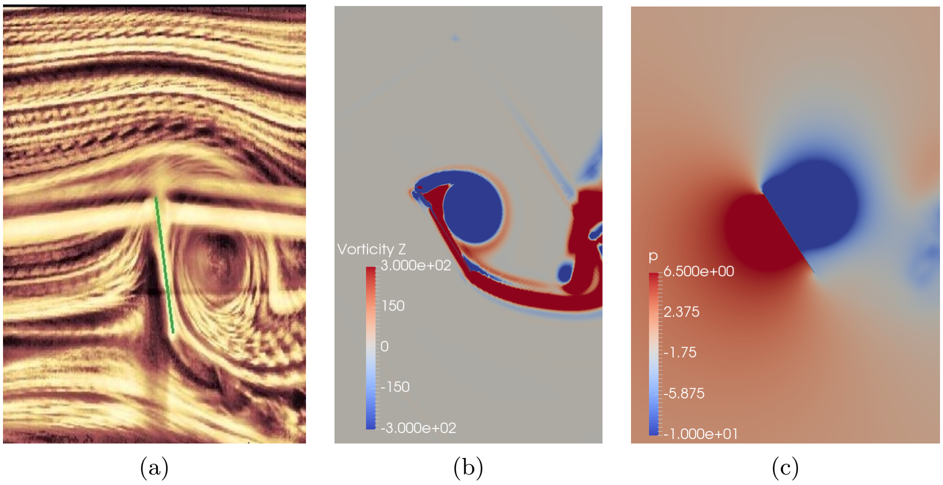


(a)

(b)

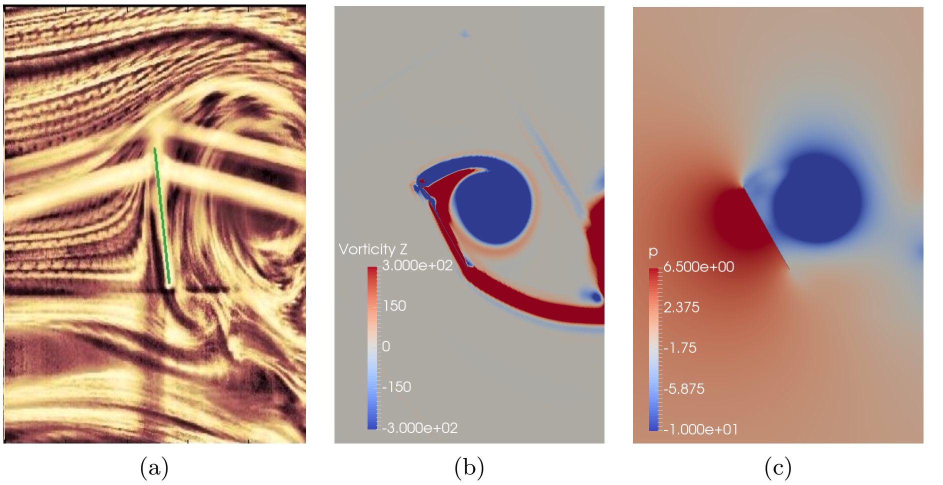
(c)

Figure 6.7: Snapshot from t/T=0.1



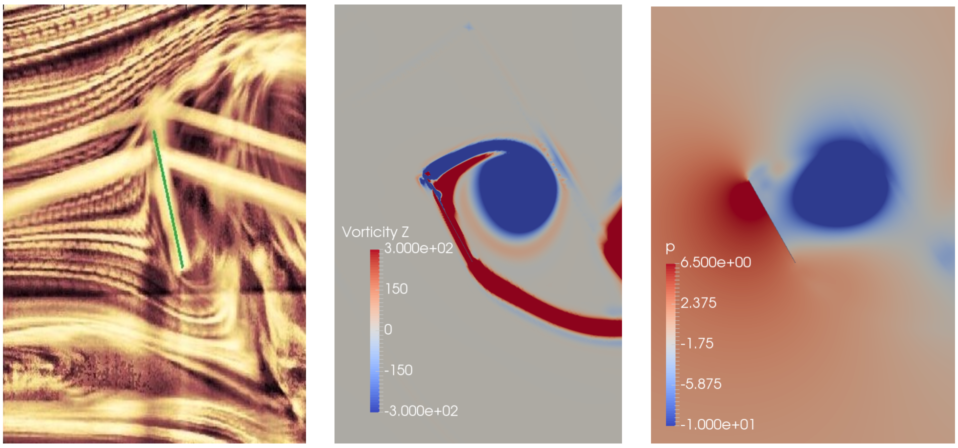
(a) (b) (c)

Figure 6.8: Snapshot from t/T=0.2



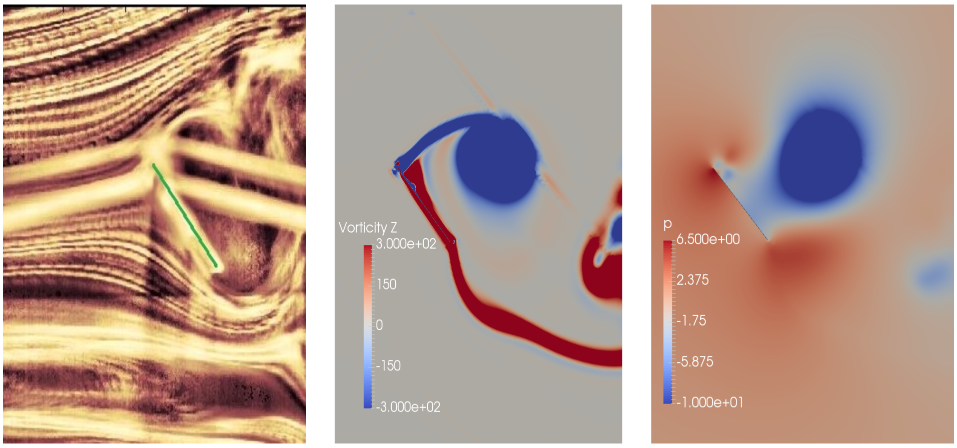
(a) (b) (c)

Figure 6.9: Snapshot from t/T=0.3



(a) (b) (c)

Figure 6.10: Snapshot from t/T=0.4



(a) (b) (c)

Figure 6.11: Snapshot from t/T=0.5

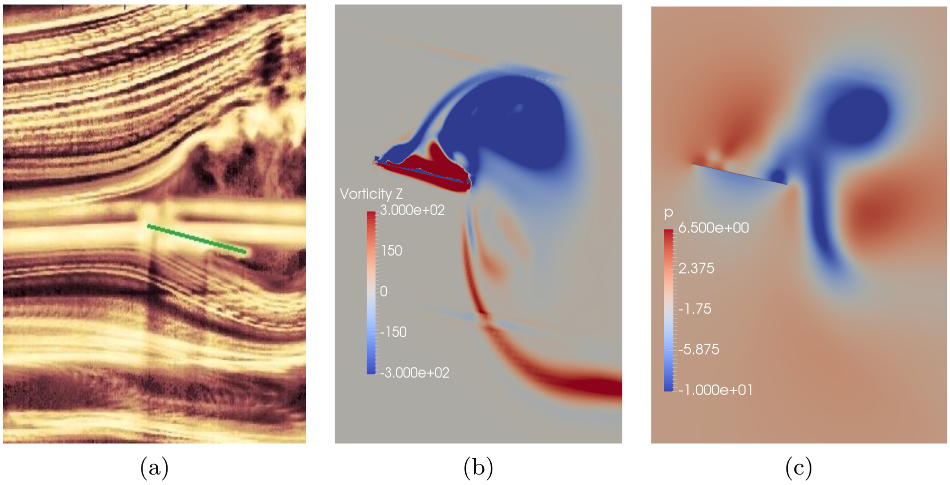


Figure 6.12: Snapshot from t/T=0.6

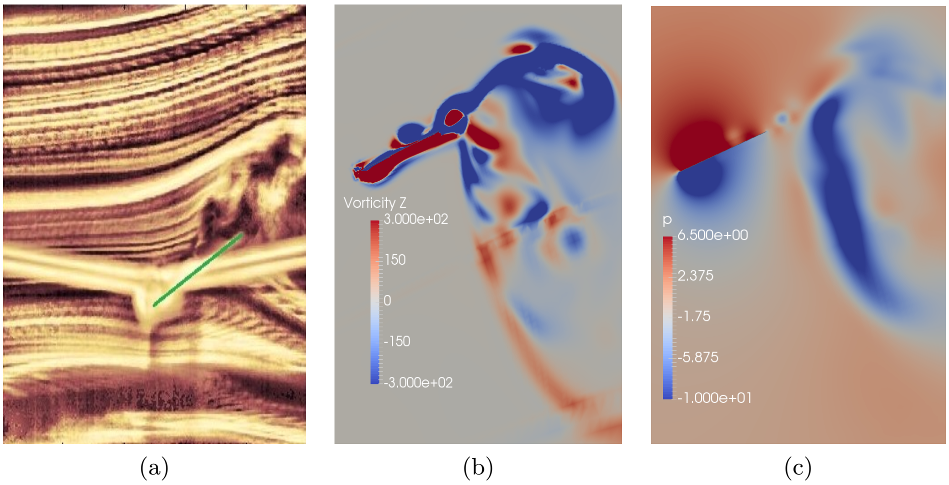
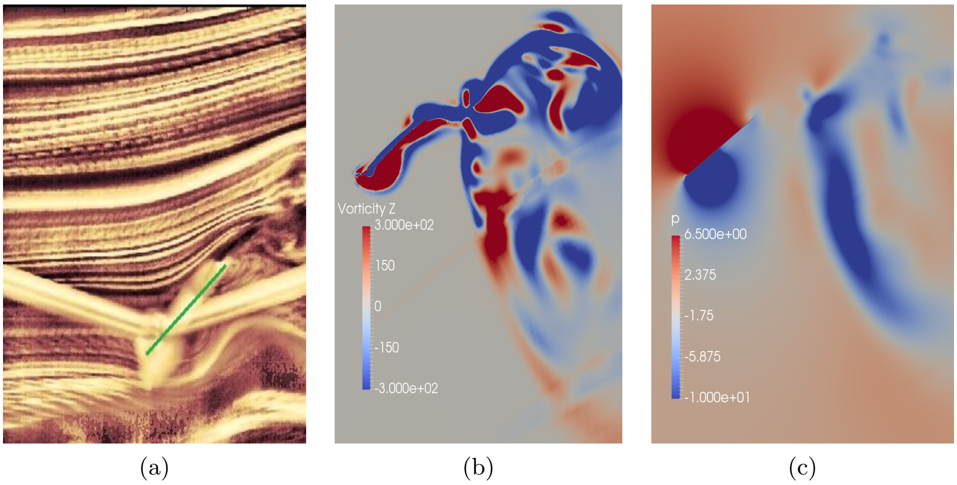
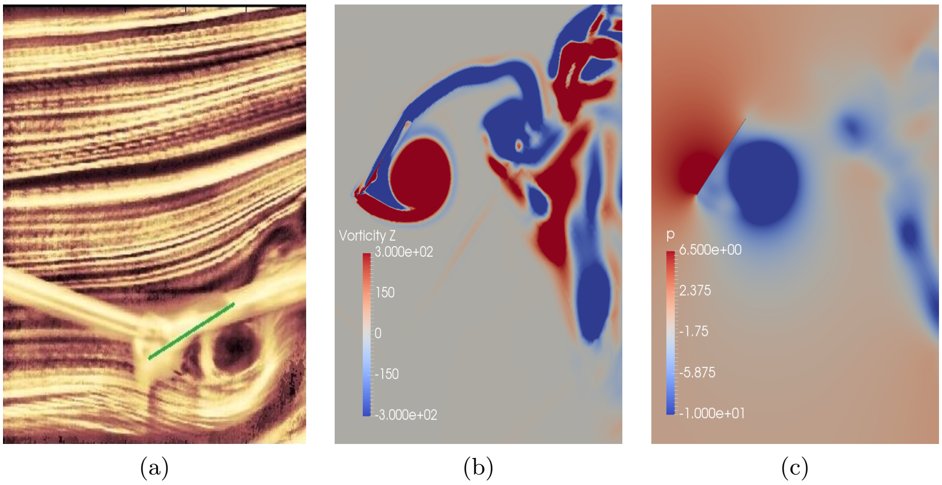


Figure 6.13: Snapshot from t/T=0.7



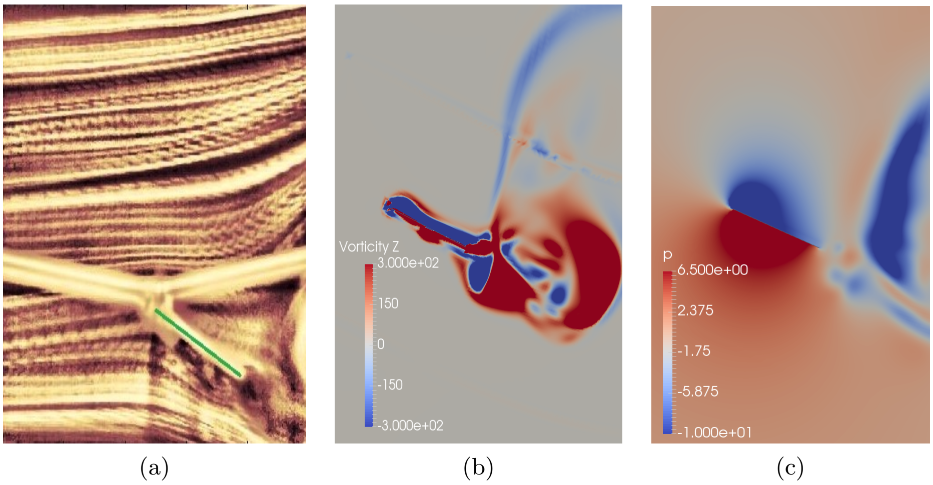
(a) (b) (c)

Figure 6.14: Snapshot from t/T=0.8



(a) (b) (c)

Figure 6.15: Snapshot from t/T=0.9



(a) (b) (c)

Figure 6.16: Snapshot from t/T=1.0

vortex, smaller than previous, which retards the lift fall (Fig. 6.10), but however the suction region created by this vortex is not powerful enough to move up the wing. Indeed as we can see from Fig. 6.11 (t/T=0.5) there is a pressure imbalance between trailing and leading edge that tends to rotate the wing. This imbalance is caused by first vortex which interacts still with the structure. Once the pitch angle has become negative, the lift force becomes negative in turn (Fig. 6.12). The pitch angle continues to decrease, and another leading edge vortex is growing at intradox, but this time is clockwise (Fig. 6.13). At t/T=0.7 the vortex sheds from the wing, decreasing the downforce. We have then the same mechanism described before (Fig. 6.14). At t/T=0.8 we have a stop decreasing of downforce, due to secondary vortex (Fig. 6.15). When the angle of attack is near to 0 there is a strong pressure imbalance between the trailing and leading edge, which tends to rotate the wing, bringing it to positive pitch angle again.

In the Fig. 6.6 is also reported the drag trend as a function of dimensionless time (green line) The drag is a sum of three component principally: the skin friction, the pressure drag and the wave drag. The first one is caused by the friction of the fluid against the "skin" of the body and it is related to wetted surface. The second one arises because of the shape of the object, bodies with larger cross section will have a higher drag than thinner bodies. The latter one arises only in case the Mach number is sufficiently high to create shock wave, which are regions where a lot of energy is dissipated.

We can note that the drag coefficient has not a specular trend in a period, because of the different pitch angle reached in upward and downward movement. Indeed, for positive pitch angle, the pitch amplitude it is greater of 7° than negative pitch angle amplitude. From the flat plate theory, we know that pressure drag is proportional to angle of attack, so there is agreement between the theory and numerical simulations.

We have also found that when the angle of attack is changing of sign (Fig. 6.12), there is still a strong interaction between the detached vortex and the wing, causing a low pressure region at trailing edge. The fluid therefore exerts a force in opposition to flow velocity, generating a little propulsion. When the low pressure vanishes at the trailing edge, the drag becomes positive again.

The pitching moment coefficient computed at pivot point ($0.01c$ far from leading edge), has the trend reported in Fig. 6.6 in light blue. The pitching moment is an index of how much the wing is loaded in torsion. The two extreme values of pitching moment are positioned halfway between the two peaks of lift and drag coefficient. The pitching moment coefficient has been used to find out the centre of pressure (x_{CP}, y_{CP}, z_{CP}) during the time the vortex is being forming. The centre of pressure is the point on the body where the aerodynamic force are applied. We will not compute the z_{CP} coordinate because the thickness it is negligible compared to the chord and the span. To obtain the centre of pressure position we have to resolve this equation:

$$M_{PP} = L(x_{CP} - x_{PP}) + D(y_{CP} - y_{PP}) \tag{6.1}$$

where M_{PP} is the moment computed at pivot point, x_{PP} and y_{PP} are the coordinates of pivot point, L is the lift and D is the drag. These are all quantities computed by the code, the only unknown is the centre of pressure coordinates. Knowing that:

$$C_M = \frac{M_{PP}}{\frac{1}{2}\rho U^2 Sc}, \qquad C_L = \frac{L}{\frac{1}{2}\rho U^2 S}, \qquad C_D = \frac{D}{\frac{1}{2}\rho U^2 S} \tag{6.2}$$

and writing down the centre of pressure coordinates as a function of the pitch angle (α) and the distance from pivot point (d), we can compute the dimensionless distance of centre of pressure from pivot point ($d^* = d/c$):

$$d^* = \frac{C_M}{C_L cos(\alpha) + C_D sin(\alpha)} \tag{6.3}$$

We have reported in figure 6.17 the trend of dimensionless distance d in the period from the vortex is growing until it sheds from the wing. We can see clearly that the centre of pressure moves to pivot point during the increasing of suction region, because the vortex moves downward the wing.

6.2.4 Comparison between LES and RANS results

In Tab. 6.4 we can see the difference of the values between the LES and RANS simulation. The frequency does not practically change, the dimen-

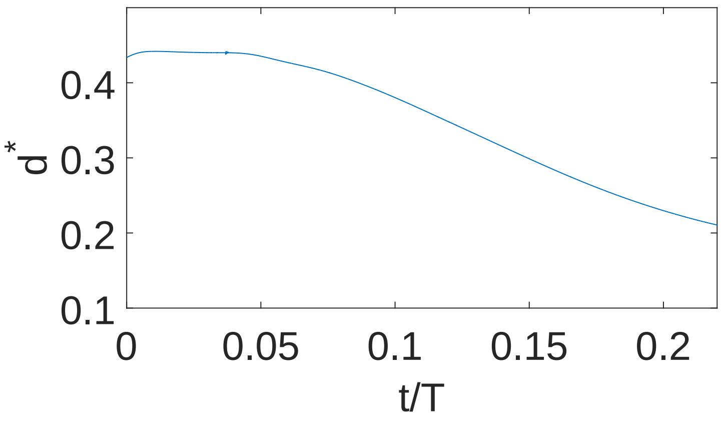
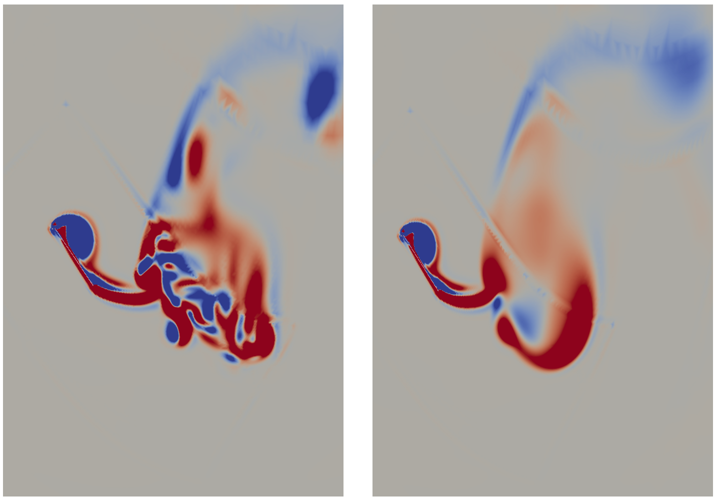


Figure 6.17: Trend of dimensionless distance between x_{CP} and x_{PP}.

	f [Hz]	A^*	ϕ [°]
LES simulation	17.04	0.56	73.33
RANS simulation	17.06	0.59	68.18

Table 6.4: Numerical results for frequency, dimensionless amplitude and phase for LES and RANS simulation for grid c_4.

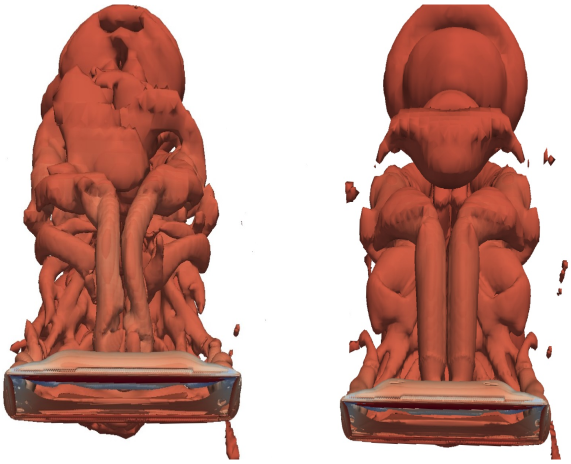
sionless plunge amplitude differs of 5%, while the phase between pitch and plunge differs of 6%. We can see from Fig. 6.18 that the leading edge vortex is resolved in the same way by the LES and RANS approach. There are marked differences in vorticity field behind the wing, indeed the swirling structures are more detailed for LES simulation because of the turbulence method allows to resolve in a direct way all the vortex structures bigger than the grid scale. However the turbulence modelling has no effect on the solution near the wing for this kind of problem. We can also see a front 3D image of the wing (Fig. 6.19) showing the iso-surface of Q (Sec. 4.8). We can appreciate that the vortex structures are similar and the surfaces resolved by LES are more detailed and less smooth than the surfaces of RANS simulation.



(a) LES simulation

(b) RANS simulation

Figure 6.18: Snapshot of z component of vorticity field for LES simulation (a) and RANS (b) simulation for the same time.



(a) LES simulation

(b) RANS simulation

Figure 6.19: Snapshot of iso-surfaces of Q for LES simulation (a) and RANS (b) simulation for the same time.

Chapter 7

Conclusions

This work was aimed to give a contribution to improve the FLEHAP device, a novel energy harvester exploiting the fluid structure interaction which is under active development.

The campaign has been made varying the wind velocity showing that numerical simulation are in agreement with the experimental data for the pre-transition condition ($U < 3.5$ m/s) and for the post-transition condition ($U > 4.5$ m/s).

The main focus has been on to resolve the critical issue of mismatch between numerical simulations and experimental data at $U = 4$ m/s. This case has been revealed hard to reproduce because it is a condition of transition, so the dependence by parameters it is much more sensitive compared to pre and post-transition phase. The major issue is that CFD tends to anticipates the post-transition phase.

The first result is that is necessary adopting a numerical damping value above 0.2 Ns/m in order to obtain a limit cycle of oscillation.

We have led a grid dependency analysis, which has showed that for this kind of problem the grid resolution does not influence the solution.

We have compared the vortex dynamics between the CFD simulations and the experiment, showing that the process of vortex growth and vortex shedding are in agreement. A force analysis has been led, correlating the aerodynamic coefficient trend with the vortex dynamics in a limit cycle of oscillation.

The lift coefficient reaches the maximum value when the leading edge vortex is still attached to the wing and it is at its maximum extension. When the vortex sheds the lift falls but not in a constant way because of the presence of a secondary vortex at leading edge.

The drag coefficient is proportional to the angle of attack of the wing and at nearly 0° the drag is negative. This can be explained looking at pressure field, indeed at trailing edge there is still an interaction between the vortex and the wing, creating a powerful suction region. This pressure imbalance leads to generate a little propulsion.

The pitching moment has been used to compute the centre of pressure of the wing. We have found that during the growth of the vortex the centre of pressure moves to pivot point because of the fluid is going to detach on the wing extradox.

Eventually the LES simulation does not give a marked difference solution compared to the RANS simulation, with the same grid and the same elastomer damping. The display of Q-criterion has showed that the RANS simulations tend to smooth much the vortex intensity at downstream the wing compared to LES simulations. Futures works plan to study the tandem configuration. Seen these latter results, a LES approach is recommended for studying a tandem configuration, because the LES does not smooth the vorticity field far from the wing, thus simulating a better incoming turbulence for the trailing devices.

For future activities, it will be more important working on the modelling of the wing, adding the cylindrical rod at leading edge where the coils are fixed. The adding of this rod can be crucial because the flow is changed and, for the transition conditions, we could obtain very different solutions. We have also to change the physical model: it can be decisive adding a torsional spring in addition to the elastomers, and also adopting a flexible model instead of a rigid body model.

Bibliography

[1] A Andersen, U Pesavento, and Z Jane Wang. Unsteady aerodynamics of fluttering and tumbling plates. *Journal of Fluid Mechanics*, 541:65–90, 2005.

[2] MA Ashraf, J Young, JC S. Lai, and MF Platzer. Numerical analysis of an oscillating-wing wind and hydropower generator. *AIAA journal*, 49(7):1374–1386, 2011.

[3] C Boragno and G Boccalero. A new energy harvester for fluids in motion. In *SPIE Smart Structures and Materials+ Nondestructive Evaluation and Health Monitoring*, pages 94310G–94310G. International Society for Optics and Photonics, 2015.

[4] C Boragno, R Festa, and A Mazzino. Elastically bounded flapping wing for energy harvesting. *Applied Physics Letters*, 100:253906, 2012.

[5] DL Brown, WD Henshaw, and DJ Quinlan. Overture: An object-oriented framework for solving partial differential equations on overlapping grids. *Object Oriented Methods for Interoperable Scientific and Engineering Computing, SIAM*, pages 245–255, 1999.

[6] AR Collar. The first fifty years of aeroelasticity. *Aerospace*, 5(2):12–20, 1978.

[7] JH Ferziger and M Peric. *Computational methods for fluid dynamics*. Springer Science & Business Media, 2012.

[8] LM Gasendo. Wave energy megawatts harvester, 2010. US Patent 7,687,931.

[9] M Germano, U Piomelli, P Moin, and WH Cabot. A dynamic subgrid-scale eddy viscosity model. *Physics of Fluids A: Fluid Dynamics*, 3(7):1760–1765, 1991.

[10] OpenFOAM User Guide. The openfoam foundation, 2011.

[11] G Haller. An objective definition of a vortex. *Journal of Fluid Mechanics*, 525:1–26, 2005.

[12] WD Henshaw. A fourth-order accurate method for the incompressible navier-stokes equations on overlapping grids. *Journal of computational physics*, 113(1):13–25, 1994.

[13] WD Henshaw. Ogen: An overlapping grid generator for overture. *LANL unclassified report*, pages 96–3466, 1998.

[14] WD Henshaw. On multigrid for overlapping grids. *SIAM Journal on Scientific Computing*, 26(5):1547–1572, 2005.

[15] W Huang, H Liu, F Wang, J Wu, and HP Zhang. Experimetal study of a freely falling plate with an inhomogeneous mass distribution. *Physical Review E*, 88(5):053008, 2013.

[16] F Juretić. Cfmesh user guide. *Creative Fields, Zagreb*, 2015.

[17] E Lefeuvre, D Audigier, C Richard, and D Guyomar. Buck-boost converter for sensorless power optimization of piezoelectric energy harvester. *IEEE Transactions on Power Electronics*, 22(5):2018–2025, 2007.

[18] R Liu, Q Zhang, and LE Cross. Experimental investigation of electrostrictive polarization biased direct apparent piezoelectric properties in polyurethane elastomer under quasistatic conditions. *Journal of applied polymer science*, 73(13):2603–2609, 1999.

[19] WJ McCroskey. The phenomenon of dynamic stall. Technical report, DTIC Document, 1981.

[20] W McKinney and J DeLaurier. Wingmill: an oscillating-wing windmill. *Journal of energy*, 5(2):109–115, 1981.

[21] CH Moeng. A large-eddy-simulation model for the study of planetary boundary-layer turbulence. *Journal of the Atmospheric Sciences*, 41(13):2052–2062, 1984.

[22] S Olivieri, G Boccalero, A Mazzino, and C Boragno. Fluttering conditions of an energy harvester for autonomous powering. *Renewable Energy*, 2016.

[23] A Orchini, A Mazzino, J Guerrero, R Festa, and C Boragno. Flapping states of an elastically anchored plate in a uniform flow with applications to energy harvesting by fluid-structure interaction. *Physics of Fluids (1994-present)*, 25(9):097105, 2013.

[24] MP Païdoussis, SJ Price, and E De Langre. *Fluid-structure interactions: Cross-flow-induced instabilities.* Cambridge University Press, 2010.

[25] M Piñuela, PD Mitcheson, and S Lucyszyn. Ambient rf energy harvesting in urban and semi-urban environments. *IEEE Transactions on Microwave Theory and Techniques*, 61:2715–2726, 2013.

[26] D Poirel and SJ Price. Response probability structure of a structurally nonlinear fluttering airfoil in turbulent flow. *Probabilistic engineering mechanics*, 18(2):185–202, 2003.

[27] JB Rose and PA Staniland. Thermoplastic aromatic polyetherketones, 1982.

[28] L Shen, E Chan, and P Lin. Calculation of hydrodynamic forces acting on a submerged moving object using immersed boundary method. *Computers & Fluids*, 38(3):691–702, 2009.

[29] D Shiels, A Leonard, and A Roshko. Flow-induced vibration of a circular cylinder at limiting structural parameters. *Journal of Fluids and Structures*, 15(1):3–21, 2001.

[30] K Singh, S Michelin, and E De Langre. The effect of non-uniform damping on flutter in axial flow and energy-harvesting strategies. In *Proc. R. Soc. A*, volume 468, pages 3620–3635. The Royal Society, 2012.

[31] AH Squillacote and J Ahrens. *The paraview guide*, volume 366. Kitware, 2007.

[32] HK Versteeg and W Malalasekera. *An introduction to computational fluid dynamics: the finite volume method.* Pearson Education, 2007.

[33] RJM Vullers, Rob van Schaijk, Inge Doms, Chris Van Hoof, and R Mertens. Micropower energy harvesting. *Solid-State Electronics*, 53(7):684–693, 2009.

[34] S Wang, DB Ingham, L Ma, M Pourkashanian, and Z Tao. Numerical investigations on dynamic stall of low reynolds number flow around oscillating airfoils. *Computers & Fluids*, 39(9):1529–1541, 2010.

[35] F Yildiz, J Zhu, and L Guo. Ac 2007-2254: Energy scavenging for wireless sensor nodes with a focus on rotation to electricity conversion. *age*, 12:1, 2007.

[36] S Yoon and A Jameson. Lower-upper symmetric-gauss-seidel method for the euler and navier-stokes equations. *AIAA journal*, 26(9):1025–1026, 1988.

[37] Q Zhu, MJ Wolfgang, DKP Yue, and MS Triantafyllou. Three-dimensional flow structures and vorticity control in fish-like swimming. *Journal of Fluid Mechanics*, 468:1–28, 2002.