ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA MECÁNICA

IMPLICIT LARGE EDDY SIMULATION FOR UNSTEADY CAVITATING FLOW AROUND HYDROFOILS USING OPENFOAM SOFTWARE

JORGE CARLOS BAQUERO DUQUE

jorge.baquero@epn.edu.ec

DIRECTOR:
ING. ESTEBAN ALEJANDRO VALENCIA TORRES PhD.

esteban.valencia@epn.edu.ec

CO-DIRECTOR:
ING. VÍCTOR HUGO HIDALGO DÍAZ MSc.

victor.hidalgo@epn.edu.ec

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO MECÁNICO

September 30, 2015

DECLARACIÓN

Yo, Jorge Carlos Baquero Duque, declaro que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.
La Escuela Politécnica Nacional, puede hacer uso de los derechos correspondientes a este trabajo, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Jorge Carlos Baquero Duque

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Jorge Carlos Baquero Duque bajo mi supervisión.

Ing. Esteban Valencia PhD.
DIRECTOR DEL PROYECTO

Ing. Victor Hidalgo MSc.
CO-DIRECTOR DEL PROYECTO

Son tantas las personas que han hecho posible que uno llegue a donde esté, son tantas las casualidades que permiten que seamos lo que somos.

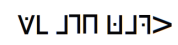
Maygo, gracias por todo lo que has hecho para que llegue a donde he llegado, si alguien se merece un mérito aún mayor, esa persona eres tú.
Pa, gracias por el apoyo incondicional y por todas las oportunidades que me ha dado, no tengo palabras para agradecerle y felicitarle por todo lo que ha hecho para salir adelante junto a su familia, gracias de todo corazón.
Pau, gracias por todo, gracias por aguantarme tantos años.
Jhon, gracias por ser un amigo más.
Angelita, gracias por ser mi segunda mamá, usted es otra de las personas que se merece un mérito enorme.
Lenno, Pablo, Santi, Diego, Andrés, Fercho y desde hace un par de años atrás Stalin, gracias por ser como mis hermanos y por haberme apoyado de una u otra forma en toda mi vida.
A mis primos, a mis tíos, abuelitos y amigos cercanos, gracias infinitas por lo que hacen y han hecho para mí y para mi familia, disculpen por no incluirles uno a uno, pero saben que si lo hiciese me faltaría hojas y no quiero pensar que olvidé el nombre de alguno por incluir.
A todos mis amigos de la U, con quienes compartí tantos años, gracias por todo el afecto recíproco.
A los Superconocidos, por ser mi segunda familia, es cierto eso que dicen, de que los amigos es la familia que uno escoge.
A los Chuchan y todos sus allegados, porque sin ustedes no sabría el verdadero significado de lo que es una entidad, un grupo que busca siempre un bien común.
A mi tutores, Esteban y Victor, por darme la oportunidad de realizar este trabajo y por todo lo que han podido enseñarme para conseguir el mismo.
A José y Jkz por toda la ayuda y el apoyo para lograr sacar adelante este proyecto.

A ti Dany, por cambiar mi vida.

Jor

"¿Qué es lo peor
que te pueden decir?"
Lenno

Per te,
Jor



"This is what happens
when an unstoppable force
meets an immovable object"
Joker

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

Capital letters

C	—	Courant number.
C_{max}	—	Maximum value of C.
C_{mean}	—	Mean value of C.
$C_{p_{min}}$	—	Coefficient of pressure.
$C_{\overline{p}}$	—	Main pressure field.
D	—	Runner diameter.
H_n	—	Net head.
L	—	Liquid subindex.
N	—	Power.
ND	—	Number of nodes.
NE	—	Number of elements.
P	—	Constant pressure.
P_s	—	Spinodal pressure.
Q	—	Flow rate.
R	—	Bubble radius.
R_0	—	Initial bubble radius.
Re	—	Reynolds number.
$\dot{R}$	—	Radius velocity.
T	—	Period.
		Ambient temperature.
T_b	—	Boiling temperature.
T_{cav}	—	Cavitation temperature.
T_∞	—	Flow temperature.
U_∞	—	Flow velocity.
V	—	Velocity.
		Vapor subindex.
V_V	—	Cavity Volume.
X	—	Common factor for turbomachinery equations.

Lower case letters

alt	—	Altitude.
b_1	—	Guide vane height.
c	—	Chord length.
c_0	—	Guide vane inlet velocity.
c_1	—	Runner absolute velocity.
$c_1^{'}$	—	Guide vane outlet velocity.
e	—	Maximum thickness of the cavity.
f	—	Frequency.
i, j, k	—	Space axes subindices.
g	—	Gravity.
l	—	Cavity length.

lat — Latitude.
m — Model subindex.
$\dot{m}$ — Interface rate mass transfer per volume.
n — Runner angular speed.
n_s — Specific velocity.
p — Prototype subindex.
p_A — Ambient pressure.
p_c — Critical pressure.
p_{cav} — Cavity pressure.
p_g — Partial gas pressure.
p_r — Reference pressure.
p_{max} — Maximum pressure.
p_v — Saturated vapor pressure.
p_∞ — Flow pressure.
$p^{'}$ — Liquid pressure fluctuation.
p_2 — Runner outlet pressure.
r_{max} — Maximum bubble radius.
t — Time.
Δt — Time step.
u — Instant velocity.
u_1 — Runner peripheral velocity.
u_x — Instant velocity in x axis.
u_y — Instant velocity in y axis.
u_* — Friction velocity.
x — Space axis.
Δx — Length interval in x axis.
y — Space axis (coordinates).
Distance to the nearest wall (y^+ number).
y^+ — Y plus number.
y^+_{max} — Maximum value of y^+.
y^+_{mean} — Mean value of y^+.
Δy — Length interval in y axis.

Greek letters

α — Attack angle (Hydrofoils).
Vapor fraction (cavitation).
α_b — Blade rotation angle.
α_1 — Angle between u_1 and c_1.
γ — Surface tension.
η — Efficiency.
λ — Scale prototype factor.
μ — Dynamic viscosity.
ν — Local kinematic velocity.
Ω — Omega number.
Π — Relation between bubble, fluid and vapor pressures.

Π_{max} — Maximum value of Π.
ϕ_d — Discharge coefficient.
ϕ_1 — Runner diameter.
ϕ_0 — Guide vane structure diameter.
ρ — Liquid density.
σ — Cavitation number.
σ_i — Incipient cavitation number.
σ_x — Impact load.
σ_u — Ultimate tensile strength.
σ_y — Yield strength.
$\bar{\sigma}$ — Mean amplitude of the impact loads.
τ — Bubble collapse time.
τ_w — Wall shear.

Others

$\forall$ — Volume.

ABSTRACT

Cavitation simulation allows to understand the generation of this phenomenon in turbomachinery. The present thesis analyze the cavitating flow around Francis-99 turbine blades using OpenFOAM software. The study include an analysis of the phenomenon of cavitation, a description of the open-source software used, the explanation of a methodology and the process to obtain the simulations results; starting from the points of the blade obtained in Blender and reaching with GMSH a refined structured mesh to be simulated. The solver use Implicit Large Eddy Simulation (ILES) method and Zwart cavitation model, it was obtained from PhD. candidate of Tsinghua University, engineer Victor Hugo Hidalgo, MSc. A similitude analysis was performed to prove the behavior of the cavitating flow varying the angle of attack, size of the mesh and cavitation number. The analysis of the different behaviors of the cavitation when vary the section of the blade are also presented. Finally is included a mesh with the guide vane and blade of the turbine to analyze the behavior in time. To validate the meshes are used some parameters like the Yplus, y^+ and Courant, C, numbers and was performed a hexahedral quality mesh distortion analysis. Results show the similitude in the behavior of the cavities with proposed experimental data, an assessment of the study of the cavitation phenomenon and a determination of the behavior of the cavities volume generated in time to reach minimums, maximums and its cycle.

keywords: Cavitation, OpenFOAM, structured mesh, ILES.

SUMMARY

La simulación de cavitación permite entender la generación de este fenómeno en turbomaquinaria. La presente tesis analiza el flujo cavitante alrededor de los álabes de la turbina Francis-99 utilizando OpenFOAM. El estudio incluye el análisis del fenómeno de cavitación, una descripción del software libre utilizado, la explicación de la metodología y el proceso de obtención de los resultados de la simulaciones; desde los puntos del álabe obtenidos en Blender, hasta alcanzar con GMSH un mallado estructurado refinado para ser simulado. El solver utiliza el método ILES y el modelo de cavitación Zwart, mismo que fue obtenido a través del candidato a PhD. de la Universidad de Tshinghua, el ingeniero Victor Hugo Hidalgo, MSc. Un análisis de similitud fue realizado para probar el comportamiento del flujo cavitante variando el ángulo de ataque, tamaño de malla y número de cavitación. El análisis de los comportamientos de la cavitación variando la secciones del álabe también es presentado. Finalmente se incluye una malla con un álabe de distribuidor y turbina para analizar el comportamiento en el tiempo. Para validar las mallas se utilizaron varios parámetros como los números Y-más, y^+, y Courant, C. Fue realizado además un análisis de calidad hexahédrico de distorsión de la malla. Los resultados indican una similitud en el comportamiento de las cavidades con los datos experimentales propuestos, la valoración del estudio del fenómeno y una determinación del comportamiento del volúmen generado de las cavidades en el tiempo alcanzando mínimos, máximos y sus ciclos.

Palabras clave: *Cavitación, OpenFOAM, mallado estructurado, ILES.*

OBJECTIVE

To simulate cavitation on hydrofoils using OpenFOAM simulation platform with Implicit Large Eddy Simulation Method (ILES) and a structured mesh.

SPECIFIC OBJECTIVES

- To realize a literature research about cavitation and its simulation with CFD.
- To perform a geometric and operating conditions characterization for Francis-99 hydrofoils.
- To define and realize the structured mesh of the hydrofoils geometry using Gmsh.
- To validate the obtained meshes for its use in OpenFOAM.
- To simulate the fluid behavior on the hydrofoils using Implicit Large Eddy Simulation method (ILES) with the CFD opensource software OpenFOAM.
- To compare the simulation results and define the accuracy of the prediction of the developed method for the assessment of cavitation using experimental results.

SCOPE

- This investigation will realize a geometric characterization of Francis-99 hydrofoils, for determine the conditions of the structured mesh and the OpenFOAM simulation using Implicit Large Eddy Simulation method (ILES).
- The characteristics of the structured mesh will be defined, that will be used in the software Gmsh. These characteristics are: size, nodes and elements range, and the values of Ω and y^+ numbers.
- Results will be validated using previous experimental studies and numerical research.

MOTIVATION

This project arises from the necessity to understand unsteady cavitating flow and to prevent damage on hydraulic machinery.

Nowadays, the hydraulic potential energy has been used as an axis of the energetic matrix and industry in Ecuador. The hydropower stations plants will transform the country economy and studies of cavitation are fundamental to prevent machinery stops.

The aforementioned reasons show the importance of improving the efficiency of energy production by reducing sources of damage in equipment.

1 INTRODUCTION

Cavitation is considered a big problem in many fields in engineering, where a reduction of its effects represent an advantage to increase efficiency and life time of the components of the turbomachinery. Two of the ways to assess this phenomenon are the simulation process and the experimental way.

The simulation process concerns the computational analysis, which requires an element to analyze, a type of mesh, a specific software and the methodology and equations to perform the solvers.

The experimental way comprise test-rigs used to model process and analysis of operating plants of the electric generation industry based in turbomachinery, like *Hidroagoyán* project of *Corporación Eléctrica del Ecuador - CELEC EP* or the Francis-99 project, by Noewegian University of Science and Technology - NTNU and Norway and Lulea University of Technology - LTU (NTNU & LTU, 2015a), where the conditions of the phenomenon give important information.

The analysis of cases like *Hidroagoyán* and Francis-99 projects with its respective simulation will allow to contribute at the investigation of the adverse effects and prevention techniques of cavitation.

1.1 HYDROPOWER GENERATION IN ECUADOR

Actually in Ecuador the change of the productive matrix generate a substantial growing in the construction and repowering of power plants, where the use of water resources is a cornerstone for the development.

Guarantee the electric sovereignty, change the productive matrix and consolidate the business activity of the electric sector are part of the strategic plan in the development of the public company in Ecuador (CELEC-EP, 2013, p.2); which become reality in *Hidroagoyán*, *Hidronación*, *Hidropaute* business units and others. The actual situation of the electricity generation process in the country could be improved with the analysis of the effects that are produced in the machinery, which reduces the efficiency and generate an increment in maintenance costs.

The present investigation proposes to analyze the cavitation phenomenon in Francis turbines, considering at *Unidad de Negocio Hidroagoyán* as pattern for investigation because a link between *Escuela Politécnica Nacional* with the project PIMI1403 (Aguinaga, Hidalgo, Valencia, Luo, & Cando, 2014) which proposes to improve a method to reduce erosion and enhance the operation by numerical simulation. In this thesis are used the results and information of the Francis-99 project, by Noewegian University of Science and Technology - NTNU and Norway and Lulea University of Technology - LTU (NTNU & LTU, 2015a), which made a simulation study of a Francis turbine in stable and transient conditions, to analyse the behavior of the cavitating flow in the turbine. The results of the project Francis-99 are used because it is a free knowledge work with a high similarity with the *Hidroagoyán* phenomenon.

Hidroagoyán hydroelectric is located in *Baños*, *Tungurahua*, at 180[km] to the south-east of *Quito*. It was created to utilize the water from *Pastaza* river to generate electric energy. Nowadays, this hydroelectric central is able to generate 156[MW]. *Hydroagoyán* hydroelectric has a net head of 163[m] in the maximum point with a caudal of 60[m^3/s] in the entrance of the 2 Francis vertical axis turbines. These Francis turbines rotates at 225[rpm] and generates 5[%] of the Ecuador total's energy. Actually exist an accumulation of sediment by the river contamination that is one of the factor for the continuous maintenance of the central.

Hidroagoyán central can not operate if the accumulation of sediment reaches more than 5[m] in the desanders. The efficiency drops when rains, when these are present the level of the particles reaches 6000[ppm]; in a normal operation without rain this value is approximately 600[ppm].

The maintenance include wear test each 6 months, and a complete overhaul each 7 or 8 years. The blades reparation is realized each 2 years. The impeller is repaired every 2 years, with a life time of 7 or 8 years (Abad, 2015). The process include covering the blades with high hardness paste and the inclusion of filler welding to cover the damage produced by cavitation in the runner or erosion in the body.

Cases like *Unidad de Negocio Hidroagoyán* are common to be found in Ecuador with another hydroelectric power plants, which opens a big field of investigation to reach an improvement of the generation efficiency.

To have a better knowledge of the phenomenon of cavitation it is important to make a literature research, this information is mentioned in the next subchapter.

1.2 A BRIEF DESCRIPTION OF CAVITATION MECHANISMS

Cavitation is defined as the process of formation of the vapor phase of a liquid, when it is subjected to reduced pressures at constant ambient temperature and reaches the substance vapor pressure. Due to the reduction of the pressure, bubbles form and grow and the liquid consists in a two-phase flow, liquid and vapor, denominated cavitating flow (Eisenberg, N/A, p.121).

Cavitation could be defined by two different regimes (Franc & Michel, 2005, p.5):

- Inception, the limiting regime between the non-cavitating and the cavitating flow.
- Developed cavitation, in which occurs a certain permanency and extent of the cavitation or a significant fall in performance of machines.

In this thesis are presented the basic aspects of: the formation of cavitation, nucleation, inception, bubble grow and collapse, types of cavitation and its effects. The scheme of the cavitation basic information researched is showed in the next graphic.

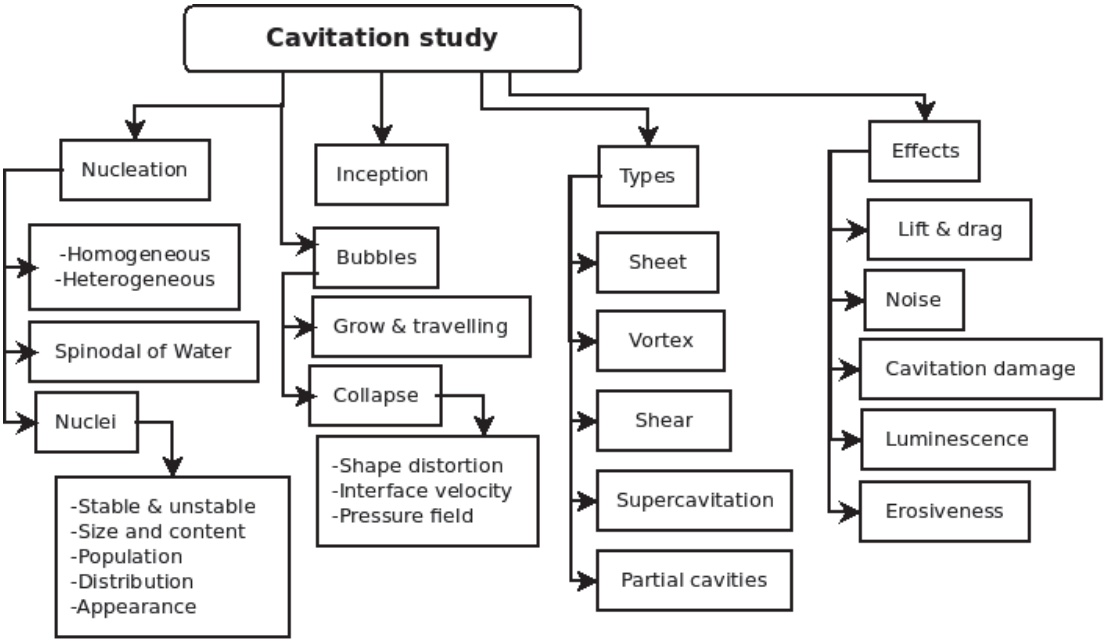


Figure 1.1: Cavitation study researched

1.2.1 NUCLEATION

- Homogeneous & heterogeneous
  When occurs thermal fluctuations in a liquid a new phase appear in an homogeneous nucleation, a metastable state is present respect to its vapor, like when a liquid is superheated at its boiling temperature T_b at constant pressure P or when its leaded below its saturated vapor pressure p_v. This liquid cavitate when try to return to equilibrium by nucleation of vapor bubbles (Caupin & Herbert, 2006, p.1000-1001). To generate a significant volume of vapor a small amount of heat is required (Franc & Michel, 2005, p.2).

  From a theoretical view, the steps of formation of cavitation are: the breakdown or void creation, filling the void with vapor and eventual saturation with vapor. In reality these steps are simultaneous (Franc & Michel, 2005, p.3). To remain the system metastable, molecules present its natural mutual attraction. When its present a large intermolecular distance, the system and consequently the nucleation becomes unstable (Caupin & Herbert, 2006, p.1001).

  Generally cavitation is heterogeneous due to a reduction of the threshold for nucleation, by the impurities. The analysis of this common type of cavitation could be by the mechanisms on flat hydrophobic surfaces or on pre-existing bubbles. When the liquid is contact with a solid substrate and exists a variation of the wetting properties, denoted by hydrophilic or hydrophobic surfaces, cavitation temperature T_{cav} decreases from its original value. The reduction is in a higher proportion in hydrophobic surfaces. When exists floating bubbles of air that are not dissolved in the fluid with diffusion and the pressure in the liquid decrease, cavitation appears. In some cases to reduce this problem, liquid is pre-pressured, but in many cases bubbles are small but exists yet and are trapped in the cracks of the surface; if the time not is the enough to generate the diffusion a heterogeneous nuclei occurs (Caupin & Herbert, 2006, p.1010-1011).

- Spinodal of water
  When the system becomes unstable, critical density is called spinodal density ρ_s, which correspond to a spinodal pressure P_s, in this state the compressibility of metastable liquid diverges and long wavelength perturbations can grow without limit. The spinodal of water is a behavior different from other liquids, water possesses a line of density maxima (LDM) and shrinks upon heating bellow $4[^{\circ}C]$ at atmospheric normal pressure (Caupin & Herbert, 2006, p.1002-1003).

- Nuclei
  Cavitation nuclei are called points of weakness in a liquid, formed by small gas and vapor inclusions who operate as starting points for the liquid breakdown. Their size is between a few micrometers and some hundreds of micrometers, and it shape is spherical due to surface tension, they can be referred to as microbubbles (Franc & Michel, 2005, p.15-16).

  – Stable & unstable
    Cavitation only occur when is subjected to a pressure reduction and nuclei becomes unstable and grow; from the static equilibrium analysis it is obtained the next equation: (Eisenberg, N/A, p.126-127)

$$p_A + \frac{2\gamma}{R} = p_v + \frac{cons}{R^3} \tag{1.1}$$

    Which is a relation between the ambient pressure plus surface-tension pressure and the vapor pressure plus the gas pressure. In the next graphic is indicated this relationship:

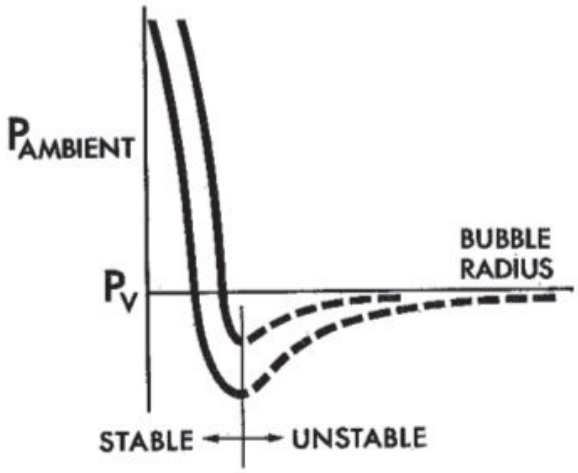


Figure 1.2: Static equilibrium conditions for 2 different gas contents, right plot has less gas content that left plot (Eisenberg, N/A, p.127).

The behavior is stable when nuclei never reach a large size. When nuclei expand over the unstable region grow explosively (Eisenberg, N/A, p.127). The minimum pressure, that the nucleus can withstand under stable conditions is a limiting value called the critical pressure, p_c. The difference $p_c - p_v$ is the static delay to cavitation (Franc & Michel, 2005, p.18-19).

- Size and content
  Because a stable nucleus the size of nuclei are small when gas is diffused in the surrounding fluid. If nuclei are very large its rise to the surface and vent, also decrease in size by diffusion too. That cause cavitation and happens when a reduction of the pressure is present (Eisenberg, N/A, p.127-128).
- Population
  Exists two common methods to make the measurement of the nucleus population: optical methods and dynamical methods, actually the second one is the most used, because give information of the critical pressure p_c in a model of a venturi device. This device uses an ultrasonic transducer to count the quantity of nuclei explosions (Franc & Michel, 2005, p.27-30).
- Distribution
  The pressure around bubble cavities rises, until vapor pressure is reached, when bubble growth increases the surrounding pressure. Bubble coalesce and individual bubbles cannot be distinguished when large nuclei affect the flow; smaller bubbles not grow anymore or become unstable. The number of cavitation bubbles in bubble cavitation only reflects a band of the large available nuclei (Kuiper, 2010, p.32).
- Appearance
  If large nuclei scarce exist a decrease of the amount of bubble cavities, which produces that cavities grow in size, if there many nuclei present, cavities reduce its size. When the number of bubble cavities grow when nuclei generated the maximum size is practically equal, but with an increase of the nuclei density the maximum size decreases (Kuiper, 2010, p.33). In the moment of inception, cavitation can take different patterns like transient isolated bubbles, attached or sheet cavities and cavitating vortices, which depends of the structure of the non-cavitating flow and the cavitation. Attached cavities can be partial cavities, when are close to the wall, or supercavities, when are close away from the boundary (typically a foil) (Franc & Michel, 2005, p.4-5).

1.2.2 INCEPTION

When a reduction of the pressure, which is below the vapor pressure, is detected too late, the risk of lack of nuclei grows. Because of this, is common to believe that bubble cavitation is erosive, but that is true when the detection is too late (Kuiper, 2010, p.32). When occurs nucleation the inception happens, which is defined as the limiting regime between the non-cavitating and the cavitating flow (Franc & Michel, 2005, p.5).

Some situations favorable to cavitation are: wall geometry who rise to sharp, shear flows due to large turbulent pressure fluctuation, basic unsteady nature of some flows with strong fluid acceleration, the local roughness of the walls, the vibratory motion of the walls with large oscillation amplitude and in a last case, solid bodies that are suddenly accelerated by a shock in a quiescent liquid (Franc & Michel, 2005, p.5).

To measure the probability of inception is used the cavitation number σ, defined by:

$$\sigma = \frac{p_\infty - p_v(T_\infty)}{\frac{1}{2}\rho_L U_\infty^2} \tag{1.2}$$

Where, p_∞ is the pressure of the flow, p_v is the saturated vapor pressure; function of the temperature of the flow T_∞, ρ is the liquid density and U_∞ is the velocity of the flow.

In a first observation, the incipient cavitation number σ_i is defined as the value which cause an increase of the number and size of the vapor bubbles. This occur when cavitation number σ get smaller than σ_i. Another number, called the coefficient of pressure $C_{p_{min}}$, which determines the lower pressure in the single phase flow, may be used to measure the cavitation inception, because an approximation predicts that $\sigma_i = -C_{p_{min}}$. This prediction in reality is not totally true, because some factors, such as the capacity of the liquid of sustain a tension which is a function of the contamination, the necessity of a finite residence time of the cavitation nuclei to grow and reach and observable size, and also the measure of the $C_{p_{min}}$ value, because is calculated or estimated.(C. Brennen, 2005, p.128-130)

1.2.3 BUBBLES

- Grow & travelling
  When in a nucleus occurs a drop of the pressure below the vapor pressure p_v and reaches the critical pressure p_c, happens a growth of the bubble in an unstable behavior, this situation is mentioned in the next figure:

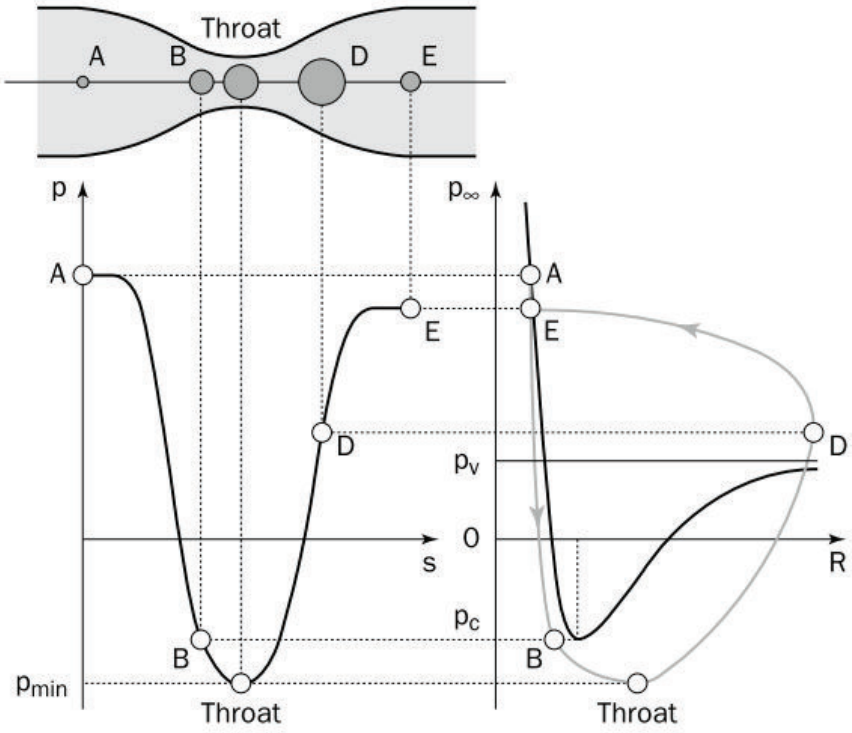


Figure 1.3: Behavior of nucleus in a venturi in an unstable case. (Franc & Michel, 2005, p.20)

In an initial condition, the nucleus start from a pressure A, which is bigger than vapor pressure p_v, and fall to a pressure B, which is smaller than vapor pressure, in this moment the bubble grows and continue traveling to a condition C, called Throat. When the bubble is in the Throat the bubble turns unstable because the decreasing value of the pressure, which reaches the critical pressure p_c. The maximum size is reached at point D and the bubble, because its unstable condition, starts to implode violently. The time spent in the low-pressure region is long compared to the collapse time (Franc & Michel, 2005, p.20-21).

- Collapse
  - Shape distortion
    By the pressure gradients in the flow, single cavitation bubbles are often highly distorted. Bubble is separated from the wall by a thin layer during the initial growth phase, the layer thickness has the same order magnitude as the boundary layer. In region of adverse pressure gradient the exterior frontal surface is pushed inward and the bubble reach a profile of a wedge-like, that produces the bubble fissioning into two parts, frontal and rear (C. Brennen, 2005, p.139-142).

    In some cases, the observed distortion in the bubbles during its collapse seems like the shape spherical becomes oblate spherical, then the bottom become flatter and latter concave. The concavity of the bottom apparently may increase until the central portion of the bottom reaches the top, so that a final torus form is suggested (Fabula, 1958, p.19-24), like in the next graphic:

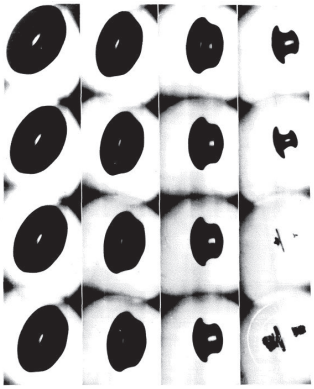


Figure 1.4: Plots of shape distortion in bubbles during time (Fabula, 1958, p.38)

In the collapse, bubbles evolves into cavitating vortices with spanwise axes and seem to rebound as a cloud of much smaller bubbles. Fission prior collapse can have effect on the noise produced. The bubble also have streaks or tails stretched out behind, which are attached to the solid surface in the trailing end. All the cavitation bubbles are substantially deformed and their dynamics and acoustics altered by the flow fields in which they occurs (C. Brennen, 2005, p.141-142).

– Interface velocity
The value of the change of size of the radius in time is negative when the bubble collapse. The radius tends to 0 and the radial inwards motion accelerate without limit. In the next graphic is represented this phenomenon:

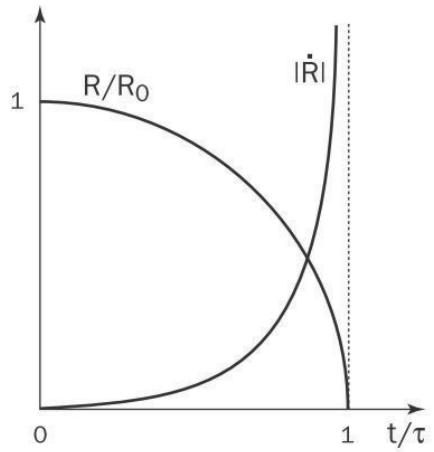


Figure 1.5: Behavior or the radius and its velocity in time when a bubble collapse.(Franc & Michel, 2005, p.39)

where R is the radius of the bubble in time, R_0 is the initial radius of the bubble before initiates the collapse, $\dot{R}$ is the velocity, t is time. τ is a good agreement with the experimental value when bubble collapse, like in an example in water, a bubble with an initial radius of $1[cm]$ collapse in one millisecond with an external pressure of $1[bar]$. With the previous values, $|\dot{R}| \approx 720[m/s]$ when $R/R_0 = 1/20$ which is like the half of the velocity of sound in water. This analysis and results could be different when some physical aspects vary, like the presence of non-condensable gas, finite rate of vapor condensation or others. (Franc & Michel, 2005, p.39)

– Pressure field
The pressure has a special behavior when cavitation occurs, this can be measured by a non-dimensional form denominated Π (Franc & Michel, 2005, p.40-41), which is the relationship between the pressure of the bubble, the pressure of the fluid and the vapor pressure:

$$\Pi = \frac{p(r,t) - p_\infty}{p_\infty - p_v} \tag{1.3}$$

The maximum value of Π is obtained at a radius of $(1/\sqrt[3]{4})R_0 \approx 0.63R_0$ by the next expression:

$$\Pi_{max} = \frac{p_{max} - p_\infty}{p_\infty - p_v} = \frac{\left[\frac{R_0^3}{4R^3} - 1\right]^{4/3}}{\left[\frac{R_0^3}{R^3} - 1\right]^{1/3}} \tag{1.4}$$

And occurs at a distance r_{max} of the center of the bubble:

$$\frac{r_{max}}{R} = \left[\frac{\frac{R_0^3}{R^3} - 1}{\frac{R_0^3}{4R^3} - 1} \right]^{1/3} \tag{1.5}$$

When the relationship between the radius and the initial radius is small, the two previous relations give approximately:

$$\Pi_{max} = \frac{1}{4^{4/3}} \left[\frac{R_0}{R} \right]^3 \approx 0.157 \left[\frac{R_0}{R} \right]^3 \tag{1.6}$$

$$\frac{r_{max}}{R} \approx \sqrt[3]{4} \approx 1.59 \tag{1.7}$$

The behavior of Π versus the relationship between the bubble radius and the initial radius $\frac{r}{R_0}$ is represented in the next graphic:

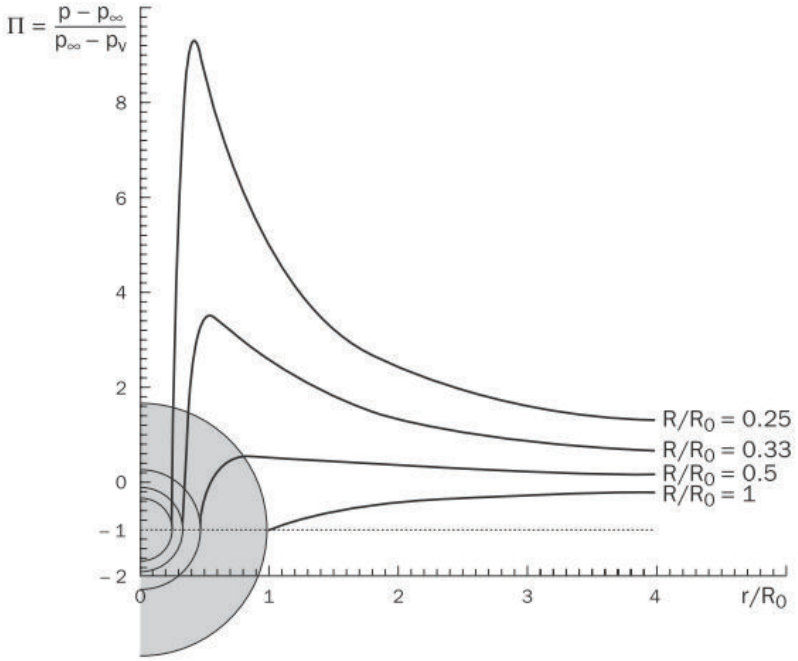


Figure 1.6: Plots of the behavior of the pressure field Π as function of the relationship between the bubble radius and the initial radius r/R_0. (Franc & Michel, 2005, p.41)

1.2.4 TYPES

An analysis of the behavior of the cavitation it is necessary to understand the principal differences between all the types.

For example, in the analysis of the variation of the cavitation number σ_v versus the angle of attack in an NACA 16012 hydrofoil is possible to determine four different forms of cavitation in the foil (Franc & Michel, 2005, p.172-173).

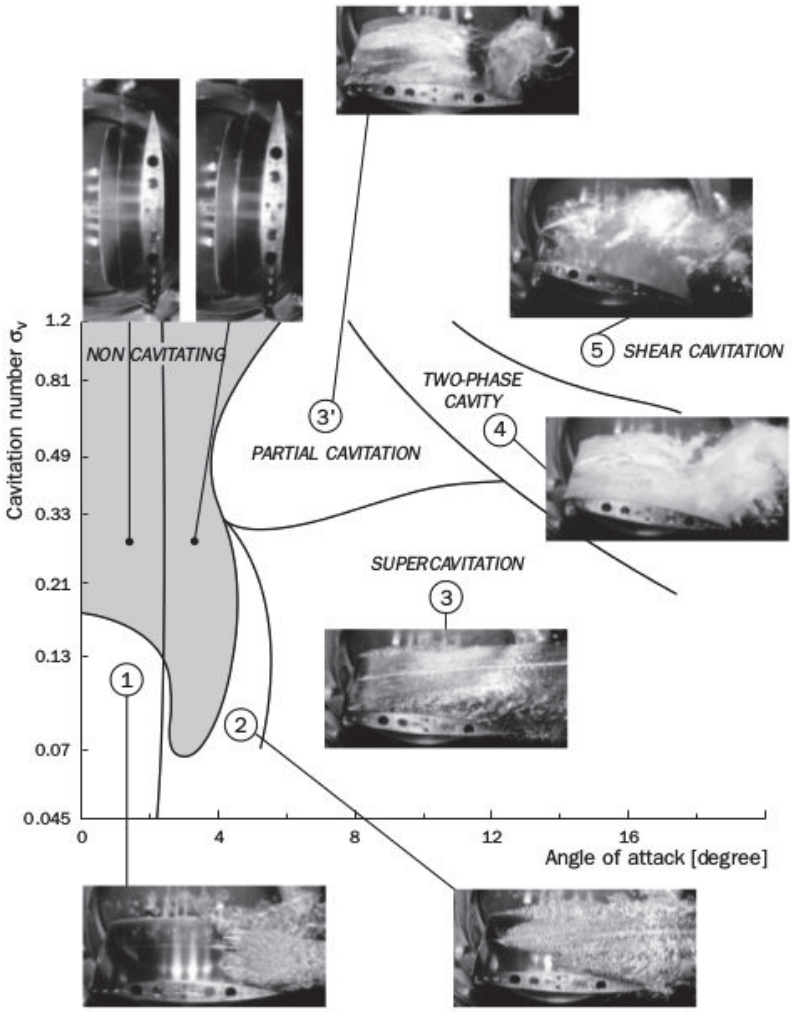


Figure 1.7: Cavitation forms in an NACA 16012 hydrofoil at $Re = 10^6$ in strongly deaerated water. (Franc & Michel, 2005, p.173)

The gray area correspond a zone without cavitation. Zones 1, 2 and 3 correspond a supercavitation, in which the development of it has an observable variation. Zone 3' correspond an intermediate state denominated "partial cavitation". Zone 4 is the two-phase cavity region, in which coexist the partial with the shear cavitation and supercavitation with shear cavitation. Zone 5 correspond at the shear cavitation region, this has high cavitation numbers and angles of attack. We can understand a few more analyzing each shape in the next items, Vortex and sheet cavitation are also studied.

- Sheet cavitation
  Is a region of vapor which remains approximately at the same position relative to the profile or propeller blade. Without diffusion the pressure in the sheet cavity will be close to the equilibrium vapor pressure and the surface of the cavity can be considered as a free surface (Kuiper, 2010, p.37). A 2D sketch of sheet cavity is presented in the next picture:

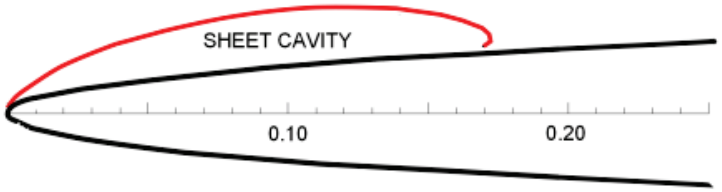


Figure 1.8: Sketch 2D of sheet cavitation. (Kuiper, 2010, p.37)

Is the most frequently observed type on a surface ship propeller. The interface can be stable or not, depending on the flow conditions (SCCMPC, N/A, p.3-4). This form of cavitation on a hydrofoil or propeller blade is usually termed "sheet" cavitation; in the context of pumps it is known as "blade" cavitation (E. Brennen, 1995, p.121). When cavitation is "fully developed", this cavitation is attached completely to the surface, in this case, large-scale cavitation structures are present and form part of the sheet cavities. The next graphic is an example of sheet cavitation (right) and bubble cavitation (left).

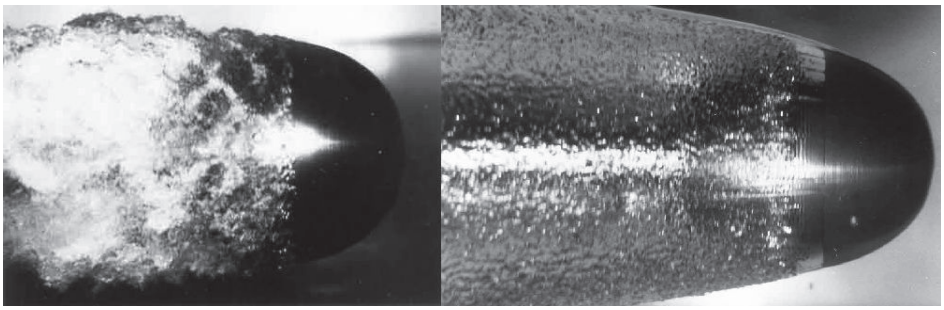


Figure 1.9: Bubble (left) and sheet (right) cavitation example, with velocities of $5.6m/s$ and $10.7m/s$ respectively. The flow is from right to left. (E. Brennen, 1995, p.122)

In some cases, other surprising forms of sheet cavitation appears, for example the case of cavitation on a surface of a biconvex hydrofoil, where cavitation is present in streaks. The next picture include this behavior:

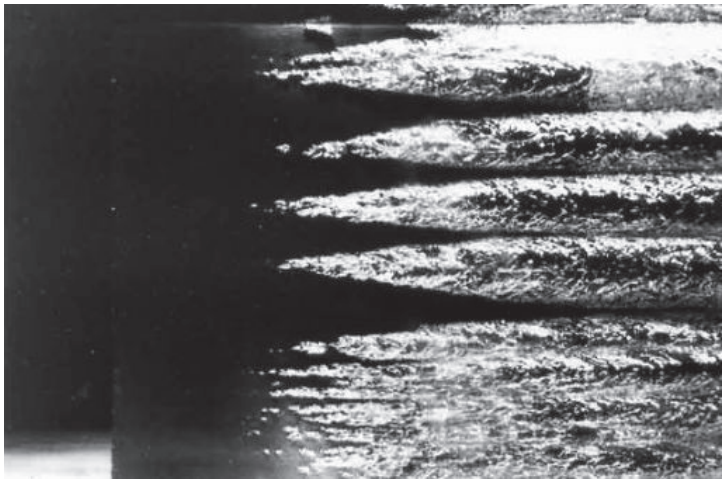


Figure 1.10: Cavitation in streaks on a biconvex hydrofoil at $15.5m/s$ and $\sigma = 0.11$. Flow from left to right, leading edge at approximately $1cm$ from the left-hand edge. (E. Brennen, 1995, p.123)

It is interesting to know that a sheet cavity attached to the leading edge of a blade generally splits into many tiny bubbles in its closure region, this can generate clouds full of small bubbles. Using measurements of bubble population in the wake and assuming that the vaporization rate is equal to the flow rate of bubbles generated, it is possible to estimate the number and size of small scale vapor structures which are potential source of erosion (Franc, 2009, p.1).

Varying the angle of attack generate an unsteady state in the sheet cavitation with a periodic growth and a periodic decline. The sheet cavity grows when the angle of attack increases; the re-entrant jet in this state is absent or weak. When the angle of attack begin to decrease, the cavity stops and the re-entrant flow emerges. The collision of the re-entrant flow begins in the rear end of the cavity. This flow cuts the cavity in two halves and initiates the generation of two clouds in the centerline in which violent implosion occurs (Kuiper, 2010, p.44-46).

- Vortex cavitation
  Vortex cavitation are rotational structures and vortices, like apex vortices, developed along the leading edge of delta wings, hub vortices and tip vortices (Franc & Michel, 2005, p.223).

  Many high Reynolds number flows contain a region of concentrate vorticity where the pressure in the core is smaller than in the rest of the flow. In the mentioned zone, cavitation inception occurs, and with a further reduction of the cavitation number, the entire core of the vortex become filled of vapor (E. Brennen, 1995, p.117).

  The characteristics of a vortex change due to the inception of cavitation. For example, when occurs cavitation, liquid particles situated at $2\mu m$ from the axis are ejected at $0.5mm$. Phase change occurs in a short time, typically smaller than $0.1ms$ (Franc & Michel, 2005, p.224-225).

  When continuous cavitating tip vortices occur at the tips of the blades of a propeller they create a surprisingly stable flow structure (E. Brennen, 1995, p.118), like in the next graphic:

Figure 1.11: Tip vortex cavitation on a model propeller. (E. Brennen, 1995, p.118)

When cavitation develops in a liquid vortex, the geometry together with the pressure and the velocity fields are usually drastically changed. During its life-time, a cavitating vortex generally experiences simultaneously changes in length and in ambient pressure. If the ambient pressure is constant, stretching induces an increase in the rotation rate and hence an increase in the vapor core radius (Franc & Michel, 2005, p.225).

Cavitation can occur in any vortex, like is described in the next graphic, which has a cavitation vortex in the swirling flow in the draft tube of a Francis turbine. Often these

draft tube vortices have a complex patterns of unsteady flow (E. Brennen, 1995, p.118).

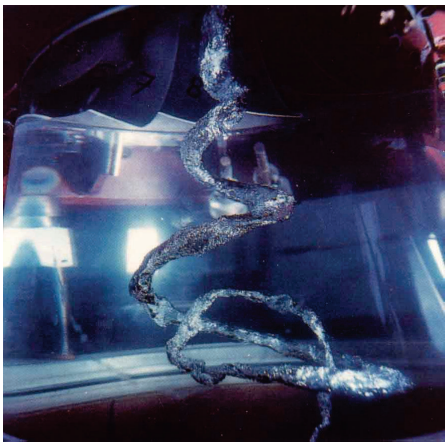


Figure 1.12: Example of a cavitating vortex in the draft tube of a Francis turbine. (E. Brennen, 1995, p.119)

Two models of collapse are expected for a cavitating vortex who returns to the non-cavitating state. An axial mode, especially for vortices ending on solid walls, and a radial mode, which requires viscous dissipation since the rotation rate of the particles at the interface would become infinite as they reach the axis (Franc & Michel, 2005, p.225). The vortices in a turbulent mixing layer or wake will also cavitate, as illustrated in the next figure; a photograph of the separated wake behind a lifting flat plate with a flap (E. Brennen, 1995, p.119).

Figure 1.13: Example of cavitating vortices in the separated wake of a lifting flat plate with a flap. (E. Brennen, 1995, p.119)

Are two main types of cavitating vortex: Axisymmetric and toroidal. Axisymmetric are limited by an external circular cylinder of time-dependent radius $R(t)$. The assumption of a finite external radius is necessary to avoid the singular logarithmic behavior which classically arises in two-dimensional configurations. Toroidal vortices are encountered at the periphery of submerged round liquid jets; they are produced at a regular frequency f. For various purposes and especially for erosion enhancement, it may be of interest to reinforce the strength of the vortices by exciting the jet at the same frequency (Franc & Michel, 2005, p.226-230).

- Partial cavities
  In region of separated flow are common the partial cavities. In hydrofoils, when the cavitation number is further decreased, partial cavities turns into supercavities, because that, partial cavities in hydrofoils can be regarded as an intermediate stage of development of cavitation. In internal flows partial cavitation is considered the ultimate stage because always close on the solid wall, whatever their size (Franc & Michel, 2005, p.131).

Exist a variety of cavitation patterns who can occur in a hydrofoil according to the operating conditions. In the next graphic is present the results of the variation of the angle of attack and the cavitation number maintaining the Reynolds number constant.

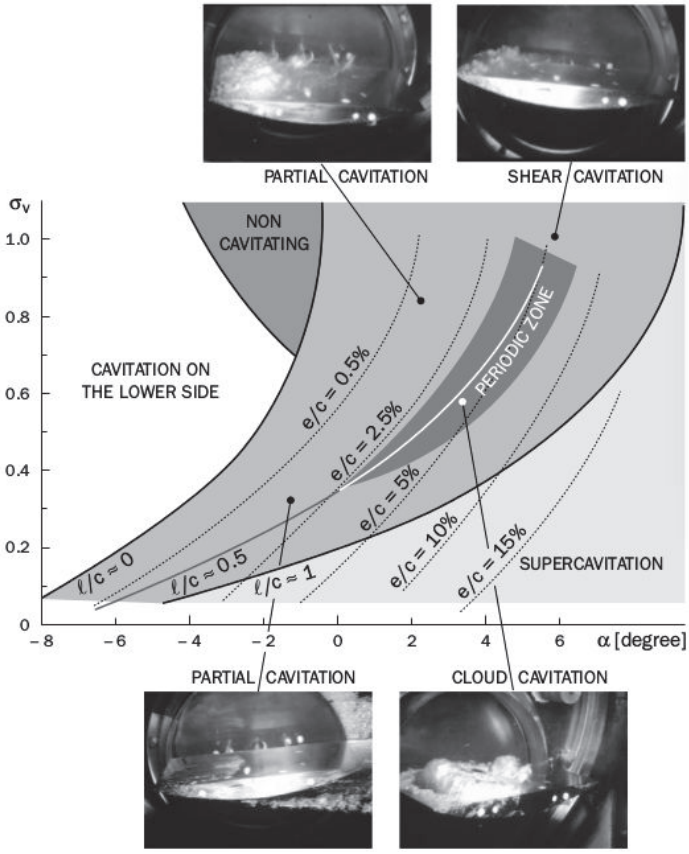


Figure 1.14: Variety of cavitation patterns on the upper side of a *plano-circular* hydrofoil at $Re = 2x10^6$ ($U_\infty = 10m/s$) (Franc & Michel, 2005, p.132)

Where, $\sigma_v = \sigma$ is the cavitation number, α the attack angle in degrees, c the chord length (in this case $200mm$, thickness $20mm$, radius of curvature at the leading edge $1mm$), l the cavity length and e the maximum thickness of the cavity.

When occurs the minimum pressure inside the cavity, the streamlines curvature are redirected to the cavity reattaching the solid wall, causing the division of the surrounding liquid flow into two parts: the re-entrant jet, which travels upstream with a

small quantity of the liquid inside the cavity, and the outer flow, which reattaches to the wall (Franc & Michel, 2005, p.133). The next figure describes this phenomenon.

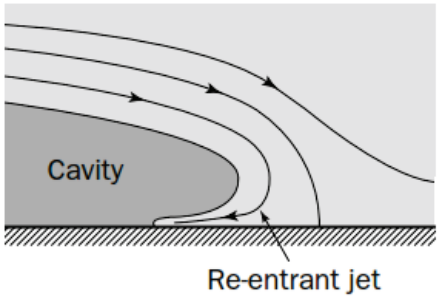


Figure 1.15: Behavior of the cavity closure in a solid wall (Franc & Michel, 2005, p.134)

In a period of oscillation is present partial cavitation in a hydrofoil. Initially the re-entrant jet flow upwards and requires about one third of the period to reach the frontal part of the cavity. Next, a new leading edge cavity grows while the generated vapor cavity shed near the leading edge is sent downstream, loses the shape and collapse quasi with the beginning of the new re-entrant jet. Also, with a variation, the collocation of small obstacle placed on the wall it is possible to prevent the generation of cloud cavitation by stopping the progression of the re-entrant jet (Franc & Michel, 2005, p.141-142). In the next graphic is presented the mentioned phenomenon of generation of re-entrant jets.

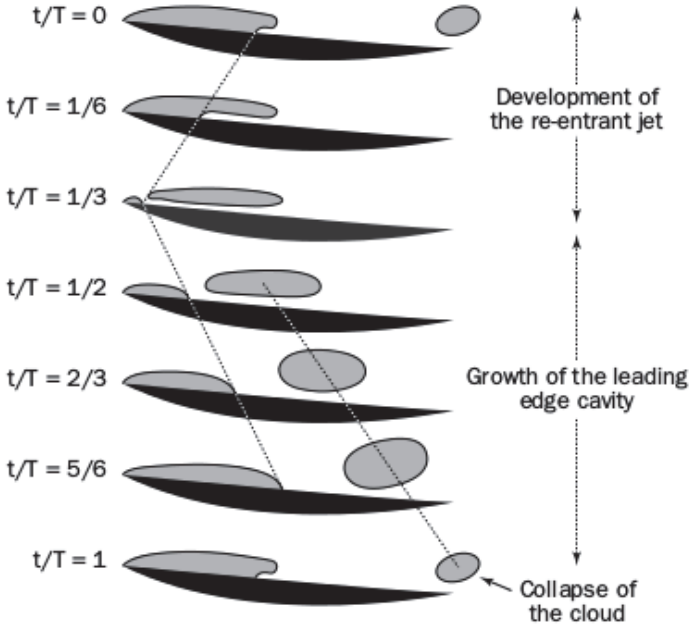


Figure 1.16: Behavior of re-entrant jets in partial cavitation: t: time, T: period (Franc & Michel, 2005, p.142)

- Supercavitation
  When a cavity extend and grow longer because a reduction of the cavitation parameter for very high relative velocities between the liquid and the body, becomes a supercavity (Franc & Michel, 2005, p.97). A continuous vapor-filled cavity is formed, rather than a mass of small individual bubbles. The cavity grows when the angle of attack is increased, when the ambient pressure is reduced, or when the water speed is increased. When the cavity extends beyond the trailing edge of the hydrofoil, the flow is called a "supercavitating" flow (Eisenberg, N/A, p.122-123). This phenomenon decreases the lift on hydrofoils and increases the drag (Franc & Michel, 2005, p.97).
  The next graphic determine visually the difference between a partial cavitation and a supercavitation in a foil:

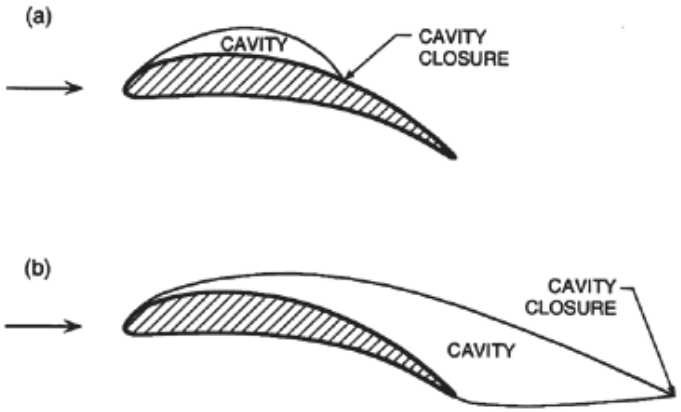


Figure 1.17: Sketch of partial cavitation (a) and supercavitation (b) in a foil.(E. Brennen, 1995, p.124)

Some applications include developing projects of the construction of torpedoes which can reach more than 400 kilometers per hour because a generated cavitation bubble at the nose which reduces substantially the drag that would be present if the object were in direct contact with water (Wikipedia, 2015f). A supercavity in a high-speed body can be maintained in one of two ways: by achieving such a high speed that the water vaporizes near the nose of the body or by supplying gas to the cavity at nearly ambient pressure (Kirschner, Gieseke, & Stinebring, 2001, p.1). Supercavitating foils have been designed for better efficiency, such truncated foils or supercavitating foils (Franc & Michel, 2005, p.97).

The cavity is made of a mixture of vapor and non-condensable gas, the pressure inside is:

$$p_{cav} = p_v + p_g$$

Where, p_{cav} is the cavity pressure, p_v is the vapor pressure and p_g is the partial gas pressure. Cavity pressure is considered as constant in time and uniform throughout the cavity, the shear stress on the cavity is usually negligible. The presence of gas in the cavity is due to diffusion through the interface of gases dissolved in the liquid (Franc & Michel, 2005, p.98).
To realize the modeling and simulation of supercavitating high-speed bodies' two forms

of study are used: Boundary-element and Large-Scale computational fluid dynamics (CFD) (Kirschner et al., 2001, p.4).

Without a special element, such as a step or a sharp edge, a detachment criterion is necessary to predict the location of the cavity detachment. Two main criteria are available:

- Villat-Armstrong criterion
  Cavity must detach tangentially to the solid wall with the condition of constant pressure or constant velocity along the whole free streamline. The detachment point is the point of minimum pressure on the wall calculated by an iterative procedure.
- Laminar separation criterion
  A supercavity detaches at the laminar separation point of the boundary layer. A well-developed cavity always detaches downstream of laminar separation of the boundary layer. The existence of separation is the only opportunity for a cavity to remain attached to the wall and to be sheltered from the incoming flow. If the boundary layer does not separate, the cavity is swept away by the flow and cannot attach to the smooth wall.

It is only in a few special cases that both criteria do not agree. For example, in the case of hydrofoils at very low angle of attack (Franc & Michel, 2005, p.98-101).

In supercavitation, the small gas bubbles produced by cavitation expand and combine to form one large, stable, and predictable bubble around the supercavitating object. The bubble is longer than the object, so only the leading edge of the object actually contacts liquid water. The rest of the object is surrounded by low-pressure water vapor, significantly lowering the drag on the supercavitating object. Modern propellers intentionally induce supercavitation to reap the benefits of lower drag (Sturgeon, 2001, p.3).

The length of a supercavity is one of the most important parameters of the cavity flow, the length of a supercavity increases when the relative cavity underpressure σ_c decreases. To make a calculation of steady supercavity flows different techniques can be used: in the past, non-linear, analytic techniques, for bodies limited by a few straight lines, at the present time, the non-linear, analytic technique remains a reference rather than an operative method to solve practical problems. Linearized, analytic methods were developed to model supercavitating flows around slender lifting bodies (Franc & Michel, 2005, p.108-109).

- Shear cavitation
  Is present in wakes and submerged jets at high Reynolds number and separated regions which develop on foils at large angle of attack. Their inception and development are controlled by their non-cavitating structure, are limited by regions of high shear where vorticity is produced. Coherent rotational structures are formed and the pressure level drops in the core of the vortices which become potential sites of cavitation (Franc & Michel, 2005, p.247).

Developed cavitating shear layers are complex multiphase flows. The continuous flow field is turbulent and characterized by relatively high Reynolds numbers typically greater than 10 000 based on integral length scales. The disperse vapor has a non-spherical and complex geometry and is often not in thermodynamic equilibrium (Lyer & Ceccio, 2002, p.3414).

From an erosion view point, cavitating vortices shed by attached cavities are often considered as particularly aggressive and can cause significant damages when collapsing in regions of adverse pressure gradient. Jet cavitation appears at the periphery of submerged jets, for example: in boat propulsion or in discharge control valves and at the frontier of any separated flow. Axisymmetric jets are rather complicated, several kind of vortices are formed such as toroidal, linear streamwise and helix vortices (Franc & Michel, 2005, p.247-248).

Experimental observations have not revealed which interactions between the vapor and the liquid flow fields constitute the dominant mechanisms responsible for overall flow modification (Lyer & Ceccio, 2002, p.3414). Some experiments demonstrated the influence of water quality and jet velocity and nozzle diameter. In the first case nuclei concentration is clearly correlated to the dissolved air content and that the critical cavitation parameter (σ_{vi}) decreases when the air content is decreased. For smaller jets, the cavitation inception number changes considerably with the air content (Franc & Michel, 2005, p.248). In the next table we can see this phenomenon:

Table 1.1: Influence of water quality in shear cavitation. (Franc & Michel, 2005, p.249)

Dissolved air content (ppm)	σ_{vi} *Jet diameter (mm)*			**Nuclei concentration (per cm³)** *Nuclei size*	
	3.17	*4.76*	*6.35*	*5 μm < R < 10 μm*	*10 μm < R < 20 μm*
14.1	0.16	0.15	0.27	150	25
10.9	0.08	0.11	0.24	74	15
7.7	0.06	0.07	0.23	24	8.2
4.2	0.03	0.06	0.22	5.3	2.1

For a given nozzle geometry, the cavitation inception number is almost independent of the Reynolds number bases on the jet diameter and the exit velocity (Franc & Michel, 2005, p.250).

For generate an analysis of cavitation inception is used a static modeling of the turbu-

lent flow. Based in the equation:

$$\sigma < -C_{\overline{p}} - \frac{p_v - p_{cav}}{\frac{1}{2}\rho V^2} + \frac{p'}{\frac{1}{2}\rho V^2} \tag{1.8}$$

Where σ is the cavitation number, $C_{\overline{p}}$ the main pressure field, p_v the vapor pressure, p_{cav} the cavity pressure, ρ the density, V the velocity, p' the liquid pressure fluctuation. The first term on the right hand side takes into account the mean pressure field of the turbulent flow, the second expresses the static delay to cavitation inception due to the difference between the vapor pressure and the actual critical pressure of the nuclei. The third term expresses the advance to cavitation inception due to the pressure fluctuations (Franc & Michel, 2005, p.251-252).

When cavitation occurs, wakes appear and have three different shapes in agreement of the distance from the wedge, like in the next graphic:

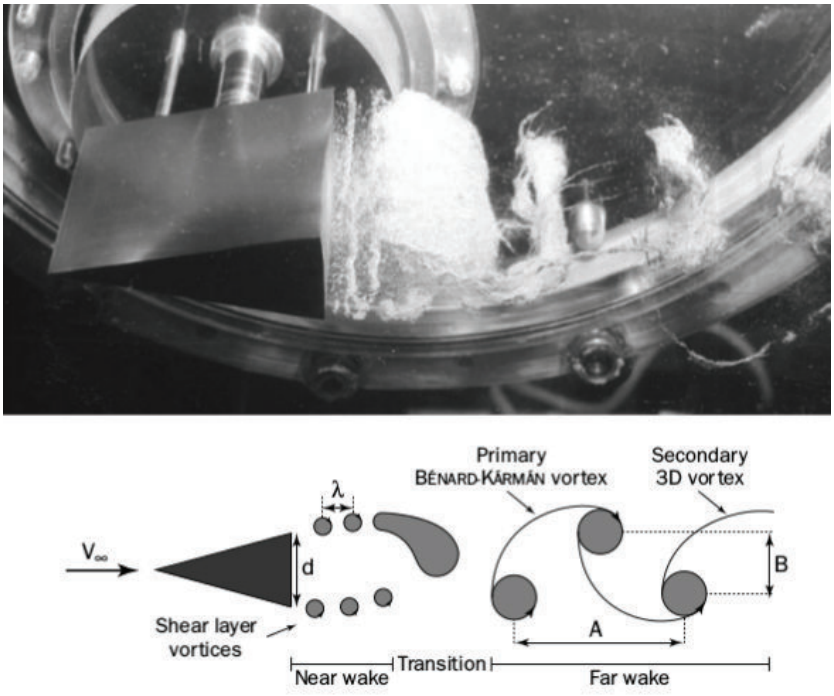


Figure 1.18: Shapes of cavitation wakes in shear cavitation.(Franc & Michel, 2005, p.256)

In the near wake, with a length of about $0.7d$, small-scale vortices are periodically shed in the two shear layers which originate in the wedge trailing edges. The far wake is made up of the classical 2D BÃ©nard-KÃ¡rmÃ¡n vortices, which are made of a repeating pattern of swirling vortices. They are connected together by streamwise 3D vortex filaments. This phase initiates at a distance of $0.9d$ to $4.4d$ from a moderately developed cavitation to a small values of the cavitation parameter respectively. The transition region is made up of a two-phase mixture (Franc & Michel, 2005, p.257).

1.2.5 EFFECTS

Some of the main effects of the cavitation in hydraulics are: alteration of the performance of the system, appearance of additional forces, production of noise and vibrations, wall erosion, and others. Commonly a certain degree of cavitation development is allowed due to the excessive financial charges to eliminate all the phenomenon. Cavitation also has positive applications for example: cleaning of surfaces, dispersion of particles, production of emulsions, electrolytic deposition, therapeutic massage and bacteria destruction, and others (Franc & Michel, 2005, p.6). In the next subsections are discussed some of the mentioned effects:

- Lift & drag
  In the case of hydrofoils, does not exist any variation between the lift and drag prior the cavitation at a constant flow speed. As cavitation develops, the lift decreases and the drag rises, such as the next graphic: (Eisenberg, N/A, p.124).

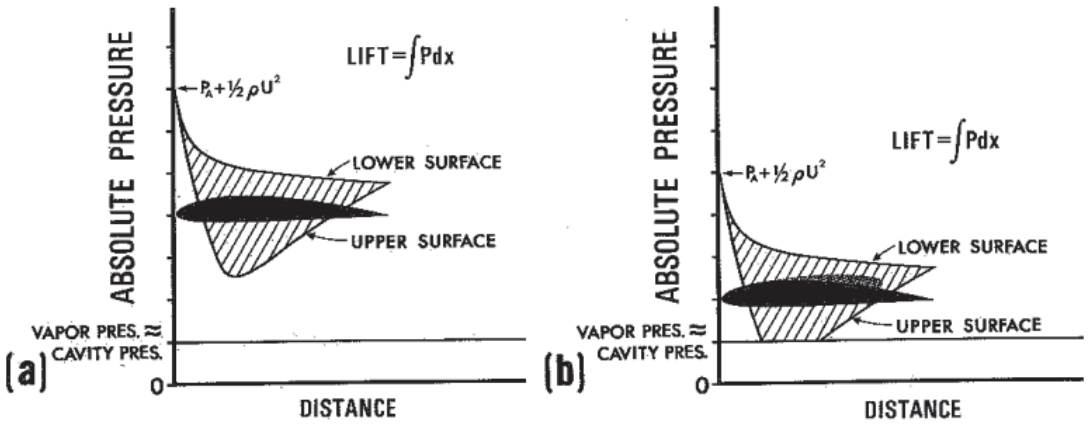


Figure 1.19: Effect of cavitation to lift and drag in a hydrofoil. (a)Prior cavitation, (b)cavitation developed. (Eisenberg, N/A, p.124)

The lift is represented as the area between the lower surface and the upper surface. When the absolute pressure start to decrease, this area get down. The particular behavior initiates when cavitation stars, in this moment the area below the hydrofoil disappears, such as the graphic (b), i.e., the lift decrease. The resultant pressure distribution is such as to cause an increase in drag (Eisenberg, N/A, p.124-125).

- Noise
  When cavitation happens, the collapse of the bubbles causes noise, which is the result of shock waves generated upon bubble collapse (Eisenberg, N/A, p.125). Cavitation noise is the consequence of the momentary large pressures when the bubble is highly compressed. The crackling noise is one of the most evident characteristics of the phenomenon. From a record of the noise of the individual cavitation bubbles in a flow, could find two pulses with high pressure, which correspond at the first and second collapse; rebound, of the bubbles (C. Brennen, 2005, p.142-148), like in the next figure:

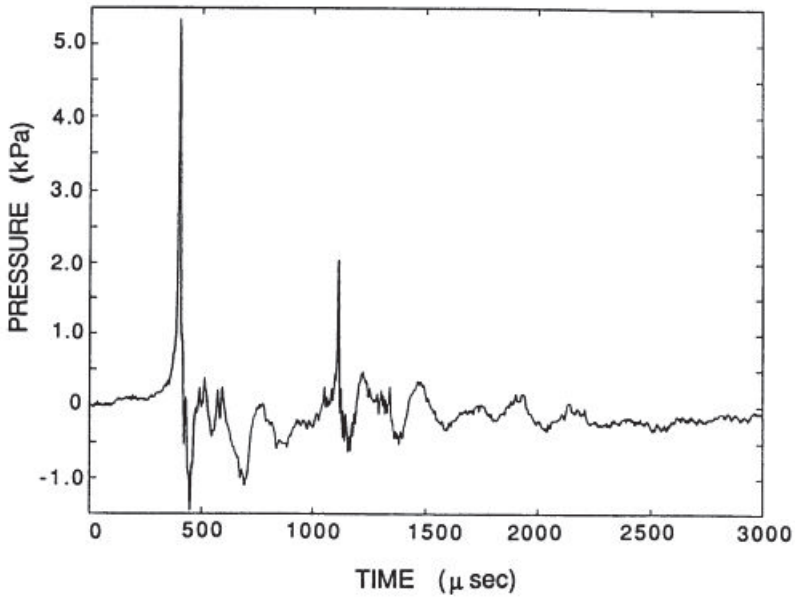


Figure 1.20: Acoustic signal of the noise produced by cavitation from a single bubble collapse. (C. Brennen, 2005, p.146)

The first collapse occurs at approximately 450[μs] and the second one, which is the rebound of the first one, occurs at a value like 1100[μs].

- Cavitation damage
  Cavitation bubble collapse is a violent process that generates highly localized, large-amplitude shock waves and microjets. When the collapse occurs close to a solid surface, is generated surface stresses which could induce fatigue failure (C. Brennen, 2005, p.136-138). In the next figure we can see the effect of the cavitation in the blades of a Francis turbine (a) and mixed flow pump impeller (b).

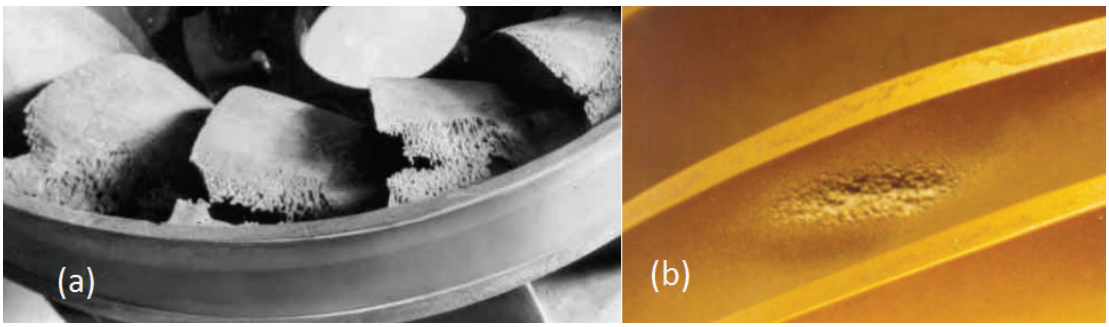


Figure 1.21: Examples of the cavitation damage, (a) in the blades of a Francis turbine and (b) mixed flow pump impeller. (C. Brennen, 2005, p.137)

Damage arises during the cavity collapse by two mechanisms: large shock waves emitted in the liquid and due to an asymmetric collapse (Caupin & Herbert, 2006, p.1013-1014).

- Luminescence
  Cavitation luminescence would be a better description for emissions that result from the cavitation generated by hydrodynamic flow, liquid impact, laser and spark discharge, and the collapse of vacuums, colloquially these are often termed "sonoluminescence" (Leighton, Farhat, Field, & Avellan, 2003, p.44).

  During the collapse, emission of light is observed, this is believed to be due to the extremely high temperatures and pressures on the bubble. The phenomenon is caused by the compression and adiabatic heating of the non-condensable gas in the collapsing bubble that cause in some cases, temperatures of thousands of Celsius degrees in really short times, like fractions of microseconds (C. Brennen, 2005, p.149). Measurements of the volume of gas luminescing show that luminescence takes place when the gas volume has been compressed to less than 10[%] of its initial volume. The luminescence shows a definite structure with some brighter regions, which indicates inhomogeneous conditions (Leighton et al., 2003, p.46).

- Erosiveness
  The phenomenon of erosion could be observed in cases like cavitation on hydroturbines. To study it, three types of devices are used: rotating disk, hydrodynamic tunnel and vibratory device (Khurana, Singh, & Singh, 2012, p.173).

  One of the reasons because cavitation has the reputation of be erosiveness its late detection, the amount of nuclei at model scale is nearly always too low and when cavitation is observed at model scale it may has already a significant size at full scale (Kuiper, 2010, p.34-35). Up to 90[%] of hydro-turbines suffer cavitation damage to some extent (Khurana et al., 2012, p.173).

  To predict the cavitation erosion are used three principal techniques: empirical correlations, simulation techniques using special test and analytical methods. The last method is still in development and represent a real challenge to research workers because requires extensive research efforts (Berchiche, Franc, & Michel, 2002, p.601).

  The erosion damage depends primarily on the mean amplitude $\bar{\sigma}$ of the impact loads (σ_x) relative to the yield strength σ_y and ultimate tensile strength σ_u of the material, three cases may occur (Franc, 2009, p.2-2):

  - $\sigma_x < \sigma_y$: No damage
  - $\sigma_y < \sigma_x < \sigma_u$: Initial progressive hardening without any mass loss and then materials rupture.
  - $\sigma_x > \sigma_u$: Since the beginning of exposure to cavitation mass loss appears.

  An example of an evolution of a surface when cavitation occurs is presented in the next figure:

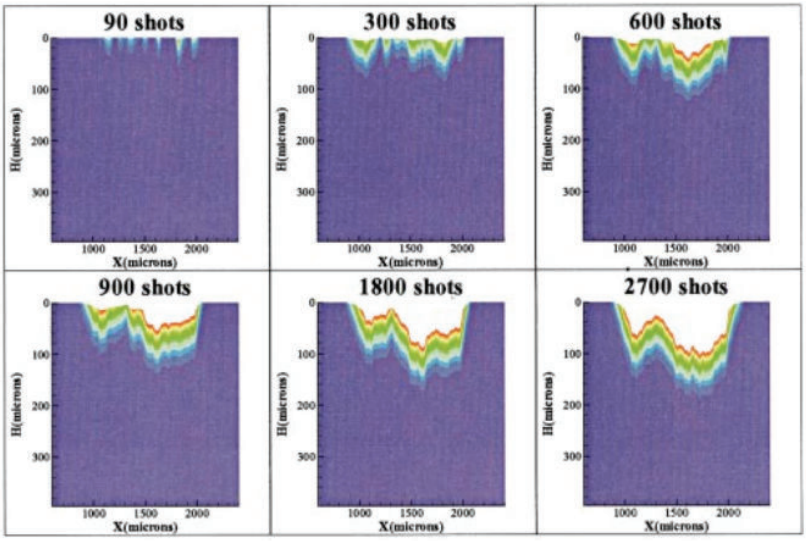


Figure 1.22: Computed evolution of the strain field on a cross section of the material and of the shape of the eroded surface (Berchiche et al., 2002, p.605).

During the incubation period, the pit depth results only of plastic deformation. Once the rupture strain is reached on the surface, mass loss occurs (Berchiche et al., 2002, p.605). With the increment of the shots the wear in the surface is bigger, e.g. with a number of 2700 shots the erosion could reach approximately 150 microns.

1.3 NUMERICAL SIMULATION WITH FREE AND OPEN-SOURCE SOFTWARE

In this thesis the use of free and opensource software is a fundamental axis. These class of software is applied since the writing process to the performed simulations.

The explanation of each software used is presented below:

- Python & Spyder
  To start the programming process in the present thesis is used the open source software Python and its scientific environment called Spyder.

Python is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale (Wikipedia, 2015d). In this thesis is used the version 2.7.5 of python to generate the code.

Spyder is a Scientific Python Development Environment distributed in open source format, for the present thesis is used the version 2.2.5 of the software. Spyder integrates NumPy, SciPy, Matplotlib and IPython, as well as other open source software (Wikipedia, 2015e). This represent an advantage at the moment of generate python code with the facilities of see the object inspector, variables and files explorer, console, script section and more.

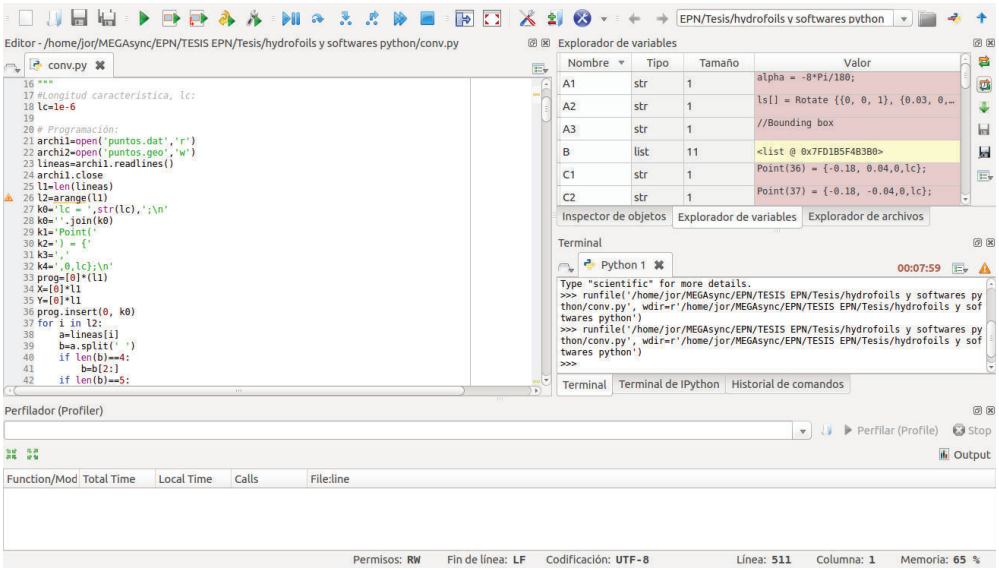


Figure 1.23: Example of use of Spyder to generate the code for Gmsh.

In this thesis Spyder was used to create the files used in Gmsh. These files generate the mesh of the hydrofoils. The appendix B includes the code generated in Spyder.

- Blender
  To obtain the points of the shapes of Francis - 99 and *Hidroagoyán* turbines is used the software denominated Blender.

  Blender is the free and open source 3D creation suite. It supports the entirety of the 3D pipeline—modeling, rigging, animation, simulation, rendering, compositing and motion tracking, even video editing and game creation. Is cross-platform and runs equally well on Linux, Windows and Macintosh computers (Blender™, 2015).

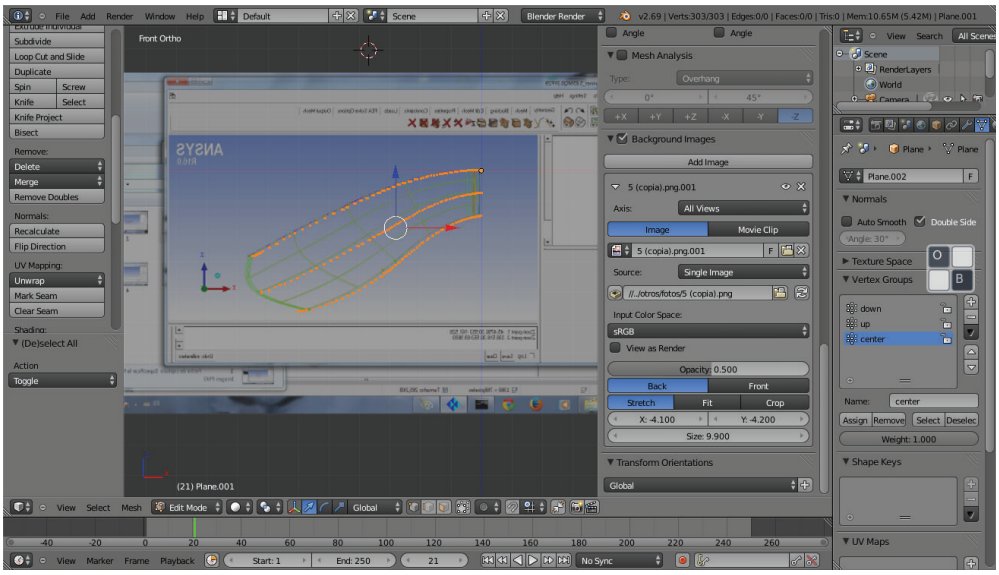


Figure 1.24: Example of use of Blender to obtain the points of the profile shape of Francis-99 runner.

The use of this software to generate the points of the blade sections is presented in the section 2.1.

- Gmsh
  To generate all the meshes in the present thesis is used the software Gmsh, are used all the principles studied in the section 2.2.

  Gmsh is an open source software used to generate finite element meshes. Its design goal is to provide a fast, light and user-friendly meshing tool with parametric input and advanced visualization capabilities. Gmsh is built around four modules: geometry, mesh, solver and post-processing. The specification of any input to these modules is done either interactively using the graphical user interface or in ASCII text files using Gmsh's own scripting language (Geuzaine & Remacle, 2015).

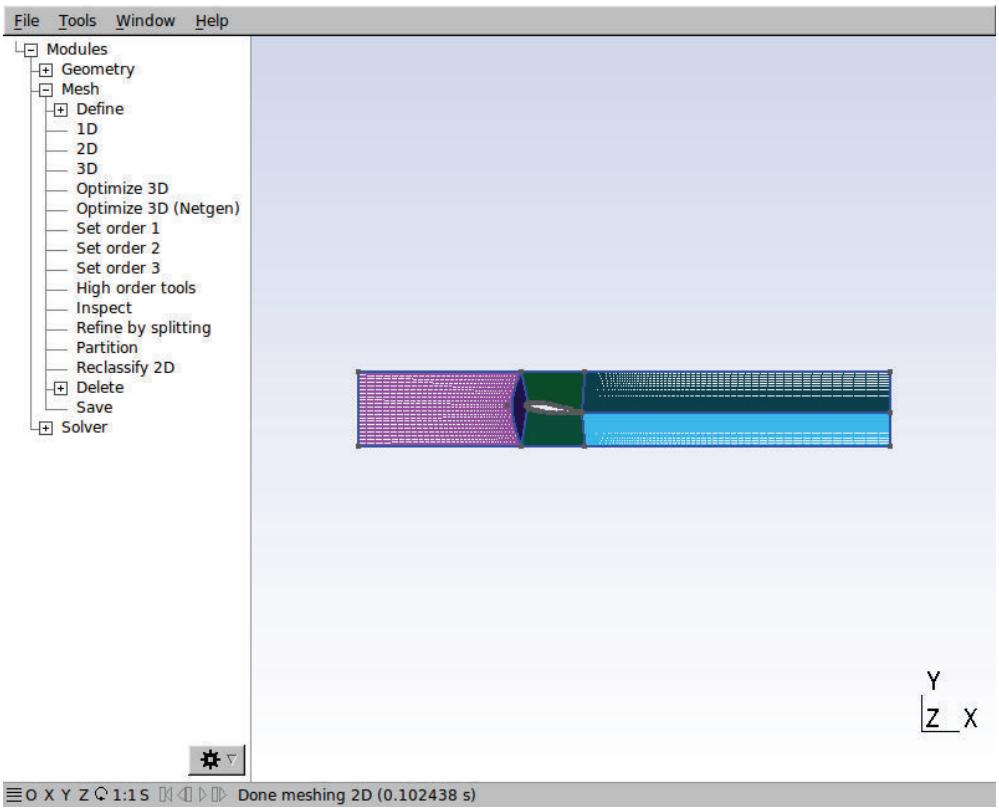


Figure 1.25: Example of use of Gmsh to generate a 2D mesh of a hydrofoil.

The spyder-generated files to be used in Gmsh as points, lines and the mesh-description are included in the appendix C.

A small Gmsh programming tutorial is included in the appendix D

- OpenFOAM
  To generate all the simulations in the present thesis is used the software called OpenFOAM (Open Field Operation and Manipulation).

  OpenFOAM CFD Toolbox is a free, open source CFD software package which has a large user base across most areas of engineering and science. OpenFOAM includes a lot of solver applications, for example: incompressible flows, multiphase flows, combustion, conjugate heat-transfer, compressible flows, particle methods and others (OpenFOAM-Fundation, 2015).

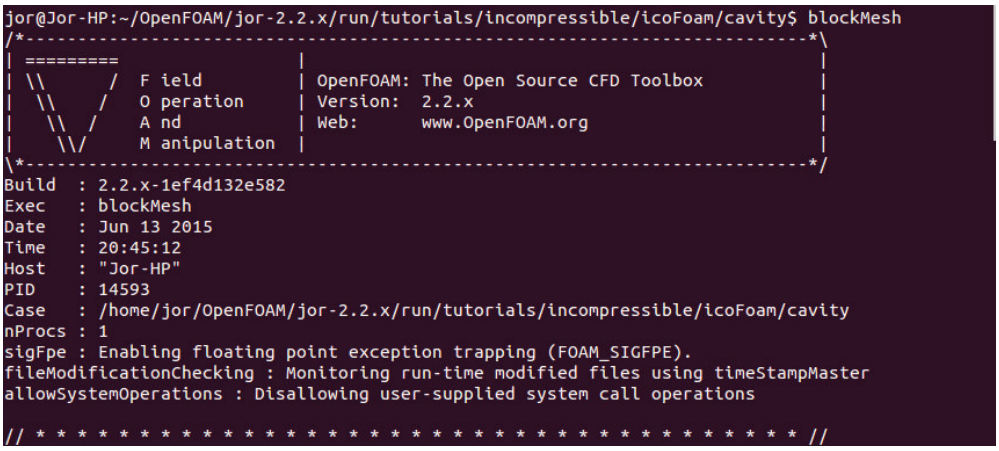


Figure 1.26: Example of use of OpenFOAM in an incompressible cavity tutorial.

Because its condition of open-source software is possible: to customize the code of the solvers and extend all program functionalities, to include generated meshes from others software and to generate the graphic visualization with ParaView; the graphical user interface to see results.

- Advantages (Wikipedia, 2015c)
  - Friendly syntax for partial differential equations
  - Unstructured polyhedral grid capabilities
  - Automatic parallelization of applications written using OpenFOAM high-level syntax
  - Wide range of applications and models ready to use
  - Commercial support and training provided by the developers
  - No license costs
- Disadvantages (Wikipedia, 2015c)
  - Absence of an integrated graphical user interface
  - Programmer's guide does not provide sufficient details. The learning curve very steep

- ParaView
  To make the visualization of the results of the OpenFOAM analysis is used the software denominated ParaView.

  ParaView is an open-source, multi-platform data analysis and visualization application. ParaView users can quickly build visualizations to analyze their data using qualitative and quantitative techniques. The data exploration can be done interactively in 3D or programmatically using ParaView's batch processing capabilities. ParaView was developed to analyze extremely large datasets using distributed memory computing resources. It can be run on supercomputers to analyze datasets of petascale as well as on laptops for smaller data. ParaView is an application framework as well as a turn-key application (ParaView, 2015).

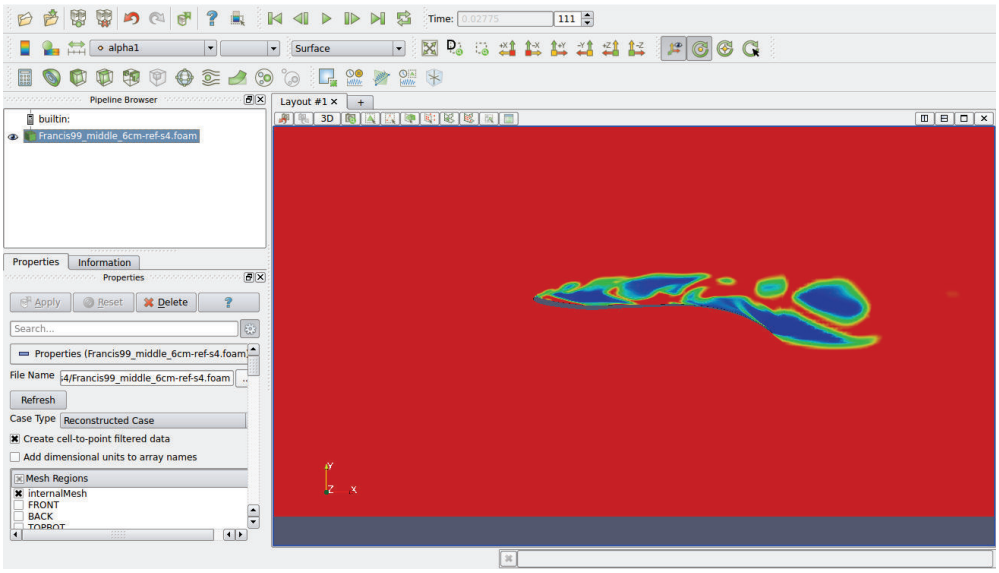


Figure 1.27: Example of use of ParaView in the visualization of the results of cavitation in a hydrofoil.

2 METHODOLOGY

This section describe the methodology used in the analysis of a hydrofoil. In the first part is described the form to obtain the geometry profile. The second part describe the process of meshing. The third part its conformed by the explanation of the used mesh in the thesis and the form to generate the files in Gmsh. Finally is include: a description of the CFD solver with verification of the behavior of the mesh varying parameters, the behavior of the cavitation varying the blade section and a presentation a new type of mesh to understand better the combined effect of cavitation in turbines.

To initiate with the analysis of the cavitation in a hydrofoil it is necessary to introduce a methodology that includes all the relevant process involved.

The next graphic present the used methodology in the present thesis:

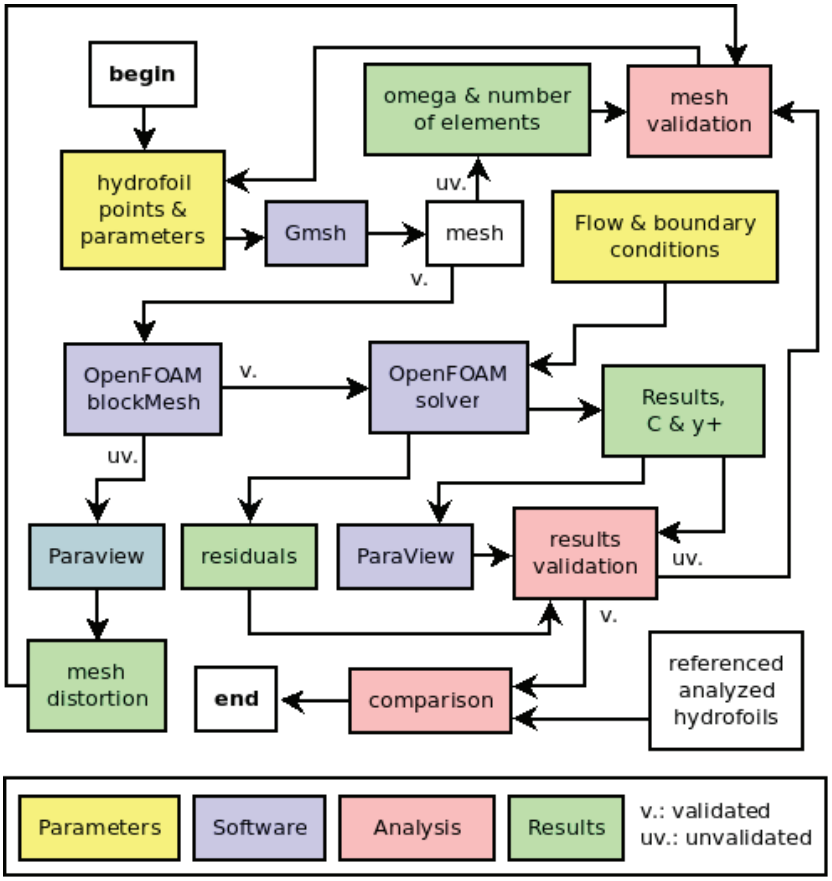


Figure 2.1: Methodology used to realize the analysis process of cavitation in a hydrofoil, from the points to the comparison with other analyzed profiles.

This methodology begins with the hydrofoil points that have to reproduce a validated mesh. The process of generation of the mesh is in charge of Gmsh software, like is described in the subsections 1.3 and 2.3. Continue with the transformation of the mesh to OpenFOAM format and the validation of the distortion analysis results obtained in ParaView. The mesh is resolved by the cavitation solver, sections 2.6.2 and 2.6.4

respectively. From OpenFOAM are obtained the residuals plots, the results, courant numbers and Yplus numbers to be analyzed by the user to confirm the validation of the mesh. The modification of the mesh correspond a non-validated results or a search of better accuracy. In ParaView is realized also the verification of the behavior of the results: vapor fraction, pressures, dimensionless numbers, etc. The comparison is made by referenced analyzed hydrofoils as well as the other profiles analysis.

2.1 GEOMETRY PROFILE

The process to obtain the geometry profile of Francis-99 turbine include the description of the shape and the steps to obtain it with their points. This process is explained below:

2.1.1 INITIAL PROFILE

In the next figure is presented the runner to be analyzed. This runner is from Francis-99 turbine and by it characteristic of open-knowledge was downloaded of the web site of the project (NTNU & LTU, 2015b) in ANSYS ICEM® format.

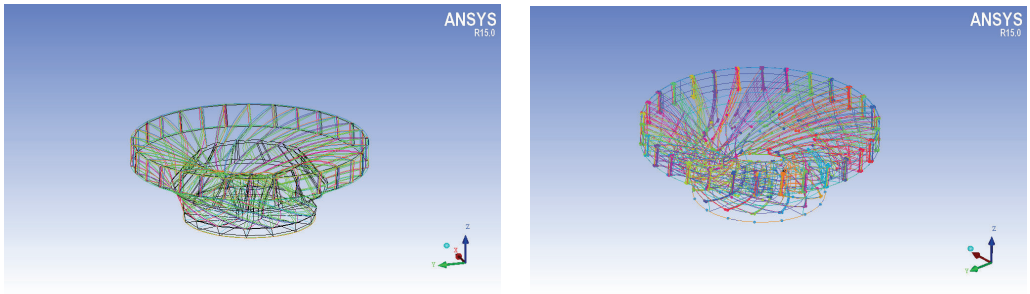


Figure 2.2: Modeled runner in ANSYS ICEM® of Francis-99 project (NTNU & LTU, 2015b).

2.1.2 OBTAINING THE SHAPE

Turbine runners have specified profiles for its blades. These blades probably are patented and have a different shape from the results that become from the theoretical approximation, because that, is necessary to find a way to obtain the hydrofoil shape for its analysis. The process to obtain the profile shape of the blade for Francis-99 runner is not so complicated because it is obtained by a 3D mesh realized in ANSYS ICEM CFD® (NTNU & LTU, 2015b).

The process begins with the decomposition of the runner mesh into a simple shape of the lower and upper blade section with the external profile. In the next graphics is included this process:

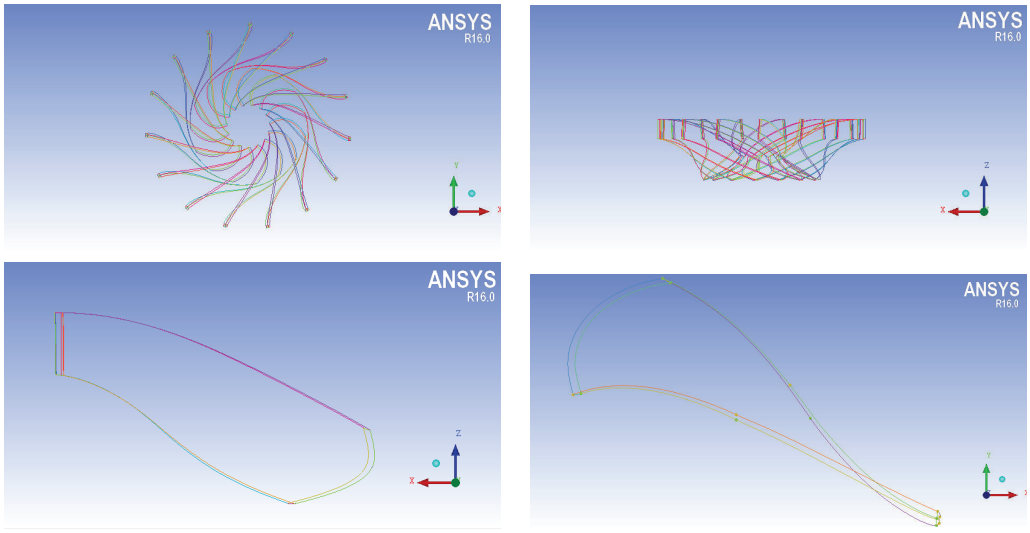


Figure 2.3: Process of obtaining of the shape in ANSYS ICEM® of Francis-99 project (NTNU & LTU, 2015b).

The decomposition continues with the elimination of non used profiles to obtain two more graphics which will be the base to generate the points in Blender.

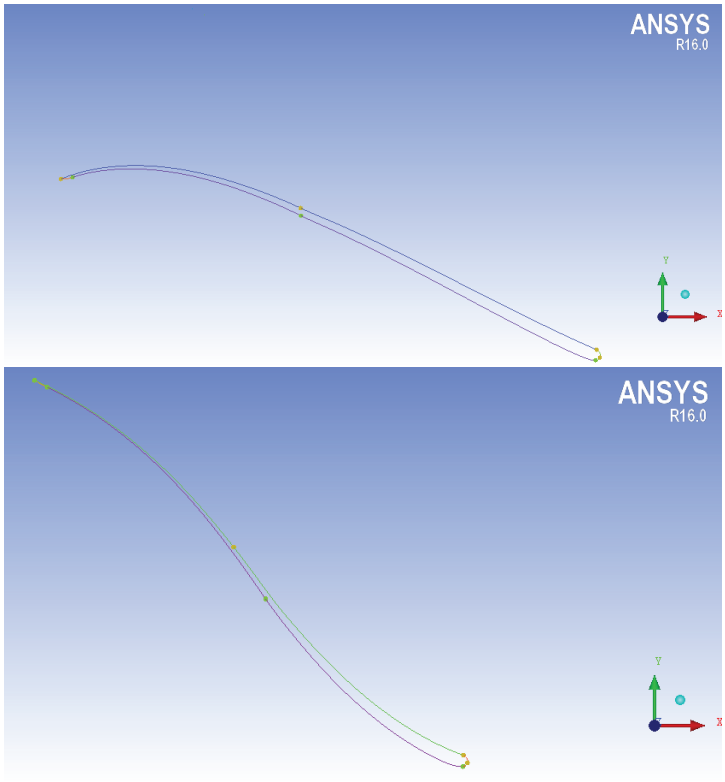


Figure 2.4: Shape of the upper (up) and lower (down) blade sections in the modeled runner in ANSYS ICEM® of Francis-99 project (NTNU & LTU, 2015b).

2.1.3 HYDROFOIL POINTS

After having the shape of the hydrofoils it is necessary to define the points. The objective is to reach a file denominated "puntos.dat" which contain a text file with the x and y coordinates of the hydrofoil.

The process begins with the obtained shape and its dimensions, continue with a posterior analysis in a software CAD. Next is present the obtained points of the profiles in Blender.

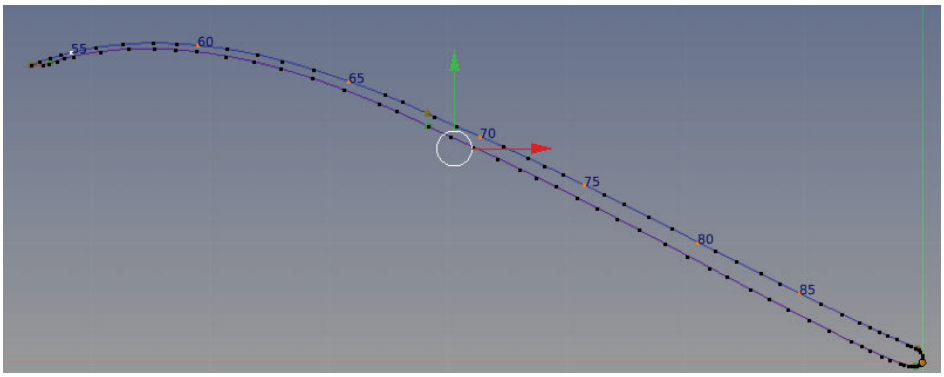


Figure 2.5: Use of Blender to obtain the coordinates of the profile shape.

The points of the upper, medium and lower blade sections are obtained with blender, using the obtained graphic of ICEM® and generating the profile points.

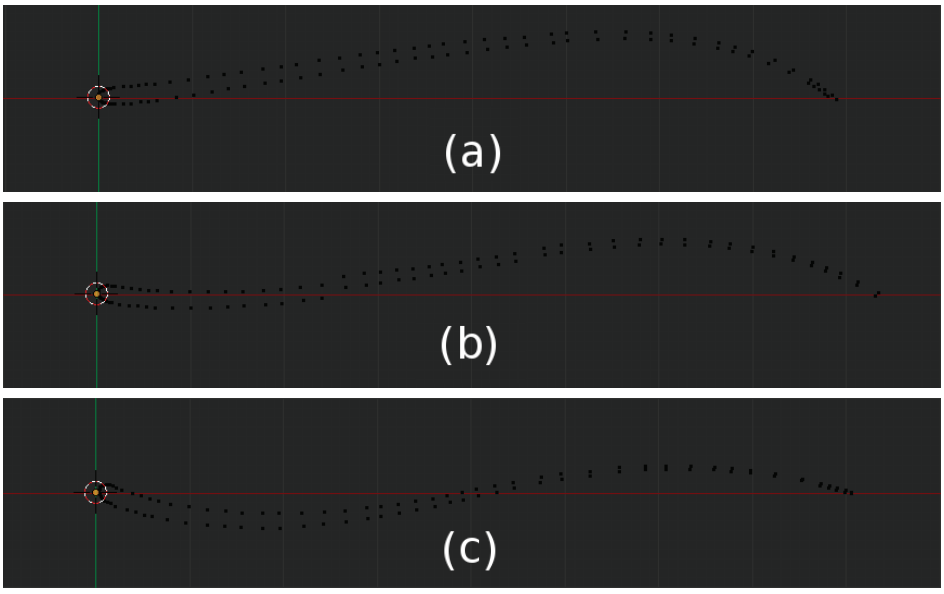


Figure 2.6: Upper (a), medium (b) and lower (c) rotated section shapes of Francis-99 blades modeled in Blender.

2.2 MESHING

For realize the evaluation of the behavior of the phenomenon, the computational method of the resolution of the differential equations by computer is performed thanks to the meshing system. This concerns the classification of the meshing, its adaptation and improvement.

2.2.1 CELL SHAPES

The meshing system could be in two dimensions or three dimension, each one could be structured or unstructured. For two-dimensional cell the utilized shapes are the triangles and quadrilateral, for three-dimensional are used tetrahedrons, pyramids, triangular prims or hexahedron. These forms are presented in the next graphic:

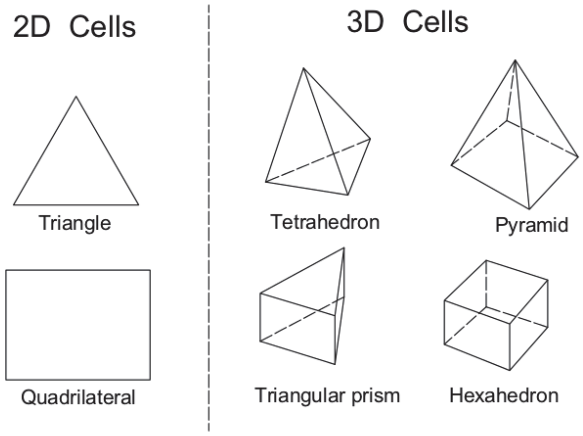


Figure 2.7: 2D and 3D cells commonly used for meshing.

2.2.2 CLASSIFICATION

The shape of the mesh could be structured or unstructured. Visually the difference is clearly differentiable.

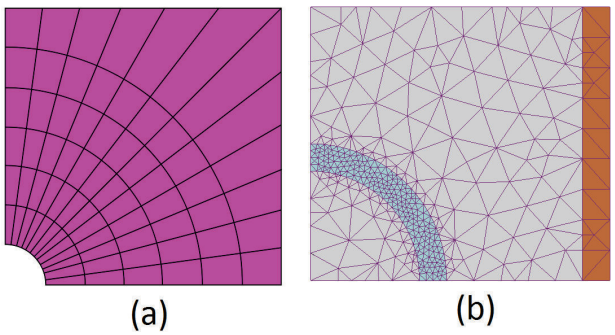


Figure 2.8: Classification of mesh: Structured (a) (Slffea & Mysid, 2006) and unstructured (b) (Zureks, 2007).

Structured meshes are identified by regular connectivity with the use of quadrilaterals in 2D and hexahedrons in 3D, unstructured meshes have irregular connectivity. Structured meshes presents several general advantages than unstructured meshes (Chawner, 2013):

- Less computation time and memory usage.
  A system could be filled with a less number of hexahedrons than tetrahedrons. Each hexahedron can be decomposed into 5 tetrahedron that share it edges.

- Resolution
  To obtain accurate CFD solutions is better to generate high quality cells with hex grid than tetra. Hex grids are easily generated with high aspect ratio in comparison of tetra grids that have to be stretched.

- Alignment
  When the grid is aligned with the predominant flow direction, solvers can produce more accurate results and converge better. Alignment is almost achieved implicitly because grid lines follows the contour of the geometry

- Definable normals
  If there is a well-defined computational direction normal, the application of boundary conditions and turbulence models work well. Transverse normals are easily defined in a structured grid.

2.2.3 IMPROVEMENT

In the evaluation of a CFD simulation it is necessary to improve the mesh to generate a good simulation. To do that some dimensionless numbers are used:

- It is important to capture the boundary layer near the wall property. In turbulent flows, the parameter who helps to do this labor is the dimensionless value denominated y-plus, y^+

$$y^+ = \frac{u_* y}{\nu} \tag{2.1}$$

Where u_* is the friction velocity nearest the wall, y is the distance to the nearest wall and ν is the local kinematic viscosity of the fluid (CFD-Online, 2015b).
The friction velocity is defined as:

$$u_* = \sqrt{\frac{\tau_w}{\rho}} \tag{2.2}$$

The wall shear τ_w can probably not be determined until after a simulation has been completed, so it is usually necessary to estimate a value, and then check after a simulation is complete (CFD-Online, 2015c).

- A number that is important to evaluate the mesh quality is the omega number Ω, which is the relationship between the number of elements NE and the number of nodes ND (Hidalgo, Luo, & Yu, 2014, p.4).

$$\Omega = \frac{NE}{ND} \tag{2.3}$$

- In the moment to solve partial differential equations (PDEs) a criterion to determine the stability is the Courant-Friedrichs-Lewy condition, known commonly as Courant number, C. This is defined by the ratio between the time interval into the residence time in a finite volume. This is one used condition to convergence in which a big time step will produce incorrect results (Wikipedia, 2015a). For one-dimensional case has the form:

$$C = \frac{\Delta t}{\frac{\Delta x}{u}} = \frac{u \Delta t}{\Delta x} \leq C_{max} \tag{2.4}$$

Where, Δt is the time step, Δx is the length interval and u is the velocity.

The value of C_{max} changes with the method used to solve the discretized equation. If an explicit (time-marching) solver is used then typically $C_{max} = 1$. Implicit (matrix) solvers are usually less sensitive to numerical instability and so larger values of C_{max} may be tolerated (CFD-Online, 2015a).

In the two-dimensional case, the condition becomes:

$$C = \frac{u_x \Delta t}{\Delta x} + \frac{u_y \Delta t}{\Delta y} \tag{2.5}$$

Where, the interval lengths, Δx and Δy have not to be the same value. This can be used in order to somewhat optimize the value of the time step for a particular problem, by varying the values of the different interval in order to keep it not too small (CFD-Online, 2015a).

2.2.4 ADAPTATION

Exist some method to modify an existing mesh so as to accurately capture flow features. The principal objective is to improve resolution of flow features without excessive increase of computational work (CFD-Online, 2015e).

- r-Refinement
  Is the modification of the mesh resolution without changing the number of nodes or the connectivity. The increase of the resolution is reached by moving the grid points in regions of activity. The movement of the nodes is realized by trying the mesh as an elastic solid and solving the system of equations on it. Solving process deforms the original mesh to reach the searched-one. The next graphic describes the mentioned process:

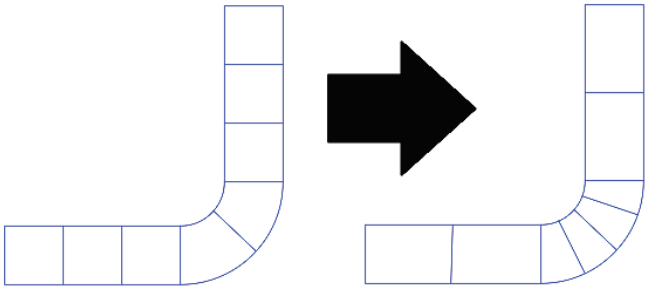


Figure 2.9: Process of r-refinement method (CFD-Online, 2015e).

- h-Refinement
  Is the modification of the mesh resolution by a change of the mesh connectivity doing a subdivision. New points are added in the centroid, there forms new child cells, e.g., for 2-D quadrilaterals or triangles, a new point added get 4 and 3 new child cells respectively. The overall mesh topology remains the same, but subdivision increases the number of points and the number of cells. The next graphic describes this process:

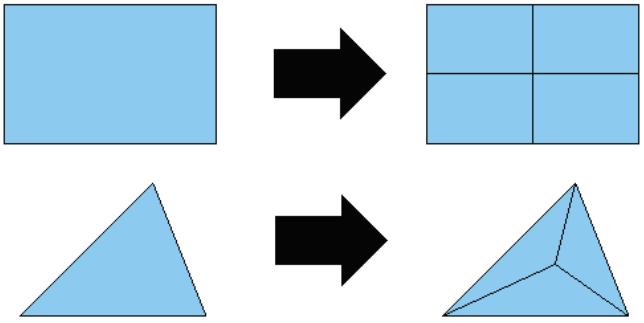


Figure 2.10: Process of h-refinement method (CFD-Online, 2015e).

When h-refinement is performed is important to know about hanging nodes. These types of nodes are created when a cell is divided and the neighboring face not. The

node not belong properly to both of the cells, the node "hangs" on the face and the cells become an arbitrary polyhedron. The next image represents the "hanging node" creation:

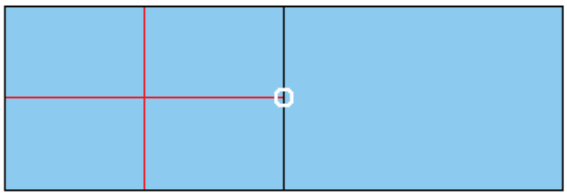


Figure 2.11: Process of generation of "hanging nodes" in an h-refinement method (CFD-Online, 2015e).

The process of h-refinement could be:

- Isotropic, when new points are added in both directions (x and y).
- Anisotropic, when the subdivision is made in one predominant direction.

- p-Refinement
  Is the process to increase the resolution by increasing the order of accuracy of the polynomial of each cell or element. To make it possible is necessary sensing parameters referred as "error indicators", which detect the regions of refinement. The next graphic describes the difference between p-refinement (increase of the polynomial degree) and the h-refinement (addition of nodes) between an original six-node mesh.

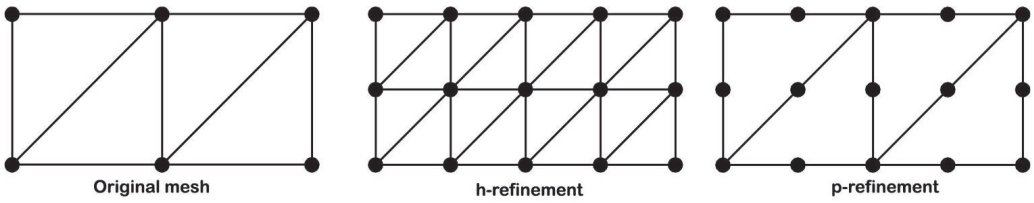


Figure 2.12: Difference between h-Refinement and p-Refinement (Timbobong, 2015).

At the moment to make the refinement is useful to know what area is the most important to evaluate. In some cases the division of the cells it is really necessary, but other cases requires the agglomeration of smaller cells into a larger one. All this depend of the necessity of the evaluation.

2.3 MESH DESCRIPTION

The generated mesh must be realized to satisfy some requirements:

- Hydrofoil with the required angle of inclination.

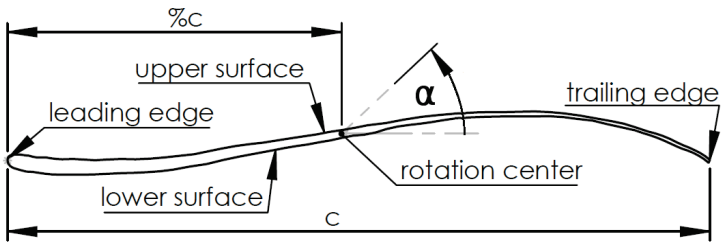


Figure 2.13: Hydrofoil geometry description.

- Definition of the computation domain: inlet, outlet and extrusion values, like the next graphic:

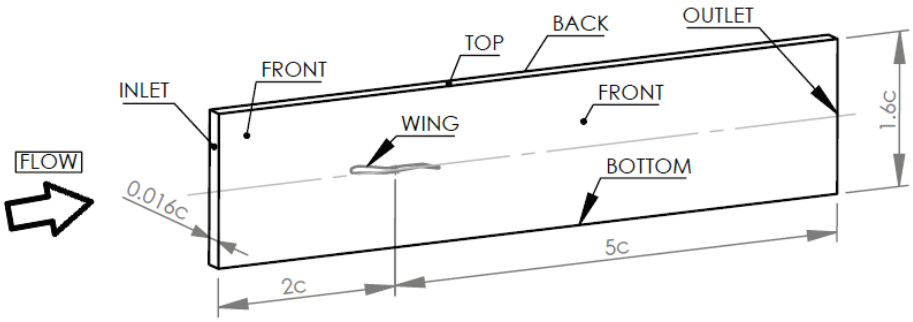


Figure 2.14: Computation domain for simulation.

- Different zones conformation to capture all the phenomenon.

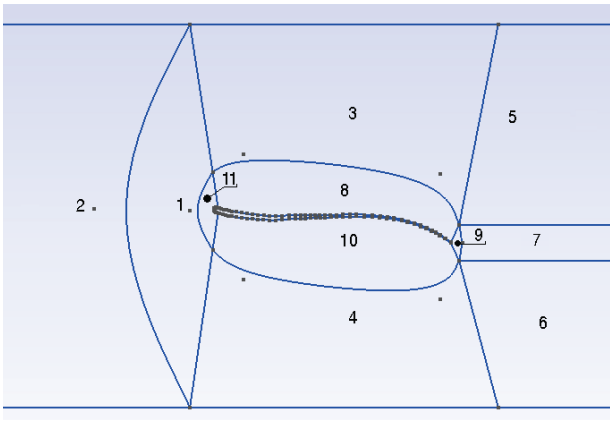


Figure 2.15: Zones conformation for the hydrofoil.

- High refinement in the zone nearest the wing.

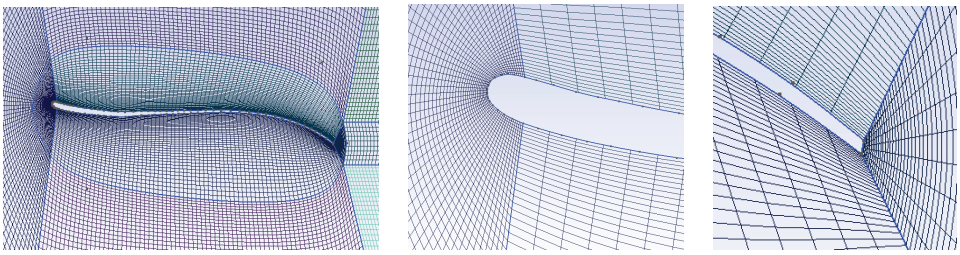


Figure 2.16: High refinement zones near the wing.

The three graphics shows the special refinement in the blade and the special mesh configuration in the leading and trailing edge zones.

The process to obtain the mesh with the required parameters to perform the simulation is showed in the next graphics:

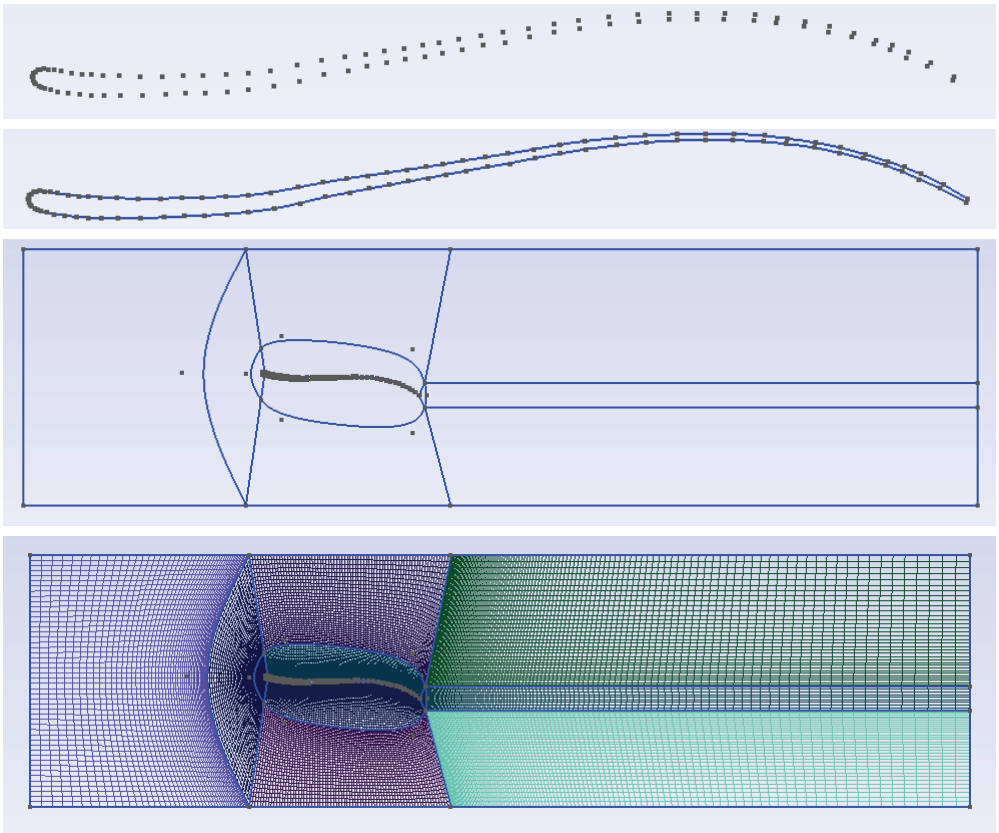


Figure 2.17: Process of obtaining the profile mesh in Gmsh (Points, lines, boundary profile, generated structured mesh.

The process initiates with the points, continue with the generation of the lines and finish with the boundaries and mesh.

Another values to take account in the optimization of the mesh are:

- Relative low number of elements, NE.
  With less number of elements the computational resources are reduced, this is because there are less elements to be analyzed. There be a correct relation between the quantity of elements and the approximate computational use to generate accurate results with the less use of computational resources.

- Low omega number, Ω.
  This value implies the relation between the number of elements, NE, divided to the number of nodes, ND, as is mentioned in the equation 2.3. The omega number indicates the quantity of elements per node; when Ω grows, the computational resources rise equally.

  The use of a structured mesh is justified with this parameter and also with the less y^+ number that is obtained with this type of mesh. The use of an unstructured mesh implies the increasing of Ω like 3 times with the same quantity of elements (Hidalgo, Luo, & Yu, 2014, p.4-6).

- Results with low residuals.
  As is mentioned in the appendix A subsection A.6, it is totally important to have residuals near zero. When the values of the residuals are smaller, this is a good sign of convergence and an initial point to consider accurate results.

- Courant number, C, in agree to the LES method conditions.
  In this case, the LES method used is implicit to perform the simulation. The Courant number, C, has not to be smaller than 1, as is mentioned in the sub-subsection 2.2.3.

  Exist a relation between the time step, Δt, the length interval, Δx, and the velocity u, defined by the equation 2.4. This determines the values that must be the last mentioned parameters to reach good results.

- Yplus number, y^+, smaller than 10 to LES method (Hidalgo, Luo, & Yu, 2014, p.5).
  This value, as is mentioned in sub-subsection 2.2.3, gives a relation between the friction velocity u_* and the local kinematic viscosity ν, by the equation 2.1.

- Mesh distortion value, bigger than 0.9 for the present thesis, that describe the quality of the mesh with a hexahedral mesh distortion analysis. The acceptable values are in the range of $[0.5, 1.0]$ (Sandia National Laboratories, 2007, p.84) (Hidalgo, Luo, Escaler, Ji, & Valencia, n.d.). The assumed value for the present thesis its taken to guarantee accurate results.

  The parameters of Courant number, C, Yplus number, y^+, and the residuals are automatically calculated with OpenFOAM.

2.4 GMSH FILES GENERATION

To manage the mesh of the hydrofoils in Gmsh it is necessary to define three different files as is noted in the appendix C: “puntos.geo” that contain the description of the points that conforms the blade, “surface.geo” that includes the generation of the lines for the surfaces of the hydrofoil and “mesh.geo” that includes all the description of the mesh to generate.

The next graphic describes the process that makes the Python-Spyder file denominated “conv.py” to generate the three necessary files for Gmsh. The complete code of the file is included in the appendix B.

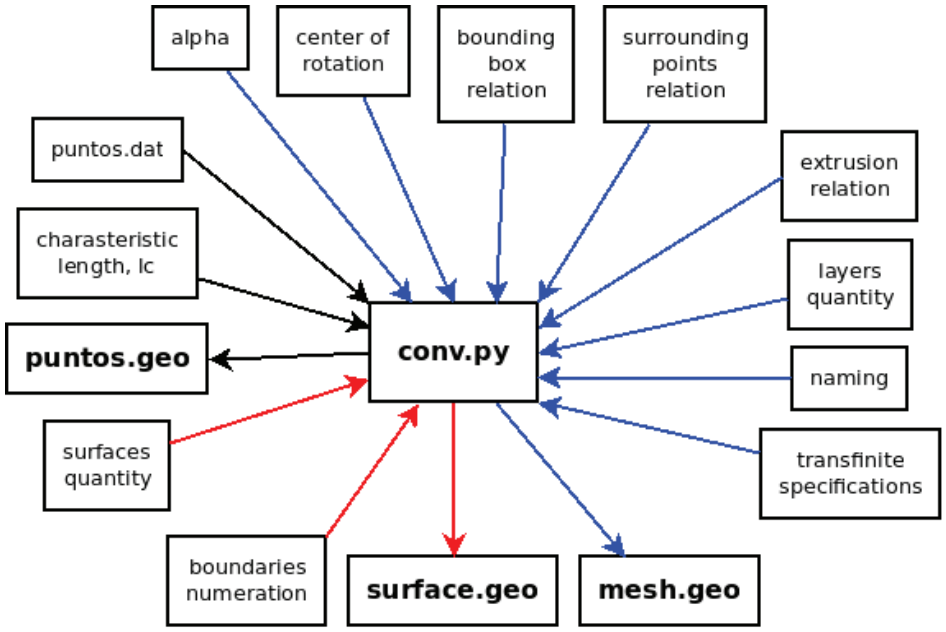


Figure 2.18: Methodology used to generate the necessary files for Gmsh using a programmed file in Python-Spyder called conv.py.

The color of the lines indicate the correspondence of the inputs to the outputs, e.g. “puntos.dat” and “characteristic length, lc” are the inputs that the file “conv.py” use to generate the output “puntos.geo”.

2.5 DESCRIPTION OF THE NUMERICAL METHOD

To begin the computational analysis of a cavitating flow is important to know the procedure of Computational Fluids Dynamics CFD and the turbulence models used to simulate it. In the appendix A: CFD, is present all the theory that involve the Computational Fluid Dynamics behavior and the most common turbulence models.

Large Eddy simulation method (LES) is a turbulence model developed to simulate turbulent flows. The method explicitly solve for the large eddies in a calculation and implicitly account for the small eddies by using a subgrid-scale model (SGS model) (CFD-Online, 2015d).

The unsteady cavitating flows commonly shows high Reynolds number that RANS method can not solve accurately. Large Eddy Simulation (LES) is a better way to solve this problem and show better results with good precision and accuracy to predict large scale turbulent eddies (Hidalgo et al., N/A, p.2).

The idea underlying LES is so called convergent evolution. Behavior of the large-scale eddies depends strongly on the forces acting on the flow and on initial and boundary conditions; they are flow-dependent. Small-scale eddies are generally independent from what is happening on the larger scales; they are flow-independent. Hence large eddies are directly resolved while small eddies are modeled (Sodja, 2007, p.13).

To perform the simulation with LES method in CFD are used the continuity (2.6) and momentum (2.7) equations. Next it is generated a filtration to reduce them for explicit LES (ELES) and implicit LES (ILES) methods. Below are present the continuity and momentum equations without filtering.

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u_i)}{\partial x_i} = 0 \tag{2.6}$$

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}(\rho u_i u_j) = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\rho\nu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right] \tag{2.7}$$

Where u is the instant velocity, t is time, i and j are the space axes subindices (Hidalgo et al., 2015, p.29).

Equations (2.6) and (2.7) have to be filtered to be solved by numerical methods; Favre-filtering operation is applied to continuity and momentum equations (Hidalgo, Luo, & Yu, 2014, p.2). Filtered equations are observed with over bar in equations 2.8 and 2.9.

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho \bar{u_i})}{\partial x_i} = 0 \tag{2.8}$$

$$\frac{\partial}{\partial t}(\rho \bar{u_i}) + \frac{\partial}{\partial x_j}(\rho \overline{u_i u_j}) = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\rho\nu\left(\frac{\partial \bar{u_i}}{\partial \bar{x_j}} + \frac{\partial \bar{u_j}}{\partial x_i}\right)\right] \tag{2.9}$$

Filtering in the context of large eddy simulation (LES) is a mathematical operation intended to remove a range of small scales from the solution to the Navier-Stokes equations. Because the principal difficulty in simulating turbulent flows comes from the wide range of length and time scales, this operation makes turbulent flow simulation cheaper by reducing the range of scales that must be resolved. The LES filter operation is low-pass, meaning it filters out the scales associated with high frequencies (Wikipedia, 2015b).

To reduce the momentum filter equation are used the following considerations (Hidalgo et al., 2015, p.29):

1. The product of filter velocities is $\overline{u_i u_j} = \bar{u}_i \bar{u}_j + \bar{u}_i^{'} \bar{u}_j^{'}$
2. The subgrid stress tensor, which is the Reynolds stress tensor is $\tau_{ij}^{'} = \rho u_i^{'} u_j^{'} = \rho(\overline{u_i u_j} - \bar{u}_i \bar{u}_j)$
3. The filtered strain tensor rate is $\bar{S}_{ij} = \frac{1}{2}\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right)$
4. The filtered viscous stress tensor is $\bar{\tau}_{ij} = 2\rho\nu\bar{S}_{ij}$

The result of the reduction of the momentum filter equation is:

$$\frac{\partial}{\partial t}(\rho \bar{u_i}) + \frac{\partial}{\partial x_j}(\rho \overline{u_i u_j}) = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_j}(\bar{\tau}_{ij} - \tau_{ij}^{'}) \tag{2.10}$$

There are some considerations for ELES and ILES that are expressed next (Hidalgo et al., 2015, p.29):

- ELES considerations
  Smagorinsky model is selected, in which $\tau_{ij}^{'}$ is considered proportional to $\bar{S}_{ij}$ as:

$$\tau_{ij}^{'} - \frac{1}{3}\tau_{kk}\delta_{ij} = -2\rho\nu_{sgs}\bar{S}_{ij} \tag{2.11}$$

  Where ν_{sgs} is the kinematic turbulent viscosity that models subgrid turbulence. Is modelled as

$$\nu_{sgs} = (C_s\Delta)^2(2\bar{S}_{ij}\bar{S}_{ij})^{0.5} \tag{2.12}$$

  Where C_s is the Smagorinsky constant that in OpenFOAM is calculated dynamically from the flow properties using Germano procedure and Δ is defined as the cubic root of mesh cell volume.

The equation 2.13 is simplified again using the incompressibility constrain and the pressure has the trace term $\frac{1}{3}\tau_{kk}\delta_{ij}$, this is indicates below:

$$\frac{\partial \bar{u}_i}{\partial t} + \bar{u}_j \frac{\partial \bar{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left([\nu + \nu_{sgs}]\frac{\partial \bar{u}_i}{\partial x_j}\right) \tag{2.13}$$

In addition for OpenFOAM, van Driest damping, is used to calculate the filter by

$$\Delta = min(\Delta_{mesh}, (k/C_{delta})y(1 - exp(-y^+/A^+))) \tag{2.14}$$

Where Δ_{mesh} is cubic root of the cell volume, k is von Kármán constant, $C_{delta} = 0.158$, $A^+ = 26$, y is the distance to the wall, and y^+ is the dimensionless number based on the wall shear stress.

- ILES considerations
  In this method the subgrid stress tensor $\tau_{ij}^{'}$ is expressed as:

$$\tau_{ij}^{'} = \rho(\overline{\bar{u}_i\bar{u}_j} - \bar{\bar{u}}_i\bar{\bar{u}}_j + \tilde{\tau}_{ij}^{'}) \tag{2.15}$$

  Where the tensor $\tilde{\tau}_{ij}^{'}$ is considered equal to subgrid dissipation scale action.

  In this thesis is used ILES method in the solver, which was developed by PhD candidate, engineer Victor Hugo Hidalgo MSc. as is mentioned in 2.6.4.

2.6 CFD SOLVER

To determine all the parameters to perform the simulation it is necessary to define some conditions: flow and boundaries, conversion of the mesh, conditions for the solver and conditions of evaluation of the results.

To initiate the simulation OpenFOAM requires the analysis via folders that contain each one different specific files. Before the simulation there are 3 folders that contain the initial conditions, the solver conditions and the mesh and simulation properties respectively. The names are: "0", "constant" and "system".

2.6.1 FLOW & BOUNDARY CONDITIONS

This parameters of the flow and boundary conditions are defined in the next folders:

- In the folder "0" are defined the next files:

Table 2.1: Description of the files of the folder "0".

File	Description
alpha1	Initial conditions for the fluid phase
p_rgh	Initial conditions for the pressure
U	Initial conditions for the velocity

Each file contain a description with the dimensions of each parameter and the conditions for each boundary of the mesh. This is included in the appendix E.

- In the folder "constant" are defined the next files:

Table 2.2: Description of the files of the folder "constant" for flow, boundary and environmental conditions.

File	Description
g	Conditions for gravity
transportProperties	Flow properties

The file "g" is specified with the dimension that correspond and a value of zero in all 3 directions in the form that is described in the appendix E. The description of the transport properties is included in the same appendix.

To perform the simulation it is necessary to determine the conditions of the flow. In Francis-99 project, using the provided information from (NTNU & LTU, 2015a) and (Stoessen, 2014) about the behavior of the flow in the turbine and the equations for turbomachinery and its similitude from (Fernandez, n.d.), are presented next the work conditions:

Table 2.3: Operation conditions for Francis 99 model and prototype turbine in best efficiency point (BEP).

Parameter	**Symbol**	**Model**	**Prototype**
Runner diameter $[m]$	D	0.63	3.215
Net head $[m]$	H_n	11.91	387.2
Flow rate $[m^3/s]$	Q	0.203	30.1
Runner angular speed $[rpm]$	n	335.4	375.0
Guide vane angle [°]	α	9.84	
Blade rotation angle [°]	α_b	80.16	80.16
Angle between u_1 and c_1 [°]	$\alpha_1 = \alpha_{u_1 c_1}$	9.97	9.97
Chord length $[m]$	c	0.24	1.22
Efficiency [%]	η	92.6	92.6
Power	N	22.0$[kW]$	105.9$[MW]$
Specific velocity	n_s	82.8	
Runner peripheral velocity $[m/s]$	u_1	11.1	63.1
Runner absolute velocity $[m/s]$	c_1	9.9	56.6
Guide vane inlet velocity $[m/s]$	c_0	8.3	47.2
Guide vane outlet velocity $[m/s]$	$c_1^{'} = U_\infty$	10.1	57.4
Discharge coefficient	ϕ_d	0.649	0.649
Runner outlet pressure $[kPa]$	p_2	102.53	
Vapor pressure at 22.0[°C] $[kPa]$	p_v	2.671	
Density $[kg/m^3]$	ρ	999.19	
Gravity in Tokke Power Station $[m/s^2]$	g	9.818	
Kinematic viscosity $[m^2/s]$	ν	$9.57x10^{-7}$	
Reynolds number	Re	2.52×10^6	73.4×10^6
Cavitation number	σ	1.974	0.061

In the analyzed conditions, Francis-99 turbine has a discharge coefficient ϕ_d of 0.649 and a specific velocity n_s of 82.8. With this n_s value by the mentioned in (Fernandez, n.d., p.59), the value of ϕ must be approximately 0.641, which indicates a variation of the ideal construction conditions. Which change would generate an reduction of c_1 from 9.9 to 9.8$[m/s]$ in the model and from 56.6 to 55.9$[m/s]$ in the prototype. The angle between u_1 and c_1 would go from 10.0 to 10.1[°] In this case, the real analyzed conditions, with the obtained values from (NTNU & LTU, 2015a) and the theoretical, with the obtained values from (Fernandez, n.d., p.59), have practically the same results. The description of the mathematical resolution is included in the appendix G.

The conditions used for the simulation are presented next:

Table 2.4: Simulation conditions for Francis 99 turbine.

Parameter	Symbol	Value
Chord length $[m]$	c^*	0.05
Blade rotation angle [°]	α_b	80.2
Runner outlet pressure $[Pa]$	p_2	102500.0
Guide vane outlet velocity $[m/s]$	U_∞	57.4
Vapor pressure at 22.0[°C] $[Pa]$	p_v	2670.0
Density $[kg/m^3]$	ρ	999.19
Kinematic viscosity $[m^2/s]$	ν	$9.57x10^{-7}$
Cavitation number	σ	0.061

In this case, it is used the chord length value of $c^* = 0.05[m]$ because it is necessary to reduce the computational resources required. In an ideal analysis must be correct to use the real value of the chord length $c_{prototype} = 1.22[m]$ but this will increase the mesh size 24 times. The use of a different mesh size will change the behavior of the cavitation in time as is analyzed in section 2.6.5, but with the analysis realized in the same mentioned section is able to use this chord length approximation.

2.6.2 MESH CONVERSION

The converted files are located in the subfolder "polyMesh" in the folder "constant". This contain the next file:

Table 2.5: Description of the file of the folder "polyMesh" that would be modified.

File	Description
boundary	Definition of the conditions for the boundaries

Also are present some automatically generated files with "gmshToFoam": "cellZones", "faces", "facesZones", "neighbour", "owner", "points", "pointZones". Is included the folder "sets" which contain the file "internal" that describe the composition of the internal zone of the mesh; this file is also generated automatically. More information about the file "boundary" is present in the appendix E.

2.6.3 PHYSICAL MODEL

The physical model is described in 4 different parts: Dimensionless numbers, mathematics considerations, flow considerations and cavitation model used in this thesis.

- Dimensionless numbers
  To initiate the analysis of unsteady cavitation on hydrofoils is necessary to evaluate two dimensionless numbers: Reynolds, Re, and cavitation number, σ. Reynolds number is indicated below and cavitation number is rewrite from the equation 1.2 to better understand:

$$Re = \frac{U_\infty c}{\nu} \tag{2.16}$$

$$\sigma = \frac{p_r - p_v}{\frac{1}{2}U_\infty^2 \rho} \tag{2.17}$$

  Where U_∞ is the stream velocity, c is the chord length, ν is the kinematic velocity, p_r is the reference pressure, p_v is the vapor pressure and ρ is the density of the fluid (Hidalgo et al., 2015, p.29).

- Mathematics considerations
  This concern at all equations and considerations for the ILES method that is used in this thesis in the solver, they are presented in the subsection 2.5

- Flow considerations
  Unsteady cavitating flows are caused by the pressure changes, so that, they are considered as multiphase flows with a two phase homogeneous mixture (Hidalgo et al., 2015, p.30), this is mentioned below:

$$\alpha = \frac{\forall_V}{\forall} \tag{2.18}$$

$$\rho = (1-\alpha)\rho_L + \alpha\rho_V \tag{2.19}$$

$$\mu = (1-\alpha)\mu_L + \alpha_V \tag{2.20}$$

  Where α is the vapor fraction, $\forall$ is the volume, ρ is the density, μ is the dynamic viscosity and L and V are the subindices for liquid and vapor respectively.

When the equation 2.18 is implemented in the continuity filtered equation 2.8 based on multiphase flow considerations, is obtained the equation 2.21:

$$\frac{\partial(\alpha\rho_V)}{\partial t} + \frac{\partial(\alpha\rho_V \bar{u}_i)}{\partial x_i} = \dot{m} \tag{2.21}$$

Where $\dot{m}$ is the interface rate mass transfer per volume. Due to the mass transfer between phases, the velocity divergence has a no-homogeneous expression (Hidalgo et al., 2015, p.30), as indicated next:

$$\frac{\partial \bar{u}_i}{\partial x_i} = \dot{m}\left(\frac{1}{\rho_V} - \frac{1}{\rho_L}\right) \tag{2.22}$$

- Cavitation model used
  A model that describe the behavior for cavitation is required to perform the simulations. In the appendix E is presented three models of cavitation commonly used. For the present thesis is used the model "Zwart", which is based on Rayleigh Plesset's equation with second derivative neglected and non-symmetrical condensation and evaporation (Hidalgo et al., 2015, p.30). The model is presented below:

$$\frac{dR}{dt} = \sqrt{\frac{2}{3}\left(\frac{|p - p_V|}{\rho_L}\right)} \tag{2.23}$$

$$\dot{m} = \begin{cases} \dot{m}^+ = F_V \dfrac{3 r_{nuc}(1-\alpha)\rho_V}{R_B}\sqrt{\dfrac{2}{3}\left(\dfrac{p_V - p}{\rho_L}\right)} & \text{if } p < p_V \\ \dot{m}^- = -F_C \dfrac{3\alpha\rho_V}{R_B}\sqrt{\dfrac{2}{3}\left(\dfrac{p - p_V}{\rho_L}\right)} & \text{if } p > p_V \end{cases} \tag{2.24}$$

  Where $F_V = 300$ and $F_C = 0.03$ are the selected calibration constants for vaporization and condensation, $r_{nuc} = 5.0 \times 10^{-6}$ is the nucleation site volume fraction and $R_B = 1.9 \times 10^{-6} m$ is the typical bubble size in water (Hidalgo et al., 2015, p.30).

All presented equations in this subsection are incorporated in the solver as is mentioned in section 2.6.4, for that there are not mathematics resolutions to do. This thesis is focused in the analysis of behavior with the performed solver.

2.6.4 SOLVER

The present OpenFOAM solver was obtained from PhD. candidate of Tsinghua University, engineer Victor Hugo Hidalgo, MSc. who makes all the modifications of system values, transport and ILES properties to perform the simulation. This solver was used to perform the simulation in many publications: (Hidalgo, Luo, Escaler, Ji, & Aguinaga, 2014), (Hidalgo, Luo, & Yu, 2014), (Hidalgo, Luo, Ji, & Aguinaga, 2014), (Hidalgo et al., N/A) and (Hidalgo et al., 2015) in which the principal variation is due to the cavitation model used but the solver structure is the same.

More detailed information of the system files and the transport and ILES properties are included in the appendix E.

To execute the simulation it is necessary to run in the terminal the next lines:

```
1 gmshToFoam mesh.msh
2 pyFoamPlotRunner.py vInterPhaseChangeFoam
3 paraview
```

The first line is to convert the mesh with a file denominated "mesh.msh" that is obtained from the software Gmsh. The second line runs the solver and generate the graphic visualization of the residuals. The third line opens ParaView to view the results. The three commands have to be ran in the "terminal" in the specific folder which are located the three initial folders of the simulation.

Another useful OpenFOAM commands are:

```
1 Co
2 yPlusLES
```

Which evaluate the Courant and Y-plus numbers respectively in all the created folders of the simulation. These two commands are executed in the terminal after the run of the solver. With a python-script is realized the analysis of the mean and maximum values of the Courant and Y-plus numbers in the simulations results.

2.6.5 VERIFICATION OF CAVITATION SIMILITUDE BEHAVIOR VARYING PARAMETERS

Due to the necessity of determine the behavior of the process of cavitation in the blade, it is necessary to verify if there is a concordance between a change of the properties with the obtained results. In this case it is analyzed the variation of the chord length of the blade, c, cavitation number, σ and the rotation angle of the blade, α_b.

The variations of the chord length c will be 0.05, 0.10 and 0.20$[m]$. The sigma number σ will be 0.05, 0.25 and 0.50. The rotation angle of the blade α_b will be 0[°], 10[°] and 80[°].

The constant values used for the analysis are presented in the next table:

Table 2.6: Conditions for constant values used in the similitude analysis.

Parameter	**Symbol**	**Value**
Outlet pressure $[Pa]$	p_2	102500.0
Vapor pressure at 22.0[°C] $[Pa]$	p_v	2670.0
Density $[kg/m^3]$	ρ	999.19
Kinematic viscosity $[m^2/s]$	ν	$9.57x10^{-7}$
Simulation end time $[s]$	$--$	0.002

Three evaluation times are used in base of the total time of the simulation: at 10[%], $t = 0.0002[s]$, 50[%], $t = 0.001[s]$ and 100[%], $t = 0.002[s]$

For define the cavitation number σ for each variation of the blade length it is necessary to calculate the velocity, this is presented next:

Table 2.7: Velocities used in the similitude analysis

Cavitation number, σ	0.05	0.25	0.5
Absolute velocity, $U\infty$ $[m/s]$	63.2	28.3	20.0

The results of the analysis are showed in the next two tables, in which are presented the omega number, Ω, the time step, $\triangle t$, maximum and mean calculated Y-plus number, y^+, and maximum and mean calculated Courant number, C.

Table 2.8: Results of similitude analysis part I.

Similitude Analysis		$c = 0.05[m]$					$c = 0.1[m]$					$c = 0.2[m]$				
		Ω	NE	$\triangle t$	y^+_{max}	C_{max}	Ω	NE	$\triangle t$	y^+_{max}	C_{max}	Ω	NE	$\triangle t$	y^+_{max}	C_{max}
0 [°]	$\sigma = 0.05$	2.35	2.1	1.0	10.1	24.1	2.35	2.4	1.0	8.6	17.8	2.35	2.7	1.0	7.4	13.3
	$\sigma = 0.25$	2.35	2.1	1.0	6.7	10.7	2.35	2.4	1.0	5.7	7.5	2.35	2.7	1.0	5.0	7.9
	$\sigma = 0.5$	2.35	2.1	1.0	5.7	4.4	2.35	2.4	1.0	4.8	5.1	2.35	2.7	1.0	4.2	3.7
10.0 [°]	$\sigma = 0.05$	2.35	2.1	1.0	10.7	24.3	2.35	2.4	1.0	9.5	25.7	2.35	2.8	0.1	12.1	4.8
	$\sigma = 0.25$	2.35	2.1	1.0	7.2	7.8	2.35	2.4	1.0	6.1	10.2	2.35	2.8	1.0	6.5	4.5
	$\sigma = 0.5$	2.35	2.1	1.0	6.0	4.0	2.35	2.4	1.0	5.2	4.9	2.35	2.8	0.05	7.2	1.5
80.0 [°]	$\sigma = 0.05$	2.35	4.7	0.1	9.2	6.5	2.35	5.0	0.1	9.4	4.0	2.35	2.5	0.1	43.1	20.3
	$\sigma = 0.25$	2.35	4.7	0.5	8.3	12.2	2.35	5.0	0.1	7.4	3.5	2.35	2.5	0.1	39.1	12.7
	$\sigma = 0.5$	2.35	4.7	0.5	5.5	11.6	2.35	5.0	0.1	6.5	4.3	2.35	2.5	$A1^*$	25.4	5.1

NE: This value is divided to 10^5.
$\triangle t$: This value is multiplied to 10^6.
$A1^*$: Automatic adjustment of the time step, maximum delta $= 1 * 10^{-3}$.

The modification realized in "controlDict" to generate an automatic adjustment of the time step, $\triangle t$, is presented next:

```
/*
OpenFOAM programation for automatic adjustment of time step
*/
adjustTimeStep   on;
maxCo            5;
maxDeltaT        1e-3;
```

Table 2.9: Results of similitude analysis part II.

Similitude Analysis		$c = 0.05[m]$		$c = 0.1[m]$		$c = 0.2[m]$	
		y^+_{mean}	C_{mean}	y^+_{mean}	C_{mean}	y^+_{mean}	C_{mean}
0 [°]	$\sigma = 0.05$	8.0	12.9	7.6	15.7	5.4	11.6
	$\sigma = 0.25$	5.1	8.4	4.5	6.6	3.2	4.1
	$\sigma = 0.5$	4.1	2.7	3.6	4.2	2.5	2.3
10.0 [°]	$\sigma = 0.05$	8.9	10.7	7.7	19.9	5.2	0.5
	$\sigma = 0.25$	5.5	3.2	4.3	5.2	3.3	1.7
	$\sigma = 0.5$	4.6	2.1	3.5	2.5	2.6	0.1
80.0 [°]	$\sigma = 0.05$	4.5	3.3	4.4	2.2	9.0	0.8
	$\sigma = 0.25$	2.9	7.4	2.7	1.0	11.0	1.4
	$\sigma = 0.5$	2.3	5.4	2.1	0.5	14.0	5.0

The results show a constant omega number, Ω, in all of simulated meshes. The number of elements, NE, varies from 200 000 to 280 000 approximately in the meshes with 0[°] and 10[°], in the case of the mesh with 80[°] the values go from 250 000 to 500 000 approximately. The time step, $\triangle t$, necessary decrease with the increment of the chord length, c, and with lows cavitation numbers, σ.

The mean values of Y-plus, y^+, and the Courant number, C, present a variation between 2.1 to 14.0 for y plus and 0.1 to 19.9 for the Courant number. Y-plus numbers are in the acceptable range with only one value out of the acceptable range of 10. It is observed also that to reach low Courant numbers its necessary to reduce the time step; that it is in accord of the behavior of the phenomenon as it is described in equation 2.4. In the next pages are presented the graphical results of the behavior of the cavitation in the simulations.

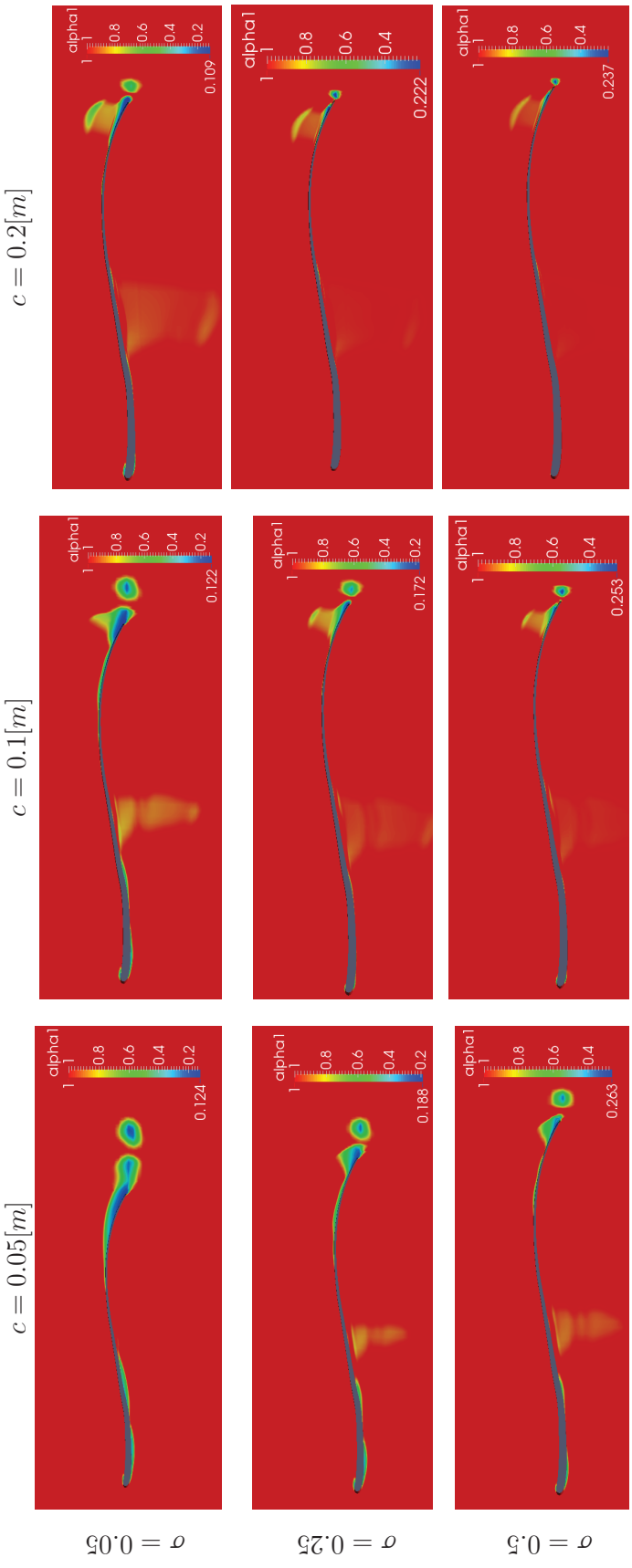


Figure 2.19: Vapor fraction (alpha1=1 is water) for similitude analysis for 0 [°] and 10% of time, $t = 0.0002[s]$

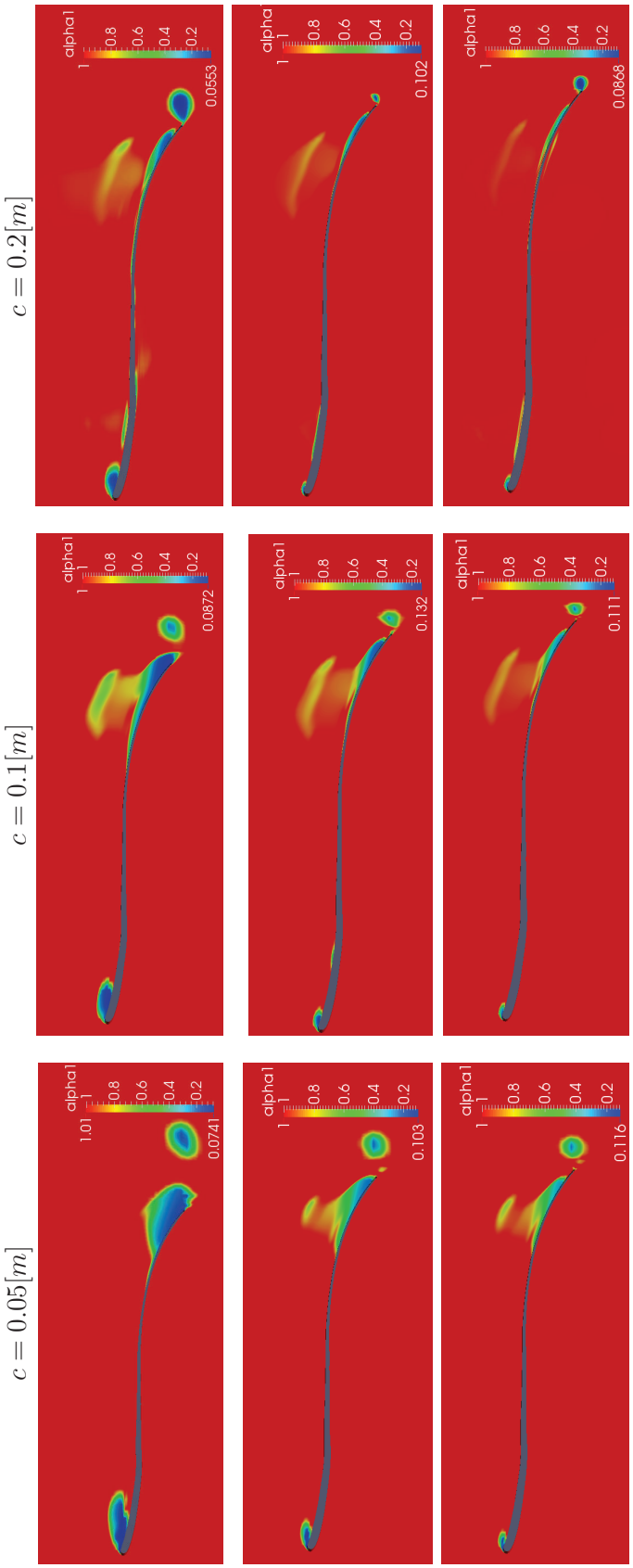


Figure 2.20: Vapor fraction (alpha1=1 is water) for similitude analysis for 10 [°] and 10% of time, $t = 0.0002[s]$

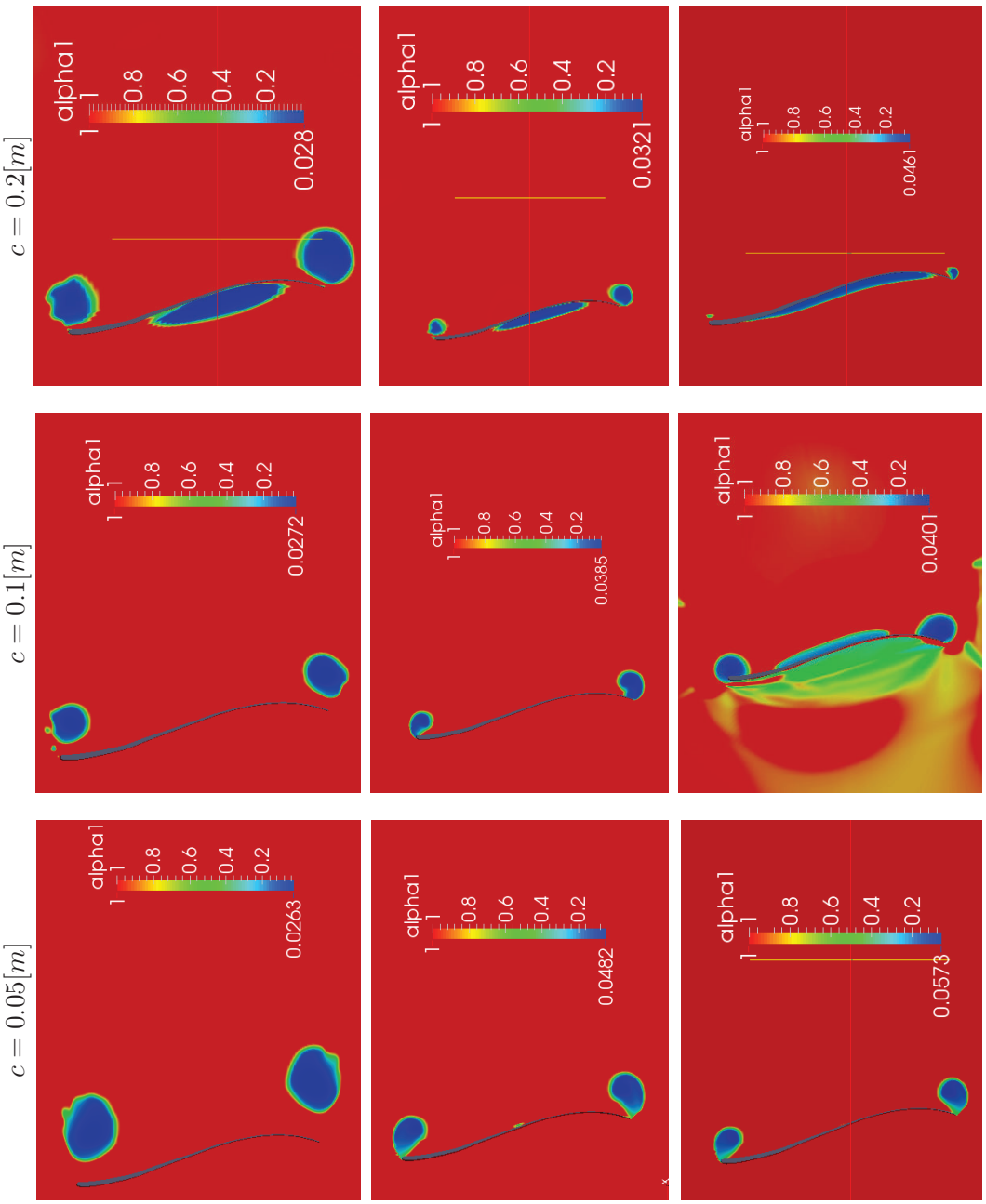


Figure 2.21: Vapor fraction (alpha1=1 is water) for similitude analysis for 80 [°] and 10% of time, $t = 0.0002[s]$

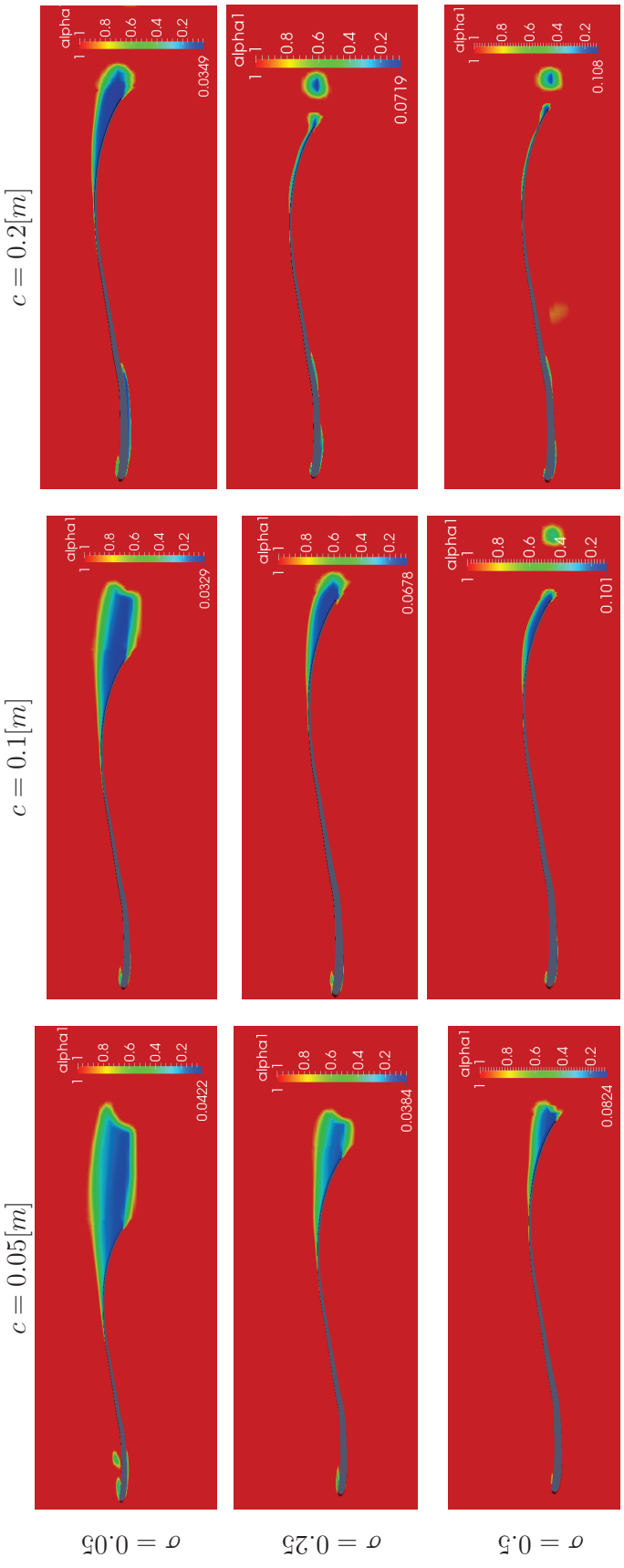


Figure 2.22: Vapor fraction (alpha1=1 is water) for similitude analysis for 0 [°] and 50% of time, $t = 0.001[s]$

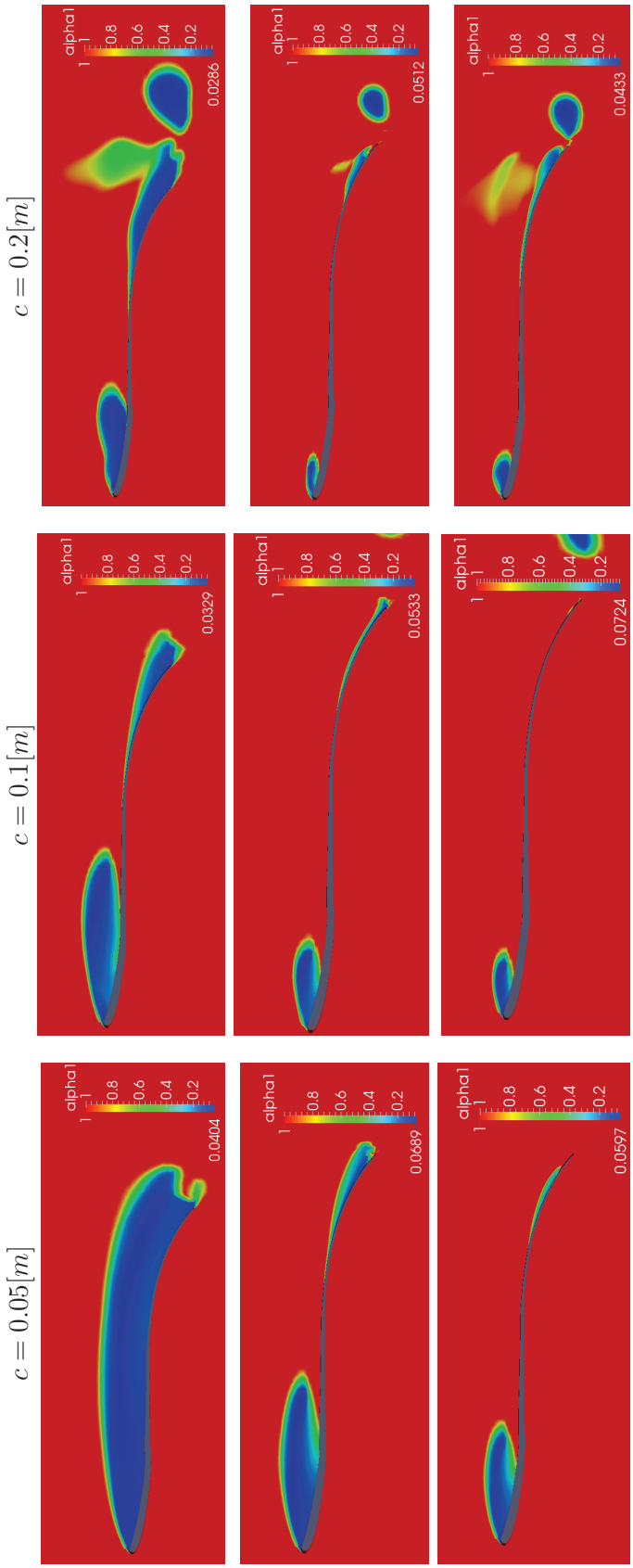


Figure 2.23: Vapor fraction (alpha1=1 is water) for similitude analysis for 10 [°] and 50% of time, $t = 0.001[s]$

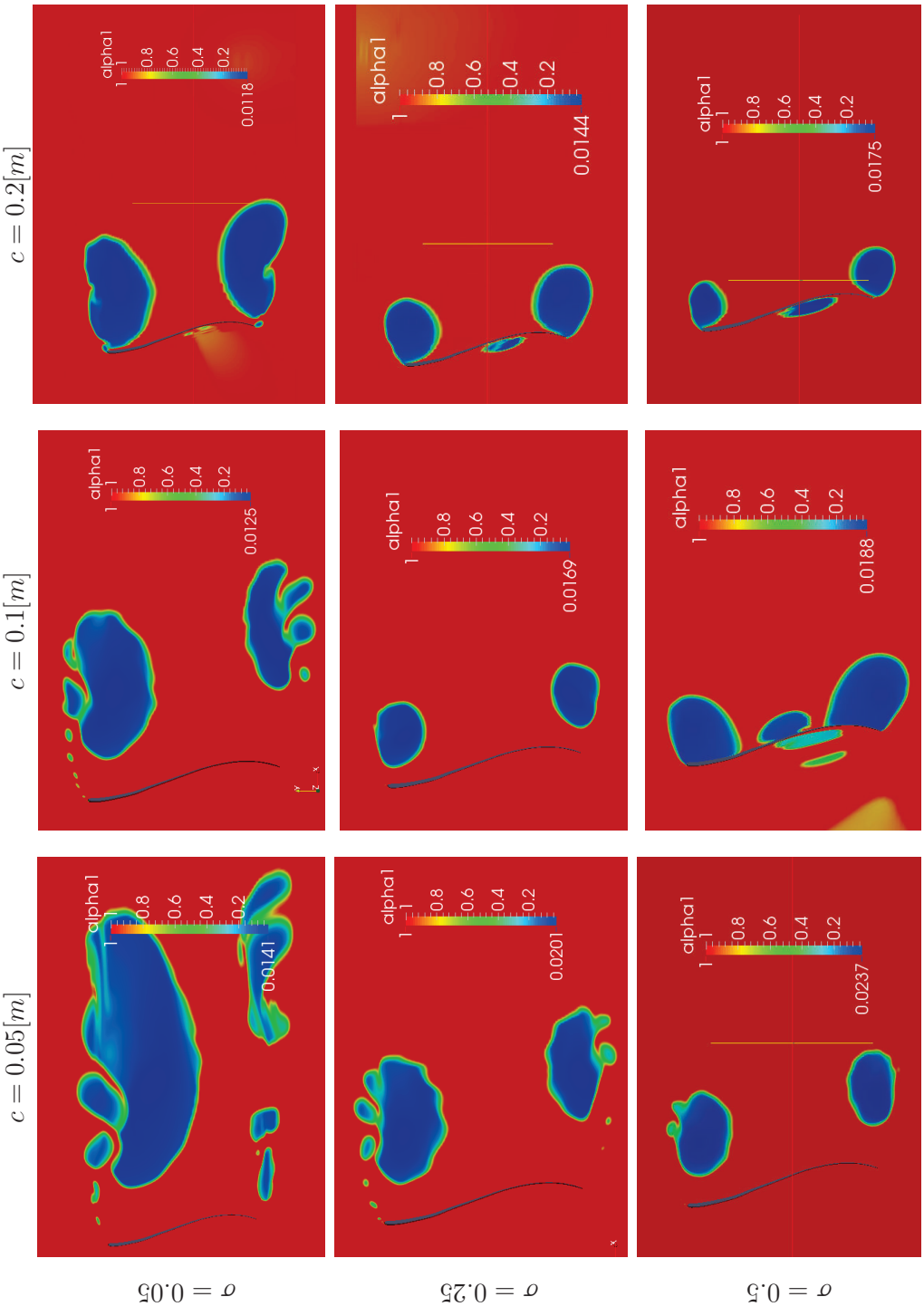


Figure 2.24: Vapor fraction (alpha1=1 is water) for similitude analysis for 80 [°] and 50% of time, $t = 0.001[s]$

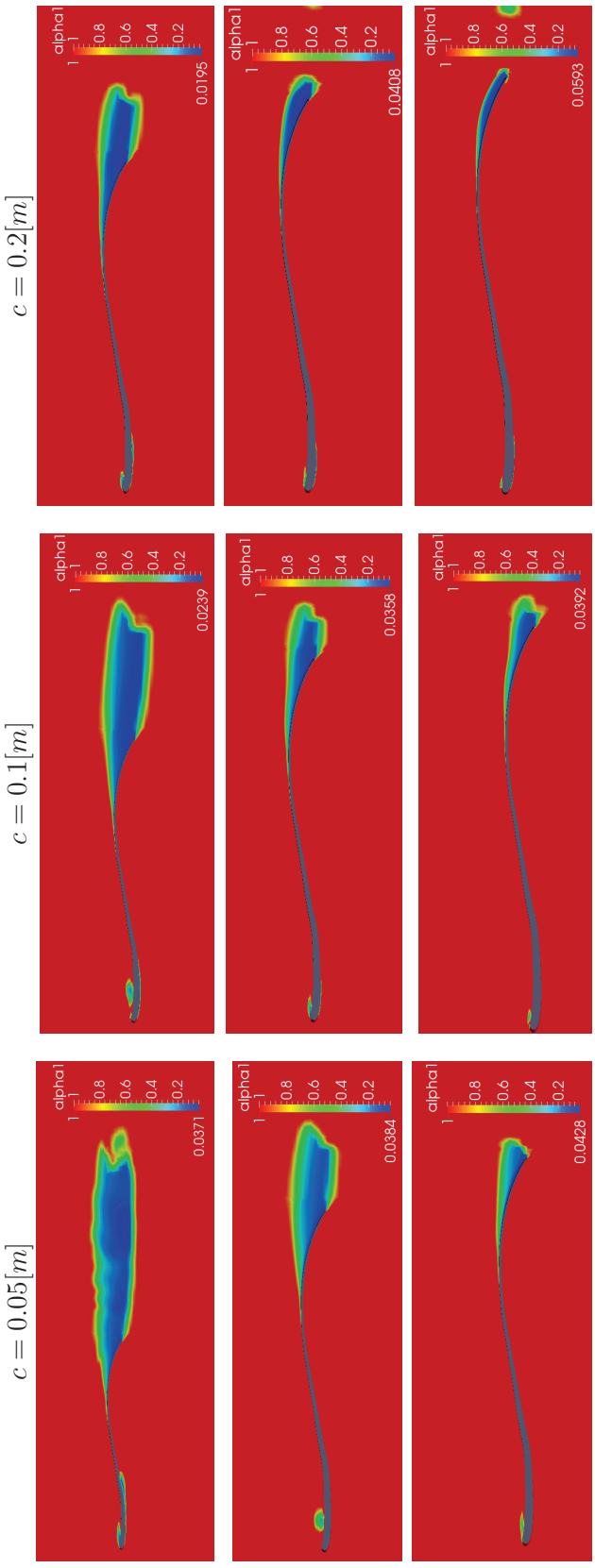


Figure 2.25: Vapor fraction (alpha1=1 is water) for similitude analysis for 0 [°] and 100% of time, $t = 0.002[s]$

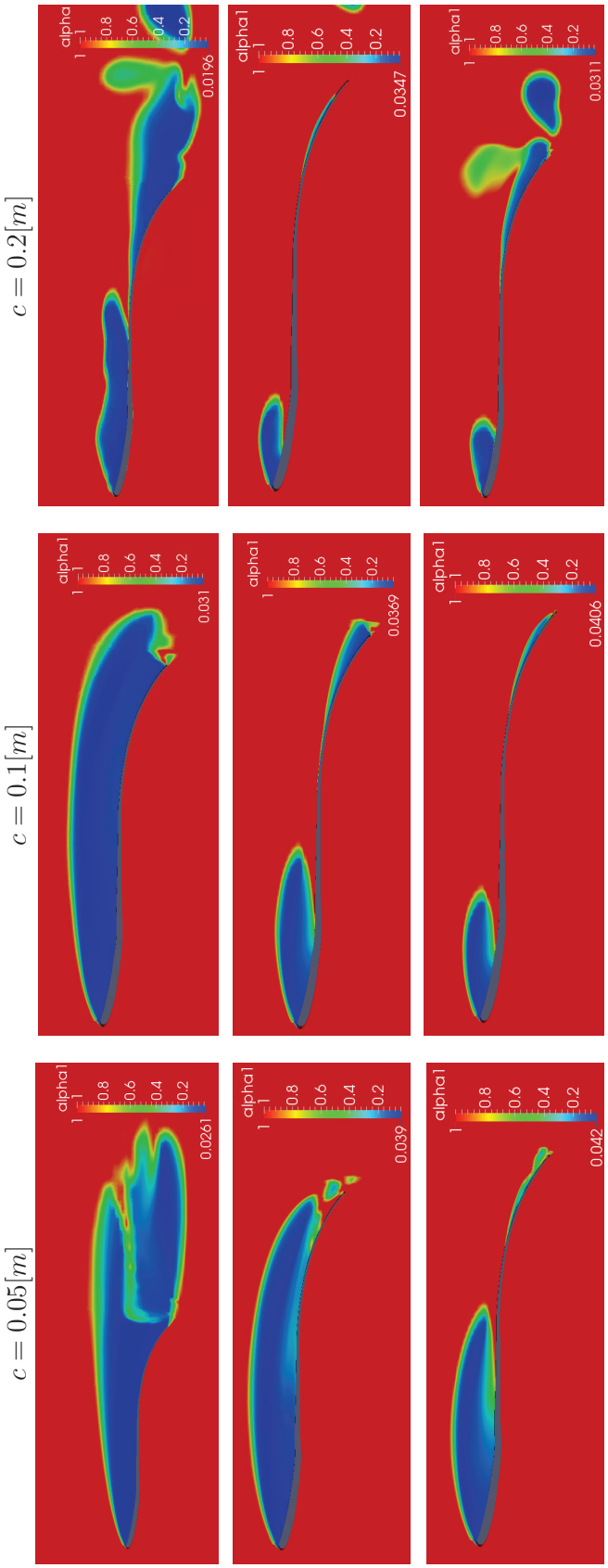


Figure 2.26: Vapor fraction (alpha1=1 is water) for similitude analysis for 10 [°] and 100% of time, $t = 0.002[s]$

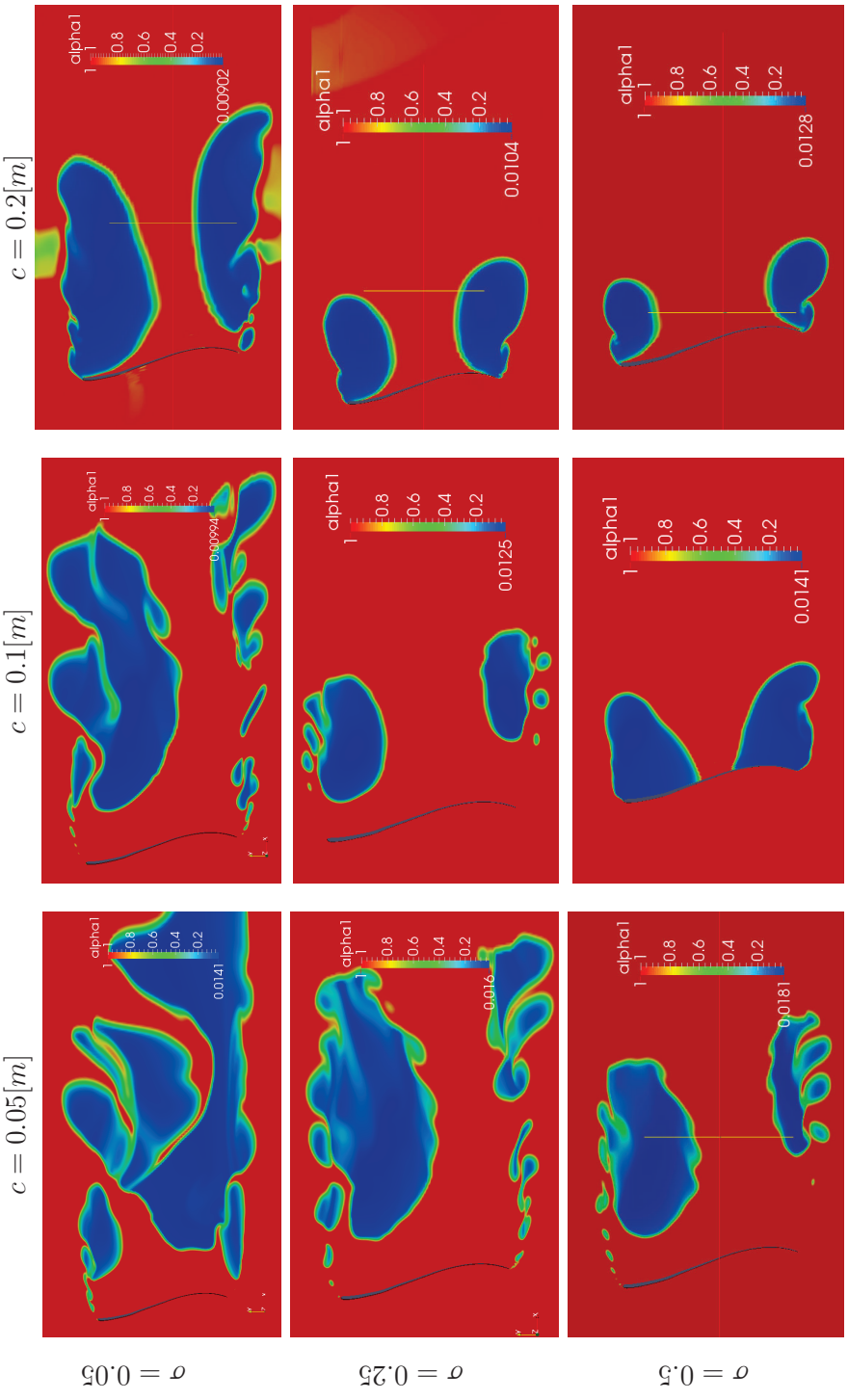


Figure 2.27: Vapor fraction (alpha1=1 is water) for similitude analysis for 80 [°] and 100% of time, $t = 0.002[s]$

As is seen in the graphics, does not exist a direct relation between the increment of the chord length of the blade in the mesh and the behavior of the cavitation generated.

In the same analyzed time, the cavitation generated is less in the bigger mesh, this correspond probably at a reduction of the effect of the cavitation while the mesh grows. For that reason, next it is analyzed the behavior of the cavitation in different chord lengths to prove that exist a relation between the cavitation generated that result with the same effect in less analyzed times in the smaller meshes that bigger ones.

For the analysis is used the blade turned 80 [°] at a sigma value of $\sigma = 0.05$. Meshes with chord lengths of $c = 0.05$, $c = 0.1$ and 0.2 [m] are taken. The values used and obtained in this analysis are presented below:

Table 2.10: Obtained values used in similitude analysis comparison

		c	Ω	NE	Δt	y^+_{max}	y^+_{mean}	C_{max}	C_{mean}
80 [°]	$\sigma = 0.05$	0.05	2.346	4.7	1.0	9.2	4.5	6.5	3.3
		0.1	2.346	5.0	1.0	9.4	4.4	4.0	2.2
		0.2	2.352	2.5	1.0	43.1	9.0	20.3	0.76

NE: This value is divided to 10^5.
$\triangle t$: This value is multiplied to 10^7.

The results shows that exist practically the same Ω number in the three meshes but with an increment of the number of elements, NE, in the first and second mesh of practically two times of the third mesh value. This difference in the number of element probably would produce a different behavior of the phenomenon in the analysis.

The Δt value is the same for the three meshes. The mean values of y_+ are in accord of the ideal value of 10. But in the bigger mesh the value of y^+_{max} is really high, this situation will be improved making the mesh more refined.

The mean Courant number, C_{mean}, of the first mesh it is practically 4 times bigger than the third, but for this analysis it is acceptable. The maximum values have the adverse behavior, the third mesh has the bigger value, this must be in relation with the refinement of the mesh.

To prove the validation of the mesh is used also the function of ParaView denominated "mesh distortion analysis" as well the values of Ω, y^+_{mean} and C_{mean} that are explicated before. The results of the distortion analysis are presented next:

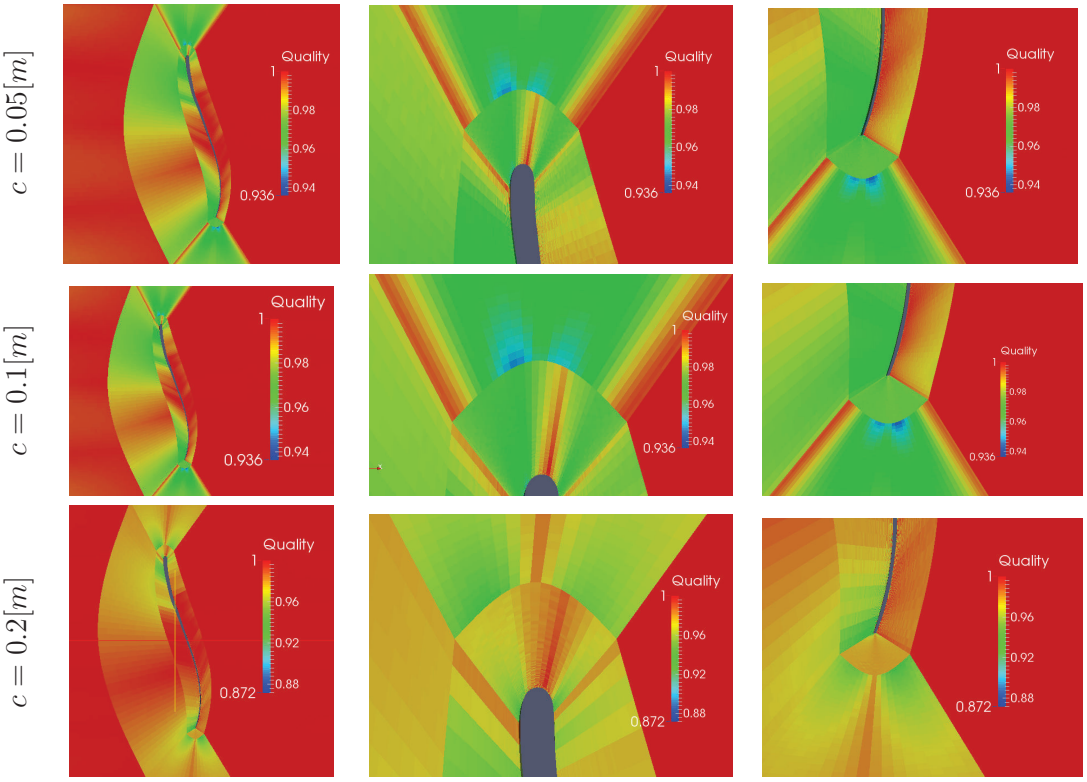


Figure 2.28: Distortion analysis in similitude comparison.

The values obtained from the distortion analysis shows that the minimum value is 0.936 for $c = 0.05$ and $c = 0.1$, and 0.872 for $c = 0.2$. These values are acceptable because optimal hexahedral quality mesh distortion values must been between near to 1.0, with an acceptable range from $[0.5, 1.0]$ (Sandia National Laboratories, 2007, p.84) (Hidalgo et al., n.d.), for this thesis the optimal minimum value for distortion is taken 0.9.

In the present case the mesh for $c = 0.2$ has a minimum value lower than 0.9, but with an error of 3% from the ideal value. The difference produced in the quality of the third mesh will cause the high numbers of y^+ and C in previous page and probably a different behavior of the phenomenon of cavitation. For a future work the mesh with less quality could be optimized.

The comparison of the behavior of the cavitation in time in the three meshes is showed in the next graphics:

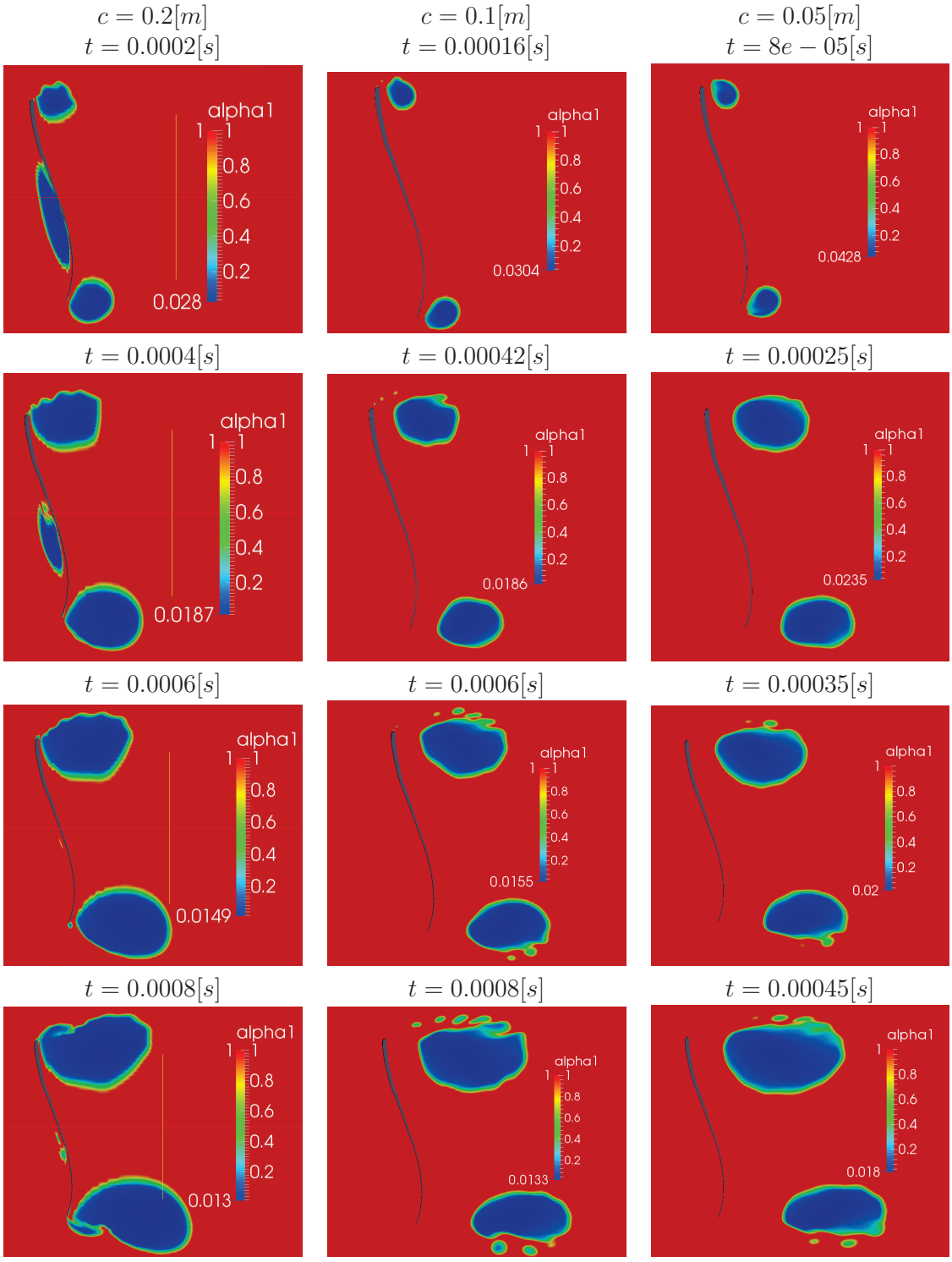


Figure 2.29: Vapor fraction (alpha=1 is water) behavior in similitude comparison in time part I.

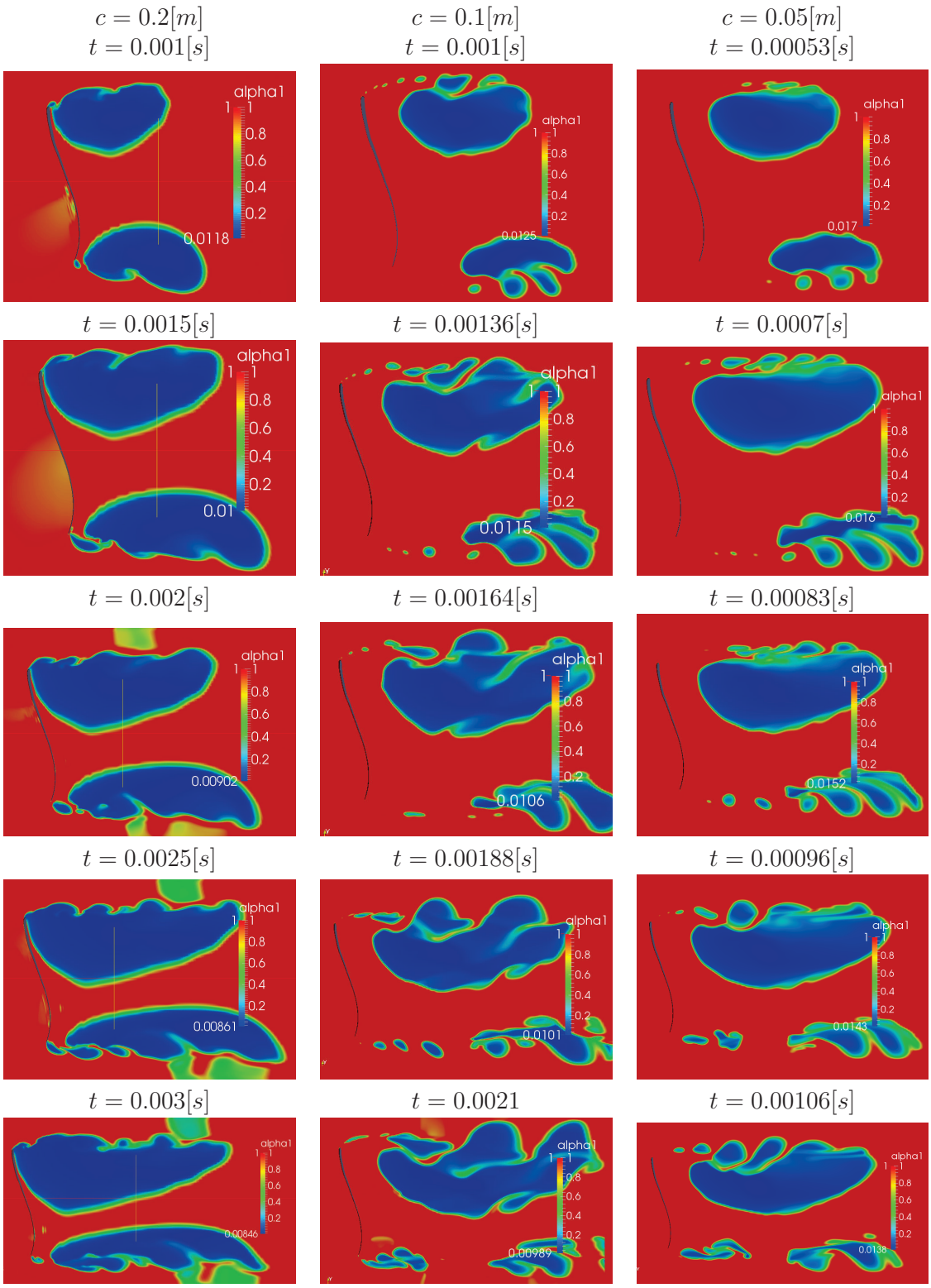


Figure 2.30: Vapor fraction (alpha=1 is water) behavior in similitude comparison in time part II.

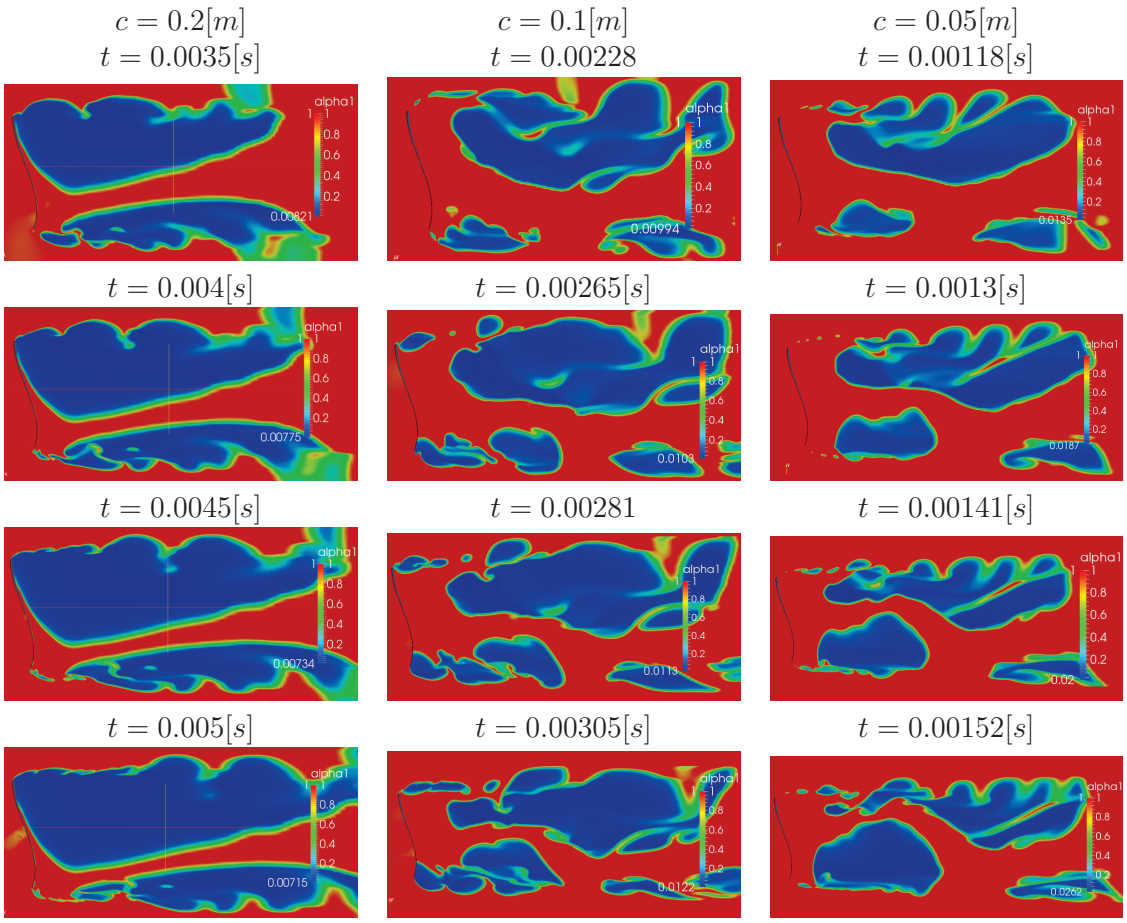


Figure 2.31: Vapor fraction (alpha=1 is water) behavior in similitude comparison in time part III.

In the initial case, the bigger mesh present a cavity in the left section of the blade and two cavities that are similar at the located in the other meshes. From pictures of the bigger mesh of $t = 0.0002[s]$ (initial) to $0.002[s]$ (seventh) the behavior of the small mesh is very similar, discarding the left bubble generated, but with a clear difference in time and in the position of the bubble respect at the blade. In the case of the central mesh, the cavitation produced is very similar at the generated in the small mesh, would be a relation between the quantity of elements and mesh quality to produce the difference in the bigger mesh. In this analyzed time the generated bubble is not attached at the wall of the blade like in the first mesh.

From $t = 0.0025[s]$ to the $0.005[s]$ in the left mesh, the bubble is attached to the wall of the blade, this does not occur in the other two meshes. Exist a similar behavior between the central and the right meshes but with a difference on the collapse produced in the central mesh at high times. The conduct of the bubble in the big mesh is obviously different because the attachment produced.

The delay present between the meshes is compared in the next table:

Table 2.11: Obtained values used in alpha similitude comparison

$t_{c=0.05[m]}[s]$	$t_{c=0.1[m]}[s]$	$t_{c=0.1} - t_{c=0.05}[s]$	$t_{c=0.2[m]}[s]$	$t_{c=0.2} - t_{c=0.05}[s]$
0.00008	0.00016	0.00008	0.0002	0.00012
0.00025	0.00042	0.00017	0.0004	0.00015
0.00035	0.0006	0.00025	0.0006	0.00025
0.00045	0.0008	0.00035	0.0008	0.00035
0.00053	0.001	0.00047	0.001	0.00047
0.0007	0.0014	0.00066	0.0015	0.0008
0.00083	0.0016	0.00081	0.002	0.00117
0.00096	0.00188	0.00092	0.0025	0.00154
0.00106	0.0021	0.00104	0.003	0.00194
0.00118	0.00228	0.0011	0.0035	0.00232
0.0013	0.00265	0.00135	0.004	0.0027
0.00141	0.00281	0.0014	0.0045	0.00309
0.00152	0.00305	0.00153	0.005	0.00348

Exist clearly a delay present between the times in the mesh of $c = 0.05[m]$ and the other two meshes, the difference between the times $t_{c=0.1} - t_{c=0.05}$ and $t_{c=0.2} - t_{c=0.05}$ indicate a growth tendency that is bigger in the mesh with $c = 0.2[m]$. It is clearer to observe this behavior watching the next graphic:

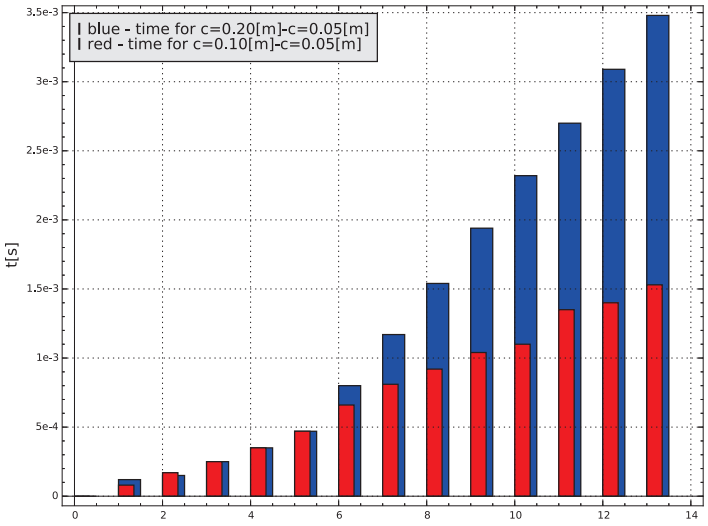


Figure 2.32: Bar plot of the alpha similitude comparison.

In blue and red are present the behavior of the difference with the meshes with $c = 0.2[m]$ and $c = 0.1[m]$ respectively. It is observed that exist a bigger delay when the mesh grows. This tendency could predict the behavior of the cavitating flow in the mesh of the blade at a real scale.

For generate a mathematical expression of the behavior of the delay was made the assumption of that is expressed in the form:

$$t = At_{c=0.05}^{B} \tag{2.25}$$

This assumption was made because the simulations begin in the point $(0, 0)$ and could have an exponential behavior. Next is present the graphical resolution:

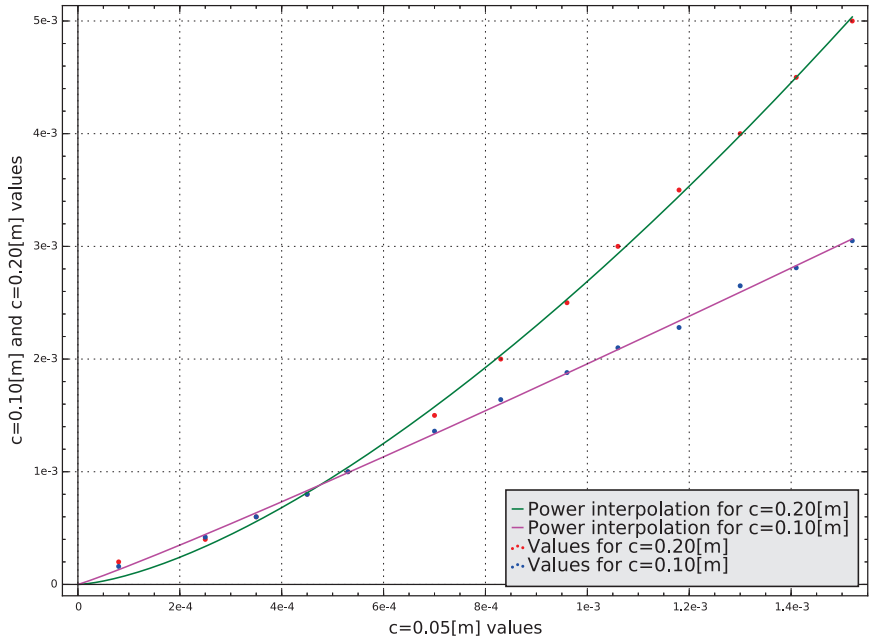


Figure 2.33: Power interpolation of alpha in similitude comparison.

In this case it is observed that the behavior of the mesh with $c = 0.1[m]$ it is practically linear and the other, with the mesh with $c = 0.2[m]$, has a curvature. In the real scale case the results would have a bigger slope. The obtained constants are expressed below:

$$For \quad c = 0.1[m] \quad \{A = 3.21, B = 1.07\} \tag{2.26}$$

$$t_{c=0.1} = 3.21t_{c=0.05}^{1.07}$$

$$For \quad c = 0.2[m] \quad \{A = 84.06, B = 1.5\} \tag{2.27}$$

$$t_{c=0.2} = 84.06t_{c=0.05}^{1.5}$$

To have better results it's necessary to perform the analysis with optimized meshes with practically the same value of distortion and number of elements. The use of a bigger mesh, with the real scale size of the blade, it is absolutely necessary but this will require high computational resources to perform the simulation.

2.6.6 BEHAVIOR VARYING BLADE SECTIONS IN FRANCIS-99

Because the different blade sections present in the Francis-99 turbine, between the upper, middle and lower part of it, was realized the analysis of the behavior to understand what happen when cavitation start. The values used to perform the simulation are presented in the table 2.4 and the obtained results of the analysis are presented next:

Table 2.12: Obtained values used in blade sections comparison

Section	Ω	NE	Δt	y^+_{max}	y^+_{mean}	C_{max}	C_{mean}
up	2.35	3.9	$A2^*$	29.1	11.7	10.6	10.0
middle	2.35	3.7	$A2^*$	21.3	8.5	12.7	10.02
down	2.35	4.0	$A2^*$	40.8	7.4	11.7	9.98

NE: This value is divided to 10^5.
$A2^*$: Automatic adjustment of the time step
maximum values: $\Delta t = 1 * 10^{-3}$, $C = 10.0$.

The results shows that exist the same Ω number in the two meshes with practically the same number of elements, NE. In the three cases is used an automatic adjustment of the Δt to reach a maximum value of the Courant number of 10.0.
The maximum values of y^+ are extremely high, but, the mean value in the three cases is near at 10; the ideal condition. Only for the upper blade case the value is bigger, in this case the error is near at 20% from the ideal value. The mean Courant number in the three cases it is around 10, this is determined by the condition of the automatic adjustment of the time step.

The results of the distortion analysis of these three meshes are presented next:

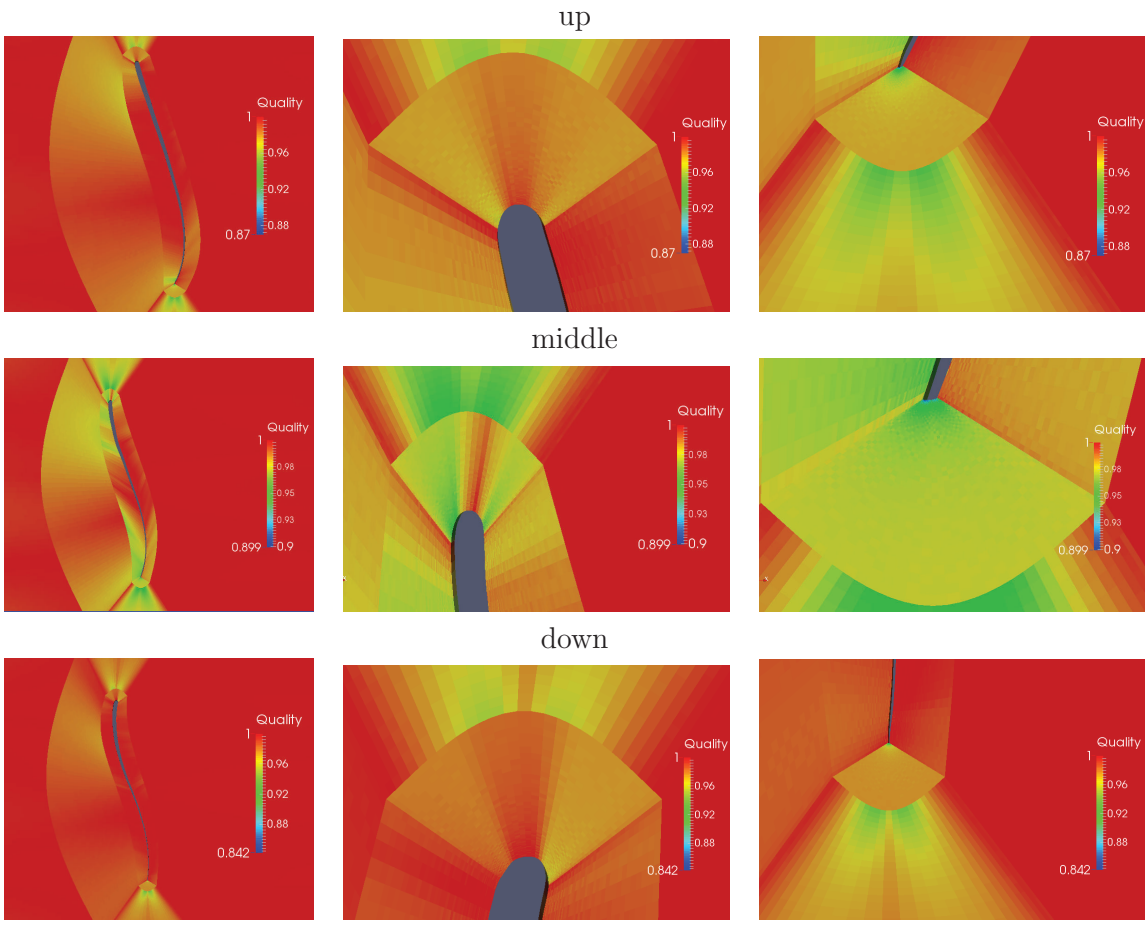
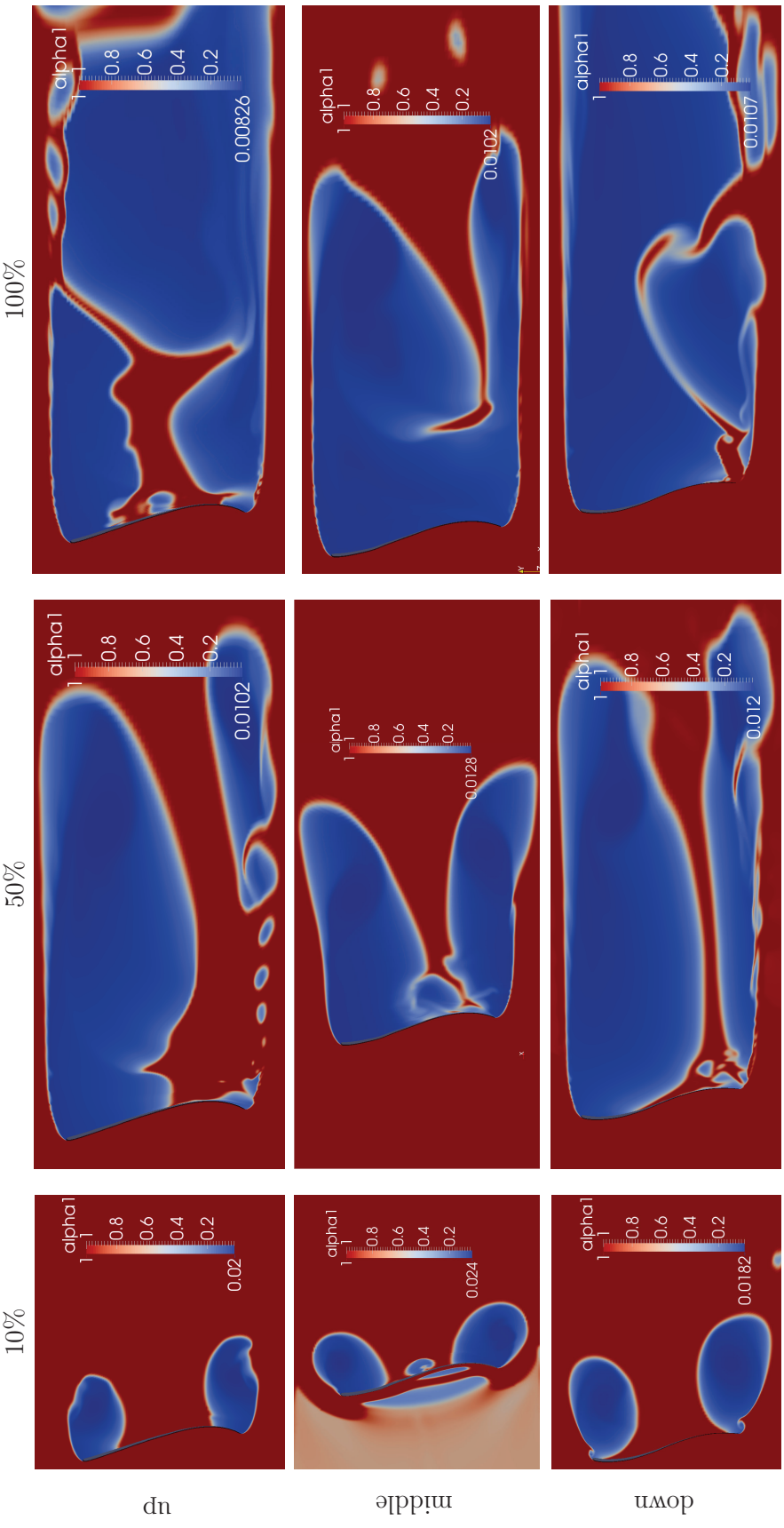


Figure 2.34: Distortion analysis for Francis-99 blade sections

The values obtained from the distortion analysis shows that the minimum value is 0.87 for the upper section, 0.899 for the middle section and 0.842 for the lower section. From the optimal value of 0.9 the lower mesh has an error of 6% approximately that is acceptable. For a future work these meshes could be optimized.

The behavior of the cavitation in time in the three blades is present in the next graphic:



Percent of the total analyzed time: $t = 0.002[s]$

Figure 2.35: Vapor fraction (alpha1=1 is water) for Francis-99 blade sections in time.

In this case it is possible to do three analysis of the blades, in each case, it is take in account the effect only in the superior section of the blade because in a real case the set of blades of the runner blocks the direct contact of the water to the inferior section.

- 10 % of the simulated time:
  In this case the biggest cavitation is formed in the down section and the effect of it decrease while reach the middle section. In the upper section the behavior is similar. Exist also a formation of cavities in the middle side of the blade that doesn't appear in the middle or upper sections.

- 50 % of the simulated time:
  The formed cavity has a considerable size, it is bigger in the upper and lower sections. The less effect of the generation of the cavity is present in the middle section of the blade but with the presence of a re-entrant jet. The upper section present a supercavity similar at the down section, exists also the presence of a re-entrant jet that would stretch the bubble.

- 100 % of the simulated time:
  At this time of the simulation the greater effect of cavitation is in the down section, in which a supercavity is completely developed. The middle section present a big bubble released in the water. The upper section of the blade present a cavity attached to the wall in the middle of the blade and a stretched supercavity in the trailing edge. This behavior correspond at a break of the bubble due to the re-entrant jet indicated in the analysis of the 50% of the time.

  In the three cases is present cavitation in an aggressive form, but in the lower section of the blade the supercavity is the biggest. This could trigger in undesirable effects in the draft tube. The different shapes that have each blade section generates a different behavior of the cavitation.

  The analyzed phenomenon present a behavior different of the reality due to the contact produced of the water, that in this case include all the surface of the blade. This produce an exaggerated cavitation..

  To reach a model that correspond of the analyzed phenomenon of cavitation in Francis turbines must be defined another type of mesh, which it is present in the next section.

2.6.7 MESH WITH GUIDE VANE AND BLADE

Due to the necessity to understand the behavior when the water goes into the turbomachinery and generate cavitation, was proposed a new mesh model in which was analyzed the guide vane and the blade in one single mesh. The proposed model is exposed below:

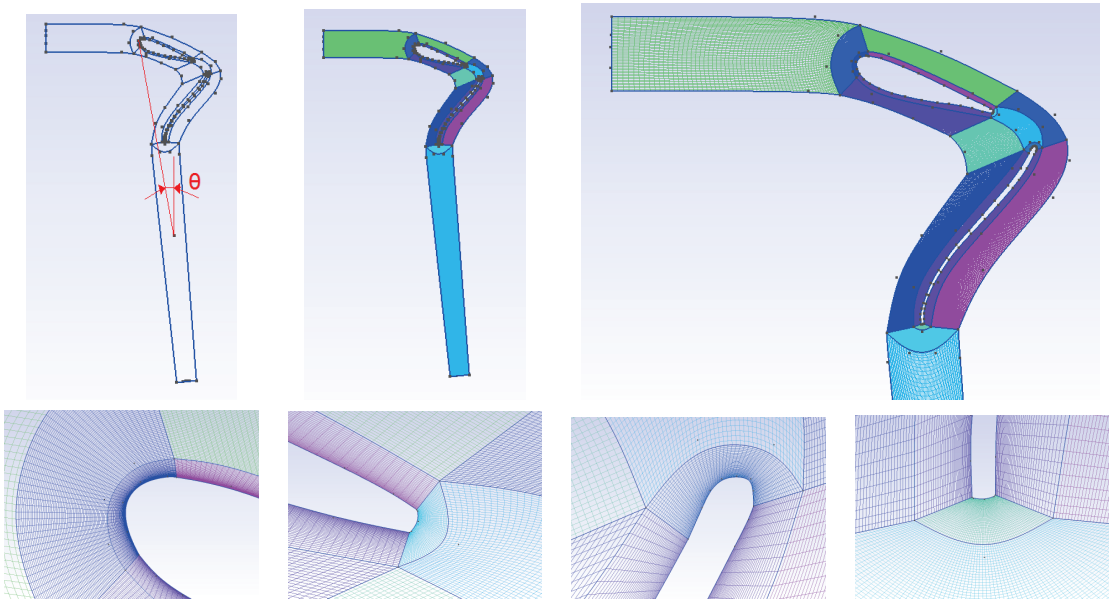


Figure 2.36: Mesh model for guide vane and blade.

This mesh uses the BEP configuration in the position of the guide vane and the blade. The inlet is located horizontal, this was realized to obtain by the calculus an inlet velocity in the x direction. The outlet has a radial direction. This arrangement of the elements was realized rotating by the central axis of the runner.

The values assumed for the simulation are the same mentioned in the table 2.4, the only change is in the inlet velocity at $U_\infty = 47.2[m/s]$ to reach the same cavitation number in the blade zone. The form to obtain the value is mentioned in the appendix G.

To validate the mesh is necessary to prove that the distortion analysis reach good values, these results are indicated below:

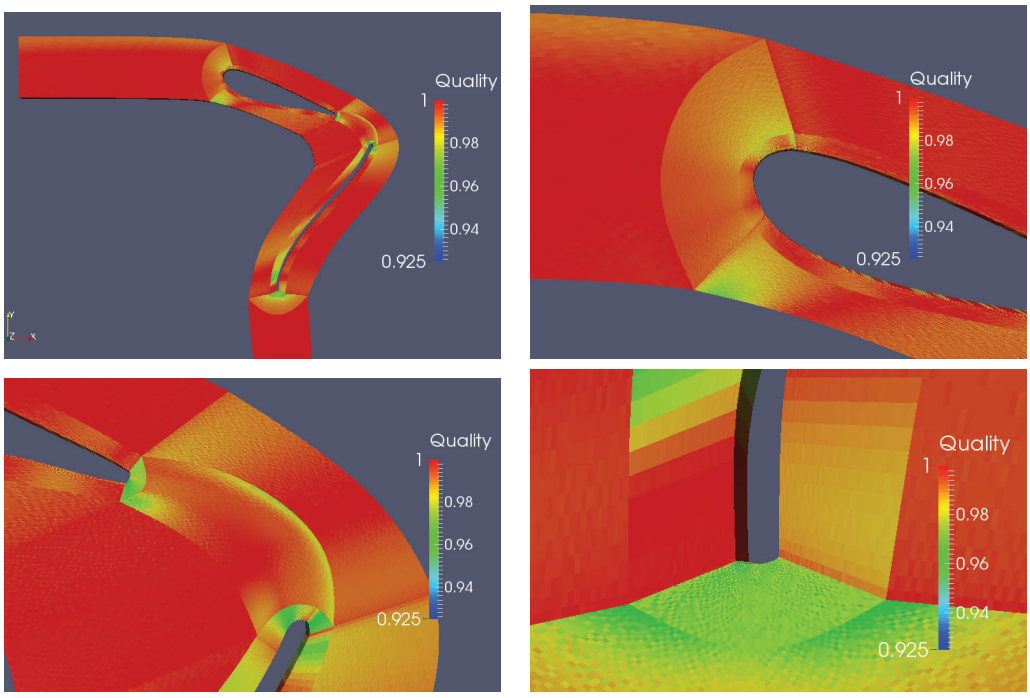


Figure 2.37: Distortion analysis for guide vane and blade mesh.

The results shows that exist a minimum value of quality of 0.925, this result is in accord to the necessity of a good mesh.

The obtained results of the simulation of the cavitation in the mesh are presented in the next table:

Table 2.13: Obtained values in the mesh with guide vane and blade.

Ω	NE	Δt	y^+_{max}	y^+_{mean}	C_{max}	C_{mean}
2.35	5.5	$A3^*$	6.63	3.66	0.926	0.9

NE: This value is divided to 10^5.
$A3^*$: Automatic adjustment of the time step
maximum values: $\Delta t = 1 * 10^{-3}$, $C = 0.9$.

The mesh with approximately 550 000 elements has values of y^+ in agreement of the ILES method, also the Courant number, C, has been optimized to reach maximum 0.9 to obtain less permanency time in the finite volumes and therefore better accuracy.

This simulation was made with the help of PhD candidate, engineer Victor Hugo Hidalgo MSc., in State Key Laboratory of Hydro science & Engineering, Tsinghua University, Beijing, China, due to the necessity of high computational resources and a small simulation time to reach the results of this thesis.

The behavior of the vapor fraction in the mesh is presented in the next graphic:

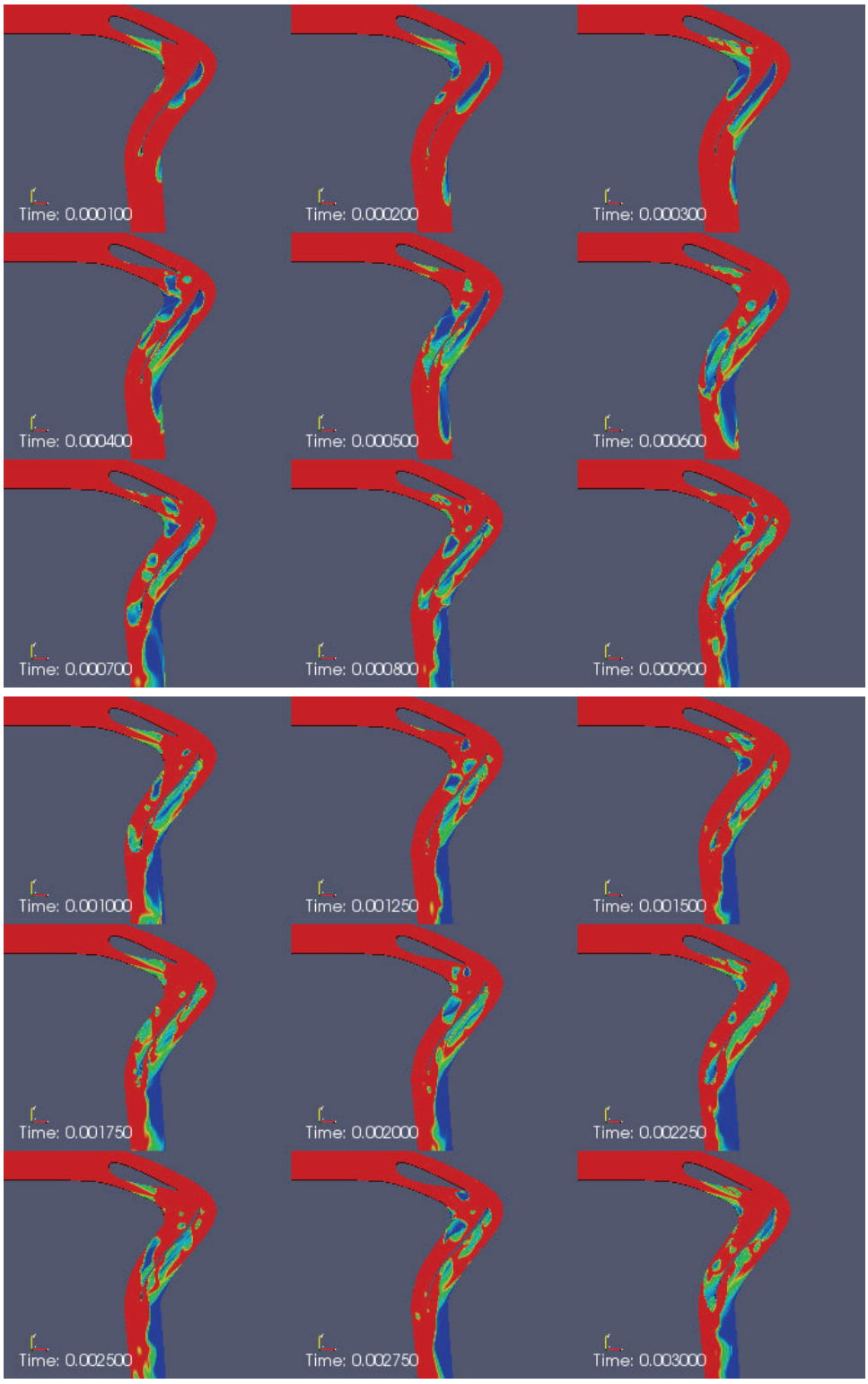


Figure 2.38: Vapor fraction (alpha1=1 is water) behavior in time in mesh with guide vane and blade.

The results shows that does not exist a bigger generation of cavitation in the guide vane, the affected zone of it is the closest at the trailing edge in the down side. In the analyzed time some bubbles are generated in the mentioned zone.

In the blade exist the generation of cavitation in the initial zone of the leading edge in the right side. This cavitation is affected by re-entrant jets that cut the cavity many times and, by the other hand, the residual cavitation ends joining the created cavities near the draft tube zone. Exist also the continuation of the cavitating flow generated in the guide vane that is conducted through the left zone of the blade.

The next picture present the behavior of the cavitating flow in the draft tube zone:

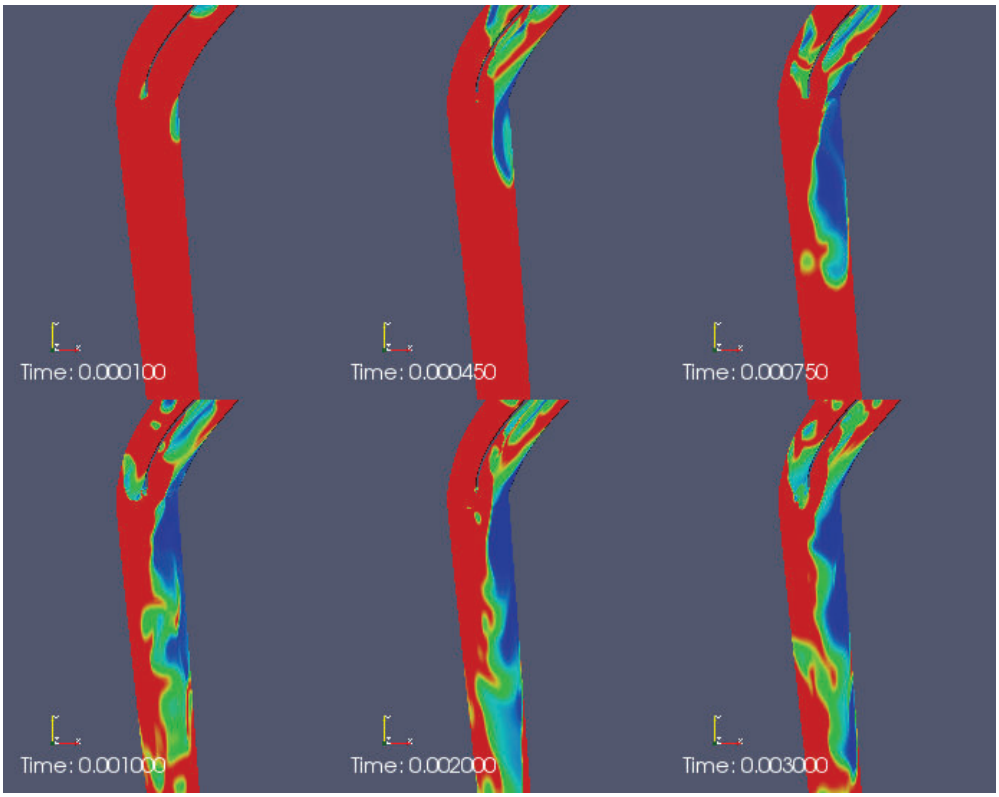


Figure 2.39: Vapor fraction (alpha1=1 is water) behavior in time in draft tube zone in mesh with guide vane and blade.

In the draft tube zone exist the presence of a cavitating flow in a severe condition. The supercavity grows in time and seems that does not stop when it is produced. This could produce severe damage in the assembly of the turbomachinery.

3 RESULTS AND DISCUSSION

3.1 COMPARISON OF THE CAVITATION MODEL DEVELOPED IN OPENFOAM WITH EXPERIMENTAL DATA

The quantitative analysis of the phenomenon of cavitation in Francis turbines is not possible to do due to the reduced research made and the differences presented between one and other model of the turbines. In this thesis are analyzed the phenomenons of generation of re-entrant jets and the main types of cavitation in Francis turbines with emphasis in the vortex rope in the draft tube.

When a cavity is produced in hydrofoils exist the generation of re-entrant jets in the area near the surface, as is mentioned in partial cavities in section 1.2.4, in research like (Hidalgo, Luo, & Yu, 2014) and (Hidalgo, Luo, Ji, & Aguinaga, 2014) is explained this phenomenon, the figures of it are presented below:

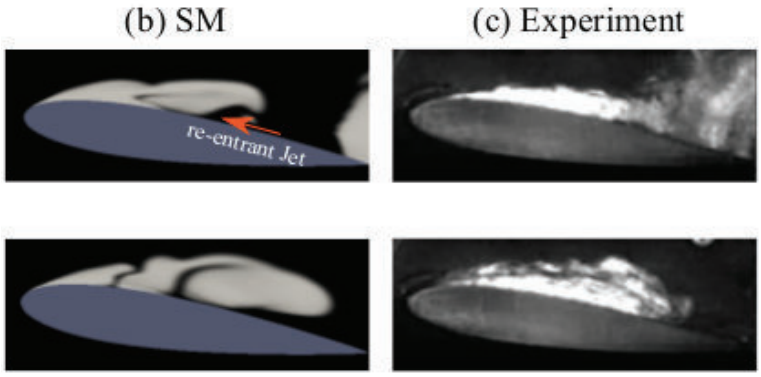


Figure 3.1: Re-entrant jets phenomenon in NACA 0015 hydrofoil. (b) Simulation with structured mesh (SM) and (c) Experimental results (Hidalgo, Luo, & Yu, 2014, p.7).

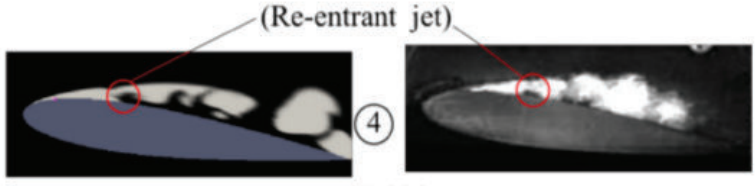
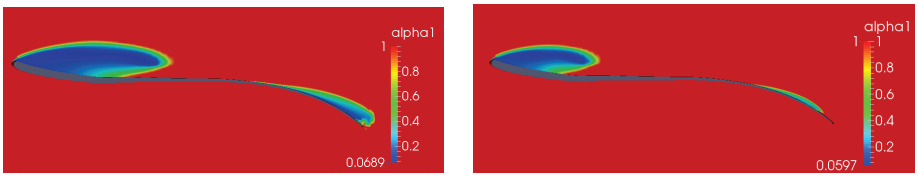


Figure 3.2: Re-entrant jets phenomenon in NACA 0015 hydrofoil. (left) Simulation and (right) Experimental result (Hidalgo, Luo, Ji, & Aguinaga, 2014, p.3280).

In the studied case the higher similitude with the presented pictures of the re-entrant jet phenomenon occur in the blade of the similitude comparison, indicated in picture 2.23, that is presented next:



$10[°], \sigma = 0.25, c = 0.05[cm]$ $10[°], \sigma = 0.5, c = 0.05[cm]$

Figure 3.3: Re-entrant jets phenomenon in studied hydrofoils.

In the behavior of the cavitation in the sections of the blade is also presented this phenomenon, like in the upper section of the Francis-99 turbine in the area near the blade, as is figure 2.35, which is presented below:

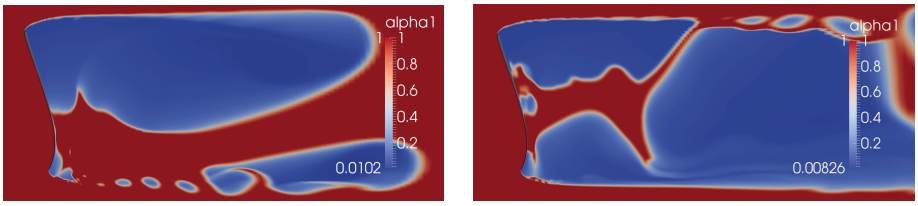


Figure 3.4: Re-entrant jets phenomenon in upper blade section of Francis-99 turbine.

The main types of cavitation produced in Francis turbines are presented in the next figure:

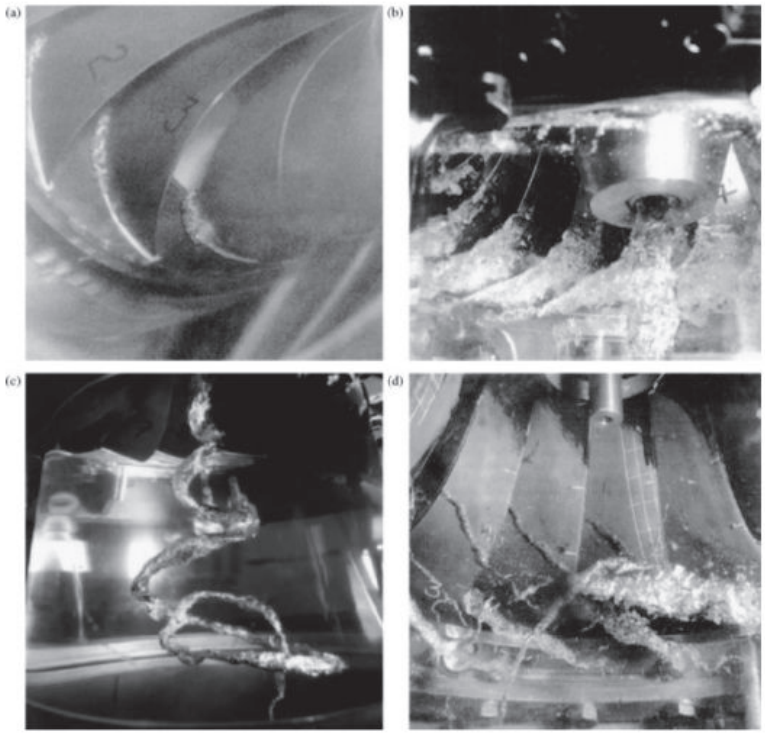


Figure 3.5: Main types of cavitation in Francis turbines: (a) leading edge cavitation, (b) travelling bubble cavitation, (c) draft tube swirl and (d) inter-blade vortex cavitation (NTNU, 2015, p.25).

As is explained in the text of (NTNU, 2015, p.24-25) these four phenomenon are:

- Leading edge cavitation
  Takes form of an attached cavity on the suction side of the runner blades, this type of cavitation is very aggressive and is likely cause several damage to the blades and provoke pressure fluctuations.

- Travelling bubble cavitation
  As the fluid moves across from a low pressure zone to higher pressure zones, a cyclic formation and collapse of bubbles is generated, the collapses of bubbles are noisy, but not necessarily harmful unless the bubbles collapse on a surface. When a bubble collapses on a surface, the liquid surrounding the bubble is first accelerated before abruptly decelerated as it collides with the surface.

- Draft tube swirl
  A low pressure zone in core of the flow is generated due to rotation and centrifugal forces. Hence, vortices are likely to cavitate in the core of the flow. The swirl may occur both on partial load due to the residual circumferential velocity component of the flow discharged from the runner. The cavitation itself is quite harmless as long as the vapor bubbles do not collapse at a surface. However, strong fluctuations may occur if the precession frequency matches one of the free natural oscillation frequencies of the draft tube or penstock.

- Inter-blade vortex cavitation
  Is formed by secondary vortices in the blade channels and is caused by flow separation. The cavitation is usually located in the area of the intersection between leading edge and hub and at the hub between two blades.

  In the study made of (Sanda, Dan, & Avellan, 2008) is explained the effect produced in the draft tube, in equal form than the analyzed in section 1.12, the obtained pictures are presented below:

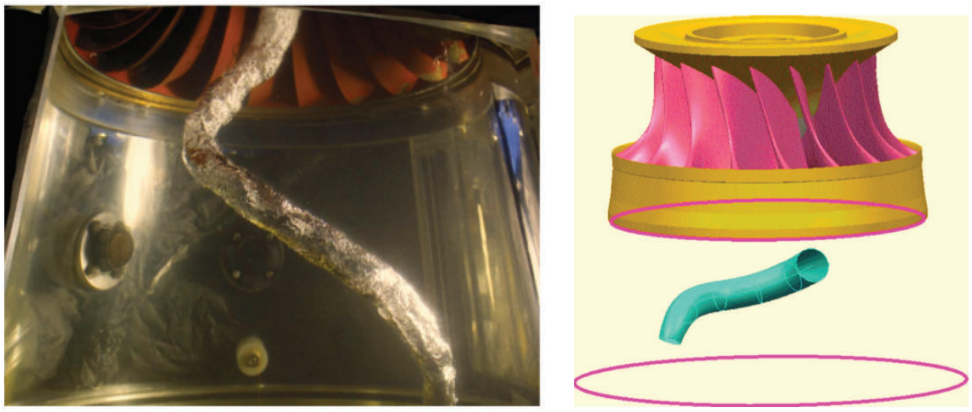


Figure 3.6: Cavitation draft tube vortex in a Francis turbine. (left) Development of a vapor core rope, (right) reconstruction of the rope volume (Sanda et al., 2008, p.2,7)

The variation of the behavior of the vortex in the draft tube was studied by (Avellan, 2004) and (Zhang, Liu, Wu, & Liu, 2012). Next is presented this behavior:

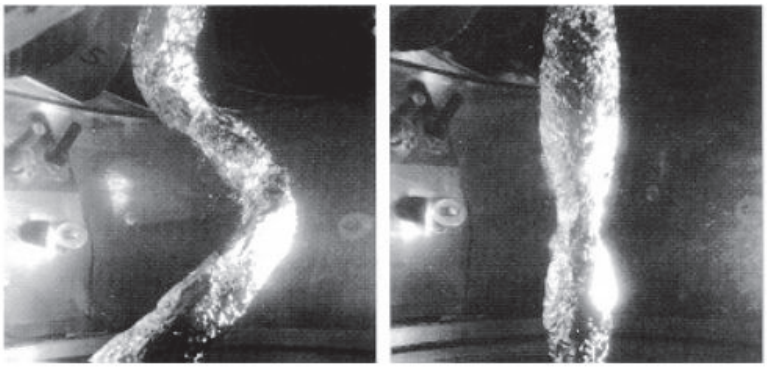


Figure 3.7: Cavitation whirl at low and high discharge operation, in a Francis turbine discharge ring (Avellan, 2004, p.8).

Figure 3.8: Vortex rope in draft tube (Zhang et al., 2012, p.7).

As seen in the pictures, the draft tube present a vortex rope of cavitation, this behavior occur in the analyzed pictures, like figures 2.38 and 2.39, this prove that the simulated effect correspond with the experimental results in a qualitative form.

Leading edge cavitation and travelling bubbles are also present in the behavior of the simulated mesh with the guide vane and blade of the runner, exist the presence of cavities in the leading edge zone of the blade and the correspondence in the effect of travelling in time, this is present in the figures 2.38 and 2.39 too, included in section 2.6.7.

3.2 ASSESSMENT OF THE CAVITATION PROBLEM ON THE HYDROFOIL

In the analysis realized in the section 2.6 is proved that the phenomenon of cavitation is a recurrent effect produced in turbines.

The flow conditions, calculated in the appendix G and presented in the section 2.6.1, show that the cavitation number, σ, is low in the prototype in BEP condition of Francis-99 turbine. This low number show supercavitating flow due to the angle between the velocity and the blade in the analysis.

Exist a relation in the delay presented between the increment of the size of the mesh that show the difference produced when a mesh grows as is indicated in the end of the section 2.6.5.

Due to the variation of the blade sections in Francis-99 turbine the analysis was reached. In this case, the results, as mentioned in section 2.6.6, showed the existence of supercavitation that is more aggressive in the lower section of the blade due to the geometry.

When the simulation of the mesh with the guide vane and the blade was performed, the reached results, included in the section 2.6.7, shown that the affected zone by the cavitating flow are the blade and the draft tube. The presence of big cavities is continuous in time.

For the aforementioned reasons is proved the magnitude of the cavitating flow behavior in the blade and the complete turbomachinery, this thesis form a preliminary study that open doors to more specific and optimized simulations.

3.3 DETERMINATION OF BREAK OFF CYCLE FOR THE CAVITATION BUBBLE

In the analysis of the behavior of the cavitation it is important to determine if exist a break off cycle for the bubble due to the possibility to find acoustic signals of the presence of cavitation in the experimental case in the turbine. In the present thesis this process was realized integrating the vapor content in each mesh time and making a Laplace transform of the point to define the frequencies.

Below is present the result of the analysis in the mesh with guide vane and blade:

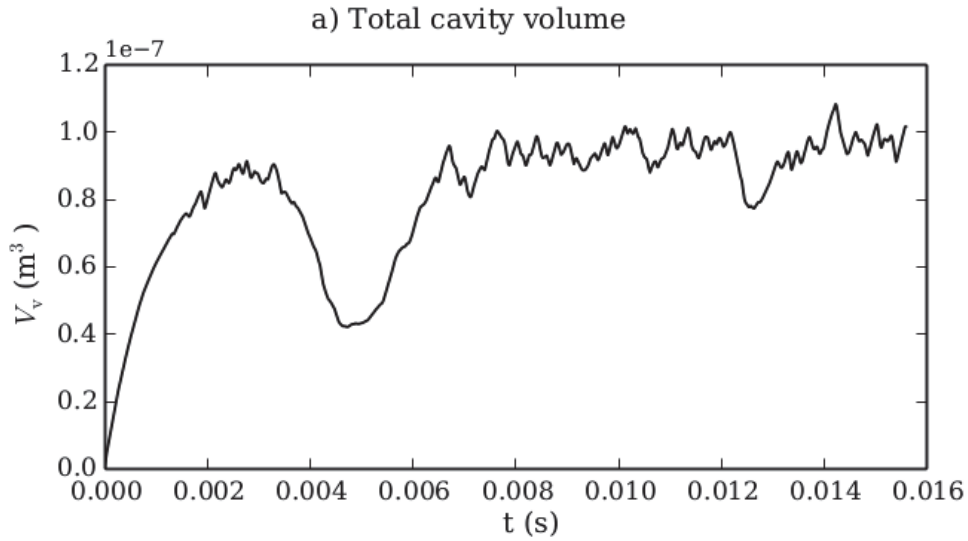


Figure 3.9: Cavity volume in time for mesh with guide vane and blade.

Exist a tendency to reach a maximum in approximately $0.9 * 10^{-7}[m^3]$ and the curve is composed by two behaviors to analyze.

The first part correspond a time with a initial development of the cavitation. The cavities volume begin from 0 to a maximum and decreases to a minimum value of approximately $0.4 * 10^{-7}[m^3]$. In the second part the cavities grows again to reach another apparent maximum with approximately the same value. In this picture has not to be take in account the first part to find the frequency due to the initial development process that correspond a period of stabilization of the simulation.

As seen in the graphic exist a tendency to reach an apparent constant volume of the cavities in the right part with high analysis times. Due to the high computational resources required to perform the simulation is not possible to reach more time. To have better resolution without peaks would be useful to reduce the time step.

The behavior of the cavities in time is presented in the next graphic:

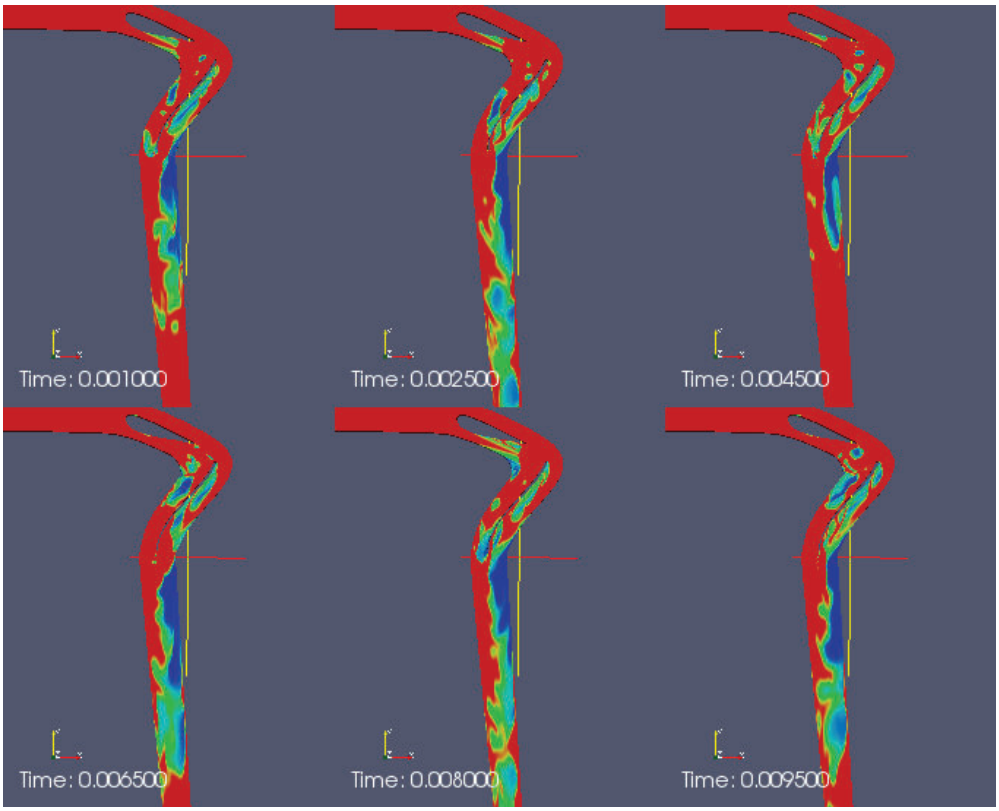


Figure 3.10: Cavities behavior in time.

As was mentioned in the explanation of the figure 3.9 exist a reduction of the volume of the cavities in a minimum due to the initial development of cavitation in the simulation process. From times bigger than $0.0065[s]$ it is observed that exist a certain quantity of bubbles in time that remain constant in the draft tube zone. The variation of the quantity of the bubbles it is more evident in the blade zone with a clear variation of the volume in time, due to the movement and implosion of the cavities.

In the next graphic is included the frequency of the peaks from 0.007 to 0.012[s] and its behavior at a better scale.

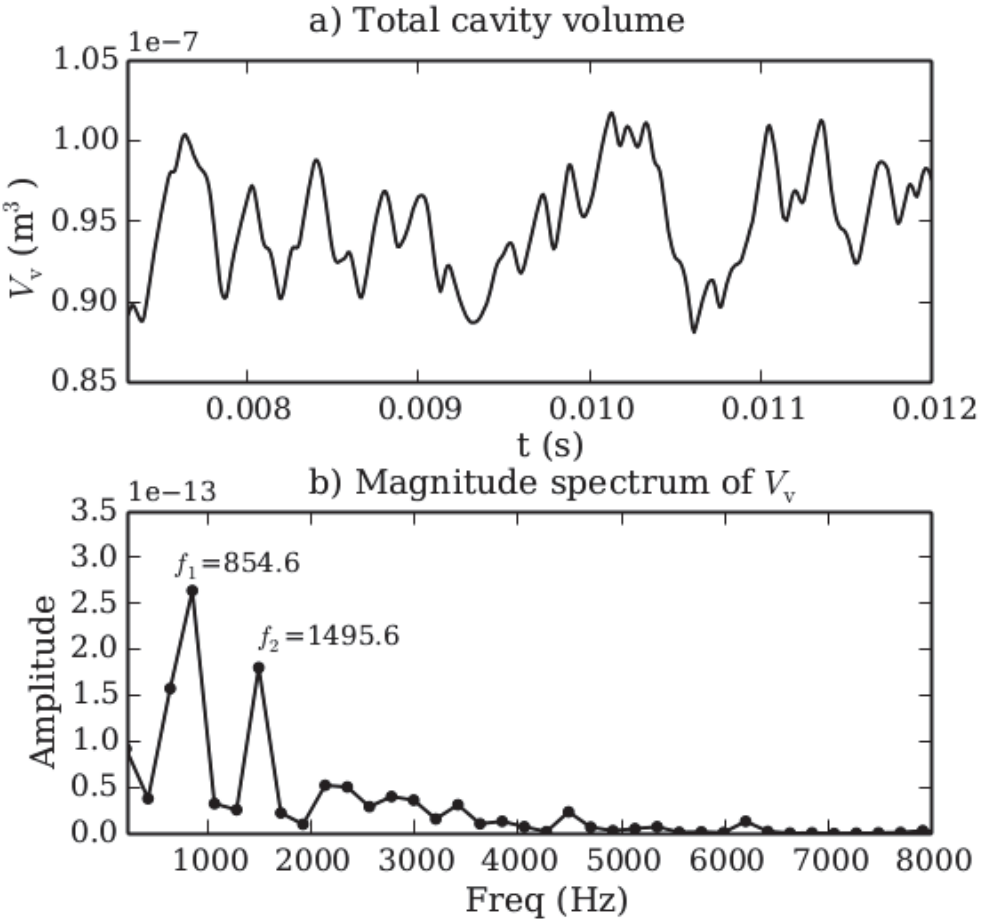


Figure 3.11: Cavity volume in time and frequency for mesh with guide vane and blade.

The behavior of the volume in time show principally two things:

- Exist a constant part of volume in time from about $0.9 * 10^{-7}[m^3]$. This behavior correspond at the effect produced in the draft tube zone when the vortex rope is generated.
- The graphic show clearly the effect of generation and implosion of the bubbles in time. This phenomenon has a random behavior that generate peaks in a varied form.

The part b) of the graphic present the frequencies generated in the simulation. The first one has a value of $f_1 = 854.6[Hz]$ with an approximate amplitude of $2.5 * 10^{-13}[m^3]$. The second one has a value of $f_2 = 1495.6[Hz]$ with an approximate amplitude of $1.7 * 10^{-13}[m^3]$.

The two found amplitudes describe a period T of:

$$T_1 = \frac{1}{f_1} = \frac{1}{854.6} \approx 0.0012[s] \quad T_2 = \frac{1}{f_2} = \frac{1}{1495.6} \approx 0,00067[s] \tag{3.1}$$

4 CONCLUSIONS & RECOMMENDATIONS

- The phenomenon of cavitation is still in research; actually it is possible to define some different aspects that conform it: The form to begin the phenomenon, described by the nucleation and inception, the behavior of the bubble, including the growing, travelling and collapse, the cavitation types produced, like sheet, vortex, shear, supercavitation and partial cavities, and finally the effects produced, for example noise, damage, luminescence and erosiveness.

- Francis-99 turbine is formed by a runner of $3.215[m]$ of diameter which generate approximately $106[MW]$ in BEP conditions. The research of the conditions was made in a model runner with a diameter of $0.63[m]$ with a generation capacity of $22.0[kW]$ by Noewegian University of Science and Technology - NTNU and Norway and Lulea University of Technology - LTU, in the two cases the obtained cavitation number, σ was 1.974 and 0.061 respectively. Due to the high computational resources required to perform the simulation of the real scale mesh, the chord length assumed was $c = 0.05[m]$.

- Using the computational open-source package denominated "Gmsh" was possible to realize the structured meshes for all simulations. The process to obtain the mesh was perform thanks another open-source software denominated "Python", with this scientific environment "Spyder", and "Blender". Python was used to run scripts and Blender to obtain the points.

- To validate the mesh was necessary to satisfy principally two parameters: y^+ must be smaller than 10 and the minimal hexahedral quality mesh distortion value of 0.9. Another values like the omega number, Ω, number of elements, NE, courant number, C were taken in account to reach accurate results.

- Due to the use of OpenFOAM software with the ILES solver developed by PhD candidate, engineer Victor Hugo Hidalgo MSc., was possible to simulate all the different situations proposed in this thesis. Was possible to determine the behavior of the vapor fraction in time for blades in different angles and size, different sections of the runner blade and also a mesh that include the guide vane and the blade in one.

- This thesis conform a preliminary study of the behavior of the cavitating flow inside a turbine. With the obtained results and the methodology used its possible to continue the research of more relationships and behaviors of cavities in time.

- For a future work it is possible to analyze the effect of the cavitation and the erosion in the blade and the damage produced, a real scale size analysis of the effects produced, the optimization of the meshes to generate better results, the analysis of another work conditions of the turbine and the analysis of these behaviors in power plants in the country.

- This thesis is an innovative project of the use of open-source software to perform cavitation simulations. This open doors to continue investigations in the field of fluids mechanics and leave a possibility to improve the methodology, the processes and the results.

REFERENCES

Abad, A. (2015, January). *Interview in a technical visit of Fluid Mechanics students.* (*Facultad de Ingeniería Mecánica - Escuela Politécnica Nacional*, Unidad de Negocio Hidroagoyán, Baños - Ecuador)

Aguinaga, A., Hidalgo, V. H., Valencia, E. A., Luo, X. W., & Cando, E. (2014). Simulación numérica de flujo inestable con validación experimental para desarrollar un método que permita reducir la erosión y optimizar la operación de las turbinas Francis de la Central Hidroeléctrica Agoyán. *Escuela Politécnica Nacional - Facultad de Ingeniería Mecánica - PIMI1403*.

Avellan, F. (2004). Introduction to Cavitation in Hydraulic Machinery. *The 6th International Conference on Hydraulic Machinery and Hydrodynamics*.

Bakker, A. (2002a). *Lecture 10 - Turbulence models, Applied Computational Fluid Dynamics.* Presentation. Retrieved from http://www.bakker.org/dartmouth06/engs150/10-rans.pdf

Bakker, A. (2002b). *Lecture 1, Introduction to CFD.* Presentation. Retrieved from http://www.bakker.org/dartmouth06/engs150/01-intro.pdf (Applied Computational Fluid Dynamics)

Berchiche, N., Franc, J. P., & Michel, J. M. (2002). A Cavitation Erosion Model for Ductile Materials. *Journal of Fluids Engineering*.

Bhaskaran, R. (N/A). *MAE 4230/5230: Introduction to CFD.* Retrieved from http://dragonfly.tam.cornell.edu/teaching/mae5230-cfd-intro-lec1.pdf (School of Mechanical & Aerospace Engineering, Cornell University)

Bhaskaran, R., & Collins, L. (2011). *Introduction to CFD Basics.* Retrieved from http://dragonfly.tam.cornell.edu/teaching/mae5230-cfd-intro-notes.pdf (Computational Science and Engineering.)

Blender™. (2015, June). *The software.* Retrieved from https://www.blender.org/about/

Brennen, C. (2005). *Fundamentals of Multiphase Flows* (C. University, Ed.). California Institute of Technology.

Brennen, E. (1995). *Cavitation and bubble dynamics* (O. University, Ed.). OPEN.

Caupin, F., & Herbert, E. (2006). Cavitation in water: a review. *C. R. Physique 7*.

CELEC-EP. (2013, April). *Plan estratégico 2013 - 2017.* Retrieved from https://www.celec.gob.ec/images/pdf/Pres_PE_2013_2017.pdf

CFD-Online. (2015a, July). *Courant-Friedrichs-Lewy condition.* Retrieved from http://www.cfd-online.com/Wiki/Courant%E2%80%93Friedrichs%E2%80%93Lewy_condition

CFD-Online. (2015b, May). *Dimensionless wall distance (y plus).* Retrieved from http://www.cfd-online.com/Wiki/Dimensionless_wall_distance_(y_plus) (Category: Dimensionless parameters)

CFD-Online. (2015c, May). *First cell height calculation.* Retrieved from http://www.cfd-online.com/Wiki/First_cell_height_calculation

CFD-Online. (2015d, May). *Large eddy simulation (LES).* Retrieved from http://www.cfd-online.com/Wiki/Large_eddy_simulation_(LES)

CFD-Online. (2015e, May). *Mesh adaptation.* Retrieved from http://www.cfd-online.com/Wiki/Mesh_adaptation

CFD-Online. (2015f, July). *What is Zero Gradient for Opening bound condi-*

tion? Retrieved from `http://www.cfd-online.com/Forums/cfx/22039-what-zero-gradient-opening-bound-condition.html` (Commentary of ghorrocks - Super Moderator)

Chawner, J. (2013, March). *Quality and Control, Two Reasons Why Structured Grids Aren't Going Away.* Retrieved from `http://www.pointwise.com/theconnector/March-2013/Structured-Grids-in-Pointwise.shtml`

Eisenberg, P. (N/A). *Cavitation.* Hydronautics Incorporated.

Fabula, A. (1958). *Some experiments in cavitation bubble dynamics* (Unpublished doctoral dissertation). California Institute of Technology.

Fernandez, P. (n.d.). *Turbinas Hidráulicas.* Universidad de Cantabria.

Franc, J. P. (2009). Incubation Time and Cavitation Erosion Rate of Work-Hardening Materials. *Journal of Fluids Engineering*.

Franc, J. P., & Michel, J. M. (2005). *Fundamentals of Cavitation* (R. Moreau, Ed.). Kluwer Academic Publishers.

Geuzaine, C., & Remacle, J. (2015, June). *Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities.* Retrieved from `http://geuz.org/gmsh/`

Hidalgo, V. H., Escaler, X., Soto, R., Valencia, E. A., Cando, E., & Luo, X. W. (2015). Large Eddy Simulation of Partial Cavitation Around a 2D Plane-Convex Hydrofoil. *Revista Politécnica, Vol.35, No.3*.

Hidalgo, V. H., Luo, X. W., Escaler, X., Ji, B., & Valencia, E. A. (n.d.). *A Cavitation Erosion Model Based on Homogeneous Mixture Flow Assumption.* (Unpublished)

Hidalgo, V. H., Luo, X. W., Escaler, X., Ji, J., & Aguinaga, A. (2014). Numerical investigation of unsteady cavitation around a NACA 66 hydrofoil using OpenFOAM. *IOPscience*.

Hidalgo, V. H., Luo, X. W., Ji, B., & Aguinaga, A. (2014). Numerical study of unsteady cavitation on 2D NACA0015 hydrofoil using free/open source software. *China: Springer*.

Hidalgo, V. H., Luo, X. W., Peña, A. P., Valencia, E. A., Soto, R., & Yu, A. (N/A). Benefits of hydropower research in Ecuador using OpenFOAM based on CFD technology. A practical cavitation study for NACA0015.

Hidalgo, V. H., Luo, X. W., & Yu, A. (2014). Cavitating flow simulation with mesh development using Salome opensource software. *ICHD, Singapore.*.

IEC. (1999). *Hydraulic turbines, storage pumps and pumpturbines, Model acceptance tests, IEC60193SE.*

Khurana, S., Singh, N., & Singh, H. (2012). Effect of Cavitation on Hydraulic Turbines - A Review. *International Journal of Current Engineering and Technology*.

Kirschner, I., Gieseke, T., & Stinebring, D. (2001). Supercavitation research and development. *Anteon Corporation, Naval Undersea WCD, Penn State University*.

Kuiper, G. (2010). *Cavitation in Ship Propulsion.* Delft University of Technology.

Kuzmin, D. (N/A). *Introduction to Computational Fluid Dynamics.* Retrieved from `http://www.mathematik.uni-dortmund.de/~kuzmin/cfdintro/lecture1.pdf` (kuzmin@math.uni-dortmund.de)

Leighton, T., Farhat, M., Field, J., & Avellan, F. (2003). Cavitation luminescence from flow over a hydrofoil in a cavitation tunnel. *Journal of Fluid Mechanics*.

Lyer, C., & Ceccio, S. (2002). The influence of developed cavitation on the flow of a turbulent shear layer. *Physics of fluids*.

NTNU. (2015, September). *Hydraulic design of Francis turbine exposed to sediment erosion.* Retrieved from http://www.diva-portal.org/smash/get/diva2:536445/FULLTEXT01.pdf

NTNU, & LTU. (2015a, June). *Francis-99.* Retrieved from www.ltu.se/research/subjects/Stromningslara/Konferenser/Francis-99 (Norwegian University of Science and Technology, Norway and Lulea University of Technology)

NTNU, & LTU. (2015b, June). *Francis-99 - Test Case.* Retrieved from http://www.ltu.se/cms_fs/1.111520!/file/Runner.zip (Runner zip file in ANSYS ICEM CFD®)

OpenFOAM. (2015, May). *User Guide - The Open Source CFD Toolbox* (2.4.0 ed.). Retrieved from http://foam.sourceforge.net/docs/Guides-a4/UserGuide.pdf

OpenFOAM-Fundation. (2015, June). *Features of OpenFOAM.* Retrieved from http://www.openfoam.org/features/index.php

ParaView. (2015, June). *Large Data Visualization Made Easier.* Retrieved from http://www.paraview.org/overview/

Sanda, M., Dan, G., & Avellan, F. (2008). Analysis of the Cavitating Draft Tube Vortex in a Francis Turbine Using Particle Image Velocimetry Measurements in Two-Phase Flow. *Journal of Fluids Engineering*.

Sandia National Laboratories. (2007). *The Verdict Geometric Quality Library.* Retrieved from http://www.csimsoft.com/download?file=Documents/sand20071751.pdf

Sayma, A. (2009). *Computational Fluid Dynamics.* Retrieved from http://www.leka.lt/sites/default/files/dokumentai/computational-fluid-dynamics.pdf

SCCMPC. (N/A). Final Report and Recommendations to the 22nd ITTC. *Specialist Committee on Computational Method for Propeller Cavitation*.

Slffea, & Mysid. (2006, November). *Curvilinear grid.svg.* Retrieved from http://upload.wikimedia.org/wikipedia/commons/6/65/Curvilinear_grid.svg (An example of a curvilinear grid)

Sodja, J. (2007). Turbulence models in CFD. *University of Ljubljana, Faculty for mathematics and physics - Department of physics*.

Stoessen, L. (2014). *Numerical simulations of the flow in the Francis-99 turbine* (Unpublished master's thesis). Chalmers University of Technology.

Sturgeon, V. (2001). Racing through water: Supercavitation.

Timbobong, D. (2015, May). *h-Method & p-Method.* Retrieved from https://deust.files.wordpress.com/2014/11/original_h_p-1.jpg

Wikipedia. (2015a, June). *Courant-Friedrichs-Lewy condition.* Retrieved from https://en.wikipedia.org/wiki/Courant-Friedrichs-Lewy_condition

Wikipedia. (2015b, July). *Filter (large eddy simulation).* Retrieved from https://en.wikipedia.org/wiki/Filter_(large_eddy_simulation)

Wikipedia. (2015c, June). *OpenFOAM.* Retrieved from https://en.wikipedia.org/wiki/OpenFOAM

Wikipedia. (2015d, June). *Python (programming language).* Retrieved from http://en.wikipedia.org/wiki/Python_(programming_language)

Wikipedia. (2015e, June). *Spyder (software).* Retrieved from http://en.wikipedia.org/wiki/Spyder_(software)

Wikipedia. (2015f, May). *Supercavitation.* Retrieved from `http://en.wikipedia.org/wiki/Supercavitation`

Zhang, L., Liu, J., Wu, Y., & Liu, S. (2012). Numerical simulation of cavitating turbulent flow through a Francis turbine. *26th IAHR Symposium on Hydraulic Machinery and Systems*.

Zureks. (2007, July). *Example of 2D mesh.png.* Retrieved from `http://upload.wikimedia.org/wikipedia/commons/8/80/Example_of_2D_mesh.png` (Example of a two-dimensional FEM (finite element method) mesh for a cylindrically shaped magnetic shield)

APPENDIX A COMPUTATIONAL FLUID DYNAMICS (CFD)

The study of the fluid motion can be performed in three different ways: experimental, theoretical and numerically. The last one concerns at computational fluid dynamics (CFD), which is able to predict fluid flow and also heat transfer, mass transfer, chemical reactions and related phenomena by solving the mathematical equations which governs these processes (Bakker, 2002b, p.14). CFD enables scientists and engineers to perform "numerical experiments" in a "virtual flow laboratory" (Kuzmin, N/A, p.3). In the next picture is present the difference between a real experiment and a CFD simulation of Von Kármán vortex street:

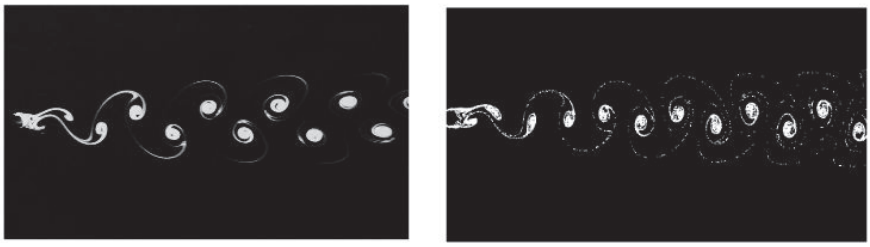


Figure A.1: Difference between a real experiment and a CFD simulation of Von Kármán vortex street (Kuzmin, N/A, p.3)

A.1 PROCESS OF ANALYSIS

The parameters which have to be defined to perform an analysis while using CFD are (Bakker, 2002b, p.15-16) (Kuzmin, N/A, p.12):

1. Problem statement: Information about the flow.
2. Mathematical model: $IBVP = PDE + IC + BC$.
   $IBVP$: Initial Boundary Value Problem.
   PDE: Partial Differential Equation.
   IC: Initial Conditions.
   BC: Boundary Conditions.

- Fluid properties are modelled empirically.
- Simplifying assumptions are made in order to make the problem tractable (e.g., steady-state, incompressible, inviscid, two-dimensional).
- Provide initial and boundary conditions.

3. Meshing: Nodes/cells, time instants.
4. Space discretization: Apply numerical methods to develop approximations of the equations in the region of interest.

5. Time discretization: Transformation of governing differential equations in algebraic.
6. Iterative solver of algebraic equations at each node or cell to obtain discrete function values.
7. CFD software: Implementation and debugging.
8. Simulation run: Solving of system of equations to provide a solution with defined parameters and stopping criteria.
9. Post-processing of the solutions to extract quantities of interest (e.g. lift, drag, torque, heat transfer, heat loss, etc.), visualization and analysis data.
10. Verification: Model validation and adjustment.

A.2 APPLICATIONS

The CFD can be used in a lot of applications (Bakker, 2002b, p.31) (Kuzmin, N/A, p.4-7) (Bhaskaran & Collins, 2011, p.2) (Sayma, 2009, p.12-17), for example:

- Flow and heat transfer in industrial processes, propulsion and power generation systems. Analysis of the temperature distribution.
- Design comfortable and safe living environment.
- Aerodynamics of ground vehicles, aircraft, missiles. Interaction of propellers or rotors with the aircraft fuselage. Prediction of the pressure field.
- Film coating, thermoforming
- Ventilation, heating and cooling flows.
- Study of turbomachinery applications and cavitation problems. Designing and optimization of turbines, compressors and components.
- Chemical vapor deposition (CVD) for integrated circuit manufacturing.
- Optimal oil recovery strategies.
- Computational hemodynamics to cure arterial diseases. Study the circulatory and respiratory systems.
- Heat transfer for electronics packaging applications.
- Forecast the weather and warn of natural disasters.

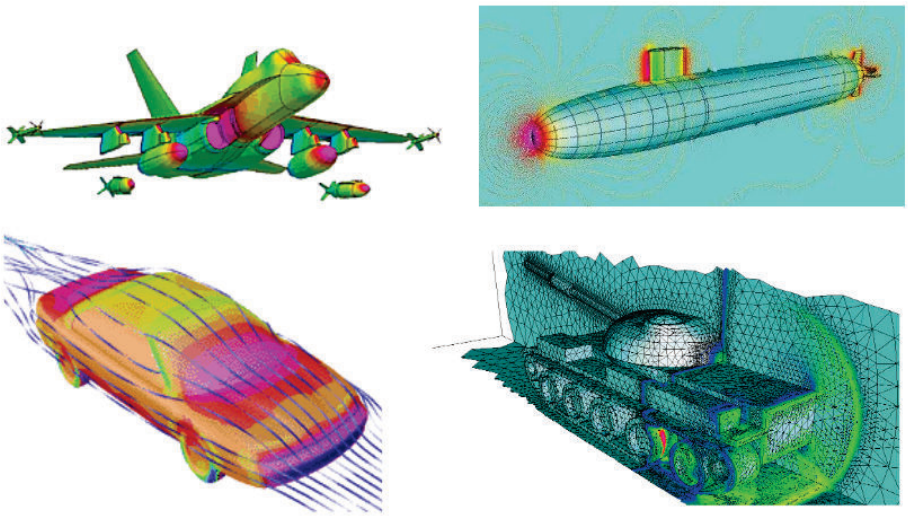


Figure A.2: Applications of CFD in aerodynamic design (Kuzmin, N/A, p.5-6).

A.3 ADVANTAGES & LIMITATIONS

- Advantages (Bakker, 2002b, p.32-33) (Kuzmin, N/A, p.8)
  - Relatively low cost compared by using physical experiments and test.
  - Simulation executed in a short period of time.
  - Ability to simulate real conditions or process that can not be easy simulated. Theoretically exists the possibility to simulate any physical condition.
  - Ability to simulate ideal conditions and isolate specific phenomena.
  - Complete information of a large number of locations in the region of interest.
  - Equipment and personnel are difficult to transport, CFD software is portable, easy to use and modify.
- Limitations (Bakker, 2002b, p.34-35) (Kuzmin, N/A, p.8-9)
  - In experiments error sources are: measurement error or flow disturbances by the probes, in simulations error are from: modeling, discretization, iteration and implementation.
  - CFD does not replace the measure completely. Result are never 100% reliable because imprecision data, inadequate mathematical models, unavailable computing power, generating of errors.
  - CFD solutions only be accurate as the physical models on which are based.
  - Solving equations numerically involves errors: Round-off errors, due to finite word size available in the computer, and truncation error, due to approximations in the numerical models that the mesh refinement can not solve.
  - The accuracy of the CFD solution is only as good as the initial/boundary conditions provided.

The basic concept of CFD methods is then to find the values of the flow quantities at a large number of points in the system. These points are usually connected together in what is called numerical grid or mesh. The system of differential equations representing the flow is converted, using some procedure, to a system of algebraic equations representing the interdependency of the flow at those points and their neighboring points. If the flow is unsteady, due to varying boundary conditions or inherent unsteadiness, the solution procedure is repeated at discrete time intervals to predict the evolution in time (Sayma, 2009, p.11).

A.4 STRATEGY OF CFD

The approach of CFD is to transform a continuous domain to a discrete domain to obtain approximate computer solutions (Bhaskaran, N/A, p.7). The discretized form of a differential equation can be derived in many ways, but the more common methods are (Sayma, 2009, p.35-42):

- Finite difference method: Approximating of the derivatives in the differential equation using a truncated Taylor series.
- Finite element method: called also "Method of weighted residuals", in which an integration of a weighting function expect vanish of the residual over the solution domain, minimizing the error of approximation.
- Finite volume method: A number of weighted residuals equations are generated by dividing the solution domain in control volumes, weighting function are setting to be unity over the control volumes one at a time, and zero elsewhere. In this form the residual over each volume must become zero.

To a practical example, a finite difference approximation is realized for the next equation (Bhaskaran & Collins, 2011, p.4):

$$\frac{du}{dx} + u = 0 \tag{A.1}$$

$$0 \leq x \leq 1$$

$$u(0) = 1$$

With the expansion in a Taylor's series:

$$\left(\frac{du}{dx}\right)_i = \frac{u_i - u_{i-1}}{\Delta x} + O(\Delta x) \tag{A.2}$$

First a continuous domain is transformed in a discrete domain

$$p_i = p(x_i), \quad i = 1, 2, \ldots, N$$

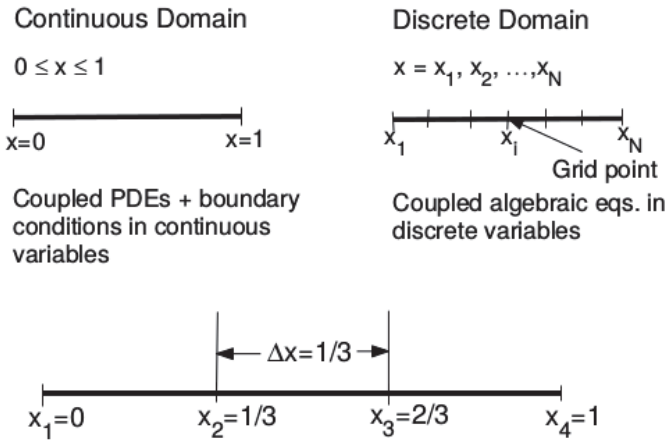


Figure A.3: Transformation from a continuous to a discrete domain assuming $\Delta x = 1/3$ (Bhaskaran & Collins, 2011, p.3-4).

The transformation is made assuming a $\Delta x = 1/3$. In this form four point are obtained: $0, 1/3, 2/3$ and 1. The equation A.1 can be denoted by:

$$\left(\frac{du}{dx}\right)_i + u_i = 0 \tag{A.3}$$

The error in $(du/dx)_i$ due to the neglected terms in the Taylor's series is called the truncation error. Since the truncation error above is $O(\Delta x)$, this discrete representation is termed first-order accurate (Bhaskaran & Collins, 2011, p.5).

Making the relation between A.2 and A.3 and excluding high-order terms in the Taylor's series, results the following discrete equation:

$$\frac{u_i - u_{i-1}}{\Delta x} + u_i = 0 \tag{A.4}$$

This equation is an algebraic equation transformed from the differential equation A.1 and is the based method used in the finite-differences.

Rearranging the equation A.4 is obtained the simplified form to apply the analyzed grid points $i = 2, 3, 4$.

$$-u_{i-1} + (1 + \Delta x)u_i = 0 \tag{A.5}$$

Applying the points gives:

$$-u_1 + (1 + \Delta x)u_2 = 0 \quad (i = 2) \tag{A.6}$$

$$-u_2 + (1 + \Delta x)u_3 = 0 \quad (i = 3) \tag{A.7}$$

$$-u_3 + (1 + \Delta x)u_4 = 0 \quad (i = 4) \tag{A.8}$$

In the left boundary, in which $i = 1$, cannot be applied the equation A.5 because u_{i-1} is not defined. In this case is used the boundary condition:

$$u_1 = 1 \tag{A.9}$$

From the equations can be performed a simultaneous algebraic equations system, which expressed in a matrix form result:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1+\Delta x & 0 & 0 \\ 0 & -1 & 1+\Delta x & 0 \\ 0 & 0 & -1 & 1+\Delta x \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{A.10}$$

Solving the system using $\Delta x = 1/3$ result:

$$u_1 = 1 \quad u_2 = 3/4 \quad u_3 = 9/16 \quad u_4 = 27/64 \tag{A.11}$$

The exact solution of the differential equation A.1 result:

$$u_{exact} = \exp(-x) \tag{A.12}$$

The comparison between the exact and the discrete solution is showed next:

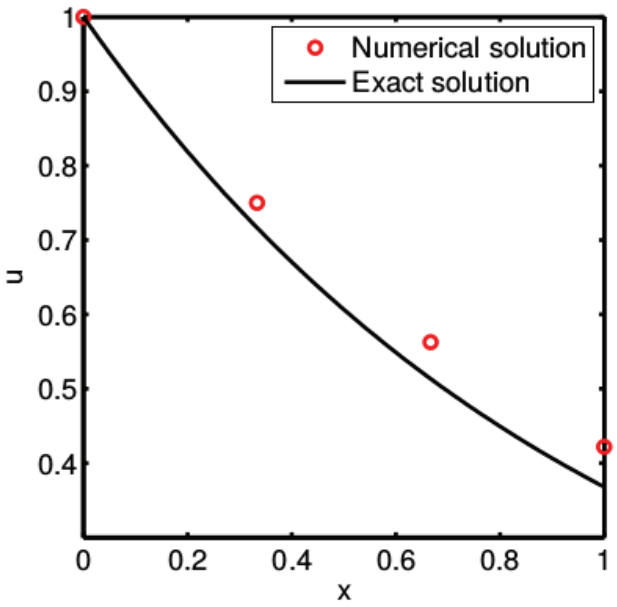


Figure A.4: Comparison of the results between the exact and the discrete solution of a finite-difference analysis (Bhaskaran & Collins, 2011, p.7).

The existence of a discrepancy between the discrete and the exact solution is due to the accumulation of errors in the analysis. If there's more subdivisions (x_i elements) the precision of the discrete solution is better.

In a real CFD application, thousands or millions of unknowns have to be solved. Computers and programmers have to make a good job to perform good methods of matrix inversions. While less is the CPU time and memory required is better, but this must be in accord to the precision and reliability obtained.

A.5 GRID CONVERGENCE

In a 1D problem, like the analyzed, the truncation error is $O(\Delta x)$. To decrease the error the number of grid points i's is increased and Δx is reduced. The agreement between the numerical from the exact solution is better, like in the next graphic:

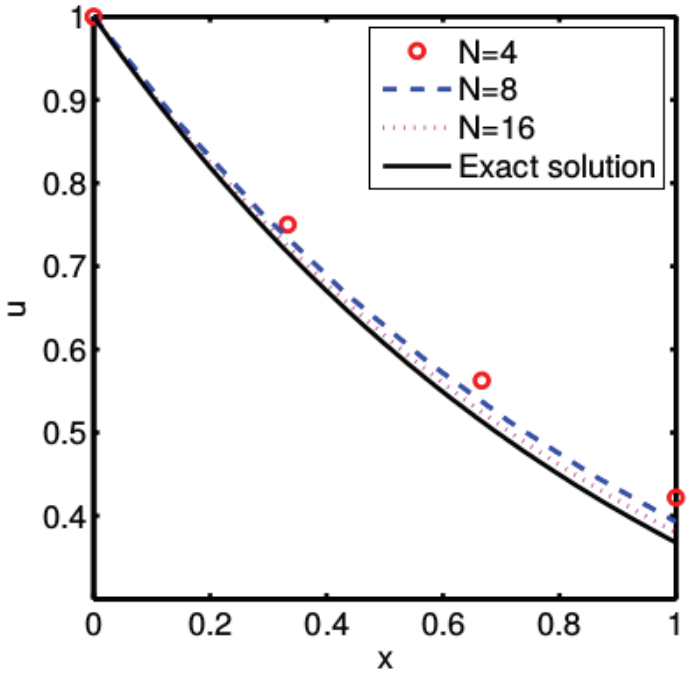


Figure A.5: Comparison of the results on a finite-difference analysis with different number of grid points (Bhaskaran & Collins, 2011, p.8).

When the numerical solutions agree within to a level of tolerance specified by the user, they are referred to as "grid converged" solutions (Bhaskaran & Collins, 2011, p.8). To trust on a CFD analysis is important to check if the solution is grid converged to an acceptable level of tolerance. This problem is dependent of each analysis performed.

A.6 NONLINEARITY

The highly nonlinear nature of the governing equations for a fluid makes it challenging to obtain accurate numerical solutions for complex flows of practical interest (Bhaskaran & Collins, 2011, p.9). To demonstrate the effect of nonlinearity a variation of the equation A.1 is analyzed:

$$\frac{du}{dx} + u^2 = 0; \quad 0 \leq x \leq 1; \quad u(0) = 1 \tag{A.13}$$

A first order finite-difference approximation is:

$$\frac{u_i - u_{i-1}}{\Delta x} + u_i^2 = 0 \tag{A.14}$$

Where the source of the nonlinearity is the u_i^2 term.

To deal with the nonlinearity a linearization about a guess value of the solution is made. This guess value must agree with the solution at a specified level of tolerance. In the example, u_{g_i} is the guess for u_i and Δu_i is the difference between the two terms:

$$\Delta u_i = u_i - u_{g_i} \tag{A.15}$$

Rearranging and squaring the equation:

$$u_i^2 = u_{g_i}^2 + 2u_{g_i}\Delta u_i + (\Delta u_i)^2 \tag{A.16}$$

Assuming that $\Delta u_i \ll u_{g_i}$ the term $(\Delta u_i)^2$ can be neglected, obtaining:

$$u_i^2 \simeq u_{g_i}^2 + 2u_{g_i}\Delta u_i = u_{g_i}^2 + 2u_{g_i}(u_i - u_{g_i}) \tag{A.17}$$

Thus,

$$u_i^2 \simeq 2u_{g_i}u_i - u_{g_i}^2 \tag{A.18}$$

The finite-difference approximation after linearization becomes:

$$\frac{u_i - u_{i-1}}{\Delta x} + 2u_{g_i}u_i - u_{g_i}^2 = 0 \tag{A.19}$$

The error due to linearization is $O(\Delta u^2)$, it tends to zero as $u_g \to u$.
To calculate the finite-difference approximation the guess of u_g is made by an iterative method. It is necessary to iterate through successive approximations until the iterations converge (Bhaskaran & Collins, 2011, p.9).

The residual R is defined as the "small" difference between u_g and u:

$$\frac{|u - u_g|}{|u|} \tag{A.20}$$

There are two different residuals: scaled and unscaled (Bhaskaran, N/A, p.23). The first-one is defined as the RMS value. The second-one is defined as the scaled residual divided by the averaged value of u.

- Unscaled residual:

$$R = \sqrt{\frac{\sum_{i=1}^{N}(u_i - u_{g_i})^2}{N}} \tag{A.21}$$

- Scaled residual:

$$R = \left(\sqrt{\frac{\sum_{i=1}^{N}(u_i - u_{g_i})^2}{N}} \right) \left(\frac{N}{\sum_{i=1}^{N} u_i} \right) = \frac{\sqrt{N \sum_{i=1}^{N}(u_i - u_{g_i})^2}}{\sum_{i=1}^{N} u_i} \tag{A.22}$$

In the analyzed example the calculation of the residual is performed and stop when falls below 10^{-9}. The iterative process converges to a level smaller than 10^{-9} in 6 iterations, like is present in the next graphic. In more complex problems, a lot more iterations would be necessary for achieving convergence (Bhaskaran & Collins, 2011, p.11).

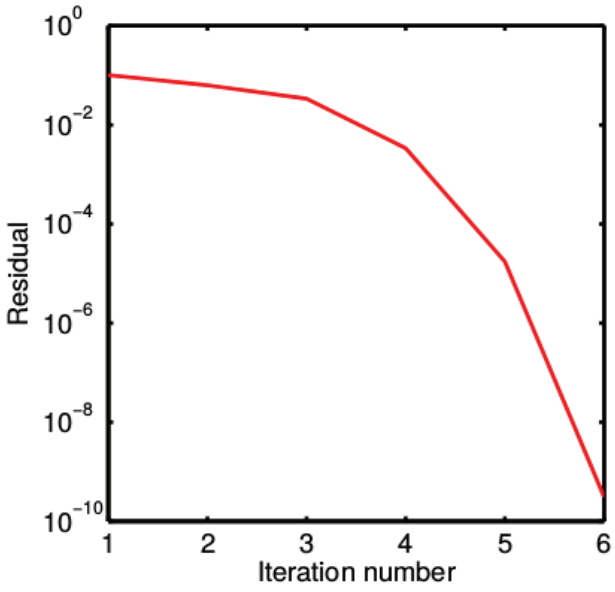


Figure A.6: Plot of the residual vs iteration number in the analyzed example (Bhaskaran & Collins, 2011, p.11).

A.7 TURBULENCE MODELING

Turbulent flows are characterized by large and nearly random fluctuations in velocity and pressure in both space and time. These fluctuations arise from instabilities that grow until nonlinear interactions cause them to break down into finer and finer whirls that eventually are dissipated into heat by the action of the viscosity in the flow (Bhaskaran & Collins, 2011, p.14).

Complexity of different turbulence models may vary strongly depends on the details that are wanted to observe and investigate by carrying out such numerical simulations. Complexity is due to the nature of Navier-Stokes equation which is inherently nonlinear, time-dependent, three-dimensional partial differential equation (Sodja, 2007, p.4).
In the next time-history graphic is included the behavior example of the flow variable u at a fixed point in a space. The dashed line in the graph (a) and (c) indicates the time average value.

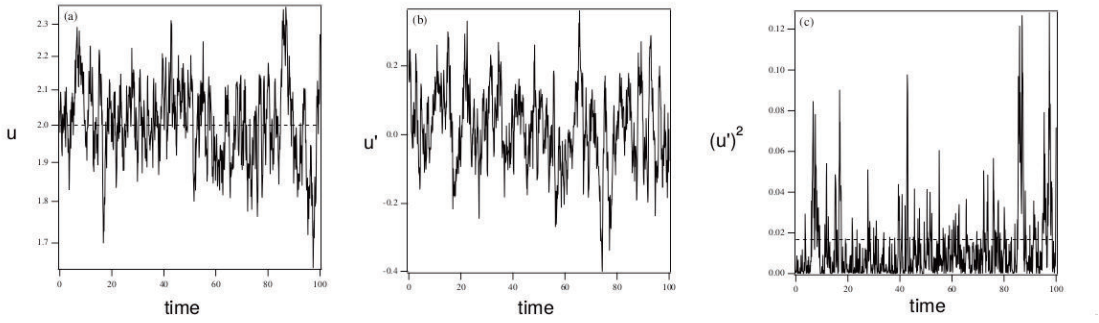


Figure A.7: Example of a time history of component of a fluctuating velocity at a point in a turbulent flow. (a) Velocity (b) Fluctuating component of velocity $u' \equiv u - \bar{u}$ (c) the square of the fluctuating velocity. Dashed lines (Bhaskaran & Collins, 2011, p.15).

Exists three types of average: time, volume and ensemble. The last one is rarely done because take a large number of experiments measuring the quantity of interest at the same position and time and averaging the quantity. Instead, time or volume average (or combination of the two) is made with the assumption that they are equivalent to the ensemble average (Bhaskaran & Collins, 2011, p.15). For example, the time average for a stationary flow is defined as:

$$\bar{u}(y) \equiv \lim_{\tau \to \infty} \frac{1}{2\tau} \int_{-\tau}^{\tau} u(y,t)dt \tag{A.23}$$

The fluctuation, defined as the deviation of the velocity from the mean value is

$$u' \equiv u - \bar{u} \tag{A.24}$$

By definition the fluctuation is zero ($u' = 0$). For that, a better measure of the "strength" of the fluctuation is the average of the square of the fluctuating variable (Bhaskaran & Collins, 2011, p.15-16). The $(u')^2$ term and its average value it is always greater than zero.

Nowadays turbulent flows may be computed using several different approaches. Either by solving the Reynolds-averaged Navier-Stokes equations with suitable models for turbulent quantities or by computing them directly. These approaches are summarized below (Sodja, 2007, p.4):

- Reynolds-Averaged Navier-Stokes (RANS) Models
  - Eddy-viscosity models (EVM)
    The turbulent stress is proportional to the main rate of strain. Eddy viscosity is derived from turbulent transport equations (k + other).
  - Non-linear eddy-viscosity models (NLEVM)
    Turbulent stress is modelled as a non-linear function of mean velocity gradients. Turbulent scales are determined by solving transport equations (k + other). Model is set to mimic response of turbulence to certain important types of strain.
  - Differential stress models (DSM)
    This category consists of Reynolds-stress transport models (RSTM) or second-order closure models (SOC). One model is required to solve transport equations for all turbulent stresses.

From this presented models is included another classification using classical models based on RANS time averaged equations (Bakker, 2002a, p.3):

- Zero equation model: mixing length model
- One equation model: Spalart-Almaras.
- Two equation models: $k - \varepsilon$ style models (standard, RNG, realizable), $k - \omega$ model, and ASM.
- Seven equation model: Reynolds stress model.

Where, the number of equations denotes the number of additional partial differential equations (PDEs) that are being solved. A comparison of the strengths and weaknesses of the method is mentioned in the next table:

Table A.1: Comparison of RANS turbulence models (Bakker, 2002a, p.46).

Model	Strengths	Weaknesses
Spalart-Allmaras	Economical (1-eq.); good track record for mildly complex B.L. type of flows.	Not very widely tested yet; lack of submodels (e.g. combustion, buoyancy).
STD k-ε	Robust, economical, reasonably accurate; long accumulated performance data.	Mediocre results for complex flows with severe pressure gradients, strong streamline curvature, swirl and rotation. Predicts that round jets spread 15% faster than planar jets whereas in actuality they spread 15% slower.
RNG k-ε	Good for moderately complex behavior like jet impingement, separating flows, swirling flows, and secondary flows.	Subjected to limitations due to isotropic eddy viscosity assumption. Same problem with round jets as standard k-ε.
Realizable k-ε	Offers largely the same benefits as RNG but also resolves the round-jet anomaly.	Subjected to limitations due to isotropic eddy viscosity assumption.
Reynolds Stress Model	Physically most complete model (history, transport, and anisotropy of turbulent stresses are all accounted for).	Requires more cpu effort (2-3x); tightly coupled momentum and turbulence equations.

- Computation of fluctuating quantities
  - Large-eddy simulation (LES)
    Computes time-varying flow, but models sub-grid-scale motions.
  - Direct numerical simulation (DNS)
    Navier-Stokes equations are numerically solved without any turbulence model.

LES method is used in this thesis to solve numerically the problem of cavitation, more explanation of the LES method is included in the section 2.5.

To understand better the difference between RANS, LES and DNS the three process are described in the next graphic:

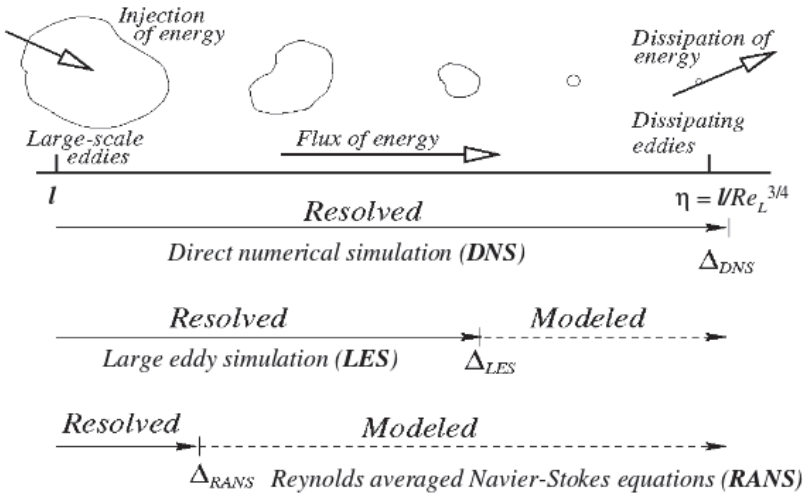


Figure A.8: Difference between prediction methods for CFD approach: DNS, LES, RANS (Sodja, 2007, p.6).

Where, DNS implies the complete resolution without modeling, LES reaches a higher resolution and lower modeling than RANS, but this implies a bigger computational power. Computing fluctuating quantities resolve shorter length scales than RANS but have the ability to provide better results (Sodja, 2007, p.5).

APPENDIX B CODE GENERATED IN SPYDER

B.1 BLENDER TO PLAIN POINTS

```
# -*- coding: utf-8 -*-
"""

 Software para transformar los puntos obtenidos en Blender
 Formato de puntos de entrada:
    |  x       z        y
v 0.000000 0.020606 0.000000
v 0.004897 0.020606 -0.014071
Formato de puntos obtenidos
    |     x        y
    |  0.000000 0.000000
    |  0.004897 -0.014071
"""
#Dimension del perfil
T=0.05 #m
archi1=open('puntos-middle.obj','r') #Lee informacion del archivo puntos-
    up.obj
archi2=open('puntos-middle.dat','w') #Crea el archivo puntos-up.dat
lineas=archi1.readlines() #Lee las lineas del archi1
archi1.close #Cierra el archi1
l1=len(lineas) #Longitud de vector lineas
l2=arange(l1) #Crea vector desde 1 a l1 con pasos de 1
prog=[0]*(l1) #Vector de ceros
X=[0]*(l1) #Vector de ceros
Y=[0]*(l1) #Vector de ceros
for i in l2: #Lazo para analizar los puntos
    a=lineas[i] #Toma valor independiente
    b=a.split(' ') #Divide donde exista un espacio
    cx=float(b[1]) #Convierte a flotante
    X[i]=cx    #Asigna valor a elemento de matriz X
    cy=float(b[3]) #Convierte en flotante
    Y[i]=cy    #Asigna valor a elemento de matriz Y
Xmax=max(X)  #Calcula el maximo de X
Xmin=min(X)  #Calcula el minimo de X
Ymax=max(Y)  #Calcula el maximo de Y
Ymin=min(Y)  #Calcula el minimo de Y
ancho=Xmax-Xmin  #Define el ancho del perfil
X[:] = [k / ancho * T for k in X] #Escalamiento del perfil en eje x
Y[:] = [k / ancho * T for k in Y] #Escalamiento del perfil en eje y
for lx in l2:  #Lazo para ingresar vectores en matriz prog
    p_x=X[lx] #Asigna valor a p_x
    p_y=Y[lx] #Asigna valor a p_y
    k=str(p_x)+' '+str(p_y)+'\n' #Convierte a texto y da espacio
    intermedio
    prog[lx]=k #ingresa nombre en vector prog
archi2.writelines( prog ) #Escribe lineas de prog en archi2
archi2.close #Cierra archi2
```

B.2 PLAIN POINTS TO GMSH FORMAT

B.2.1 BLADE WITH LOW ROTATION ANGLE

```
"""
 Software para transformar los puntos
 Formato de puntos de entrada:
 Opcion A: Con dos espacios entre puntos (uno para negativos)
           y dos espacios al inicio
      |  x      y
      |  0.001  0.012
      |  0.001 -0.012
 Opcion B: Sin espacios
      |  x      y
      |0.001 0.012
      |-0.001 -0.012
"""
#Longitud caracteristica, lc:
lc=1e-6
# Programacion:
archi1=open('puntos-middle.dat','r') #Leer archivo puntos.dat
archi2=open('puntos.geo','w') #Crear el archivo puntos.geo
lineas=archi1.readlines() #Lee las lineas del archi1
archi1.close #Cierra el archi1
l1=len(lineas) #Longitud de vector lineas
l2=arange(l1) #Crea vector desde 1 a l1 con pasos de 1
k0='lc = '+str(lc)+';\n' #Longitud caracteristica
#Textos a incluir
k1='Point('
k2=') = {'
k3=','
k4=',0,lc};\n'
prog=[0]*(l1) #Vector de ceros
X=[0]*l1 #Vector de ceros
Y=[0]*l1 #Vector de ceros
prog.insert(0, k0) #Funcion para ingresar valores
for i in l2: #Lazo para analizar los puntos
    a=lineas[i] #Toma valor independiente
    b=a.split(' ') #Divide donde exista un espacio
    if len(b)==4: #Si longitud = 4 entonces
        b=b[2:] #Tomar del tercer elemento en adelante
    if len(b)==5:    #Si longitud = 5 entonces
        b=b[2],b[4] #Tomar el tercer y quinto elemento
    c1=str(round(float(b[0]),6)) #c1 igual al primer valor de b
    c2=str(round(float(b[1]),6)) #c2 igual al segundo valor de b
    c2=c2.replace("\r\n",'') #Elimina \r\n
    c2=c2.replace("\n\r",'') #Elimina \n\r
    c2=c2.replace('\r','') #Elimina \r
    c2=c2.replace('\n','') #Elimina \n
    X[i]=float(c1) #Ingresa valor en matriz X
    Y[i]=float(c2) #Ingresa valor en matriz Y
    x=k1+str(i+1)+k2+c1+k3+c2+k4 #Generacion del nombre del punto
    prog[i+1]=x #ingresa nombre en vector prog
puntos='//Points quantity: '+str(l1)+'\n' #Indica cantidad de puntos
prog.insert(0,puntos ) #Inserta linea
title='//Code generated by software programmed in Python\n' #Titulo
prog.insert(0,title) #Inserta titulo
```

```
archi2.writelines( prog ) #Escribe lineas de prog en archi2
archi2.close #Cierra archi2
"""
 Software para generar el codigo de las superfices
"""
#Cantidad de superficies
L=4
#Superficie 1:
a_1=52 #desde
z_1=95 #hasta
#Superficie 2:
a_2=9
z_2=51
#Superficie 3:
a_3_a=95
z_3_a=101
a_3_b=1
z_3_b=9
#Superficie 4:
a_4=52
z_4=51
x_a=102
x_b=103
#Textos
k0='Include "puntos.geo";\n'
k1='BSpline('
k2=') = {'
k3='};\n'
k4='BSpline(4) = {'
k=','
archi=open('surface.geo','w') #Crea el archivo surface.geo
#Programacion:
prog=[0]*(L) #Crea vector de ceros
prog.insert(0, k0) #Inserta linea k0
#Surface 1
x1=str(range(a_1,z_1+1)) #Texto de vector de numeros desde a_1 a z_1
x1=x1.replace("[",'') #Elimina [
x1=x1.replace("]",'') #Elimina ]
L1=k1+str(1)+k2+x1+k3 #Generacion de texto de linea 1
prog[1]=L1 #Inserta linea 1
#Surface 2
x2=str(range(a_2,z_2+1))
x2=x2.replace("[",'')
x2=x2.replace("]",'')
L2=k1+str(2)+k2+x2+k3 #Generacion de texto de linea 2
prog[2]=L2 #Inserta linea 2
#Surface 3
x3a=range(a_3_a,z_3_a+1)
x3b=range(a_3_b,z_3_b+1)
x3=x3a,x3b
x3=str(x3)
x3=''.join(x3)
x3=x3.replace("[",'')
x3=x3.replace("]",'')
x3=x3.replace("(",'')
x3=x3.replace(")",'')
L3=k1+str(3)+k2+x3+k3
```

```
111 prog[3]=L3
112 #Surface 4
113 L4=k4+str(a_4)+k+str(x_b)+k+str(x_a)+k+str(z_4)+k3 #Generacion de texto
        de linea 4
114 prog[4]=L4 #Inserta linea 4
115 surfaces='//Surfaces quantity: '+str(L)+'\n' #Texto superfices
116 prog.insert(0,surfaces) #Insercion de linea
117 title='//Code generated by software programmed in Python\n'
118 prog.insert(0,title) #Insercion de linea
119 archi.writelines(prog) #Escribe lineas de prog en archi
120 archi.close #Cierra archi
121 """
122  Software para generar el codigo del mallado
123 """
124 #Valores
125 alpha=-10 #Degrees (positivo en sentido contrario a manecillas)
126 cg_porc=50 #Porcentaje respecto a la longitud del perfil
127 #Leading edge
128 #Spline ext
129 valsplineext=0.1 #veces hacia izq de longitud del blade para puntos
        exteriores para Spline
130 valsplineint=0.5 #veces hacia izq de longitud del blade para punto
        interior para Spline
131 #Spline int
132 porcubicsplineinterno=0.005 #veces hacia izq de long del blade para
        Spline interior leading edge
133 porcubicsplineinternocentral=0.1 #veces hacia izq de long de blade para
        Punto central Spline interior del leading edge
134 #Trailing edge
135 valsplineextte=0.01 #veces hacia der de long del perfil para puntos ext
        del spline
136 valsplineextteb=0.06 #veces hacia der de long del perfil para puntos ext
        sup del spline
137 valsplineintte=0.06 #veces hacia der de long de perfil para punto int del
         spline
138 porctrailedge=1.2 #porcentaje de incremento respecto de lineas que parten
         de trailing edge
139 porcanchosplinete=0.12 #porcentaje del ancho del perfil para ancho de
        spline del trailing edge
140 #Lineas internas
141 ubicp1=0.15
142 ubicp2=0.925
143 porca1=0.185 #Super izq
144 porca2=0.15 #Inf izq
145 porcb1=0.08 #Super der
146 porcb2=0.1 #Inf der
147 ###
148 #Cuadro delimitador
149 upstream_veces_x=2 #Veces de la longitud para pared de upstream
150 downstream_veces_x=5 #Veces de la longitud para pared de downstream
151 veces_ancho=1.6 #Veces de la longitud para ancho del cuadro
152 #Parametros para mallado
153   #leading edge #bump
154 nl=50
155 rl=1
156   #vertical lines #prog
157 nv=35
```

```
158 rv=1.035
159   #body #bump
160 nb=120
161 rb=0.05
162   #back #prog
163 nba=110
164 rba=1.025
165   #front #prog
166 nf=50
167 rf=1.065
168   #Small trailing edge #bump
169 nsT=30
170 rsT=0.05
171       #Small circumference #progression
172 nc=80
173 rc=1.025
174 #Extrusion
175 veces_extrusion=0.016
176 lay=2#layers quantity
177 #Otros parametros
178 cantidaddevolumenes=11 #Valor obtenido por visualizacion
179 #####
180 #Valores maximos y minimos
181 xmax=max(X)
182 xmin=min(X)
183 ymax=max(Y)
184 ymin=min(Y)
185 #Tamanio del hydrofoil
186 longitud=xmax-xmin
187 ancho=ymax-ymin
188 #Centro de giro
189 x_cg=longitud*cg_porc/100 #Ubicacion de centro de giro en x
190 ya_cg=interp(x_cg,list(reversed(X[a_1-1:z_1])),list(reversed(Y[a_1-1:z_1
        ])))
191 yb_cg=interp(x_cg,X[a_2-1:z_2],Y[a_2-1:z_2])
192 y_cg=(ya_cg+yb_cg)/2
193 cg=x_cg #Centro de giro del hydrofoil en x
194 dy=y_cg      #centro de giro del hydrofoil en y
195 dy=round(dy,8)
196 #Calculos para cuadro delimitador
197 xpos=round((-(xmax+xmin)/2)+longitud*downstream_veces_x,2) #Valor en x
        positivo del cuadro delimitador
198 xneg=round(((xmax+xmin)/2)-longitud*upstream_veces_x,2) #Valor en x
        negativo del cuadro delimitador
199 ypos=round(dy+veces_ancho*longitud/2,4) #Valor en y positivo del cuadro
        delimitador
200 yneg=round(dy-veces_ancho*longitud/2,4) #Valor en y negativo del cuadro
        delimitador
201 #Calculos para extrusion
202 ev=longitud*veces_extrusion #Extrusion value
203 #Analisis para ubicar puntos de leading edge y trailing edge
204 Xrange=arange(len(X))
205 l=0
206 for v in Xrange:
207       l=l+1
208       if X[v]==xmax:
209             Xmaxub=l
```

```
210             #if l< len(X)/2:
211             y_xmax_a=Y[Xmaxub-1]
212             #if l>len(X)/2:
213             y_xmax_b=Y[Xmaxub-2]
214         if X[v]==xmin:
215             Xminub=l
216             y_xmin=Y[Xminub-1]
217 #Puntos antes de rotacion
218 #leading edge
219 xle_ar=xmin
220 yle_ar=y_xmin
221 #Trailing edge
222 xte_ar=xmax
223 yte_ar_a=y_xmax_a
224 yte_ar_b=y_xmax_b
225 #Funcion para rotacion
226 def rot(ax,ay,dx,dy,alpha):
227     ax_p=ax-dx
228     ay_p=ay-dy
229     beta=math.degrees(math.atan(abs(ay_p/ax_p)))
230     if (ax>dx and ay>dy):
231         #I cuad
232         theta=beta
233     if (ax<dx and ay>dy):
234         #II cuad
235         theta=180-beta
236     if (ax<dx and ay<dy):
237         #III cuad
238         theta=180+beta
239     if (ax>dx and ay<dy):
240         #IV cuad
241         theta=360-beta
242     if ax==dx:
243         if ay>dy:
244             theta=90
245         if ay>dy:
246             theta=270
247     if ay==dy:
248         if ax>dx:
249             theta=0
250         if ax<dx:
251             theta=180
252     gamma=theta+alpha
253     gamma_r=math.radians(gamma)
254     a=pow(pow(ax_p,2)+pow(ay_p,2),0.5)
255     x_p=a*(math.cos(gamma_r))
256     y_p=a*(math.sin(gamma_r))
257     xb=x_p+dx
258     yb=y_p+dy
259     return(xb,yb)
260 #Puntos luego de rotacion
261 le=rot(xle_ar,yle_ar,cg,dy,alpha)
262 xle=round(le[0],5)  #posicion del leading edge en x luego de rotacion
263 yle=round(le[1],5)  #posicion del leading edge en y luego de rotacion
264 te_a=rot(xte_ar,yte_ar_a,cg,dy,alpha) #posicion del te a luego de
        rotacion
265 te_b=rot(xte_ar,yte_ar_b,cg,dy,alpha) #posicion del te b luego de rot
```

```
xte_a=round(te_a[0]*porctrailedge,6) #posicion del trailing edge mas porcentaje
xte_b=round(te_b[0]*porctrailedge,6)
ymina=round(te_a[1],8) #posicion y del trailing edge a
yminb=round(te_b[1],8) #posicion y del trailing edge b
#Puntos para Spline exterior de leading edge
l1_x=round(xle-valsplineext*longitud,6) #Puntos exteriores
l2_x=round(xle-valsplineint*longitud,6) #Punto interior
#Puntos para Spline interior de leading edge
le_int_a=rot(X[z_1-1],Y[z_1-1],cg,dy,alpha) #Punto donde inicia linea superior
le_int_b=rot(X[a_2-1],Y[a_2-1],cg,dy,alpha) #Punto donde inicia linea inferior
l1_x_le_int=round(xle-porcubicsplineinterno*longitud,6) #Puntos exteriores en X
l2_x_le_int=round(xle-porcubicsplineinternocentral*longitud,6) #Puntos interior en X
y1_le_int=ypos+(((l1_x_le_int-l1_x)*(ypos-le_int_a[1]))/(l1_x-le_int_a[0]))
y2_le_int=yneg+(((l1_x_le_int-l1_x)*(yneg-le_int_b[1]))/(l1_x-le_int_b[0]))
y_le_int=(y1_le_int+y2_le_int)/2
y1_le_int=round(y1_le_int,8)
y2_le_int=round(y2_le_int,8)
y_le_int=round(y_le_int,8)
#Puntos para Spline de trailing edge
l1_x_te=round(((te_a[0]+te_b[0])/2)+valsplineextte*longitud,6) #Punto exterior inf
l1_x_te2=round(((te_a[0]+te_b[0])/2)+valsplineextteb*longitud,6) #Punto exterior sup
l2_x_te=round(((te_a[0]+te_b[0])/2)+valsplineintte*longitud,6) #Punto interior
y1_te=((ymina+yminb)/2)+(porcanchosplinete*longitud)/2 #Punto superior spline te
y2_te=((ymina+yminb)/2)-(porcanchosplinete*longitud)/2 #Punto inf spline te
y_te=((ymina+yminb)/2)
#Puntos para lineas interiores
long_int=l1_x_te-l1_x_le_int
long_int_1=long_int*ubicp1
long_int_2=long_int*ubicp2
    #Punto1
p1x=l1_x_le_int+long_int_1
        #sup
        #Punto2
p2x=l1_x_le_int+long_int_2
y0s=(y1_le_int+y1_te)/2
y0i=(y2_le_int+y2_te)/2
y1a=round(porca1*longitud+y0s,6)
y1b=round(-porca2*longitud+y0i,6)
y2a=round(porcb1*longitud+y0s,6)
y2b=round(-porcb2*longitud+y0i,6)
         #sup
    #Cantidad de puntos
tamval=l1
#Cantidad de superficies
tamvals=L
```

```
#Cantidad de puntos a agregar
f=tamval
#Cantidad de lineas a agregar
s=2*tamval
#Textos por defecto
title='//Code generated by software programmed in Python\n'
k0='Include "surface.geo";\n'
k1='alpha = '
k2='*Pi/180;\n'
k3='ls[] = Rotate {{0, 0, 1}, {'
k4=','
k4a=', 0}, alpha} {Line{1,2,3,4};};\n'
k5='//Bounding box\n'
k6='Point('
k7=') = {'+str(xneg)+', '+str(ypos)+',0,lc};\n'
k8=') = {'+str(xneg)+', '+str(yneg)+',0,lc};\n'
k9=') = {'+str(xpos)+', '+str(ypos)+',0,lc};\n'
k10=') = {'+str(xpos)+', '+str(yneg)+',0,lc};\n'
k11='//Trailing edge\n'
k12=') = {'+str(xpos)+', '+str(y1_te)+',0,lc};\n'
k13=') = {'+str(xpos)+', '+str(y2_te)+',0,lc};\n'
k14='Line('
k15=') = {'
k16=','
k17='};\n'
k18='//Front\n'
k19=',0,lc};\n'
k20='BSpline('
k21='Spline('
k22='//Body and back\n'
#Spline te
k23=') = {'+str(l1_x_te2)+', '+str(y1_te)+',0,lc};\n'
k24=') = {'+str(l1_x_te)+', '+str(y2_te)+',0,lc};\n'
k25=') = {'+str(l2_x_te)+', '+str(y_te)+',0,lc};\n'
#Spline le
k26=') = {'+str(l1_x_le_int)+', '+str(y1_le_int)+',0,lc};\n'
k27=') = {'+str(l1_x_le_int)+', '+str(y2_le_int)+',0,lc};\n'
k28=') = {'+str(l2_x_le_int)+', '+str(y_le_int)+',0,lc};\n'
################################
#Creacion del codigo
archi3=open('mesh.geo','w')
prog=['']
#Title, include, alpha y ls
prog.insert(0, title)
prog[len(prog)-1]=k0
A1=k1+str(alpha)+k2
A2=k3+str(cg)+k4+str(dy)+k4a
A3=k5
prog.insert(len(prog),A1)
prog.insert(len(prog),A2)
prog.insert(len(prog),A3)
#Bounding box
B=range(tamval+1,tamval+1+f) #Puntos
D=range(tamvals+1,tamvals+1+s) #Superficies
C1=k6+str(B[0])+k7
C2=k6+str(B[1])+k8
C3=k6+str(B[2])+k9
```

```
368 C4=k6+str(B[3])+k10
369 prog.insert(len(prog),C1)
370 prog.insert(len(prog),C2)
371 prog.insert(len(prog),C3)
372 prog.insert(len(prog),C4)
373 #Trailing edge
374 C5=k11
375 prog.insert(len(prog),C5)
376 #Spline
377 C8=k6+str(B[11])+k23
378 C9=k6+str(B[12])+k24
379 C10=k6+str(B[13])+k25
380 prog.insert(len(prog),C8)
381 prog.insert(len(prog),C9)
382 prog.insert(len(prog),C10)
383 D6=k14+str(D[31])+k15+str(a_1)+k16+str(B[11])+k17
384 D7=k14+str(D[32])+k15+str(z_2)+k16+str(B[12])+k17
385 prog.insert(len(prog),D6)
386 prog.insert(len(prog),D7)
387 D8=k20+str(D[33])+k15+str(B[11])+k16+str(B[13])+k16+str(B[12])+k17
388 prog.insert(len(prog),D8)
389 #
390 C6=k6+str(B[4])+k12
391 C7=k6+str(B[5])+k13
392 prog.insert(len(prog),C6)
393 prog.insert(len(prog),C7)
394 D1=k14+str(D[0])+k15+str(B[11])+k16+str(B[4])+k17
395 D2=k14+str(D[1])+k15+str(B[12])+k16+str(B[5])+k17
396 D3=k14+str(D[2])+k15+str(B[4])+k16+str(B[5])+k17
397 D4=k14+str(D[3])+k15+str(B[4])+k16+str(B[2])+k17
398 D5=k14+str(D[4])+k15+str(B[5])+k16+str(B[3])+k17
399 prog.insert(len(prog),D1)
400 prog.insert(len(prog),D2)
401 prog.insert(len(prog),D3)
402 prog.insert(len(prog),D4)
403 prog.insert(len(prog),D5)
404 #Front
405  #Spline peq
406 E1=k18
407 prog.insert(len(prog),E1)
408
409 E11=k6+str(B[14])+k26
410 E12=k6+str(B[15])+k27
411 E13=k6+str(B[16])+k28
412 prog.insert(len(prog),E11)
413 prog.insert(len(prog),E12)
414 prog.insert(len(prog),E13)
415 E14=k14+str(D[34])+k15+str(a_3_a)+k16+str(B[14])+k17
416 E15=k14+str(D[35])+k15+str(z_3_b)+k16+str(B[15])+k17
417 prog.insert(len(prog),E14)
418 prog.insert(len(prog),E15)
419 E16=k20+str(D[36])+k15+str(B[14])+k16+str(B[16])+k16+str(B[15])+k17
420 prog.insert(len(prog),E16)
421
422 #
423 E2=k6+str(B[6])+k15+str(l1_x)+k16+str(ypos)+k19
424 E3=k6+str(B[7])+k15+str(l1_x)+k16+str(yneg)+k19
```

```
425 E4=k6+str(B[8])+k15+str(l2_x)+k16+str(yle)+k19
426 prog.insert(len(prog),E2)
427 prog.insert(len(prog),E3)
428 prog.insert(len(prog),E4)
429 E5=k14+str(D[5])+k15+str(B[14])+k16+str(B[6])+k17
430 E6=k14+str(D[6])+k15+str(B[15])+k16+str(B[7])+k17
431 prog.insert(len(prog),E5)
432 prog.insert(len(prog),E6)
433 E7=k20+str(D[7])+k15+str(B[6])+k16+str(B[8])+k16+str(B[7])+k17
434 E8=k21+str(D[8])+k15+str(B[6])+k16+str(B[0])+k17
435 E9=k21+str(D[9])+k15+str(B[7])+k16+str(B[1])+k17
436 E10=k14+str(D[10])+k15+str(B[0])+k16+str(B[1])+k17
437 prog.insert(len(prog),E7)
438 prog.insert(len(prog),E8)
439 prog.insert(len(prog),E9)
440 prog.insert(len(prog),E10)
441
442 #Body and back
443 F1=k22
444 prog.insert(len(prog),F1)
445 F2=k6+str(B[9])+k15+str(xte_a)+k16+str(ypos)+k19
446 F3=k6+str(B[10])+k15+str(xte_a)+k16+str(yneg)+k19
447 prog.insert(len(prog),F2)
448 prog.insert(len(prog),F3)
449 F4=k14+str(D[11])+k15+str(B[6])+k16+str(B[9])+k17
450 F5=k14+str(D[12])+k15+str(B[7])+k16+str(B[10])+k17
451 prog.insert(len(prog),F4)
452 prog.insert(len(prog),F5)
453 F6=k14+str(D[13])+k15+str(B[11])+k16+str(B[9])+k17
454 F7=k14+str(D[14])+k15+str(B[12])+k16+str(B[10])+k17
455 F8=k14+str(D[15])+k15+str(B[9])+k16+str(B[2])+k17
456 F9=k14+str(D[16])+k15+str(B[10])+k16+str(B[3])+k17
457 prog.insert(len(prog),F6)
458 prog.insert(len(prog),F7)
459 prog.insert(len(prog),F8)
460 prog.insert(len(prog),F9)
461 #Lineas splines peq
462 F12=k6+str(B[17])+k15+str(p1x)+k16+str(y1a)+k19
463 F13=k6+str(B[18])+k15+str(p1x)+k16+str(y1b)+k19
464 F14=k6+str(B[19])+k15+str(p2x)+k16+str(y2a)+k19
465 F15=k6+str(B[20])+k15+str(p2x)+k16+str(y2b)+k19
466 prog.insert(len(prog),F12)
467 prog.insert(len(prog),F13)
468 prog.insert(len(prog),F14)
469 prog.insert(len(prog),F15)
470 F10=k20+str(D[37])+k15+str(B[14])+k16+str(B[17])+k16+str(B[19])+k16+str(B
      [11])+k17
471 F11=k20+str(D[38])+k15+str(B[15])+k16+str(B[18])+k16+str(B[20])+k16+str(B
      [12])+k17
472 prog.insert(len(prog),F10)
473 prog.insert(len(prog),F11)
474 #####
475 #####Mallado
476 j1='//Meshing\n'
477 j2='//Leading edge\n'
478 j3='//Vertical lines\n'
479 j4='//Body\n'
```

```
480 j5='//Back\n'
481 j6='//Front\n'
482 j7='//Small trailing edge\n'
483 j8='nl'
484 j9='rl'
485 j10='nv'
486 j11='rv'
487 j12='nb'
488 j13='rb'
489 j14='nba'
490 j15='rba'
491 j16='nf'
492 j17='rf'
493 j18='nsT'
494 j19='rsT'
495 j20=';\n'
496 j21='Transfinite Line{'
497 j22='} = '
498 j23=' Using Bump '
499 j24=' = '
500 j25=' Using Progression '
501 j26='//Small circumference\n'
502 j27='nc'
503 j28='rc'
504 prog.insert(len(prog),j1)
505 #leading edge
506 G1=j2
507 G2=j8+j24+str(nl)+j20
508 G3=j9+j24+str(rl)+j20
509 G4=j21+str(3)+j22+j8+j23+j9+j20
510 G5=j21+str(D[7])+j22+j8+j23+j9+j20
511 G6=j21+str(D[10])+j22+j8+j23+j9+j20
512 G7=j21+str(D[36])+j22+j8+j23+j9+j20
513 prog.insert(len(prog),G1)
514 prog.insert(len(prog),G2)
515 prog.insert(len(prog),G3)
516 prog.insert(len(prog),G4)
517 prog.insert(len(prog),G5)
518 prog.insert(len(prog),G6)
519 prog.insert(len(prog),G7)
520 #vertical lines
521 H1=j3
522 H2=j10+j24+str(nv)+j20
523 H3=j11+j24+str(rv)+j20
524 H4=j21+str(D[5])+j22+j10+j25+j11+j20
525 H5=j21+str(D[6])+j22+j10+j25+j11+j20
526 H6=j21+str(D[13])+j22+j10+j25+j11+j20
527 H7=j21+str(D[14])+j22+j10+j25+j11+j20
528 H8=j21+str(D[3])+j22+j10+j25+j11+j20
529 H9=j21+str(D[4])+j22+j10+j25+j11+j20
530 prog.insert(len(prog),H1)
531 prog.insert(len(prog),H2)
532 prog.insert(len(prog),H3)
533 prog.insert(len(prog),H4)
534 prog.insert(len(prog),H5)
535 prog.insert(len(prog),H6)
536 prog.insert(len(prog),H7)
```

```
prog.insert(len(prog),H8)
prog.insert(len(prog),H9)
#Body
J1=j4
J2=j12+j24+str(nb)+j20
J3=j13+j24+str(rb)+j20
J4=j21+str(D[11])+j22+j12+j23+j13+j20
J5=j21+str(1)+j22+j12+j23+j13+j20
J6=j21+str(2)+j22+j12+j23+j13+j20
J7=j21+str(D[12])+j22+j12+j23+j13+j20
J8=j21+str(D[37])+j22+j12+j23+j13+j20
J9=j21+str(D[38])+j22+j12+j23+j13+j20
prog.insert(len(prog),J1)
prog.insert(len(prog),J2)
prog.insert(len(prog),J3)
prog.insert(len(prog),J4)
prog.insert(len(prog),J5)
prog.insert(len(prog),J6)
prog.insert(len(prog),J7)
prog.insert(len(prog),J8)
prog.insert(len(prog),J9)
#Back
K1=j5
K2=j14+j24+str(nba)+j20
K3=j15+j24+str(rba)+j20
K4=j21+str(D[15])+j22+j14+j25+j15+j20
K5=j21+str(D[0])+j22+j14+j25+j15+j20
K6=j21+str(D[1])+j22+j14+j25+j15+j20
K7=j21+str(D[16])+j22+j14+j25+j15+j20
prog.insert(len(prog),K1)
prog.insert(len(prog),K2)
prog.insert(len(prog),K3)
prog.insert(len(prog),K4)
prog.insert(len(prog),K5)
prog.insert(len(prog),K6)
prog.insert(len(prog),K7)
#Front
M1=j6
M2=j16+j24+str(nf)+j20
M3=j17+j24+str(rf)+j20
M4=j21+str(D[8])+j22+j16+j25+j17+j20
M5=j21+str(D[9])+j22+j16+j25+j17+j20
prog.insert(len(prog),M1)
prog.insert(len(prog),M2)
prog.insert(len(prog),M3)
prog.insert(len(prog),M4)
prog.insert(len(prog),M5)
#Small traling edge
N1=j7
N2=j18+j24+str(nsT)+j20
N3=j19+j24+str(rsT)+j20
N4=j21+str(4)+j22+j18+j23+j19+j20
N5=j21+str(D[2])+j22+j18+j23+j19+j20
N5a=j21+str(D[33])+j22+j18+j23+j19+j20
prog.insert(len(prog),N1)
prog.insert(len(prog),N2)
prog.insert(len(prog),N3)
```

```
prog.insert(len(prog),N4)
prog.insert(len(prog),N5)
prog.insert(len(prog),N5a)
#New small circunference
N6=j26
N7=j27+j24+str(nc)+j20
N8=j28+j24+str(rc)+j20
N9=j21+str(D[31])+j22+j27+j25+j28+j20
N10=j21+str(D[32])+j22+j27+j25+j28+j20
N11=j21+str(D[34])+j22+j27+j25+j28+j20
N12=j21+str(D[35])+j22+j27+j25+j28+j20
prog.insert(len(prog),N6)
prog.insert(len(prog),N7)
prog.insert(len(prog),N8)
prog.insert(len(prog),N9)
prog.insert(len(prog),N10)
prog.insert(len(prog),N11)
prog.insert(len(prog),N12)

#Loops
k0='//Loops\n'
k1='Line Loop('
k2=') = {'
k3=', '
k4='};\n'
k5='Plane Surface('
k6='//Surfaces\n'
k7='Transfinite Surface {'
k8='//Transfinite Surfaces\n'
k9='Recombine Surface {'
k10='//Recombine Surfaces\n'
P1=k0
prog.insert(len(prog),P1)
P2=k1+str(D[17])+k2+str(D[36])+k3+str(D[6])+k3+str(-D[7])+k3+str(-D[5])+
    k4
P3=k1+str(D[18])+k2+str(D[8])+k3+str(D[10])+k3+str(-D[9])+k3+str(-D[7])+
    k4
P4=k1+str(D[19])+k2+str(D[11])+k3+str(-D[13])+k3+str(-D[37])+k3+str(D[5])
    +k4
P5=k1+str(D[20])+k2+str(D[38])+k3+str(D[14])+k3+str(-D[12])+k3+str(-D[6])
    +k4
P6=k1+str(D[21])+k2+str(D[13])+k3+str(D[15])+k3+str(-D[3])+k3+str(-D[0])+
    k4
P7=k1+str(D[22])+k2+str(D[14])+k3+str(D[16])+k3+str(-D[4])+k3+str(-D[1])+
    k4
P8=k1+str(D[23])+k2+str(D[0])+k3+str(D[2])+k3+str(-D[1])+k3+str(-D[33])+
    k4
P8a=k1+str(D[39])+k2+str(1)+k3+str(D[34])+k3+str(D[37])+k3+str(-D[31])+k4
P8b=k1+str(D[40])+k2+str(D[31])+k3+str(D[33])+k3+str(-D[32])+k3+str(-4)+
    k4
P8c=k1+str(D[41])+k2+str(D[32])+k3+str(-D[38])+k3+str(-D[35])+k3+str(2)+
    k4
P8d=k1+str(D[42])+k2+str(D[35])+k3+str(-D[36])+k3+str(-D[34])+k3+str(3)+
    k4
prog.insert(len(prog),P2)
prog.insert(len(prog),P3)
prog.insert(len(prog),P4)
```

```
641 prog.insert(len(prog),P5)
642 prog.insert(len(prog),P6)
643 prog.insert(len(prog),P7)
644 prog.insert(len(prog),P8)
645 prog.insert(len(prog),P8a)
646 prog.insert(len(prog),P8b)
647 prog.insert(len(prog),P8c)
648 prog.insert(len(prog),P8d)
649 #Plane surface
650 P9=k6
651 P10=k5+str(D[24])+k2+str(D[17])+k4
652 P11=k5+str(D[25])+k2+str(D[18])+k4
653 P12=k5+str(D[26])+k2+str(D[19])+k4
654 P13=k5+str(D[27])+k2+str(D[20])+k4
655 P14=k5+str(D[28])+k2+str(D[21])+k4
656 P15=k5+str(D[29])+k2+str(D[22])+k4
657 P16=k5+str(D[30])+k2+str(D[23])+k4
658 P16a=k5+str(D[43])+k2+str(D[39])+k4
659 P16b=k5+str(D[44])+k2+str(D[40])+k4
660 P16c=k5+str(D[45])+k2+str(D[41])+k4
661 P16d=k5+str(D[46])+k2+str(D[42])+k4
662 prog.insert(len(prog),P9)
663 prog.insert(len(prog),P10)
664 prog.insert(len(prog),P11)
665 prog.insert(len(prog),P12)
666 prog.insert(len(prog),P13)
667 prog.insert(len(prog),P14)
668 prog.insert(len(prog),P15)
669 prog.insert(len(prog),P16)
670 prog.insert(len(prog),P16a)
671 prog.insert(len(prog),P16b)
672 prog.insert(len(prog),P16c)
673 prog.insert(len(prog),P16d)
674 #Transfinite surfaces
675 P17=k8
676 P18=k7+str(D[24])+k4
677 P19=k7+str(D[25])+k4
678 P20=k7+str(D[26])+k4
679 P21=k7+str(D[27])+k4
680 P22=k7+str(D[28])+k4
681 P23=k7+str(D[29])+k4
682 P24=k7+str(D[30])+k4
683 P24a=k7+str(D[43])+k4
684 P24b=k7+str(D[44])+k4
685 P24c=k7+str(D[45])+k4
686 P24d=k7+str(D[46])+k4
687 prog.insert(len(prog),P17)
688 prog.insert(len(prog),P18)
689 prog.insert(len(prog),P19)
690 prog.insert(len(prog),P20)
691 prog.insert(len(prog),P21)
692 prog.insert(len(prog),P22)
693 prog.insert(len(prog),P23)
694 prog.insert(len(prog),P24)
695 prog.insert(len(prog),P24a)
696 prog.insert(len(prog),P24b)
697 prog.insert(len(prog),P24c)
```

```
prog.insert(len(prog),P24d)
#Recombine surfaces
P25=k10
P26=k9+str(D[24])+k4
P27=k9+str(D[25])+k4
P28=k9+str(D[26])+k4
P29=k9+str(D[27])+k4
P30=k9+str(D[28])+k4
P31=k9+str(D[29])+k4
P32=k9+str(D[30])+k4
P32a=k9+str(D[43])+k4
P32b=k9+str(D[44])+k4
P32c=k9+str(D[45])+k4
P32d=k9+str(D[46])+k4
prog.insert(len(prog),P25)
prog.insert(len(prog),P26)
prog.insert(len(prog),P27)
prog.insert(len(prog),P28)
prog.insert(len(prog),P29)
prog.insert(len(prog),P30)
prog.insert(len(prog),P31)
prog.insert(len(prog),P32)
prog.insert(len(prog),P32a)
prog.insert(len(prog),P32b)
prog.insert(len(prog),P32c)
prog.insert(len(prog),P32d)
#Extrude
m1='//Extrusion\n'
m2='Extrude {0, 0, '
m3='}{ \n'
m4='    Surface{'
m5=','
m6='};\n'
m7='    Layers{'
m8='    Recombine;\n'
m9='  }\n'
Q0=m1
Q1=m2+str(ev)+m3
Q2=m4+str(D[24])+m5+str(D[25])+m5+str(D[26])+m5+str(D[27])+m5+str(D[28])+
    m5+str(D[29])+m5+str(D[30])+m5+str(D[43])+m5+str(D[44])+m5+str(D[45])+
    m5+str(D[46])+m6
Q3=m7+str(lay)+m6
Q4=m8
Q5=m9
prog.insert(len(prog),Q0)
prog.insert(len(prog),Q1)
prog.insert(len(prog),Q2)
prog.insert(len(prog),Q3)
prog.insert(len(prog),Q4)
prog.insert(len(prog),Q5)
#Names
n0='//Names\n'
n1='"internal"'
n2='"INL"'
n3='"OUTL"'
n4='"TOPBOT"'
n5='"WING"'
```

```
753 n6='"FRONT"'
754 n7='"BACK"'
755 n8='Physical Volume('
756 n9='Physical Surface('
757 n10=') = {'
758 n11=','
759 n12='};\n'
760 #Calculos
761 vol=range(1,cantidaddevolumenes+1)
762 vol=str(vol)
763 vol=vol.replace("[",'') #Elimina [
764 vol=vol.replace("]",'') #Elimina ]
765 front=range(D[24],D[30]+1)
766 frontb=range(D[43],D[46]+1)
767 front=front+frontb
768 front=str(front)
769 front=front.replace("[",'') #Elimina [
770 front=front.replace("]",'') #Elimina ]
771 #Valores se deben colocar visualizando en Gmsh
772 #Front-back 29 73, 30 95, 31 117, 32 139, 33 161, 34 183, 35 205, 48
    227,49 249, 50 271, 51 293
773 R1=n0
774 R2=n8+n1+n10+vol+n12
775 R3=n9+n2+n10+str(86)+n12
776 R4=n9+n3+n10+'156,196,178'+n12
777 R5=n9+n4+n10+'82,104,152,90,134,174'+n12
778 R6=n9+n5+n10+'292,214,248,270'+n12
779 R7=n9+n6+n10+str(front)+n12
780 R8=n9+n7+n10+'73,95,117,139,161,183,205,227,249,271,293'+n12
781 prog.insert(len(prog),R1)
782 prog.insert(len(prog),R2)
783 prog.insert(len(prog),R3)
784 prog.insert(len(prog),R4)
785 prog.insert(len(prog),R5)
786 prog.insert(len(prog),R6)
787 prog.insert(len(prog),R7)
788 prog.insert(len(prog),R8)
789 archi3.writelines(prog)
790 archi3.close
```

B.2.2 BLADE WITH HIGH ROTATION ANGLE

```
1  """
2   Software para transformar los puntos
3   Formato de puntos de entrada:
4   Opcion A: Con dos espacios entre puntos (uno para negativos)
5             y dos espacios al inicio
6         |  x       y
7         |  0.001   0.012
8         |  0.001  -0.012
9   Opcion B: Sin espacios
10        |  x       y
11        |0.001  0.012
12        |-0.001  -0.012
13  """
14 #Longitud caracteristica, lc:
15 lc=1e-6
```

```
# Programacion:
archi1=open('puntos-middle.dat','r') #Leer archivo puntos.dat
archi2=open('puntos.geo','w') #Crear el archivo puntos.geo
lineas=archi1.readlines() #Lee las lineas del archi1
archi1.close #Cierra el archi1
l1=len(lineas) #Longitud de vector lineas
l2=arange(l1) #Crea vector desde 1 a l1 con pasos de 1
k0='lc = '+str(lc)+';\n' #Longitud caracteristica
#Textos a incluir
k1='Point('
k2=') = {'
k3=','
k4=',0,lc};\n'
prog=[0]*(l1) #Vector de ceros
X=[0]*l1 #Vector de ceros
Y=[0]*l1 #Vector de ceros
prog.insert(0, k0) #Funcion para ingresar valores
for i in l2: #Lazo para analizar los puntos
    a=lineas[i] #Toma valor independiente
    b=a.split(' ') #Divide donde exista un espacio
    if len(b)==4: #Si longitud = 4 entonces
        b=b[2:] #Tomar del tercer elemento en adelante
    if len(b)==5:    #Si longitud = 5 entonces
        b=b[2],b[4] #Tomar el tercer y quinto elemento
    c1=str(round(float(b[0]),6)) #c1 igual al primer valor de b
    c2=str(round(float(b[1]),6)) #c2 igual al segundo valor de b
    c2=c2.replace("\r\n",'') #Elimina \r\n
    c2=c2.replace("\n\r",'') #Elimina \n\r
    c2=c2.replace('\r','') #Elimina \r
    c2=c2.replace('\n','') #Elimina \n
    X[i]=float(c1) #Ingresa valor en matriz X
    Y[i]=float(c2) #Ingresa valor en matriz Y
    x=k1+str(i+1)+k2+c1+k3+c2+k4 #Generacion del nombre del punto
    prog[i+1]=x #ingresa nombre en vector prog
puntos='//Points quantity: '+str(l1)+'\n' #Indica cantidad de puntos
prog.insert(0,puntos ) #Inserta linea
title='//Code generated by software programmed in Python\n' #Titulo
prog.insert(0,title) #Inserta titulo
archi2.writelines( prog ) #Escribe lineas de prog en archi2
archi2.close #Cierra archi2
"""
 Software para generar el codigo de las superfices
"""
#Cantidad de superficies
L=4
#Superficie 1:
a_1=52 #desde
z_1=95 #hasta
#Superficie 2:
a_2=9
z_2=51
#Superficie 3:
a_3_a=95
z_3_a=101
a_3_b=1
z_3_b=9
#Superficie 4:
```

```
a_4=52
z_4=51
x_a=102
x_b=103
x_ab=104
#Textos
k0='Include "puntos.geo";\n'
k1='BSpline('
k2=') = {'
k3='};\n'
k4='BSpline(4) = {'
k=','
archi=open('surface.geo','w') #Crea el archivo surface.geo
#Programacion:
prog=[0]*(L) #Crea vector de ceros
prog.insert(0, k0) #Inserta linea k0
#Surface 1
x1=str(range(a_1,z_1+1)) #Texto de vector de numeros desde a_1 a z_1
x1=x1.replace("[",'') #Elimina [
x1=x1.replace("]",'') #Elimina ]
L1=k1+str(1)+k2+x1+k3 #Generacion de texto de linea 1
prog[1]=L1 #Inserta linea 1
#Surface 2
x2=str(range(a_2,z_2+1))
x2=x2.replace("[",'')
x2=x2.replace("]",'')
L2=k1+str(2)+k2+x2+k3 #Generacion de texto de linea 2
prog[2]=L2 #Inserta linea 2
#Surface 3
x3a=range(a_3_a,z_3_a+1)
x3b=range(a_3_b,z_3_b+1)
x3=x3a,x3b
x3=str(x3)
x3=''.join(x3)
x3=x3.replace("[",'')
x3=x3.replace("]",'')
x3=x3.replace("(",'')
x3=x3.replace(")",'')
L3=k1+str(3)+k2+x3+k3
prog[3]=L3
#Surface 4
L4=k4+str(a_4)+k+str(x_b)+k+str(x_ab)+k+str(x_a)+k+str(z_4)+k3 #
    Generacion de texto de linea 4
prog[4]=L4 #Inserta linea 4
surfaces='//Surfaces quantity: '+str(L)+'\n' #Texto superfices
prog.insert(0,surfaces) #Insercion de linea
title='//Code generated by software programmed in Python\n'
prog.insert(0,title) #Insercion de linea
archi.writelines(prog) #Escribe lineas de prog en archi
archi.close #Cierra archi
"""
 Software para generar el codigo del mallado
"""
#Valores
alpha=-80.2#Degrees (positivo en sentido contrario a manecillas)
cg_porc=50 #Porcentaje respecto a la longitud del perfil
#Leading edge
```

```
#Spline ext
valsplineext_a=0.2 #veces hacia izq de longitud del blade para puntos exteriores para Spline
valsplineext_b=0.1 #veces hacia izq de longitud del blade para puntos exteriores para Spline
valsplineint=0.5 #veces hacia izq de longitud del blade para punto interior para Spline
#Spline int
porcubicsplineinterno=0.05 #veces ancho de long del blade para Spline interior leading edge
porcubicsplineinternoalt=0.03 #veces hacia arriba de long del blade
porcubicsplineinternocentral=0.00 #veces hacia izq de long de blade para Punto central Spline interior del leading edge
porcubicsplineinternocentralalt=0.135 #veces hacia arriba de long de blade
#Trailing edge
valsplineextte=0.05 #veces ancho de long del perfil para puntos ext del spline
valsplineexttealt=0.03 #veces abajo long perfil para puntos ext del spline
valsplineintte=0.00 #veces hacia der de long de perfil para punto int del spline
valsplineinttealt=0.075 #veces abajo long perfil para puntos int del spline
#lineas exteriores
porctrailedge_a=1.15 #porcentaje de incremento respecto de lineas que parten de trailing edge
porctrailedge_b=1.4 #porcentaje de incremento respecto de lineas que parten de trailing edge
#Lineas internas
#ubicacion en x del ancho del blade
#p1a1 arriba der
ubicp1=0.6 #mayor va mas a la derecha
porca1=0.3 #mayor va mas alto
#p2b1 abajo der
ubicp2=1.3 #mayor va mas a la derecha
porcb1=-0.2 #menor va mas bajo
#p3a2 arriba izq
ubicp3=-0.1 #menor va mas a la izquierda
porca2=-0.2 #mayor va mas bajo
#p4b2 abajo izq
ubicp4=0.65 #menor va mas a la izquierda
porcb2=0.3 #mayor va mas bajo
###
#Cuadro delimitador
upstream_veces_x=2 #Veces de la longitud para pared de upstream
downstream_veces_x=7 #Veces de la longitud para pared de downstream
veces_ancho=1.6 #Veces de la longitud para ancho del cuadro
#Parametros para mallado
  #upperside #bump
nl=70
rl=0.15
  #vertical lines #prog #external
nv=35
rv=1.035
  #body #prog #leading y trailing divisions
nb=35
```

```
rb=0.1
  #back #prog #derecha
nba=255
rba=1.01
  #front #prog
nf=45
rf=1.03
  #Small trailing edge #bump
nsT=100
rsT=0.1
     #Small circumference #progression
nc=70
rc=1.05
#Extrusion
veces_extrusion=0.016
lay=2#10  #2 #layers quantity
#Otros parametros
cantidaddevolumenes=11 #Valor obtenido por visualizacion
#####
#Valores maximos y minimos
xmax=max(X)
xmin=min(X)
ymax=max(Y)
ymin=min(Y)
#Tamanio del hydrofoil
longitud=xmax-xmin
ancho=ymax-ymin
#Centro de giro
x_cg=longitud*cg_porc/100 #Ubicacion de centro de giro en x
ya_cg=interp(x_cg,list(reversed(X[a_1-1:z_1])),list(reversed(Y[a_1-1:z_1
    ])))
yb_cg=interp(x_cg,X[a_2-1:z_2],Y[a_2-1:z_2])
y_cg=(ya_cg+yb_cg)/2
cg=x_cg #Centro de giro del hydrofoil en x
dy=y_cg     #centro de giro del hydrofoil en y
dy=round(dy,8)
#Calculos para cuadro delimitador
xpos=round((-(xmax+xmin)/2)+longitud*downstream_veces_x,2) #Valor en x
    positivo del cuadro delimitador
xneg=round(((xmax+xmin)/2)-longitud*upstream_veces_x,2) #Valor en x
    negativo del cuadro delimitador
ypos=round(dy+veces_ancho*longitud/2,4) #Valor en y positivo del cuadro
    delimitador
yneg=round(dy-veces_ancho*longitud/2,4) #Valor en y negativo del cuadro
    delimitador
#Calculos para extrusion
ev=longitud*veces_extrusion #Extrusion value
#Analisis para ubicar puntos de leading edge y trailing edge
Xrange=arange(len(X))
l=0
for v in Xrange:
     l=l+1
     if X[v]==xmax:
          Xmaxub=l
          #if l< len(X)/2:
          y_xmax_a=Y[Xmaxub-1]
          #if l>len(X)/2:
```

```
226             y_xmax_b=Y[Xmaxub-2]
227         if X[v]==xmin:
228             Xminub=l
229             y_xmin=Y[Xminub-1]
230 #Puntos antes de rotacion
231 #leading edge
232 xle_ar=xmin
233 yle_ar=y_xmin
234 #Trailing edge
235 xte_ar=xmax
236 yte_ar_a=y_xmax_a
237 yte_ar_b=y_xmax_b
238 #Funcion para rotacion
239 def rot(ax,ay,dx,dy,alpha):
240     ax_p=ax-dx
241     ay_p=ay-dy
242     beta=math.degrees(math.atan(abs(ay_p/ax_p)))
243     if (ax>dx and ay>dy):
244         #I cuad
245         theta=beta
246     if (ax<dx and ay>dy):
247         #II cuad
248         theta=180-beta
249     if (ax<dx and ay<dy):
250         #III cuad
251         theta=180+beta
252     if (ax>dx and ay<dy):
253         #IV cuad
254         theta=360-beta
255     if ax==dx:
256         if ay>dy:
257             theta=90
258         if ay>dy:
259             theta=270
260     if ay==dy:
261         if ax>dx:
262             theta=0
263         if ax<dx:
264             theta=180
265     gamma=theta+alpha
266     gamma_r=math.radians(gamma)
267     a=pow(pow(ax_p,2)+pow(ay_p,2),0.5)
268     x_p=a*(math.cos(gamma_r))
269     y_p=a*(math.sin(gamma_r))
270     xb=x_p+dx
271     yb=y_p+dy
272     return(xb,yb)
273 #Puntos luego de rotacion
274 le=rot(xle_ar,yle_ar,cg,dy,alpha)
275 xle=round(le[0],5)  #posicion del leading edge en x luego de rotacion
276 yle=round(le[1],5)  #posicion del leading edge en y luego de rotacion
277 te_a=rot(xte_ar,yte_ar_a,cg,dy,alpha) #posicion del te a luego de
        rotacion
278 te_b=rot(xte_ar,yte_ar_b,cg,dy,alpha) #posicion del te b luego de rot
279 xte_a=round(te_a[0]*porctrailedge_a,6) #posicion del trailing edge mas
        porcentaje
280 xte_b=round(te_b[0]*porctrailedge_b,6)
```

```
ymina=round(te_a[1],8) #posicion y del trailing edge a
yminb=round(te_b[1],8) #posicion y del trailing edge b
ymin=(ymina+yminb)/2
#Puntos para Spline exterior de leading edge
l1_x_a=round(xle-valsplineext_a*longitud,6) #Puntos exteriores
l1_x_b=round(xle-valsplineext_b*longitud,6) #Puntos exteriores
l2_x=round(xle-valsplineint*longitud,6) #Punto interior
#Puntos para Spline interior de leading edge
le_int_a=rot(X[z_1-1],Y[z_1-1],cg,dy,alpha) #Punto donde inicia linea
    superior
le_int_b=rot(X[a_2-1],Y[a_2-1],cg,dy,alpha) #Punto donde inicia linea
    inferior
l1_x_le_int=round(xle-porcubicsplineinterno*longitud,6) #Puntos
    exteriores en X
l1_x_le_int_a=round(xle+porcubicsplineinterno*longitud,6) #Puntos
    exteriores en X
l1_x_le_int_b=round(xle-porcubicsplineinterno*longitud,6) #Puntos
    exteriores en X
l2_x_le_int=round(xle-porcubicsplineinternocentral*longitud,6) #Puntos
    interior en X
y1_le_int=yle+porcubicsplineinternoalt*longitud
y2_le_int=yle+porcubicsplineinternocentralalt*longitud
y_le_int=(y1_le_int+y2_le_int)/2
y1_le_int=round(y1_le_int,8)
y2_le_int=round(y2_le_int,8)
y_le_int=round(y_le_int,8)
#Puntos para Spline de trailing edge
l1_x_te=round(((te_a[0]+te_b[0])/2)+valsplineextte*longitud,6) #Puntos
    exteriores
l1_x_te_a=round(((te_a[0]+te_b[0])/2)+valsplineextte*longitud,6) #Puntos
    exteriores
l1_x_te_b=round(((te_a[0]+te_b[0])/2)-valsplineextte*longitud,6) #Puntos
    exteriores
l2_x_te=round(((te_a[0]+te_b[0])/2)+valsplineintte*longitud,6) #Punto
    interior
y1_te=ymin-valsplineexttealt*longitud
y2_te=ymin-valsplineinttealt*longitud
y_te=((ymina+yminb)/2)
#Puntos para lineas interiores
long_int=l1_x_te-l1_x_le_int
long_int_1=long_int*ubicp1
long_int_2=long_int*ubicp2
long_int_3=long_int*ubicp3
long_int_4=long_int*ubicp4
    #Punto1
p1x=l1_x_le_int+long_int_1
p3x=l1_x_le_int+long_int_3
        #sup
        #Punto2
p2x=l1_x_le_int+long_int_2
p4x=l1_x_le_int+long_int_4
y0s=(y1_le_int+y1_te)/2
y0i=(y2_le_int+y2_te)/2
y1a=round(porca1*longitud+y0s,6)
y1b=round(-porca2*longitud+y0i,6)
y2a=round(porcb1*longitud+y0s,6)
y2b=round(-porcb2*longitud+y0i,6)
```

```
328         #sup
329     #Cantidad de puntos
330 tamval=l1
331 #Cantidad de superficies
332 tamvals=L
333 #Cantidad de puntos a agregar
334 f=tamval
335 #Cantidad de lineas a agregar
336 s=2*tamval
337 #Textos por defecto
338 title='//Code generated by software programmed in Python\n'
339 k0='Include "surface.geo";\n'
340 k1='alpha = '
341 k2='*Pi/180;\n'
342 k3='ls[] = Rotate {{0, 0, 1}, {'
343 k4=','
344 k4a=', 0}, alpha} {Line{1,2,3,4};};\n'
345 k5='//Bounding box\n'
346 k6='Point('
347 k7=') = {'+str(xneg)+', '+str(ypos)+',0,lc};\n'
348 k8=') = {'+str(xneg)+', '+str(yneg)+',0,lc};\n'
349 k9=') = {'+str(xpos)+', '+str(ypos)+',0,lc};\n'
350 k10=') = {'+str(xpos)+', '+str(yneg)+',0,lc};\n'
351 k11='//Trailing edge\n'
352 k12=') = {'+str(xpos)+', '+str(y1_le_int)+',0,lc};\n'
353 k13=') = {'+str(xpos)+', '+str(y1_te)+',0,lc};\n'
354 k14='Line('
355 k15=') = {'
356 k16=','
357 k17='};\n'
358 k18='//Front\n'
359 k19=',0,lc};\n'
360 k20='BSpline('
361 k21='Spline('
362 k22='//Body and back\n'
363 #Spline te
364 k23=') = {'+str(l1_x_te_a)+', '+str(y1_te)+',0,lc};\n'
365 k24=') = {'+str(l1_x_te_b)+', '+str(y1_te)+',0,lc};\n'
366 k25=') = {'+str(l2_x_te)+', '+str(y2_te)+',0,lc};\n'
367 #Spline le
368 k26=') = {'+str(l1_x_le_int_a)+', '+str(y1_le_int)+',0,lc};\n'
369 k27=') = {'+str(l1_x_le_int_b)+', '+str(y1_le_int)+',0,lc};\n'
370 k28=') = {'+str(l2_x_le_int)+', '+str(y_le_int)+',0,lc};\n'
371 ################################
372 #Creacion del codigo
373 archi3=open('mesh.geo','w')
374 prog=['']
375 #Title, include, alpha y ls
376 prog.insert(0, title)
377 prog[len(prog)-1]=k0
378 A1=k1+str(alpha)+k2
379 A2=k3+str(cg)+k4+str(dy)+k4a
380 A3=k5
381 prog.insert(len(prog),A1)
382 prog.insert(len(prog),A2)
383 prog.insert(len(prog),A3)
384 #Bounding box
```

```
385 B=range(tamval+1,tamval+1+f) #Puntos
386 D=range(tamvals+1,tamvals+1+s) #Superficies
387 C1=k6+str(B[0])+k7
388 C2=k6+str(B[1])+k8
389 C3=k6+str(B[2])+k9
390 C4=k6+str(B[3])+k10
391 prog.insert(len(prog),C1)
392 prog.insert(len(prog),C2)
393 prog.insert(len(prog),C3)
394 prog.insert(len(prog),C4)
395 #Trailing edge
396 C5=k11
397 prog.insert(len(prog),C5)
398 #Spline
399 C8=k6+str(B[11])+k23
400 C9=k6+str(B[12])+k24
401 C10=k6+str(B[13])+k25
402 prog.insert(len(prog),C8)
403 prog.insert(len(prog),C9)
404 prog.insert(len(prog),C10)
405 D6=k14+str(D[31])+k15+str(a_1)+k16+str(B[11])+k17
406 D7=k14+str(D[32])+k15+str(z_2)+k16+str(B[12])+k17
407 prog.insert(len(prog),D6)
408 prog.insert(len(prog),D7)
409 D8=k20+str(D[33])+k15+str(B[11])+k16+str(B[13])+k16+str(B[12])+k17
410 prog.insert(len(prog),D8)
411 #
412 C6=k6+str(B[4])+k12
413 C7=k6+str(B[5])+k13
414 prog.insert(len(prog),C6)
415 prog.insert(len(prog),C7)
416 D1=k14+str(D[0])+k15+str(B[14])+k16+str(B[4])+k17
417 D2=k14+str(D[1])+k15+str(B[11])+k16+str(B[5])+k17
418 D3=k14+str(D[2])+k15+str(B[4])+k16+str(B[5])+k17
419 D4=k14+str(D[3])+k15+str(B[4])+k16+str(B[2])+k17
420 D5=k14+str(D[4])+k15+str(B[5])+k16+str(B[3])+k17
421 prog.insert(len(prog),D3)
422 prog.insert(len(prog),D4)
423 prog.insert(len(prog),D5)
424 #Front
425  #Spline peq
426 E1=k18
427 prog.insert(len(prog),E1)
428
429 E11=k6+str(B[14])+k26
430 E12=k6+str(B[15])+k27
431 E13=k6+str(B[16])+k28
432 prog.insert(len(prog),E11)
433 prog.insert(len(prog),E12)
434 prog.insert(len(prog),E13)
435 #
436 prog.insert(len(prog),D1)
437 prog.insert(len(prog),D2)
438 #
439 E14=k14+str(D[34])+k15+str(a_3_a)+k16+str(B[14])+k17
440 E15=k14+str(D[35])+k15+str(z_3_b)+k16+str(B[15])+k17
441 prog.insert(len(prog),E14)
```

```
442 prog.insert(len(prog),E15)
443 E16=k20+str(D[36])+k15+str(B[14])+k16+str(B[16])+k16+str(B[15])+k17
444 prog.insert(len(prog),E16)
445
446 #
447 E2=k6+str(B[6])+k15+str(l1_x_a)+k16+str(ypos)+k19
448 E3=k6+str(B[7])+k15+str(l1_x_b)+k16+str(yneg)+k19
449 E4=k6+str(B[8])+k15+str(l2_x)+k16+str(dy)+k19
450 prog.insert(len(prog),E2)
451 prog.insert(len(prog),E3)
452 prog.insert(len(prog),E4)
453 E5=k14+str(D[5])+k15+str(B[15])+k16+str(B[6])+k17
454 E6=k14+str(D[6])+k15+str(B[12])+k16+str(B[7])+k17
455 prog.insert(len(prog),E5)
456 prog.insert(len(prog),E6)
457 E7=k20+str(D[7])+k15+str(B[6])+k16+str(B[8])+k16+str(B[7])+k17
458 E8=k21+str(D[8])+k15+str(B[6])+k16+str(B[0])+k17
459 E9=k21+str(D[9])+k15+str(B[7])+k16+str(B[1])+k17
460 E10=k14+str(D[10])+k15+str(B[0])+k16+str(B[1])+k17
461 prog.insert(len(prog),E7)
462 prog.insert(len(prog),E8)
463 prog.insert(len(prog),E9)
464 prog.insert(len(prog),E10)
465
466 #Body and back
467 F1=k22
468 prog.insert(len(prog),F1)
469 F2=k6+str(B[9])+k15+str(xte_a)+k16+str(ypos)+k19
470 F3=k6+str(B[10])+k15+str(xte_b)+k16+str(yneg)+k19
471 prog.insert(len(prog),F2)
472 prog.insert(len(prog),F3)
473 F4=k14+str(D[11])+k15+str(B[6])+k16+str(B[9])+k17
474 F5=k14+str(D[12])+k15+str(B[7])+k16+str(B[10])+k17
475 prog.insert(len(prog),F4)
476 prog.insert(len(prog),F5)
477 F6=k14+str(D[13])+k15+str(B[14])+k16+str(B[9])+k17
478 F7=k14+str(D[14])+k15+str(B[11])+k16+str(B[10])+k17
479 F8=k14+str(D[15])+k15+str(B[9])+k16+str(B[2])+k17
480 F9=k14+str(D[16])+k15+str(B[10])+k16+str(B[3])+k17
481 prog.insert(len(prog),F6)
482 prog.insert(len(prog),F7)
483 prog.insert(len(prog),F8)
484 prog.insert(len(prog),F9)
485 #Lineas splines peq
486 F12=k6+str(B[17])+k15+str(p1x)+k16+str(y1a)+k19
487 F13=k6+str(B[18])+k15+str(p3x)+k16+str(y1b)+k19
488 F14=k6+str(B[19])+k15+str(p2x)+k16+str(y2a)+k19
489 F15=k6+str(B[20])+k15+str(p4x)+k16+str(y2b)+k19
490 prog.insert(len(prog),F12)
491 prog.insert(len(prog),F13)
492 prog.insert(len(prog),F14)
493 prog.insert(len(prog),F15)
494 F10=k20+str(D[37])+k15+str(B[14])+k16+str(B[17])+k16+str(B[19])+k16+str(B
      [11])+k17
495 F11=k20+str(D[38])+k15+str(B[15])+k16+str(B[18])+k16+str(B[20])+k16+str(B
      [12])+k17
496 prog.insert(len(prog),F10)
```

```
497 prog.insert(len(prog),F11)
498 #####
499 #####Mallado
500 j1='//Meshing\n'
501 j2='//Leading edge\n'
502 j3='//Vertical lines\n'
503 j4='//Body\n'
504 j5='//Back\n'
505 j6='//Front\n'
506 j7='//Small trailing edge\n'
507 j8='nl'
508 j9='rl'
509 j10='nv'
510 j11='rv'
511 j12='nb'
512 j13='rb'
513 j14='nba'
514 j15='rba'
515 j16='nf'
516 j17='rf'
517 j18='nsT'
518 j19='rsT'
519 j20=';\n'
520 j21='Transfinite Line{'
521 j22='} = '
522 j23=' Using Bump '
523 j24=' = '
524 j25=' Using Progression '
525 j26='//Small circumference\n'
526 j27='nc'
527 j28='rc'
528 prog.insert(len(prog),j1)
529 #leading edge
530 G1=j2
531 G2=j8+j24+str(nl)+j20
532 G3=j9+j24+str(rl)+j20
533 G4=j21+str(2)+j22+j8+j23+j9+j20
534 G5=j21+str(D[7])+j22+j8+j23+j9+j20
535 G6=j21+str(D[10])+j22+j8+j23+j9+j20
536 G7=j21+str(D[38])+j22+j8+j23+j9+j20
537 prog.insert(len(prog),G1)
538 prog.insert(len(prog),G2)
539 prog.insert(len(prog),G3)
540 prog.insert(len(prog),G4)
541 prog.insert(len(prog),G5)
542 prog.insert(len(prog),G6)
543 prog.insert(len(prog),G7)
544 #vertical lines
545 H1=j3
546 H2=j10+j24+str(nv)+j20
547 H3=j11+j24+str(rv)+j20
548 H4=j21+str(D[5])+j22+j10+j25+j11+j20
549 H5=j21+str(D[6])+j22+j10+j25+j11+j20
550 H6=j21+str(D[13])+j22+j10+j25+j11+j20
551 H7=j21+str(D[14])+j22+j10+j25+j11+j20
552 H8=j21+str(D[3])+j22+j10+j25+j11+j20
553 H9=j21+str(D[4])+j22+j10+j25+j11+j20
```

```
prog.insert(len(prog),H1)
prog.insert(len(prog),H2)
prog.insert(len(prog),H3)
prog.insert(len(prog),H4)
prog.insert(len(prog),H5)
prog.insert(len(prog),H6)
prog.insert(len(prog),H7)
prog.insert(len(prog),H8)
prog.insert(len(prog),H9)
#Body
J1=j4
J2=j12+j24+str(nb)+j20
J3=j13+j24+str(rb)+j20
J4=j21+str(D[11])+j22+j12+j23+j13+j20
J5=j21+str(3)+j22+j12+j23+j13+j20
J6=j21+str(4)+j22+j12+j23+j13+j20
J7=j21+str(D[12])+j22+j12+j23+j13+j20
J8=j21+str(D[36])+j22+j12+j23+j13+j20
J9=j21+str(D[33])+j22+j12+j23+j13+j20
prog.insert(len(prog),J1)
prog.insert(len(prog),J2)
prog.insert(len(prog),J3)
prog.insert(len(prog),J4)
prog.insert(len(prog),J5)
prog.insert(len(prog),J6)
prog.insert(len(prog),J7)
prog.insert(len(prog),J8)
prog.insert(len(prog),J9)
#Back
K1=j5
K2=j14+j24+str(nba)+j20
K3=j15+j24+str(rba)+j20
K4=j21+str(D[15])+j22+j14+j25+j15+j20
K5=j21+str(D[0])+j22+j14+j25+j15+j20
K6=j21+str(D[1])+j22+j14+j25+j15+j20
K7=j21+str(D[16])+j22+j14+j25+j15+j20
prog.insert(len(prog),K1)
prog.insert(len(prog),K2)
prog.insert(len(prog),K3)
prog.insert(len(prog),K4)
prog.insert(len(prog),K5)
prog.insert(len(prog),K6)
prog.insert(len(prog),K7)
#Front
M1=j6
M2=j16+j24+str(nf)+j20
M3=j17+j24+str(rf)+j20
M4=j21+str(D[8])+j22+j16+j25+j17+j20
M5=j21+str(D[9])+j22+j16+j25+j17+j20
prog.insert(len(prog),M1)
prog.insert(len(prog),M2)
prog.insert(len(prog),M3)
prog.insert(len(prog),M4)
prog.insert(len(prog),M5)
#Small traling edge
N1=j7
N2=j18+j24+str(nsT)+j20
```

```
611 N3=j19+j24+str(rsT)+j20
612 N4=j21+str(1)+j22+j18+j23+j19+j20
613 N5=j21+str(D[2])+j22+j18+j23+j19+j20
614 N5a=j21+str(D[37])+j22+j18+j23+j19+j20
615 prog.insert(len(prog),N1)
616 prog.insert(len(prog),N2)
617 prog.insert(len(prog),N3)
618 prog.insert(len(prog),N4)
619 prog.insert(len(prog),N5)
620 prog.insert(len(prog),N5a)
621 #New small circunference
622 N6=j26
623 N7=j27+j24+str(nc)+j20
624 N8=j28+j24+str(rc)+j20
625 N9=j21+str(D[31])+j22+j27+j25+j28+j20
626 N10=j21+str(D[32])+j22+j27+j25+j28+j20
627 N11=j21+str(D[34])+j22+j27+j25+j28+j20
628 N12=j21+str(D[35])+j22+j27+j25+j28+j20
629 prog.insert(len(prog),N6)
630 prog.insert(len(prog),N7)
631 prog.insert(len(prog),N8)
632 prog.insert(len(prog),N9)
633 prog.insert(len(prog),N10)
634 prog.insert(len(prog),N11)
635 prog.insert(len(prog),N12)
636
637 #Loops
638 k0='//Loops\n'
639 k1='Line Loop('
640 k2=') = {'
641 k3=', '
642 k4='};\n'
643 k5='Plane Surface('
644 k6='//Surfaces\n'
645 k7='Transfinite Surface {'
646 k8='//Transfinite Surfaces\n'
647 k9='Recombine Surface {'
648 k10='//Recombine Surfaces\n'
649 P1=k0
650 prog.insert(len(prog),P1)
651 P2=k1+str(D[17])+k2+str(D[38])+k3+str(D[6])+k3+str(-D[7])+k3+str(-D[5])+
      k4
652 P3=k1+str(D[18])+k2+str(D[8])+k3+str(D[10])+k3+str(-D[9])+k3+str(-D[7])+
      k4
653 P4=k1+str(D[19])+k2+str(D[11])+k3+str(-D[13])+k3+str(D[36])+k3+str(D[5])+
      k4
654 P5=k1+str(D[20])+k2+str(-D[33])+k3+str(D[14])+k3+str(-D[12])+k3+str(-D
      [6])+k4
655 P6=k1+str(D[21])+k2+str(D[13])+k3+str(D[15])+k3+str(-D[3])+k3+str(-D[0])+
      k4
656 P7=k1+str(D[22])+k2+str(D[14])+k3+str(D[16])+k3+str(-D[4])+k3+str(-D[1])+
      k4
657 P8=k1+str(D[23])+k2+str(D[0])+k3+str(D[2])+k3+str(-D[1])+k3+str(-D[37])+
      k4
658 P8a=k1+str(D[39])+k2+str(1)+k3+str(D[34])+k3+str(D[37])+k3+str(-D[31])+k4
659 P8b=k1+str(D[40])+k2+str(D[31])+k3+str(D[33])+k3+str(-D[32])+k3+str(-4)+
      k4
```

```
660 P8c=k1+str(D[41])+k2+str(D[32])+k3+str(-D[38])+k3+str(-D[35])+k3+str(2)+
        k4
661 P8d=k1+str(D[42])+k2+str(D[35])+k3+str(-D[36])+k3+str(-D[34])+k3+str(3)+
        k4
662 prog.insert(len(prog),P2)
663 prog.insert(len(prog),P3)
664 prog.insert(len(prog),P4)
665 prog.insert(len(prog),P5)
666 prog.insert(len(prog),P6)
667 prog.insert(len(prog),P7)
668 prog.insert(len(prog),P8)
669 prog.insert(len(prog),P8a)
670 prog.insert(len(prog),P8b)
671 prog.insert(len(prog),P8c)
672 prog.insert(len(prog),P8d)
673 #Plane surface
674 P9=k6
675 P10=k5+str(D[24])+k2+str(D[17])+k4
676 P11=k5+str(D[25])+k2+str(D[18])+k4
677 P12=k5+str(D[26])+k2+str(D[19])+k4
678 P13=k5+str(D[27])+k2+str(D[20])+k4
679 P14=k5+str(D[28])+k2+str(D[21])+k4
680 P15=k5+str(D[29])+k2+str(D[22])+k4
681 P16=k5+str(D[30])+k2+str(D[23])+k4
682 P16a=k5+str(D[43])+k2+str(D[39])+k4
683 P16b=k5+str(D[44])+k2+str(D[40])+k4
684 P16c=k5+str(D[45])+k2+str(D[41])+k4
685 P16d=k5+str(D[46])+k2+str(D[42])+k4
686 prog.insert(len(prog),P9)
687 prog.insert(len(prog),P10)
688 prog.insert(len(prog),P11)
689 prog.insert(len(prog),P12)
690 prog.insert(len(prog),P13)
691 prog.insert(len(prog),P14)
692 prog.insert(len(prog),P15)
693 prog.insert(len(prog),P16)
694 prog.insert(len(prog),P16a)
695 prog.insert(len(prog),P16b)
696 prog.insert(len(prog),P16c)
697 prog.insert(len(prog),P16d)
698 #Transfinite surfaces
699 P17=k8
700 P18=k7+str(D[24])+k4
701 P19=k7+str(D[25])+k4
702 P20=k7+str(D[26])+k4
703 P21=k7+str(D[27])+k4
704 P22=k7+str(D[28])+k4
705 P23=k7+str(D[29])+k4
706 P24=k7+str(D[30])+k4
707 P24a=k7+str(D[43])+k4
708 P24b=k7+str(D[44])+k4
709 P24c=k7+str(D[45])+k4
710 P24d=k7+str(D[46])+k4
711 prog.insert(len(prog),P17)
712 prog.insert(len(prog),P18)
713 prog.insert(len(prog),P19)
714 prog.insert(len(prog),P20)
```

```
715 prog.insert(len(prog),P21)
716 prog.insert(len(prog),P22)
717 prog.insert(len(prog),P23)
718 prog.insert(len(prog),P24)
719 prog.insert(len(prog),P24a)
720 prog.insert(len(prog),P24b)
721 prog.insert(len(prog),P24c)
722 prog.insert(len(prog),P24d)
723 #Recombine surfaces
724 P25=k10
725 P26=k9+str(D[24])+k4
726 P27=k9+str(D[25])+k4
727 P28=k9+str(D[26])+k4
728 P29=k9+str(D[27])+k4
729 P30=k9+str(D[28])+k4
730 P31=k9+str(D[29])+k4
731 P32=k9+str(D[30])+k4
732 P32a=k9+str(D[43])+k4
733 P32b=k9+str(D[44])+k4
734 P32c=k9+str(D[45])+k4
735 P32d=k9+str(D[46])+k4
736 prog.insert(len(prog),P25)
737 prog.insert(len(prog),P26)
738 prog.insert(len(prog),P27)
739 prog.insert(len(prog),P28)
740 prog.insert(len(prog),P29)
741 prog.insert(len(prog),P30)
742 prog.insert(len(prog),P31)
743 prog.insert(len(prog),P32)
744 prog.insert(len(prog),P32a)
745 prog.insert(len(prog),P32b)
746 prog.insert(len(prog),P32c)
747 prog.insert(len(prog),P32d)
748 #Extrude
749 m1='//Extrusion\n'
750 m2='Extrude {0, 0, '
751 m3='}{ \n'
752 m4='    Surface{'
753 m5=','
754 m6='};\n'
755 m7='    Layers{'
756 m8='    Recombine;\n'
757 m9='  }\n'
758 Q0=m1
759 Q1=m2+str(ev)+m3
760 Q2=m4+str(D[24])+m5+str(D[25])+m5+str(D[26])+m5+str(D[27])+m5+str(D[28])+
        m5+str(D[29])+m5+str(D[30])+m5+str(D[43])+m5+str(D[44])+m5+str(D[45])+
        m5+str(D[46])+m6
761 Q3=m7+str(lay)+m6
762 Q4=m8
763 Q5=m9
764 prog.insert(len(prog),Q0)
765 prog.insert(len(prog),Q1)
766 prog.insert(len(prog),Q2)
767 prog.insert(len(prog),Q3)
768 prog.insert(len(prog),Q4)
769 prog.insert(len(prog),Q5)
```

```
770 #Names
771 n0='//Names\n'
772 n1='"internal"'
773 n2='"INL"'
774 n3='"OUTL"'
775 n4='"TOPBOT"'
776 n5='"WING"'
777 n6='"FRONT"'
778 n7='"BACK"'
779 n8='Physical Volume('
780 n9='Physical Surface('
781 n10=') = {'
782 n11=','
783 n12='};\n'
784 #Calculos
785 vol=range(1,cantidaddevolumenes+1)
786 vol=str(vol)
787 vol=vol.replace("[",'') #Elimina [
788 vol=vol.replace("]",'') #Elimina ]
789 front=range(D[24],D[30]+1)
790 frontb=range(D[43],D[46]+1)
791 front=front+frontb
792 front=str(front)
793 front=front.replace("[",'') #Elimina [
794 front=front.replace("]",'') #Elimina ]
795 #Valores se deben colocar visualizando en Gmsh
796 #Front-back 29 73, 30 95, 31 117, 32 139, 33 161, 34 183, 35 205, 48
    227,49 249, 50 271, 51 293
797 R1=n0
798 R2=n8+n1+n10+vol+n12
799 R3=n9+n2+n10+str(86)+n12
800 R4=n9+n3+n10+'156,196,178'+n12
801 R5=n9+n4+n10+'82,104,152,90,134,174'+n12
802 R6=n9+n5+n10+'292,214,248,270'+n12
803 R7=n9+n6+n10+str(front)+n12
804 R8=n9+n7+n10+'73,95,117,139,161,183,205,227,249,271,293'+n12
805 prog.insert(len(prog),R1)
806 prog.insert(len(prog),R2)
807 prog.insert(len(prog),R3)
808 prog.insert(len(prog),R4)
809 prog.insert(len(prog),R5)
810 prog.insert(len(prog),R6)
811 prog.insert(len(prog),R7)
812 prog.insert(len(prog),R8)
813 archi3.writelines(prog)
814 archi3.close
```

APPENDIX C FILES USED IN GMSH

C.1 POINTS - FRANCIS99 - UP SECTION (puntos.geo)

```
//Code generated by software programmed in Python
//Points quantity: 104
lc = 1e-06;
Point(1) = {0.0,-0.0,0,lc};
Point(2) = {3.8e-05,-0.000344,0,lc};
Point(3) = {0.000164,-0.000752,0,lc};
Point(4) = {0.000398,-0.001133,0,lc};
Point(5) = {0.000742,-0.001466,0,lc};
Point(6) = {0.001297,-0.001824,0,lc};
Point(7) = {0.001958,-0.002191,0,lc};
Point(8) = {0.002776,-0.002486,0,lc};
Point(9) = {0.003894,-0.002786,0,lc};
Point(10) = {0.005405,-0.003067,0,lc};
Point(11) = {0.007755,-0.0034,0,lc};
Point(12) = {0.010293,-0.003669,0,lc};
Point(13) = {0.012654,-0.003809,0,lc};
Point(14) = {0.01551,-0.003952,0,lc};
Point(15) = {0.019179,-0.00403,0,lc};
Point(16) = {0.023785,-0.003944,0,lc};
Point(17) = {0.028852,-0.003771,0,lc};
Point(18) = {0.03313,-0.003549,0,lc};
Point(19) = {0.037427,-0.003415,0,lc};
Point(20) = {0.042305,-0.003076,0,lc};
Point(21) = {0.046934,-0.002671,0,lc};
Point(22) = {0.052457,-0.001963,0,lc};
Point(23) = {0.057853,-0.00082,0,lc};
Point(24) = {0.06334,0.000552,0,lc};
Point(25) = {0.068447,0.001464,0,lc};
Point(26) = {0.072653,0.002282,0,lc};
Point(27) = {0.076961,0.003059,0,lc};
Point(28) = {0.080804,0.003771,0,lc};
Point(29) = {0.085188,0.004497,0,lc};
Point(30) = {0.088911,0.005184,0,lc};
Point(31) = {0.093172,0.005971,0,lc};
Point(32) = {0.097749,0.006842,0,lc};
Point(33) = {0.102523,0.007516,0,lc};
Point(34) = {0.107836,0.008692,0,lc};
Point(35) = {0.113333,0.009614,0,lc};
Point(36) = {0.118701,0.010553,0,lc};
Point(37) = {0.125319,0.011339,0,lc};
Point(38) = {0.131607,0.012039,0,lc};
Point(39) = {0.138303,0.012368,0,lc};
Point(40) = {0.144193,0.012551,0,lc};
Point(41) = {0.150086,0.012501,0,lc};
Point(42) = {0.156639,0.012141,0,lc};
Point(43) = {0.1615,0.011664,0,lc};
Point(44) = {0.167381,0.010934,0,lc};
Point(45) = {0.17284,0.009763,0,lc};
Point(46) = {0.177849,0.008649,0,lc};
Point(47) = {0.182596,0.007117,0,lc};
Point(48) = {0.18619,0.005872,0,lc};
Point(49) = {0.189766,0.004235,0,lc};
```

```
53 Point(50) = {0.194328,0.002322,0,lc};
54 Point(51) = {0.199863,-0.000656,0,lc};
55 Point(52) = {0.199915,0.00019,0,lc};
56 Point(53) = {0.195051,0.003117,0,lc};
57 Point(54) = {0.190172,0.005335,0,lc};
58 Point(55) = {0.186528,0.006802,0,lc};
59 Point(56) = {0.183012,0.007991,0,lc};
60 Point(57) = {0.178201,0.009487,0,lc};
61 Point(58) = {0.173014,0.010899,0,lc};
62 Point(59) = {0.167741,0.011963,0,lc};
63 Point(60) = {0.161782,0.012768,0,lc};
64 Point(61) = {0.156762,0.013399,0,lc};
65 Point(62) = {0.150202,0.013832,0,lc};
66 Point(63) = {0.144289,0.013785,0,lc};
67 Point(64) = {0.138244,0.013662,0,lc};
68 Point(65) = {0.131458,0.013458,0,lc};
69 Point(66) = {0.125132,0.012873,0,lc};
70 Point(67) = {0.118486,0.012282,0,lc};
71 Point(68) = {0.113233,0.011499,0,lc};
72 Point(69) = {0.107638,0.010488,0,lc};
73 Point(70) = {0.102225,0.009572,0,lc};
74 Point(71) = {0.097227,0.008871,0,lc};
75 Point(72) = {0.092629,0.008093,0,lc};
76 Point(73) = {0.088166,0.007363,0,lc};
77 Point(74) = {0.084645,0.006797,0,lc};
78 Point(75) = {0.080562,0.006201,0,lc};
79 Point(76) = {0.076383,0.005584,0,lc};
80 Point(77) = {0.072407,0.005017,0,lc};
81 Point(78) = {0.067877,0.004411,0,lc};
82 Point(79) = {0.062798,0.003521,0,lc};
83 Point(80) = {0.05733,0.00251,0,lc};
84 Point(81) = {0.051645,0.001376,0,lc};
85 Point(82) = {0.046836,0.000832,0,lc};
86 Point(83) = {0.041974,0.000523,0,lc};
87 Point(84) = {0.036945,0.000404,0,lc};
88 Point(85) = {0.032698,0.00031,0,lc};
89 Point(86) = {0.028232,0.00018,0,lc};
90 Point(87) = {0.023386,0.000145,0,lc};
91 Point(88) = {0.018748,0.000282,0,lc};
92 Point(89) = {0.015183,0.000449,0,lc};
93 Point(90) = {0.012854,0.000635,0,lc};
94 Point(91) = {0.010766,0.000732,0,lc};
95 Point(92) = {0.008578,0.000936,0,lc};
96 Point(93) = {0.006314,0.001291,0,lc};
97 Point(94) = {0.004595,0.001526,0,lc};
98 Point(95) = {0.00344,0.001666,0,lc};
99 Point(96) = {0.002421,0.001762,0,lc};
100 Point(97) = {0.001737,0.001726,0,lc};
101 Point(98) = {0.001188,0.001533,0,lc};
102 Point(99) = {0.000561,0.001227,0,lc};
103 Point(100) = {0.000196,0.000816,0,lc};
104 Point(101) = {5.6e-05,0.000398,0,lc};
105 Point(102) = {0.199891,-0.000662,0,lc};
106 Point(103) = {0.199971,0.000146,0,lc};
107 Point(104) = {0.2,-0.000304,0,lc};
```

C.2 LINES FOR SURFACES - FRANCIS99 - UP SECTION (surface.geo)

```
//Code generated by software programmed in Python
//Surfaces quantity: 4
Include "puntos.geo";
BSpline(1) = {52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66,
    67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83,
   84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95};
BSpline(2) = {9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23,
   24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,
   41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51};
BSpline(3) = {95, 96, 97, 98, 99, 100, 101, 1, 2, 3, 4, 5, 6, 7, 8, 9};
BSpline(4) = {52,103,102,51};
```

C.3 MESH - FRANCIS99 - UP SECTION (mesh.geo)

```
//Code generated by software programmed in Python
Include "surface.geo";
alpha = -10*Pi/180;
ls[] = Rotate {{0, 0, 1}, {0.1,0.00820986, 0}, alpha} {Line{1,2,3,4};};
//Bounding box
Point(105) = {-0.3, 0.1682,0,lc};
Point(106) = {-0.3, -0.1518,0,lc};
Point(107) = {0.9, 0.1682,0,lc};
Point(108) = {0.9, -0.1518,0,lc};
//Trailing edge
Point(116) = {0.209041, -0.00531789,0,lc};
Point(117) = {0.199041, -0.02931789,0,lc};
Point(118) = {0.209041, -0.01731789,0,lc};
Line(36) = {52,116};
Line(37) = {51,117};
BSpline(38) = {116,118,117};
Point(109) = {0.9, -0.00531789,0,lc};
Point(110) = {0.9, -0.02931789,0,lc};
Line(5) = {116,109};
Line(6) = {117,110};
Line(7) = {109,110};
Line(8) = {109,107};
Line(9) = {110,108};
//Front
Point(119) = {-0.00091, 0.04811558,0,lc};
Point(120) = {-0.00091, -0.01685799,0,lc};
Point(121) = {-0.01991, 0.01562879,0,lc};
Line(39) = {95,119};
Line(40) = {9,120};
BSpline(41) = {119,121,120};
Point(111) = {-0.01991,0.1682,0,lc};
Point(112) = {-0.01991,-0.1518,0,lc};
Point(113) = {-0.09991,0.01749,0,lc};
Line(10) = {119,111};
Line(11) = {120,112};
BSpline(12) = {111,113,112};
Spline(13) = {111,105};
Spline(14) = {112,106};
Line(15) = {105,106};
```

```
40 //Body and back
41 Point(114) = {0.236403,0.1682,0,lc};
42 Point(115) = {0.236403,-0.1518,0,lc};
43 Line(16) = {111,114};
44 Line(17) = {112,115};
45 Line(18) = {116,114};
46 Line(19) = {117,115};
47 Line(20) = {114,107};
48 Line(21) = {115,108};
49 Point(122) = {0.02908265,0.058399,0,lc};
50 Point(123) = {0.02908265,-0.053088,0,lc};
51 Point(124) = {0.184044675,0.037399,0,lc};
52 Point(125) = {0.184044675,-0.043088,0,lc};
53 BSpline(42) = {119,122,124,116};
54 BSpline(43) = {120,123,125,117};
55 //Meshing
56 //Leading edge
57 nl = 50;
58 rl = 1;
59 Transfinite Line{3} = nl Using Bump rl;
60 Transfinite Line{12} = nl Using Bump rl;
61 Transfinite Line{15} = nl Using Bump rl;
62 Transfinite Line{41} = nl Using Bump rl;
63 //Vertical lines
64 nv = 35;
65 rv = 1.035;
66 Transfinite Line{10} = nv Using Progression rv;
67 Transfinite Line{11} = nv Using Progression rv;
68 Transfinite Line{18} = nv Using Progression rv;
69 Transfinite Line{19} = nv Using Progression rv;
70 Transfinite Line{8} = nv Using Progression rv;
71 Transfinite Line{9} = nv Using Progression rv;
72 //Body
73 nb = 120;
74 rb = 0.05;
75 Transfinite Line{16} = nb Using Bump rb;
76 Transfinite Line{1} = nb Using Bump rb;
77 Transfinite Line{2} = nb Using Bump rb;
78 Transfinite Line{17} = nb Using Bump rb;
79 Transfinite Line{42} = nb Using Bump rb;
80 Transfinite Line{43} = nb Using Bump rb;
81 //Back
82 nba = 110;
83 rba = 1.025;
84 Transfinite Line{20} = nba Using Progression rba;
85 Transfinite Line{5} = nba Using Progression rba;
86 Transfinite Line{6} = nba Using Progression rba;
87 Transfinite Line{21} = nba Using Progression rba;
88 //Front
89 nf = 50;
90 rf = 1.065;
91 Transfinite Line{13} = nf Using Progression rf;
92 Transfinite Line{14} = nf Using Progression rf;
93 //Small trailing edge
94 nsT = 30;
95 rsT = 0.05;
96 Transfinite Line{4} = nsT Using Bump rsT;
```

```
97 Transfinite Line{7} = nsT Using Bump rsT;
98 Transfinite Line{38} = nsT Using Bump rsT;
99 //Small circumference
100 nc = 80;
101 rc = 1.025;
102 Transfinite Line{36} = nc Using Progression rc;
103 Transfinite Line{37} = nc Using Progression rc;
104 Transfinite Line{39} = nc Using Progression rc;
105 Transfinite Line{40} = nc Using Progression rc;
106 //Loops
107 Line Loop(22) = {41, 11, -12, -10};
108 Line Loop(23) = {13, 15, -14, -12};
109 Line Loop(24) = {16, -18, -42, 10};
110 Line Loop(25) = {43, 19, -17, -11};
111 Line Loop(26) = {18, 20, -8, -5};
112 Line Loop(27) = {19, 21, -9, -6};
113 Line Loop(28) = {5, 7, -6, -38};
114 Line Loop(44) = {1, 39, 42, -36};
115 Line Loop(45) = {36, 38, -37, -4};
116 Line Loop(46) = {37, -43, -40, 2};
117 Line Loop(47) = {40, -41, -39, 3};
118 //Surfaces
119 Plane Surface(29) = {22};
120 Plane Surface(30) = {23};
121 Plane Surface(31) = {24};
122 Plane Surface(32) = {25};
123 Plane Surface(33) = {26};
124 Plane Surface(34) = {27};
125 Plane Surface(35) = {28};
126 Plane Surface(48) = {44};
127 Plane Surface(49) = {45};
128 Plane Surface(50) = {46};
129 Plane Surface(51) = {47};
130 //Transfinite Surfaces
131 Transfinite Surface {29};
132 Transfinite Surface {30};
133 Transfinite Surface {31};
134 Transfinite Surface {32};
135 Transfinite Surface {33};
136 Transfinite Surface {34};
137 Transfinite Surface {35};
138 Transfinite Surface {48};
139 Transfinite Surface {49};
140 Transfinite Surface {50};
141 Transfinite Surface {51};
142 //Recombine Surfaces
143 Recombine Surface {29};
144 Recombine Surface {30};
145 Recombine Surface {31};
146 Recombine Surface {32};
147 Recombine Surface {33};
148 Recombine Surface {34};
149 Recombine Surface {35};
150 Recombine Surface {48};
151 Recombine Surface {49};
152 Recombine Surface {50};
153 Recombine Surface {51};
```

```
154 //Extrusion
155 Extrude {0, 0, 0.0032}{
156     Surface{29,30,31,32,33,34,35,48,49,50,51};
157     Layers{2};
158     Recombine;
159   }
160 //Names
161 Physical Volume("internal") = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11};
162 Physical Surface("INL") = {86};
163 Physical Surface("OUTL") = {156,196,178};
164 Physical Surface("TOPBOT") = {82,104,152,90,134,174};
165 Physical Surface("WING") = {292,214,248,270};
166 Physical Surface("FRONT") = {29, 30, 31, 32, 33, 34, 35, 48, 49, 50, 51};
167 Physical Surface("BACK") = {73,95,117,139,161,183,205,227,249,271,293};
```

APPENDIX D PROGRAMATION IN GMSH

In this section are mentioned the different possibilities to generate a mesh in Gmsh.

This section start with a differentiation between different types of characteristics lengths, continue with transfinite option and finish with the recombination parameters to generate a structured mesh in a simple example.

The simple example is a square figure, denominated “points.geo”, established by the next code:

```
//Initial points (square)
lc = 0.01;
a=0.1;
Point ( 1 ) = { 0 , 0 , 0.0 ,lc } ;
Point ( 2 ) = { a , 0 , 0.0 ,lc } ;
Point ( 3 ) = { a , a , 0.0 ,lc } ;
Point ( 4 ) = { 0 , a , 0.0 ,lc } ;
```

That reproduces the next figure in Gmsh:

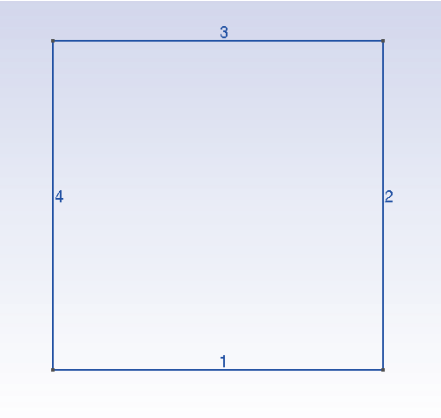


Figure D.1: Square-base example generated in Gmsh

The lines in the figure, included in the file “lines.geo” are generated by the code:

```
// Lines
Include "points.geo";
Line(1) = {1,2};
Line(2) = {2,3};
Line(3) = {3,4};
Line(4) = {4,1};
```

D.1 CHARACTERISTICS LENGTHS

An initial auto-generation of the 2D mesh is generated by Gmsh assuming the next code for the surface:

```
1 //Mesh
2 Include "lines.geo";
3 Line Loop(5) = {1, 2, 3, 4};
4 Plane Surface(6) = {5};
```

That reproduces:

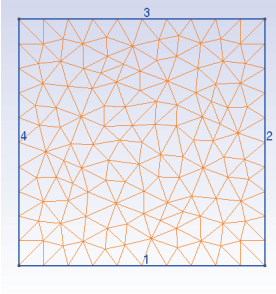


Figure D.2: Auto 2D mesh generated by Gmsh in the example figure

Varying the value of the characteristic length lc between $0.001, 0.005, 0.01$ and 0.05 its clear the difference of the auto-generation mesh:

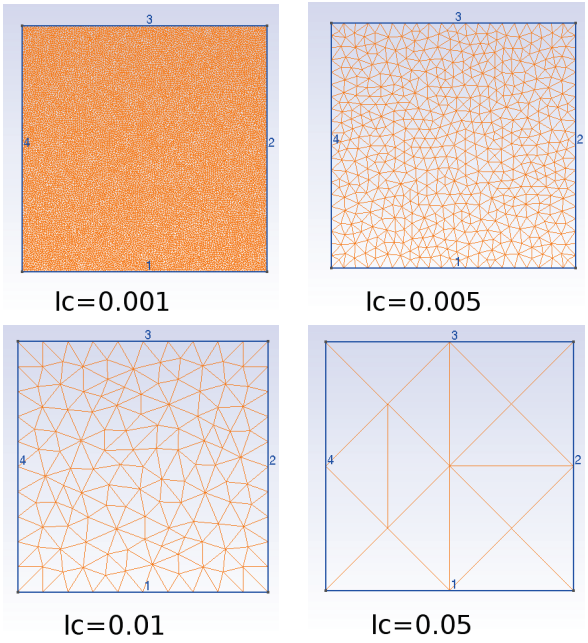


Figure D.3: Auto 2D meshes generated by Gmsh varying the lc between $0.001, 0.005, 0.01$ and 0.05. As lc increase, the mesh refinement decreases. The quantity of divisions in the lines is equal at a/lc (line length / characteristic length)

D.2 TRANSFINITE OPTION

Using the parameter of transfinite is possible to generate a different distribution in the mesh. The next code, in the created file "mesh-b.geo", generates a variation in the mesh by the application of transfinite in the line 1.

```
1 //Mesh
2 Include "lines.geo";
3 //Transfinite parameters:
4 nl = 1;
5 rl = 1;
6 Transfinite Line{1} = nl Using Progression rl;
7 Line Loop(5) = {1, 2, 3, 4};
8 Plane Surface(6) = {5};
```

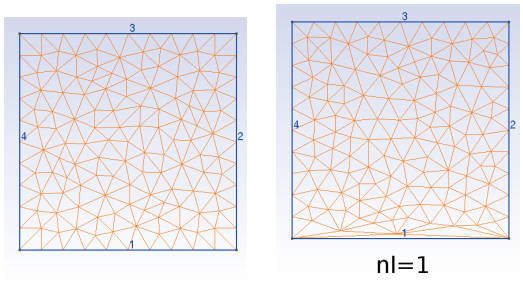


Figure D.4: Meshes generated without (left) and with (right) transfinite option

By varying the parameters nl and rl for transfinite it is possible to generate a complete different mesh distribution, for example varying nl between $1, 5, 10$ and 50 is possible to see the difference generated:

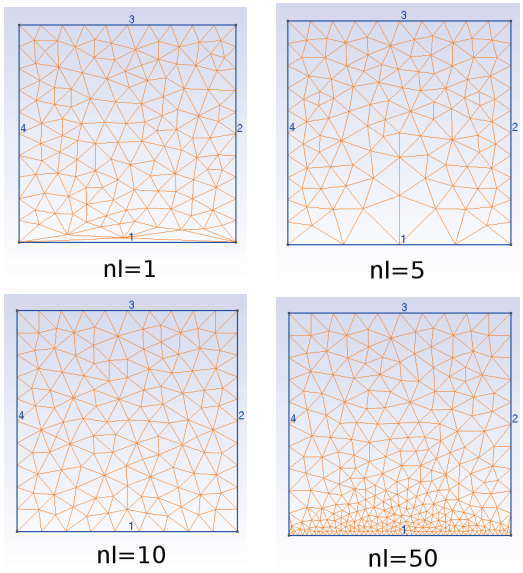


Figure D.5: Meshes generated varying nl in transfinite option between $1, 5, 10$ and 50. As nl increase, the mesh refinement increase near the line 1.

With $nl = 50$ and varying rl between $0.95, 1.00, 1.05$ and 1.09 there another difference in the generated meshes:

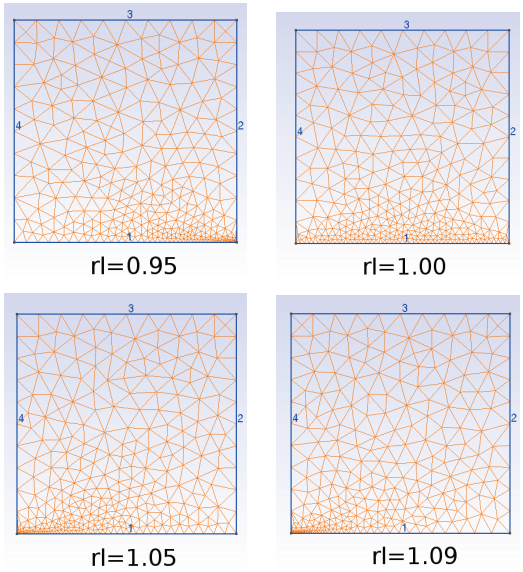


Figure D.6: Meshes generated with $nl = 50$ and varying rl in transfinite option between $0.95, 1.00, 1.05$ and 1.09. As rl decrease, the mesh refinement increase near the end section of the line. As rl increase, mesh refinement is bigger near the initial section.

The figures D.4, D.5 and D.6 are generated using the functionality "Progression". Another form to generate a transfinite is by using "Bump". This is described below using $rl = 0.1$ and 10.

```
//Mesh
Include "lines.geo";
//Transfinite parameters:
nl = 50;
rl = 0.1;
Transfinite Line{1} = nl Using Bump rl;
Line Loop(5) = {1, 2, 3, 4};
Plane Surface(6) = {5};
```

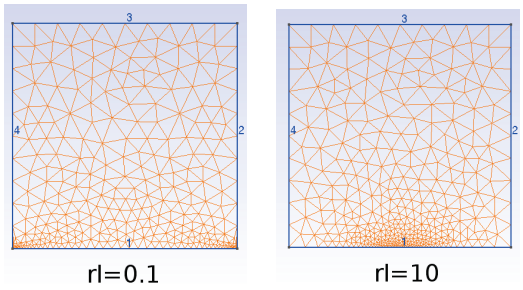


Figure D.7: Meshes generated with Bump transfinite, with values of $rl = 0.1$ (left) and 10 (right). As rl decrease, the refinement is bigger at the ends of the line 1. As rl increase, the refinement grows at the center.

D.3 RECOMBINATION

To generate structured meshes is necessary to recombine the triangles to obtain quadrilaterals. This process is generated using the functionality "Recombine" of Gmsh, like in the next code:

```
//Mesh
Include "lines.geo";
Line Loop(5) = {1, 2, 3, 4};
Plane Surface(6) = {5};
Recombine Surface {6};
```

That produces the next mesh:

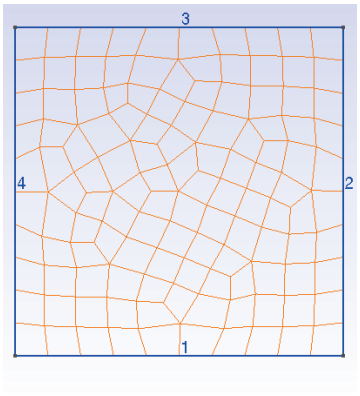


Figure D.8: Recombined mesh in a square

With the addition of a "Transfinite Surface" at the code is possible to obtain the searched structured mesh.

```
//Mesh
Include "lines.geo";
Line Loop(5) = {1, 2, 3, 4};
Plane Surface(6) = {5};
Transfinite Surface {6};
Recombine Surface {6};
```

That produces the next mesh:

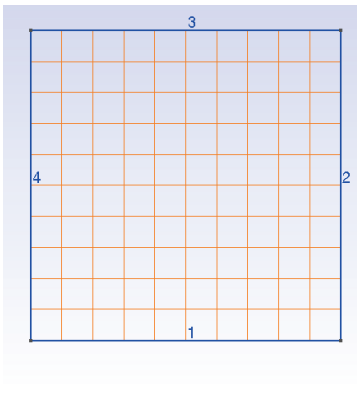


Figure D.9: Structured mesh in a square

By the combination of “Transfinite Line”, “Transfinite Surface” and “Recombine” is possible to obtain a structured mesh with a refinement near a line:

```
//Mesh
Include "lines.geo";
//Transfinite parameters:
nl = 31;
rl = 1.075;
rl2=1/rl;
Transfinite Line{2} = nl Using Progression rl;
Transfinite Line{4} = nl Using Progression rl2;
Line Loop(5) = {1, 2, 3, 4};
Plane Surface(6) = {5};
Transfinite Surface {6};
Recombine Surface {6};
```

That produces the structured mesh with a refinement near the line 1:

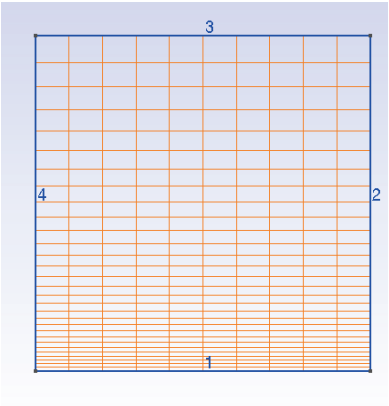


Figure D.10: Structured mesh with a refinement near the line 1.

As seen, the “Transfinite Line” command generate a subdivision in the line. The parameter “Progression” generates the refinement in one specific side, like figure D.6. The parameter “Bump” generates the refinement in both sides or in the center, like in figure D.7. The parameter “nl” is the quantity of subdivision of the line minus 1. The parameter rl is used to locate de greater refinement zone.

APPENDIX E OPENFOAM FILES DESCRIPTION

E.1 DESCRIPTION OF FILES IN FOLDER "0"

Each file that conform the folder "0" is formed with the description of: the dimensions of the file, the value for the internal field and the values for each boundary field.

- The description of the dimension of the file is based in a vector of base units, in this time expressed in *Système International*, SI.

$$[0\,0\,0\,0\,0\,0\,0] \tag{E.1}$$

Where each element of the vector represents:

Table E.1: Form to express the dimension of a file (OpenFOAM, 2015, p.112).

No.	Property	SI unit
1	Mass	kg
2	Length	m
3	Time	s
4	Temperature	K
5	Quantity	mol
6	Current	A (ampere)
7	Luminous intensity	cd (candela)

It is possible to express the units also in the United States Customary System, USCS. This must be expressed to the software. By default is used SI units.

The units for each file are mentioned below:

Table E.2: Dimensions of files from folder "0".

File	Value	Unit	Simplified unit
alpha1	[0 0 0 0 0 0 0]	dimensionless	$----$
p_rgh	[1 -1 -2 0 0 0 0]	$^{kg}/_{ms^2}$	Pa
U	[0 1 -1 0 0 0 0]	$^{m}/_{s}$	$----$

- The internal field for each file express the condition at which the parameter is. In the next table is included this.

Table E.3: Internal field of files from folder "0".

File	Type	Value	Description
alpha1	uniform	1	Initially in liquid phase (1), (gas phase is (0))
p_rgh	uniform	95977	Uniform pressure
U	uniform	(39.7, 0, 0)	Uniform velocity in x direction

- The boundary field of each file describe the type and value for all the boundaries. This is included in the next table.

Table E.4: Internal field of files from folder "0" for "alpha1".

Boundary	**alpha1**	
	type	**value**
INL	fixedValue	$internalField
OUTL	inletOutlet	$internalField
WING	intelOutlet	$internalField
TOPBOT	slip	—-
FRONT	symmetryPlane	—-
BACK	symmetryPlane	—-

Table E.5: Internal field of files from folder "0" for "r_rgh".

Boundary	**p_rgh**	
	type	**value**
INL	zeroGradient	—-
OUTL	fixedValue	$internalField
WING	zeroGradient	—-
TOPBOT	slip	—-
FRONT	symmetryPlane	—-
BACK	symmetryPlane	—-

Table E.6: Internal field of files from folder "0" for "U".

Boundary	**U**	
	type	**value**
INL	fixedValue	$internalField
OUTL	zeroGradient	—-
WING	fixedValue	uniform (0 0 0)
TOPBOT	slip	—-
FRONT	symmetryPlane	—-
BACK	symmetryPlane	—-

The description of each type and value of the boundaries is mentioned at the end of the appendix.

E.2 DESCRIPTION OF FILES IN FOLDER "constant"

The folder denominated "constant" is formed by 4 files and one folder. This files are described below:

Table E.7: Description of the files of the folder "constant".

File	**Description**
g	Conditions for gravity
LESProperties	Parameters for define LES method
transportProperties	Flow properties
turbulenceProperties	Simulation type definition
Folder	**Description**
polyMesh	Converted files of the mesh

- Gravity, "g"

Table E.8: Description of the file "g".

"g"		
Dimensions	**Unit**	**Value**
[0 1 -2 0 0 0 0]	m/s^2	(0 0 0)

- "LESProperties"
  Is expressed the laminar LESModel and all the coefficients derived to perform the simulation. The file is presented in the appendix F.

- "transportProperties"
  Is expressed an active configuration for phase change and a model to perform it denominated "Zwart". Next is present the different types of it.

Table E.9: Description of "phaseChangeTwoPhaseMixture".

"phaseChangeTwoPhaseMixture"	
Type	**Description**
Zwart	Cavitation model implemented in OpenFOAM with second derivative neglected and non-symmetrical condensation and evaporation model (Hidalgo et al., 2015, p.30)
SchnerrSauer	Cavitation native model in OpenFOAM to study flow with inter- phase change like vapor-liquid water (Hidalgo et al., N/A, p.3)
Kunz	Mass transport cavitation model proposed by Kunz, Native implemented in OpenFOAM to simulate cavitating flows (Hidalgo, Luo, Ji, & Aguinaga, 2014, p.3277)

The complete file "transportProperties" is presented in the appendix F.

- "turbulenceProperties"
  In this file is activated the LESModel for the simulation type, as is described below:

Table E.10: Description of the file "turbulenceProperties".

"turbulenceProperties"	
Type	**Value**
simulationType	LESModel

- polyMesh
  The "boundary" file describe each boundary of the mesh with its type, quantity of faces and start face. The file is automatically generated by the command "gmshToFOAM" but has to be modified to accord with OpenFOAM solver used. This modification is indicated next:

Table E.11: Description of types for boundaries of the file "boundary".

boundary	**type**
INL	patch
OUTL	patch
WING	wall
TOPBOT	wall
FRONT	symmetryPlane
BACK	symmetryPlane

The description of each type and value of the boundaries is mentioned at the end of the appendix.

E.3 DESCRIPTION OF FILES IN FOLDER "system"

The folder denominated "system" is formed by 7 files that are described below:

Table E.12: Description of the files of the folder "system".

File	Description
controlDict	Define the solver, times for the run, time step, type and interval of results write, precisions, formats and other important values.
decomposeParDict	Dictionary that contain information to perform the simulation in parallel (OpenFOAM, 2015, p.66).
forceCoeffs	Function object to calculate forces, Courant number, alpha and coefficients.
fvSchemes	Specification of finite volume discretization schemes (OpenFOAM, 2015, p.23).
fvSolution	Specification of the linear equation solvers and tolerances and other algorithm controls that be used(OpenFOAM, 2015, p.23).
libs	Specification of used libraries
mapFieldsDict	Mapping of the patches of the mesh.

The complete file of "controlDict" is included in the appendix F for better understand. The most important values of "controlDict" are included next:

Table E.13: Description of the file "controlDict".

"controlDict"		
Parameter	**Value**	**Description**
startTime	0	Initial time for the simulation
endTime	0.5	End time for the simulation in [s]
deltaT	1.0e-5	Incremental time for the simulation in [s]
writeInterval	25	Quantity of increments to consider to write the results
writePrecision	6	Number of decimals of the results
timePrecision	6	Number of decimals of time directory names

E.4 DESCRIPTION OF TYPES AND VALUES OF BOUNDARIES

All types and values of all the programming of all boundaries to perform the simulation in OpenFOAM are explained below:

Table E.14: Explanation of types used to perform the simulation.

Type	**Description**
inletOutlet	Switches U and p between fixedValue and zeroGradient depending on direction of U (OpenFOAM, 2015, p.142).
fixedValue	A value for ϕ is specified (OpenFOAM, 2015, p.141).
patch	A condition that contains no geometric or topological information about the mesh (OpenFOAM, 2015, p.139).
slip	zeroGradient if ϕ is a scalar; if ϕ is a vector, normal component is fixedValue zero, tangential components are zeroGradient (OpenFOAM, 2015, p.141).
symmetryPlane	For define a symmetry plane (OpenFOAM, 2015, p.140).
wall	Identification of a wall geometry, the distance from the wall to the cell centers next to the wall are stored as part of the patch (OpenFOAM, 2015, p.139-140).
zeroGradient	Normal gradient of ϕ is zero, the actual value is constant, is "fully developed" (OpenFOAM, 2015, p.141) & (CFD-Online, 2015f) .

Where, ϕ is the specified patch field

Table E.15: Explanation of values used to perform the simulation.

Value	**Description**
uniform (0 0 0)	Makes the value of the type uniform, with a value of zero in all directions.
$internalField	Takes the value specified before in "internalField".

APPENDIX F FILES FROM OPENFOAM

F.1 "LESProperties"

```
/*---------------------------------------------------------------------------*- C++ -*----------------------------------------------------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  1.5                                   |
|   \\  /    A nd           | Web:      http://www.OpenFOAM.org               |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      LESProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

LESModel            laminar;

delta               smooth;

printCoeffs         on;

laminarCoeffs
{
}

oneEqEddyCoeffs
{
    ck               0.07;
    ce               1.05;
}

dynOneEqEddyCoeffs
{
    ce               1.05;
    filter          simple;
}

locDynOneEqEddyCoeffs
{
    ce               1.05;
    filter          simple;
}

```

```
45 SmagorinskyCoeffs
46 {
47     ce                  1.05;
48     ck                  0.07;
49 }
50
51 Smagorinsky2Coeffs
52 {
53     ce                  1.05;
54     ck                  0.07;
55     cD2                 0.02;
56 }
57
58 spectEddyViscCoeffs
59 {
60     ce                  1.05;
61     cB                  8.22;
62     cK1                 0.83;
63     cK2                 1.03;
64     cK3                 4.75;
65     cK4                 2.55;
66 }
67
68 dynSmagorinskyCoeffs
69 {
70     ce                  1.05;
71     filter              simple;
72 }
73
74 mixedSmagorinskyCoeffs
75 {
76     ce                  1.05;
77     ck                  0.07;
78     filter              simple;
79 }
80
81 dynMixedSmagorinskyCoeffs
82 {
83     ce                  1.05;
84     filter              simple;
85 }
86
87 LRRDiffStressCoeffs
88 {
89     ce                  1.05;
90     ck                  0.09;
91     c1                  1.8;
92     c2                  0.6;
93 }
94
95 DeardorffDiffStressCoeffs
96 {
97     ce                  1.05;
98     ck                  0.09;
99     cm                  4.13;
100 }
101
```

```
102 SpalartAllmarasCoeffs
103 {
104     alphaNut          1.5;
105     Cb1               0.1355;
106     Cb2               0.622;
107     Cw2               0.3;
108     Cw3               2;
109     Cv1               7.1;
110     Cv2               5.0;
111     CDES              0.65;
112     ck                0.07;
113 }
114
115 cubeRootVolCoeffs
116 {
117     deltaCoeff        1;
118 }
119
120 PrandtlCoeffs
121 {
122     delta             cubeRootVol;
123     cubeRootVolCoeffs
124     {
125         deltaCoeff        1;
126     }
127     smoothCoeffs
128     {
129         delta             cubeRootVol;
130         cubeRootVolCoeffs
131         {
132             deltaCoeff        1;
133         }
134         maxDeltaRatio     1.1;
135     }
136     Cdelta            0.158;
137 }
138
139 vanDriestCoeffs
140 {
141     delta             cubeRootVol;
142     cubeRootVolCoeffs
143     {
144         deltaCoeff        1;
145     }
146     smoothCoeffs
147     {
148         delta             cubeRootVol;
149         cubeRootVolCoeffs
150         {
151             deltaCoeff        1;
152         }
153         maxDeltaRatio     1.1;
154     }
155     Aplus             26;
156     Cdelta            0.158;
157 }
158
```

```
159 smoothCoeffs
160 {
161     delta            cubeRootVol;
162     cubeRootVolCoeffs
163     {
164         deltaCoeff       1;
165     }
166     maxDeltaRatio    1.1;
167 }
168
169 kappa             0.4187;
170
171 wallFunctionCoeffs
172 {
173     E                 9;
174 }
175
176 //
    ************************************************************************
     //
```

F.2 “transportProperties”

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  1.5                                   |
|   \\  /    A nd           | Web:      http://www.OpenFOAM.org               |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

phaseChange on;

phaseChangeTwoPhaseMixture Zwart;//SchnerrSauer;//Kunz;

//Implemented
pSat             pSat        [1 -1 -2 0 0]     2300;    //Saturation pressure
//Implemented
sigma            sigma [1 0 -2 0 0 0 0] 0.00;

//Anylisis with water
transportModel   Newtonian;
nu               nu [0 2 -1 0 0 0 0] 0.0000148;
rho              rho [1 -3 0 0 0 0 0] 0.023;


ZwartCoeffs
{
    Cc               Cc              [0 0 0 0 0 0 0] 0.03;//morgout default 0.001
    Cv               Cv              [0 0 0 0 0 0 0] 300.0;//morgout defacult 50
    rnuc             rnuc            [0 0 0 0 0 0 0] 5.0e-6;//2e-8; Morgout
    Rb               Rb              [0 1 0 0 0 0 0] 1.0e-6; // from Morgout

cavitation
{
    pSat             pSat            [1 -1 -2 0 0 0 0] 2300;
    restart          no;
    rampN            200;
    startN           1000;
```

```
}
}



SchnerrSauerCoeffs
{
    Cc                  Cc              [0 0 0 0 0 0 0] 2;
    Cv                  Cv              [0 0 0 0 0 0 0] 1;
    n                   n               [0 -3 0 0 0 0 0] 10.0e7;
    dNuc                dNuc            [0 1 0 0 0 0 0] 2.0e-06;//2e-8;


cavitation
{
    pSat                pSat            [1 -1 -2 0 0 0 0] 2300;
    restart             no;
    rampN               200;
    startN              200;
}
}


KunzCoeffs
{
    Cc                  Cc              [0 0 0 0 0 0 0] 1000;
    Cv                  Cv              [0 0 0 0 0 0 0] 10000;
    UInf                UInf            [0 1 -1 0 0 0 0] 5.4772;//6;
    tInf                tInf            [0 0 1 0 0 0 0] 0.1;

cavitation
{
    pSat                pSat            [1 -1 -2 0 0 0 0] 2300;
    restart             no;
    rampN               200;
    startN              1000;
}
}



twoPhase
{
    transportModel  twoPhase;
    phase1          phase1;
    phase2          phase2;
}

phase2
{
    transportModel  Newtonian;
    nu              nu [0 2 -1 0 0 0 0]0.000693;
    rho             rho [1 -3 0 0 0 0 0] 0.01389;
    CrossPowerLawCoeffs
    {
        nu0             nu0 [0 2 -1 0 0 0 0] 1e-06;
        nuInf           nuInf [0 2 -1 0 0 0 0] 1e-06;
```

```
102         m               m [0 0 1 0 0 0 0] 1;
103         n               n [0 0 0 0 0 0 0] 0;
104     }
105     BirdCarreauCoeffs
106     {
107         nu0             nu0 [0 2 -1 0 0 0 0] 0.0142515;
108         nuInf           nuInf [0 2 -1 0 0 0 0] 1e-06;
109         k               k [0 0 1 0 0 0 0] 99.6;
110         n               n [0 0 0 0 0 0 0] 0.1003;
111     }
112 }
113
114 phase1
115 {
116     transportModel  Newtonian;
117     nu              nu [0 2 -1 0 0 0 0] 1.1e-6;
118     rho             rho [1 -3 0 0 0 0 0] 998.85;
119     CrossPowerLawCoeffs
120     {
121         nu0             nu0 [0 2 -1 0 0 0 0] 1e-06;
122         nuInf           nuInf [0 2 -1 0 0 0 0] 1e-06;
123         m               m [0 0 1 0 0 0 0] 1;
124         n               n [0 0 0 0 0 0 0] 0;
125     }
126     BirdCarreauCoeffs
127     {
128         nu0             nu0 [0 2 -1 0 0 0 0] 0.0142515;
129         nuInf           nuInf [0 2 -1 0 0 0 0] 1e-06;
130         k               k [0 0 1 0 0 0 0] 99.6;
131         n               n [0 0 0 0 0 0 0] 0.1003;
132     }
133 }
134
135
136
137 // ************************************************************************* //
```

F.3 "controlDict"

```
/*---------------------------------------------------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  1.1                                   |
|   \\  /    A nd           | Web:      http://www.openfoam.org               |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/

// FoamX Case Dictionary.

FoamFile
{
    version         2.0;
    format          ascii;

    root            "/home/nikwik/OpenFOAM/nikwik-1.1/run/cavInterFoam";
    case            "naca-0012-2d";
    instance        "system";
    local           "";

    class           dictionary;
    object          controlDict;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Foam Application Class
applicationClass interFoam;

// Start point of run
//startFrom        latestTime;
startFrom        startTime;

// Calculation start time
startTime        0;

// End point of run
stopAt           endTime;

// Calculation end time
endTime          0.01;

// Calculation time step
deltaT           1.0e-5;

// Type of write output control
```

```
47 writeControl    timeStep;
48
49 // Interval with which the results are output
50 writeInterval   25;
51
52 // Limits number of time directories before overwriting
53 cycleWrite      0;
54
55 // Write Format
56 writeFormat     ascii;
57
58 // Significant figures of written ASCII data
59 writePrecision  6;
60
61 // Write Compression
62 writeCompression uncompressed;
63
64 // Time directories name format
65 timeFormat      general;
66
67 // Decimal precision of time directory names
68 timePrecision   6;
69
70 // Can parameters be modified during run time?
71 runTimeModifiable yes;
72
73 // Automatic adjustment of time step?
74 adjustTimeStep  off;
75
76 // maxCo
77 maxCo           0.9;
78
79 // maxDeltaT
80 maxDeltaT       1e-3;
81
82 //functions
83
84 #include "libs";
85 #include "forceCoeffs";
86
87 //
    ************************************************************************
     //
```

APPENDIX G MATHEMATICAL RESOLUTION IN FRANCIS-99

G.1 INITIAL VALUES

The obtained values from (NTNU & LTU, 2015a) and (Stoessen, 2014) about the behavior of the flow in the Francis-99 model and prototype are presented below:

Table G.1: Initial operation conditions of Francis-99 model turbine in best efficiency point (BEP).

Parameter	**Symbol**	**Model**
Runner diameter $[m]$	ϕ_1	0.6305
Guide vane structure diameter $[m]$	ϕ_0	0.7664
Guide vane height $[m]$	b_1	0.0596
Net head $[m]$	H_n	11.91
Flow rate $[m^3/s]$	Q	0.203
Runner angular speed $[rpm]$	n	335.4
Guide vane angle $[°]$	α	9.84
Efficiency $[\%]$	η	92.6
Outlet pressure $[kPa]$	p_2	102.53
Tokke ambient temperature $[°C]$	T	22.0
Vapor pressure at 22.0$[°C]$ $[kPa]$	p_v	2.671
Density $[kg/m^3]$	ρ	999.19
Kinematic viscosity $[m^2/s]$	ν	$9.57x10^{-7}$
Chord length $[m]$	c	0.24
Tokke altitude $[m]$	alt	135.0
Tokke latitude $[°]$	lat	59.433

Table G.2: Initial operation conditions of Francis-99 prototype turbine in best efficiency point (BEP).

Parameter	**Symbol**	**Model**
Scale prototype factor	λ	5.0994
Runner angular speed $[rpm]$	n	375.0
Design max power $[MW]$	N	110.0

G.2 MODEL BEHAVIOR

From the equations for turbomachinery from (Fernandez, n.d.).

G.2.1 POWER

$$N_m = \rho * g * Q_m * Hn_m * \eta_m = 21966[W] = 21.97[kW] = 29.87[CV] \tag{G.1}$$

G.2.2 SPECIFIC VELOCITY

$$ns_m = \frac{n_m[rpm] * N_n^{1/2}[CV]}{Hn_m^{5/4}[m]} = 82.84 \tag{G.2}$$

G.2.3 PERIPHERAL VELOCITY

$$u_{1m} = \frac{\pi * \phi_m * n_m[rpm]}{60} = 11.07[m/s] \tag{G.3}$$

G.2.4 MERIDIAN VELOCITY

$$c_{1mm} = \frac{Q_m}{\pi * \phi_m * b_{1m}} = 1.72[m/s] \tag{G.4}$$

G.2.5 ANGLE BETWEEN VELOCITIES

$$\alpha_{u_1c_1} = \alpha_{1m} = cot^{-1}\left(\frac{g * Hn_m * \eta_m}{c_{1mm} * u_{1m}}\right) = 9.97[^\circ] \tag{G.5}$$

G.2.6 ABSOLUTE RUNNER VELOCITY

$$c_{1m} = \frac{c_{1mm}}{sin(\alpha_{1m})} = 9.93[m/s] \tag{G.6}$$

G.2.7 DISCHARGE COEFFICIENT

$$\phi_{dm} = \frac{c_{1m}}{\sqrt{2 * g * Hn_m}} = 0.649 \tag{G.7}$$

G.2.8 GUIDE VANE

- Common factor

$$X_m = \frac{Q_m}{2 * \pi * b_{1m} * sin(\alpha)} \tag{G.8}$$

- Inlet absolute velocity

$$c_{0mgv} = \frac{2}{\phi_{0m}} * X_m = 8.28[m/s] \tag{G.9}$$

- Outlet absolute velocity

$$c_{1mgv} = U_{\infty m} = \frac{2}{\phi_{1m}} * X_m = 10.06[m/s] \tag{G.10}$$

- Runner blade rotation angle

$$\alpha_b = 90 - \alpha = 80.16[^\circ] \tag{G.11}$$

G.2.9 REYNOLDS NUMBER

$$Re_m = \frac{c_{1mgv} * c_m}{\nu} = 2523357 \approx 2.52 \times 10^6 \tag{G.12}$$

G.2.10 CAVITATION NUMBER

$$\sigma_m = \frac{p_2 - p_v}{\frac{1}{2}\rho U_{\infty m}^2} = 1.974 \tag{G.13}$$

G.3 PROTOTYPE BEHAVIOR

From the equations for turbomachinery and its similitude from (Fernandez, n.d.).

G.3.1 LENGTHS

- Runner diameter

$$\phi_p = \phi_m * \lambda = 3.215[m] \tag{G.14}$$

- Guide vane structure diameter

$$\phi_{0p} = \phi_{0m} * \lambda = 3.908[m] \tag{G.15}$$

- Guide vane height

$$b_{1p} = b_{1m} * \lambda = 0.3039[m] \tag{G.16}$$

- Chord length

$$c_p = c_m * \lambda = 1.224[m] \tag{G.17}$$

G.3.2 SPECIFIC VELOCITY

$$ns_p = ns_m = 82.84 \tag{G.18}$$

G.3.3 NET HEAD

$$Hn_p = \left(\frac{n_p}{ns_p} * \sqrt{N_m[CV] * \lambda^2} * \frac{1}{Hn_m^{3/4}}\right)^2 = 387.2[m] \tag{G.19}$$

G.3.4 POWER

$$N_p = N_m[kW] * \lambda^2 * \left(\frac{Hn_p}{Hn_m}\right)^{3/2} = 105869[kW] = 105.9[MW] = 143942[CV] \tag{G.20}$$

G.3.5 FLOW RATE

$$Q_p = Q_m * \lambda^2 * \sqrt{\frac{Hn_p}{Hn_m}} = 30.1[m^3/s] \tag{G.21}$$

G.3.6 PERIPHERAL VELOCITY

$$u_{1p} = \frac{\pi * \phi_p * n_p[rpm]}{60} = 63.13[m/s] \tag{G.22}$$

G.3.7 MERIDIAN VELOCITY

$$c_{1mp} = \frac{Q_p}{\pi * \phi_p * b_{1p}} = 9.8[m/s] \tag{G.23}$$

G.3.8 ANGLE BETWEEN VELOCITIES

$$\alpha_{u_1c_1} = \alpha_{1p} = cot^{-1}\left(\frac{g * Hn_p * \eta_m}{c_{1mp} * u_{1p}}\right) = 9.97[^\circ] \tag{G.24}$$

G.3.9 ABSOLUTE RUNNER VELOCITY

$$c_{1p} = \frac{c_{1mp}}{sin(\alpha_{1p})} = 56.62[m/s] \tag{G.25}$$

G.3.10 DISCHARGE COEFFICIENT

$$\phi_{dp} = \frac{c_{1p}}{\sqrt{2 * g * Hn_p}} = 0.649 \tag{G.26}$$

G.3.11 GUIDE VANE

- Common factor

$$X_p = \frac{Q_p}{2 * \pi * b_{1p} * sin(\alpha)} \tag{G.27}$$

- Inlet absolute velocity

$$c_{0pgv} = \frac{2}{\phi_{0p}} * X_p = 47.2[m/s] \tag{G.28}$$

- Inlet radial velocity

$$c_{0rpgv} = \frac{2}{\phi_{0p}} * X_p * sin(\alpha) = 8.066[m/s] \tag{G.29}$$

- Inlet normal velocity

$$c_{0npgv} = \frac{2}{\phi_{0p}} * X_p * \sqrt{1 - sin^2(\alpha)} = 46.501[m/s] \tag{G.30}$$

- Reference rotation angle

$$\theta = 9.85[^\circ] \tag{G.31}$$

  This is visualized in the graphic 2.36.

- Inlet radial velocity components

$$c_{0rpgvx} = C_{0rpgv} * sin(\theta) = 1.38i[m/s] \tag{G.32}$$

$$c_{0rpgvy} = C_{0rpgv} * cos(\theta) = -7.95j[m/s] \tag{G.33}$$

- Inlet normal velocity components

$$c_{0npgvx} = C_{0npgv} * cos(\theta) = 45.82i[m/s] \tag{G.34}$$

$$c_{0npgvy} = C_{0npgv} * sin(\theta) = 7.95j[m/s] \tag{G.35}$$

- Inlet absolute velocity components

$$c_{0pgvx} = c_{0rpgvx} + c_{0npgvx} = 47.2i[m/s] \tag{G.36}$$

$$c_{0pgvy} = c_{0rpgvy} + c_{0npgvy} = 0.0j[m/s] \tag{G.37}$$

$$c_{0pgv} = 47.2i + 0.0j[m/s] = 47.2i[m/s] \tag{G.38}$$

- Outlet absolute velocity

$$c_{1pgv} = U_{\infty p} = \frac{2}{\phi_{1p}} * X_p = 57.37[m/s] \tag{G.39}$$

G.3.12 REYNOLDS NUMBER

$$Re_p = \frac{c_{1pgv} * c_p}{\nu} = 73365466 \approx 73.4 \times 10^6 \tag{G.40}$$

G.3.13 CAVITATION NUMBER

$$\sigma_p = \frac{p_2 - p_v}{\frac{1}{2}\rho U_{\infty p}^2} = 0.061 \tag{G.41}$$

G.4 OTHERS

G.4.1 GRAVITY

From (IEC, 1999, p.165):

$$g = 9.7803 * (1 + 0.0053 sin^2(lat) - 3 * 10^{-6} alt = 9.818[m/s^2] \tag{G.42}$$