UNIVERSITA' DI PADOVA
FACOLTA' DI INGEGNERIA

SCUOLA DI DOTTORATO DI RICERCA IN INGEGNERIA INDUSTRIALE

INDIRIZZO ENERGETICA

CICLO: XXIV

Aerodynamic shape optimization of rotary wing aircraft components using advanced multiobjective evolutionary algorithms

DIRETTORE DELLA SCUOLA:
Prof. Alberto Mirandola

SUPERVISORE:
Prof. Ernesto Benini

DOTTORANDO: Claudio Comis Da Ronco

31 gennaio 2012

A chi mi vuole bene

Sommario

Lo scopo di questa tesi di Dottorato in Energetica, finanziata da AgustaWestland, consiste nella progettazione e sviluppo di una procedura di ottimizzazione multi-obiettivo, che comprende l'applicazione del GeDEA-II, un algoritmo genetico/evolutivo recentemente sviluppato dall'autore presso l'Università di Padova. Tale algoritmo permette di effettuare analisi di ottimizzazione multi-obiettivo, sfruttando l'approccio del tutto generale che va sotto il nome di "Ricerca del Fronte di Pareto". Rispetto ad altri algoritmi evolutivi multi-obiettivo "state-of-the-art", esso presenta operatori di crossover e mutazione innovativi, che ne migliorano in maniera significativa le performance.

Questo ottimizzatore è accoppiato con i codici CFD commerciali e gratuiti, rispettivamente Fluent® and OpenFOAM®. Il pacchetto Altair Hyperworks, codice ufficiale presso AgustaWestland per il pre-processing di fusoliere di elicottero, è scelto quale software per la parametrizzazione *free-form*.

I casi test scelti per dimostrare l'efficacia di tale procedura consistono nell'ottimizzazione aerodinamica della regione frontale del tilt rotor dimostrativo ERICA, e della presa d'aria ♯1 dell'elicottero AW101. Tali casi costituiscono problemi stimolanti sia da un punto di vista puramente ingegneristico, sia da un punto di vista industriale. A partire dall'elaborazione della geometria, e procedendo con la discussione dei risultati ottenuti, ogni passo della procedura di ottimizzazione è descritto in dettaglio, con particolare enfasi dedicata al ciclo di ottimizzazione, sviluppato dall'autore sia in ambiente UNIX, sia in ambiente Windows.

I risultati ottenuti dimostrano l'efficacia dell'approccio di ottimizzazione basato sull'algoritmo GeDEAII, scelto per sviluppare questo lavoro.

Inoltre, vengono forniti alcuni consigli inerenti lo sviluppo futuro di questa procedura di ottimizzazione, con l'obiettivo di migliorare ulteriormente le capacità e la robustezza del ciclo di ottimizzazione.

Abstract

The aim of this Doctoral Thesis, sponsored by AgustaWestland, is the design and development of a multi-objective optimization procedure that involves the application of the GeDEA-II, a powerful and time-saving evolutionary algorithm recently developed by the author at the University of Padova, able to perform multi-objective optimization analyses with the general approach of the Pareto frontier search. When compared to other state-of-the-art multi-objective evolutionary algorithms, it features novel crossover and mutation operators, and demonstrated superior performance.

This optimizer supervises an automatic optimization loop involving the CFD commercial and free, open source solvers, respectively Fluent® and OpenFOAM®. Altair Hyperworks package is chosen as the free-form-deformation parameterization engine.

The test cases chosen to demonstrate the strength of the procedure implemented concern the aerodynamic optimization of the AgustaWestland ERICA nose region, and the optimization of the intake ♯1 of the AW101 helicopter, that is really challenging problems from both the engineering and the industrial point of view.

Starting from the the geometry elaboration and proceeding to the results discussion, each step of the optimization procedure is described in details, with particular focus on the automatic optimization loop, directly programmed by the author in both UNIX/Linux and Windows environments.

The results obtained surely demonstrate the effectiveness of the multi-objective approach chosen to carry out this work.

Furthermore, some suggestions for future improvements and developments are provided, with the purpose to increase the strength of the discussed multi-objective optimization tool.

Why research?

When I first started my PhD three years ago I must admit I was a bit skeptical about my future. I don't know what I will do in my future but I certainly believe that professional, international experience of a high level is essential to be able to complete my accademic studies. In any case, three years later as a result this PhD course has given me much personal satisfaction.

Why embark on a PhD research programme? Why do research? Well I believe what makes human beings different from animals isn't either their soul or their intelligence but what makes man unique is his knowledge and the desire to push himself ahead beyond his limits.

Giacomo Leopardi wrote between 1829 and 1830 *"Night Song of a Wandering Shepherd in Asia"*, and at one point he says:

And when I gaze upon the stars at night,
In thought I ask myself,
"Why all these torches bright?
What mean these depths of air,
This vast, this silent sky,
This nightly solitude? And what am I?"

Gazing towards the sky the poet contemplates on the meaning of life and looks for an answer to his question : "what is life"?

I think that researchers should draw inspiration from Leopardi's poems. "Striving for the infinite" is one of the main trains of thought of the Romantic poets and this is what should distinguish every researcher: push yourself and go beyond your limits.

So what could be more wonderful and inspiring than doing research? Of course it means making a lot of sacrifices and getting discouraged at times, but is there anything more humane than doing research? I don't think so, I believe it's important not to stifle one's ambitions but keep the flame alive.

In conclusion I urge whoever reads this Thesis to reflect upon my words.

The future of research programmes in Italy at the moment is undergoing some dark moments. Without Universities and Research programmes hypothetically speaking we would all become like a bunch of trained circus animals. I think a great country should be made up of well-educated people with strong critical and thinking skills in order to make the right decisions for themselves about their future.

This country must invest in research to keep up the standard of research programmes: you can't put a price on knowledge and learning.

As far as I'm concerned I intend carrying on with this belief and heaven help those who want to prevent me from pursuing my dreams!

Long-live Research! Long-live Italy!

Contents

List of Figures

List of Tables

Acronyms

AIP	Aerodynamic Interface Plane.
$DC(60)$	Distortion Coefficient calculated assuming a 60 ° angular sector.
D.O.E.	Design Of Experiments.
EP	Evolutionary Programming.
ES	Evolution Strategies.
FUSELAGE INCIDENCE	Angle between Fuselage Water Line and the free-stream wind direction [Positive nose-up].
GeDEAII	Genetic Diversity Evolutionary Algorithm II.
GeDEM	Genetic Diversity Evaluation Method.
HPC	High Performance Computing.
MOEA	Multi Objective Evolutionary Algorithm.
MOOP	Multi Objective Optimization Problem.
OAT	Outside Air Temperature.
SQP	Sequential Quadratic Programming.

Introduction

Aviation is an essential element of today's global society, bringing people and cultures together and creating economic growth. The air transport industry is paying a lot of attention to growing public concern about the environmental issues of air pollution, noise and climate change. Although today air transport only produces 2% of man-made CO_2 emissions, this is expected to increase to 3% by 2050.

Clean Sky[1] is a *Joint Technology Initiative* (JTI) that will develop breakthrough technologies to reduce aviation environmental impact. The Clean Sky JTI will be one of the largest European research projects ever, with a budget estimated at 1.6 billion euros, equally shared between the European Commission and industry, over the period 2008 - 2013. This public-private partnership will speed up technological breakthrough developments and shorten the time to market for new solutions tested on Full Scale Demonstrators. The accelerated research process that Clean Sky offers represents an unprecedented opportunity for rapid progress in the introduction of green technology into aviation. The Clean Sky JTI gathers 86 organization over 16 countries:

- 54 industries, including AgustaWestland S.p.A, which, as leader of the European helicopter market, is involved in the *Green Rotocraft* ITD (Integrated Technology Demonstrator) project;
- 15 research centers;
- 17 Universities, including University of Padova as AgustaWestland partner.

One of the goals of the Clean Sky JTI is the 50% reduction of CO_2 emission through drastic reduction of aircraft fuel consumption, that means, from the aeronautic industries point of view, to adapt the aircraft and rotorcraft design procedures to new severe standards about aerodynamic drag and engines efficiency.

Organized around six major platforms or *Integrated Technology Demonstrators* (ITDs), the Clean Sky project will lead to the development of in-flight or ground demonstrators. Among these ITDs, the Green Rotorcraft platform is specifically devoted to helicopters and tilt-rotor aircrafts, whose operations are expected to grow sharply in the future to meet the increasing demands of European citizens:

- Medical service for safe and quick transport of patients and living organs needed for transplantation;

[1]Please visit the web site `http://www.cleansky.eu` for more elucidations.

- Passenger transport from city heliports to airports, and also between cities or areas where an efficient surface transport network cannot be developed for geographical or economical reasons;
- Search, rescue, police and border patrol missions.

In turn, within Green Rotorcraft ITD, six technological projects have been defined and divided between the two leaders AgustaWestland and Eurocopter.

The first project (GRC1) focuses on blades, where the optimization of shapes and the use of active control systems are intended to reduce noise and consumption.

The second project (GRC2) concerns airframe design, which must be made more aerodynamic for more efficient power use in flight. Depending on the rotorcraft configuration and weight class the aerodynamic cumulated drag benefits are currently estimated in the range $10 - 15\%$ which would translate into a 4 to 5% fuel consumption reduction for the same payload and mission.

The third project (GRC3) covers the integration of innovative electrical systems that will eliminate engine air bleeds and the need for hydraulic fluid.

The fourth project (GRC4) examines the integration of diesel engines on light helicopters and will eventually lead to the production and flight testing of a demonstrator.

The fifth (GRC5) is devoted to aeroacoustics research.

The sixth (GRC6) but by no means least important project tackles eco-design, an area that is of equal interest to airplane manufacturers. Nevertheless, the Eco-Design ITD also offers certain specificities for helicopter manufacturers, such as the surface treatment of dynamic components in the power transmission system. These six projects will be followed by the synthesis work that will technologically assess the solutions after integration.

The subject of this Thesis deals with the second Green Rotorcraft project (GRC2), since the content of the research illustrated here aims at gaining a deeper insight into the computer-based optimization procedure of helicopters and tilt rotor airframe components.

In this context, AgustaWestland and the University of Padova are focusing their attentions on the opportunity to integrate the classical trial and error design approach with a more advanced approach, based on the automatic search of optimal solutions using optimization algorithms.

The present Thesis becomes part of the AgustaWestland/University of Padova aerodynamic optimization research program, with the purpose to build up an automatic optimization procedure able to deal with mono and multi-objective aerodynamic, but even multi-disciplinary, optimization problems, in order to create an optimization tool able to drastically improve the aerodynamic characteristics of fuselage component under investigation. The AgustaWestland requirements for this optimization tool are the following:

- It has to include the softwares at the moment available at AgustaWestland, regarding CAD geometry elaboration (CATIA), mesh generation (HyperMesh) and parametrization (HyperMorph), and advanced aerodynamic analysis calculation (Fluent® and/or OpenFOAM®);

- It has to be able to run also on high computational power machines, like the UNIX/Linux based cluster computers[2];
- It has to enable the user to treat not only single-objective, but also multi-objective optimization problems.

The last item of the previous list, enforces the choice of an intrinsically multi-objective optimization algorithm, like the home-made algorithm developed during my PhD course, i.e. the GeDEA-II, as optimization engine. The central point of the present thesis is, therefore, to implement the proper interface among the optimization engine GeDEAII, the aerodynamic solvers (Fluent® and/or OpenFOAM®), and parameterization tool Hypermorph, all of them capable of running on both Windows-based workstations and Linux/UNIX-based clusters.

To prove the applicability of that optimization procedure on real engineering problems, three test problems were addressed.

The aerodynamic optimization of the nose region, belonging to the AgustaWestland new tilt-rotor concept ERICA [37], has been carried out, demonstrating that such kind of parametric analysis may become a really useful design tool in the future. To perform this work, both Fluent® and OpenFOAM® CFD solvers where exploited. Related test cases are Test Case B and Test Case C, respectively, presented in Chapter 7.

Moreover, the aerodynamic optimization of the AW101 left air intake system has been performed, in order to prove further the versatility and robustness of the optimization loop, and presented within Test case A of Chapter 7.

The Thesis is structured as follows.

Chapter 1 provides all the theory knowledge about optimization methods.

GeDEAII algorithm is presented in *Chapter 2*, with a particular emphasis on its framework, its operators and its performance measured against some state-of-the-art algorithms, on state-of-the-art benchmark problems.

Morphing technologies are introduced in *Chapter 3*.

A general introduction to CFD is then presented in *Chapter 4*.

Chapter 5 gives general informations about aircraft external aerodynamics, along with the most promising techniques aiming at reducing aircraft overall drag.

Chapter 6 constitutes a brief introduction to the intake aerodynamics, in order to prepare the ground for the optimization problem regarding the AW101 left air-intake system discussed in Part I of Chapter 7.

Chapter 7 represents the main segment of the Thesis, where the optimization procedure and its application to the AW101 left intake and ERICA nose region are wholly described.

Finally the conclusions which can be drawn from the whole work are presented, along with some suggestions for future works and developments.

[2]A computer cluster is a group of linked computers, working together closely so that in many respects they form a single computer. The components of a cluster are commonly, but not always, connected to each other through fast local area networks. Clusters are usually deployed to improve performance and/or availability over that of a single computer, while typically being much more cost-effective than single computers of comparable speed or availability

Chapter 1

Optimization techniques and Evolutionary algorithms

Most of the real world system models involve nonlinear optimization with complicated objective functions or constraints for which analytical solutions (solutions using quadratic programming, geometric programming, etc.) are not available. In such cases one of the possible solutions is the search algorithm in which, the objective function is first computed with a trial solution and then the solution is sequentially improved based on the corresponding objective function value till convergence. A generalized flowchart of the search algorithm in solving a nonlinear optimization with decision variable X_i, is presented in Figure 1.1.

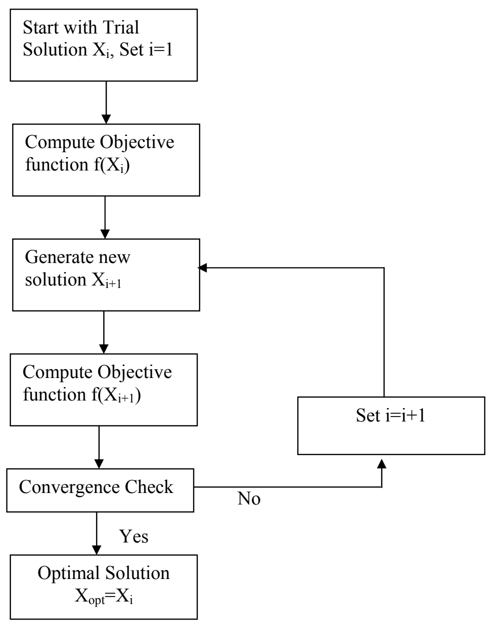

Start with Trial Solution X_i, Set i=1
Compute Objective function $f(X_i)$
Generate new solution X_{i+1}
Compute Objective function $f(X_{i+1})$
Set i=i+1
Convergence Check
No
Yes
Optimal Solution $X_{opt}=X_i$

Figure 1.1: Flowchart of Search Algorithm.

Optimization methods can be basically classified into three main categories, based on the approach followed to reach the optimal solution:

1. Direct or derivative-free methods;
2. Classical, gradient-based, methods;
3. Stochastic methods.

A *direct search* algorithm for numerical search optimization depends on the objective function only through ranking a countable set of function values. It does not involve the partial derivatives of the function and hence it is also called non-gradient or zero$_{th}$ order method.

Gradient-based algorithms, also called the *descent methods*, depend on the first (first-order methods) and often second derivatives (second-order methods) of the objective function.

Stochastic methods are optimization algorithms which incorporate probabilistic (random) elements, either in the problem data (the objective function, the constraints, etc.), or in the algorithm itself (through random parameter values, random choices, etc.), or in both [83]. The concept contrasts with the deterministic optimization methods, where the values of the objective function are assumed to be exact, and the computation is completely determined by the values sampled so far. Stochastic optimization methods include:

- Evolutionary algorithms [39];
- Simulated annealing algorithms [51];
- Particle Swarm algorithms [50];

In this chapter a short introduction to general optimization problems formulation is provided, including a discussion about the *Direct methods*. Solving strategies for both unconstrained and constrained optimizations using some classical, *gradient based*, techniques are then briefly introduced while the description of *stochastic methods* is focused on genetic algorithms, since it is the method chosen to perform the aerodynamic shaoe optimizations presented in Chapter 7. In the last part of the chapter an introduction to multi-objective optimization theory is also provided, with particular focus on the *Genetic Diversity Evolutionary Method* (GeDEM) that is implemented into the multi-objective optimization code *Genetic Diversity Evolutionary AlgorithmII* (GeDEAII) used to perform the work described in the following chapters.

1.1 Optimization strategies: An overview

Optimization techniques are used to find a set of design parameters $x = \{x_1, x_2, \ldots x_n\}$, that can in some way optimize a given function relative to a design problem. In a simple case this may be the minimization or maximization of some system characteristic that is dependent on x. In a more advanced formulation, the optimization problem of the objective function $f(x)$, to be minimized or maximized, can be stated as in Equation 1.1 (n representing the number of decision variables):

$$\min_{x \in \mathbb{R}^n} f(x) \tag{1.1}$$

subject to the constraints in the form of:

- equality constraints, G_i(x) = 0 (i = 1,. . . ,m_e);
- inequality constraints, G_i(x)$\leq$ 0 (m_e+1,. . . ,m);
- parameter bounds like $x_l \leq 0\ x \leq x_u$, where x_l is the parameters lower bound and x_l is the parameters upper bound respectively.

where x is the vector of the design variables, $x \in \mathbb{R}^n$, f(x) is the objective function that returns a scalar value, in mono-objective optimization cases $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$, or a vector of p elements where p is the number of objectives in the multi-objective problem considered, f(x):$\mathbb{R}^n \rightarrow \mathbb{R}^p$), and the vector function $G(x)$ returns the values of the equality and inequality constraints evaluated at the actual value of the design variables x, G(x):$\mathbb{R}^n \rightarrow \mathbb{R}^m$. An efficient and accurate solution to this problem is not only dependent on the size of the problem in term of design variables and constraints, but also on characteristics of the objective and constraint functions. When both the objective function and the constraints are linear function of the design parameters the problem is know as a Linear Programming problem (LP). Quadratic Programming (QP) concerns the minimization or maximization of a quadratic objective function that is linearly constrained. For both these kinds of problems, reliable solution procedures are readily available. More complicated to solve is the Non-linear Programming problem (NP) where both the objective function and constraints are non-linear function of the design parameters. Generally a solution of the non-linear problem requires an iterative procedure to establish a search direction at each iteration; this is usually achieved by the solution of an LP, a QP, or an unconstrained subproblem.

1.2 Direct search methods

A lot of problems derived from real applications are characterized by an unknown analytical expression of both the objective function and constraints. This situation occurs, for example, when complex physic systems are described, analyzed and controlled optimizing the results of computer based simulations, and the objective function and/or constraints values are evaluated by post-processing the results of these simulations. This is also the case of the application studied in this Thesis. It is clear that in these situations the gradient based approach may not to be suitable, therefore it is of practical interest the development of optimization methods which do not require derivative informations. *Direct search* or *Derivative-free* is the name of a category of optimization methods which respond to this requirement. As opposed to the more traditional methods discussed from section

References, which use informations about the gradient and higher derivatives to search for an optimal solution, a direct search algorithm searches a set of points around the current point , looking for one where the value of the objective function is lower than the value at the current point. *Direct search methods* are suitable to solve problems for which the objective function is not differentiable, stochastic, or even discontinuous.

Some of the direct search algorithms for solving nonlinear optimization, which requires objective functions, are described below:

1. **Random Search Method**: This method generates trial solutions for the optimization model using random number generators for the decision variables. Random search method includes random jump method, random walk method and random walk method with direction exploitation. Random jump method generates huge number of data points for the decision variable assuming a uniform distribution for them and finds out the best solution by comparing the corresponding objective function values. Random walk method generates trial solution with sequential improvements which is governed by a scalar step length and a unit random vector. The random walk method with direct exploitation is an improved version of random walk method, in which, first the successful direction of generating trial solutions is found out and then maximum possible steps are taken along this successful direction.

2. **Grid Search Method**: This methodology involves setting up of grids in the decision space and evaluating the values of the objective function at each grid point. The point which corresponds to the best value of the objective function is considered to be the optimum solution. A major drawback of this methodology is that the number of grid points increases exponentially with the number of decision variables, which makes the method computationally costlier.

3. **Univariate Method**: This procedure involves generation of trial solutions for one decision variable at a time, keeping all the others fixed. Thus the best solution for a decision variable keeping others constant can be obtained. After completion of the process with all the decision variables, the algorithm is repeated till convergence.

4. **Pattern Directions**: In univariate method the search direction is along the direction of coordinate axis which makes the rate of convergence very slow. To overcome this drawback, the method of pattern direction is used, in which, the search is performed not along the direction of the coordinate axes but along the direction towards the best solution. This can be achieved with *Hooke and Jeeves'*method or *Powell*'s method. In the Hooke and Jeeves'-method [46], a sequential technique is used consisting of two moves: *exploratory move* and the *pattern move*. Exploratory move is used to explore the local behavior of the objective function, and the pattern move is used to take advantage

of the pattern direction. Powell's method [69] is a direct search method with conjugate gradient, which minimizes the quadratic function in a finite number of steps. Since a general nonlinear function can be approximated reasonably well by a quadratic function, conjugate gradient minimizes the computational time to convergence.

5. **Rosen Brock's Method of Rotating Coordinates**: This method [76] is a modified version of Hooke and Jeeves'method, in which, the coordinate system is rotated in such a way that the first axis always orients to the locally estimated direction of the best solution and all the axes are made mutually orthogonal and normal to the first one.

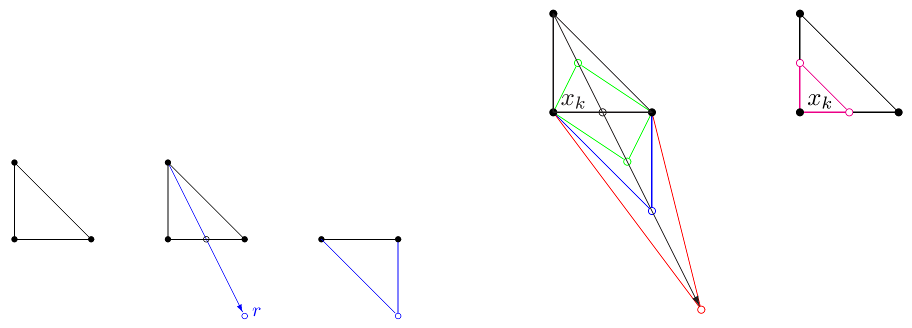
6. **Simplex Method**: Simplex method is a conventional direct search algorithm where the best solution lies on the vertices of a geometric figure in N-dimensional space made of a set of N+1 points. The method compares the objective function values at the N+1 vertices and moves towards the optimum point iteratively. The movement of the simplex algorithm is achieved by reflection, contraction and expansion.

Due to its semplicity, easy-to-use, and because of its popularity, the latter method is briefly introduced in the next section.

1.2.1 The Simplex Algorithm

Simplex search methods are characterized by the simple device that they use to guide the search. The first of the simplex methods is due to Spendley, Hext, and Himsworth [84] in a paper that appeared in 1962. They were motivated by the fact that earlier direct search methods required anywhere from $2n$ to 2^n objective evaluations to complete the search for improvement on the iterate. Their observation was that it should take no more than $n + 1$ values of the objective to identify a downhill (or uphill) direction. This makes sense, since $n + 1$ points in the graph of f(x) determine a plane, and $n + 1$ values of f(x) would be needed to estimate f(x) via finite-differences. At the same time, $n + 1$ points determine a simplex. This leads to the basic idea of simplex search: construct a non-degenerate simplex in $\mathbb{R}^n$ and use the simplex to drive the search. A simplex is a set of n + 1 points in $\mathbb{R}^n$. Thus one has a triangle in $\mathbb{R}^2$, and tetrahedron in $\mathbb{R}^3$, etc. A non-degenerate simplex is one for which the set of edges adjacent to any vertex in the simplex forms a basis for the space. In other words, we want to be sure that any point in the domain of the search can be constructed by taking linear combinations of the edges adjacent to any given vertex. Not only does the simplex provide a frugal design for sampling the space, it has the added feature that if one replaces a vertex by reflecting it through the centroid of the opposite face, then the result is also a simplex, as shown in Figure 1.2 a).

Once an initial simplex is constructed, the single move specified in the original Spendley, Hext, and Himsworth simplex algorithm is that of *reflection*. This move first identifies the "worst" vertex in the simplex (i.e., the one with the least desirable



(a) Reflection move (b) Nelder and Mead's additional moves

Figure 1.2: The original simplex, the *r*eflection of one vertex through the centroid of the opposite face, and the resulting reflection simplex.

objective value, that is the one with the highest value in the case of a minimization problem, or the one with the smallest one in the case of a maximizing problem) and then reflects the worst simplex through the centroid of the opposite face. The new formed simplex is now subject to a new process.

The contribution of Nelder and Mead [64] was to turn simplex search into an optimization algorithm with additional moves designed to accelerate the search. In particular, it was already well-understood that the reflection move preserved the original shape of the simplex, regardless of the dimension. What Nelder and Mead proposed was to supplement the basic reflection move with additional options designed to accelerate the search by deforming the simplex in a way that they suggested would better adapt to the features of the objective function. To this end, they added what are known as *expansion* and *internal* and *external contraction* moves, as shown in Figure 1.2 b). In addition to these improvements, Nelder and Mead also resolved the question of what to do if none of the steps tried bring acceptable improvement by adding a shrink step: when all else fails, reduce the lengths of the edges adjacent to the current best vertex by half, as is also illustrated on the right side of Figure 1.2 b).

1.3 Gradient-based Methods for Unconstrained Problems

The *derivative-free*, or *gradient-based* search algorithms are based on the derivatives or gradients of the objective function. The gradient of a function in *n*-

dimensional space is given by:

$$\nabla f = \begin{bmatrix} \partial f/\partial x_1 \\ \partial f/\partial x_2 \\ . \\ . \\ . \\ \partial f/\partial x_n \end{bmatrix} \tag{1.2}$$

Gradient-based search algorithms include:

1. **Steepest Descent (Cauchy) Method**: In this method, the search starts from an initial trial point X_1, and iteratively moves along the steepest descent directions until the optimum point is found. Although the method is straightforward, it is not applicable to the problems having multiple local optima. In such cases the solution may get stuck at local optimum points.

2. **Conjugate Gradient (Fletcher-Reeves) Method**: The convergence technique of the steepest descent method can be greatly improved by using the concept of conjugate gradient [29] with the use of the property of quadratic convergence.

3. **Newton's Method**: Newton's method is a very popular method which is based on Taylor's series expansion. The Taylor's series expansion of a function $f(x)$ at x=x_i is given by (indicating with T the transposed matrix in all of the following equations):

$$f(x) = f(x_i) + \nabla f_i^T(x - x_i) + \frac{1}{2}(x - x_i)^T[J_i](x - x_i) \tag{1.3}$$

where $[J_i] = [J]|x_i$, is the Hessian matrix of f evaluated at the point x_i. Setting the partial derivatives of Eq. 1.3, to zero, the minimum value of $f(x)$ can be obtained.

$$\frac{\partial f(x)}{\partial x_j} = 0 \quad for\ j = 1, \ldots, N \tag{1.4}$$

From Eq. 1.3 and 1.4

$$\nabla f = \nabla f_i + [J_i](x - x_i) = 0 \tag{1.5}$$

Eq. 1.5 can be solved to obtain an improved solution $x_i + 1$

$$x_{i+1} = x_i - [J_i]^{-1}\nabla f_i \tag{1.6}$$

The procedure is repeated till convergence for finding out the optimal solution.

4. **Marquardt Method**: Marquardt method is a combination method of both the steepest descent algorithm and Newton's method, which has the advantages of both the methods, movement of function value towards optimum point and fast convergence rate. By modifying the diagonal elements of the Hessian matrix iteratively, the optimum solution is obtained in this method.

5. **Quasi-Newton Method**: Quasi-Newton methods are well-known algorithms for finding maxima and minima of nonlinear functions. They are based on Newton's method, but they approximate the Hessian matrix, or its inverse, in order to reduce the amount of computation per iteration. The Hessian matrix is updated using the secant equation, a generalization of the secant method for multidimensional problems. For completeness of informations, the following subsection will be entirely dedicated to this powerful method.

1.3.1 The Quasi-Newton Method

This method builds up curvature information at each iteration to formulate a quadratic model problem of the form:

$$\min_x \frac{1}{2} x^T H x + c^T x + b \tag{1.7}$$

where H, the Hessian matrix, is a positive definite matrix, c is a constant vector and b is a constant. The optimal solution for the problem occurs when the partial derivative of x go to zero:

$$\nabla f x^* = H x^* + c = 0 \tag{1.8}$$

x* being the optimal solution point, which can be written as:

$$x^* = -H^{-1} c = 0 \tag{1.9}$$

Unlike this method, Newton-type methods, opposed to Quasi-Newton methods, calculate H directly and proceed in a direction of descent to locate the minimum after a number of iterations. To avoid the direct evaluation of H, that requires a grate amount of computation, Quasi-Newton methods use the observed behavior of $f(x)$ and $\nabla f(x)$ to build up a curvature information to make an approximation to H using an appropriate update technique. Generally the updating formula given by *Broyden*, *Fletcher*, *GoldFarb*, and *Shanno* (BFGS) is thought to be the most effective for the use in general purpose methods. The BFGS formula is presented in Eq. 1.10:

$$H_{k+1} = H_k + \frac{q_x q_x^T}{q_x^T s_k} - \frac{H_k^T s_k^T s_k H_k}{s_k^T H_k s_k} \tag{1.10}$$

where:

$$s_k = x_{k+1} - x_k$$
$$q_k = \nabla f(x_{k+1}) - \nabla f(x_k)$$

As starting point, H_0 can be set to any symmetric positive definite matrix. To avoid the inversion of the Hessian, one can use a formula that makes an approximation of the inverse Hessian at each update; a well know procedure of this kind is the DFP formula of *Davidon*, *Fletcher* and *Powell*. This uses the same formula as the BFGS method (Eq.1.10) except that q_k is substituted for s_k . The gradient information is either supplied through analytically calculated gradients, or derived by partial derivatives using a numerical differentiation method via finite differences. This involves perturbing each of the design variables in turn and calculating the rate of change in the objective function. At each major iteration, k, a line search is performed in the following direction:

$$d = -H_k^{-1}\nabla f(x_k) \tag{1.11}$$

with the purpose to update the solution to:

$$x_{k+1} = x_k + \alpha_k d \tag{1.12}$$

in the order to meet the minimization condition f(x_{k+1}) f(x_k), where α_k is the line search step evaluated with a line search technique [101].

1.4 Gradient-based Methods for Constrained Problems

In constrained optimization the general aim is to transform the problem into an easier subproblem that can be solved and used as the basis of an iterative process. Today the most widely used methods are all based on the solution of the *Karush-Kuhn-Tucker* equations (KKT) which are necessary conditions for optimality for a constrained optimization problem. The description of the KKT conditions can be started with some simple definitions:

Convex programming problem: Problem in which both the objective function and the inequality constraints are convex and the equality constraints are linear functions of the design variables; in this category we can find both the LP and QP problem;

Feasible point: Point that satisfies all the constraints;

Active constraint: Inequality constraint that is characterized by a null value in a feasible point. By definition all the equal constraints are active.

It can be demonstrated that, if a given problem is a so called convex programming problem, the KKT conditions are both necessary and sufficient for a global solution point. Referring to the general problem stated in Eq.1.1, the KKT conditions can be written as follows:

$$\begin{aligned}
\nabla f(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla G_i(x^*) &= 0 \\
G_i(x^*) &= 0 (i = 1, \ldots, m_e) \\
G_i(x^*) &\leq 0 (i = m_e + 1, \ldots, m) \\
\lambda_i^* &> 0 \\
\sum_{i=1}^{m} \lambda_i^* G_i(x^*) &= 0
\end{aligned} \tag{1.13}$$

The first equation describes a cancelling of the gradient between the objective function and the active constraints at the solution point. For the gradient to be cancelled, Lagrangian multipliers (λ_i = 1,. . .,m) are necessary to balance the magnitude deviation of the objective function and the constraints gradients. Since only active constraints are included in this cancelling operation, constraints that are not active not active must not be included in this operation and be set with a corresponding Lagrangian multiplier equal to zero. This is stated implicitly in the last two equations in Eq. 1.13.

The solution of the KKT equations forms the basis to many nonlinear programming algorithms. These algorithms attempt to compute directly the Lagrange multipliers. Constrained Quasi-Newton methods guarantee super-linear convergence by accumulating second order information regarding the KKT equations using a Quasi-Newton updating procedure. These method are commonly referred to as *S*equential Quadratic Programming (SQP) methods since a QP subproblem is solved at each major iteration. These methods are briefly introduced in the following section.

1.4.1 Sequential Quadratic Programming

Given the general problem in Eq. 1.1, the principal idea of the SQP concept is the formulation of a QP subproblem based on a quadratic approximation of the Lagrangian function:

$$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i G_i(x) \tag{1.14}$$

The QP subproblem is obtained by linearizing the nonlinear constraints. It can be written as follows:

$$\begin{gathered} \min_{d \in \Re^n} \frac{1}{2} d^T H_k d + (x_k)^T d \\ \nabla G_i(x_k)^T d + G_(x_k) = 0 (i = 1, \dots, m_e) \\ \nabla G_i(x_k)^T d + G_(x_k) \leq 0 (i = m_e + 1, \dots, m) \end{gathered} \tag{1.15}$$

This subproblem can be solved using any QP algorithm and the solution is used to form a new iterate:

$$x_{k+1} = x_k + \alpha_k d \tag{1.16}$$

The step length parameter α_k is determined by a line search procedure so that an appropriate decrease in a merit function is obtained. The matrix H_k is an approximation of the Hessian matrix of the Lagrangian function and it can be updated using the BFGS method previously discussed in section 1.3.1. A nonlinearly constrained problem can often be solved with fewer iteration then an unconstrained problem using SQP; the reason for this is that, because of the limit in the feasible area, the optimizer can make well-informed decision regarding directions of search and step length.

1.5 Evolutionary Algorithms

The genetic algorithms are methods of solving both constrained and unconstrained optimization problems based on a biological evolutionary strategy, also called natural selection.

These methods repeatedly modifies a population of individual solutions. At each optimization step, the genetic algorithm selects and randomly modifies individuals from the current population with the purpose to generate the individuals for the next generation. Over the successive generations the population "evolves" toward an optimal solution.

The genetic algorithm uses three main rules at each optimization step to create the new population from the previous one:

1. **Selection rules** select the individuals, called *parents*, which contribute to the population at the next generation;

2. **Crossover rules** combine two parents to form the children for the next generation;

3. **Mutation rules** apply random changes to individual parents to form children.

Genetic algorithms are really effective in general applications because they can be used to solve a variety of optimization problems which are not well suited for the standard algorithms just discussed in sections 1.2, 1.3, 1.4, including problems in which the objective function is discontinuous, non-differentiable, stochastic, or highly non-linear, such as the CFD functions. The main differences between genetic algorithms and classical, derivative based, optimization algorithms are summarized in the Table 1.1.

Classical Algorithms	**Evolutionary Algorithms**
They generate a single point at each iteration. The sequence of points approaches an optimal solution	*They generate a population of points at each iteration. The generations sequence converges toward an optimal solution*
They select the next point in the sequence by deterministic calculations	*They select the next population by computations which uses random number generators*
Suitable for continuous, differentiable functions	*Suitable for every kind of function*
Appropriate for local optimum search	*Appropriate for global optimum search*
They are intrinsically single-objective algorithms	*The fact that they evolve a population, rather than a single solution, makes them intrinsically multi-objective algorithms*

Table 1.1: Main differences between Classical and Genetic Algorithms.

In the genetic algorithm practice is common the use of some biological terms, which are now introduced in the follows, for better clarify the meaning of the terminology used thorough the whole text of this Thesis.

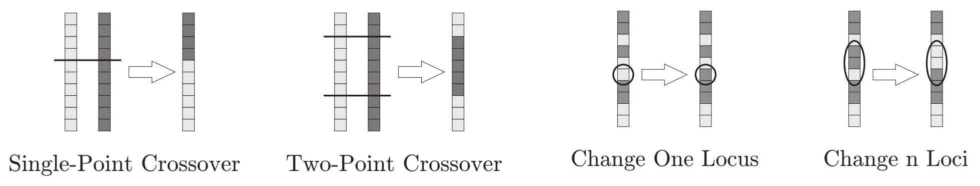
- **Fitness Function**: the *fitness function* is the function that have to be optimized; another way to call the objective function just named from section 1.1 to 1.4.1;
- **Individual**: an *individual* is any point in the search space to which the fitness function is applicable; it corresponds to a given set of decision variables;
- **Gene**: a *gene* is a vector component of an individual. It represents another way to call a design variable;
- **Score**: the *score* of an individual is its fitness function value;
- **Population**: the *population* is an array of individuals. In a problem characterized by a population size of n individuals and a number of design variables equal to m, the population is an $n \times n$ matrix;
- **Generation**: a *generation* is a single Evolutionary algorithm iteration;
- **Parents and Children**: to create a new *generation* the Evolutionary Algorithm selects certain individuals in the current population as *parents*, and uses them to create individuals in the next generation, called *children*. Thanks to a *selection* procedure, the algorithm is more likely to select parents with better fitness value;
- **Diversity**: it refers to the average distance between individuals in a population. This quantity is essential for the performance of a genetic algorithm because it enables the algorithm to search a larger region of the space. Different diversity preservation mechanism have been built.

For more elucidations regarding EA terminology, the reader is referred to the reference [19].

1.5.1 Framework of an Evolutionary Algorithm

The block scheme in Figure 1.4 summarizes how a typical Evolutionary algorithm works. In this scheme, the three methods used to generate the individuals of a new generation are highlighted:

- **Crossover children** are created by combining pairs of parents in the cur- rent population;
- **Mutation children** are created by randomly changing the genes of the individual parents;
- **Elite children** are the individuals in the current generation characterized by the best fitness values. The number of elite children changes depending on the specific algorithm considered;



(a) Crossover children (b) Mutation children

Figure 1.3: Crossover and mutation children.

Two common type of crossover and mutation reproduction strategies are presented Figure 1.3.

The operations outlined in the block scheme 1.4 are repeated until a stopping criterion is met during the process; usually the algorithm stops when the maximum number of generations has been reached.

1.6 Multi-Objective Optimization

In the engineering practice often happens that the problem to be addressed is characterized by a multi-objective nature. In those cases a single-objective problem formulation, even with several constraints, may not adequately represent the problem itself. If so, the right representation of the objective function is a vectorial one:

$$F(x) = [F(x_1), F(x_2), \dots, F(x_k)] \tag{1.17}$$

The relative importance of these objectives is not generally know until the system best capabilities are determined and trade-offs between the objectives found. Multi-objective optimization is concerned with the minimization of a vector of objectives *F(x)* that can be subjected to a number of constraints or bounds, as in the case of the mono-objective optimization problem formulated in Eq. 1.1 (n being the number of the decision variables, whereas m the number of conflicting objective functions) :

$$\min_{x \in \mathbb{R}^n} F(x), \quad F \in \mathbb{R}^n \tag{1.18}$$

subject to the constraints:

- equality constraints, G_i(x) = 0 (i = 1,. . . ,m_e);
- inequality constraints, G_i(x)$\leq$ 0 (m_e+1,. . . ,m);
- parameter bounds like $x_l \leq 0\ x \leq x_u$, where x_l is the parameters lower bound and x_l is the parameters upper bound respectively.

Note that because $F(x)$ is a vectorial function, if any of the components of $F(x)$ are competing, there is no unique solution to this problem. For this reason, it is

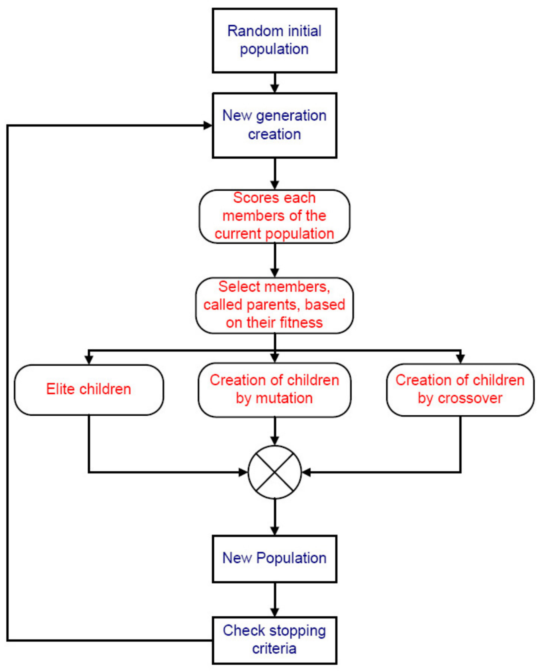

Random initial population
New generation creation
Scores each members of the current population
Select members, called parents, based on their fitness
Elite children
Creation of children by mutation
Creation of children by crossover
New Population
Check stopping criteria

Figure 1.4: Typical framework of an Evolutionary Algorithm.

important to introduce some Pareto[3] optimality concepts, which are introduced in the following paragraph.

1.6.1 Pareto concepts

Optimization problems involving multiple, conflicting objectives are often approached by aggregating the objectives into a scalar function and solving the resulting single-objective optimization problem. In contrast, in this study, we are concerned with finding a set of optimal trade-offs, the so-called Pareto-optimal set.

In the following, we formalize this well-known concept and also define the difference between local and global Pareto-optimal sets.

A multiobjective search space is partially ordered in the sense that two arbitrary solutions are related to each other in two possible ways: either one dominates the other or neither dominates.

Mathematically, a Multi Objective Optimization Problem (MOOP) minimizes the components of a vector $F(\mathrm{x}) = (f_1(\mathrm{x}); f_2(\mathrm{x}); \ldots; f_m(\mathrm{x}))$, where x is an n-dimensional decision variable vector $x \in N$, subject to $\mathrm{g}_i(\mathrm{x}) \leq 0$; i = 1;...;k. The evaluation function $\mathrm{F} : N \rightarrow M$ maps vectors $\mathrm{x} = (x_1; \ldots ; x_n)$ of the decision variable space N to vectors $\mathrm{y} = (\mathrm{y}_1; \ldots; \mathrm{y}_m)$ of the objective function space M (Figure 1.5).

The solution to a MOOP is a (possibly uncountable) set of decision variable vectors in N: the components of the corresponding vectors in M represent the

[3]Vilfredo Pareto (1848-1923): Italian engineer, economist and sociologist.

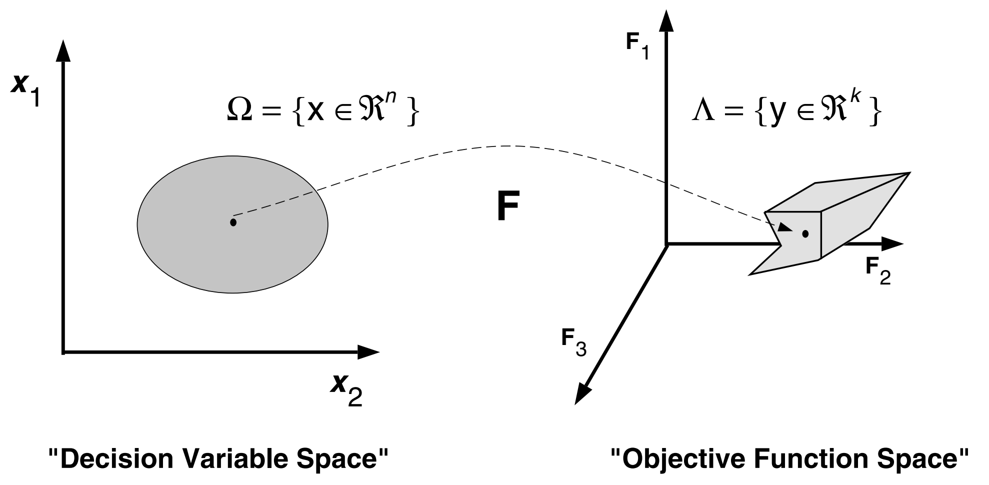

x_1
$\Omega = \{x \in \Re^n\}$
x_2
F
F_1
$\Lambda = \{y \in \Re^k\}$
F_2
F_3
"Decision Variable Space"
"Objective Function Space"

Figure 1.5: MOOP evaluation mapping.

best trade-offs in the objective function space. For the reader's convenience, the rest of the section is devoted to the mathematical definitions of the key concepts concerning Pareto optimality; examples of these concepts can be found elsewhere [19, 52].

Pareto Dominance: A vector $\mathbf{u} = (u_1,\ldots,u_m)$ is said to dominate $\mathbf{v} = (v_1\ldots,v_m)$ (denoted by $\mathbf{u} \prec \mathbf{v}$) if and only if $\forall i\ 1,\ldots,m,\ u_i \leq v_i \wedge \exists j \in 1,\ldots,m : u_j < v_j$. $\mathbf{u}$ is also said to *cover* $\mathbf{v}(\mathbf{u} \preceq \mathbf{v})$ if and only if $\mathbf{u} \prec \mathbf{v}$ or $\mathbf{u} = \mathbf{v}$.

Pareto Optimality: A solution $x \in N$ is said to be Pareto optimal with respect to the whole set N if and only if there is no other solution $x' \in N$ for which F(x') dominates F(x).

Pareto Optimal Set: For a given MOOP evaluation function $F : N \to M$, the Pareto optimal set (POS) is defined as the subset of all the Pareto optimal vectors in the decision variable set:

$$POS := \{x \ \in N := \nexists x' \in F(x') \prec F(x)\} \tag{1.19}$$

Pareto Front: For a given MOOP evaluation function $F : N \to M$ and Pareto optimal set POS, the Pareto front (PF) is defined as the set of the vectors mapped from POS to M by F:

$$PF := \{y = F(x) = (f_1(x),\ldots,f_m(x)) : x \in \text{POS}\} \tag{1.20}$$

1.6.2 Traditional Solution Methods

The classical approach to the solution of multi-objective optimizations is based on the idea to transform the original problem in a mono-objective one, with the purpose to apply then a mono-objective optimization strategy. Several methods are based on this concept:

- **Goal method**: with this strategy, the equivalent scalar function to be optimized is the distance between the vectorial objective function value and a certain reference vector (*Goal vector*) F^{Goal} chosen by the designer:

  $$\min d(x) = \|F(x) - F^{Goal}\|_p \tag{1.21}$$

  where

  $$\|v\|_p = (\sum_{i=1}^{k} |v|^p)^{1/p}, \quad 1 \leq p \leq \infty \tag{1.22}$$

  is the classic vector norm.

- **Weights method**: in this case the objective function is a linear combination of the objectives:

  $$\min \sum_{i=1}^{k} \omega_i F_i(x) \tag{1.23}$$

  where $\omega = (\omega_1, \omega_2, \ldots, \omega_k)^T \in \mathbb{R}_+^k (\geq 0)$ is the normalized weights vector $\sum_{i=1}^{k} \omega_i = 1$. The weights are chosen by the designer.

- **Trade off method**: one objective function $F_l(x)$ is chosen and the remaining objectives $F_i(x)(i = 1, 2, \ldots, k, i \neq l)$ are dealt as constraints, imposing their upper bound values ϵ_i . The problem to be solved is the follow:

  $$\begin{aligned} \min F_l(x), \quad l \in \{1, 2, \ldots, k\} \\ F_i(x) \in \epsilon_i, \forall i = 1, 2, \ldots, k \quad i \neq l \end{aligned} \tag{1.24}$$

  If the point x^* is the unique problem solution for at least an $l \in 1, 2, \ldots, k$ and $\epsilon_i = F_i(x^*)$ for all $i \neq l$, x^* is a Pareto optimal point for the original problem.

These are only few scalarization method suggested in literature. For more details about these and other scalarization methods see references [100], [88], [87].

1.6.3 Multi-Objective Evolutionary Algorithms

The main disadvantage of all the methods discussed in section 1.6.2 is the introduction of arbitrariness in the scalarization process; the solution obtained is highly sensitive to the decisions made by the designer about the scalarization (as the weight vector for example) and demands that the user have knowledge about the underlying problem. Moreover, in solving multi-objective problems, designers may be interested in a set of Pareto optimal points, instead of a single point solution. Since Evolutionary algorithms work with a population of points, it seems natural to use them in multi-objective optimization problems to capture a number of solutions simultaneously.

The ability of genetic algorithms to simultaneously search different regions of a solution space makes it possible to find a diverse set of solutions for very difficult problems with non-convex, discontinuous and multi-modal solutions spaces. The crossover operator may exploit structures of good solutions with respect to different objectives to create non-dominated solutions in unexplored parts of the Pareto front. In addition, most multi-objective Evolutionary algorithms do not require the user to prioritize, scale or weigh the objectives. Therefore Evolutionary algorithms have been the most popular heuristic approach to multi-objective design and optimization problems.

Multi-objective evolutionary algorithms (MOEA) differ from their mono-objective version about the fitness assignment; the most effective fitness assignment strategy for multi-objective optimization is the Pareto ranking approach which explicitly utilize the concept of Pareto dominance (section 1.6.1) [39] in evaluating fitness and assigning selection probability to solutions.

The population is ranked according to a dominance rule, and than each individual is assigned a fitness value based on its rank in the population, not its actual objective function value

For more details about multi-objective optimization using evolutionary algorithms, see references [23], [19], [31].

Chapter 2

GeDEAII: A powerful and robust MOEA

In this chapter the Evolutionary Algorithm designed and developed during my PhD course, namely the GeDEAII, will be presented. It shares the same framework of its precedessor (presented in [14]), whereas it features a new crossover operator, the Simplex-Crossover, and a novel mutation operator, the Shrink-Mutation. Finally, a new step was added to the ones described in the GeDEA reference paper: the Self-tuning operator. GeDEM operator was left unchanged and completed using the non-dominated-sorting based on crowding distance. The comparison among GeDEA-II and GeDEA, as well as with three other modern elitist methods, on different extremely multidimensional test problems, clearly indicates that the performance of GeDEA-II is, at least in these cases, superior.

2.1 Multiobjective Evolutionary Algorithms and the Problem of Diversity Preservation

The search process for the solutions to a MOOP has two main objectives:

1. To drive the search towards the true Pareto-optimal set/front;
2. To prevent premature convergence and distribute the solutions along the set/front itself.

Therefore, MOEAs have to face two major issues in order to accomplish these tasks, that is, respectively:

1. How to perform fitness assignment and selection (using aggregating, non-Pareto population-based or Pareto-based approaches);
2. How to maintain genetic diversity within the population of solutions.

To the best of the authors' knowledge, these requirements are nowadays fulfilled (in different manner) by most of state-of-the-art MOEAs.

However, two additional features are demanded of an efficient and robust MOEA, which are:

1. To perform the optimization containing at a minimum the overall optimization time;

2. To perform multiple runs while achieving the same results.

The latter feature is referred to as *repeatability*, and it is important in order to judge the efficiency of the evolutionary algorithm under investigation. In the following, we briefly analyze some of the most popular strategies used by the respective MOEAs to promote diversity. For comprehensive overviews of evolutionary approaches to multi-objective optimization the reader is referred to the following more specific studies [23, 19].

Many evolutionary algorithms for multiobjective optimization have been proposed [102, 23, 21]. Probably, the most popular multiobjective evolutionary algorithms today are the Strength Pareto Evolutionary Approach-2 (SPEA-2) [104] and the Nondominated Sorting Genetic Algorithm-II [24], which we have used for comparison in the Experimental results section. The two algorithms similarly maintain a separate population of size N (current population, or offspring population) and a fixed-capacity archive (previous population, or parent population), often (as in NSGA-II) also dimensioned N. In each generation, the current population is merged with the archive. This merged set is then subject to the process of elite preservation, wherein the individual solutions are ranked. The new archive is filled by taking the best-ranked solutions from the merged list. Rank conflicts are resolved via a diversity metric. Individuals are also subject to tournament selection, crossover, and mutation to form the population for the next generation. The main difference between the two is the way elite preservation is applied. NSGAII invokes a procedure called nondominated sorting. Nondominated sorting assigns domination ranks to each individual solution in a population, in such a manner that solutions are assigned lower ranks than the ones they dominate. Solutions are further prioritized based on a diversity metric that calculates the largest enclosing objective space hypercube that does not contain any other solutions with the same non-dominated sorting rank. The perimeter is then used to compute the density around each solution. SPEA-2, on the other hand, uses a procedure for identifying and preserving elites, first by calculating the strength of each solution as the number of solutions it dominates, and then summing the strengths of other solutions that dominate it as its raw fitness. A diversity term is then added to this to obtain the final fitness.

Another interesting multiobjective evolutionary algorithm is the Indicator-Based Evolutionary Algorithm (IBEA), proposed in [103]. It is a multiobjective evolutionary algorithm that calculates fitness values by comparing individuals on the basis of a quality indicator. Thereby, no particular diversity preservation technique such as fitness sharing, clustering, etc. is necessary.

2.2 Genetic Diversity Evolutionary Algorithm (GeDEA)

As GeDEA (Genetic Diversity Evolutionary Algorithm) forms the basis for GeDEA-II, we give a brief summary of the algorithm here. For a more detailed description, the interested reader is referred to [14]. The Genetic Diversity Evolutionary Algorithm (GeDEA) is a framework that is strictly designed around GeDEM to exalt its characteristics. Some of the design choices follow from the basic features of GeDEM (e.g., the replacement of clones, the use of an elitist strategy), the others are inspired by the will to make things as simple as possible, and neither introducing arbitrary parameters nor using sophisticated heuristics.

To briefly introduce the GeDEM principle, it is worth to conceptually go back to the section 2.1, where it has been stated that that the multi-objective optimization process has two objectives, which are themselves conflicting: the convergence to the Pareto-optimal set and the maintenance of genetic diversity within the population. The basic idea of GeDEM is to actually use these objectives during the evaluation phase and to rank the solutions with respect to them, emphasizing the non-dominated solutions as well as the most genetically different.

When the GeDEM is applied, the actual ranks of the solutions are determined maximizing (i) the ranks scored with respect to the objectives of the original MOOP, the non-dominated solutions having the highest rank, and (ii) the values assigned to each individual as a measure of its genetic diversity, calculated according to the chosen distance metric, i.e. the (normalized) Euclidean distance in the decision variable space. The structure of GeDEA follows the main steps of a $(\mu + \lambda)$ Evolution Strategy [10]. The evolution, however, is considered at its genotypic level, with the traditional binary coding of the decision variables. In the following the framework of the GeDEA is recalled for clarity.

Step 1: An initial population of μ individuals is generated at random.

Step 2: A mating pool of 2λ individuals is formed, each individual having the same probability of being selected.

Step 3: λ offspring are generated by crossover. Some bits of the offspring are also randomly mutated with a probability p_{mut}.

Step 4: The whole population of $\mu + \lambda$ individuals is checked to discover possible clones. These clones are removed and replaced with new randomly generated individuals (this is done to encourage the exploration of the search space and also to have the algorithm evaluate, for convenience, new λ different offspring every generation; still the occurrence of clones,Äô birth is not so frequent if clones are removed generation after generation).

Step 5: The objective function values of the $\mu + \lambda$ individuals are evaluated and the non-dominated sorting procedure by Goldberg (1989) is performed to assign the ranks to the solutions according to the objectives of the MOOP.

Step 6: The whole population of $\mu + \lambda$ individuals is processed to determine the value of the distance-based genetic diversity measure for each individual.

Step 7: GeDEM is applied according to the ranks scored in Step 5 and the values of the diversity measure assigned in Step 6. The non-dominated sorting procedure by [39] is used again to assign the ranks.

Step 8: The best μ solutions among parents and offspring, according to the ranks assigned in Step 7 by GeDEM, are selected for survival and the remaining λ are eliminated.

Step 9: If the maximum number of generations is reached then stop, else go to Step 2.

While GeDEAII shares with its predecessor the same framework, it is presented in this work in a real-coded fashion, hence it features the same parameters representation of the competitor algorithms already presented in Section 2.1. Moreover, this choice was made in view of using it for solving real-world engineering optimization problems. In order to prove the efficiency of the $(\mu + \lambda)$ Evolution Strategy, whose steps are strictly followed in GeDEAII, a comparison with the competitor algorithms framework was performed. For each algorithm, the same crossover, mutation and selection operators were exploited, that is the SBX Crossover [2], the Polynomial Mutation (implemented as described in [25]) and the Tournament Selection [9] operators, respectively. Results of these comparisons are depicted in Figure 2.1. The results presented here refer to the ZDT_1 bi-objective test problem, which involves 30 decision variables, and is thoroughly described in Section 2.4.1. Results hint that the GeDEAII framework provides remarkable results in terms of both convergence and Pareto Set coverage, and therefore underlies the good performance of the algorithm itself.

2.3 Genetic Diversity Evolutionary Algorithm-II (GeDEA-II)

GeDEA proved to be an efficient algorithm, able to explore widely the search space, while exploiting the relationships among the solutions. In order to enhance GeDEA algorithm performance further, several main features were added to the previous GeDEA version, yet retaining its constitutive framework. The main innovation is the novel crossover function, namely the Simplex-crossover, which takes place in lieu the previous Uniform crossover. Novel selection and mutation operators were also developed. The first one, namely the Neighbour-selection operator, allows exploring the design space more effectively. The second one, namely the Shrink-mutation, allows exploring more effectively the design space. Finally, a new step was added to the ones described in the GeDEA reference paper ([14]): the Self-tuning operator. The remaining steps characterizing GeDEA algorithm, in particular the GeDEM, were left unchanged. The latter was integrated with the Non-Dominating sorting procedure based on the crowding distance.

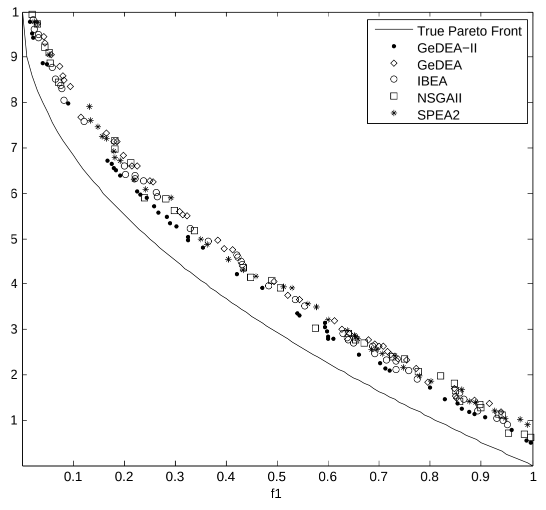


Figure 2.1: Approximate Pareto-optimal set reached by GeDEAII, GeDEA, IBEA, NSGA-II and SPEA2 provided with the same evolutionary operators, on ZDT_1 test function.

2.3.1 The SIMPLEX crossover

In many Evolutionary Algorithms (EAs), a recombination with two parents is commonly used to produce offspring. At the end of the 90's, in several studies the use of more than two parents for recombination in EAs have been reported [27, 65, 90].

In 1999, Tsutsui et. al [91] proposed the simplex crossover (SPX), a new multi-parent recombination operator for real-coded GAs. The experimental results with test functions used in their studies showed that SPX works well on functions having multimodality and/or epistasis with a medium number of parents: 3-parent on a low dimensional function or 4 parents on high dimensional functions. However, the authors did not consider the application of the SPX to multiobjective problems. Moreover, they did not consider the possibility to take into account the fitness of the objective function/s as the driving force of the simplex. Therefore, we decided to integrate in the GeDEA-II the simplex crossover with these and further new distinctive features.

Before introducing the Simplex crossover exploited in the GeDEA-II, some words are spent to elucidate the Simplex algorithm, whose first release was presented in [84] . A simplex in n-dimensions is a construct consisting of n+1 solutions x_k, $k = 1, 2, \ldots, n + 1$ [64]. In a two dimensional plane, this corresponds to a triangle. The solutions are evaluated in each step and the worst solution **w**, i.e., the one with the highest fitness value, is identified. The centroid, **M** , of the remaining points is computed as $\mathbf{M} = \frac{1}{n} \sum_x x_k$ (k identifying the two best solutions) and a new solution **r**, replacing **w**, is obtained by *reflection*, $\mathbf{r} = \mathbf{M} + (\mathbf{M} - \mathbf{w})$. In the Nelder and Mead version of the simplex algorithm, further operators are considered, such as expansion, internal contraction and external contraction. However, they are not taken into account in this work, since this choice would result in additional functions evaluations, as well as add complexity to the algorithm. In Figure 2.2, the reflection step of the Nelder and Mead simplex algorithm is depicted, applied to a problem in $\mathbb{R}^2$.

w is the worst point, to be replaced by point **r**. **M** is the centroid between the two other points, $\mathbf{x_1}$ and $\mathbf{x_2}$.

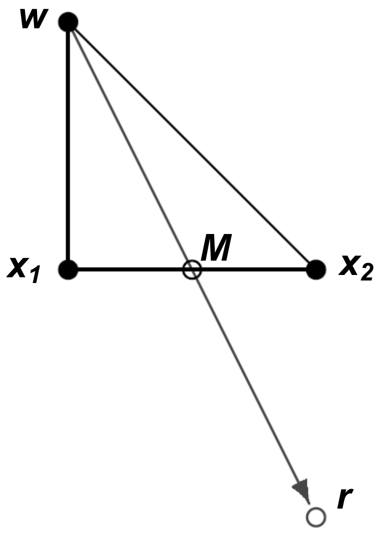


Figure 2.2: The reflection step of the simplex algorithm applied to a problem in $\mathbb{R}^2$.

GeDEA-II utilizes the Simplex concept as the crossover operator, in order to speed-up the evolution process, as stated below. As a matter of fact, crossover function plays an important role in the EAs, since it combines two individuals, or parents, to form a new individual, or child, for the next generation. Since the Simplex is itself an optimization algorithm, the generated children are expected to feature best fitness values when compared to the parents. Unlike the Simplex-crossover presented in [91], *Simplex-crossover* exploited in GeDEA-II requires only two parents to form a new child. This choice was motivated by the following considerations. First of all, it is reminded here that two is the minimum number required to form a simplex. From linear algebra, it can be easily demonstrated[4] that, given k vectors, there can be found more couples of mutually linearly independent vectors than can be done when considering triplets (or, in general, n-tuples) of independent vectors. As a straightforward consequence, it follows that this statement is even more true if not independence but only diversity (that is, at least one component different from a vector to another one) is required. Therefore, every time a new child is created, this characteristic of the *Simplex-crossover* ensures that this child comprises genes different from those featuring the other children, and allows the greatest design space exploration, due to the diversity of the parents.

[4]Let us consider three vectors in $\mathbb{R}^3$ design space, namely $\vec{\mathbf{a}}$=(1,0,0), $\vec{\mathbf{b}}$=(0,1,0), and $\vec{\mathbf{c}}$=(0,0,1). There can be found three couples of linearly independent vectors, that is [$\vec{\mathbf{a}}$,$\vec{\mathbf{b}}$], [$\vec{\mathbf{a}}$,$\vec{\mathbf{c}}$] and [$\vec{\mathbf{b}}$,$\vec{\mathbf{c}}$] but only a triplet of mutually and simultaneously independent vectors, that is the triplet [$\vec{\mathbf{a}}$,$\vec{\mathbf{b}}$,$\vec{\mathbf{c}}$]. This simple demonstration remains valid when extended to the $\mathbb{R}^n$ space.

These two parents are selected according to the selection procedure from the previous population, and combined following the guidelines of the simplex algorithm. Let assume *p1*, *p2* being the two parent vectors, characterized by different, multiple fitness values, the child vector *Child* is formed according to the reflection move described in ([64]:

$$Child := (1 + refl) \cdot M - Refl \cdot \mathrm{p}_2 \tag{2.1}$$

where *Child* is the new formed child and *Refl* is the reflection coefficient.

It is assumed that *p1* is the best fitness individual among the two chosen to form the *Child*, whereas *p2* the worst one. Belove the strategy followed to decide every time the best and the worst individual is highlighted.

M is the centroid of *p1*, calculated in the following manner:

$$M := \left(\frac{1}{n}\right) \cdot (p_1) \tag{2.2}$$

where *n* is the number of the remaining individual, excluded the worst one. A dedicated discussion will be done concerning this coefficient, and reported below. *Refl* coefficient is set equal to a random number ($refl \in [0, 1]$), unlike the elemental Simplex theory, which assumes a value equal to 1 for the *Refl* coefficient. This choice allows to create a child every time distant in a random manner from the parents, hence to explore more deeply the design space.

Moreover, unlike the Simplex algorithm theory, it was decided to give the algorithm the possibility to switch between 1 and 2 the coefficient *n* in Equation 2.2. This choice proved to be effective when used in conjunction with the Self-tuning operator described in Section 2.3.4, since the algorithm is given the possibility to choose every time the proper parameter setting.

The information about the fitness values is the key issue of this version of crossover: unlike the Simplex-crossover operator presented in Tsutsui et. al [91], this characteristic allows the crossover process to create a new individual, which is expected to be better than the parents. This new crossover operator was expected to combine both exploration and exploitation characteristics. In fact, the new formed child comprises the genes of two parents, that means a good exploration of the design space. However, it explores a design space region opposite to that covered by the parent number 2, that means it explores a region potentially not covered so far. In the early stages of the evolution, this means that child moves away from regions covered from bad parents, while exploring new promising ones.

Since the Simplex algorithm is itself a single-objective optimizer, a strategy was implemented to adapt it to a multi-objective algorithm. To deeply exploit the characteristics of the simplex, at each generation the mean of each objective function, extended to all the μ individuals, is computed. This mean is then compared to the one characterizing the previous generation, and the objective function having the greatest variation is selected as the fitness function used within the Simplex algorithm to decide every time the best and the worst individual. This choice was made after several experiments, which showed how a correct balance between exploration of the search space and the convergence to the P.F. can be achieved by means of a switching among multiple objective functions, each time selecting the most promising one.

In Figure 2.3, the flow chart related to the application of the Simplex crossover in a multi-objective context is presented, extended to the most general case involving M objective functions. It it assumed that all of the objectives are to be minimized. At each generation $ignr$, the mean of each objective function $mean$ is calculated. Based on these values, the percentage variations PV are subsequently derived. At this point, the two selected parents are sorted according to these values, and the child created according to Eqs. 2.1 and 2.2. This choice guarantees that the objective function characterized by the greatest variation is selected every time, therefore ensuring the highest convergence rate to the PF. For test problem involving more than two objective functions, the objective function considered to form the new child is chosen randomly in order to enhance the design space exploration of the crossover, required in highly dimensional objective spaces.

In order to demonstrate the effectiveness of the Simplex Crossover, two optimization runs, regarding two test problems, were performed, where each competitor MOEA was equipped first with the Simplex Crossover and then with the Simulated Binary crossover (while keeping the other operators unchanged). Two figures are presented that show the Approximate Pareto-optimal Sets reached from the competitors on two test functions selected among those presented Section 2.4.1.

Both the two test problems involve 100 decision variables. In the first problem, regarding ZDT_6, and shown in Figure2.4(a), Simplex Crossover is used instead of Simulated Binary crossover (described for the first time in [2]), with a probability of 75 percent. In Figure2.4(b), concerning the $DTLZ_6$ test problem, this probability is increased up to 90 percent. Results hint that the Simplex Crossover is very powerful when used in conjunction with the SBX in speeding up the evolution process without penalizing design space exploration. During evolution, GeDEA-II makes use exclusively of the Simplex Crossover until half of the generations has been reached. After that, Simplex Crossover is used alternatively with the SBX with a switching probability of 50 percent. This choice is motivated by the will of improving further the distribution and uniformity of the candidate solutions on the Approximate Pareto-optimal Set.

2.3.2 The Shrink mutation

As far as mutation is concerned, a new Shrink-mutation operator is introduced in the GeDEA-II.

In the open literature, this kind of mutation strategy is referred to as *Gaussian mutation* [11], and conventional implementations of Evolutionary Programming (EP) and Evolution Strategies (ES) for continuous parameter optimization using Gaussian mutations to generate offspring are presented in [10] and [30], respectively.

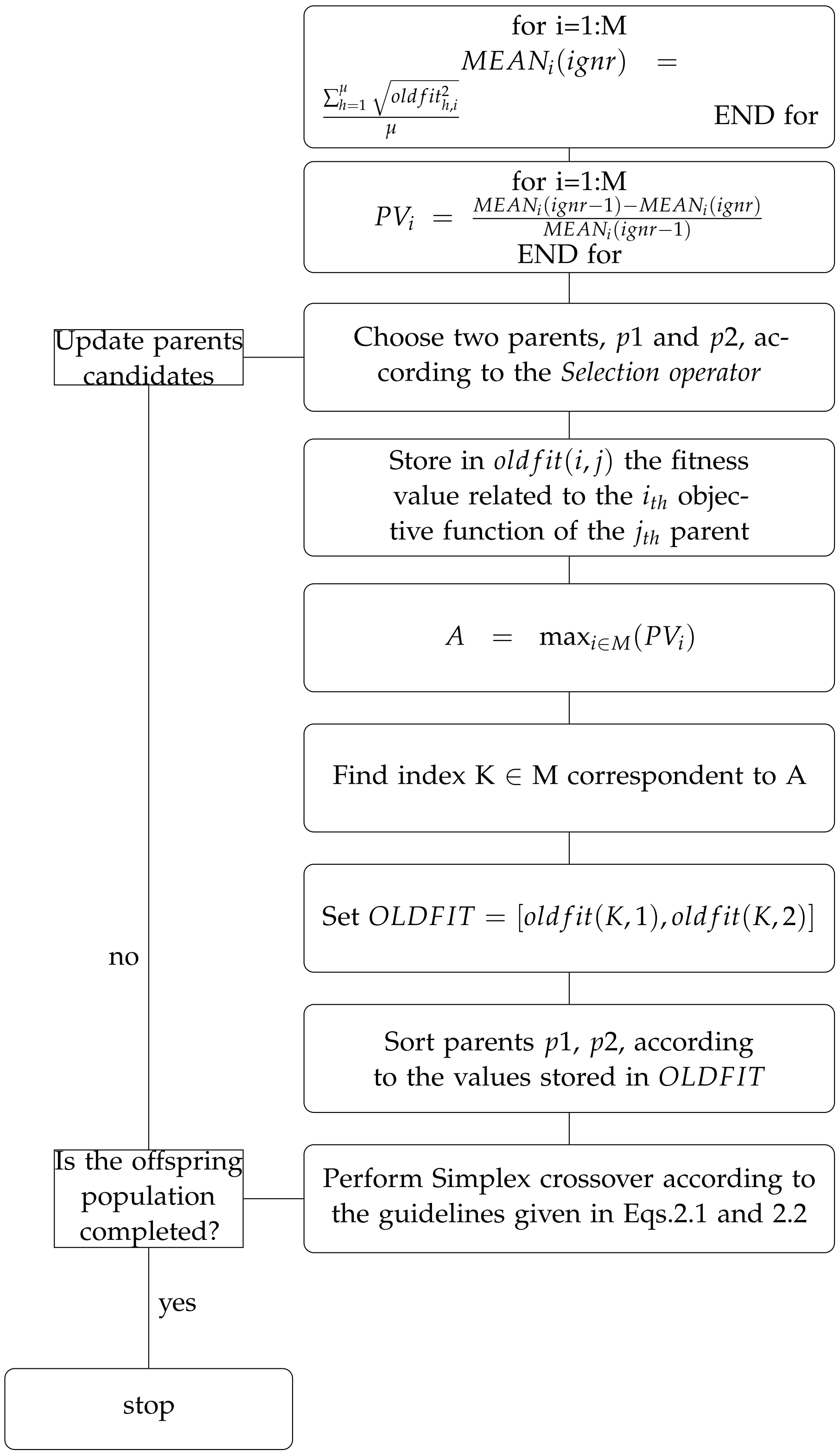
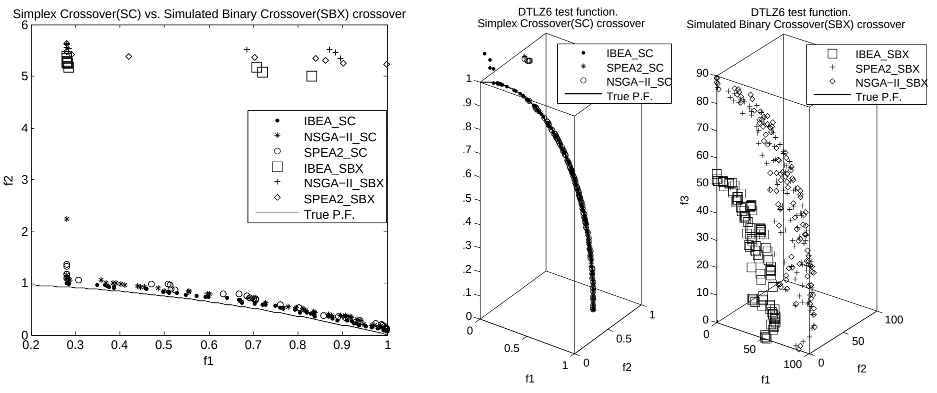

for i=1:M
$MEAN_i(ignr) = \frac{\sum_{h=1}^{\mu}\sqrt{oldfit_{h,i}^2}}{\mu}$
END for
for i=1:M
$PV_i = \frac{MEAN_i(ignr-1)-MEAN_i(ignr)}{MEAN_i(ignr-1)}$
END for
Choose two parents, $p1$ and $p2$, according to the *Selection operator*
Store in $oldfit(i,j)$ the fitness value related to the i_{th} objective function of the j_{th} parent
$A = \max_{i \in M}(PV_i)$
Find index K $\in$ M correspondent to A
Set $OLDFIT = [oldfit(K,1), oldfit(K,2)]$
Sort parents $p1$, $p2$, according to the values stored in $OLDFIT$
Perform Simplex crossover according to the guidelines given in Eqs.2.1 and 2.2
Is the offspring population completed?
yes
no
Update parents candidates
stop

Figure 2.3: Application of Simplex crossover in a multi objective context.



(a) ZDT_6 test function (b) $DTLZ_6$ test function

Figure 2.4: Approximate Pareto-optimal set reached by IBEA, NSGA-II and SPEA2 provided with and without the *SimplexCrossover*, on ZDT_6 (on the left) and $DTLZ_6$ (on the right) test functions.

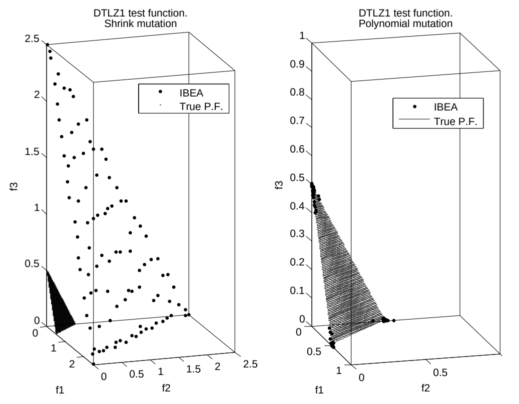
In general, mutation operator specifies how the genetic algorithm makes small random changes in the individuals in the population to create mutation children. Mutation provides genetic diversity and enables the genetic algorithm to search a broader space. Unlike the previous version of mutation featuring GeDEA algorithm, where some bits of the offspring were randomly mutated with a probability p_{mut}, here the mutation operator adds a random number taken from a Gaussian distribution with mean equal to the original value of each decision variable characterizing the entry parent vector. The shrinking schedule employed is:

$$Shrink_i := Shrink_{i-1} \cdot \left(1 - \frac{ignr}{ngnr}\right) \tag{2.3}$$

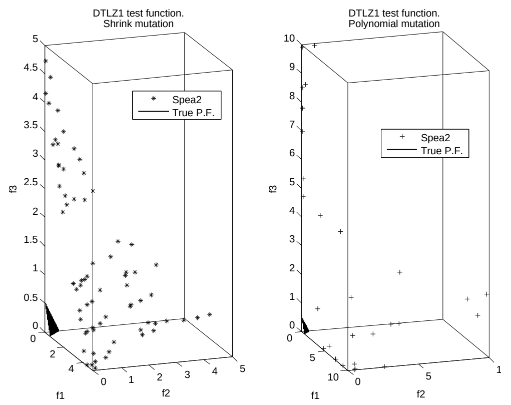
where $Shrink_i$ is a vector representing the current mutation range allowed for that particular design variable, *ignr* represents the current generation and *ngnr* the total number of generations. The shape of the shrinking curve was decided after several experimental tests. The fact that the variation is zero at the last generation is also a key feature of this mutation operator. Being conceived in this manner, the mutation allows to deeply explore the design space during the first part of the optimization, while exploiting the non-dominated solutions during the last generations. Once the current variation range has been calculated, one decision variable of the mutated child is randomly selected, and mutated according to the following formula, resulting from an extensive experimental campaign:

$$Child_{mut} := Child_{cross} + [(\sqrt{Shrink_i} \cdot random) \cdot Shrink_i] \tag{2.4}$$

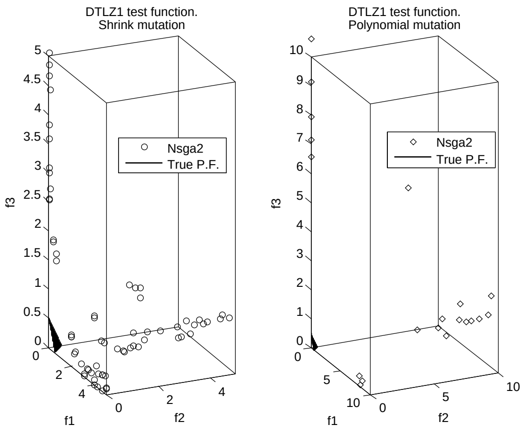
where $Child_{mut}$ is the mutated decision variable, $Child_{cross}$ is the decision variable generated by the previously introduced crossover operator and *random* is a random number taken from a normal distribution in the open interval [-1,1]. Unlike crossover operator, which generates all the offspring, mutation is applied only



(a) $DTLZ_1$ test function. IBEA algorithm



(b) $DTLZ_1$ test function. Spea2 algorithm



(c) $DTLZ_1$ test function. Nsga2 algorithm

Figure 2.5: Approximate Pareto-optimal sets reached by IBEA, NSGA-II and SPEA2 provided with the *ShrinkMutation* operator (on the left side of each subfigure) and with the *PolynomialMutation* operator (on the right side of each subfigure).

on a selected part of the offspring, depending on the strategy selected by means of the *Self-tuning* operator. Before starting offspring mutation, offspring population is randomly shuffled to prevent locality effects. After that, a pre-established percentage (based on the *Self-tuning* operator) of the individuals are selected of mutation. The initial Shrink factor was set equal to ten percent of the variation range of the design variables. This mutation operator was found to be powerful especially in multi-objective problems requiring a huge exploration of the design space. In Figure 2.5, a comparison between Shrink mutation and Simulated Binary mutation is depicted. Results refer to $DTLZ_1$ test function, a tri-objective test function involving 7 decision variables, and described in Section 2.4.1. The algorithms chosen for the test are the same introduced in Section 2.4. Similar behaviors were found with the other test function, not reported here for brevity and clarity.

Figure 2.5(a) shows that Polynomial mutation used in IBEA algorithm gives the best results in terms of convergence towards the True Pareto Front. However, as far as distribution of solutions is concerned, Shrink mutation performs better. However, the results of this comparison are slightly different when NSGA-II, whose test is presented in Figure 2.5(c) and SPEA2, whose test is presented in Figure 2.5(b), are provided with the Shrink mutation operator. In these cases, the latter gives the best results in terms of both convergence and distribution when compared to Polynomial mutation. Thank to these results, as far as GeDEA-II is concerned, it was decided to use them alternatively during the evolution, in order to capture the best of the two mutation operators.

2.3.3 Diversity preservation

As underlined in Sections 2.1 and 2.2, maintaining the genetic diversity within the population is mandatory for a robust EA. To this purpose, in GeDEA-II two diversity preservation mechanism are used, namely the GeDEM, already employed in GeDEA [14] and the Non-Dominated Sorting [24]. Both of the two mentioned mechanism are adopted since in authors' opinion each of them has unique features which can take benefit from each other. To make this assertion clearer, it is worth to briefly go back to the mathematical definition of GeDEM and non-dominated sorting based on crowding distance.The definition of dominance used in the non-dominated sorting procedure performed by GeDEM is:

$$\text{Vector } \mathbf{u} = (\text{rank}_u; \text{dist}_u) \text{ dominates vector } \mathbf{v} = (\text{rank}_v; \text{dist}_v)$$
$$\text{if and only if } (\text{rank}_u > \text{rank}_v) \bigvee (\text{dist}_u \geq \text{dist}_v)$$

On the contrary, the definition of dominance used in the non-dominated sorting based on crowding distance is:

$$\text{Vector } \mathbf{u} = (\text{rank}_u; \text{dist}_u) \text{ dominates vector } \mathbf{v} = (\text{rank}_v; \text{dist}_v)$$
$$\text{if and only if } (\text{rank}_u > \text{rank}_v) \bigwedge [(\text{rank}_u = \text{rank}_v) \bigvee (\text{dist}_u \geq \text{dist}_v)]$$

Clearly, the logical operator is the great difference between the aforementioned diversity mechanisms, which entails the slightly different behavior of the two algorithms. In particular, GeDEM tends to create less non-dominated individuals, since both the rank and the diversity conditions are to be fulfilled simultaneously. Therefore, the evolution process results faster. On the other hand, non-dominated sorting based on crowding distance tends to create more non-dominated individuals, which results in a better Pareto front coverage. In order to take advantage of both the characteristics, in GeDEA-II the diversity preservation is accomplished by means of GeDEM, in the first three quarters of the generations, whereas in the remainder of the generations the Non-Dominated Sorting mechanism is exploited.

2.3.4 The Self-tuning operator

The last enhancement introduced in the GeDEA-II is the *Self-tuning operator*, which allows to automatically switch between two set of parameters, and to give the algorithm the possibility to better adapt itself to the particular multi-objective problem being optimized.

These parameters were selected on the basis of a preliminary experimental procedure, aiming at pointing out the most influent parameters on both exploration and convergence characteristics of the GeDEAII.

Initially, they were organized into different sets, and after several experimental tests two sets were identified, namely Set_0 and Set_1. Therefore, the GeDEA-II behaviour can be changed by simply switching between Set_0 and Set_1.

The first parameter regards the n value in the *Simplex-Crossover*, 0 meaning a n value equal to 2, 1 meaning a n value equal to 1.

The second regards the switching among the objective functions in order to select the one which will act as the driving force of the Simplex Crossover: 0 means that the choice comes down to the one having the greatest variation, as anticipated in Section 2.3.1, whereas 1 means that the switching is performed in a random manner.

The third parameter regards mutation operator, 0 referring to the original mutation operator employed in GeDEA version (all the individuals undergo mutation), whereas 1 referring to the new *Shrink-mutation* operator (10% of the individuals undergo mutation).

Only two distinctive sets were established to make the procedure as much simple as possible and at the same time to reduce at a minimum the number of generations required for the optimization.

Details of the *Self-tuning operator* constitutive framework are given below:

1. The first four generations are calculated by means of Set_1, starting from the initial population randomly created;

2. The second four generations are calculated by means of Set_0, once again starting from the same the initial population;

3. The mean of all the objective functions are calculated for the initial population; calculations are repeated for the final populations created with each set (the fourth and the eighth population, respectively);

4. For each of the objective functions, the percentage variations are calculated, for both the two sets.

5. For both the two sets, the greatest variation is memorized;

6. Finally, the set is selected, which yielded the greatest variation between the two stored;

In particular, an experimental campaign showed us that four generations per each set were sufficient to select the best set. The remaining generations are calculated with the finally selected set.

In Table 2.1, the two sets are introduced. Despite its simplicity, this operator proved to give the GeDEA-II the ability to well adapt itself to multi-objective problems of different nature.

Operator	**Set$_0$**	**Set$_1$**
Crossover	n=2	n=1
Crossover	Best objective function	Random switching
Mutation	*Shrink-mutation* (10% of the individuals)	*Shrink-mutation* (100% of the individuals)

Table 2.1: Settings of the Self-tuning operator.

2.4 Comparison with Other Multiobjective Evolutionary Algorithms

In order to judge the performance of the GeDEA-II, a comparison with other different state-of-the-art multi-objective EAs was performed. SPEA-2 [102], NSGA-II [24] and IBEA [103] were chosen as competitors, and their performance against GeDEA-II were measured on five test problems featuring the characteristics that may cause difficulties in converging to the Pareto-optimal front and in maintaining diversity within the population [22]: convexity, non-convexity, discrete Pareto fronts, multimodality, and biased search spaces. In addition, their performance were tested also on Kursawe test Function KUR [54]. The latter consists of a multi-modal function and function with pair-wise interactions among the variables, the Pareto front is disconnected consisting of concave and convex parts and an isolated point. Finally, GeDEA-II performance was tested on three more recent and more challenging benchmark test functions, i.e. the scalable Test Problems presented in [26]. The simplicity of construction, scalability to any number of decision variables and objectives, knowledge of the shape and the location of the resulting Pareto-optimal front, and introduction of controlled difficulties in both converging to the true Pareto-optimal front and maintaining a widely distributed set of solutions are the main features of the suggested test problems. The thirteen test functions, the methodology and the metric of performance used in the comparison are briefly recalled in the following for easy reference.

2.4.1 Test functions

Each of the five test functions T_1, T_2, T_3, T_4 and T_6 introduced by [102] is a two-objective minimization problem that involves a distinct feature among those identified by [22]. All the test functions are constructed in the same way, according to the guidelines in [22]:

$$\begin{aligned} \mathit{Minimize} : T(\mathrm{x}) &= (f_1(x_1), f_2(x)) \\ \mathit{subject\ to} : f_2(\mathrm{x}) &= g(x_2;\ldots;x_m)h(f_1(x_1), g(x_2;\ldots;x_m)) \\ \mathit{where} : \mathrm{x} &= (x_1,\ldots,\mathrm{x}_\mathrm{M}) \end{aligned} \tag{2.5}$$

Function f controls vector representation uniformity along the Pareto Approximation Set. Function g controls the resulting MOP characteristics (whether it is multifrontal or has an isolated optimum). Function h controls the resulting Pareto front characteristics (e.g., convex, disconnected, etc.) These functions respectively influence search along and towards the true Pareto front, and the shape of a Pareto front in $\mathbb{R}^2$. Deb [22] implies that a MOEA has difficulty finding PF$_{true}$ because it gets ‚Äútrapped‚Äù in the local optimum, namely PF$_{local}$. Test functions reported in this work feature an increased number of decision variables, when compared to their original versions reported in [102]. This choice was motivated by the authors' will of testing exploration capabilities of the algorithms also on highly dimensional test problems, and contributes to justify the results presented in Section 2.4.4.

- Test function T_1 has a convex Pareto-optimal front:

$$\begin{aligned} f_1(x_1) &= (x_1) \\ g(x_2;\ldots;x_n) &= 1+9\cdot\sum_{i=2}^{n}\frac{x_i}{(n-1)} \\ h(f_1,g) &= 1-\sqrt{\frac{f_1}{g}} \end{aligned} \tag{2.6}$$

  where n = 100 and x$_i$∈ [0,1]. The Pareto-optimal front corresponds to g(x) = 1. The original version presented in [102] featured 30 decision variables.

- Test function T_2 has a non-convex Pareto-optimal front:

$$\begin{aligned} f_1(x_1) &= (x_1) \\ g(x_2;\ldots;x_n) &= 1+9\cdot\sum_{i=2}^{n}\frac{x_i}{(n-1)} \\ h(f_1,g) &= 1-\left(\frac{f_1}{g}\right)^2 \end{aligned} \tag{2.7}$$

  where n = 100 and x$_i$∈ [0,1]. The Pareto-optimal front corresponds to g(x) = 1. The original version presented in [102] featured 30 decision variables.

- Test function T_3 features a Pareto-optimal front disconnected, consisting of several noncontiguous convex parts:

$$\begin{aligned} f_1(x_1) &= (x_1) \\ g(x_2;\ldots;x_n) &= 1+9\cdot\sum_{i=2}^{n}\frac{x_i}{(n-1)} \\ h(f_1,g) &= 1-\left(\sqrt{\frac{f_1}{g}}\right)-\left(\frac{f_1}{g}\right)\cdot\sin(10\pi f_1) \end{aligned} \tag{2.8}$$

  where n = 100 and x$_i$∈ [0,1]. The Pareto-optimal front corresponds to g(x) = 1. The original version presented in [102] featured 30 decision variables.

- Test function T_4 contains 21^9 local Pareto-optimal fronts and, therefore, tests for the EA ability to deal with multifrontality:

$$\begin{aligned} f_1(x_1) &= (x_1) \\ g(x_2;\ldots;x_n) &= 1+10(n-1)\cdot\sum_{i=2}^{n}\left(x_i{}^2-10\cos(4\pi x_i)\right) \\ h(f_1,g) &= 1-\sqrt{\frac{f_1}{g}} \end{aligned} \tag{2.9}$$

where n = 100 and x$_i$∈ [0,1]. The Pareto-optimal front is convex and corresponds to g(x) = 1. The original version presented in [102] featured 10 decision variables.

- Test function T_6 features two difficulties caused by the nonuniformity of the search space: first, the Pareto optimal solutions are nonuniformly distributed along the PF$_{true}$ (the front is biased for solutions for which $f_1(x_1)$ is near one); and second, the density of the solutions is lowest near the PF$_{true}$ and highest away from the front::

$$\begin{aligned} f_1(x_1) &= 1-\exp(-4x_1)\sin^6(6\pi x_1) \\ g(x_2;\ldots;x_n) &= 1+9\cdot\left(\sum_{i=2}^{n}\frac{x_i}{(n-1)}\right)^{1/4} \\ h(f_1,g) &= 1-\left(\frac{f_1}{g}\right)^2 \end{aligned} \tag{2.10}$$

  where n = 100 and x$_i$∈ [0,1]. The Pareto-optimal front is non-convex and corresponds to g(x) = 1. The original version presented in [102] featured 10 decision variables.

The multi-objective test function designed by Kursawe [54] is included because this two-objective function PF$_{true}$ has several disconnected and unsymmetric areas in solution space. Its PF$_{true}$ consists of three disconnected Pareto curves. Its solution mapping into dominated objective space is quite convoluted. Its number of decision variables is arbitrary. However, changing the number of decision variables appears to slightly change PF$_{true}$ shape and does change its location in objective space. We use it here with three decision variables. The variation range is augmented for all of the variables involved by a factor two, when compared to the original range, which is [-5,5], in order to increase the difficulty of the problem.

$$\begin{aligned} f_1(x) &= \sum_{i=1}^{n-1}\left(-10\cdot\exp\left(-0.2\sqrt{x_i^2+x_{i+1}^2)}\right)\right) \\ f_2(x) &= \sum_{i=1}^{n}\left(|x_i|^{0.8}+5\sin(x_i^3)\right) \end{aligned} \tag{2.11}$$

Finally, the entire set of tri-objective test functions designed by Kalyanmoy Deb, Lothar Thiele, Marco Laumanns and Eckart Zitzler, and presented in [26], are considered, in order to demonstrate the GeDEA-II capabilities on more than two-objectives test problems. In the following, n identifies the number of decision variables, M the number of objective functions, and $k = |x_M| = n - M + 1$ the number of variables of the functional $g(x_M)$. The number of variables was always increased when compared to that suggested by the authors in [26], whereas the decision variables range was left unchanged. These features help clarify the different results between those reported in Section 2.4.4 and the original ones [26].

- Test function *DTLZ1* contains $(11^k - 1)$ local Pareto-optimal fronts, each of which can attract an MOEA [26]. To increase the difficulty of the problem, the number of variables and the frequency of the cosine function were doubled when compared to those suggested in [26].

$$
\begin{aligned}
f_1(x) &= (1+g(x_M))(x_1)(x_2) \\
f_2(x) &= \frac{1}{2}(1+g(x_M))(x_1)(1-x_2) \\
f_3(x) &= \frac{1}{2}(1+g(x_M))(1-x_1) \\
\textit{and } g &= 100\left[k + \sum_{x_1 \in x_M} (x_i - 0.5)^2 - \cos(40\pi(x_i - 0.5))\right]
\end{aligned}
\tag{2.12}
$$

  where n = 14 and $x_i \in [0,1]$.

- Test function *DTLZ2* is the Generic sphere problem, according to the definition given to it in [26].

$$
\begin{aligned}
f_1(x) &= (1+g(x_M))\cos(x_1\text{ß}/2)\cos(x_2\text{ß}/2) \\
f_2(x) &= (1+g(x_M))\cos(x_1\text{ß}/2)\sin(x_2\text{ß}/2) \\
f_3(x) &= (1+g(x_M))\sin(x_1\text{ß}/2) \\
\textit{and } g &= \sum_{x_1 \in x_M} (x_i - 0.5)^2
\end{aligned}
\tag{2.13}
$$

  where n = 22 and $x_i \in [0,1]$. The number of variables suggested in [26] is 12.

- Test function *DTLZ3* is similar to test function *DTLZ2*, except for the function g, which introduces $(3^k - 1)$ local Pareto-optimal fronts, and only one global Pareto-optimal front.

$$
\begin{aligned}
f_1(x) &= (1+g(x_M))\cos(x_1\text{ß}/2)\cos(x_2\text{ß}/2) \\
f_2(x) &= (1+g(x_M))\cos(x_1\text{ß}/2)\sin(x_2\text{ß}/2) \\
f_3(x) &= (1+g(x_M))\sin(x_1\text{ß}/2) \\
\textit{and } g &= 100\left[k + \sum_{x_1 \in x_M} (x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5))\right]
\end{aligned}
\tag{2.14}
$$

  where n = 22 and $x_i \in [0,1]$. The number of variables suggested in [26] is 12.

- Test function *DTLZ4* is a modified version of *DTLZ2*, since it features a different meta-variable mapping.

$$\begin{aligned} f_1(x) &= (1+g(x_M))\cos(x_1^{ff}\text{ß}/2)\cos(x_2^{ff}\text{ß}/2) \\ f_2(x) &= (1+g(x_M))\cos(x_1^{ff}\text{ß}/2)\sin(x_2^{ff}\text{ß}/2) \\ f_3(x) &= (1+g(x_M))\sin(x_1^{ff}\text{ß}/2) \\ \textit{and } g &= 100\left[k+\sum_{x_1\in x_M}(x_i-0.5)^2-\cos(20\pi(x_i-0.5))\right] \end{aligned} \tag{2.15}$$

  where n =22, $\alpha = 100$ and x$_i\in$ [0,1]. Deb, Thiele, Laumanss and Zitzler [26] also suggest n=12 here.

- Test function *DTLZ5* features a different mapping compared to the one of *DTLZ4*. This problem will test a MOEA ability to converge to a curve and will also allow an easier way to visually demonstrate the performance of the algorithm.

$$\begin{aligned} f_1(x) &= (1+g(x_M))\cos(x_1\text{ß}/2)\cos(\text{`}_2) \\ f_2(x) &= (1+g(x_M))\cos(x_1\text{ß}/2)\sin(\text{`}_2) \\ f_3(x) &= (1+g(x_M))\sin(x_1\text{ß}/2) \\ g &= 100\left[k+\sum_{x_1\in x_M}(x_i-0.5)^2\right] \\ \textit{and } \theta_2 &= \frac{\pi}{(4(1+g))}(1+2gx_2) \end{aligned} \tag{2.16}$$

  where n =22 and x$_i\in$ [0,1]. The number of variables suggested in [26] for this test problem is 12.

- Test function *DTLZ6* is a modified, harder-to-optimize version of the above test problem. The number of decision variables was dramatically increased when compared to the original one.

$$\begin{aligned} f_1(x) &= (1+g(x_M))\cos(x_1)\cos(\text{`}_2) \\ f_2(x) &= (1+g(x_M))\cos(x_1)\sin(\text{`}_2) \\ f_3(x) &= (1+g(x_M))\sin(x_1\text{ß}/2) \\ g &= \sum_{x_1\in x_M}\left(x_i\right)^{0.1} \\ \textit{and } \theta_2 &= \frac{\pi}{(4(1+g))}(1+2gx_2) \end{aligned} \tag{2.17}$$

  where n = 100 and x$_i\in$ [0,1]. The number of variables suggested in [26] for this test problem is 12.

- Test function *DTLZ7* features 2^{M-1} disconnected local Pareto-optimal regions in the search space. It is chosen to test the MOEA ability in finding and maintain stable and distributed subpopulations in all four disconnected global Pareto-optimal regions.

$$\begin{aligned} f_1(x) &= x_1 \\ f_2(x) &= x_2 \\ f_3(x) &= (1+g(x_M))h \\ g &= 1+\frac{9}{k}\sum_{x_1\in x_M}(x_i) \\ \textit{and } h &= M-\sum_{i=1}^{M-1}\left[\frac{f_i}{1+g}(\sin((1+3\pi f_i))\right] \end{aligned} \tag{2.18}$$

where n = 100 and $x_i \in$ [0,1]. Once again, the number of decision variables was dramatically increased when compared to the original one, suggested in [26] for this test problem, and equal to 22.

2.4.2 Methodology

The methodology used in [102] is strictly followed. GeDEA and competitors are executed 30 times on each test function. There are different parameters associated with the various algorithms, some common to all and some specific to a particular one. In order to make a fair comparison among all the algorithms, most of these constants are kept the same. In GeDEA-II, GeDEA and in competitors algorithms, the population size is set to 100. In the following, the parameters of the competitors MOEA are reported following the terminology used in PISA implementation[5]. The individual mutation probability is always 1 and the variable mutation probability is fixed at $1/n$, n being the number of the decision variables of the test problem considered. The individual recombination probability along with the variable recombination probability are set to 1. The variable swap probability is set to 0.5. $\eta_{mutation}$ is always set to 20 and $\eta_{recombination}$ is fixed to 15. For IBEA algorithm, tournament size is always set to 2, whereas additive epsilon is chosen as the indicator. Scaling factor *kappa* is set to 0.05, and *rho* factor is fixed to 1.1. For both NSGA-II and SPEA2, tournament size is given a value equal to 2. NSGA-II, SPEA-2 and IBEA are run with the PISA[6] implementation [15], with exactly the same parameters and variation operators. The maximum number of generations for test functions T_1, T_2 and T_6 is set to 20 for all the algorithms, for test functions T_3 and T_4 the individuals are evolved for 40 generations but for the Kursawe test function the number of generations is set to 50. For test function $DTLZ_1$ and $DTLZ_7$, the number of generations is set to 100. This number of generation is increased up

[5] *Individual mutation probability* (probability that a certain individual undergoes mutation); *individual recombination probability* (probability that a certain pair of individuals undergoes recombination); *variable mutation probability* (probability that a certain variable in a given individual is mutated); *variable swap probability* (probability that a certain pair of variables is swapped during recombination); *variable recombination probability* (probability that the SBX recombination operator is used for a given pair of variables; this decision is independent from variable swap probability); $\eta_{mutation}$ (distribution index for mutation operator); $\eta_{recombination}$ (distribution index for recombination operator).

[6] This software is available for public use at PISA website `http://www.tik.ee.ethz.ch/pisa/`

to 150 for $DTLZ_3$ test function. In test functions $DTLZ_4$ and $DTLZ_5$, individuals are evolved for 80 generations, whereas on $DTLZ_2$ and $DTLZ_6$ the number of generation is set to 70 and 50, respectively.

In Table 2.2, the original number of generations characterizing test problems presented in [102] and [26], is compared to the ones used here. The number of generations was reduced in order to test the convergence properties of the investigated algorithms. In fact, being the number of individuals constituting a generation left unchanged, a reduction of generations directly translates into a reduction of the objective functions evaluations.

	Original version problems	Proposed test problems
ZDT1	250	20
ZDT2	250	20
ZDT3	250	30
ZDT4	250	40
ZDT6	250	20
DTLZ1	300	100
DTLZ2	300	70
DTLZ3	500	150
DTLZ4	200	80
DTLZ5	200	80
DTLZ6	500	50
DTLZ7	200	100

Table 2.2: Original and proposed number of generations for the *ZDT* and *DTLZ* test suites.

The different settings contribute to justify the different results reported here, when compared to those presented in the original papers [102, 26].

2.4.3 Metric of Performance

Different metrics can be defined to compare the performance of EAs with respect to the different goals of optimization itself [102]: how far is the resulting non-dominated set from the Pareto front, how uniform is the distribution of the solutions along the Pareto Approximation set/front, how wide is the Pareto Approximation set/front. For measuring the quality of the results, we have employed the Hypervolume approach, due to its construction simplicity and for the reason, which will be soon explained. The hypervolume approach by [102] (modified in [104]) measures how much of the objective space is dominated by a given non-dominated set. Zitzler et al. state it as the most appropriate scalar indicator since it combines both the distance of solutions (towards some utopian trade-off surface) and the spread of solutions. To better understand the reason for this choice, it is worth to see at Figure 2.6 (a) and (b). The reference point is indicated as R.P..

As regards the convergence of the known P.F. to the True P.F., please consider the case depicted in Figure 2.6 (a). The non-dominated set A has a great Hypervolume indicator when compared to set B, due to its superior proximity to the True P.F.. As far as the spread of the solution on the Pareto Approximation Set is concerned,

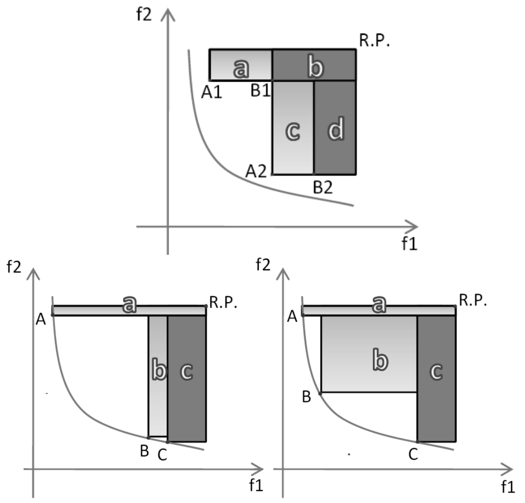


Figure 2.6: Significance of the Hypervolume indicator as far as convergence (a, AT THE TOP), and diversity (b, AT THE BOTTOM) is concerned.

Figure 2.6 (b) qualitatively shows that a more uniform distribution of the solutions (on the right) yields a greater Hypervolume indicator. Therefore, this indicator is intrinsically able to compare performance of different EAs as regards both the convergence to the Pareto Approximation Set and its coverage. In general, it is not sufficient for a set of candidate solutions to be closer than another one to the True Pareto front, to have a higher hypervolume value. It is the blend of convergence and uniformity of the final Approximation Set that counts.

The Hypervolume[7] is defined as the area of coverage of PF_{known} with respect to the objective space for a two-objective MOP. As illustrated in Figure 2.6, this region consists of an orthogonal polytope, and may be seen as the union of n axis-aligned hyper-rectangles with one common vertex (the reference point, R.P.). Mathematically, this is described in equation 2.19 (for a generic n-objectives problem):

$$Hypervolume := \left[\bigcup_i vol_i | vec_i \in P_{known} \right] \tag{2.19}$$

where vec_i is a nondominated vector in PF_{known} and vol_i is the Hypervolume (an area in two objectives problems) between the reference point and vector vec_i.

In this work we use the version implemented by Fonseca et al. and presented in [32].

2.4.4 Results of Comparison

As in Zitzler et al. [102], Figures 2.7-2.12 and 2.14-2.20 show an excerpt of the non-dominated fronts obtained by the EAs and the Pareto-optimal fronts (continuous curves). The points plotted are the non-dominated solutions extracted

[7]The Hypervolume is a Pareto compliant indicator as stated in [19].

from the union set of the outcomes of the first five runs, the best and the worst one being discarded. The performance of GeDEA-II is also compared to NSGA-II, SPEA2 and IBEA according to the hypervolume metric defined in Equation 2.19. The distribution of these values is shown using box plots in Figure 2.13 and 2.21. On each box, the central line represents the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered outliers, and outliers are plotted individually, with a Plus sign. Results are normalized with the best Hypervolume value coming from the union set of all of the runs, extended to all of the algorithms. For each test problem, the reference point is assumed equal for all of the algorithms, and equal to the maximum value for each objective function from the union of all of the output points. In presenting the *DTLZ* test suit results, the exact order is not strictly followed due to layout reasons.

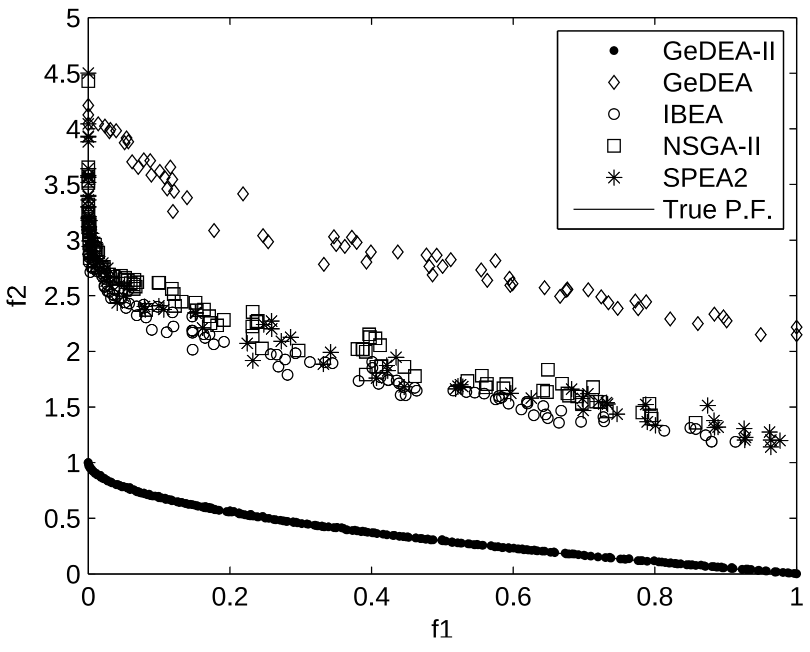


Figure 2.7: Test function T_1 (convex).

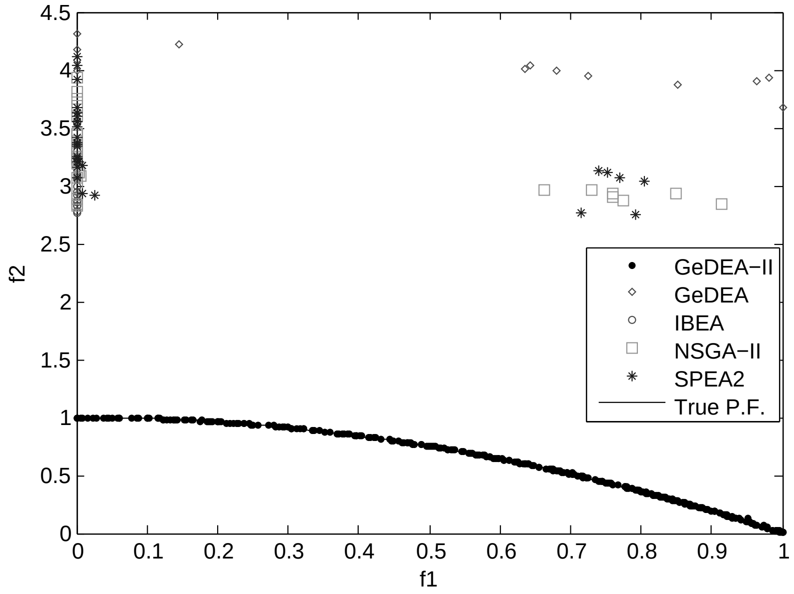


Figure 2.8: Test function T_2 (non-convex).

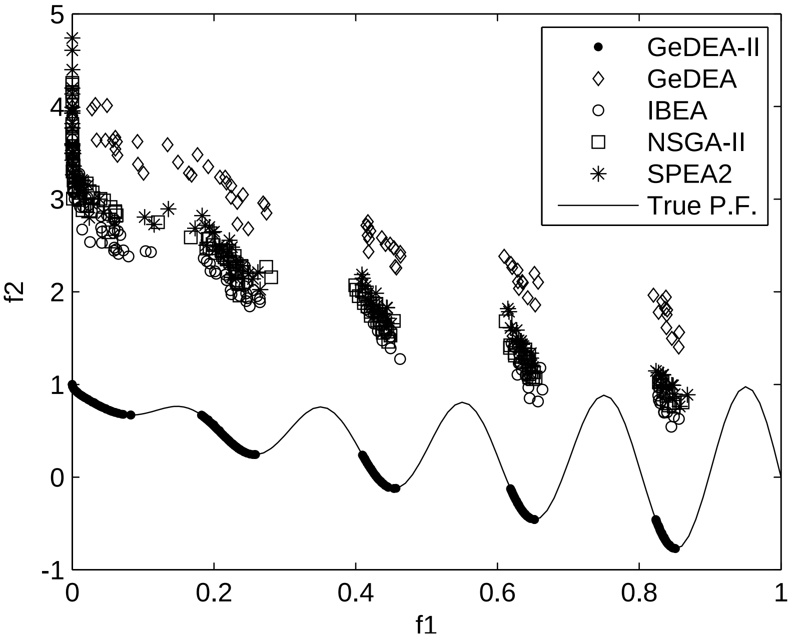
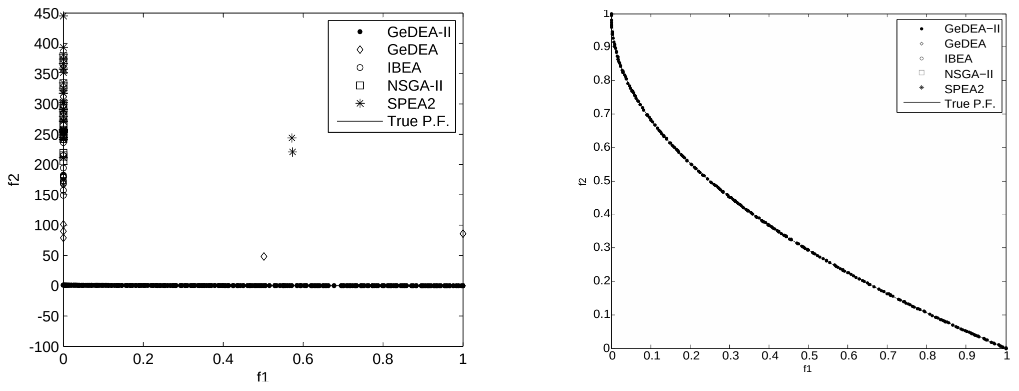


Figure 2.9: Test function T_3 (discrete).



(a) ZDT_4test function. A proper scale in order for all the populations to be included in the plot.

(b) ZDT_4test function. True Pareto Front region.

Figure 2.10: Test function T_4 (multi-modal).

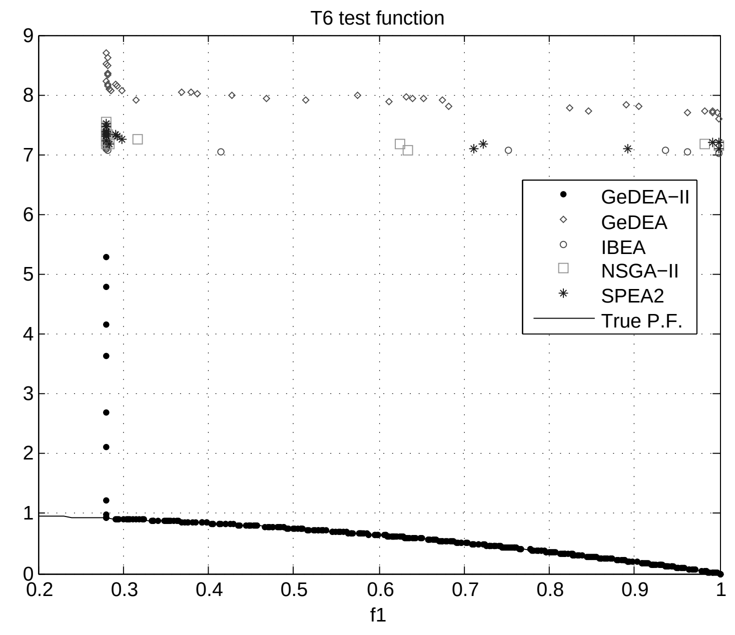


Figure 2.11: Test function T_6 (non-uniform).

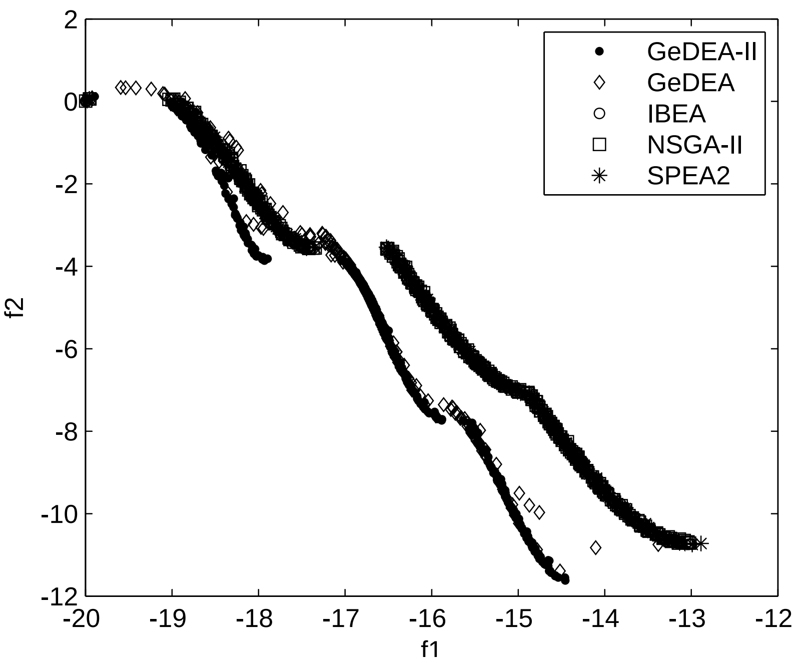
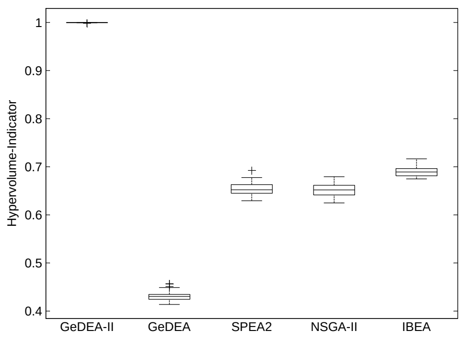
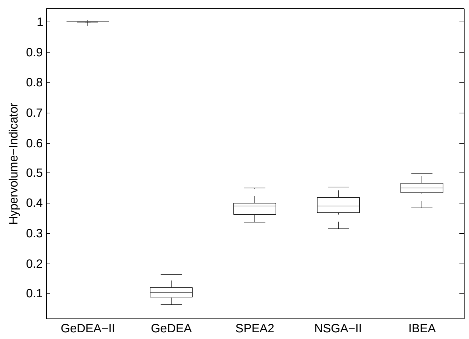


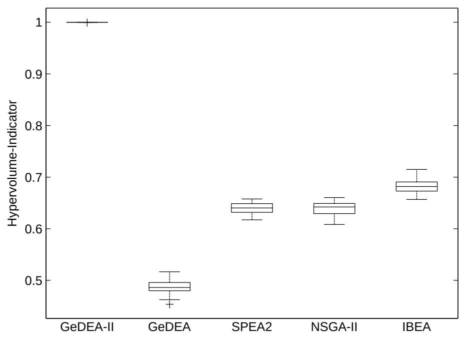
Figure 2.12: Test function *KUR*.



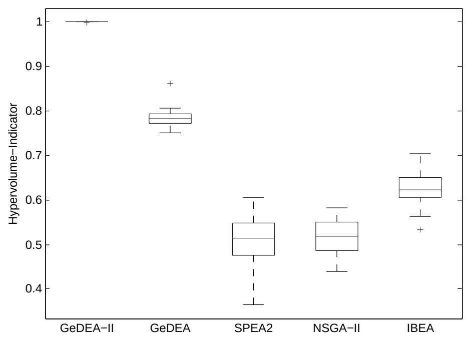
(a) ZDT_1 boxplot



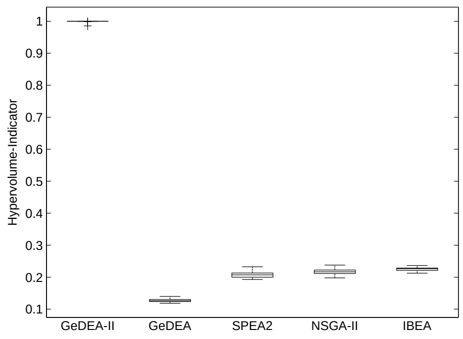
(b) ZDT_2 boxplot



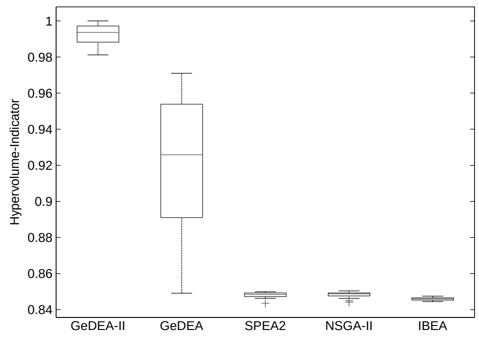
(c) ZDT_3 boxplot



(d) ZDT_4 boxplot

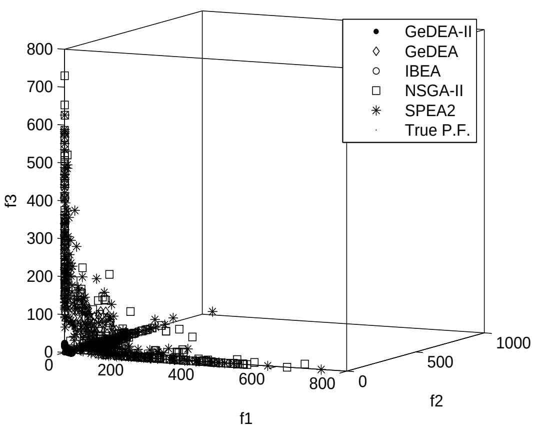


(e) ZDT_6 boxplot

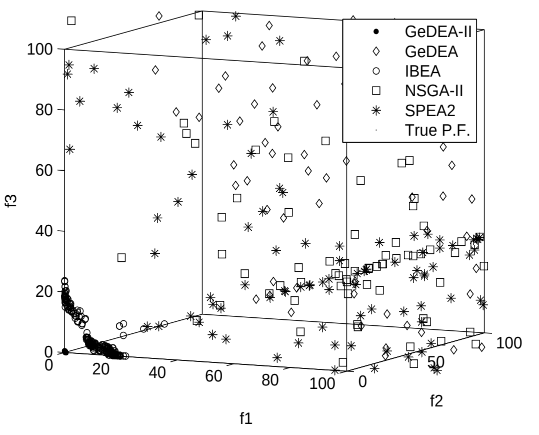


(f) KUR boxplot

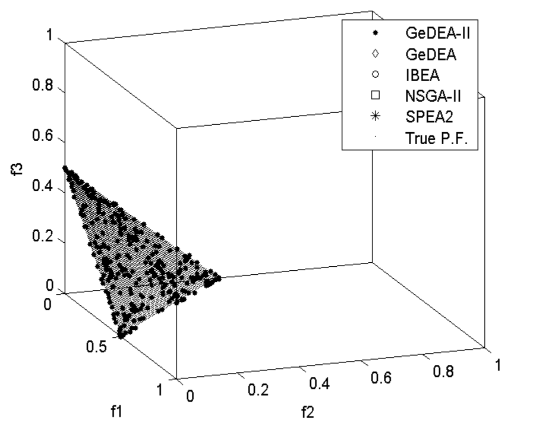
Figure 2.13: Box plots based on the *Hypervolume* metric. Each square contains six box plots representing the distribution of *Hypervolume* values for the six algorithms. Results refer to the *ZDT* test suite.



(a) $DTLZ_1$ test function. A proper scale in order for all the populations to be included in the plot.



(b) $DTLZ_1$ test function. A proper scale in order for the individuals lying in a region closer to the True Pareto Front to be included in the plot.



(c) $DTLZ_1$ test function. True Pareto Front region

Figure 2.14: Test function *DTLZ1*.

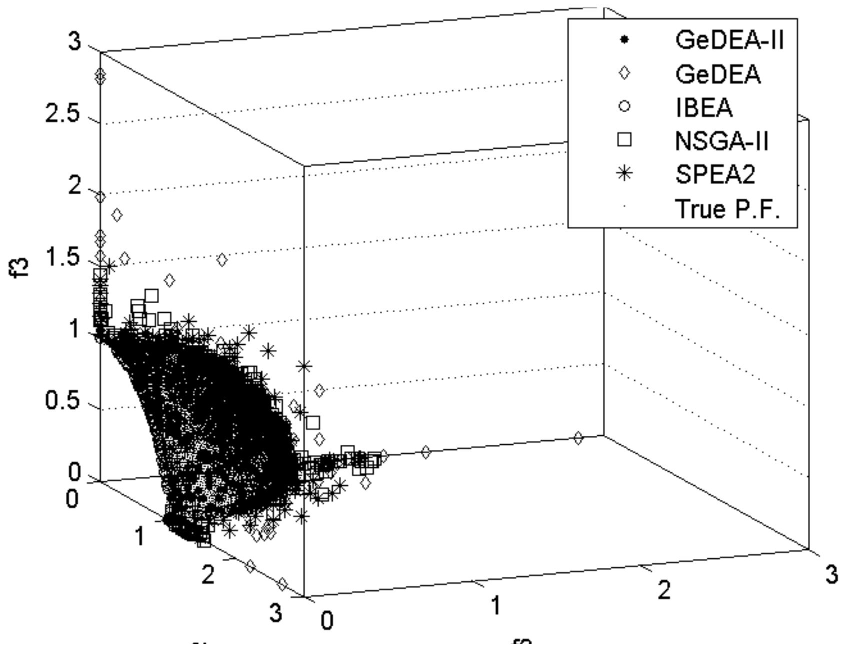


Figure 2.15: Test function $DTLZ_2$.

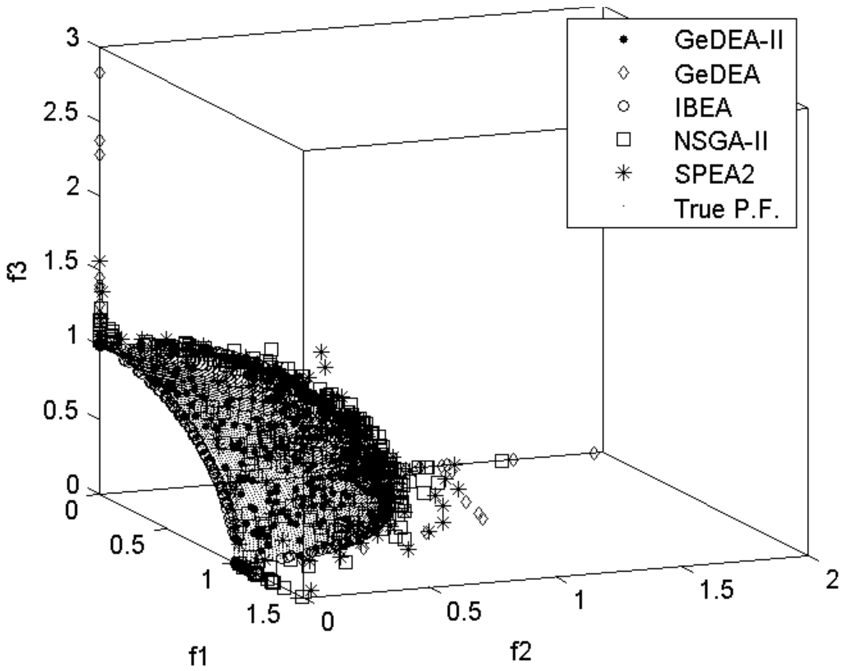
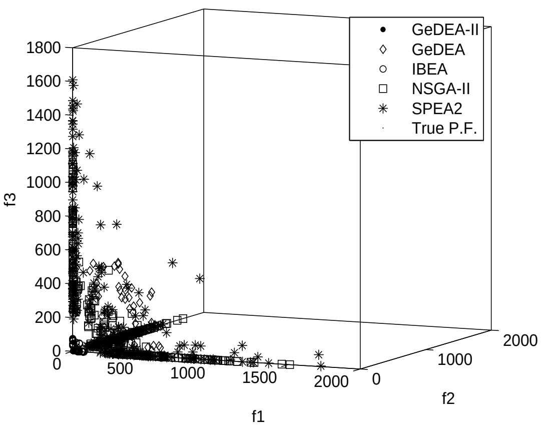
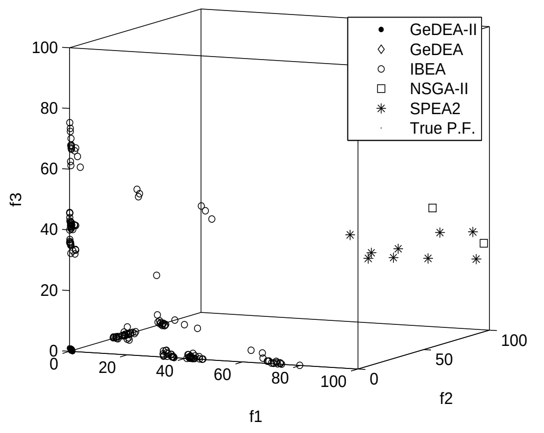


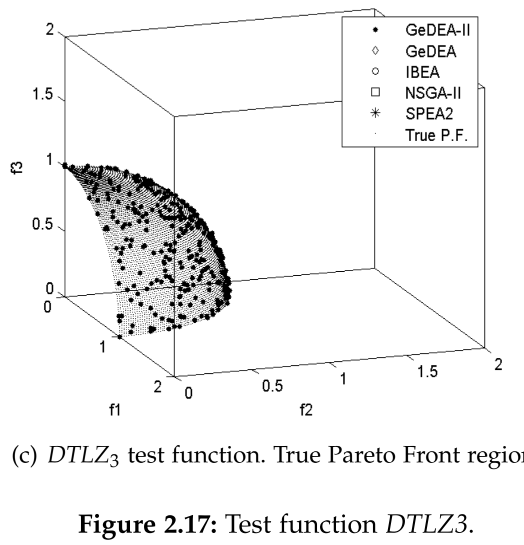
Figure 2.16: Test function $DTLZ_4$.



(a) *DTLZ3* test function. A proper scale in order for all the populations to be included in the plot.



(b) *DTLZ3* test function. A proper scale in order for the individuals lying in a region closer to the True Pareto Front to be included in the plot.



(c) *DTLZ3* test function. True Pareto Front region.

Figure 2.17: Test function *DTLZ3*.

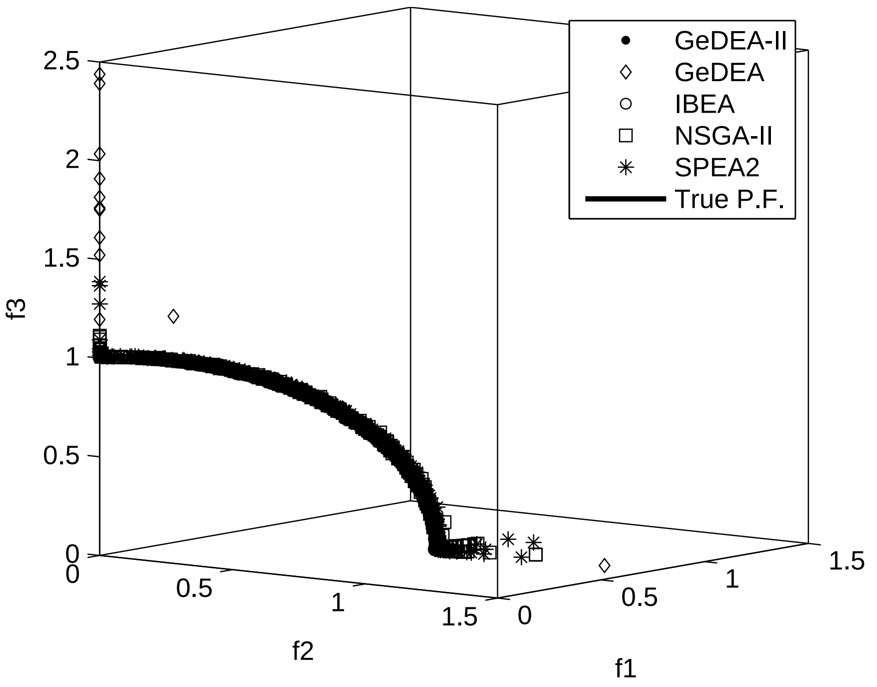


Figure 2.18: Test function *DTLZ*5.

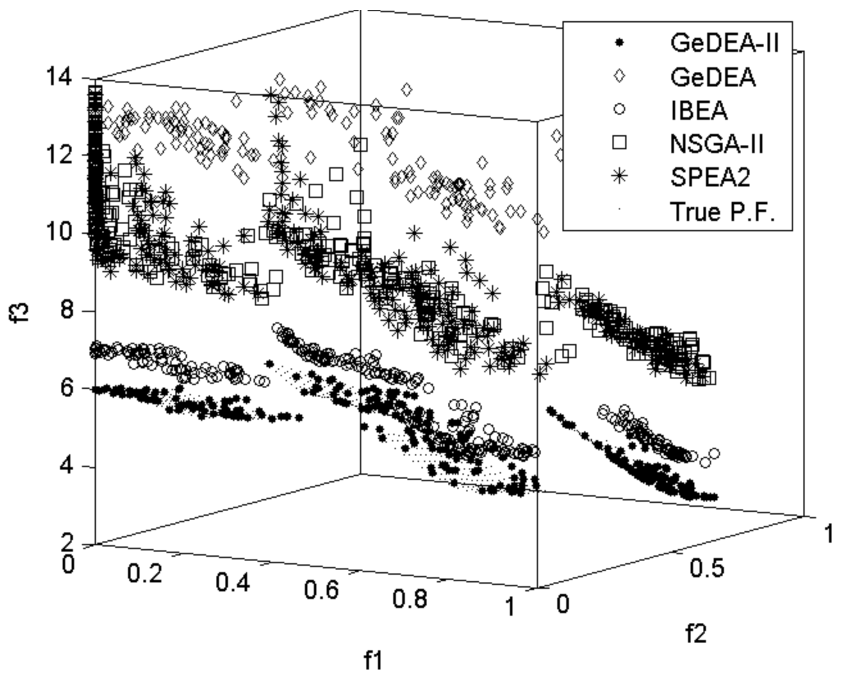
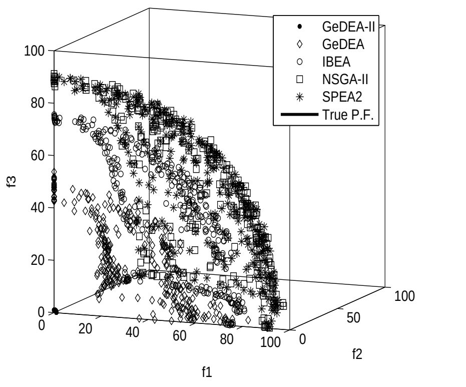
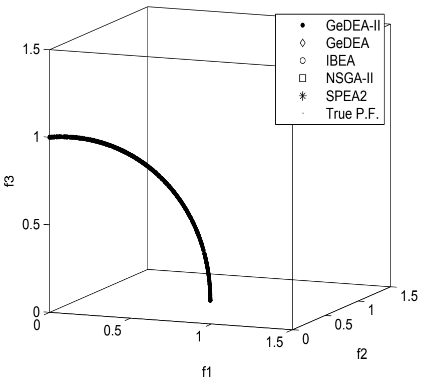


Figure 2.19: Test function *DTLZ7*.

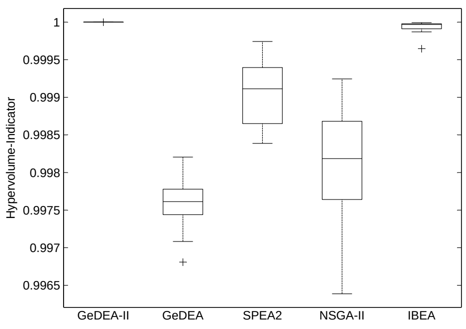


(a) $DTLZ_6$ test function. A proper scale in order for all the populations to be included in the plot.

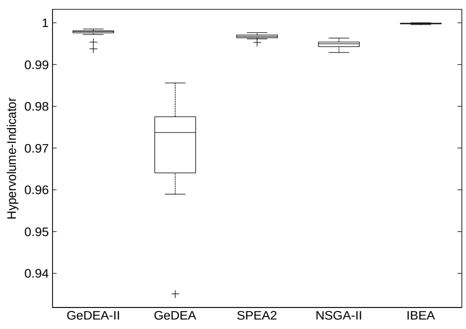


(b) $DTLZ_6$ test function. True Pareto Front region.

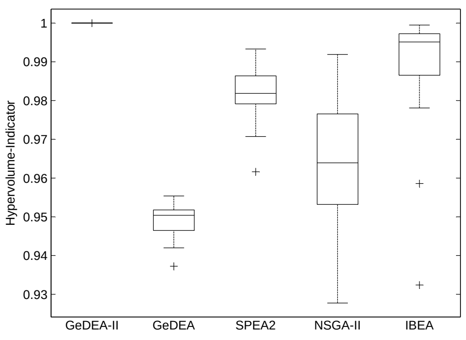
Figure 2.20: Test function *DTLZ6*.



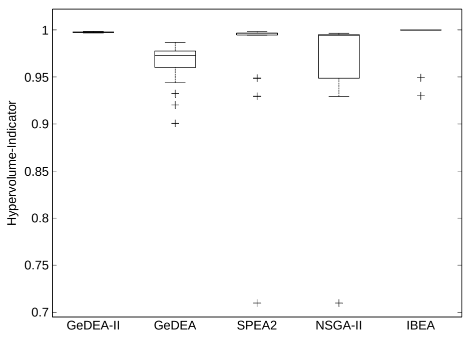
(a) $DTLZ_1$ boxplot



(b) $DTLZ_2$ boxplot



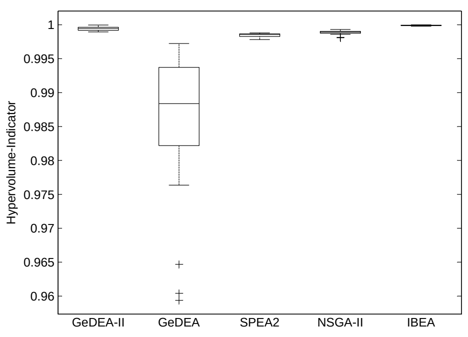
(c) $DTLZ_3$ boxplot



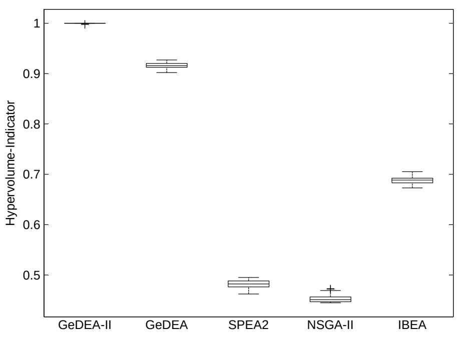
(d) $DTLZ_4$ boxplot

In general, the experimental results show that GeDEA-II is able to converge towards the True Pareto-optimal front and to develop a widely and well distributed non-dominated set of solutions. The comparison with the other three best-performing MOEAs according to the *Hypervolume* metric proves that the performance of GeDEA-II is somewhat superior. Considering the specific features of the six test functions, GeDEA-II shows similar performance both on convex and non-convex Pareto-optimal fronts. NSGA-II, SPEA-2 and IBEA seem instead to have more difficulties with non-convexity, since the non-dominated solutions found on test function T_1 are nearest to the Pareto-optimal front than those found on T_2. Also discreteness (test function T_3) does not affect the performance of the GeDEA-II, which reach the True Pareto Front and completely covers it. The performance of GeDEA-II is particularly remarkable in the case of multi-modality, which is considered by Ziztler et al. [102] as the hardest problem feature. GeDEA-II reaches the Pareto-optimal front even of test function T_4, unlike the competitors, that are far away from the True Pareto front. GeDEA-II outperforms NSGAII and SPEA2 in the case of biased search space (test function T_6) and is also able to evolve a well-distributed non-dominated set. These results gain even more significance, since the number of decision variables was set to 100, unlike the original values of 30 (10 for the test functions T_4 and T_6).

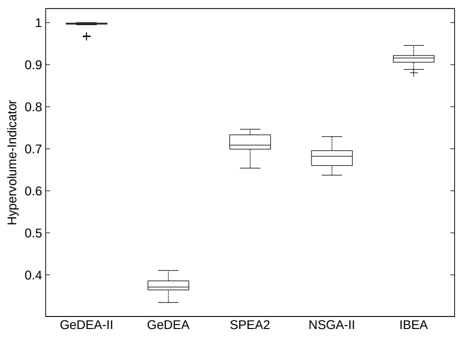
GeDEA-II performs better than the competitors also on Kursawe test function, even if boxplots show that its performance are slightly inferior to those it features



(e) *DTLZ*$_5$ boxplot



(f) *DTLZ*$_6$ boxplot



(g) *DTLZ*$_7$ boxplot

Figure 2.21: Box plots based on the *Hypervolume* metric. Each square contains six box plots representing the distribution of *Hypervolume* values for the six algorithms. Results refer to the *DTLZ* test suite.

on the preceding test functions. This is due to the fact that the variation range of the decision variables was doubled when compared to the original one. Moreover, the presence of the outlier, negligible from a statistical point of view, affects the median value. As far as *DTLZ*$_1$ test function is concerned, GeDEA-II is able to reach the True Pareto Front unlike the competitors. Performance of GeDEA-II are at the same level of those of the competitors on *DTLZ*$_2$ test function, whereas it is once again the only MOEA among those investigated able to reach the True Pareto Front of the *DTLZ*$_3$ test function. GeDEA-II shows similar behaviors to those of the competitors on *DTLZ*$_4$ and *DTLZ*$_4$ test functions. Particularly remarkable are the performance of our algorithm when the last two test functions of the *DTLZ* test suite are considered. GeDEA-II is able to reach the True Pareto Fronts, whereas the competitors remain trapped in the local Pareto Approximation Sets, as shown in Figure2.19 and 2.20(a).

Finally, box plots prove, in general, that the performance of GeDEA-II are superior to those of the competitors also as far as the repeatability of the results is concerned.

2.5 Conclusions

In this chapter the writer has presented GeDEA-II, an improved multi-objective evolutionary algorithm that employs novel genetic operators compared to its predecessor GeDEA as well as a new technique for adapt itself to different multiobjective problems. Extensive numerical comparisons of GeDEA-II with GeDEA and with NSGAII, SPEA-2 and IBEA, three state-of-the-art recently proposed algorithms, have been carried out on various test problems. Moreover, optimization difficulties have been enhanced further, in order to test the robustness of the codes. The key results of the comparison are:

- GeDEA-II reaches the True Pareto fronts on all of the investigated test problems, while covering them in a satisfactory manner.
- GeDEA-II outperforms both its predecessor and the competitors on all the test problems investigated.
- In extremely high dimensional spaces, GeDEA-II clearly shows unparalleled performance over the other algorithms.
- Boxplots shows that the reproducibility of results of GeDEA-II is high-level, when compared to that of the NSGAII, SPEA-2 and IBEA.

In addition to these characteristics, GeDEA-II performs these tasks with a reduced number of objective functions evaluations, which is not negligible when considering its application to real-world engineering problems. In chapter 7, two Test cases will be considered, concerning mono and multi-objective problems where GeDEAII will constitute the optimization engine. Results will demonstrate the performance of GeDEAII.

Chapter 3

Parameterization techniques for 3D shape optimization in aerodynamics

This chapter provides the basic classification of the most common shape deformation techniques employed in the context of the aerodynamic shape optimization. Finally, a brief introduction to Altair HyperMorph is presented, which is the morphing tool chosen by *AgustaWestland* and the *University of Padova* to be used in their optimization research program.

3.1 Shape parameterization techniques: An Overview

A survey of shape parameterization techniques is proposed in [78]. The most used approches are as the following:

- **Discrete Approach**: it is based on using the coordinates of the boundary points as design variables. This approach is easy to implement. However, it is too much connected to the solution mesh and does not enable adaptive-mesh flow evaluations. Due to the excessive refinement of the parametrization, comparable to the resolution of the mesh, it is difficult to maintain a smooth geometry of the optimized shape. For a model with a large number of grid points, the number of design variables often becomes very large, which leads to hight cost and a difficult optimization problem to solve. This technique allows to easily parameterize complex 3D objects, but there are some problems with definition of normals to a discrete surface. Hierarchical parameterization using this approach might be done for example by agglomeration [60].
- **Polynomial and Spline Approaches**: the shape is described by a polynomial curve in a very compact form with a small set of design variables . For example, with a Bezier curve, the control points are used as design variables. Despite recent progress, it is still difficult to parameterize and construct complex, three-dimensional models based only on polynomial and spline

representations. Complex shape requires a large number of control points. The smoothness of the shape deformation is assured.

- **CAD-based Approach**: most solid modelling CAD systems use either a boundary representation (B-Rep) or a constructive solid geometry method to represent a physical, solid object. Based on a complete mathematical definition of a solid, it is possible to create a complete geometry. These systems use Boolean operations such as intersection and union of simple features (e.g. holes, slots, cuts, protrusion, fillets, chamfers, etc.). Existing Feature-based solid modelling (FBSM) CAD tools are not capable of calculating sensitivity derivatives analytically. After reconstructing the shape geometry, usually a new mesh must be generated.

- **Analytical Approach**: the formulation is based on adding shape functions (analytical functions) linearly to the baseline shape. All participating coefficients are initially set to zero, so the first computation gives the baseline geometry. The shape functions are smooth functions based on previous airfoil design. This method results very good for wing parameterization, but not simple to generalize for complex geometries.

- **Free Form Deformation (FFD) Approach**: FFD is one of the techniques of deforming computer-generated objects, which comes from Computer Graphics [80]. The FFD approach operates on the whole space regardless of the representation of the deformed objects embedded in the space. Instead of manipulating the object directly, FFD deforms a lattice that was built around the object. The lattice is a space, in the shape of a cube or an arbitrary volume [3], which wraps around the object. This lattice is basically a composite of Bezier tensor patches in 3D called a Bezier volume, but it is also possible to use B-spline or NURBS [55]. When we move the control points of the lattice, the lattice is deformed. At the same time, the object inside the lattice is deformed (see Figure 3.2).

3.2 The HyperMorph Parameterization Tool

For the purpose of this Thesis the geometry parameterization has been carried out by means of the *Altair HyperMorph* morphing tool, implemented within the *Altair HyperMesh* environment. This is a commercial software, distributed by *Altair Engineering*, capable to assign parametric deformations to every kind of finite elements model, allowing the designer to set up design of experiments and optimization studies in a very simple way. *HyperMorph* provides a wide range of warping and morphing techniques belonging to the first and the last groups mentioned in section 3.1. See reference [3] to learn more about *HyperMorph* shape deformation capabilities. In particular, as it will be see later on, the most effective methods for aerodynamic optimization purpose are the *Domain-Handles* and the *Morph-volume* approaches, both belonging to the powerfull *FFD* technique

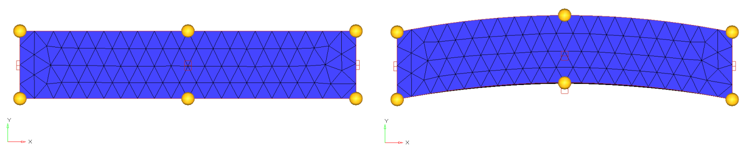

(a) Base model (b) Morphed model

Figure 3.1: Handles-Domain technique.

briefly discussed in the following sections. They will be both exploited during the optimization works.

3.2.1 Domains-Handles Approach

When the Domains-handles approach is used, the model is divided into domains where handles are used to control the relative domain shapes. When the handles are moved, the shape of the domains touching those handles change, which in turn, changes the positions of the nodes inside those domains. During the morphing process the mesh morphs in a logical way with nodes near the moving handles moving more and nodes near the stationary handles moving less. In the area between the handles, the mesh is stretched or compressed to match the desired shape. An example of application of the *Handles-Domains* approach is shown in Figure 3.1. In this example the prisms shown in figure are placed in a 3D domain which is then partitioned in six 3D domain, corresponding to the six faces of the prism, by the auto-partition algorithm [3]. This process generate the yellow handles on the corners of the prisms. Then some handles are manually placed in the middle section of the prism (Figure 3.1-a); the vertical displacement of these handles leads in the morphed model shown in Figure 3.1-b.

As regards aerodynamic shape optimization, the *Handles-Domains* approach is the best way to generate a parametric model when small, local displacements are required, and a precise control of the deformed region has to be achieved.

3.2.2 Morph-Volume Approach

When the morph volumes approach is used, the model is enclosed in some control volumes, the *Morph-Volumes*, which are characterized by the presence of *Global Handles* on their corners. When some of these handles are moved, the nodes and elements inside the corresponding Morph-Volumes are accordingly moved in a really smooth manner. The same prism, previously deformed with the *Domain-Handles* (Figure 3.1), is now dealt with the *Morph-Volumes* (Figure 3.2). In this example, two Morph-Volumes are created by means of the *Pick on Screen* tool [3]. The volumes can be visualized in Figure 3.2 as the green regions around the prism. The morphed model is obtained by assigning a vertical displacement to the central located, red, handles. Because of the second order continuity in the deformed mesh achievable with this morph-strategy, Morph-Volumes are really suited, for

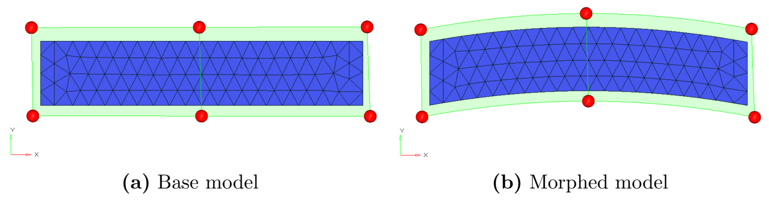


Figure 3.2: Morph-Volumes application technique.

aerodynamic optimization purpose, when global and smooth shape changes are required.

3.2.3 Shapes as Design Variables

Into HyperMorph environment, shapes are collection of handles and/or nodes perturbations. When a deformation on a given model is carried out, HyperMorph stores the morph internally as a collection of perturbations. Saving a shape, obtained by overlapping more simpler morphing operations, the handles and/or nodes perturbations are stored in a new shape entity, which can be applied to the undeformed model with any given scaling factor. Therefore, shapes created in HyperMorph are suited to be use as parametric design variables in DOE and optimization studies.

Chapter 4

Introduction to CFD

This chapter is intended as an introduction for Computational Fluid Dynamics (CFD). Due to its introductory nature, only the basic principles of CFD are introduced here. For a more detailed description, readers are referred to the precious guide present in [94].

4.1 Introduction

Even though a univocal definition of CFD does not exist, here we can state that CFD deals with the simulation of fluids engineering systems using modeling (mathematical physical problem formulation) and numerical methods (discretization methods, solvers, numerical parameters, and grid generations, etc.). CFD provides a numerical approximation to the equations that govern fluid motion. Application of the CFD to analyze a fluid problem requires the following steps.

First, the mathematical equations describing the fluid flow are written. These are usually a set of partial differential equations.

These equations are then discretized to produce a numerical analogue of the equations.

The domain is then divided into small grids or elements.

Finally, the initial conditions and the boundary conditions of the specific problem are used to solve these equations. The solution method can be direct or iterative. In addition, certain control parameters are used to control the convergence, stability, and accuracy of the method.

All CFD codes contain three main elements:

- A **pre-processor**, which is used to input the problem geometry, generate the grid, define the flow parameter and the boundary conditions to the code.
- A **flow solver**, which is used to solve the governing equations of the flow subject to the conditions provided. There are four different methods used as a flow solver:
    * The finite difference method;
    * The finite element method;

  * The finite volume;
  * The spectral method.

- A **post-processor**, which is used to manage the data and show the results in graphical and easy to read format.

In this chapter we are mainly concerned with the flow solver part of CFD, in particular by paying attention on the system of equations to be solved. A brief description of the RANS technique and of the finite-volume method will be finally undertaken.

4.2 Governing equations

The *Navier-Stokes equations* are the basic governing equations for a viscous, heat conducting fluid. It is a vector equation obtained by applying Newton's Law of Motion to a fluid element and is also called the *momentum equation*. It is supplemented by the *mass conservation equation*, also called continuity equation and the *energy equation*. Usually, the term Navier-Stokes equations is used to refer to all of these equations. In addition to these equations, further equation can be solved, like, for example, the equations of state for real or ideal gas, cavitation models, magnetohydrodynamics models, and so on.

However, in this chapter, only the Navier-Stokes equations will be considered.

4.2.1 The Continuity Equation

The equations governing the fluid motion are the three fundamental principles of mass, momentum, and energy conservation. Let start with the first principle, which states that

...the total amount of mass inside any region can only change by the amount that passes in or out of the region through the boundary.

One considers the infinitesimal control volume depicted in Figure 4.1.

The *law of conservation of mass* states that

$$\frac{dm}{dt} = \sum_{in} \dot{m} - \sum_{out} \dot{m} \tag{4.1}$$

In the three-dimensional space, the Eq. 4.1 assumes the following form:

$$\begin{aligned}
\frac{\partial \rho \Delta x \Delta y \Delta z}{\partial t} &= (\rho\, u)\Delta y \Delta z + (\rho\, v)\Delta x \Delta z + (\rho\, w)\Delta x \Delta y \\
&- \left[(\rho\, u) + \frac{\partial(\rho u)}{\partial x}\Delta x\right] \Delta y \Delta z \\
&- \left[(\rho\, v) + \frac{\partial(\rho v)}{\partial y}\Delta y\right] \Delta x \Delta z \\
&- \left[(\rho\, w) + \frac{\partial(\rho w)}{\partial z}\Delta z\right] \Delta x \Delta y
\end{aligned} \tag{4.2}$$

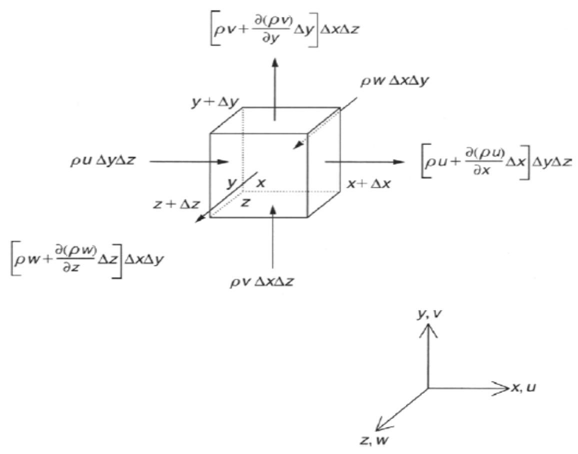

$\left[\rho v+\frac{\partial(\rho v)}{\partial y}\Delta y\right]\Delta x\Delta z$
$\rho w\,\Delta x\Delta y$
$y+\Delta y$
$\rho u\,\Delta y\Delta z$
$\left[\rho u+\frac{\partial(\rho u)}{\partial x}\Delta x\right]\Delta y\Delta z$
y
x
$x+\Delta x$
$z+\Delta z$
z
$\left[\rho w+\frac{\partial(\rho w)}{\partial z}\Delta z\right]\Delta x\Delta y$
$\rho v\,\Delta x\Delta z$
y, v
x, u
z, w

Figure 4.1: An infinitesimal fluid control volume - The Continuity equation.

which, after some rearrangements is given the following form:

$$\frac{\partial \rho}{\partial t}+\frac{\partial(\rho u)}{\partial x}+\frac{\partial(\rho v)}{\partial y}+\frac{\partial(\rho w)}{\partial z}=0. \tag{4.3}$$

When the flow is at steady-state, ρ does not change with respect to time. The continuity equation is reduced to:

$$\frac{\partial(\rho u)}{\partial x}+\frac{\partial(\rho v)}{\partial y}+\frac{\partial(\rho w)}{\partial z}=0. \tag{4.4}$$

When the flow is incompressible, ρ is constant and does not change with respect to space. The continuity equation is reduced to:

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}=0. \tag{4.5}$$

By using the substantive derivative[8] concept, the mass continuity equation assumes the general form:

$$\frac{D\rho}{Dt}+\rho(\nabla\cdot\mathbf{v})=0. \tag{4.6}$$

[8]The *substantial derivative* is a derivative taken along a path moving with velocity v, and is often used in fluid mechanics and classical mechanics. It describes the time rate of change of some quantity (such as heat or momentum) by following it, while moving with a space- and time-dependent velocity field. For example, in fluid dynamics, take the case that the velocity field under consideration is the flow velocity itself, and the quantity of interest is the temperature of the fluid. Then the material derivative describes the temperature evolution of a certain fluid parcel in time, as it is being moved along its pathline (trajectory) while following the fluid flow. The *total or substantial* derivative assumes the following mathematical expression, for a scalar, φ, on the left, or a vector, $\mathbf{u}$, on the right:

$$\frac{D\varphi}{Dt}=\frac{\partial\varphi}{\partial t}+\mathbf{v}\cdot\nabla\varphi, \qquad \frac{D\mathbf{u}}{Dt}=\frac{\partial\mathbf{u}}{\partial t}+\mathbf{v}\cdot\nabla\mathbf{u}$$

4.2.2 The Momentum Equation

The principle of *momentum conservation* comes directly from the Newton's second law, which states that

...the net force on a particle is equal to the time rate of change of its linear momentum p in an inertial reference frame.

is given the following mathematical expression:

$$\vec{F} = m\,\vec{a} \tag{4.7}$$

Let one apply the principle stated in Eq. 4.7 to the infinitesimal, space fixed control volume depicted in Figure 4.2. The forces to be considered are the *superficial forces*

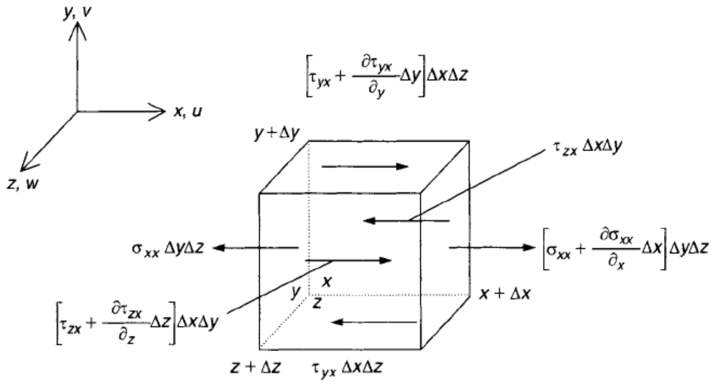

y, v
x, u
z, w
$\left[\tau_{yx} + \frac{\partial \tau_{yx}}{\partial_y}\Delta y\right]\Delta x \Delta z$
$y+\Delta y$
$\tau_{zx}\,\Delta x \Delta y$
$\sigma_{xx}\,\Delta y \Delta z$
$\left[\sigma_{xx} + \frac{\partial \sigma_{xx}}{\partial_x}\Delta x\right]\Delta y \Delta z$
$\left[\tau_{zx} + \frac{\partial \tau_{zx}}{\partial_z}\Delta z\right]\Delta x \Delta y$
y
x
z
$x + \Delta x$
$z + \Delta z$
$\tau_{yx}\,\Delta x \Delta z$

Figure 4.2: An infinitesimal fluid control volume - The Momentum Equation.

(normal and tangential to the body) and the volume forces, such as the gravity forces. The Eq. 4.8 and 4.9 represent the overall normal and tangential stresses, respectively, acting on the control volume depicted Figure 4.2, along x-direction.

$$\left[\sigma_{xx} + \frac{\partial \sigma_{xx}}{\partial x}\Delta x\right]\Delta y \Delta z - \sigma_{xx}\Delta y \Delta z \quad \textit{overall normal stresses acting on the control volume along x-direction} \tag{4.8}$$

$$\begin{aligned} &\left[\tau_{yx} + \frac{\partial \tau_{yx}}{\partial y}\Delta y\right]\Delta x \Delta z - \tau_{yx}\Delta x \Delta z \quad \textit{overall tangential stresses acting on} \\ &\left[\tau_{zx} + \frac{\partial \tau_{zx}}{\partial z}\Delta z\right]\Delta x \Delta y - \tau_{zx}\Delta x \Delta y \quad \textit{the control volume along x-direction} \end{aligned} \tag{4.9}$$

By adding the terms in Eq. 4.8 and 4.9 and dividing by the volume, one can obtain

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} \quad \textit{results of the superficial stress along x-direction (per volume unit)} \tag{4.10}$$

The Newton's second law, evaluated along x-direction, becomes

$$\rho \frac{Du}{Dt} = \frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} + \sum F_x^{bodyforces} \tag{4.11}$$

Similarly, along y-direction and y-direction the conservation of momentum gives, respectively,

$$\begin{aligned} \rho \frac{Dv}{Dt} &= \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \tau_{zy}}{\partial z} + \sum F_y^{bodyforces} \\ \rho \frac{Dw}{Dt} &= \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} + \sum F_z^{bodyforces} \end{aligned} \tag{4.12}$$

Consider now a Newtonian[9] and isotropic[10] fluid. Thanks to this assumption, it can be stated that

$$\begin{aligned} \sigma_{xx} &= -p + \tau_{xx} \\ \sigma_{yy} &= -p + \tau_{yy} \\ \sigma_{zz} &= -p + \tau_{zz} \\ \tau_{xx} &= 2\mu \frac{\partial u}{\partial x} + \lambda \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \\ \tau_{yy} &= 2\mu \frac{\partial v}{\partial y} + \lambda \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \\ \tau_{zz} &= 2\mu \frac{\partial z}{\partial z} + \lambda \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \\ \tau_{xy} &= \tau_{yx} = \mu \left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right) \\ \tau_{xz} &= \tau_{zx} = \mu \left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z} \right) \\ \tau_{yz} &= \tau_{zy} = \mu \left(\frac{\partial w}{\partial y} + \frac{\partial v}{\partial z} \right) \end{aligned} \tag{4.13}$$

λ is the second coefficient of viscosity (related to *bulk viscosity*). The value of λ, which produces a viscous effect associated with volume change, is very difficult to determine, not even its sign is known with absolute certainty. Even in compressible flows, the term involving λ is often negligible; however it can occasionally be important even in nearly incompressible flows and is a matter of controversy. When taken nonzero, the most common approximation is $\lambda \approx -2/3\mu$ [12].

[9] A *Newtonian fluid* (named after Isaac Newton) is a fluid whose shear stress τ versus strain rate $\frac{\partial u}{\partial y}$ curve is linear and passes through the origin. The constant of proportionality is known as the *dynamic viscosity* μ.

[10] A fluid whose properties are not dependent on the direction along which they are measured.

By substituting relations given in 4.13, in Eq. 4.11 and 4.12, one can obtain the *Navier-Stokes equations.*

$$\begin{aligned}
\rho\frac{Du}{Dt} &= -\frac{\partial p}{\partial x} + \frac{\partial}{\partial x}\left[2\mu\frac{\partial u}{\partial x} + \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\right] \\
&\quad \frac{\partial}{\partial y}\left[\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right] + \frac{\partial}{\partial z}\left[\mu\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right)\right] + \sum F_x^{bodyforces} \\
\rho\frac{Dv}{Dt} &= -\frac{\partial p}{\partial y} + \frac{\partial}{\partial y}\left[2\mu\frac{\partial v}{\partial y} + \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\right] \\
&\quad \frac{\partial}{\partial x}\left[\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right] + \frac{\partial}{\partial z}\left[\mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right)\right] + \sum F_y^{bodyforces} \\
\rho\frac{Dw}{Dt} &= -\frac{\partial p}{\partial z} + \frac{\partial}{\partial z}\left[2\mu\frac{\partial w}{\partial z} + \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\right] \\
&\quad \frac{\partial}{\partial x}\left[\mu\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right)\right] + \frac{\partial}{\partial y}\left[\mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right)\right] + \sum F_z^{bodyforces}
\end{aligned} \tag{4.14}$$

or, after accounting for the gravity as a body force, and developing the substantial derivatives,

$$\begin{aligned}
\rho\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial z}\right) &= -\frac{\partial p}{\partial x} \\
&\quad + \mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) + \rho g_x \\
\rho\left(\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial z}\right) &= -\frac{\partial p}{\partial y} \\
&\quad + \mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2}\right) + \rho g_y \\
\rho\left(\frac{\partial w}{\partial t} + u\frac{\partial w}{\partial x} + v\frac{\partial w}{\partial y} + w\frac{\partial w}{\partial z}\right) &= -\frac{\partial p}{\partial z} \\
&\quad + \mu\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}\right) + \rho g_z.
\end{aligned} \tag{4.15}$$

Finally, the vector form of the Momentum equation is presented in Eq.4.16, in order to draw a parallel with the Eq. 4.6

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v}\cdot\nabla\mathbf{v}\right) = -\nabla p + \mu\nabla^2\mathbf{v} + (\lambda+\mu)\nabla\mathbf{v} + \mathbf{f}. \tag{4.16}$$

4.2.3 The Energy Equation

The principle of *Energy conservation* comes directly from the First Thermodynamic law, which states that

There is a state function E, called "energy", whose differential equals the work exchanged with the surroundings during an adiabatic process.[11]

[11]This definition was given by Rudolf Clausius in 1850.

Mathematically, one writes

$$\Delta E = W + Q \tag{4.17}$$

where E is the total energy of the system, whereas Q and W are the amounts of heat supplied to the system and work done by the system, respectively.

Let one consider the contribution given by work, W, in Eq. 4.17, applied to the infinitesimal, space fixed control volume depicted in Figure 4.3.

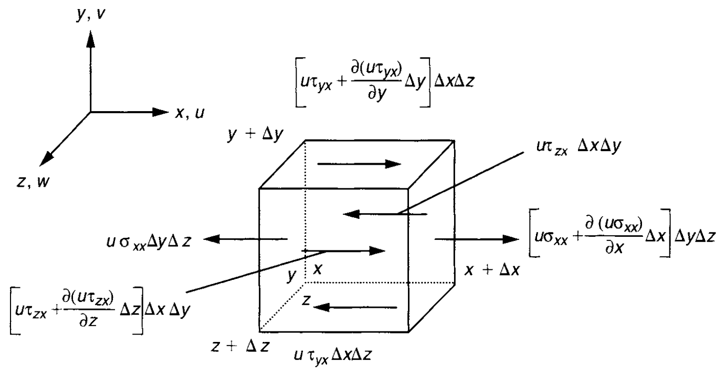

y, v
x, u
z, w
$\left[u\tau_{yx} + \frac{\partial(u\tau_{yx})}{\partial y}\Delta y\right]\Delta x\Delta z$
$y + \Delta y$
$u\tau_{zx}\ \Delta x\Delta y$
$u\sigma_{xx}\Delta y\Delta z$
$\left[u\sigma_{xx} + \frac{\partial(u\sigma_{xx})}{\partial x}\Delta x\right]\Delta y\Delta z$
$\left[u\tau_{zx} + \frac{\partial(u\tau_{zx})}{\partial z}\Delta z\right]\Delta x\,\Delta y$
y
x
z
$x + \Delta x$
$z + \Delta z$
$u\,\tau_{yx}\Delta x\Delta z$

Figure 4.3: An infinitesimal fluid control volume. The Work contribution to the Energy equation.

Let one consider both work and energy terms, separately. The forces considered are the *superficial forces* (normal and tangential to the body). Work done (per time unit) by the normal and tangential stresses acting on the control volume depicted Figure 4.3, along x-direction, is given in Eq. 4.18 and 4.19, respectively.

$$\left[u\sigma_{xx} + \frac{\partial u\sigma_{xx}}{\partial x}\Delta x\right]\Delta y\Delta z - u\sigma_{xx}\Delta y\Delta z \quad \textit{Work done by the normal stresses acting on the control volume along x-direction} \tag{4.18}$$

$$\begin{aligned}&\left[u\tau_{yx} + \frac{\partial u\tau_{yx}}{\partial y}\Delta y\right]\Delta x\Delta z - u\tau_{yx}\Delta x\Delta z \quad \textit{Work done by the tangential stresses acting on}\\ &\left[u\tau_{zx} + \frac{\partial u\tau_{zx}}{\partial z}\Delta z\right]\Delta x\Delta y - u\tau_{zx}\Delta x\Delta y \quad \textit{the control volume along x direction}\end{aligned} \tag{4.19}$$

By adding the terms in Eq. 4.18 and 4.19 and dividing by the volume, one can obtain

$$\frac{\partial u\sigma_{xx}}{\partial x} + \frac{\partial u\tau_{yx}}{\partial y} + \frac{\partial u\tau_{zx}}{\partial z} \quad \textit{Resulting Work done by the superficial stress along x-direction (per volume and time unit)} \tag{4.20}$$

Similarly, along y-direction and y-direction, the Work done by superficial forces is, respectively,

$$\frac{\partial v\tau_{xy}}{\partial x} + \frac{\partial v\sigma_{yy}}{\partial y} + \frac{\partial v\tau_{zy}}{\partial z} \quad \textit{Resulting Work done by the the superficial stress along y-direction (per volume and time unit)} \tag{4.21}$$

$$\frac{\partial w\sigma_{xz}}{\partial x} + \frac{\partial w\tau_{yz}}{\partial y} + \frac{\partial w\sigma_{zz}}{\partial z} \quad \textit{Resulting Work done by the the superficial stress along z-direction (per volume and time unit)} \tag{4.22}$$

Now, let us consider the contribution given by Heat, Q, by referring to the infinitesimal control volume depicted in Figure 4.4.

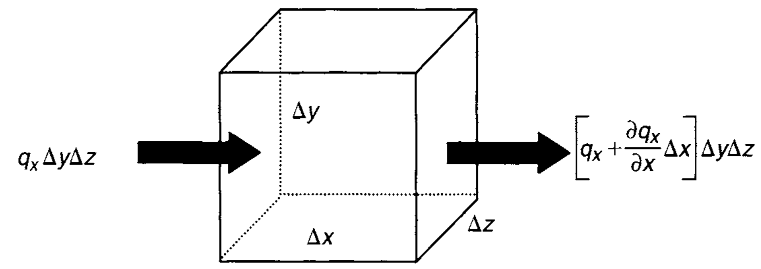

$q_x \Delta y \Delta z$
Δy
Δx
Δz
$\left[q_x + \frac{\partial q_x}{\partial x}\Delta x\right]\Delta y \Delta z$

Figure 4.4: An infinitesimal fluid control volume. The Heat contribution to the Energy equation.

The net Heat flux, along *x-direction*, is

$$\left[q_x + \frac{\partial q_x}{\partial x}\Delta x\right]\Delta y\Delta z - q_x\Delta y\Delta z \quad \textit{Heat flux along x-direction} \tag{4.23}$$

Similarly, along y-direction and z-direction, the Heat flux is, respectively,

$$\left[q_y + \frac{\partial q_y}{\partial y}\Delta y\right]\Delta x\Delta z - q_y\Delta x\Delta z \quad \textit{Heat flux along y-direction} \tag{4.24}$$

$$\left[q_z + \frac{\partial q_z}{\partial z}\Delta z\right]\Delta x\Delta y - q_z\Delta x\Delta y \quad \textit{Heat flux along z-direction} \tag{4.25}$$

By adding the terms in Eq. 4.23, 4.24 and 4.25, and dividing by the volume, one can obtain the overall heat flux crossing control volume in Figure 4.4:

$$\frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y} + \frac{\partial q_z}{\partial z} \tag{4.26}$$

By gathering all the terms in Eqs. 4.20, 4.21, 4.22 and 4.26, the First Thermodynamic law, can be stated as:

$$\rho\frac{DE}{Dt} = \frac{\partial u\sigma_{xx}}{\partial x} + \frac{\partial v\sigma_{yy}}{\partial y} + \frac{\partial w\sigma_{zz}}{\partial z} + \frac{\partial u\tau_{yx}}{\partial y} + \frac{\partial u\tau_{zx}}{\partial z} + \frac{\partial v\tau_{xy}}{\partial x}$$
$$+\frac{\partial v\tau_{zy}}{\partial z} + \frac{\partial w\tau_{xz}}{\partial x} + \frac{\partial w\tau_{yz}}{\partial y} - \frac{\partial q_x}{\partial x} - \frac{\partial q_y}{\partial y} - \frac{\partial q_z}{\partial z} \tag{4.27}$$

By considering the Fourier's Law of Conduction for the thermal fluxes:

$$q_x = -k\frac{\partial T}{x} \quad q_y = -k\frac{\partial T}{y} \quad q_z = -k\frac{\partial T}{z}, \tag{4.28}$$

where k is the referred to as *thermal conductivity*, and by resolving normal stresses as indicated in Eq. 4.13, one can finally obtain:

$$\rho\frac{DE}{Dt} = \frac{\partial}{\partial x}\left(k\frac{\partial T}{x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial T}{y}\right) + \frac{\partial}{\partial z}\left(k\frac{\partial T}{z}\right)$$
$$-\frac{\partial up}{\partial x} - \frac{\partial vp}{\partial y} - \frac{\partial wp}{\partial z} + \Phi \tag{4.29}$$

Φ collects the following terms and takes into account the energy dissipation due to the fluid viscosity (that is, the heat created because of friction, at the expense of mechanical energy).

$$\begin{aligned}\Phi = &\frac{\partial u\tau_{xx}}{\partial x} + \frac{\partial u\tau_{yx}}{\partial y} + \frac{\partial u\tau_{zx}}{\partial z}\\ &+ \frac{\partial v\tau_{xy}}{\partial x} + \frac{\partial v\tau_{yy}}{\partial y} + \frac{\partial v\tau_{zy}}{\partial z}\\ &+ \frac{\partial w\tau_{xz}}{\partial x} + \frac{\partial w\tau_{yz}}{\partial y} + \frac{\partial w\tau_{zz}}{\partial z}\end{aligned} \tag{4.30}$$

Now, it is worth noting that for a compressible flow, it can be stated that

$$h = e + \frac{p}{\rho} + \frac{1}{2}\left(u^2 + v^2 + w^2\right) = E + \frac{p}{\rho} \tag{4.31}$$

where h is the fluid *total enthalpy*, whereas e is the *internal energy*.
By neglecting kinetic contribution,

$$h = h_{st} = c_p T \tag{4.32}$$

where c_p is the specific heat capacity at constant pressure, not temperature dependent.

Finally, let us consider the special case, where the fluid is incompressible and the continuity equation applies. As we said before, by neglecting the kinetic energy the enthalpy h can be reduced to CpT, where Cp is the specific heat and is assumed to be constant. Equation 4.29 can be expressed as

$$\rho Cp\frac{DT}{Dt}=\frac{\partial}{\partial x}\left(k\frac{\partial T}{x}\right)+\frac{\partial}{\partial y}\left(k\frac{\partial T}{y}\right)+\frac{\partial}{\partial z}\left(k\frac{\partial T}{z}\right)+\frac{\partial p}{\partial t}+\Phi \tag{4.33}$$

4.3 Turbulence Modeling

There are two radically different states of flows that are easily identified and distinguished: *laminar flow* and *turbulent flow*.

Laminar flows are characterized by smoothly varying velocity fields in space and time in which individual "laminae" (sheets) move past one another without generating cross currents. These flows arise when the fluid viscosity is sufficiently large to damp out any perturbations to the flow that may occur due to boundary imperfections or other irregularities. These flows occur at low-to-moderate values of the Reynolds number.

In contrast, *turbulent flows* are characterized by large, 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 viscosity. Turbulent flows occur in the opposite limit of high Reynolds numbers. In Figure 4.5, the laminar-to-turbulent transition over a submarine hull is depicted[12]. In the upstream region the flow is laminar, but soon after the transition to turbulent flow occurs. One of the most important characteristics of turbulent flow

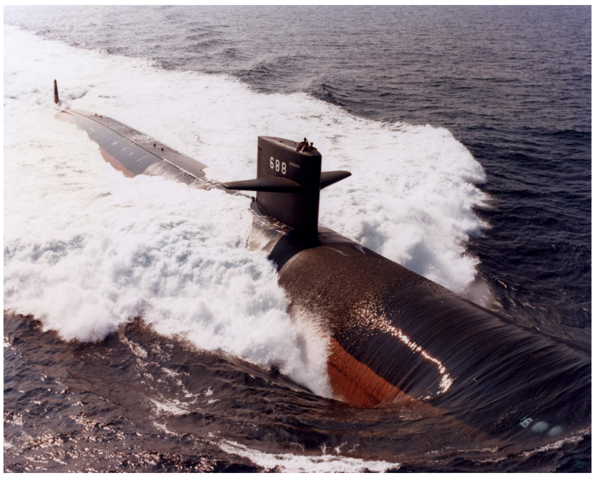


Figure 4.5: Laminar-to-turbulent transition over a submarine hull (image taken from the NavSource Naval History website).

concerns its dissipative behaviour, when compared to that characterizing laminar flow. In Figure 4.6, the skin friction coefficient of a NACA 23012 airfoil is plotted. Results are obtained with XFOIL CFD code[13]. The sudden increase of the friction

[12]Please visit the website `http://www.navsource.org/archives/08/08688.htm`

[13]XFOIL is an interactive program for the design and analysis of subsonic isolated airfoils. Please visit website `http://web.mit.edu/drela/Public/web/xfoil/`

coefficient once the transition is occurred is clearly visible.

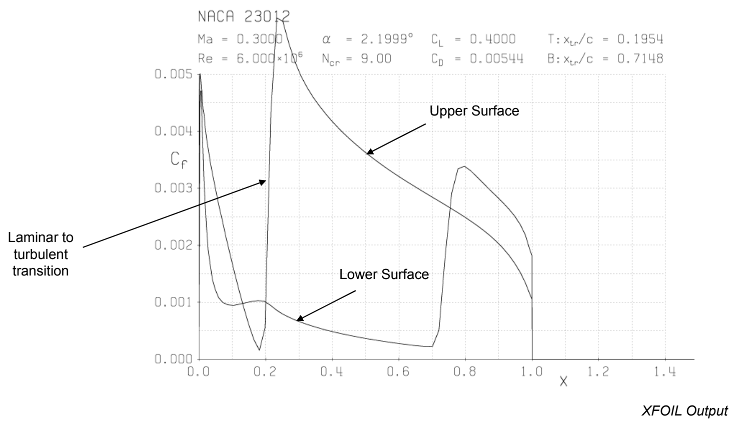


Figure 4.6: Skin friction coefficient of NACA 23012 airfoil. XFOIL output

Another distinctive feature of turbulent flows is that turbulent boundary layer is more "energetic" than the laminar one. Thanks to this characteristics, turbulent flows can withstand higher adverse pressure gradients than those sustainable by laminar flows. This aspect of turbulent flows is exploited in devices such as *vortex generator* or *synthetic jets*, which will be described in Sections 5.5.2 and 5.5.1, respectively. Possible applications of such devices regard *turbomachinery aerodynamics* (to postpone or delay flow separation means increased performances) and *external aerodynamics* (to postpone or delay flow separation means reduced drag and, in general, better flight performances).

Let one reconsider now the laminar-to-turbulent transition phenomenon introduced above and look at Figure 4.7, which shows an induced transition experiment. A typical time history of the flow variable u at a fixed point in space is shown in the lower side of Figure 4.7. The dashed line through the curve indicates the "average" velocity. Instantaneous velocity can be defined as:

$$u(t) = \bar{u} + u'(t) \tag{4.34}$$

where $u'(t)$ represents the fluctuating part, whereas $\bar{u}$ is the average part.

Two questions now arise:

1. *Ho can we numerically compute any fluid-dynamics turbulent flow field?*

2. *In the case that turbulence cannot be directly computed due to computational limitations, how can we take into account its effects?*

In order to answer to these fundamental questions, section 4.3.1 will be devoted to a widely used technique, that is the RANS technique, able to take into account the effects of the fluctuating part in Eq. 4.34, without directly solve it.

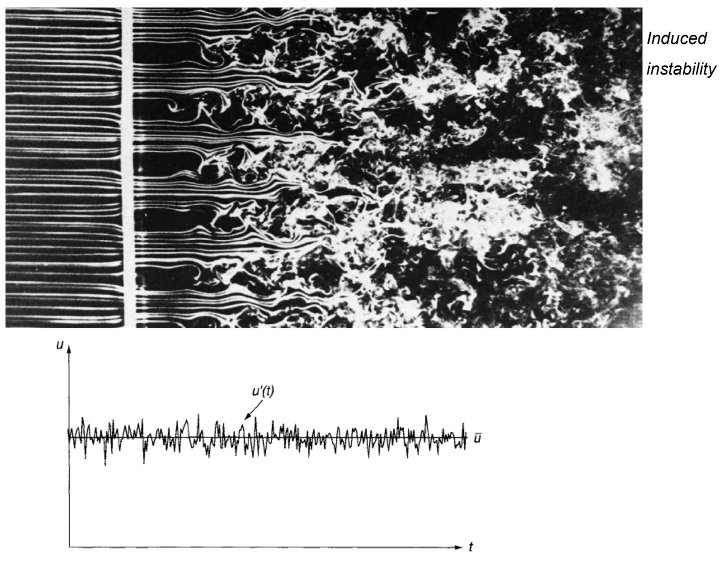


Figure 4.7: Laminar to turbulent induced transition.

4.3.1 The RANS Technique

In studying turbulent flows, the objective is to obtain a theory or a model that can yield quantities of interest, such as velocities. For turbulent flow, the range of length scales and complexity of phenomena make most approaches impossible. The primary approach in this case is to create numerical models to calculate the properties of interest. A selection of some commonly-used computational models for turbulent flows are presented in this section.

The chief difficulty in modelling turbulent flows comes from the wide range of length and time scales associated with turbulent flow. As a result, turbulence models can be classified based on the range of these length and time scales that are modeled and the range of length and time scales that are resolved. The more turbulent scales that are resolved, the finer the resolution of the simulation, and therefore the higher the computational cost. If a majority or all of the turbulent scales are modeled, the computational cost is very low, but the tradeoff comes in the form of decreased accuracy.

In addition to the wide range of length and time scales and the associated computational cost, the governing equations of fluid dynamics contain a non-linear convection term and a non-linear and non-local pressure gradient term. These nonlinear equations must be solved numerically with the appropriate boundary and initial conditions. The three main approaches commonly used to manage turbulence in CFD calculations are:

- **Direct Numerical Simulation (DNS)**: *Direct numerical simulation* resolves the entire range of turbulent length scales. This means that the whole range of spatial and temporal scales of the turbulence must be resolved. All the spatial scales of the turbulence must be resolved in the computational mesh, from the smallest dissipative scales (*Kolmogorov scales*), up to the integral scale L, associated with the motions containing most of the kinetic energy. This marginalizes the effect of models, but is extremely expensive. The computational cost is proportional to Re^3 [68]. Hence, only *low − Reynolds* flows can be studied with this technique, that are not so common in practical engineering problems. DNS is intractable for flows with complex geometries or flow configurations. Figure 4.8 shows two fluids reacting in a turbulent flow. The image on the left, shows where the reaction is occurring, while the one on the right shows how the two fluids have mixed.

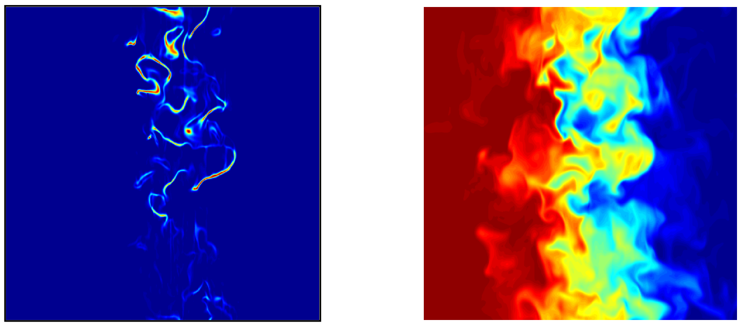


Figure 4.8: Direct Numerical Simulation of two reacting flows (image available at website `http://www.ecs.umass.edu/mie/faculty/debk/Research/dns.html`).

- **Large eddy simulation (LES)**: *Large eddy simulation* is a technique in which the smallest scales of the flow are removed through a filtering operation, and their effect modelled using subgrid scale models. This allows the largest and most important scales of the turbulence to be resolved, while greatly reducing the computational cost incurred by the smallest scales. This method requires greater computational resources than RANS methods, but is far cheaper than DNS. Figure 4.9 shows an example of LES application.

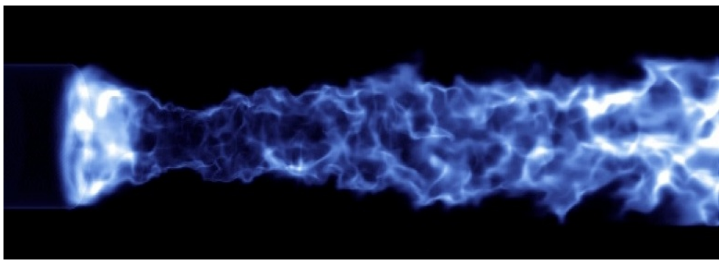


Figure 4.9: Volume rendered image of a Large Eddy Simulation of a non-premixed swirl flame (taken from [86]).

- **Reynolds-averaged Navier-Stokes (RANS)**: *Reynolds-averaged Navier-Stokes* equations are the oldest approach to turbulence modeling. An ensemble version of the governing equations is solved, which introduces new apparent stresses known as Reynolds stresses. This adds a second order tensor of unknowns for which various models can provide different levels of closure. To put it simply, neither the smallest scale nor the largest ones of the turbulence are resolved, but its macroscopic effects are taken into account anyhow. It is a common misconception that the RANS equations do not apply to flows with a time-varying mean flow because these equations are "time-averaged". In fact, statistically unsteady (or non-stationary) flows can equally be treated. This is sometimes referred to as URANS. There is nothing inherent in Reynolds averaging to preclude this, but the turbulence models used to close the equations are valid only as long as the time over which these changes in the mean occur is large compared to the time scales of the turbulent motion containing most of the energy. In Figure 4.10 an application of the unsteady-RANS technique is presented, regarding turbomachinery flow.

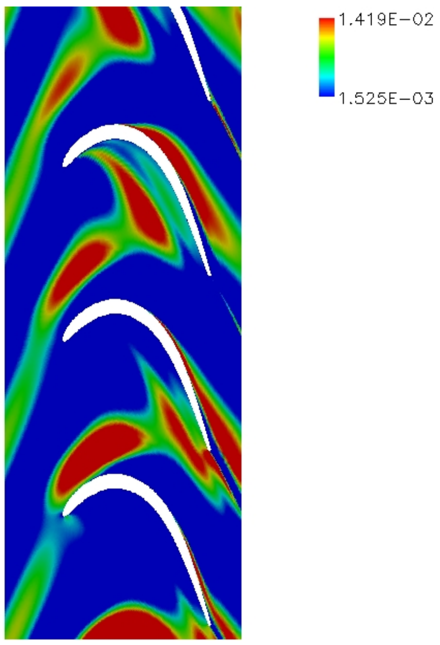


Figure 4.10: Viscous wake impinging a stator vane row of an aero-engine LP turbine. (Axial velocity perturbation magnitude coloured)(taken from [20])

Since the RANS approach implemented in the CFD codes Fluent will be exploited in the following introduced optimization loop, it is now deeply discussed.

RANS models can be divided into two broad approaches:

* **Boussinesq hypothesis**[14]: This method involves using an algebraic equation for the Reynolds stresses which include determining the turbulent viscosity, and depending on the level of sophistication of the model, solving transport equations for determining the turbulent kinetic energy κ and dissipation (ϵ,

[14]Joseph Valentin Boussinesq (13 March 1842 - 19 February 1929) was a French mathematician and physicist who made significant contributions to the theory of hydrodynamics, vibration, light, and heat.

or *turbulent dissipation rate* in the $\kappa - \epsilon$ model [56], ω, or *specific dissipation rate* in the $\kappa - \omega$ model [61]). The models available in this approach are often referred to by the number of transport equations associated with the method. For example, the Mixing Length model is a "Zero Equation" model because no transport equations are solved; the $\kappa - \epsilon$ is a "Two Equation" model because two transport equations (one for κ and one for ϵ) are solved.

* **Reynolds stress model (RSM)**: This approach attempts to actually solve transport equations for the Reynolds stresses. This means introduction of several transport equations for all the Reynolds stresses and hence this approach is much more costly in CPU effort.

Let consider now the 2-D version of continuity equation expressed in Eq. 4.5 for incompressible flows. It assumes the following expression:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0. \tag{4.35}$$

At this point, let substitute the expression of u given in Eq. 4.34 in each velocity term of the previous equation, that is, let substitute each instantaneous quantity into its averaged and fluctuating components fields.

One obtains

$$\frac{\partial\left(\bar{u} + u'(t)\right)}{\partial x} + \frac{\partial\left(\bar{v} + v'(t)\right)}{\partial y} = 0. \tag{4.36}$$

Time-averaging this equation yields,

$$\overline{\frac{\partial\left(\bar{u} + u'(t)\right)}{\partial x} + \frac{\partial\left(\bar{v} + v'(t)\right)}{\partial y}} = \overline{\frac{\partial\left(\bar{u} + u'(t)\right)}{\partial x}} + \overline{\frac{\partial\left(\bar{v} + v'(t)\right)}{\partial y}} = 0 \tag{4.37}$$

It is worth noting that the temporal mean of the velocity fluctuating part is

$$\overline{u'(t)} = \lim_{T\to\infty} \int_t^{t+T} \left[\overline{u(t)} - \overline{u}\right] dt = 0 \tag{4.38}$$

Moreover, the following properties inherent to time-averaging are to be taken into account:

$$\begin{aligned} \overline{\frac{\partial u}{\partial x}} &= \overline{\frac{\partial \overline{u}}{\partial x} + \frac{\partial u'(t)}{\partial x}} = \frac{\partial \overline{u}}{\partial x} \\ \overline{uv} &= \overline{\left(\bar{u} + u'(t)\right) + \left(\bar{v} + v'(t)\right)} = \bar{u}\bar{v} + \overline{u'(t)v'(t)} \end{aligned} \tag{4.39}$$

Eq. 4.37 becomes finally,

$$\tfrac{\partial \overline{u}}{\partial x} + \tfrac{\partial \overline{v}}{\partial y} = 0 \tag{4.40}$$

Similarly, momentum equation and energy equation become respectively, after time-averaging

$$\begin{aligned}
&\frac{\partial\overline{u}}{\partial t} + \frac{\partial\overline{u}\overline{u}}{\partial x} + \frac{\partial\overline{u}\overline{v}}{\partial y} = -\frac{1}{\rho}\frac{\partial\bar{p}}{\partial x} + \frac{\partial}{\partial x}\left(\nu\frac{\partial\bar{u}}{\partial x}\right) + \frac{\partial}{\partial y}\left(\nu\frac{\partial\bar{v}}{\partial y}\right) \\
&+\frac{\partial}{\partial x}\left(\nu\frac{\partial\bar{u}}{\partial x}\right) + \frac{\partial}{\partial y}\left(\nu\frac{\partial\bar{u}}{\partial y}\right) - \left[\frac{\partial\overline{[u'(t)u'(t)]}}{\partial x} + \frac{\partial\overline{[u'(t)v'(t)]}}{\partial y}\right] \\
&\frac{\partial\overline{v}}{\partial t} + \frac{\partial\overline{u}\overline{v}}{\partial x} + \frac{\partial\overline{v}\overline{v}}{\partial y} = -\frac{1}{\rho}\frac{\partial\bar{p}}{\partial y} + \frac{\partial}{\partial x}\left(\nu\frac{\partial\bar{v}}{\partial x}\right) + \frac{\partial}{\partial y}\left(\nu\frac{\partial\bar{v}}{\partial y}\right) \\
&+\frac{\partial}{\partial x}\left(\nu\frac{\partial\bar{u}}{\partial x}\right) + \frac{\partial}{\partial y}\left(\nu\frac{\partial\bar{u}}{\partial y}\right) - \left[\frac{\partial\overline{[u'(t)v'(t)]}}{\partial x} + \frac{\partial\overline{[v'(t)v'(t)]}}{\partial y}\right]
\end{aligned} \tag{4.41}$$

and

$$\begin{aligned}
\frac{\partial\overline{v}}{\partial t} + \frac{\partial\overline{u}\overline{v}}{\partial x} + \frac{\partial\overline{v}\overline{v}}{\partial y} = -\frac{1}{\rho}\frac{\partial\bar{p}}{\partial y} + \frac{\partial}{\partial x}\left(\nu\frac{\partial\bar{v}}{\partial x}\right) + \frac{\partial}{\partial y}\left(\nu\frac{\partial\bar{v}}{\partial y}\right) + \frac{\partial}{\partial x}\left(\nu\frac{\partial\bar{u}}{\partial x}\right) \\
+\frac{\partial}{\partial y}\left(\nu\frac{\partial\bar{u}}{\partial y}\right) - \left[\frac{\partial\overline{[u'(t)v'(t)]}}{\partial x} + \frac{\partial\overline{[v'(t)v'(t)]}}{\partial y}\right]
\end{aligned} \tag{4.42}$$

At this point, closing the previous *RANS equations* requires modeling the Reynold's stress R_{ij}[15]

Joseph Boussinesq was the first practitioner of this, introducing the concept of eddy viscosity. The Boussinesq assumption states that the Reynolds stress tensor, R_{ij}, is proportional to the mean strain rate tensor, S^*_{ij}, and can be written in the following way:

$$R_{ij} = 2\,\mu_t\,S^*_{ij} - \frac{2}{3}\rho k \delta_{ij} \tag{4.43}$$

where μ_t is a scalar property called the *eddy viscosity*. The same equation can be written more explicitly as:

$$\begin{aligned}
-\rho\overline{u'u'} &= 2\mu_t\frac{\partial\overline{u}}{\partial x} - \frac{2}{3}\frac{\partial\rho}{k} \\
-\rho\overline{v'v'} &= 2\mu_t\frac{\partial\overline{v}}{\partial y} - \frac{2}{3}\frac{\partial\rho}{k} \\
-\rho\overline{u'v'} &= \mu_t\left(\frac{\partial\overline{v}}{\partial x} + \frac{\partial\overline{v}}{\partial y}\right)
\end{aligned} \tag{4.44}$$

The right-hand side is analogous to *Newton's law of viscosity*, except for the appearance of the turbulent or eddy viscosity μ_t and turbulent kinetic energy κ.

In Eq. 4.44, the turbulent momentum transport is assumed to be proportional to the mean gradients of velocity. Similarly, the turbulent transport of temperature is taken to be proportional to the gradient of the mean value of the transported quantity. In other words,

[15]The non-linear term $\overline{v'_i v'_j}$ from the convective acceleration is known as the *Reynolds stress*,

$$R_{ij} = \overline{v'_i v'_j}$$

$$
\begin{aligned}
-\rho\overline{u'T'} &= \Gamma_t \frac{\partial \overline{T}}{\partial x} \\
-\rho\overline{v'T'} &= \Gamma_t \frac{\partial \overline{T}}{\partial y}
\end{aligned}
\tag{4.45}
$$

where $\kappa = \frac{1}{2}u'u'$ is the *turbulent kinetic energy* whereas $\Gamma_t = \frac{\mu_T}{Pr_T}$ is referred to as the *turbulent thermal diffusivity, Pr_T* being the turbulent Prandtl number.

By substituting the Reynolds stress expressions in Eq. 4.44 and the extra temperature transport terms in Eq. 4.45 into the governing Eqs. 4.40, 4.41, and 4.42, and removing the overbar, that is by default indicating the average quantities, we obtain

$$
\tfrac{\partial u}{\partial x} + \tfrac{\partial v}{\partial y} = 0 \tag{4.46}
$$

$$
\begin{aligned}
\tfrac{\partial u}{\partial t} + \tfrac{\partial uu}{\partial x} + \tfrac{\partial uv}{\partial y} &= -\tfrac{1}{\rho}\tfrac{\partial p}{\partial x} + \tfrac{\partial}{\partial x}\left[(\nu+\nu_T)\tfrac{\partial u}{\partial x}\right] + \tfrac{\partial}{\partial y}\left[(\nu+\nu_T)\tfrac{\partial v}{\partial y}\right] \\
&\quad + \tfrac{\partial}{\partial x}\left[(\nu+\nu_T)\tfrac{\partial u}{\partial x}\right] + \tfrac{\partial}{\partial y}\left[(\nu+\nu_T)\tfrac{\partial u}{\partial y}\right] \\
\tfrac{\partial v}{\partial t} + \tfrac{\partial uv}{\partial x} + \tfrac{\partial vv}{\partial y} &= -\tfrac{1}{\rho}\tfrac{\partial p}{\partial y} + \tfrac{\partial}{\partial x}\left[(\nu+\nu_T)\tfrac{\partial v}{\partial x}\right] + \tfrac{\partial}{\partial y}\left[(\nu+\nu_T)\tfrac{\partial v}{\partial y}\right] \\
&\quad + \tfrac{\partial}{\partial x}\left[(\nu+\nu_T)\tfrac{\partial u}{\partial x}\right] + \tfrac{\partial}{\partial y}\left[(\nu+\nu_T)\tfrac{\partial u}{\partial y}\right]
\end{aligned}
\tag{4.47}
$$

$$
\begin{aligned}
\frac{\partial v}{\partial t} + \frac{\partial uv}{\partial x} + \frac{\partial vv}{\partial y} &= -\frac{1}{\rho}\frac{\partial p}{\partial y} + \frac{\partial}{\partial x}\left[\left(\frac{\nu}{Pr} + \frac{\nu_T}{Pr_T}\right)\frac{\partial v}{\partial x}\right] + \frac{\partial}{\partial y}\left[\left(\frac{\nu}{Pr} + \frac{\nu_T}{Pr_T}\right)\frac{\partial v}{\partial y}\right] \\
&\quad + \frac{\partial}{\partial x}\left[\left(\frac{\nu}{Pr} + \frac{\nu_T}{Pr_T}\right)\frac{\partial u}{\partial x}\right] + \frac{\partial}{\partial y}\left[\left(\frac{\nu}{Pr} + \frac{\nu_T}{Pr_T}\right)\frac{\partial u}{\partial y}\right]
\end{aligned}
\tag{4.48}
$$

Eq. 4.46, 4.47 and 4.48 constitute respectively the *Time averaged Continuity equation*, the *Time averaged Momentum equation* and the *Time averaged Energy equation*.

4.3.2 The $\kappa - \omega$ turbulence model

In the previous time-averaged equations a new variable appears, that is the *kinematic eddy viscosity ν_T*. In order to make it possible to solve this system of equations, one or more additional equation are to be written, which constitute the "Turbulence models".

In this section, the $\kappa - \omega$ model will be addressed, since it is one of the most common turbulence models and it will be exploited during the CFD simulations reported in the chapter 7. It is a two equation model, that means, it includes two extra transport equations to represent the turbulent properties of the flow. This allows a two equation model to account for history effects like convection and diffusion of turbulent energy.

The first transported variable is turbulent kinetic energy, κ. The second transported variable in this case is the specific dissipation, ω. It is the variable that determines the scale of the turbulence, whereas the first variable, κ, determines the energy in the turbulence.

In particular, the SST (acronym for Shear Stress Transport) $\kappa-\omega$ turbulence model [61] is a two-equation eddy-viscosity model which has become very popular. The shear stress transport (SST) formulation combines the best of two worlds. The use of a $\kappa-\omega$ formulation in the inner parts of the boundary layer makes the model directly usable all the way down to the wall through the viscous sub-layer, hence the SST $\kappa-\omega$ model can be used as a Low-Re turbulence model without any extra damping functions. The SST formulation also switches to a $\kappa-\epsilon$ behaviour in the free-stream and thereby avoids the common $\kappa-\omega$ problem that the model is too sensitive to the inlet free-stream turbulence properties. Authors who use the SST $\kappa-\omega$ model often merit it for its good behavior in adverse pressure gradients and separating flow. The SST $\kappa-\omega$ model does produce a bit too large turbulence levels in regions with large normal strain, like stagnation regions and regions with strong acceleration. This tendency is much less pronounced than with a normal $\kappa-\epsilon$ model though.

The kinematic eddy viscosity is calculated as

$$\nu_T = \frac{a_1 k}{\max(a_1\omega, SF_2)} \tag{4.49}$$

Turbulence Kinetic Energy is modeled via the following equation:

$$\frac{\partial k}{\partial t} + U_j \frac{\partial k}{\partial x_j} = P_k - \beta^* k\omega + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_k \nu_T)\frac{\partial k}{\partial x_j}\right] \tag{4.50}$$

Specific Dissipation Rate transport equation is

$$\begin{aligned}\frac{\partial \omega}{\partial t} + U_j \frac{\partial \omega}{\partial x_j} = \alpha S^2 - \beta\omega^2 + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_\omega \nu_T)\frac{\partial \omega}{\partial x_j}\right] \\ + 2(1-F_1)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}\end{aligned} \tag{4.51}$$

Closure Coefficients and Auxiliary Relations are

$$
\begin{aligned}
F_2 &= \tanh\left[\left[\max\left(\frac{2\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right)\right]^2\right] \\
P_k &= \min\left(\tau_{ij}\frac{\partial U_i}{\partial x_j}, 10\beta^* k\omega\right) \\
F_1 &= \tanh\left\{\left\{\min\left[\max\left(\frac{\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right), \frac{4\sigma_{\omega 2}k}{CD_{k\omega}y^2}\right]\right\}^4\right\} \\
CD_{k\omega} &= \max\left(2\rho\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}, 10^{-10}\right) \\
\phi &= \phi_1 F_1 + \phi_2(1 - F_1) \\
\alpha_1 &= \frac{5}{9}, \alpha_2 = 0.44 \\
\beta_1 &= \frac{3}{40}, \beta_2 = 0.0828 \\
\beta^* &= \frac{9}{100} \\
\sigma_{k1} &= 0.85, \sigma_{k2} = 1 \\
\sigma_{\omega 1} &= 0.5, \sigma_{\omega 2} = 0.856
\end{aligned}
\tag{4.52}
$$

4.4 Generic form of the Governing Equations for CFD

From the governing equations derived above, there are significant commonalities between these various equations. If we introduce a general variable Φ expressing all the fluid flow equations, including equations of temperature and turbulent quantities, in the conservative incompressible form, the equation can usually be written as

$$
\underbrace{\frac{\partial \rho\Phi}{\partial t}}_{Transient\ term} + \underbrace{\nabla \cdot (\rho\vec{u}\Phi)}_{Convection\ term} = \underbrace{\nabla \cdot (\Gamma\nabla\Phi)}_{Diffusion\ term} + \underbrace{S_\Phi}_{Source\ term} \tag{4.53}
$$

where Γ is the *diffusion coefficient* or *diffusivity*.

- *The transient term*, $\frac{\partial \rho\Phi}{\partial t}$, accounts for the accumulation of Φ in the concerned control volume;
- *The convection term*, $\nabla \cdot (\rho\vec{u}\Phi)$, accounts for the transport of Φ due to the existence of the velocity field (note the velocity $\vec{u}$ multiplying Φ);
- *The diffusion term*, $\nabla \cdot (\Gamma\nabla\Phi)$, accounts for the transport of Φ due to its gradients;
- *The source term*, S_Φ , accounts for any sources or sinks that either create or destroy Φ. Any extra terms that cannot be cast into the convection or diffusion terms are considered as source terms.

After developing all the terms in eq. 4.53, we can obtain

$$\begin{aligned}\frac{\partial \Phi}{\partial t}+\frac{\partial u\Phi}{\partial x}+\frac{\partial v\Phi}{\partial y}+\frac{\partial w\Phi}{\partial z}=\frac{\partial}{\partial x}\left[\Gamma\frac{\partial \Phi}{\partial x}\right]\\ +\frac{\partial}{\partial y}\left[\Gamma\frac{\partial \Phi}{\partial y}\right]+\frac{\partial}{\partial z}\left[\Gamma\frac{\partial \phi}{\partial z}\right]+S_{\Phi}\end{aligned} \tag{4.54}$$

Equation 4.54 is the so-called *Transport equation for the property Φ.*

The goal of all discretization techniques (Finite Difference, Finite Element, Finite Volume, Boundary Element...) is to derive a mathematical formulation to transform each of these terms into an algebraic equation. Once applied to all control volumes in a given mesh, and after setting the transport property ϕ equal to ρ u, v, w, T, k, E, and selecting appropriate values for the diffusion coefficient Γ and source terms S_{ϕ} we obtain a full linear system of equations that needs to be solved.

4.5 The Finite Volume method: an overview

In order to discretize in the physical spaces the integral form of the conservation equations, whose general form is expressed in Eq. 4.54, the Finite-Volume method method is required.

The *Finite Volume Method* (FVM) is one of the most versatile discretization techniques used in CFD.

The computational domain is first subdivided into a finite number of contiguous control volumes, where the resulting statements express the exact conservation of relevant properties for each of the control volumes. At the centroid of each of the control volumes, the variable values are then calculated. To do that, integration of the differential form of the governing equations over each control volume is required. Interpolation is used to express variable values at the control volume surface in terms of the center values and suitable quadrature formulae are applied to approximate the surface and volume integrals. An algebraic equation for each of the control volumes can be obtained, in which a number of the neighboring nodal values appear.

As with other discretization methods, a numerical grid must be initially defined to discretize the physical flow domain of interest. For the finite-volume method, there is the possibility of representing the grid by either structured or unstructured mesh.

For illustration purposes of the finite-volume method, we consider a typical representation of structured (quadrilateral) and unstructured (triangle) finite-volume elements in two-dimensional shown in Figure 4.11 for the discretization of the partial differential equations.

In particular, the so called Viscous hybrid meshes have proved to be suitable for external aerodynamic purposes. This kind of mesh features several layers of prismatic elements along walls, with tetrahedral elements in the core flow region.

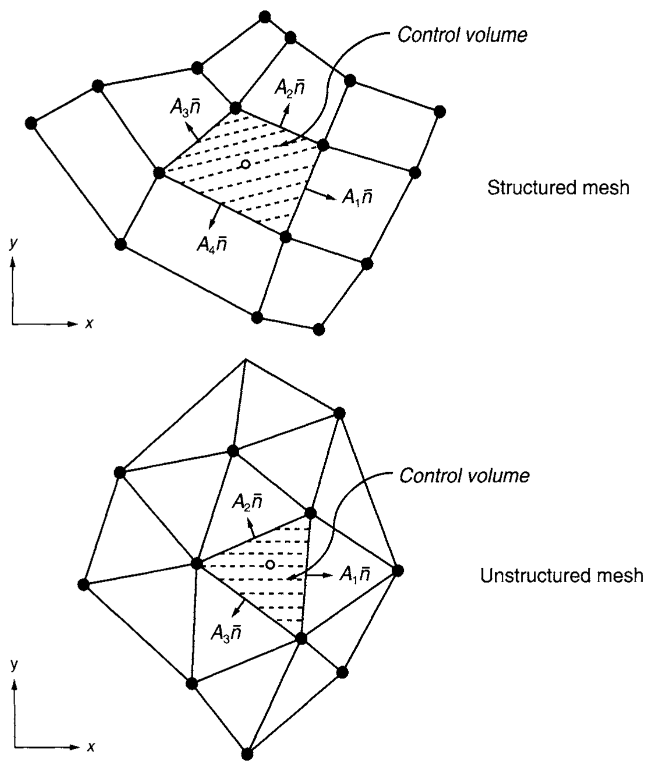


Figure 4.11: A representation of structured and unstructured mesh for the finite-volume method (full symbols denote element vertices and open symbols at the center of the control volumes denote computational nodes).

Compared to all-tetrahedral meshes, viscous hybrid meshes result in dramatic savings, with far fewer elements required to accurately resolve boundary layers and give good near-wall prediction of shear stress, heat transfer, and flow separation.

Let us consider the two unstructured grid depicted in Figure 4.12. In the case of the fully tetrahedral unstructured grid (Figure 4.12, on the left), the demand of small $y+$ (please read notes included in [7] and [8] for a deeper insight) would translate into an unacceptable grid distortion near the wall. On the contrary, the hybrid mesh depicted in Figure 4.12, on the right, allows to capture the near-wall fluid dynamic phenomena by means of the thin prismatic layers, without the need to increase the grid density in the core region. As a result, boundary layer is captured in an effective way, while containing at a minimum the number of elements.

Let us consider the two-dimensional continuity equation for an incompressible flow:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0. \tag{4.55}$$

Integrating it on a generic control volume yields the following expression,

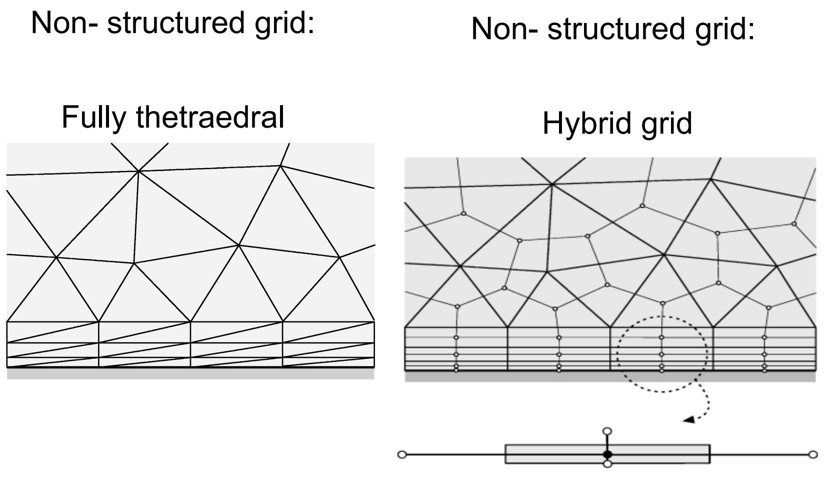


Figure 4.12: A comparison between a fully tetrahedral unstructured grid (ON THE LEFT) and a viscous hybrid mesh (ON THE RIGHT).

which is applicable to both structured and unstructured grids:

$$\int_V \frac{\partial u}{\partial x} dV + \int_V \frac{\partial v}{\partial y} dV = 0. \tag{4.56}$$

where V stands for Volume.

Now, the above equation can be discretized by applying Gauss' divergence theorem to the volume integral (where A stands for Area, and N means the total number of cells),

$$\int_V \frac{\partial u}{\partial x} dV = \int_A u dA^x \approx \sum_{k=1}^{N} u_i A_i^y \tag{4.57}$$

$$\int_V \frac{\partial v}{\partial y} dV = \int_A v dA^y \approx \sum_{k=1}^{N} v_i A_i^y \tag{4.58}$$

Finally, the following equation is obtained:

$$\sum_{k=1}^{N} u_i A_i^y + \sum_{k=1}^{N} v_i A_i^y = 0 \tag{4.59}$$

Eq. 4.59 represents the *algebrized* or *discretized* form of the continuity equation. Now let us consider an elemental control volume of the two-dimensional structured grid shown in Figure 4.13.

The centroid of the control volume is indicated by the point P, which is surrounded by the adjacent control volumes having their respective centroids indicated

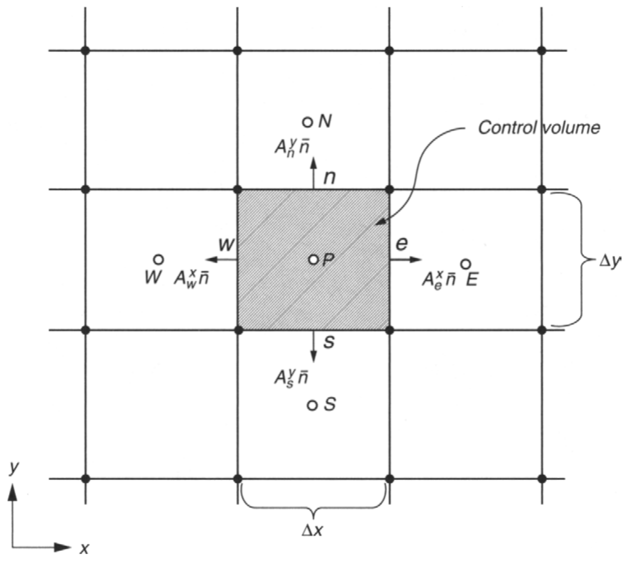


Figure 4.13: Control volume for the two-dimensional continuity equation problem. Image taken from [92]

by the points: east, E; west, W; north, N; and south, S. The control volume face between points P and E is denoted by the area A_e^x. Subsequently, the rest of the control volume faces are A_w^x, A_n^y, and A_s^y, respectively.

The analysis starts by introducing the control volume integration, which forms the key step of the finite-volume method. Applying Gauss' divergence theorem yields the following expressions,

$$\sum_{k=1}^{4} u_i A_i^x = \frac{1}{\Delta V}\left(u_e A_e^x - u_w A_w^x + \overbrace{u_n A_n^x}^{=0} - \overbrace{u_s A_s^x}^{=0} \right) \tag{4.60}$$

$$\sum_{k=1}^{4} v_i A_i^y = \frac{1}{\Delta V}\left(\overbrace{v_e A_e^y}^{=0} - \overbrace{v_w A_w^y}^{=0} + v_n A_n^y - v_s A_s^y \right) \tag{4.61}$$

For the structured uniform grid arrangement, the projection areas A_n^x and A_s^x in the x direction, and the projection areas, A_e^y and A_w^y in the y direction, are zero. Since the grid has been considered to be uniform, the face velocities u_e, u_w, v_n, and v_s are located midway between each of the control volume centroids, which allows us to determine the face velocities from the values located at the centroids of the control volumes.

By assuming a linear velocity profile,

$$u_e = \tfrac{u_P+u_E}{2} \qquad u_w = \tfrac{u_P+u_W}{2} \qquad u_n = \tfrac{u_P+u_N}{2} \qquad u_s = \tfrac{u_P+u_S}{2}$$

By substituting the above expressions to the discretized form of the velocity first-order derivatives, the final form of the discretized continuity equation becomes

$$\left(\frac{u_P + u_E}{2}\right) A_e^x - \left(\frac{u_P + u_W}{2}\right) A_w^x + \left(\frac{u_P + u_N}{2}\right) A_n^y - \left(\frac{u_P + u_S}{2}\right) A_s^y \tag{4.62}$$

From Figure 4.13, $A_e^x = A_w^x = \Delta y$ and $A_n^y = A_s^y = \Delta x$, the above equation can then be expressed by

$$\left(\frac{u_P + u_E}{2}\right) \Delta y - \left(\frac{u_P + u_W}{2}\right) \Delta y + \left(\frac{u_P + u_N}{2}\right) \Delta x - \left(\frac{u_P + u_S}{2}\right) \Delta x \tag{4.63}$$

and reduced to

$$\left(\frac{u_E + u_W}{2\Delta x}\right) \Delta y + \left(\frac{v_N + v_S}{2\Delta y}\right) \Delta x = 0 \tag{4.64}$$

Eq. 4.64 constitutes the *algebraic form* of the continuity equation.

It is important to note that the above equation referred to the central node of the control volume depicted in Figure 4.13, i.e. P, contains the unknowns concerning the adjacent points, that is N, S, W, and E. Therefore, for a n cells grid, the finite volume method yields a system of n similar equations, to be solved after imposing the so called *boundary conditions* at the domain boundaries.

Chapter 5

A Survey of the Techniques for Drag Reduction

As just clarified in the abstract, the aim of this Thesis is to build up an automatic optimization procedure, applicable to aircraft and rotorcraft aerodynamic components, and to prove the effectiveness of that procedure by performing an optimization run on a real industrial product. To achieve the latter purpose, AgustaWestland has chosen the ERICA tilt-rotor intake configuration as a test case. Therefore, it seems natural, at this point, to provide some notions about the tiltrotor concept.

Moreover, for completeness of informations, a detailed survey of the most promising techniques suitable for External Aerodynamic drag reduction is provided. For conceptual clarity, the different devices will be classified based on the kind of drag to be lowered. As far as their application is concerned, both advantages and disadvantages will be discussed with critical sense. For a deeper overview, reader is referred to the precious notes given in the Von Karman Instiute *Lecture Series* "Flow Control: Fundamentals, Advances and Applications" [33].

5.1 The Tilt-Rotor Concept

The peculiar characteristic of the tilt-rotor is the capability to take-off and land like an helicopter and at the same time to cruise like an airplane by tilting the rotor nacelles perpendicular to the flight direction, in helicopter mode, or parallel to the flight direction, in aircraft mode. Moreover, the new tilt-rotor concept, of which ERICA (Enhanced Rotorcraft Innovative Concept Achievement) is a technological demonstrator (Fig .5.1), has the additional capability to tilt the outboard portion of the wings independently from the nacelles, depicted in Figure 5.3 [49, 37].

In Figure 5.2 [66], the ERICA configuration as a function of the nacelle tilt angle is depicted. This feature leads in some benefits about the thrust loss due to rotors down-wash on the wings in helicopter mode, giving the opportunity to reduce the rotor diameter, so as to improve the cruise performance. In addition, its independent tilt freedom allows the wings to avoid stall, then improving the operational capability of the aircraft during the helicopter-airplane conversion phase. Moreover, the smaller dimensions of the rotors allow a tilt-rotor like ERICA to perform STOL (Short Take-Off and Landing) operations, that means that also take-off and landing in airplane mode is achievable [37].

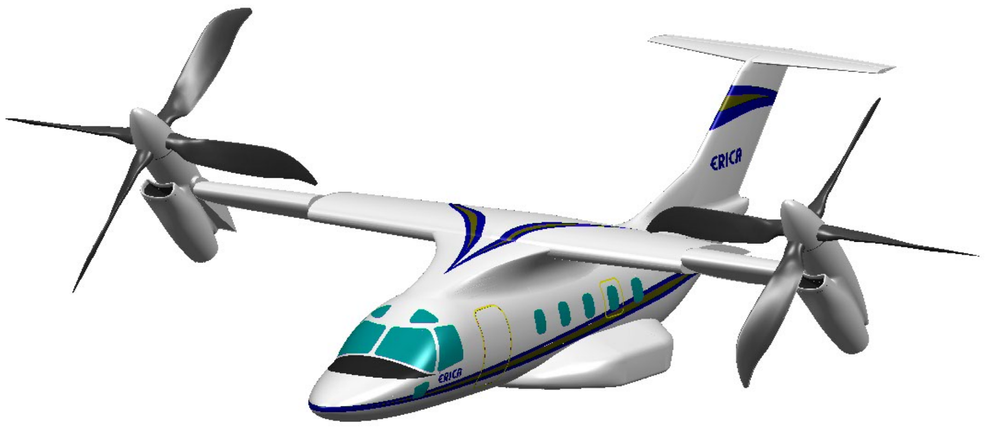


Figure 5.1: An artistic impression of the ERICA concept (taken from [37]).

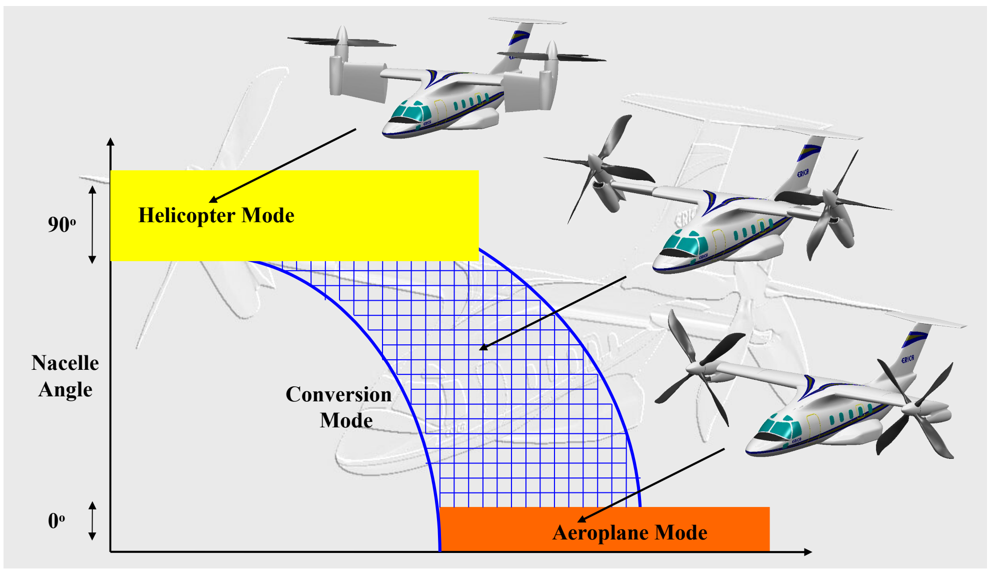


Figure 5.2: Different ERICA possible configurations (taken from [66]).

The most suitable mission profile, for a tilt-rotor with these characteristics, is the point to point service operated from and to heliports, but also to and from short field in order to increase the range and/or payload capabilities. A comparison of the estimated operative costs among the presently operating aircrafts and tilt-rotors shows a competitive role of tilt-rotor with respect to helicopters for all those missions whose required range is grater than 150 nautical miles, while the competition with airplanes must be played on the field of tilt-rotor operational

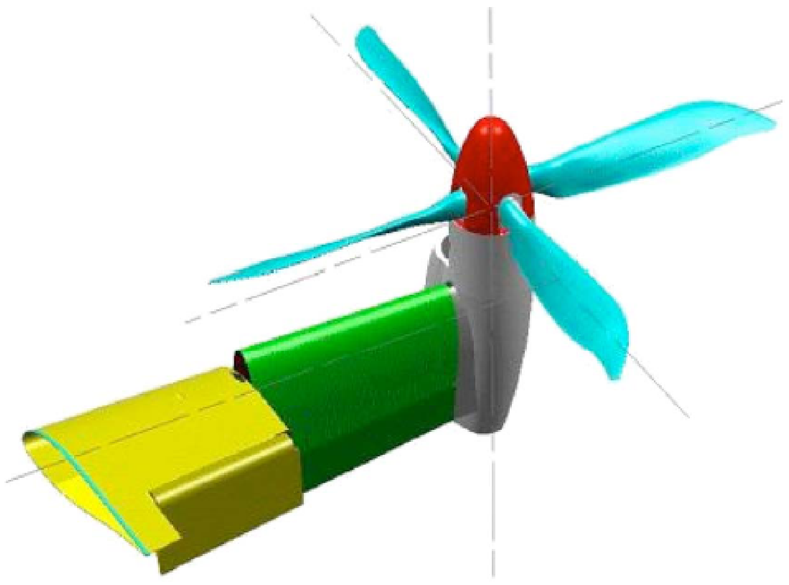


Figure 5.3: ERICA tiltable wing (taken from [37]).

flexibility that allows the minimization of the ground infrastructure needs.

5.2 Power requirements

The purpose of an aircraft, such as the future European civil tilt-rotor based on ERICA architecture, is to transport people from place to place in a short time. To do this, it must take off from land, climb to a desired flying altitude, cruise at a prescribed distance, manoeuvre, descend and finally land at the desired destination. An aircraft is considered to have merit insofar as it performs these things quickly, safely and, by no means the least important, with the minimum expenditure of energy. The quantitative measure of the way it performs such function is known as the *performance of the aircraft*. A simple and accurate method to understand it is the *energy* method [36]. This method is based on the fact that, in steady flight, a power balance exists for an aircraft. As far as cruise flight condition is concerned, the lift force L counterbalances the airplane weight W, whereas thrust T force has to exactly overcome the total drag force D. Therefore, two equations must be fulfilled, in the case of steady flight condition:

$$W = L \quad T = D \tag{5.1}$$

By multiplying the second member of Eq. 5.1 by the factor V (speed), an equation between power quantities is found, that is

$$VD = VT \tag{5.2}$$

where VD is called *required power*, whereas VT is the *available power*. If steady level flight is to be maintained, these powers must be equal [62]. Eq. 5.2 clearly shows that the power required for level flight directly depends on the drag force.

Hence, *a reduction of the drag force will translate into lower power consumption*.

Conventionally, the total power required by the aircraft during steady flight is split into three contributes. These power-absorbing elements are, according to Gessow's terminology:

- Induced-drag power;
- Propeller blades profile-drag power;
- Parasite-drag power.

Induced-drag power is the fraction required to produce lift, and can be reduced by optimizing the shape and geometry of the lifting parts; it is due to both rotorcraft body and rotors.

Profile-drag power is the power required to drag the propeller blades through the air, and can be lowered by means of a proper shaping of the blades.

Parasite-drag power is, generally speaking, the power required to move a solid object through a fluid. As far as tilt-rotor is concerned, this power arises from the necessity of move the airframe and the rotating non-lifting components through the air.

Parasitic drag is made up of many components, such as *pressure drag*, *skin friction drag* and *interference drag*, which will be introduced in the next Section. In addition, assuming that both transmission losses and various auxiliary equipment losses account for a fixed fraction of the required power P_R, the total shaft power P_S required of the engines is

$$P_S = \frac{P_R}{\eta_T} + P_A \tag{5.3}$$

where P_A is the power required to operate accessories, and η_T is the transmission efficiency.

Moreover, installation of the engines on the airframe generally results in a performance deterioration when compared to the engine manufacturer's performance specifications. Losses associated with the engine installation can be divided into *inlet losses*, *exhaust losses*, and losses due to *bleed air extraction* (e.g. air extraction from the compressor for anti-ice protection). Engine installation effects are not to be forgotten, since they play a smaller but not negligible role in the overall power expenditure (as shown in Table 5.1) and they will be evaluated and reported in a separate document. As high speed typical of cruise condition are reached, the role played by parasite drag in the overall power balance becomes more and more fundamental. Hence, a great reduction in power requirement is expected to be achieved by improving the aerodynamic efficiency of the tilt rotor fuselage. In Table 5.1, an example of power required breakdown is provided, showing the different contributions to the overall power balance.

Table 5.1 clearly shows that fuselage parasite drag and body-induced drag provide by far the greatest contributions to the overall power balance. Hence, an airframe drag reduction is expected to be useful to gain an actual power reduction.

	% Required Power
Engine Installation	*8%*
Fuselage	*37%*
Rotors (induced, viscous, vortex)	*15%*
Transmission	*3%*
Accessories	*7%*
Body Induced	*30%*

Table 5.1: Power required breakdown.

5.2.1 Fuselage and body induced drag breakdown

After performing power-required breakdown, a drag breakdown is to be carried out. To this purpose, fuselage and body induced drag are taken into account only, in order to put in evidence the contribution of the fixed and non-lifting components. Table 5.2 provides fuselage and rotating non-lifting components drag breakdown on a Tilt-rotor (ERICA configuration). The values, even if obtained on a scale model, are referred to the full scale size. The different contributions are presented in terms of equivalent flat-plate area, instead of drag coefficient. The term “flat-plate” is the area of a flat plate having a C_D of 1.0 with the same drag of the shape under consideration. In other words, it represents the ratio of the Drag over the dynamic pressure and is a meaningful way to compare bodies flying at the same speed and density. The reason why drag coefficients of the single components will not be considered is that they cannot be added since reference areas are different. Considering flat plate area f of each component is a convenient way of handling the drag, since they can be added to give the total f of an airplane, which, in turn, is proportional to the drag force.

Fuselage and body induced component	**Equivalent flat-plate area, $f[m^2]$**	**Contribution $[\%]$**
Base fuselage	*0.627*	*19.5*
Fuselage-wing fairings	*0.128*	*4.1*
Sponsons[16]	*0.403*	*12.5*
Wings	*0.691*	*21.6*
Nacelles and spinners	*0.211*	*6.6*
Rotor hub and stubs	*0.941*	*29.3*
Fin	*0.122*	*3.8*
Tailplane	*0.090*	*2.7*
TOTAL	**3.213**	**100**

Table 5.2: Fuselage and body-induced drag breakdown. (NICETRIP wind tunnel tests, 2008. [67])

[16]*Sponsons* are projections from the sides of an aircraft or helicopter, for protection, stability, or the mounting of equipment such as armaments or lifeboats, etc. They are often used in larger helicopters where the internal space of the sponson can be used for fuel or to house landing gear without reducing cargo or passenger space in the fuselage

As can be clearly seen from Table 5.2, rotor hub and stubs, wings and base fuselage represent by far the greatest contributions to parasite and body-induced drag. The next Section will provide an overview of the several drag reduction techniques employable mainly on non-rotating components of a next-generation Tilt-Rotor.

5.3 Overview of aircraft drag sources

Although the relative importance of different drag sources varies for each aircraft type and mission that is flown, a representative breakdown is shown in Figure 5.4. The most important contributions to the total drag are the following:

1. *Skin friction drag* due to viscous boundary layer formation and development;
2. *Pressure drag* due to open separations in the afterbody and other regions;
3. *Interference effects* between aerodynamic components;
4. *Lift induced drag* due to the conserved circulation developed around the wings;
5. *Miscellaneous effects* such as roughness effects and leakage, etc;
6. *Wave drag* due to compressibility effects at near-sonic flight conditions;

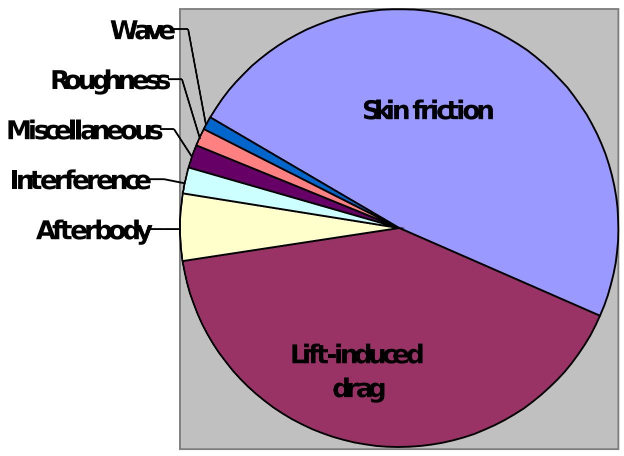


Figure 5.4: Contributions of different drag sources for a typical transport aircraft (taken from [1]).

The greatest contribution arises from turbulent *skin friction drag*, a fact that has justified the impetus for most of the friction drag reduction work that will be described here. The next most significant contribution comes from the lift induced drag and this, together with the friction drag, accounts for about 85% of the total aircraft drag. Interference drag, wave drag, and miscellaneous effects account for the remainder. In particular, wave drag will not be taken into account since it plays a negligible role in the overall drag of a tilt-rotor fuselage.

5.4 Skin friction drag reduction

Skin friction drag is the component resulting from viscous shear stresses, whose integral is resolved in the drag direction. For the reduction of this kind of drag, either (both, in the sense that will be clarified soon) of two different philosophies may be followed:

1. The first consists in stabilizing the laminar boundary layer (BL) as to prevent or delay transition to a turbulent one. In fact, because of the greater velocity gradient characterizing turbulent layer than in a laminar one (due to the more agitated motion in a turbulent flow), the frictional effects are more severe for a turbulent BL;

2. An alternative philosophy for friction reduction that has recently emerged is to accept the inevitability of turbulent flow and to attempt to modify or interact with the turbulent structures so as to reduce the friction.

The two precedent methods are not in conflict with each other, but are complementary since the former are exploitable on the wings, where the boundary layer, at least near the leading edge, keeps laminar, whereas the latter is applied mainly on fuselage where turbulent boundary layer exists. As regards the first class of methods, they can be divided into active and passive, depending on whether additional power from the propulsion units is required or not.

5.4.1 Laminar Flow Control (LFC)

An interesting active method for preventing laminar to turbulent transition is the so called *Laminar Flow Control* (LFC), which means the maintenance of laminar flow through the use of wall suction. Laminar to turbulent transition is a complex phenomenon, which relates to instabilities and contaminations encountered by the flow along the walls. Instabilities and contaminations can be classified as follows ([99, 75]):

- *Streamwise instability;*
- *Cross flow instabilities;*
- *Leading edge contamination.*

Streamwise instability is caused by amplification of the *Tollmien-Schlichting* waves and is responsible for transition when the sweep angle ϕ is less than 25°. The point of transition depends on the flow Reynolds number, pressure distribution and the presence of surface roughness elements.

Cross-flow instability has its origin in high cross flow ($\phi \geq 25°$), which makes high level Reynolds numbers the first cause of transition. This instability develops around the leading edge, where acceleration is high. It is a major concern for high Reynolds number flow in subsonic transport aircraft, such as a *Tilt-rotor*.

Leading edge contamination (also called *attachment line contamination*) is the third cause of transition, and by no means the least. Attachment line contamination is the phenomenon by which turbulent air at the wing root (coming from a turbulent boundary layer on the fuselage) is propagated along the attachment line of a swept leading edge, causing the flow over the whole of the wing (or empennage surface) to become turbulent. In the case of an engine nacelle, which has no leading edge sweep, this kind of transition mechanism is not present.

These three sources of transition, which are frequently found together and prematurely trigger transition near the leading edge, are well documented in the open literature ([85, 47, 99]) and are depicted in Figure 5.5.

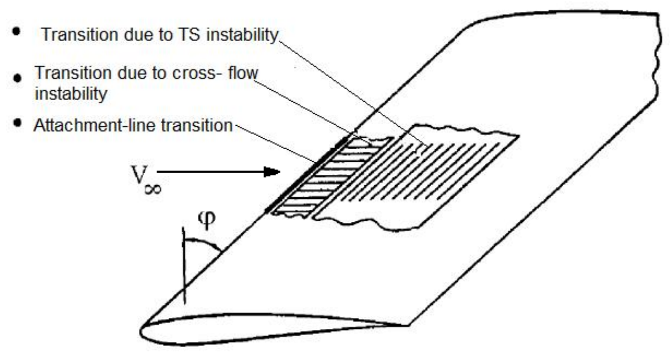

Transition due to TS instability
Transition due to cross- flow instability
Attachment-line transition
V_∞
φ

Figure 5.5: Transition mechanism on swept wings (taken from [85]).

Suction may be implemented in the form of distributed porosity over the surface, in the form of a series of spanwise-running slots or in the form of discrete drilled holes. Porous materials like foams, fabrics, nylons and sintered meshes, tend however to have little structural integrity and shear stiffness, rendering them unsuitable for use on their own as a skin material, unless supported by a load bearing underlayer. Distributed suction acts in two main ways to suppress laminar-turbulent transition [41]: first, it reduces the boundary-layer thickness; second, it creates a much fuller velocity profile within the boundary layer, somewhat similar to the effect of a favourable pressure gradient. This makes the boundary layer much more stable with respect to the growth of small disturbances (e.g. *Tollmien-Schlichting* waves, cross-flow instabilities, etc.). In Figure 5.6, the LFC working principle is depicted. Although suction modifies the velocity profile in the boundary layer, in such a way that in some cases the local skin friction coefficient is reduced by more than 10% [75], several reasons make this technique unattractive, that are itemized in the following:

- It requires a very fine definition of the suction system parameters. These parameters involve hole size, suction flow rate, hole spacing and geometry, and hole inclination [48];

- The results obtained depend very much on the state of the surface and on the absence of contaminations (e.g. insects or dust), which is a not negligible issue when considering its application to *Tilt-rotor wings*;
- It is characterized by high maintenance costs related to the high amount of energy necessary for the suction;
- Acoustic waves and vibrations on the wings and tail assembly could impair the effectiveness of LFC system, since laminar to turbulent transitions would be promoted [48]. This is an issue for Tilt-rotor aircrafts, since tiltable rotors are mounted on the wings;
- Contamination due to fuselage boundary layer is a real concern;
- From an aerodynamic viewpoint, the greatest difficulty lies in being able to confidently predict where transition will occur, although CFD techniques could be helpful with regard to this.

An essential problem with any laminar flow condition is its susceptibility to dirt and other particulates, such as insect debris accumulating near the leading edge during low altitude flight. These can trip the flow to turbulence, which will then spread over a wide area of the wing. To avoid this, close manufacturing tolerances must be followed and some kind of in flight cleaning system or leading-edge protection must be employed. Moreover, de-icer inserts on shields for ice protection and supplementary nozzles for protection from insects and ice are required.

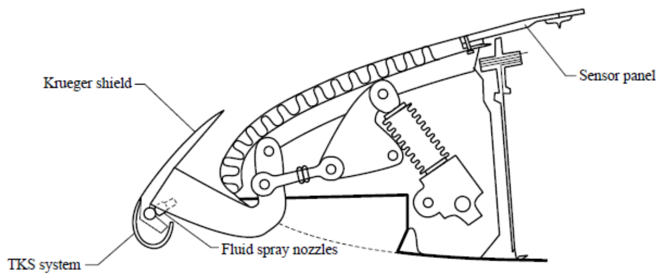


Figure 5.6: LFC system designed by McDonnell- Co. (taken from [48]).

In Figure 5.6, the Douglas concept for LFC application is showed; it involves an electron-beam-perforated titanium sheet bonded to a fiberglass-corrugated substructure. Suction was applied from just below the attachment line back to the front spar. A Krueger shield was used at the leading edge to deflect or block insects. TKS anti-ice system was used on the Krueger shield, and a spray nozzle system was appended to the back of the Krueger shield as a backup system for anti-insect and anti-ice protection of the leading edge. The Krueger shield was typically retracted after reaching an altitude of 6000 ft, with the goal of leaving an insect-free leading edge for cruise flight. This LFC system was employed during the Jetstar LFC Leading-Edge Flight test Program (1983-1986) [41] and showed the effectiveness of LFC systems for airline service.

5.4.2 Natural Laminar Flow (NLF)

Among passive techniques exploitable to BL control, attention has to be paid on *Natural Laminar Flow* (NLF), a technique that uses optimized pressure gradients to maintain laminar flow over a large part of the wings. To obtain these results, maximum profile thickness must be as far aft as possible in order to create extensive regions of favourable pressure gradient over the wing surface. Although this passive technique is less expensive than LFC one, it presents some limits in term of sweep angle and Mach number. In fact, if the Reynolds number exceeds a critical value, *Tollmien-Schlichting* waves within a laminar BL will be amplified as they propagate downstream, resulting in transition. Moreover, attachment line contamination is the phenomenon by which turbulent air at the wing root (coming from a turbulent boundary layer on the fuselage) is propagated along the attachment line of a swept leading edge, causing the flow over the whole of the wing (or tail assembly surface) to become turbulent. These two phenomena occur when Mach number and sweep angle increase excessively. For these reasons, passive laminar flow control is suitable for regional aircraft only [70].

Even though the concept can be employed without the need for considering the attendant weight and structural penalties associated with the LFC suction system, some disadvantages characterize this device, that are:

- Structural weight penalties to be taken into account;
- Permissible wing sweep is limited by the onset of cross-flow instability at the leading edge;
- The challenge is to design wings maintaining natural laminarity with a high performance level (as, for example, the Mach number). The Mach number effect frequently increases the sweep angle which is contrary to the application of this technology;
- Only true savings, including the effect of the system for cleaning the leading edge required to prevent insect contamination, ice formation, and the production costs of such wings will produce tangible benefits where the Direct Operating Costs is concerned;
- The high production costs of such a wing system. For example, to have a laminar upper surface it must not have any roughness greater than 0.1 mm [75];
- Contamination due to fuselage boundary layer is a real concern;

- The presence of the wing tip-mounted propellers impairs the effectiveness of NFL devices, since an early laminar to turbulent transition is promoted [44, 18].

5.4.3 Hybrid Laminar Flow Control (HLFC)

An attractive method for preventing laminar to turbulent transition is called *Hybrid laminar flow control* (HLFC), an ingenious compromise between Natural and Controlled laminarity. This compromise makes it possible to do away with the disadvantages of the two technologies described above. By means of suction in the area between the first 15-20% of the chord and a favourable pressure gradient as for NLF to about 50% of the chord, the effects of cross flow instabilities are minimized and a natural laminar flow may then develop in the mid-part of the wing profile [99]. The key features of HLFC are [48]:

- Suction is required only in the leading-edge region, in order to suppress the cross-flow vortices and leading-edge contamination;
- NLF is maintained over the wing through proper tailoring of the geometry (effect on free-stream flow pressure);
- The HLFC wing design has good performance in the turbulent mode.

In Figure 5.7, a conceptual comparison among three previous techniques is presented.

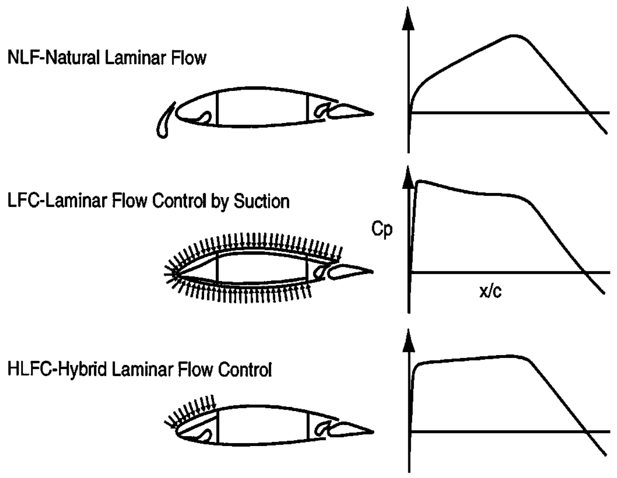

NLF-Natural Laminar Flow
LFC-Laminar Flow Control by Suction
Cp
x/c
HLFC-Hybrid Laminar Flow Control

Figure 5.7: NLF, LFC and HLFC concepts for wing (taken from [48]).

Using HLFC technology, the region of favourable pressure gradient extends until 50% of chord length. The advantage of using this device consists in that it is a HLFC system that avoids some of the problems associated with LFC and NLF. Suction is applied only at the leading edge to minimize cross-flow instability. Control of the instabilities in the mid-chord region is achieved with tailoring of the pressure gradient as with NFL. In this way, a larger wing sweep can be achieved for transonic flight than with NLF, and the weight penalties are not as great as for LFC. As far as its disadvantages is concerned, it has to be underlined that the effectiveness of this system can be impaired by cloudy conditions or rainy weather (because of the lost of laminarity during these conditions). Moreover, contamination due to fuselage boundary layer can be a real concern. Finally, the maintainability and reliability of suction surfaces and the optimization of suction rate and distribution remain the main problems regarding this preventing-transition method.

5.4.4 Riblets

An alternative approach to the reduction of skin friction is based not upon trying to maintain laminar flow, but on attempting to modify the turbulence in some way so as to reduce friction. Possible approaches may be passive, as in the case of the *riblets* and *large eddy break-up* devices (LEBUs) etc., or active as in the case of the synthetic boundary layer. Over the years, extensive research on riblets has been carried out at the NASA Langley Research Centre (USA) and ONERA/CERT (France). Riblets, (micro-grooves on the surface, aligned to the free stream direction, see Figure 5.8), have been studied deeply ([70, 95, 96]) and the results from these studies have been sufficiently promising and encouraging that the concept has been evaluated in flight tests. Despite worldwide research during

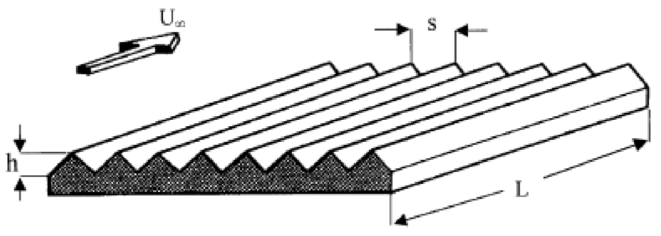


Figure 5.8: Sketch of riblet geometry (taken from [95]).

the last 15 years, detailed mechanisms by which riblets reduce the wall shear stress are not clearly understood. Several mechanisms have been suggested which include: weakening of the bursting process near the wall, significant retardation of the flow in the groove valley dominated by viscous effects, an increase in the sub layer thickness inhibition or restriction of spanwise motion of longitudinal vortices [70]. It is likely that many of the above flow features have their subtle role in altering the wall shear stress. It seems that highly turbulent activity of boundary layer is "lowered" in such a way that skin friction is reduced. Apart from these fluid-dynamics considerations, the aerodynamic advantage procured by riblets has been clearly demonstrated during many wind-tunnel and flight tests. It is worth noting that, despite the increase in the wetted area, a net drag reduction occurs when riblets technique is exploited. This is the reason why an optimization procedure has to be employed, when designing the correct riblets shape.

The main advantages of their application are:

- Their effectiveness is not affected by pressure gradient [70];
- Their effectiveness is not affected by compressibility;
- Their effectiveness is not affected by moderate misalignment to the mean flow [70];
- They are easy to install: plastic riblet films with symmetric V grooves manufactured by the 3M© are suitable to this purpose.

On the other hand , the disadvantages are

- The microscopic structures of riblets are highly sensitive to dirt and mechanical degradation;
- Acceptable installation and removal times must be achieved. Airbus Industry© and 3M© has defined a technique that has cut down 3M's first assessment by a factor 10. Concerning removal, aqua stripping seems the most suitable technique [75];
- Their effectiveness is affected by high misalignment to the mean flow [70]. At 25°-30° their capacity for drag reduction is significantly reduced; moreover, with misalignment, a mean pressure difference across each riblet occurs; such a pressure difference can result in a significant (because of the large area involved) force normal to the line of the riblets, which, in turn, can contribute directly to the drag.

5.4.5 LEBUs

Another passive techniques aiming at modifying turbulent boundary layer structure in such way that there is a net drag reduction, is represented by Large Eddy Break Up (LEBU) devices. They are designed to alter, sever, or break-up the large eddies that form in the outer regions of a turbulent boundary layer. A typical arrangement consists of one or more splitter plates placed in tandem in the outer part of a turbulent boundary layer, as sketched in Figure 5.9. An analytical attempt to explain the mechanisms involved with the LEBUs is noted here. Essentially, the LEBU acts as an airfoil on a gusty atmosphere, and a vortex unwinding mechanism is activated. The vorticity shed from the LEBU's trailing edge because of an incident line vortex convected past the device tends to cancel the effect of the incoming vortex and to reduce the velocity fluctuations near the wall. The main advantage characterizing LEBU is that their effectiveness seems not to be affected by adverse pressure gradient [35]. However, they are not effective at Reynolds and Mach numbers typical of flight conditions [35]. Moreover, what is difficult is to ensure that the device own skin-friction and pressure drag do not exceed the savings.

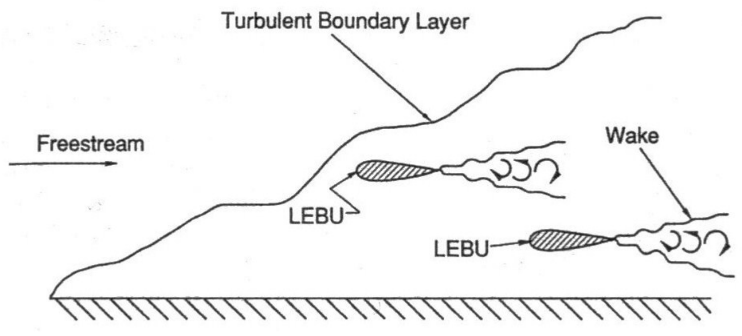


Figure 5.9: Sketch of a tandem arrangement of a LEBU device (taken from [35]).

5.5 Pressure drag reduction

Pressure drag (also named *shape drag*) is the drag of a body resulting from the integrated effect of the static pressure acting normal to its surface resolved in the drag direction. In an aircraft, two typical shape drag effects decrease the flight efficiency, i.e. flow separation in the aft region of the fuselage, and near the trailing edge of the wings, both occurring during off-design operating conditions (e.g. during take-off phases). As far as afterbody drag is concerned, it is a small but significant contribution to the overall aircraft drag, as depicted Figure 5.4. This kind of drag is related to the presence of the upswept regions, essential in order to meet operational requirements and take-off rotation, and anyway present in a tilt-rotor. The basic features of the flow field typical of upswept fuselages are also shown in Figure 5.10 [48], for a typical aircraft. It is characterized by a 3-D boundary layer with significant cross flow regions on the fuselage. This boundary layer separates into a pair of counter rotating-vortices trailing downstream. The flow is analogous to the flow about a missile at high angle of attack or the flow over a delta wing, although in the present case a hard separation line does not exist. The total drag associated with this kind of flow can be split into two components. First, the pressure drag arises because of the reduced pressures on the lower surface of the fuselage. In addition, there is a considerable loss of flow energy in the form of rotational kinetic energy of the vortex structures and this is manifested as a vortex drag component. Depending on the geometry of the aircraft, the relative contributions of each may vary.

In the following, a description of the main flow and boundary layer control devices, suitable for either fixed wing or rotary-wings aircraft will be provided.

5.5.1 Synthetic Jet actuators

Among various flow control methods, *synthetic jet actuation* is one of the most promising as it demonstrates a great potential for active flow control. *Synthetic jets* are attracting increasing interest in the aerospace community because they have many practical applications, including jet vectoring, control of separation, enhanced

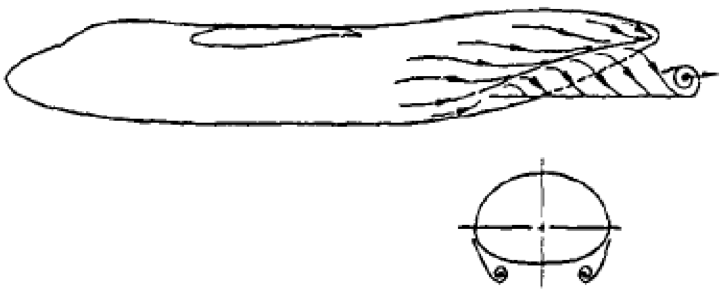


Figure 5.10: The vortex wake behind an upswept afterbody (taken from [89]).

mixing, reduction of wall skin friction, and virtual aero shaping. In particular, as far as separation is concerned, there is a strong motivation to manipulate or delay its occurrence, since it means preventing large energy losses and in most applications lift loss and drag increase.

As a practical tool, they are generally more attractive than steady blowing or suction (e.g. LFC) because they do not require any complex fluid ducting or plumbing system. Furthermore, synthetic jets have been found to achieve similar effectiveness to steady blowing or suction with considerably smaller momentum (and hence lower energy inputs) [81].

The synthetic jet is a zero net mass flux device, which consists of an oscillating diaphragm inside a sealed cavity with a small orifice or slit through which an air jet enters (suctioning) and exits (blowing) due to the diaphragm oscillatory displacement. The expelled fluid forms a shear layer with the surrounding fluid that results in a series of rolling vortices. If the oscillation of the diaphragm is sufficient in amplitude and frequency, the vortices have sufficient inertia to escape re-entrainment into the cavity resulting in an air jet consisting of propagating vortices [6, 63].

Even though the net mass change of fluid through the cavity during each cycle of the diaphragm motion is zero, the net momentum transferred into the fluid is non-zero.

Zero net mass flux synthetic jet actuators produce an air jet with unique effects, which are impossible to be obtained with steady suction or blowing. Figure 5.11 shows a schematic of a typical synthetic jet or "zero net mass flux" device.

The vortical structures promote boundary layer mixing and hence momentum exchange between the outer and inner parts of the boundary layer. The enhanced mixing causes the reattachment of the separated shear layer. Hence, the actuators act as turbulators, energizing the boundary layer and thus eliminating massive separation.

Synthetic jet systems have been produced using speakers, compressed air, air pumps, and bimorph diaphragms. All of these techniques add weight, require real estate, and add complexity to an aeroplane, making these options difficult to be used in practice.

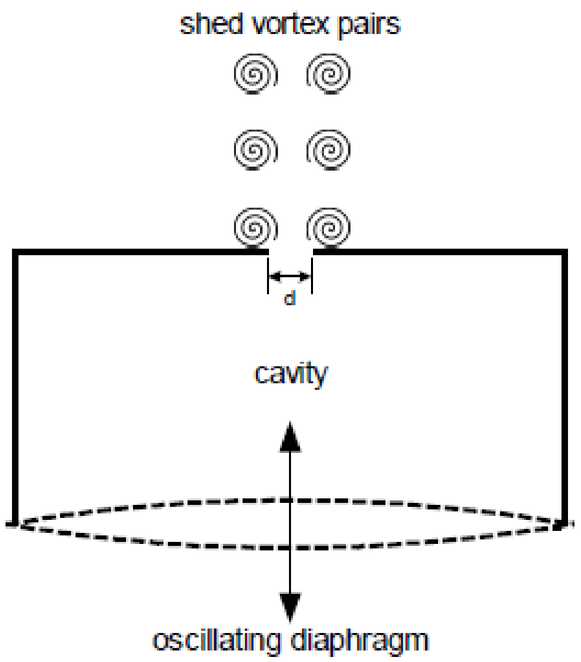


Figure 5.11: Schematic of a synthetic jet (taken from [6]).

Piezoelectric actuators are instead an attractive solution because of their lightweight and fast time response; they have no contacts, they are highly reliable, and most importantly they are cheap to produce [5]. When subject to an electric field, piezoelectric actuators such as lead zirconate titanate ceramics produce mechanical strain or alternately generate an electric charge when subjected to a mechanical strain. This property gives piezoelectric materials the ability to act as actuators, hence to be employed in active flow control.

Synthetic jet systems have been tested on the upper surface of a NACA 0015 airfoil [98]. The synthetic jet actuation not only stabilizes the boundary layer either by adding/removing the momentum to/from the boundary layer, but also enhances mixing between inner and outer parts of the boundary layer. Hence, downstream flow separation is prevented or postponed.

Main features of the flow over uncontrolled and controlled airfoils are revealed in Figure 5.12, showing iso-surfaces of the instantaneous vorticity magnitude overlapped with pressure contours predicted by the present LES. The vortical structures present over the suction surface qualitatively indicate the degree of flow separation. In the uncontrolled case (Figure 5.12a), flow massively separates from the half aft portion of the suction surface while the flow separation is dramatically prevented with the synthetic jet actuation in the controlled case (Figure 5.12b). The present synthetic jet actuation with the momentum coefficient of 1.23% produces more than a 70% increase in the lift coefficient. The drag coefficient is found to decrease approximately by 15-18% with the synthetic jet actuation.

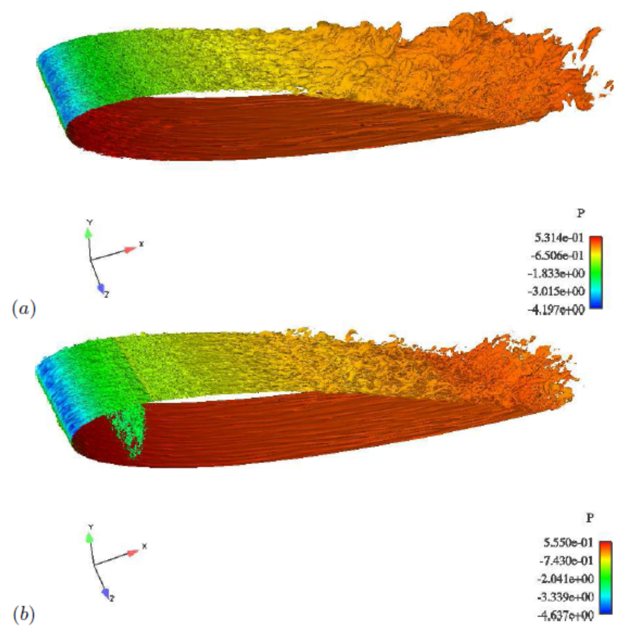


Figure 5.12: Iso surfaces of the instantaneous vorticity magnitude. (a) Uncontrolled case; (b) controlled case. (taken from [98])

The effectiveness of synthetic jets for boundary layer flow control under an adverse pressure gradient condition was investigated as well, by means of experimental tests presented in [45]. The enhancement of this effectiveness by the T-S instability waves was noticed and analyzed. The conclusions are summarized as follows [93].

- The effectiveness of synthetic jets on boundary layer flow control can be enhanced by the natural instability of the boundary layer flow;
- The synthetic jets are effective when the forcing frequency is low, in the range of T-S frequencies;
- The effectiveness of synthetic jets may depend more on the forcing frequency rather than on the forcing voltage;
- Synthetic jets can play dual roles in resisting separation by accelerating turbulence and reducing turbulence in a naturally turbulent boundary layer;
- The effectiveness of the synthetic jet actuator in boundary layer control may not be predictable with only the output of the synthetic jet actuator operating in a condition without cross flow;
- Synthetic jet actuators operating at lower forcing frequency may be more preferable than those at higher forcing frequency, because the audible noise generated at higher frequencies is avoided.

Synthetic jet actuators have shown their effectiveness in reducing pressure drag from turbulent boundary layer separation also during wind tunnel tests and numerical simulations performed on a helicopter fuselage model [40]. A similar application on a tilt rotor seems to be plausible.

The main advantages connected with these devices are here reported:

- As an active flow control device, synthetic jet actuators allow to attain a large effect using a small, localized energy;
- Since zero net mass flux is involved, the synthetic jet device has the advantage of eliminating the need for plumbing connections and an internal air supply, then allowing a reduction in weight and costs;
- Unlike Passive control devices, e.g. vortex generators, the synthetic jet device does not introduce a drag penalty when the flow does not separate;
- Piezoelectric devices have the advantages of faster response, good reliability, low cost, and a reduction in weight and space;
- Piezoelectric devices are characterized by an efficiency larger than 90% [93];
- Micro-electro-mechanical systems (MEMS) technology will allow producing arrays of a large number of these actuators (necessary for turbulence control) at a reasonable cost; item If the piezoelectric actuators were driven at a frequency of operation (but out of phase) to counter a known noise source, the technology certainly could be a viable noise reduction technology (active noise suppression).

On the other hand, the disadvantages the designer has to deal with, are

- Designing such a system is difficult. The cavity volume, neck length, slot size, diaphragm area, and frequency must be correctly designed, in order to obtain good results;
- The complex unsteady flow is hard to be computed by means of traditional computational fluid dynamics (CFD) techniques, and instead requires the unsteady CFD codes employment, which, in turn, requires huge computational resources and long turnaround time;
- Significant noise levels may be introduced by operating these devices at a resonant condition.

5.5.2 Vortex Generators

As an example of flow control on afterbodies, one can consider the effect of vortex generators on the flow along the aft fuselage of a transport aircraft and the corresponding influence on drag. In general, the aft ends of transport aircraft fuselages are tapered asymmetrically with pronounced upsweep of the lower contour to facilitate rotating in the pitch lane on landing and take-off and to meet operational requirements. The longitudinal pressure gradients due to the upsweep result in a rapid boundary layer growth and eventually in separation and vortex shedding accompanied by an increase in fuselage form drag. The use of vortex generators located, for instance, at the beginning of the upswept region as shown in Figure 5.13, is promising to reduce *pressure drag* [85]. The basic principle which

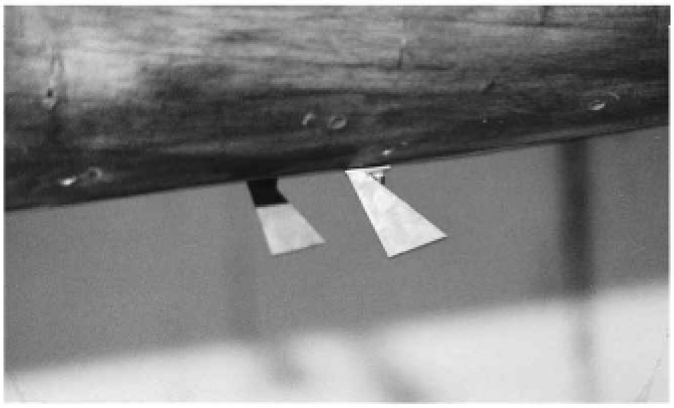


Figure 5.13: Model mounted in the wind tunnel: detail of vortex generators (taken from [85]).

suggests the use of vortex generators to energize a boundary layer in the presence of an adverse pressure gradient is based on the enhanced mixing produced by the vortices between the high energy flow at the edge of the boundary layer and the low energy flow in the boundary layer. In this manner, the energized boundary layer flow can withstand a higher adverse pressure gradient without the occurrence of separation, which considerably increases the pressure drag. Vortex generators have been tested also on a C-130 aircraft model to postpone separation [17].

The main advantages are:

- As a passive technique, they would require little maintenance;
- They do not require the expenditure of additional power from the propulsion units.

The principal disadvantages are:

- They can introduce a drag penalty when the flow does not separate, e.g. during cruise condition;
- Passive solutions like Vortex Generators cannot control the flow velocity for different flight conditions because they are mounted to the surface permanently and there is no possibility to change their shape or position during the flight conditions.

In Figure 5.14, Vortex Generators of various shapes and sizes are depicted.

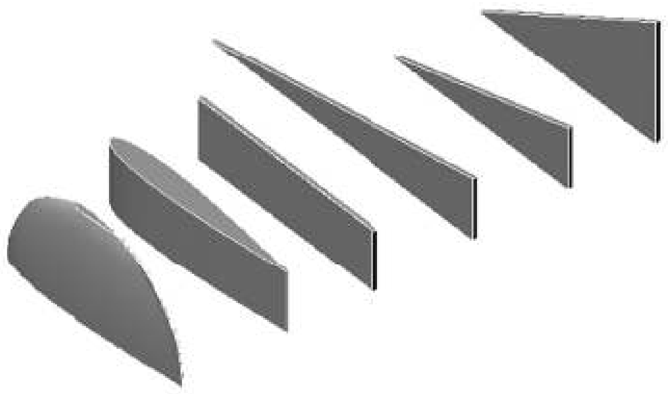


Figure 5.14: Different shapes of vortex generators (taken from [42]).

5.5.3 Active Vortex Generators

Active vortex generators (VGs) may be applicable for separation control during aircraft take-off and landing, and drag reduction during aircraft cruise conditions. Active VGs have a similar effect on the flow structure to passive VGs but these devices potentially have an advantage over conventional VGs because they can eliminate the parasitic drag that arises with VGs, and because the unsteady fluidic actuator can induce slip velocities (which may reduce drag) during cruise conditions. Pulsed, air-jet vortex generators (Figure 5.15) are placed upstream of the region of expected flow separation. When operating, the vortex generators create streamwise vortices in the lower/middle regions of the boundary layer which have been demonstrated to lead to substantial improvements in the shape factor, and hence reduced susceptibility to flow separation. Each actuator comprises an orifice which is pitched at an angle 45° to the surface and skewed at an angle of approximately 90° to the local onset flow. Typically the orifice has a diameter of between approximately 5 and 15% of the local boundary-layer thickness. Each orifice is supplied with a pressurized air mass flow, which is controlled by a valve located below the orifice. Each orifice is therefore individually controllable and capable of generating an air-jet on demand in response to a command signal. Typically the actuator is required to operate at a frequency of up to 500 [Hz] with duty cycles (ratio of the time on to the time off) of 50% or less. Each actuator has to be capable of generating an exit jet velocity which is of the same order of magnitude as the local freestream velocity. For typical full-scale application in the region of a leading-edge slat or trailing-edge flap on a medium-sized transport aircraft, the required orifice diameter would be of the order of 200-400 [μm] and the peak jet velocity would be of the order of 200-300 [m/s] [97].

The main advantage is:

- Compared with passive vortex generators, they can eliminate the parasitic

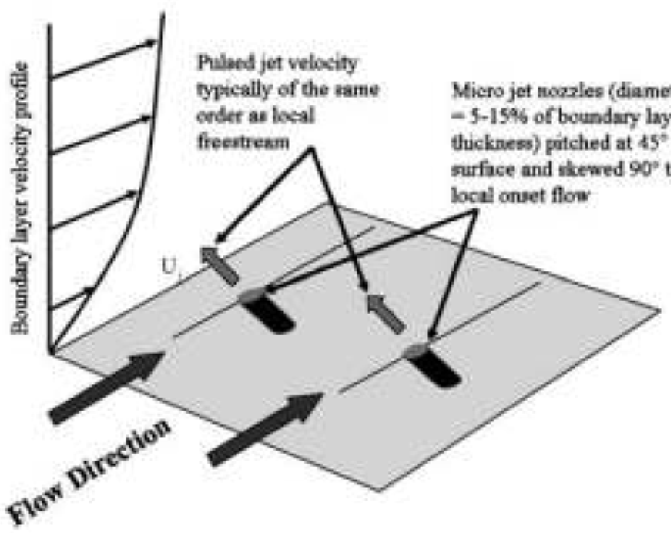

Boundary layer velocity profile
Pulsed jet velocity typically of the same order as local freestream
Micro jet nozzles (diameter = 5-15% of boundary layer thickness) pitched at 45° to surface and skewed 90° to local onset flow
U_j
Flow Direction

Figure 5.15: Schematic of the proposed separation control actuation concept (taken from [97]).

drag that arises with VGs during cruise conditions.

The principal disadvantage is:

- Their effectiveness is closely related to the the aerodynamic optimization of the flow-actuation mechanism [38].

5.5.4 Micro Vortex Generators

Unless large passive VGs which have been used on the aft portion of an aircraft to improve overall performance of the aircraft, micro-VGs (less than one-third of the boundary layer thickness) have been used on the wings of production aircraft to prevent flow separation, primarily during take-off and landing conditions, and to mitigate flap side-edge noise. Lin et al. [59] have shown that using micro-VGs on the flap of a high lift system can mitigate flow separation, leading to a 10 percent increase in lift and a 50 percent reduction in drag. The main advantage is:

- The size reduction significantly reduces the parasitic drag during cruise conditions, and is enabled by the extreme fullness of the mean velocity profile in a turbulent boundary layer.

The principal disadvantage is:

- Such devices must be placed closer to the nominal separation location and therefore may be less suitable than larger devices for situations in which the separation region is not relatively localized.

5.6 Interference drag reduction

When two shapes intersect or are placed in proximity, their pressure distributions and boundary layers can interact with each other, resulting in a net drag of the combination that is higher than the sum of the separate drags. This increment in the drag is known as interference drag.

This juncture drag is due to the occurrence of an unfavourable modification into the local pressure field and the additional rapid straining of the highly dissipative vorticity. The flow field in an unfilletted juncture region is characterized by the formation of a horseshoe vortex structure ahead of the junction [82]. An example of this type of flow occurs when the flow on the fuselage of an aircraft interacts with the wing. The resulting vortex can also reduce the lift and increase the drag of the aircraft. In addition, the regions of concentrated streamwise vorticity that trail behind the juncture decay slowly and interact with components downstream, such as tail assembly and engines. Therefore, the horseshoe vortex can also alter the stability and control characteristics of the aircraft. Proper filleting and fairings can reduce these effects. Another issue related to wing-fuselage or wing-empennage junction, is the flow contamination over the wing or tail surfaces. This phenomenon, by which the turbulence at the wing root is propagated along the attachment line causing the whole flow to become turbulent, would impair the effectiveness of drag reduction techniques, such as HLFC devices previously explained.

Actually, in order to obtain a region of laminar flow on a wing, it is first necessary to ensure that the attachment line remains laminar, namely, to avoid the problem of attachment line contamination.

5.6.1 Gaster bumps

A passive device suitable to avoid the attachment line contamination is the so-called Gaster bump. This leading-edge device, a bracelet or a bump put on the leading edge of the wing quite close to the fuselage, creates a stagnation point on the attachment line, stopping the propagation of turbulence and allowing a new laminar boundary layer to be generated downstream. For application to real aircraft, the bump has also to be aerodynamically as smooth as possible to minimize the additional drag due to its presence. Current CFD techniques are useful tools in the designing of these anti-contamination devices [72].

5.6.2 Slot suction

An active method suitable to prevent turbulent boundary layer to spread along the wing surface, thus preventing attachment line contamination, is the localized wall suction. ONERA©carried out wind tunnel tests [72], which proved the effectiveness of active attachment line anti-contamination devices (boundary layer suction). However, it is necessary to undertake studies of roughness effects and wing surface smoothness requirements even if it is clear that active devices are much more robust towards surfaces defects.

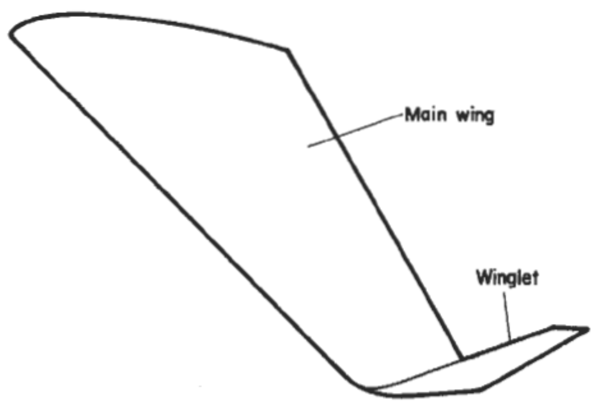


Figure 5.16: Winglet attached to wing tip (taken from [47]).

5.7 Lift-induced drag reduction

The drag due to lift of a finite span wing is usually due primarily to the induced drag associated with the shedding of vorticity along the span and, in particular, at the tip. It is also due in part to an increase in the skin friction and form drag associated with an increase in the lift coefficient (since the vortex structure is concentrated at the tip and induces a downwash over the wing, which reduces the effective angle of attack) [1]. To reduce the induced drag, wings of large *aspect ratio* should be utilized since these enable the tip vortex structures to be separated which reduces the strength of the average induced flow between them. However, the aspect ratio employed in actual airplane configurations is usually constrained by practical considerations, particular those associated with the wing structural weight. When the aspect ratio is increased, the bending moments in the wing structure also increase, with the resulting requirement for more structural material in the wing. Since the performance of the airplane is determined by the structural weight as well as the aerodynamic efficiency, the optimum aspect ratio for the best overall performance must be a compromise [41].

5.7.1 Winglets

The addition of tip-mounted surfaces to a wing can reduce and diffuse the vortex structures arising from the tips. Induced drag reductions result, but these may be offset by unfavourable interference and viscous effects. The winglet is one of the most known concept which can be thought as a device to increase the effective span of the wing. As shown in Figure 5.16, the winglet is a small wing mounted in the swirling flow at the wing tip. The lift on the winglet acts as a side force and, with proper positioning of the winglet, it will have a thrust component in the stream direction. As with the afterbody strakes, the structure of the vortices is somewhat diffused due to the winglets. However, one has to be careful during winglet design process, since such surfaces have significant effects on the structural weight. Actually, loads on the vertical surfaces and the increased loads on the outboard region of the wing associated with adding these surfaces increase the bending moments imposed on the wing structure. Moreover, adding winglets means an increase in the overall wetted area, hence an increase in the skin friction drag. The main advantages are:

- A proper design of winglets can also improve tilt-rotor stability;
- No additional energy is required.

The disadvantages are:

- Adding winglets means an increase in the overall wetted area, hence an increase in the skin friction drag;
- An increase in overall weight has to be taken into account.

5.8 Miscellaneous drag sources

Items included in the following discussion refer to the surface roughness and leakage, protuberances, and momentum losses.

Surface roughness drag includes surfaces irregularities such as rivets heads, seams, waviness in the skin, etc. It is reduced through the use of butt joints on all exterior airframe structures, and flush rivets on streamlined components (fuselage, tail assembly and nacelles). A study [34] indicates that the drag increment due to the discrete roughness of streamlined V/STOL aircraft components is reduced from 9 to 5% through the use of flush rivets over the entire aircraft. Protuberances include such items as antennas, lights, handholds Pitot tubes and windshield wipers. The related drag is relative large since they are bluff bodies. The use of fairings and selective positioning can result in sizeable drag reduction benefits.

Leakage drag results from air that enters or exits the fuselage around access doors, access panels and windows, which are to be sealed in order to reduce this kind of drag.

Momentum losses occur when air stream is diverted into the aircraft for the purposes of cooling the transmissions, hydraulic systems and engines, and providing air for heating-ventilations systems. The drag penalty can be minimized by providing generous inlet radii to prevent air spillage from producing separation, and direct the exhaust air from the system aft to recover a portion of the inlet momentum less.

A momentum loss also results due to the change in velocity of the air entering and exiting the engines at high speeds.

5.9 Regions of application

In Table 5.3, a resume of the tilt-rotor regions suitable for application is carried out. Devices are classified depending on whether they are passive or active tech-

Device	**Energy requirements**	**Region of appplication**
LFC	*Active*	*Wings, fin, tail-plane*
NLF	*Passive*	*Wings, fin, tail-plane*
HLFC	*Active*	*Wings, fin, tail-plane*
Ribelts	*Passive*	*Base fuselage*
LEBUs	*Passive*	*Base fuselage*
Synthetic Jet actuators	*Active*	*Wings, afterbody regions*
Vortex generators	*Active/Passive*	*Afterbody regions*
Micro-vortex generators	*Passive*	*Wings, rear fuselage*
Gaster bump	*Passive*	*Fin*
Slot suction	*Active*	*Wing/fuselage junction*
Winglets	*Passive*	*Tail plane*

Table 5.3: Tilt-rotor regions suitable for application of the mentioned devices.

niques.

Chapter 6

Introduction to Intake Aerodynamics

The fuel consumption of turboshaft engines depends not only on the output power but also on the aerodynamic performance of air inlets and to a lesser extent, of exhaust nozzles. Engine performance loss as compared to the ground bench reference may be quite substantial, in the order of 5% or more on some helicopters. In this chapter, the intake aerodynamics issue will be addressed. The reason for this is to conceptually contextualize the optimization problem presented in Part I of chapter 7. Moreover, in section 6.4.1, the intake performance parameters and the way they are evaluated for the AW101 Intake Test case will be addressed.

6.1 Preliminar considerations

From the point of view of the propulsive system adopted, an helicopter is equivalent to a turboprop-shaft driven aircraft.

The main difference with the turboprop-driven aircraft lies in that the exhaust velocity is negligible, whereas all the output power is required to drive the main rotor shaft.

As others air-breathing jet engines, turboshafts must be supplied with air from the atmosphere in which the aircraft is operating. The intake, or *inlet*, is that part of the aircraft structure which performs such supply duty. Its aerodynamic behaviour can be roughly understood by analyzing the progress of two stream flows from an undisturbed condition far ahead of the aircraft.

- **Internal flow**: it enters the intake and the aerodynamicist's task is to maximize the condition of pressure and flow uniformity with which it arrives at the engine face. These properties are vital to the performance and stability of engine operation.

- **External flow**: it passes around the intake as part of the general airflow over the aircraft itself. Although not quantitatively definable in the manner of the internal flow, the external flow is none-the-less important to the aircraft aerodynamics, principally to aircraft drag, and its effect must be assessed by suitable difference procedures.

The subject of intake aerodynamics is the study of both the internal and the external flow as defined in this way, with the scope of quantify their influence on the aircraft efficiency and performance. The system requirements, for an aircraft intake design, depend on the aircraft mission specification.

Hereafter, the main design goals for intake aerodynamics are summarized:

- To provide the engine with adequate mass flow rate, at the proper Mach number, at the engine face throughout the whole flight envelope;
- To guarantee a smooth and uniform feed flow into the engine compressor;
- To integrate well within the nacelle or the fuselage, leading in a low installation drag.

Nevertheless, when addressing intake design, also non-aerodynamic matters have to be taken into consideration like, for example:

- To provide acoustic absorption of fan/engine noise;
- To provide a particles separator to avoid Foreign Object Damage (FOD);
- To be characterized by low weight and low cost of manufacture;
- To provide favorable accessibility, reliability and reparability.

The next chapter will deal with the topic of subsonic inlets, and both internal and external flow patterns will be considered. The importance of intake efficiency of both these two kinds of flow patterns will be pointed out.

6.2 Subsonic inlets

An engine installed in an aircraft must be provided with an air intake and a ducting system. In Figure 6.1, a CAD drawing of the intake system, along with particle separator and exhaust duct, is illustrated. The sketch refers in particular to the intake installed into the wing-mounted engine of ERICA tiltrotor (see Figure 5.1).

At this point, it is also useful to illustrate the main features of a turbo-shaft air feeding system. For turboshaft engines, the airflow entering the compressor must have low Mach number, in the range 0.4 to 0.7, being the upper part of the range suitable for transonic compressors only. If the engine is designed for subsonic cruise, the inlet must be designed to act as a diffuser with reasonably gentle diffusion from flight Mach number to a lower Mach number.

The inlet must be designed to prevent boundary layer separation, even when the axis of the intake is not perfectly aligned with the streamline direction far upstream of the inlet. In other words, the performance of the inlet must not be

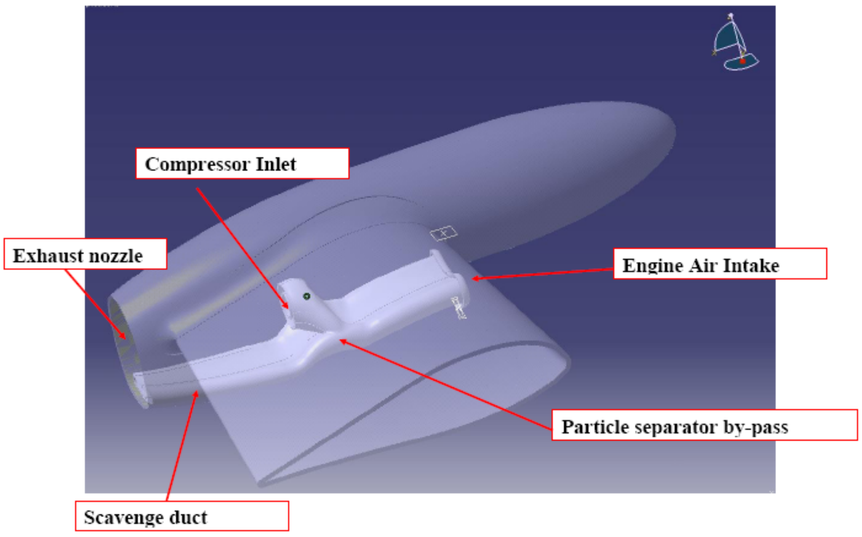


Figure 6.1: CAD model of the ERICA intake and exhaust system.

excessively sensitive to pitch (up-and-down) and yaw (side-to-side) motions of the aircraft. Moreover, this is one of the most challenging goal in intake aerodynamic design, because of the *adverse pressure gradient* within the diffuser duct, which increases the attitude of the boundary layer to separate.

As a matter of fact, one recognizes that in a real flow environment, boundary layers are formed, and have the tendency to separate when exposed to a rising static pressure, known as an adverse pressure gradient. Therefore, the behaviour of a diffuser is expected to be driven by the viscous region near its walls, i.e., the state of the boundary layer as in attached, separated, or transitory (unsteady) conditions.

The main geometrical features of a subsonic diffuser are:

1. The shape of the cross section;
2. The shape of the centerline;
3. The stream-wise area variation.

The three basic geometries of interest are presented in Figure 6.2:

Another feature of a real diffuser is the geometrical shape of its centerline. Often the engine face is hidden from an observer looking through the inlet. The feature of a hidden engine face offers the potential of masking the radar reflections off the engine face, which is advantageous in a stealth aircraft. Turboshafts, such as those mounted on the helicopter whose intake is addressed in Chapter 7, due to the presence of the propeller shaft, require an S-shaped subsonic diffuser duct to channel air to the engine face. From the fluid dynamics point of view, a curved duct induces a secondary flow pattern, which essentially sets up "pockets" of swirling flow at the duct exit. Often these pockets of swirling flow occur in pairs and are counterrotating. The existence of twin swirling flows through curved pipes

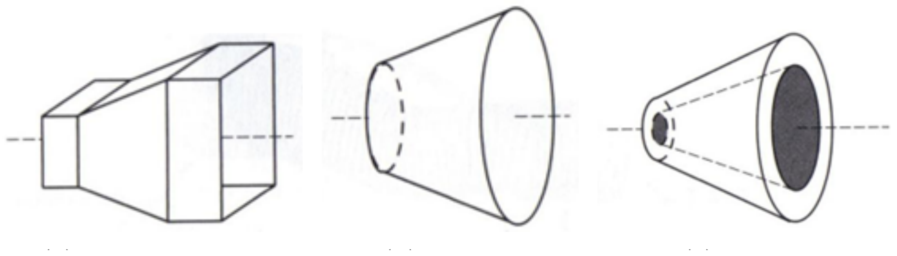


Figure 6.2: Schematic drawing of subsonic diffuser geometry with straight centerline (taken from [28].)

is well known: centrifugal forces push the higher energy stream lines towards the outer radii of the bend while the low energy streamlines are in turn forced to move inwards (see Figure 6.3).

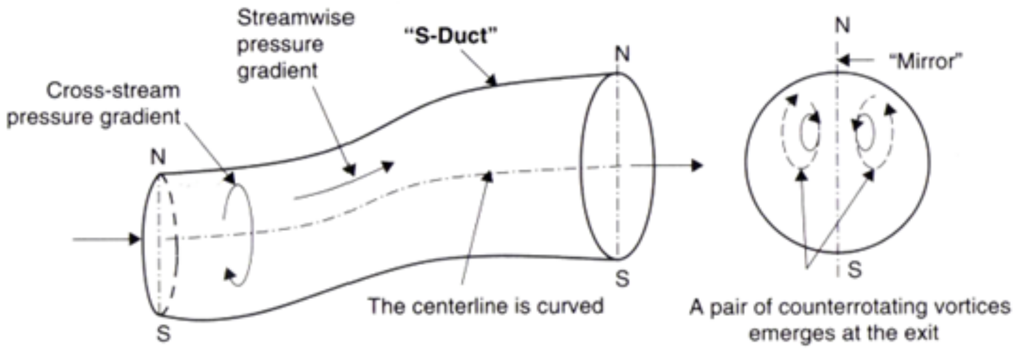

Streamwise pressure gradient
"S-Duct"
Cross-stream pressure gradient
N
S
N
S
The centerline is curved
N
"Mirror"
S
A pair of counterrotating vortices emerges at the exit

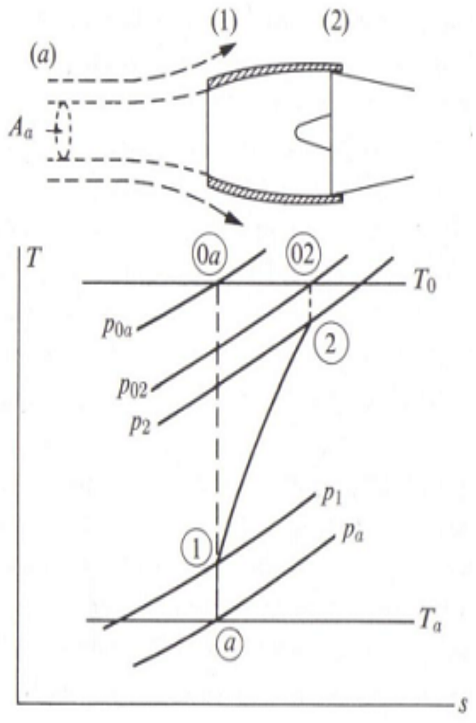
Figure 6.3: Schematic drawing of an S-duct with two pockets of swirling flows (known as the secondary flow patter) generated by the bends (taken from [28].)

The swirling flow induced by the curved duct leads in a local variation of the relative flow angle at the compressor inlet and, in severe situations, it may produce rotating stall instability of the compressor rotor.

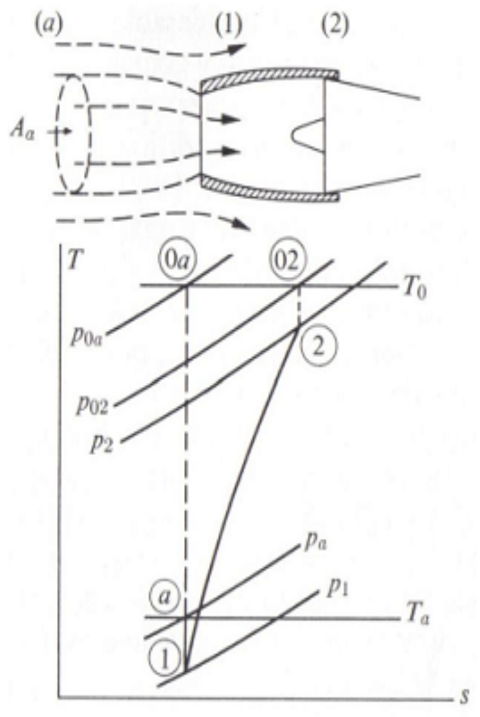
6.3 Inlet Operating Conditions

Depending on the flight speed and the mass flow demanded by the engine, the inlet may have to operate with a wide range of incident stream conditions. Figure 6.4 shows the streamline patterns for two typical subsonic conditions and the corresponding thermodynamic path of an "average" fluid particle. During level cruise, the streamline pattern may include some deceleration of the entering fluid external to the inlet plane (as depicted in Figure 6.4-a).

On the contrary, during low speed, high trust, operations like take-off and climb, the same engine requires more mass flow and the streamlines in the pre-



(**a**) High speed or low mass flow



(**b**) Low speed or high mass flow

Figure 6.4: Typical stream line patterns in subsonic inlets (taken from [43].)

entry zone may resemble as in Figure 6.4-b, which shows external acceleration of the pre-entry stream. In both cases there is an isentropic external change of thermodynamic state. For given air velocities of the undisturbed flow (station *a*, far ahead of the aircraft) and at the compressor inlet (station 2), external acceleration raises the inlet velocity and in turn decreases the inlet pressure, thereby increasing the intensity of the adverse pressure gradient along the diffuser. In severe situations this effect may lead in stall of the diffuser because of boundary layer separation induced by adverse pressure gradient. Conversely, external deceleration reduce the pressure rise across the diffuser and hence the boundary layer situation is less critical. Therefore, the inlet area is chosen so as to minimize external acceleration during high engine load operations, which also leads to external deceleration in normal operating conditions.

6.3.1 Internal flow

Qualitatively the flow in the inlet behaves as though it were in a "diffuser", which is a common element in fluid machinery.

A better term might be decelerator, since the device is not primarily concerned with molecular or turbulent diffusion, but in this work the traditional term diffuser will be retained and defined to mean any section of a duct in which fluid momentum decreases and pressure rises, no work being done.

Considerable experimental and analytical work has been carried out on cylindrical (especially conical and annular) diffusers, but little of this is directly applicable to subsonic aircraft inlets. The reason is that most of the work on diffusers focuses on the conditions that are related to maximum pressure recovery, which is usually associated with a highly non-uniform exit velocity profile and perhaps even with some flow unsteadiness.

In typical subsonic aircraft inlets, there is a stringent requirement that the flow velocity entering the compressor be steady and uniform. Consequently, inlet design does not depend so much on the results of diffusers research as on potential flow calculations, coupled with boundary layer calculations and followed by wind tunnel testing to assess inlet performance under a wide range of test conditions.

Nowadays, intake design procedure is assisted by Computational Fluid Dynamics (CFD) analyses involving 3D Navier-Stokes equations.

In actual engine inlets, separation can take place in any of the three zones shown in Figure 6.5. Separation of the external flow in zone 1 may result from local high velocities and subsequent deceleration over the outer surface. This circumstance, which leads to high nacelle drag, will be discussed in section 6.6. Separation on the internal surfaces may take place in either zone 2 or zone 3,

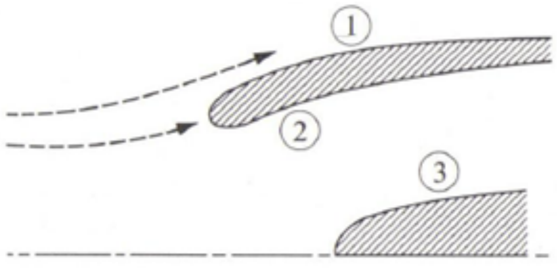


Figure 6.5: Possible locations of boundary layer separation (taken from [43].)

depending on the geometry of the duct and the operating conditions. Zone 3 may be the scene of quite large adverse pressure gradients, since the flow accelerates around the nose of the centre body, and then decelerates as the curvature decreases. At high angles of attack, all three zones could be subjected to unusual pressure gradients.

6.3.2 External flow

As it has already pointed out, intake design requires a trade-off between external and internal deceleration to avoid boundary layer separation in both regions. Figure 6.6 shows a typical flow pattern characterized by large external deceleration. Flowing over the lip of the inlet, the external flow is accelerated to high velocity, which leads in a low pressure region, adversely affecting the boundary layer flow in two different ways:

1. For entirely subsonic flow, the low-pressure region must be followed by a region of rising pressure in which the boundary layer may separate (if

pressure gradients are high enough); hence one might expect a limiting low pressure p_{min} or, equivalently, a maximum local velocity u_{max}, beyond which boundary layer separation can be expected downstream. This is the case of a typical Tiltrotor nacelle.

2. For higher flight velocities (or higher local accelerations), partially supersonic flow can occur. Local supersonic regions usually end abruptly in a shock, and the shock-wall intersection may cause a boundary layer separation.

Whatever the cause, boundary layer separation is to be avoided, since it results in poor pressure recovery in the flow over the after portions of the aircraft or engine housing. This, of course, would result in a net rearward force or drag on the body. A simple one-dimensional, incompressible, model, introduced by *Küchemann*

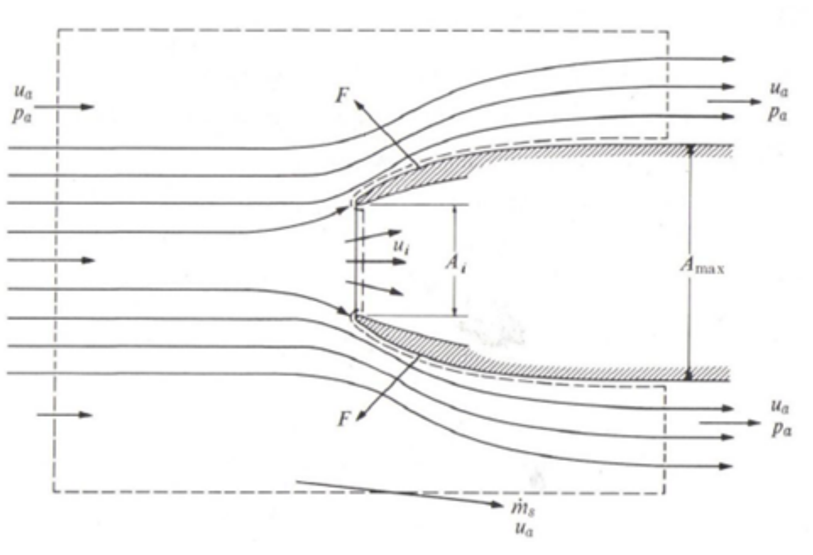


Figure 6.6: One-dimensional model for external deceleration study(taken from [28].)

and Weber ([53]), is useful to understand the relationship between the external flow behavior to the extent of external deceleration: in reference to Figure 6.6, the external cross section of the intake grows to a maximum area A_{max} , and the body remains cylindrical downstream to this point. The control volume indicated extends far from the intake on the sides and upstream ends, crosses the inlet at its minimum area A_i, passes over the inlet surface, and extends downstream, far enough to consider an exit velocity equal to the undisturbed flight velocity u_a (Figure 6.6). Thus, all the external flow enters and leaves the control volume with an axial velocity u_a, while the internal flow enters with the same velocity u_a and leaves with a velocity u_i, assuming one-dimensional flow over the inlet. The equation which describes the flow characteristics in this simplified intake model is the follow:

$$\frac{A_{max}}{A_i} = 1 + \frac{\left(1 - \frac{u_i}{u_a}\right)^2 (1 - c_{Pmax})}{s c_{Pmax}} \tag{6.1}$$

where s is a value dependent on the shape of the nacelle and c_{Pmax} is the maximum value of the pressure coefficient over the nacelle. Assuming s = 0.5, for purposes of illustration, it is shown in Figure 6.7, the dependence of the size of the external surface necessary to prevent external boundary layer separation, for any given value of $\frac{u_i}{u_a}$.

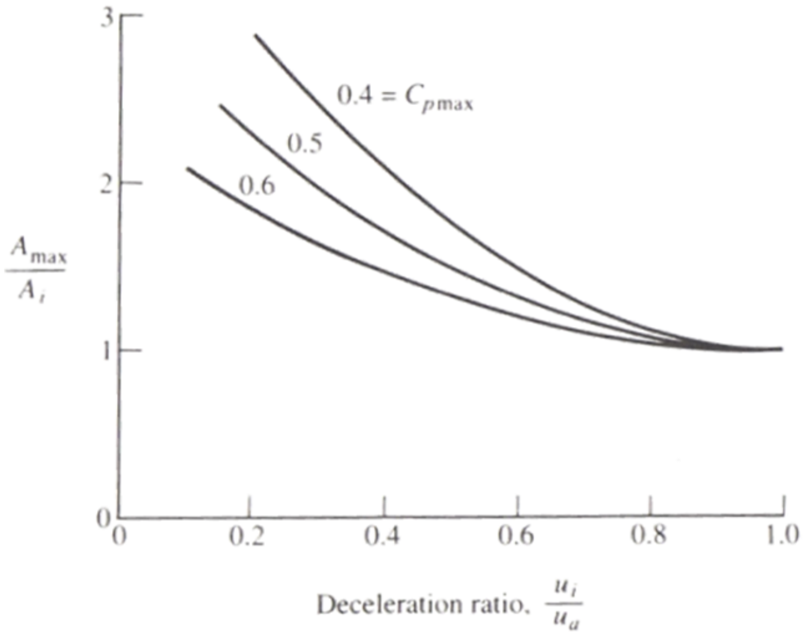


Figure 6.7: Minimum frontal area ratio for various values of c_{Pmax} (s = 0.5 in equation 6.1)(taken from [43].)

The main point here is that the larger the external deceleration (i.e., the smaller the value $\frac{u_i}{u_a}$), the larger must be the size of the nacelle if one is to prevent excessive drag. Even in the absence of separation, the larger the nacelle, the larger the aerodynamic drag on it. However, if the external deceleration is modest, its effect on minimum nacelle size is quiet small. The use of partial internal deceleration is, of course, doubly effective in reducing maximum diameter because it permits a reduction in both A_i and $\frac{A_{max}}{A_i}$. To summarize, even though this analysis pertains to a simplified picture of the real flow around inlets, nevertheless, it shows that the performance of an inlet depends on the pressure gradient on both internal and external surfaces. The external pressure rise is fixed by the external compression and the ratio $(\frac{A_{max}}{A_i})$ of maximum area to inlet area. The internal pressure rise depends on the reduction of velocity between entry to the inlet diffuser and entry to the compressor. Nacelle size required for low drag can be quiet strongly dependent on the degree of external deceleration.

As far as nacelle design is concerned, drag reduction may be achieved through a hybrid laminar flow control (please read section 5.4.3).

Moreover, in realistic analyses, one must consider compressibility effects.

It is to be pointed out that main rotor swirl effect on the boundary layer development has to be taken into account, since it intimately affects its attitude to separate or reattach. As a matter of fact, the main rotor slipstream may act so as to energize the boundary layer and, hence to prevent its separations. Moreover, 3D effects (characterizing the real flows) and boundary layer effects of a typical intake-fuselage configuration of an helicopter are not negligible. Once again, Computational Fluid Dynamics (CFD) analyses involving 3D Navier-Stokes equations can effectively assist engineers during more advanced design steps.

6.4 Intake-Engine Integration: Flow Distortion Issue

As mentioned in sections 6.3 and 6.2, the air diffusion process, which occurs in the intake system, must be accomplished with the least loss in total pressure, with the best flow distribution attainable at the compressor face and with the least amount of drag.

In particular, the level of *flow distortion*, which an intake creates at the engine aspiration plane, affects the performance and stability of the compressor. The term distortion means simply a non-uniformity in the flow, which is characterized by a destabilizing impact on the compressor performance, by means of reduction of the stability margin (of compressor or fan), potentially to the level of compressor stall or engine surge.

A lot of types of flow distortion can be defined, most of them are carefully described in reference [28].

The main sources of flow distortions, that quantifies the uniformity of the flow distribution at the intake/engine interface, can be classified into the following categories [28]:

1. Total pressure distortion $p_t(r, \theta)$;
2. Flow angle distortion $\alpha(r, \theta)$;
3. Secondary flow-swirl at the engine face $C_\theta(r, \theta)$;
4. Total temperature distortion $T_t(r, \theta)$;
5. Entropy distortion $s(r, \theta)$;
6. Combinations of some or all of the above distortions.

The common feature of all different types of distortion is found in their destabilizing impact on the compressor performance. For helicopter inlets applications, the dominant distortion effect is due to the *total pressure distortion*. Steady state total pressure distortion is a measure of spatial non-uniformity of total pressure time average values, over the engine face.

Total temperature and entropy distortions act in parallel with the total pressure one, except when hot gases from external sources are ingested (i.e. exhaust gasses from other aircrafts). Flow angle and swirl distortion impact the compressor behavior in parallel with total pressure distortion, with similar effects; it has been demonstrated that, especially at low incidence flight conditions, total pressure and swirl coefficient have the same trend (an increase of swirl leads to an increase of total pressure distortion) [79]. Moreover, total-pressure distortion may be steady or time-variant ("dynamic") and in the latter case, it may be of the spatially uniform ("buzz") type or of a spatially non-uniform ("turbulence") type. Consistent with the concept of total-total pressure ratio as a representative mean of time-averaged values of total-pressure across the engine face position, is the concept of distortion as a measure of the spatial non-uniformity of those time-averaged values. This will be referred to as "steady-state" distortion, notwithstanding the fact that where large transverse gradients of pressure are present, the flow time-wise can at best be no more than quasi-steady. However, gas turbine engine manufacturers provide limiting values of steady state total pressure distortion, which have to characterize inlets in order to guarantee good engine performance.

Therefore, computation of steady state total pressure distortion is considered sufficient for the intake performance optimization, which is the purpose of this document.

6.4.1 Total pressure distortion issue and Definition of Intake Performance Parameters

The most common measure of intake efficiency is the intake *total pressure ratio* (also referred to as *pressure recovery*). The total pressure ratio represents the efficiency of the intake compression process which transforms the free-stream kinetic energy into static pressure. The *total pressure ratio* definition adopted is the following [79]:

$$\eta_T = \frac{P_f}{P_\infty} \tag{6.2}$$

where

- P_f is total pressure measured at the AIP;
- P_∞ is the free stream total pressure. In this case, it is coincident with the total pressure imposed at the Inlet boundary.

However, for the purpose of the present work, the absolute value of total pressure losses

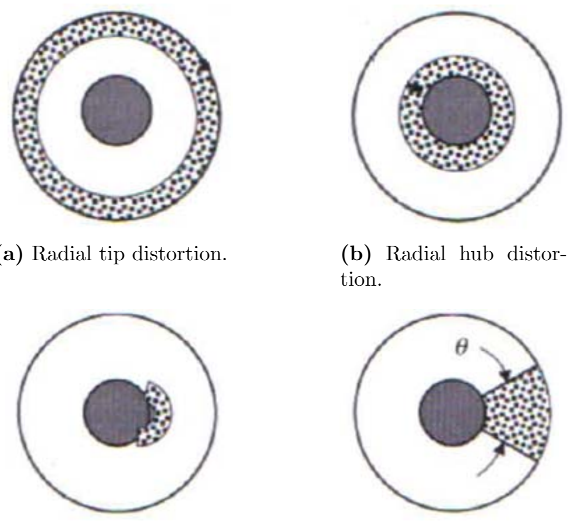
$$\Delta P_T = P_\infty - P_f \tag{6.3}$$

was preferred to the total pressure ratio, for practical reasons due to the optimization problem formulation, and specifically to the use of penalty functions for functional constraints handling, as will be better specified in chapter 7.32. Actually, the absolute values of total pressure losses along the intake, calculated both in forward flight and hovering conditions, will be used for the formulation of the multi-point optimization problem, as will be illustrated in chapter 7.32.

The total pressure loss can be evaluated form CFD simulations by means of standard Fluent® post processing functions; in particular P_f can be determined using a mass-weighted-average surface integral (see [7]) of total pressure over the

AIP surface, while P_∞ is a boundary value imposed at the virtual wind tunnel inlet (please see Figure 7.3). Since those values are known, the computation of ΔP_T is straightforward.

The air diffusion process, occurring in the intake system, must be accomplished with the minimum loss in total-pressure and with the best attainable flow distribution at the intake exit plane. In particular, since aircraft gas turbine engines operate downstream an air intake system, the level of distortion at the compressor face affects the performance and the stability of the compressor itself. Therefore, another important intake performance aspect is the AIP flow distortion.



(a) Radial tip distortion.

(b) Radial hub distortion.

(c) Circumferential hub distortion.

(d) Full-span distortion of angular extent θ.

Figure 6.8: Different types of total pressure distortion (taken from [28].)

In describing the total pressure distortion and its impact on compressor performance, the spatial extent of the spoiled flow is divided according to its radial and circumferential extent, as shown in Figure 6.8. It is necessary to derive a quantitative measure of distortion, from which both the quality of intake flow and the tolerance of an engine can be judged. Distortion coefficients may be defined in various ways; Rolls Royce use the following one:

$$DC(\vartheta) = \frac{P_f - P_\vartheta}{q_f} \tag{6.4}$$

where P_f is the weighted area average total pressure at engine face, q_f is the corresponding mean dynamic head and P_ϑ is the weighted area average total pressure in the “worst” sector of the face, of angle ϑ. The sector “ϑ ” must be of significant extent and 60° is usually regarded as a satisfactory minimum. Thus a commonly used coefficient is $DC(60)$: others which are also used are $DC(90)$ and $DC(120)$ [79].

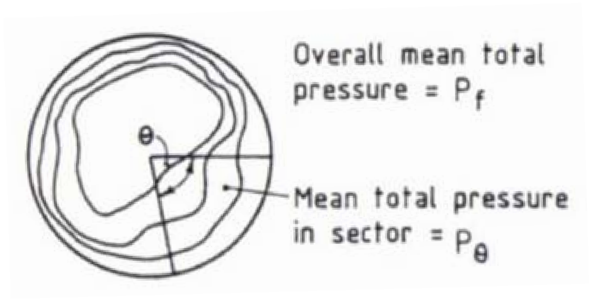

Overall mean total pressure = P_f
θ
Mean total pressure in sector = P_θ

Figure 6.9: Illustration of total-pressure contours and ϑ sector for definition of distortion coefficient (taken from [79].)

Maximum $DC(60)$ is adopted by Rolls-Royce as certification parameter for inlet total pressure distortion and it will be used as the distortion performance parameter throughout the whole document.

$q_f = \frac{1}{2}\rho_f V_f^2$ is the mean dynamic pressure evaluated on the engine face; since, strictly speaking, its definition is dependent on the CFD solution, the following conventions have been chosen:

- $\rho_f = \rho_\infty$; the free-stream density is used in place of the AIP density;
- $V_f = \frac{\dot{m}}{\rho_\infty A_f}$. The AIP velocity is considered equal to the ratio between the AIP mass flow rate and the product of the free-stream density and AIP area.

These assumptions make easier the numerical computation of the $DC(60)$. Moreover, they guarantee that every time a CFD calculation is performed during optimization run, it has the same dynamic pressure (the mass flow rate ingested by the engine being the same) of another one. Even though this definition is strictly valid for incompressible flows only, it may be extended to the case under consideration for determination of the reference velocity. The procedure for the $DC(60)$ computation using Fluent CFD results is outlined in Appendix A: $DC(60)$ computation Procedure.

6.5 Effect of Distortion on Compressor Performance

As an example of wind-tunnel test results, Figure 6.10 is presented, showing the effects of inlet total pressure distortion on compressor stability.

The normal operating line, the undistorted stall boundary, and two corrected shaft speeds of 100% and 93% design are shown in dashed lines. Four solid lines correspond to the stall boundaries of the four distortion patterns simulated at the compressor face via screens. From having the least to the most impact on the stall margin deterioration, the culprits are identified as

1. radial hub;
2. radial tip;

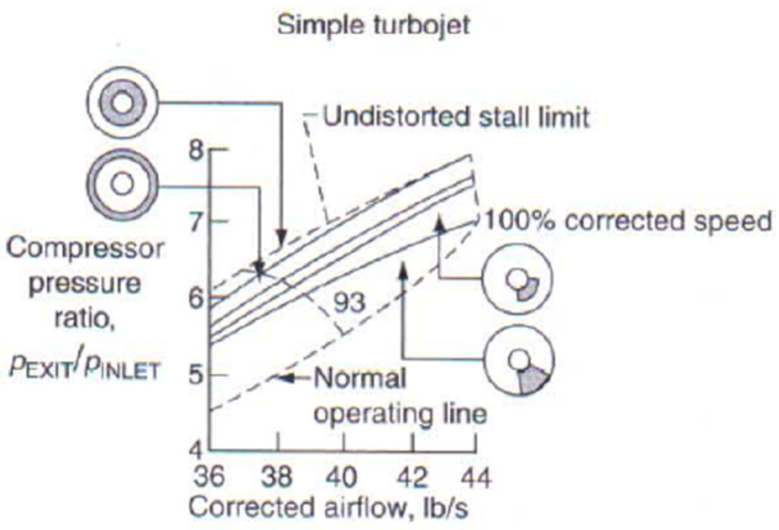


Figure 6.10: Effect of inlet total pressure distortion on compressor stability (taken from [28].)

3. circumferential hub;

4. full-span circumferential distortion

respectively. We also note that the full-span distortion at the 100% corrected speed operates at the stall boundary, i.e., zero-stall margin.

The previous results are confirmed in Figure 6.11, showing the measured effects of inlet distortion on the performance of a 9-stage axial compressor.

The measurements showed that the stall line with circumferential inlet distortion moved considerably to the right, a degradation in compressor stall margin. The general trends of compressor performance with different inlet distortions are elucidated through a series of experiments undertaken by Reid ([71]). A representative set of these results is shown in Figure 6.12, which shows the compressor delivery pressure at the surge line for different types of distortions. Two key aspects may be deduced from these data:

1. As the angular extent of the spoiled sector (low inlet total pressure) is increased, there in an angle above which the exit static pressure changes little (Figure 6.12). This angular extent is often referred to as the critical sector angle.

2. Fixing the total angular extent of the distortion, the effect of sub-dividing it into different number of equal sections is shown in Figure 6.13. The greatest effect on the loss of peak pressure rise is observed when there is only one region. This suggests that inlet distortion patterns, which have a longer length scale and a lower circumferential harmonic content, are the most important.

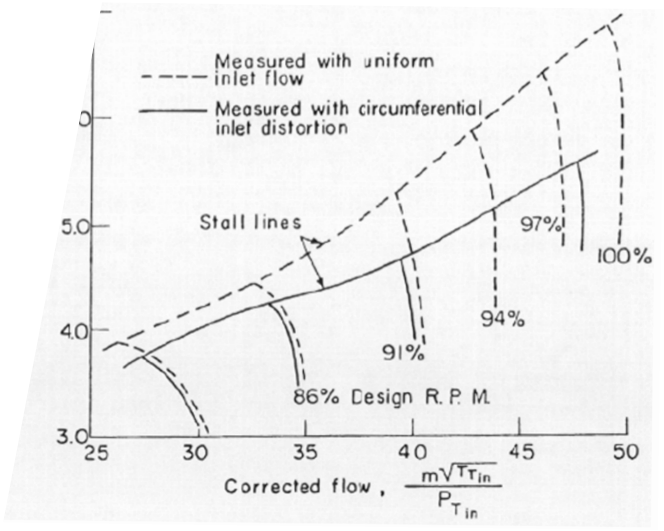


Figure 6.11: Effects of circumferential inlet distortion on multistage axial compressor performance (taken from [77].)

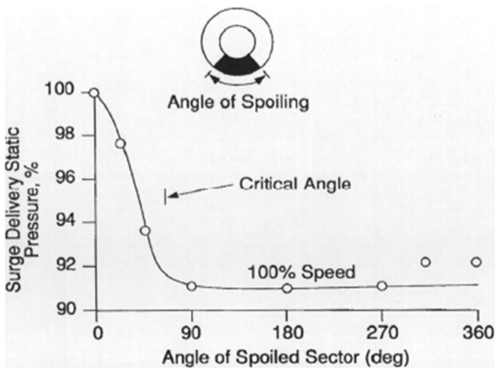


Figure 6.12: Effect of circumferential distortion sector angle on surge pressure ratio (taken from [77].)

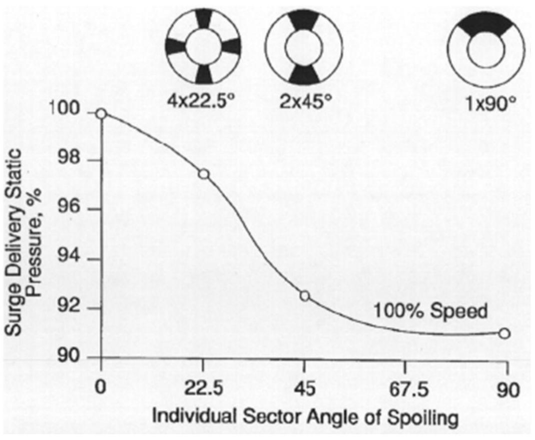


Figure 6.13: Effect of number of sectors, on surge pressure ratio (taken from [77].)

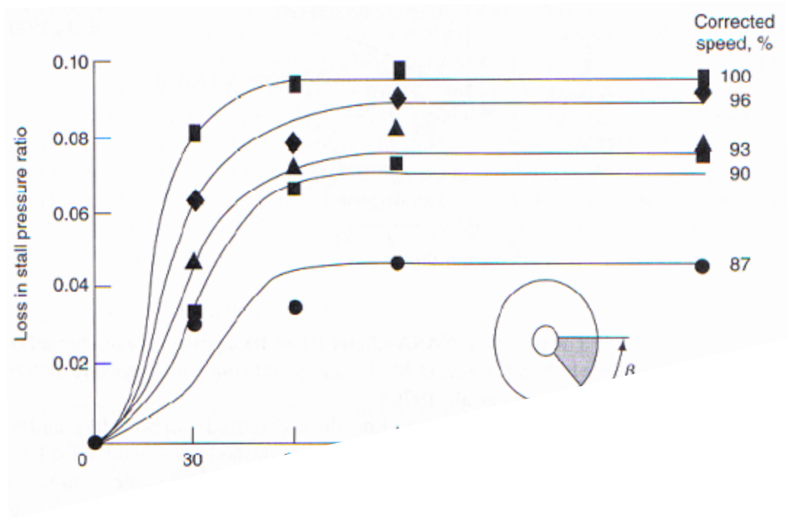


Figure 6.14: Effect of the extent of circumferential spoiling on compressor performance (taken from [28].)

Other research identified a critical circumferential extent of the spoiled sector that causes the maximum loss in the stall pressure ratio of a compressor is at nearly 60°, as evidenced in Figure 6.14.

Chapter 7

Results and Discussion

In this chapter three test cases will be considered in order to demonstrate the strength and versatility of the optimization loop designed during my PhD course. The first test case concerns the aerodynamic shape optimization of the AW101 left air intake. Two objective functions will be optimized by means of the Pareto approach, that is the total pressure losses during forward flight and hovering flight conditions. Functional constraints on the maximum total pressure distortion allowed at the compressor inlet increase further the optimization complexity. Commercial code Fluent was adopted as the CFD solver, and the optimization was run in Windows environment.

The second test case concerns the aerodynamic shape optimization of the ERICA nose region. One objective function was considered, that is the drag force during forward flight condition. Ansys Fluent® was chosen as the CFD solver.

The third test case addresses the aerodynamic shape optimization of the ERICA nose region, this time by exploiting OpenFOAM® as the CFD solver. Visibility requirements were fulfilled, by properly setting up the design variables and limiting the maximum allowable deformation displacement. This case differs from the previous one with regard to both the mesh model and the CFD simulations features.

Part I

Test case A: Aerodynamic shape optimization of the AW101 air intake $\sharp 1$

7.1 Introduction

Specifically, the present work deals with the application of a state-of-the-art optimization methodology to the shape optimization of the full scale AW101 engine air intake ♯1 mounted on a flat plate, with the final aim of reducing the total pressure losses in both forward flight and hover conditions.

The optimization problem considered was of the multi-point type, since the total pressure losses occurring in both forward flight and hover conditions were requested to be minimized simultaneously. In addition, the optimization problem featured some constraints, both of the functional and geometrical type. Specifically, the functional constraints were related to the necessity of maintaining the flow distortions at the Aerodynamic Interface Plane (AIP) within acceptable values: a Penalty Function approach was used to deal with this type of constraint. On the other hand, the requirement that the final intake design is both able to be manufactured and installed into the aircraft led to the definition of a series of geometrical constraints that were considered in this preliminary optimization as well.

First of all, an investigation was carried out in order to identify a proper CFD model set-up for optimization purposes: specifically, the overall CPU time required for optimization needed to be minimized while preserving the numerical accuracy. For this reason, in a first step the effects of varying the total length of the dummy duct located downstream the engine face on the flow field features at the AIP were investigated. As a result, a proper length was identified which made it possible to reproduce intake losses with an acceptable accuracy while featuring the smallest number of 3D elements.

On a second stage, a sensitivity analysis of the CFD model to the computational mesh refinement was performed. As a matter of fact, decreasing the number of grid elements results in a reduction of simulation time, but, at the same time, the model capacity to properly reproduce the intake flow characteristics is progressively deteriorated. Hence, the goal of this second step was to identify a suitable model resulting from a trade-off between these two aspects. In addition to that, the selected CFD model had to be sufficiently robust for optimization purposes: it was required to feature a stable convergence pattern, especially considering the large deformations it undergoes during optimization.

Once the final CFD model was set up, the geometrical parameterization was carried out and the geometrical constraints were implemented.

Finally, the air intake aerodynamic shape optimization was carried out and the results of optimization in terms of achieved margins of improvement with respect to the baseline were discussed in detail.

Results presented here are owned by the Cleansky Partner Consortium *HEAVYcOPTeR* (led by University of Padova), aimed at studying enhanced engine installations for improved performance and lower noise specifically designed for heavy helicopters configuration.

7.2 Optimization Procedure

In this chapter, the whole sequence of operations necessary to carry out the optimization study is described.

As already mentioned, the optimization procedure was not performed in a single step; instead, multiple steps were required to obtain reliable results, while reducing the overall optimization time as much as possible. Each operation will be fully analyzed in the next chapters.

The whole sequence of operations carried out to address a multi-objective optimization problem is shown in Figure 7.1. It is worth mentioning that each block was carried out independently from the others. The upper portion of the flow chart shows the set of preliminary operations that need to be accomplished before starting the successive automatic loop. The procedure started from the elaboration of a baseline geometry CAD model of the intake system of the AW101 intake ♯1 provided by AgustaWestland Ltd in CATIA® format.

The following steps focused on the construction of the baseline CFD model: first of all, the surface mesh was generated using Altair Hypermesh® ([3]) surface meshing tool, then the volume mesh was created by means of Ansys Tgrid®([8]), and finally the CFD case was set up with the commercial CFD code Ansys Fluent®([7]) which is the standard commercial code for CFD simulation adopted at AgustaWestland.

Next, a study on aerodynamic pressure loss sensitivity to the grid features was carried out as described in detail in chapter 7.8. In fact, it is well known that dummy duct length and mesh refinement level greatly influence the computed total pressure distribution over the AIP. Many combinations of these parameters were tested to obtain a model characterized by an acceptable accuracy while featuring the least possible number of cells. all, the surface mesh was generated using Hypermesh® surface meshing tool ([3]), then the volume mesh was created by means of TGrid®, and finally the CFD case was set up using the commercial CFD code Ansys Fluent®([7]), which is the standard commercial code for CFD simulation adopted at AgustaWestland.

Once this study was performed, the sensitivity analysis was concluded, and an optimal mesh model identified. The selected model was "frozen" and submitted to Fluent ® CFD code in order to define the "baseline" aerodynamic features. A section will be dedicated to the baseline model. The latter was then imported into Altair HyperMesh® environment for the geometrical parameterization and constraining.

With the independent variables identified and the parametric model set up, the optimization process could be finally addressed.

The post-processing of the optimization results refers to the lower part of the flow chart in Figure 7.1. The output of the automatic optimization loop was the optimal combination of the design variables of the individuals lying on the Pareto frontier. The designer may identify the individuals of interest among the multiple optimal solutions of the Pareto front and the meshed geometry of each selected individual could be reconstructed using HyperMorph® ([3]).

The final step is the reverse engineering process, which allows obtaining the deformed surface starting from the mesh; this process was performed using CATIA®V5 following the steps reported in section 7.12.

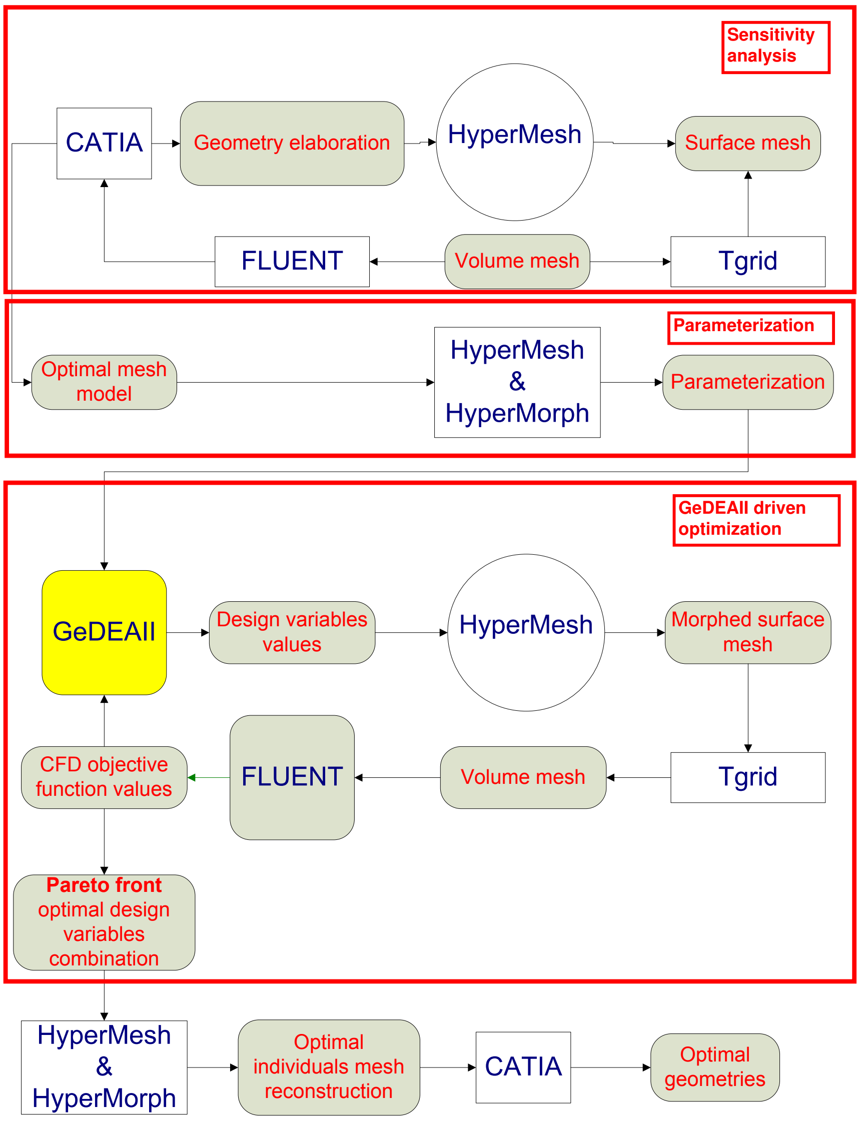


Figure 7.1: Flow chart of the complete optimization procedure.

7.3 The object of the optimization

The scope of the present work was the optimization of the full scale AW101 engine intake♯1, located at pilot left-hand side: the CATIA® model of the intake♯1 mounted on the AW101 fuselage is illustrated in Figure 7.2. However, for the purpose of the trial optimization which was carried out in the present task, the

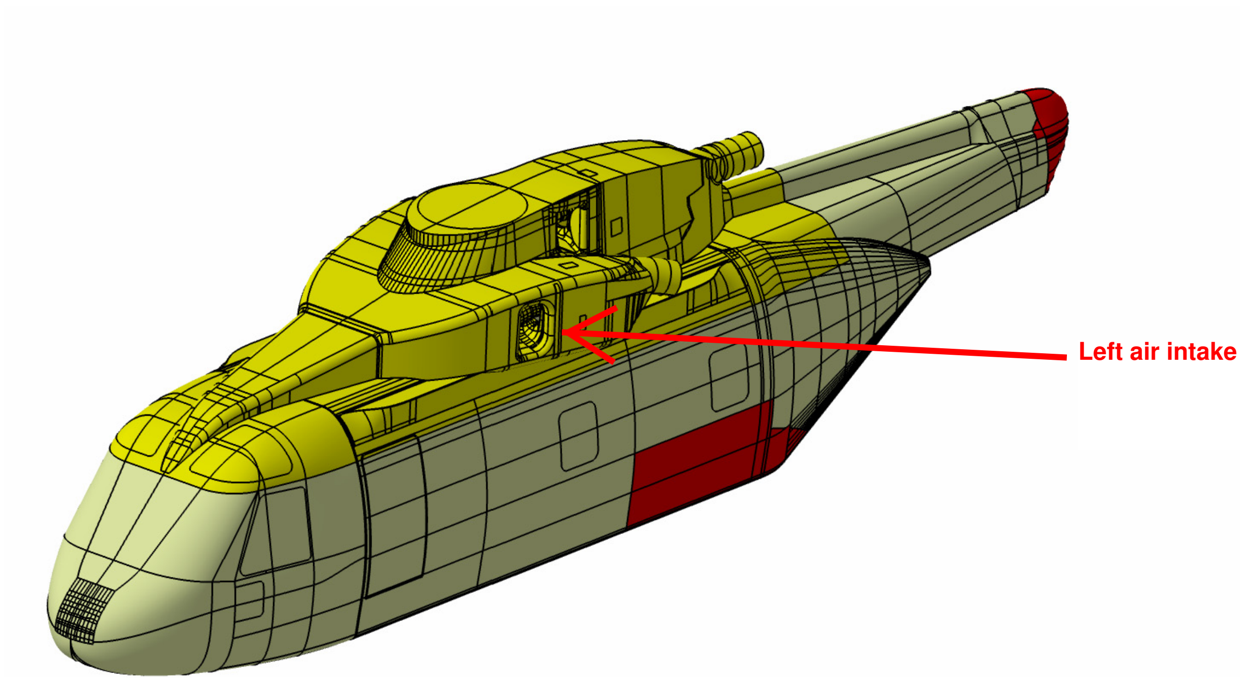


Figure 7.2: CATIA® V5 models of AW101 model without main and tail rotors.

intake CAD model was greatly simplified: actually, the effects of the installation on the AW101 fuselage were neglected in a first approximation, and the intake was mounted over a flat plate, as depicted in Figure 7.3.

A close-up of the isolated engine intake♯1 is illustrated in Figure 7.4, where the AIP location is evidenced. As apparent, a dummy duct was included into the intake model downstream the engine face, as it is usually recommended for this kind of simulations. In fact, since the pressure distribution over the AIP is one of the required outputs of the CFD computations, a pressure outlet boundary condition can not be applied directly over the engine face without forcing the flow field inside the intake to adapt to the imposed boundary condition in a somehow artificial way. Hence, the output boundary condition needs to be applied over a surface simulating the compressor face and located sufficiently downstream the actual intake exit in order not to influence the flow quantities distribution over the AIP. Obviously, the optimization carried out in the present work addressed only the intake component, while the dummy duct was excluded from parameterization.

Two flight conditions were considered simultaneously for the intake optimization, namely cruise forward flight and hover. The main features of the reference flight conditions are summarized in Table 7.1.

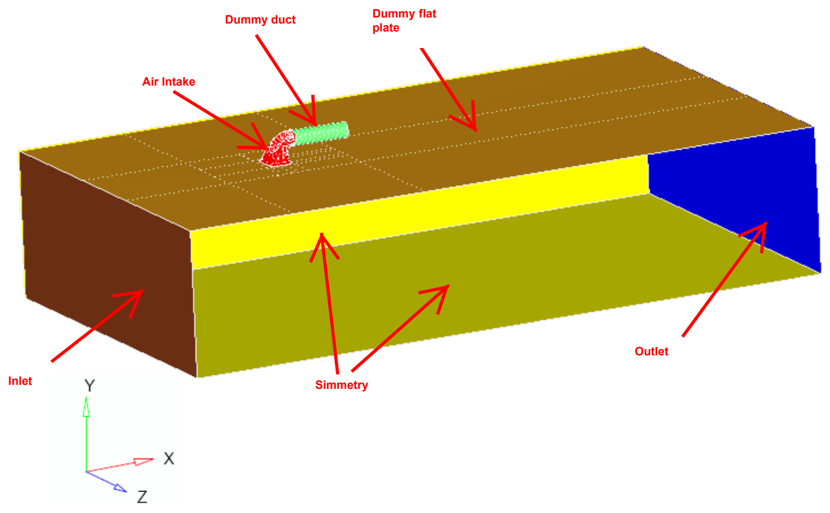


Figure 7.3: Virtual wind-tunnel layout with the AW101 left air intake system installed on a virtual flat plate.

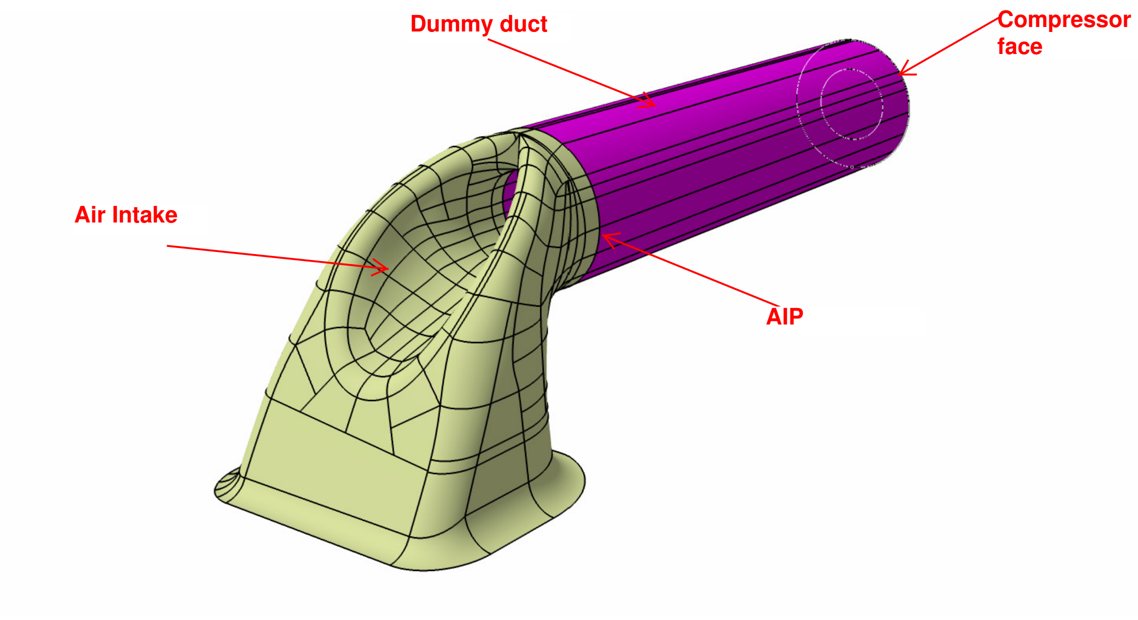


Figure 7.4: Close-up of the CAD model of the isolated air intake♯1

Flight Condition	**True air speed,** $[Kts]$	**True air speed,** $[m/s]$	**Pressure altitude,** $[m]$	**Static pressure,** $[m]$	**Compressor mass flow rate** $[Kg/s]$	**OAT,** $[C]$
Forward Flight	*120*	*61.73*	*609.6 (2000* $[ft]$*)*	*94214*	*4.7219*	284.19
Hovering Flight	*0*	*0*	*0*	*101325*	*5.6472*	288.15

Table 7.1: Flight Conditions for Air Intake optimization.

7.4 Geometry elaboration

As already mentioned, the intake system model was made up of four components, specifically the Air intake itself, the AIP, the dummy duct and the compressor face, depicted in Figure 7.4. The intake system was installed onto a virtual flat plate. The resulting model along with the virtual wind tunnel is depicted in Figure 7.3.

In Figure 7.5, a detailed view of the Air Intake and the Aerodynamic Interface Plane is depicted: the AIP and compressor face are evidenced as well. The AIP location is particularly important, since the intake performance, both in terms of total pressure losses and flow distortions, were measured over this plane.

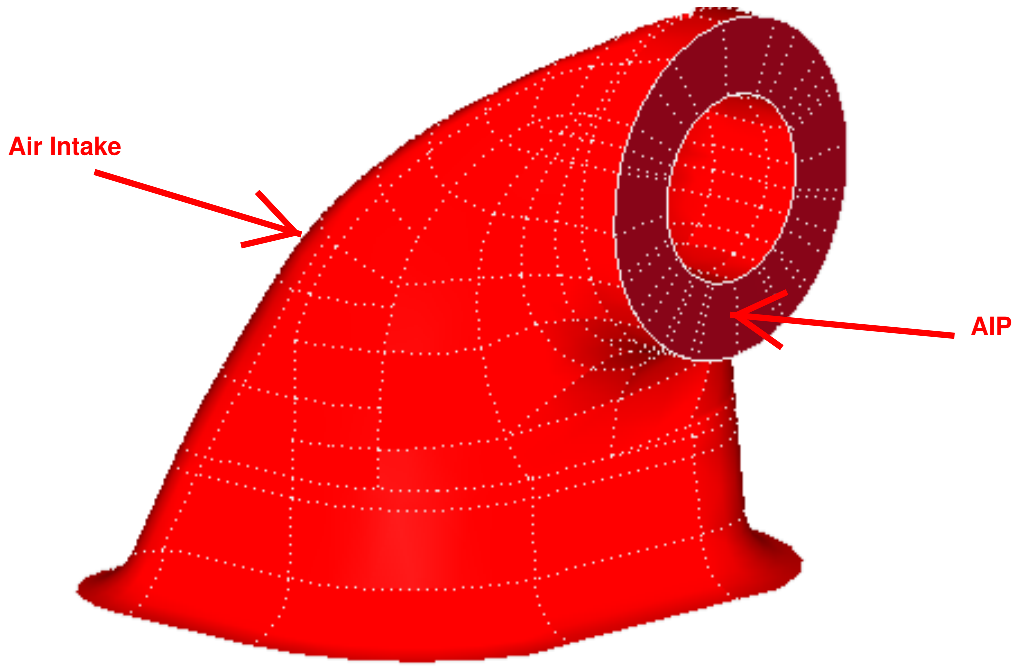


Figure 7.5: A detailed view of the Air Intake and the Aerodynamic Interface Plane.

In Hypermesh® environment, the model surfaces were organized into seven distinct components, namely:

1. **Inlet**, which is depicted in Figure 7.3;

2. **Outlet**, which is depicted in Figure 7.3;

3. **Symmetry**, which is depicted in Figure 7.3;

4. **Air Intake**, which corresponds to the object of the optimization surface, depicted in Figure 7.4;

5. **Dummy duct**, which corresponds to the cylindrical portion of the intake system, depicted in Figure 7.4;

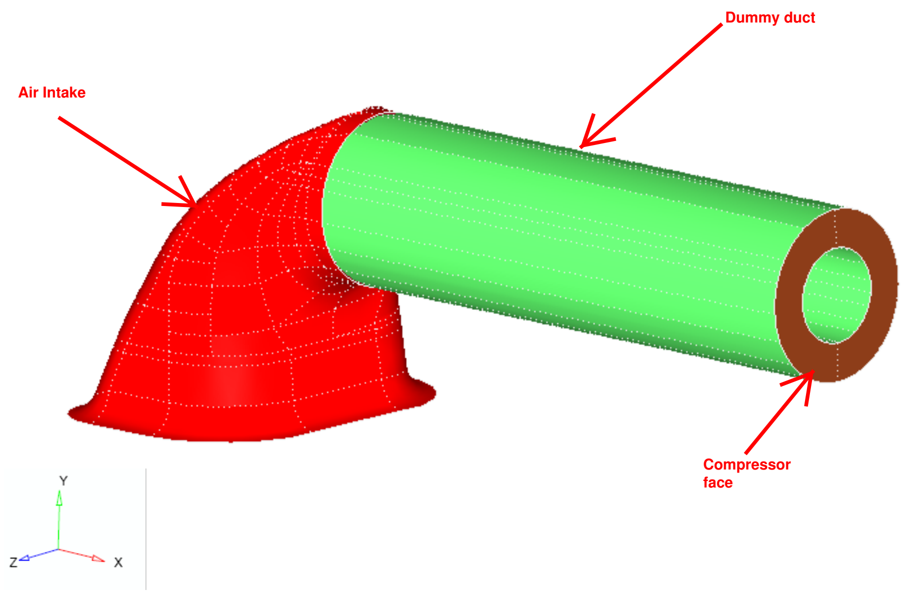


Figure 7.6: A detailed view of the Air Intake with the dummy duct and the compressor face.

6. **AIP**, which is depicted in Figure 7.5;
7. **Compressor face**, which is depicted in Figure 7.6;
8. **Dummy flat plate**, which is depicted in Figure 7.3.

7.5 Mesh generation

As far as the mesh generation is concerned, the software used to carry out the meshing operations was respectively:

1. Hypermesh® surface meshing tool for the surface mesh generation;
2. Ansys Tgrid® for the volume mesh generation.

The choice of HyperMesh® is related to the fact that it was extensively used during all the optimization activities. Therefore, it is worth to building up the mesh models directly using this tool. HyperMesh® allows to import CAD geometries in native as well as standard formats such as (IGES, STEP etc.), to clean up them and to generate high quality surfaces meshes. The choice of Tgrid® is instead related to flow solver selected to carry out the present work, that is Ansys Fluent® . Tgrid® is in fact the standard volume meshing tool when Fluent® is adopted as CFD solver.

7.5.1 Superficial mesh

The Hypermesh® surface meshing tool can be used to mesh the geometry surfaces directly from the CATIA® format, after importing a step model previously exported from CATIA® environment. Both the surface and the volume mesh sizes are discussed in chapter 7.8, since they depend on both the level of surface grid refinement and the virtual wind tunnel dimensions. The surface mesh was more refined over the Air intake, which was the object of optimization, whereas the superficial grid over the dummy duct and the flat plate was slightly coarser; finally, the symmetry, inlet and outlet surface mesh was the coarsest one.

The choice of adopting different mesh sizes was motivated by the attempt to contain the total amount of surface elements (and consequently the number of 3D cells) as much as possible. The transition between two zones with different mesh size was accomplished by properly selecting Hypermesh® Advanced meshing tool parameters. This feature is a really important characteristic of a CFD mesh, and it is fundamental for a good CFD solution convergence. Two different surface mesh distributions on the Air intake surface were investigated during the sensitivity analysis, as will be illustrated in chapter 7.8. The finally selected model was characterized by the following features:

Mesh Type	Triangular
Number of elements	15
Element size within the Air Intake [mm]	36738
Element size within the Dummy Duct [mm]	$15 \div 50$
Element size within the Dummy flat plate [mm]	$10 \div 100$
Element size inlet, outlet, and symmetry planes [mm]	$100 \div 400$
Element size within the AIP [mm]	$11 \div 11$
Maximum 2D skeweness angle [°]	52
Maximum 2D aspect ratio	2.65

Table 7.2: Finally selected number of elements and target element size for each fuselage component.

In particular, the *Surface deviation* algorithm was used in Hypermesh® environment to create the superficial mesh over the Air Intake component. In fact, *Surface deviation* is a meshing algorithm that allows HyperMesh to automatically vary node densities and biasing along curved surface edges to gain a more accurate representation of the surface being meshed. The maximum values of 2D skeweness angle were reached on the dummy flat plate, whereas on the region of greatest interest from a fluid dynamic point of view, the Air Intake component, the values are by far smaller. A sketch of the finally selected surface mesh over the Intake system component is reported in Figure 7.7, while the superficial mesh over the virtual wind tunnel walls is illustrated in Figure 7.8.

In Figure 7.7, the element size transition between the Air Intake and the compressor face is smooth, hence ensuring robustness during the following CFD calculations.

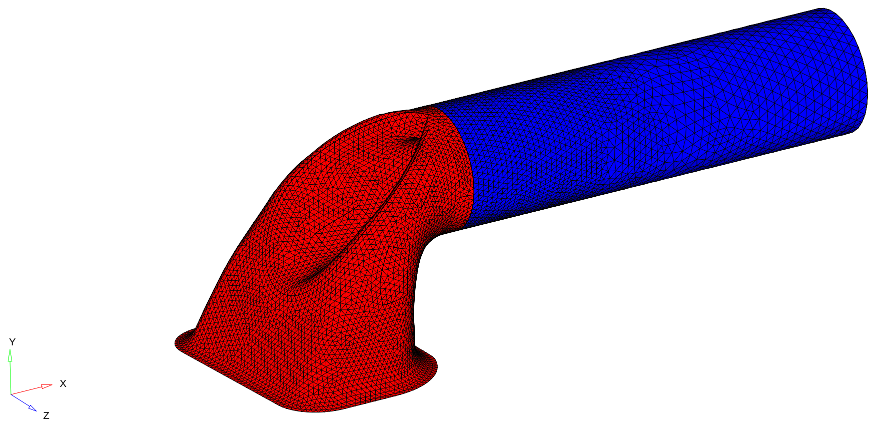


Figure 7.7: Superficial mesh over the Intake system.

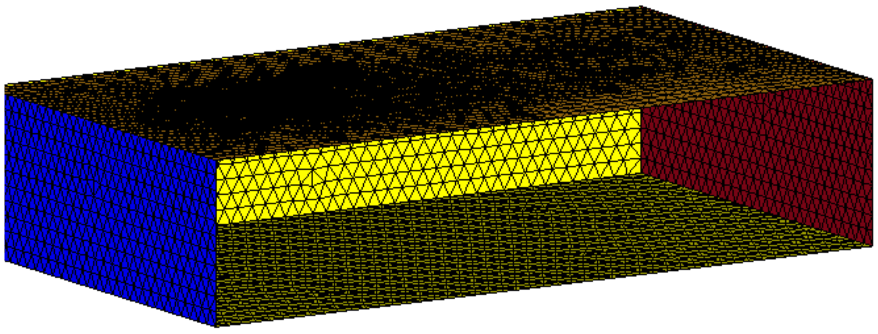


Figure 7.8: Superficial mesh over the virtual wind tunnel walls.

7.5.2 Volume grid

Starting from the surface mesh described above, the volume mesh was generated by means of the CFD meshing tool Ansys Tgrid®. Actually, the surface mesh created within Hypermesh® can be saved the Fluent format (.cas) which can be easily read by Tgrid®. Tgrid® is in fact the standard volume mesher when Fluent®. In Tgrid® environment, the volume between the intake system surface and the wind-tunnel walls was filled with tetrahedral elements, while the boundary layer region was modelled using prismatic cells. In Table 7.3, the Tgrid® settings selected to generate the internal volume mesh are presented. Seven prismatic layers were built over the intake system in order to ensure that the simulated boundary layer was entirely contained within the prismatic layers over the whole fuselage, with the exception of regions of separated flow. Moreover, a height of 1 [mm] was given to the first prismatic layer in order to meet the y+ requirements for the wall function calculation during the CFD simulation [7]. The *Aspect Ratio*[17] B.L. Offset method was used, since it allows to control the aspect ratio of the prism cells that are extruded from the base boundary zone. This was a crucial point in the Volume mesh generation, since there is a change of element size starting from the intake system and proceeding towards the wind tunnel walls. Therefore, this method allowed to keep small the gap between the last prismatic cell size and that of the first tetrahedral element, in all of the model regions. This is clearly visible in Figure 7.10. Tgrid® journal file for volume mesh generation, containing all the

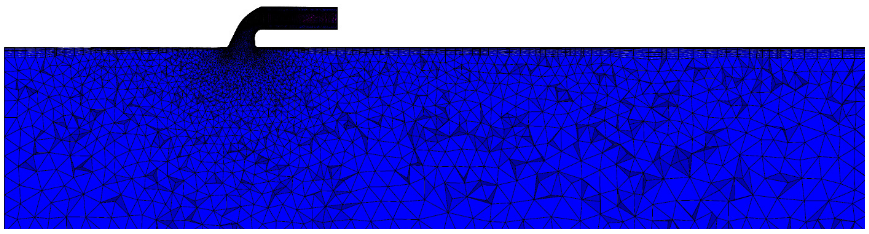


Figure 7.9: Longitudinal view of the whole volumetric mesh on a center line plane.

settings and the coordinates of refinement regions, can be found in Appendix C.4. A longitudinal view of the whole volume mesh is depicted in Figure 7.9, while Figure 7.10 illustrates a close-up of the mesh near the intake system surface. Finally, a particular of the boundary layer near the compressor face is reported in Figure 7.11.

The Tri/Tet Growth rate parameter was set to 1.1, in order to obtain a fine and uniform mesh in the regions of interest.

7.6 Fluid-dynamic model set up

A pressure-based solver type with absolute velocity formulation and steady approach was adopted for the tiltrotor simulations. The $\kappa - \omega$ SST model was

[17]The *aspect ratio* is defined as the ratio of the prism base length to the prism layer height.

Components with boundary layer	Air Intake, Dummy Duct, Dummy Flat Plate
Components without boundary layer	Inlet, Outlet, Symmetry, AIP, Compressor face
B.L. Offset method	Aspect ratio (10)
B.L. Growth method	Geometric
B.L. Number of layer	5
B.L. First height	1
B.L. Growth rate	1.2
Tri/Tet Improve surface mesh option	Enabled
Tri/Tet Refinment method	Adv/front
Tri/Tet Cell size function	Geometric
Tri/Tet Growth rate	1.1
Tri/Tet Refinement regions	No

Table 7.3: Ansys Tgrid® settings for volume mesh generation.

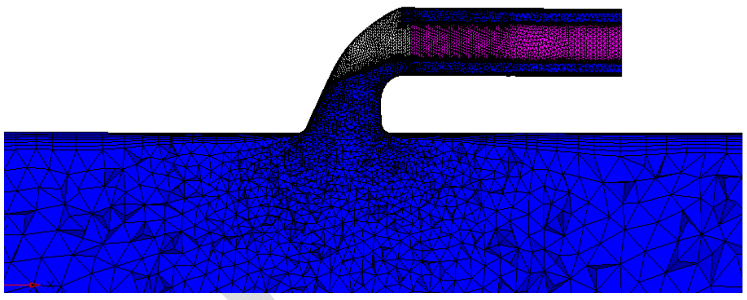


Figure 7.10: Close-up of the volume mesh near the intake system.

selected for turbulence treatment.

The air was treated as an ideal gas having constant specific heats, which automatically enables the equation energy resolution. This makes it possible to include the compressibility effects in the numerical simulations: in fact, as it can be deduced from Table 7.1, the forward-flight condition selected for the optimization features a moderately high Mach number. In Table 7.4, the air properties assigned for the intake nose optimization are summarized.

As far as the solution algorithm is concerned, a COUPLED scheme was adopted, which solves the pressure and moment equations simultaneously. The *Pseudo Transient* option was activated, which is a form of implicit under-relaxation method introduced in the last Fluent® release (please read the Theory Guide [7]). This option allowed to decrease enormously the number of required iterations, without increasing the time and computational effort needed per each iteration. Hence, the total computational time of each simulation was substantially reduced.

A Second Order Upwind discretization scheme was considered sufficient for this work purposes. It was selected for all the variables, since it guarantees a high accuracy of the numerical solution, due to its potential to improve sufficiently spatial accuracy by reducing numerical diffusion, particularly for complex three-

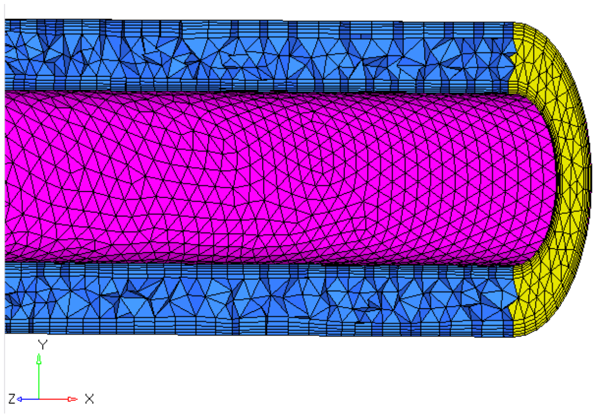


Figure 7.11: Close-up of the volume mesh: boundary layer near the compressor face.

dimensional flows, while not negatively affecting the total time requested for simulations when compared to the third order upwind scheme.

In particular, for the gradient spatial discretization, the Green-Gauss node based discretization scheme was chosen instead of the more common Green-Gauss cell based or Least squares cell based schemes, because it is more suitable for unstructured tetrahedral mesh [4], as is the case for the intake system simulation. The selected solver settings and discretization schemes are summarized in Table 7.5.

Regarding the under-relaxation factors, they were left to their default values. As regards the baseline run, the solution was initialized by means of the Hybrid Initialization approach [7]. Hybrid initialization is a collection of recipes and boundary interpolation methods. It solves Laplace's equation to determine the velocity and pressure fields. All other variables, such as temperature, turbulence, species fractions, volume fractions, etc., will be automatically patched based on domain averaged values or a particular interpolation recipe.

As far as the morphed cases is concerned, an interpolation file was used to map data from the baseline mesh onto the deformed one. To do so, an interpolation file was created for both the flown conditions. These files contained all the needed fluid dynamic variables regarding the baseline CFD run. This strategy was followed in order to decrease further the computational time, since, reasonably, the morphed cases feature a fluid dynamic variables distribution not so different from the baseline one.

For each simulation, the convergence criterion was established when normalized RMS residuals were less than $1 \cdot 10^{-6}$. Furthermore, the Mass weighted average of the total pressure on the AIP was monitored, in order to make sure it would reach stabilized values at the end of the simulation.

Gas thermo-dynamic model	Ideal Gas
Cp Specific Heat $[J/kg \cdot K]$	Constant: 1006.43
Thermal conductivity $[W/m \cdot K]$	Constant: 0.0242
Viscosity variation law	Sutherland: three coefficients methods
μ_0 *Reference viscosity* $[kg/m \cdot s]$	$1.716 \cdot e^{-5}$
T_0 *Reference temperature* $[K]$	273.11
S *Effective temperature* $[K]$	110.56

Table 7.4: Air properties for the intake optimization.

Solver type	Pressure-based steady-state
Pressure-velocity coupling scheme	COUPLED (Pseudo-transient option activated)
Spatial discretization schemes	
Gradient	Green-Gauss node based
Pressure	Second order
Density	Second-order Upwind
Momentum	Second-order Upwind
Turbulent kinetic energy	Second-order Upwind
Specific dissipation rate	Second-order Upwind
Energy	Second-order Upwind

Table 7.5: Solver settings and discretization schemes.

7.7 Boundary and operating conditions

The boundary conditions for the optimization study were selected according to the operating flight conditions reported in Table 7.1. A total pressure condition was imposed on the wind tunnel inlet, while a static pressure was assigned over the outlet section. Over the lateral walls of the wind tunnel, together with the bottom surface, a *symmetry* condition was imposed. All the other surfaces were treated as hydraulically smooth, adiabatic walls. On the compressor face patch, a pressure-outlet boundary condition was imposed, by specifying a target mass flow rate, according to the flown condition being simulated. The mass flow rate values are specified in Table 1. The operating reference condition was set to 0 [Pa] for both the investigated operating conditions. The pressure inlet boundary condition imposed on the inlet boundary zone requires the specification of both the gauge total pressure and the total temperature. These quantities were determined applying the ideal gas laws to the data of Table 7.1; in fact, from the static temperature (T_∞ in [K], corresponding to the OAT given in Table 7.1) and the velocity (V_∞, the True air speed) the operating Mach number (Ma_∞) was calculated as:

$$Ma_\infty = \frac{V_\infty}{\sqrt{kRT_\infty}} \tag{7.1}$$

where k is the specific heats ratio (1.4 for dry air) and R is the gas constant (287 $[J/(kg \cdot K)]$ for dry air). The resulting operating Mach number is 0.1826. The following equations were used to calculate the total pressure and total temperature at the inlet:

$$P_T = P_\infty \left(1 + \frac{k-1}{2} Ma_\infty^2\right)^{\frac{k-1}{k}} \tag{7.2}$$

$$T_T = T_\infty \left(1 + \frac{k-1}{2} Ma_\infty^2\right) \tag{7.3}$$

being P_∞ the static pressure given in Table 7.1.

Subtracting the value of the reference pressure to the total pressure calculated with Equation 7.2, we obtained a gauge total pressure of 96433.255 [Pa] for the forward flight, and 101325 for the hovering flight, while the resulting total temperature from Equation 7.3 was 286.086 $[K]$ and 288.15 $[K]$, respectively. Table 7.6 summarizes the settings for the pressure inlet boundary condition.

Gauge total pressure $[Pa]$	96433.255 (Forward Flight) 101325 (Hovering Flight)
Supersonic/initial gauge pressure $[Pa]$	94214 (Forward Flight) 101325 (Hovering Flight)
Direction specification method	normal to boundary
Turbulence specification method	Intensity and length scale
Turbulent intensity $[\%]$	1
Turbulent length scale $[m]$	0.5
Total temperature $[K]$	286.086 (Forward Flight) 288.15 (Hovering Flight)

Table 7.6: Pressure inlet specification at inlet.

The pressure outlet boundary condition requires the specification of the gauge static pressure and the back-flow total temperature at the outlet surfaces: a relative zero gauge pressure (corresponding to the undisturbed value) and a back-flow total temperature equal to the inlet static temperature were chosen. Table 7.7 summarizes the settings for the pressure outlet boundary condition on the outlet surfaces.

Gauge pressure $[Pa]$	94214 (Forward Flight) 101325 (Hovering Flight)
Direction specification method	normal to boundary
Turbulence specification method	Intensity and length scale
Turbulent intensity $[\%]$	1
Turbulent length scale $[m]$	0.5
Backflow Total temperature $[K]$	284.19 (Forward Flight) 288.15 (Hovering Flight)

Table 7.7: Pressure-outlet specification at outlet.

Regarding the turbulence specification method, a turbulence intensity of 1as the turbulent length scale is concerned, the final selected value 0.5 [m] was the result of an iterative calculation process. In fact, the turbulent length scale can be defined as $\sqrt{\kappa/\omega}$.

Hence, the values of κ and ω were iteratively computed at both inlet and outlet on a CFD model characterized by an initial turbulent length scale of 1 $[m]$. However, some tests were performed that allowed to verify that the tiltrotor nose drag was quite insensitive to variations of the turbulent length scale, at least for reasonable values of this parameter. As far as the compressor face boundary condition is concerned, a preliminary consideration has to be done. Intake flow field is strongly dependent on the mass flow rate imposed by the engine compressor at the AIP. The value of AIP mass flow rate is usually a function of both flight operating conditions (flight speed, altitude, external pressure and temperature etc.) and power demanded by the rotor. The compressor aspiration effects can be well modeled in Fluent® by imposing at the AIP a *pressure-outlet* boundary condition with target mass flow rate specification. Such a kind of boundary condition iteratively adjusts the static pressure on the outlet surface until the mass flow rate measured on the surface itself matches the specified value [7]. Unfortunately, the problem with this solution is the constant static pressure distribution which is imposed on the outlet surface by the boundary condition. This is not acceptable at the AIP, which is the location where inlet performance has to be measured. For this reason, an additional portion of the engine duct, behind the AIP, was modeled with the purpose of assigning the target mass flow boundary condition on a surface downstream the AIP. This additional dummy duct is clearly visible in Figure 7.6. Of course, the flow field characteristic over this additional portion of the engine duct will be not considered since the geometry results from a series of assumptions. However, it allows to unconstraint the AIP static pressure distribution, resulting in a more suitable CFD model for the intake system flow field. The location of the compressor face was established by means of a sensitivity analysis presented in section 7.8. On the compressor face, the pressure-outlet boundary condition features the following settings.

Gauge pressure $[Pa]$	94214 (Forward Flight) 101325 (Hovering Flight)
Direction specification method	normal to boundary
Turbulence specification method	Intensity and length scale
Turbulent intensity $[\%]$	1
Hydraulic diameter $[m]$	0.5
Backflow Total temperature $[K]$	284.19 (Forward Flight) 288.15 (Hovering Flight)
Target mass flow rate $[kg/s]$	4.7219 (Forward Flight) 5.6472 (Hovering Flight)

Table 7.8: Pressure-outlet specification at compressor face.

7.8 Grid sensitivity analysis

In this section, a preliminary sensitivity analysis is described, which was carried out in order to identify a suitable CFD model to be used during the intake trial optimization. Actually, the final selected CFD model must be a trade-off between numerical accuracy and required computational time and resources. As stated before, the final selected computational mesh was rather coarse: in fact, the scope of the present work was to become familiar with the problem of the AW101 air intake optimization and to demonstrate the capability of the proposed optimization methodology to successfully handle such a problem with its specific objective functions and constraints.

To this purpose, the following strategy was followed:

1. First, the dummy duct length was varied in order to point out the influence of the duct length on the pressure losses and the $DC(60)$ parameter in both the flight conditions considered.The sensitivity analysis to the dummy duct length was carried out using a coarse computational mesh, whose characteristics are discussed later on.

2. Once the proper duct length area was selected, the influence of the grid refinement was highlighted.

As far as the dummy duct length is concerned, three different values were analyzed (namely 500 [mm], 1,200 [mm] and 2,000 [mm]) while keeping all the other dimensions of the fluid domain fixed to their original values, and the effects on the intake performance were registered, both in terms of total pressure losses and maximum $DC(60)$, for both forward flight and hover conditions.

The results of the sensitivity study are summarized in Table 7.9.

Dummy duct length [mm]	Total pressure losses. Forward Flight $[Pa]$	Total pressure losses. Hovering Flight $[Pa]$	$DC(60)$ maximum value. Forward flight	$DC(60)$ maximum value. Hovering flight
500	831	200	0.461	0.027
1200	801	218	0.427	0.041
2000	802	226	0.403	0.044

Table 7.9: Total pressure on AIP sensitivity to dummy duct length.

As apparent, the total pressure losses in forward flight condition seem to reach an asymptotic value with increasing length of the virtual duct, and in particular they appear already stabilized for a dummy duct length equal to 1,200 [mm]. On the other hand, regarding the hovering conditions, the total pressure losses tend to increase with increasing length of the dummy duct, even though the maximum variation is within 10% of the baseline. Moreover, the $DC(60)$ maximum value tends to decrease as the duct length increases in forward flight conditions, while the opposite occurs in hover conditions. However, excursions of $DC(60)$ are considered within acceptable values, at least for the duct lengths of 1200 [mm] and 2000 [mm], for both forward flight and hover conditions. For the sake of completeness, in Figure 7.12 and Figure 7.13 a graphical representation of the AIP $DC(60)$ sensitivity to dummy duct length is illustrated for the forward flight and hover conditions respectively.

As apparent, the predicted $DC(60)$ distributions at the AIP using a dummy duct length equal to 1200 [mm] and 2000 [mm] respectively (corresponding to red and black dotted lines in the figures), are very close to each other in both the considered flight conditions. For this reason, the second model was adopted since

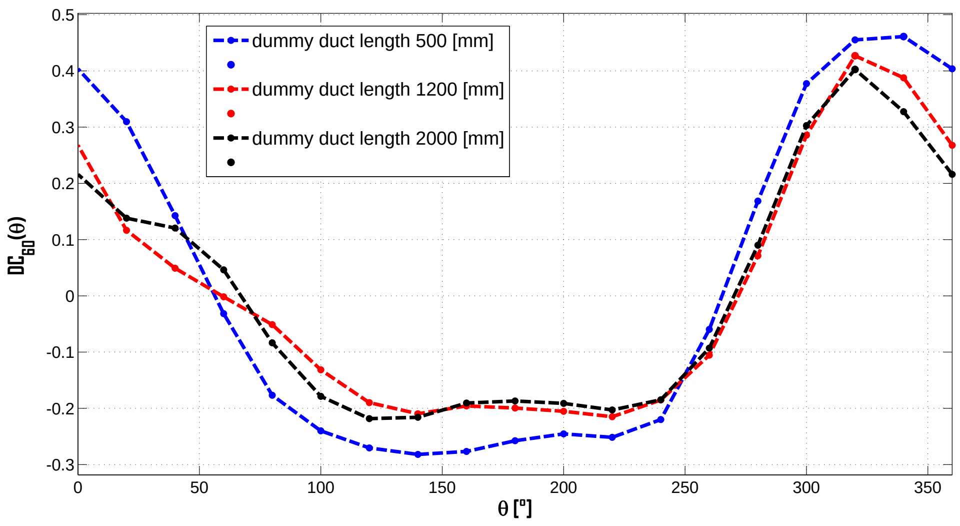


Figure 7.12: AIP $DC(60)$ sensitivity to dummy duct length. Forward Flight condition

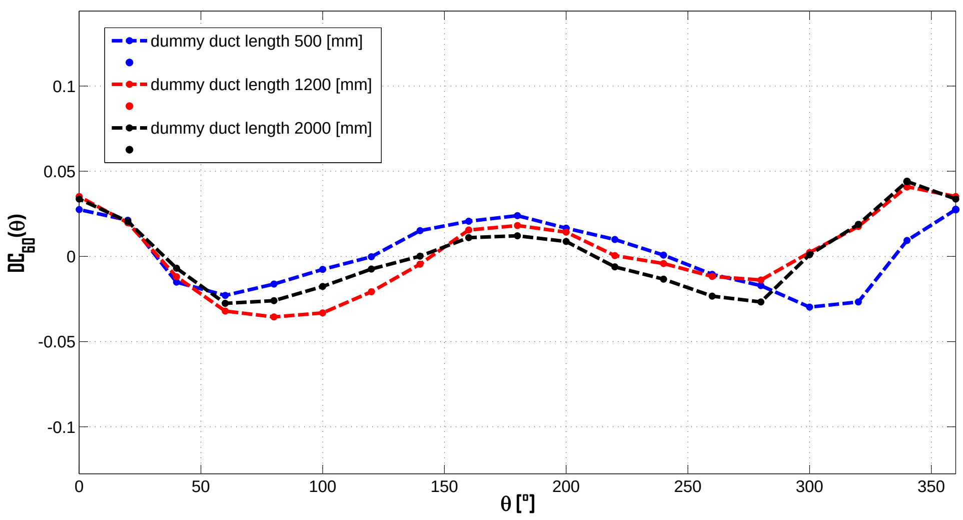


Figure 7.13: AIP $DC(60)$ sensitivity to dummy duct length. Hovering Flight condition

it appeared a good compromise solution for limiting the total amount of mesh elements.

In a second step, a grid refinement was carried out on the selected geometrical configuration, with the final aim of increasing the robustness of the CFD model for optimization purposes.

Actually, the objective here was not to implement a highly accurate numerical model fully validated against experimental data, but rather to set up a sufficiently reliable and robust computational grid, suitable for use in the optimization runs. For this reason, the final selected mesh was rather coarse.

A comparison of the characteristic features of the two analyzed grids is reported in in Table 7.10.

	Coarse Model	Refined model
Mesh type	Triangular	Triangular
Number of elements	29941	37897
Element size within the Air Intake [mm]	$16 \div 50$	15
Element size within the Dummy duct [mm]	$20 \div 200$	$15 \div 50$
Element size within the Dummy Flate plate [mm]	$50 \div 400$	$10 \div 100$
Element size within the inlet, outlet and symmetry planes [mm]	$150 \div 400$	$100 \div 400$
Element size within the AIP [mm]	$16 \div 50$	$11 \div 11$
Maximum 2D skeweness angle [°]	67	52
Maximum 2D aspect ratio	4.59	2.65

Table 7.10: Superficial mesh features of the coarse and refined model.

Moreover, the effects of the grid refinement on the air intake performance, both in terms of total pressure losses and maximum $DC(60)$, are summarized in Table 7.11.

Mesh refinement level	Total pressure losses. Forward Flight $[Pa]$	Total pressure losses. Hovering Flight $[Pa]$	$DC(60)$ maximum value. Forward flight	$DC(60)$ maximum value. Hovering flight
Coarse	801	218	0.43	0.04
Fine	794	207	0.45	0.002

Table 7.11: Total pressure on AIP sensitivity to mesh refinement.

Apparently, as far as the total pressure losses are concerned, the percentage variation was negligible in forward flight conditions (<1%), whereas it became larger in hovering flight ($\approx$ 5%). Also regarding the $DC(60)$ maximum values, a remarkable percentage variation was evidenced in hover conditions, while the forward flight value was less sensitive to the grid refinement level.

On the basis of the sensitivity analysis results, the refined mesh was selected for the subsequent optimization runs: actually, no further refinement was investigated, since the model was considered accurate enough for the trial optimization purposes. Moreover, it was also more robust than the coarser one from a computational point of view, which is a crucial characteristic during the optimization phase.

7.8.1 Aerodynamic characterization of the intake baseline geometry

Once the final CFD model to be used for optimization was selected, the aerodynamic features of the baseline AW101 intake1 were analyzed. In fact, the weaknesses and the major sources of losses occurring in the baseline geometry could be identified: this is fundamental to find out a proper geometry parameterization for a significant performance improvement.

First of all, the y+ characteristics of the associated CFD solution were investigated. In Figure 7.14, the y+ distribution over the intake walls is illustrated, for the forward and hovering flight conditions respectively. The selected parameters for grid generation were not shown to guarantee that the nondimensional mesh thickness at the intake surface strictly fell within the discretization levels ($y = 30 \div 300$) suggested for the standard wall functions implemented in the conventional turbulence models to work properly, in particular over the intake region, where the fluid-dynamic quantities necessary for the optimization objective functions evaluation are calculated.

Nevertheless, in Fluent 13 [7] the *Automatic near-wall treatment* is used as the default in all models based on the ω-equation. This option automatically switches from wall-functions to a low-Reynolds near wall formulation, as the mesh is refined. Hence, the models are relatively insensitive (as clearly stated in [7]) to the local Y plus.

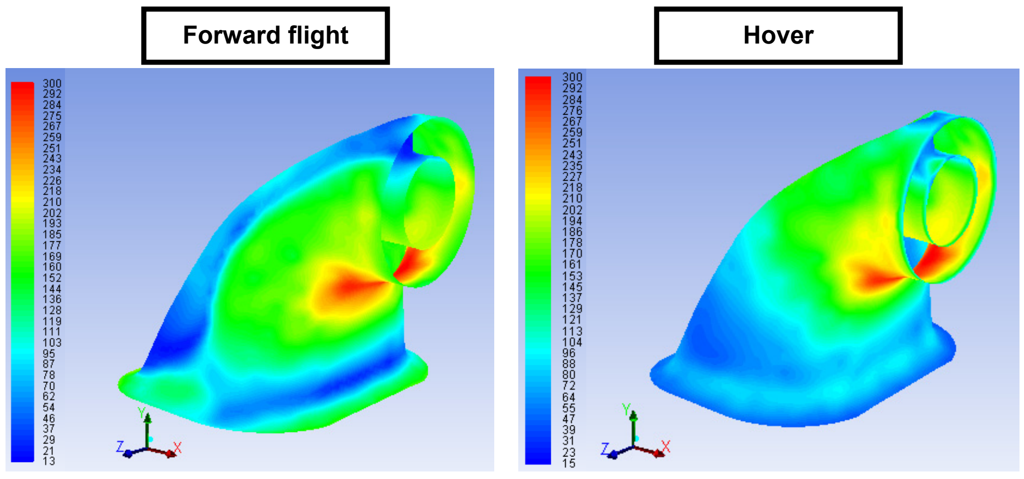


Figure 7.14: Contours of wall y+ over the Air Intake walls: forward flight (on the left) and hovering (on the right).

The total pressure contours over the air intake walls are illustrated in Figure 7.15 for both forward flight and hovering conditions.

The figure clearly highlights the presence of a concentrated total pressure loss located in the elbow of the air intake (in both the flight conditions). Actually, in this region the flow is accelerated, due to the highly curved shape of the duct; hence, a

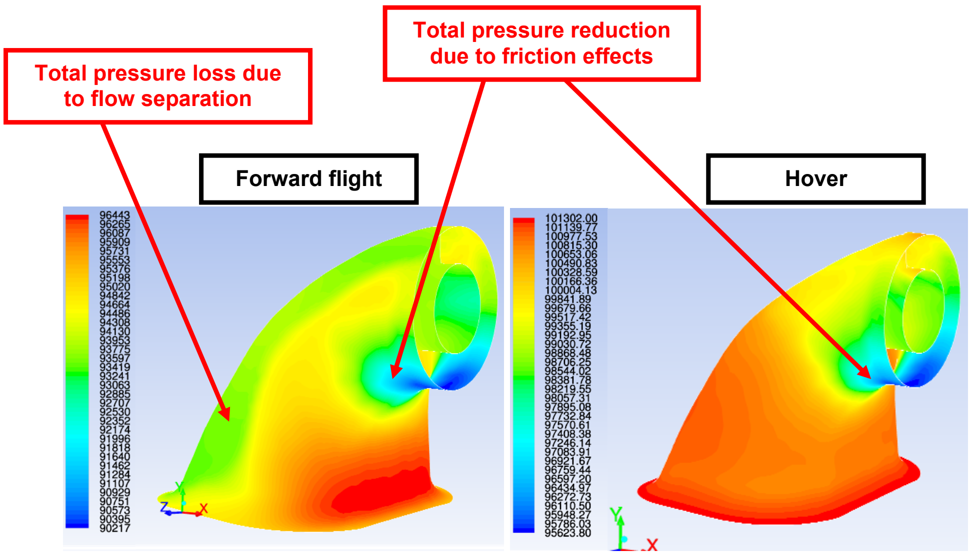


Figure 7.15: Contours of total pressure ([Pa]) over the air intake walls: forward flight (on the left) and hovering (on the right).

localized loss occurs which is mainly related to concentrated friction effects. In fact, this flow feature needs to be taken into account when identifying the geometry parameterization strategy: to this purpose, proper shape functions were selected aimed at reducing this localized total pressure loss, as will be illustrated in section 7.9.

Moreover, a total pressure drop is evidenced over the external surface of the S-shaped duct in forward flight conditions, due to a massive flow separation occurring in the entry region of the air intake, as can be visualized in Figure 7.16, where the streamlines pattern inside the air intake in both forward flight and hovering conditions are illustrated. In fact, the above mentioned separation results in a degradation of the intake performance in forward flight conditions. Moreover, a vortex flows appears inside the intake close to the AIP, due to the strong deflection the flow experiences around the engine shaft. Actually, this vortex flow is propagated towards the AIP, and it is expected to negatively affect the total pressure distribution at the engine face, as will be observed later on. As apparent from Figure 7.16, the forward flight condition is by far the most critical from the intake efficiency point of view: in fact, in hover conditions the flow field appears to be very regular and no separation occurs inside the intake duct.

The above mentioned flow characteristics have a direct impact on the total pressure distribution over the AIP, hence affecting the compressor inlet flow distortions. The total pressure contours over the AIP in both forward flight and hovering conditions are reported in Figure 7.17.

As apparent, in forward flight condition the vortex flow originating in the inner portion of the intake duct results in a non-symmetric distorted flow over the AIP. On the other hand, in hover conditions the wake originated from the sharp corner on the upper part of the air intake (visible in Figure 7.4) promotes a symmetrical

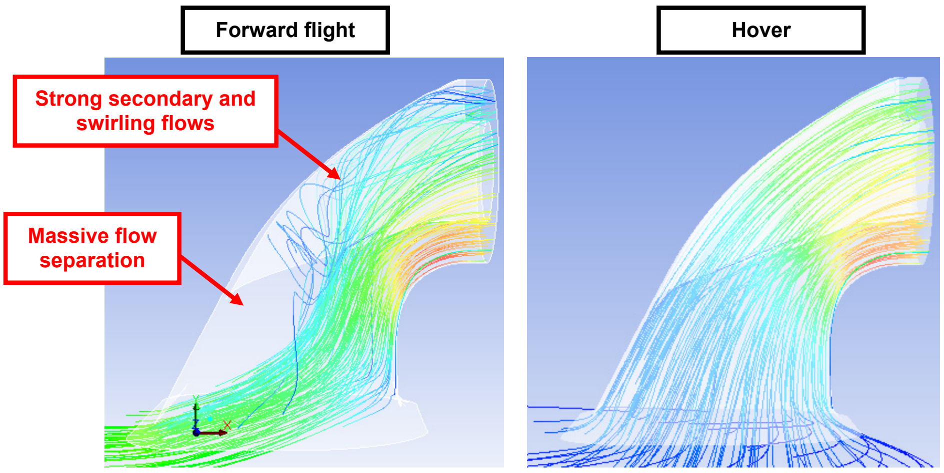


Figure 7.16: Streamlines pattern inside the air intake: forward flight (on the left) and hovering (on the right).

flow distortion in the upper portion of the AIP.

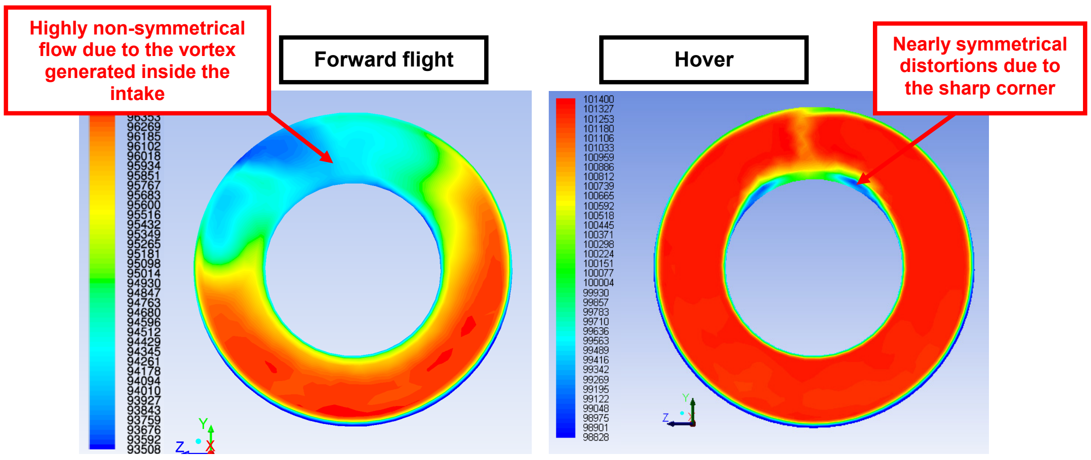


Figure 7.17: Total pressure distribution ([Pa]) over the AIP surface: forward flight (on the left) and hovering (on the right).

The circumferential distribution of $DC(60)$ over the AIP for the intake ♯1 baseline geometry is illustrated in Figure 7.18 and Figure 7.19 for forward flight and hover conditions respectively. Finally, the location of the $DC(60)$ worst sector for the two considered flight conditions is illustrated in Figure 7.20.

For the forward flight condition, the maximum $DC(60)$ has a value of 0.45, and the worst sector is located at 340 [deg]. On the other hand, in hover condition the maximum $DC(60)$ is 0.0205, and the worst sector is located at 360 [deg]. Moreover, a somehow more regular and "periodical" fluctuation of distortions along the circumferential direction is evidenced in hover (as can be appreciated in Figure 7.19), than that occurring in forward flight condition. The initial assessment of

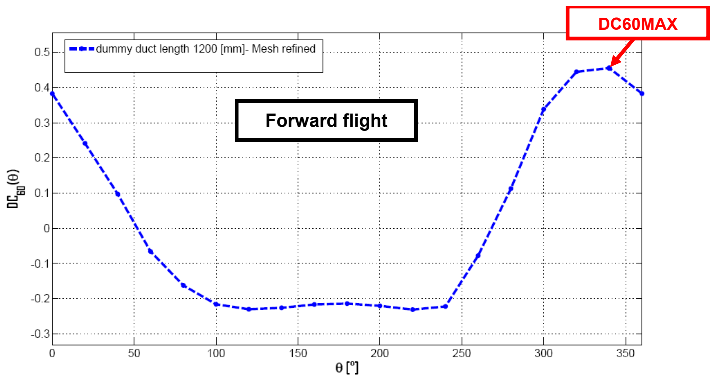


Figure 7.18: Circumferential distribution of $DC(60)$ for the intake ♯1 baseline model: forward flight.

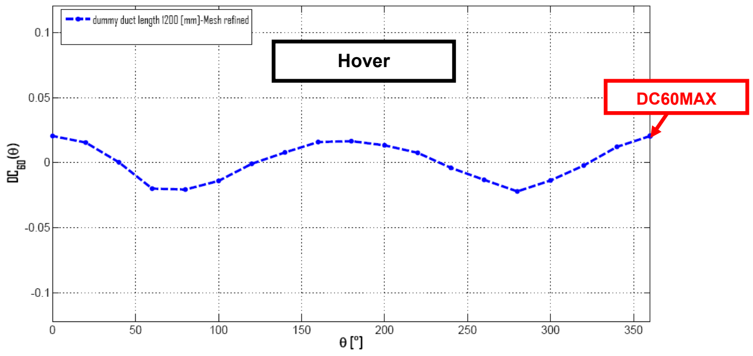


Figure 7.19: Circumferential distribution of $DC(60)$ for the intake ♯1 baseline model: hovering flight.

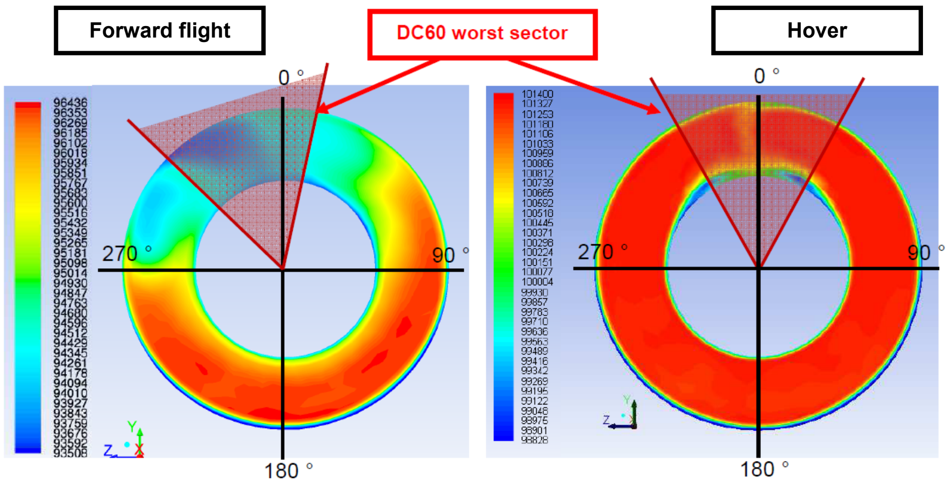


Figure 7.20: Location of the $DC(60)$ worst sector for the intake ♯1 baseline model: forward flight (on the left) and hover conditions (on the right).

the baseline intake aerodynamic features was essential for the identification of a proper parameterization strategy. Actually, the intake optimization is devoted to a reduction of the total pressure losses in both hover and forward flight conditions, and also the maximum flow distortions at the AIP must be kept within acceptable values as well. Hence, appropriate deformations will be carried out on the baseline geometry aimed at reducing vortex flows and local separations inside the intake.

7.9 Parameterization

Once the baseline CFD solution was analyzed, the geometry parameterization was carried out. This operation is of outstanding importance in the optimization process and it can be performed using various techniques. For the scope of the present work, the commercial software Altair HyperMesh® was adopted as the parameterization tool, due to the really versatile capabilities of the morphing tool HyperMorph. The choice of this software is mainly justified with its effectiveness and ease of use, which allows the user to build up complex parametric models in a relatively easy way by means of a dedicated graphical user interface. Moreover, the parameterization procedure in HyperMorph is very general: specifically, it is independent from the peculiar model (either FEM or CFD) which is the object of the optimization analysis. Therefore, whatever the geometry, the user is always allowed to use the same morphing strategies, resulting in a remarkable saving of the required working time for parameterization. Actually, both the above mentioned characteristics are fundamental in an industrial context, where time is always an essential issue. The geometry parameterization was carried out in a series of steps:

1. First of all, the baseline case file was imported into HyperMesh® in Fluent® format.

2. The desired shapes were generated, starting from the baseline geometry, through the morphing techniques available within HyperMorph®. For CFD studies, the more suitable strategy is the domains-handles approach for localized deformation, and the morph-volumes approach for global morphing requirements.

3. The generated shapes were then saved: with this operation, HyperMorph stores the current handles/nodes perturbations, allowing the user to apply them to the baseline model with any given scaling factor. Using this approach, the scaling factor of any generated shape can be dealt with as a design variable by an optimization algorithm.

4. The parameterized model was saved into an *.hm* file, which was then used for the batch-mode parameterization.

5. Finally, the HyperMesh® command file was created.

7.9.1 Morphing Constraints

The introduction of morphing constraints was mandatory to obtain a suitable optimized geometry for industrial applications. In fact, the requirement that the final intake design was both able to be manufactured and installed into the aircraft led to the definition of a series of geometrical constraints (essentially related to manufacturing and structural issues), that were considered in this preliminary optimization.

First of all, two directions were defined on the air intake surfaces, namely the "wetted" and "interior" directions: actually, the wetted direction is perpendicular to the side of the surface that is wetted by the intake airflow and the interior direction is the opposite, i.e. facing into the interior of the aircraft.

Referring to these two directions, the following geometric constraints for intake optimization were defined by AgustaWestland Ltd.:

1. All the edges where the geometry attaches to the aircraft surface (AIP edges, topdeck edges) were a fixed constraint and could not be moved.
2. All the faces shown in red in Figure 7.21 were not allowed to move in the interior direction, while in the wetted direction there was no constraint.
3. Finally, all the remaining faces, shown in green in Figure 7.21, were allowed to move up to 30$[mm]$ in the interior direction, while in the wetted direction there was no constraint.

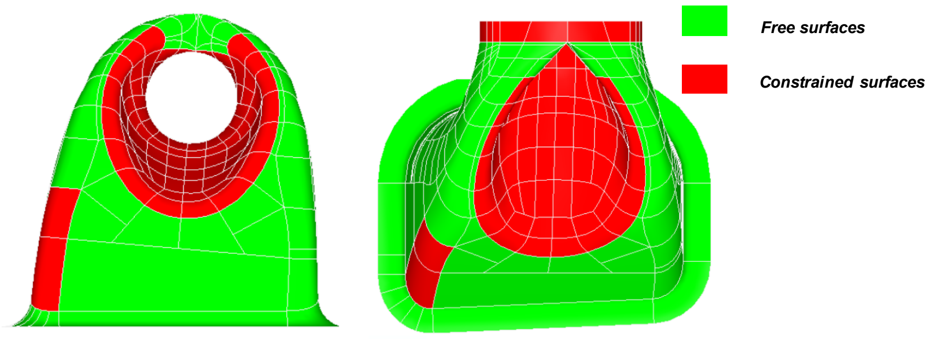


Figure 7.21: Geometrical constraints on the air intake surfaces: front view (on the left) and top view (on the right).

7.9.2 Design Variables Definition

In the present section, a brief description of the design parameters used for the AW101 intake1 trial optimization is provided. A total amount of nine design variables were considered. All the design variables were generated using the so called domains-handles approach in order to satisfy global morphing requirements. In particular, this approach allows for the application of mesh nodes displacements within a geometrical region (domain) by changing the location of specific, user defined, control points (handles) ([3]). Once applied, the nodes displacements may be saved as perturbation vectors and then they can be re-applied to the baseline model with any given scaling factor. Actually, the morphed geometry results from a linear combination of the user defined shapes multiplied by their own scaling factor:

$$v = \sum_{i=1}^{9} \alpha_i Sh_i \tag{7.4}$$

where:

- v is the global displacement vector;
- $Sh_i$is the i^{th} basic shape;
- α_i is the i^{th} shape scaling factor and it is actually generated by the optimization algorithm GeDEAII for each analyzed individual.

For the present application, the following ranges were defined for the values of α

- $\alpha_i \in [0,1], i = 3,4,5,6,8,9;$
- $\alpha_i \in [-1,1], i = 1,2,7;$

Using this approach, a scaling factor equal to zero means that the morphed geometry is identical to the baseline one, while a scaling factor equal to one produces the maximum allowed displacement within the specified range. Finally, scaling factors equal to minus one produce the maximum allowed displacement but in the opposite direction with respect to the original definition of the shape modifications.

In the following, the adopted design variables for intake optimization are described. In Figure 7.22, a series of cut planes used to define the shape functions are illustrated, while the x-z or x-y sections of the intake model parametric shapes, applied with a basic scaling factor equal to one, are reported from Figure 7.23 to Figure 7.31; the corresponding handle displacements are visualized as well.

The selected shape functions are described hereafter:

1. **sh1**: this shape consists in an initial deformation along the z-axis of the blue handle illustrated in Figure 7.23, which is located on the transversal cut plane B; in this figure, the baseline shape along with the deformed one are presented. The initial deformation range assigned to this variable was equal to $\pm$ 20 [mm];

2. **sh2**: this shape consists in an initial deformation along the z-axis of the blue handle illustrated in Figure 7.24, which lies on the transversal cut plane C. The initial deformation range was given a value of $\pm$ 20 [mm].

3. **sh3**: this shape consists in an initial deformation along the z-axis of the blue handle illustrated in Figure 7.62, located on the transversal cut plane B. It was given an initial deformation range of $\pm$ 20 [mm];

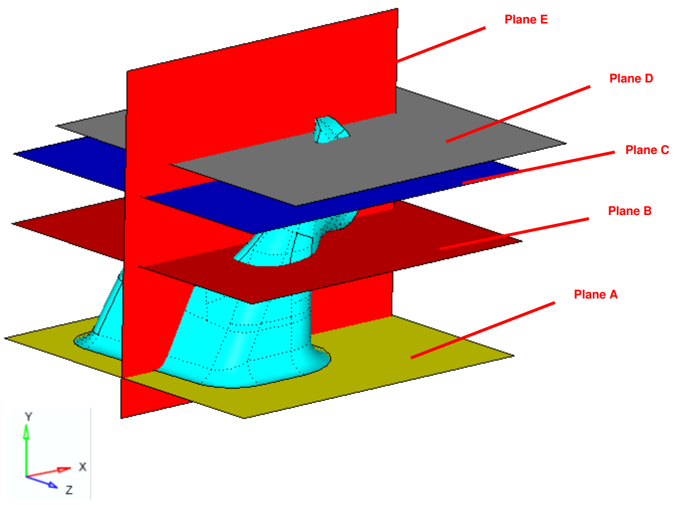


Figure 7.22: Cut planes used for the shape parameterization.

4. **sh4**:this shape consists in an initial deformation along the z-axis of the blue handle illustrated in Figure 7.63, which lies on the transversal cut plane C. It was given an initial deformation range of ± 20 [mm];

5. **sh5**:this shape consists in the deformation of the sharp corner located in the upper portion of the air intake, close to the AIP, as illustrated in Figure 7.64. It was given an initial deformation range of ± 20 [mm]; moreover, morph constraints allowed to keep unchanged the red surfaces illustrated in Figure 7.64.

6. **sh6**:his shape consists in an initial deformation along the y-axis of the blue handle located in the central region of the intake on the longitudinal cut section E, as illustrated in Figure 7.28. It was given an initial deformation range of ± 20 [mm] along the y direction;

7. **sh7**:this shape allows deforming the elbow region of the air intake both along the x and y-axes, as depicted in Figure 7.29. The initial deformation range assigned to this variable was equal to ± 20 [mm];

8. **sh8**:this shape consists in an initial deformation along both the x and y-axis of the blue handle located in the central region of the intake on the longitudinal cut section E, as illustrated in Figure 7.30. It was given an initial deformation range of +20 [mm] and -20 [mm] along the x and y direction respectively;

9. **sh9**:this shape consists in a clockwise rotation of the whole air intake, around the base point called "A" in Figure 7.31. The initially selected deformation range was equal to ± 10 [deg]. Only clockwise rotation was allowed, in order to comply with the geometrical constraints illustrated in Figure 7.21.

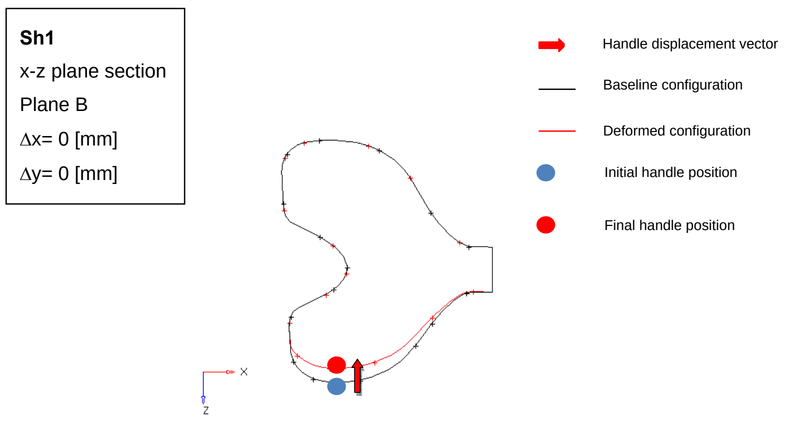


Figure 7.23: Parametric shape Sh1, applied to the intake model with scaling factor $\alpha = +1$.

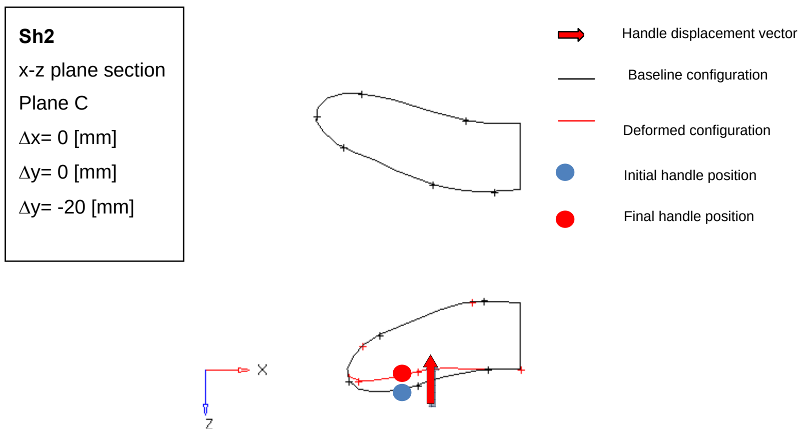


Figure 7.24: Parametric shape Sh2, applied to the intake model with scaling factor $\alpha = +1$.

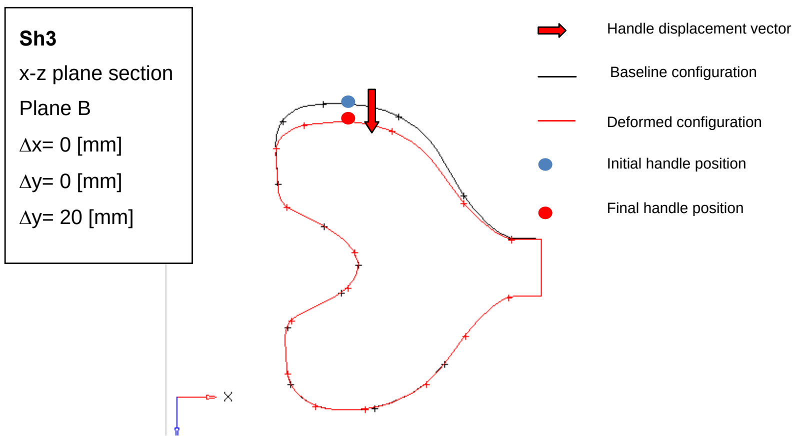


Figure 7.25: Parametric shape Sh3, applied to the intake model with scaling factor $\alpha = +1$.

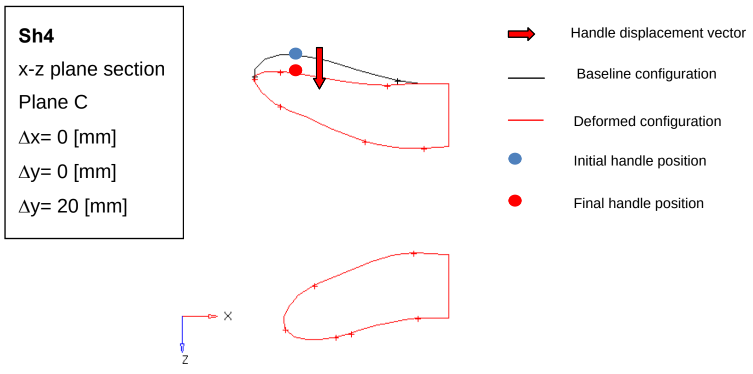


Figure 7.26: Parametric shape Sh4, applied to the intake model with scaling factor $\alpha = +1$.

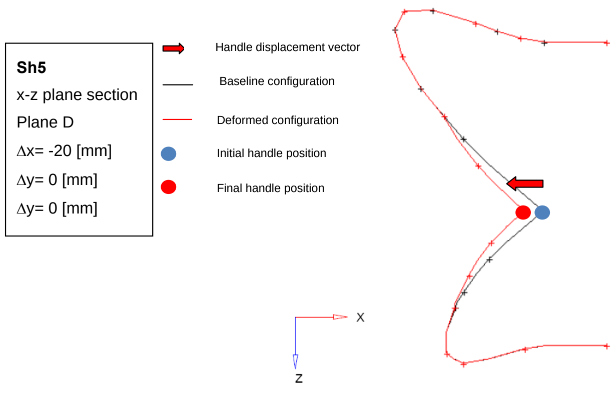


Figure 7.27: Parametric shape Sh5, applied to the intake model with scaling factor $\alpha = +1$.

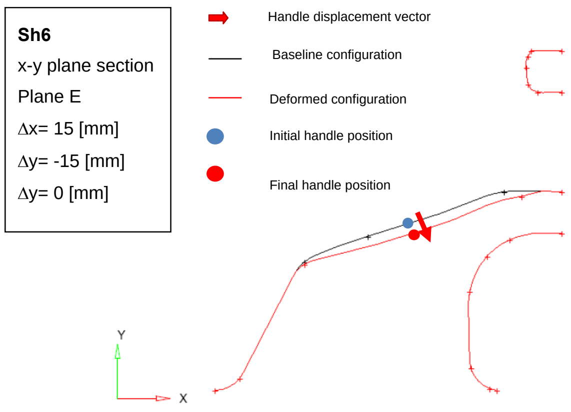


Figure 7.28: Parametric shape Sh6, applied to the intake model with scaling factor $\alpha = +1$.

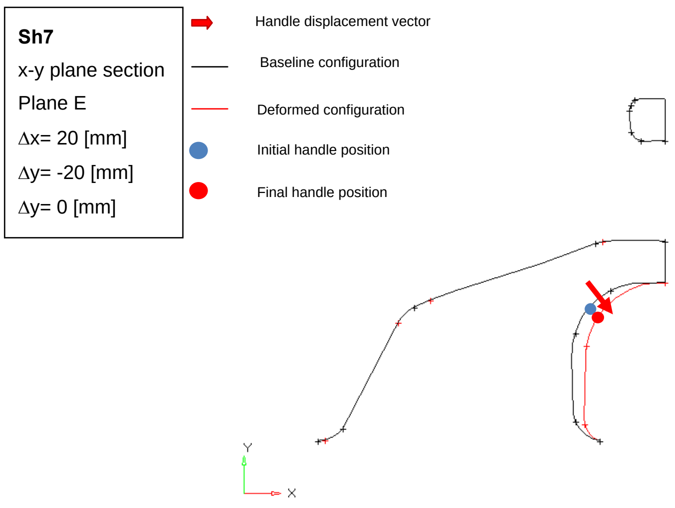


Figure 7.29: Parametric shape Sh7, applied to the intake model with scaling factor $\alpha = +1$.

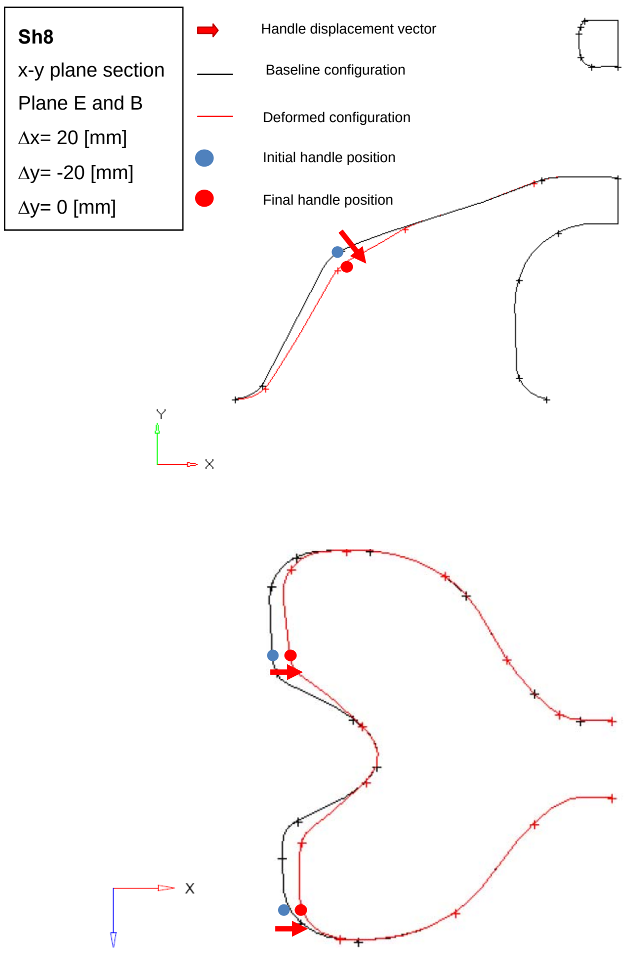


Figure 7.30: Parametric shape Sh8, applied to the intake model with scaling factor $\alpha = +1$.

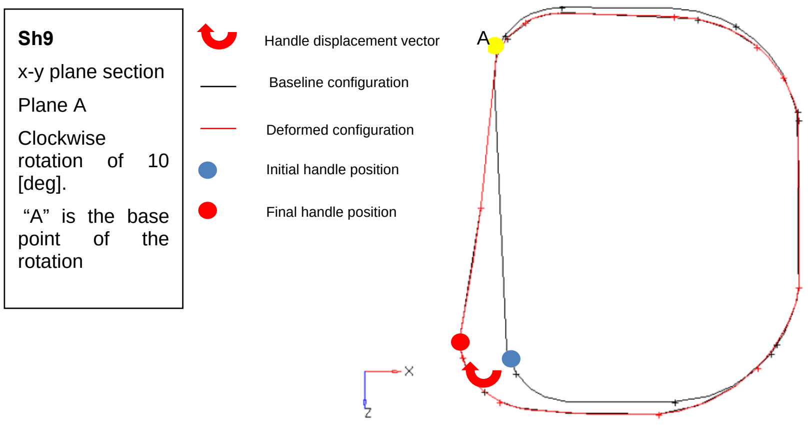


Figure 7.31: Parametric shape Sh9, applied to the intake model with scaling factor $\alpha = +1$.

7.10 Formulation of the intake optimization problem

Once the design CFD model and the parametric model for the intake geometry were built up, the successive step consisted in the GeDEAII-driven optimization. As stated above, the optimization problem considered was actually of the multi-point type, since the total pressure losses occurring in both forward flight and hover conditions needed to be minimized simultaneously. Moreover, the maximum $DC(60)$ at AIP was required to be kept to reasonable levels in both the considered flight conditions. From a mathematical point of view, the optimization problem can be expressed in the following way:

$$Minimize\,[F(\mathbf{x})] \tag{7.5}$$

where $[F(\mathbf{x})]$ is a vector function:

$$[F(\mathbf{x})] = \begin{cases} \Delta P_{tot}| \;@\; forwardflight \\ \Delta P_{tot}| \;@\; hoveringflight \end{cases} \tag{7.6}$$

and x is the design variable vector, consisting in the coefficients of the linear combination of the shape functions describing the morphed intake geometries (please read section 7.9):

$$x = [\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8, \alpha_19] \tag{7.7}$$

subject to the following variables bounds:

- $\alpha_i \in [0,1], i = 3,4,5,6,8,9;$
- $\alpha_i \in [-1,1], i = 1,2,7;$

Thanks to its multi-objective formulation, the optimization algorithm seeks for solutions featuring improved performance in terms of total pressure loss for both the flight conditions at hand. As mentioned in section 6.4.1, the mass-weighted average integral of the total pressure over the AIP was used in the present work for computation of total pressure losses. The requirement of keeping the maximum $DC(60)$ equal or lower than the baseline value in both the flight conditions was treated as a functional constraint, and a penalty function approach was used to handle this constraint. Specifically, some additional terms were added to the fitness functions expressed in Equation 7.6, which penalized the fitness values when a constraint was violated. In fact, a constraint violation results in an increase of the additional term which in turn increases the original value of the fitness function. As a consequence, during the selection mechanism the evolutionary algorithm tends to prefer individuals that do not violate the constraints, and to discard the penalized ones. In this way, it is likely that the latest generation will be constituted by individuals that do not violate these constraints, and whose penalty function values have consequently null value. The adopted penalty function for $DC(60)$ coefficient takes the following form:

$$PF = \begin{cases} 0 & if\ DC60_{configuration} \leq DC60_{baseline} \\ \beta \left(\frac{|DC60_{configuration} - DC60_{baseline}|}{DC60_{baseline}} \right)^{\alpha} & if\ DC60_{configuration} > DC60_{baseline} \end{cases} \tag{7.8}$$

where α is equal to 0.5, in order to give a convex form to the penalty function and force the constraints satisfaction; β was given a value equal to 700 and 100 for forward flight and hovering conditions respectively. These specific values were selected after a preliminary analysis aimed at finding a suitable shape for the penalty functions curves. The choice of the absolute value of total pressure losses rather than the total pressure ratio for the fitness function evaluation was actually related to the use of the above mentioned penalty function for control of the maximum flow distortion. In fact, being the values of the pressure recovery very close to 1, a suitable coupling with the typical values of the penalty function would have been more difficult, thus making the penalization for constraint violation ineffective. Therefore, the optimization problem can be finally reformulated as follows:

$$Minimize\,[PF + F(\mathbf{x})] \tag{7.9}$$

Moreover, the number of individuals per generation was set to 12, while the number of generations for the trial optimization run was set to 20.

7.11 Discussion of results

The trial intake optimization results are discussed in this section. First of all, in Figure 7.32, the overall set of simulated individuals is represented in the objective space. In particular, results refer to the modified fitness function specified in Equation 7.9, in which the penalty functions are taken into account as well. It is worth recalling that the numerical total pressure losses were determined using a mass-weighted average surface integral of the total pressure over the AIP surface.

Finally, the ultimate Pareto front after 20 generations is illustrated in Figure 7.33. Also in this case, the modified objective functions with addition of the penalty

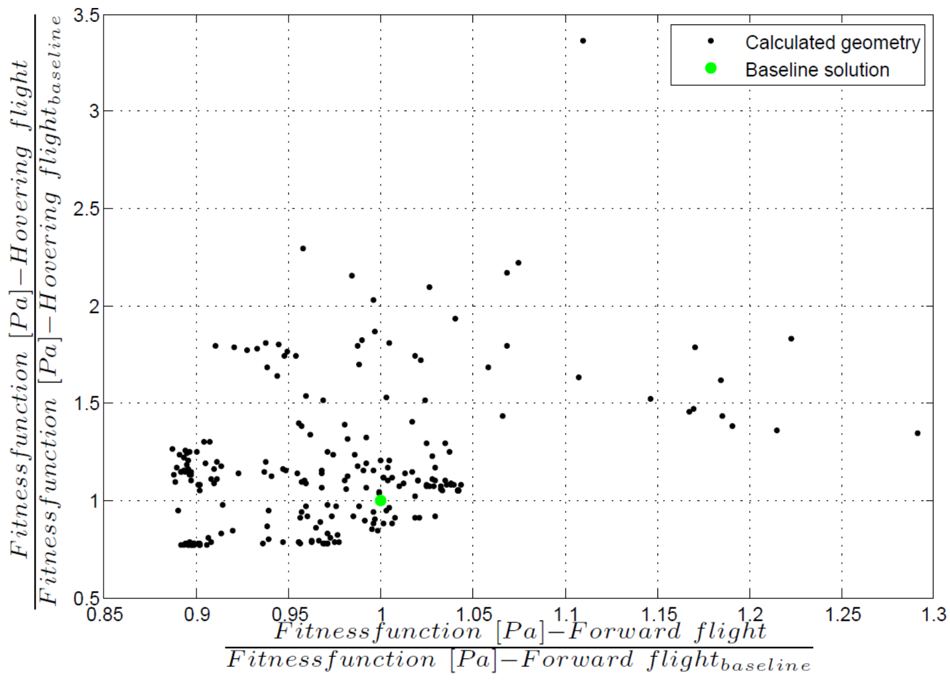


Figure 7.32: The entire set of geometries calculated during the optimization run.

functions are represented. In addition, the $DC(60)$ values for each individual on the Pareto front are reported for both forward flight and hover conditions. From the figure, the conflicting nature of the two selected objectives is clearly apparent. Furthermore, the front appears to be very sharp, in the sense that very small improvements of the total pressure losses in forward flight condition lead to remarkable degradations of the corresponding objective function in hover. In addition, neither the $DC(60)$ in forward flight nor in hover conditions show a monotonic trend with the values of the objective functions along the Pareto front.

The point on the Pareto front represented with a star in Figure 7.33 was selected as the final solution, since it was judged a satisfactory compromise between the two selected objectives. In Table 7.12, the design variables values of the selected solution are summarized (a null value indicates that the corresponding shape function was not modified with respect to the baseline). Moreover, a direct comparison of the optimized solution geometry with the baseline is illustrated in Figure 7.34. As

Parameter ♯	**Sh1**	**Sh2**	**Sh3**	**Sh4**	**Sh5**	**Sh6**	**Sh7**	**Sh8**	**Sh9**
Selected solution	−0.7	−0.373	0	-1	0	0	1	0.1686	1

Table 7.12: Trial intake optimization results: design parameters values for the selected optimized individual.

apparent from Figure 7.34, the optimized solution features an enlarged passage area at the transversal mid-section ($sh1$), which allows a local rearrangement of

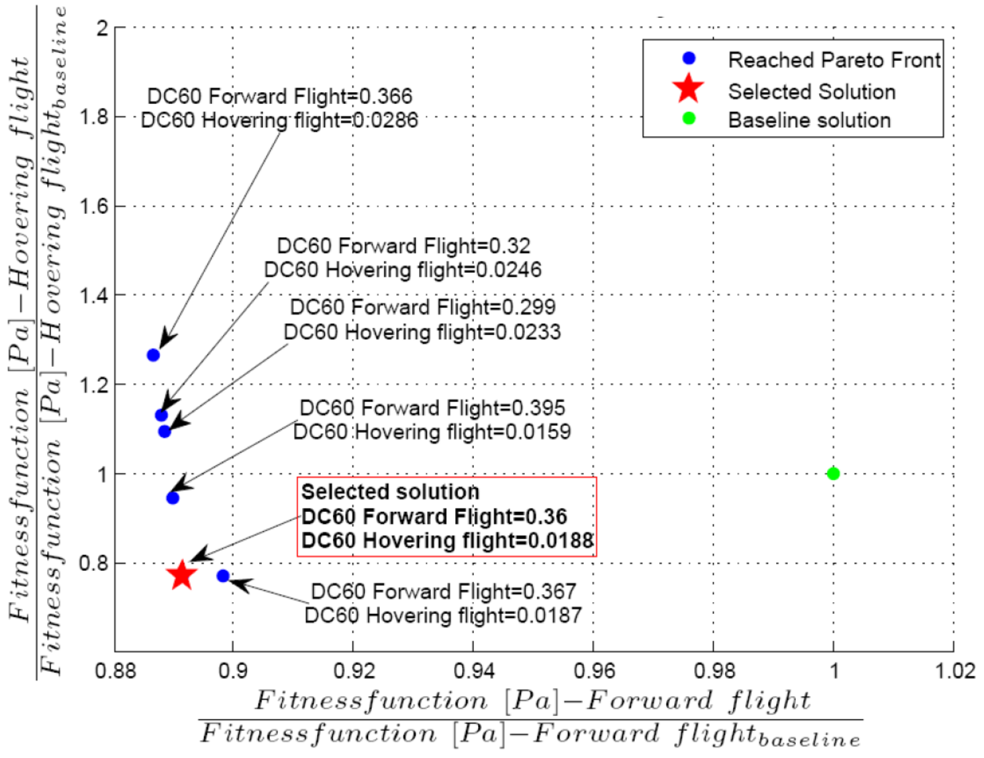


Figure 7.33: Final Pareto front after 20 generations.

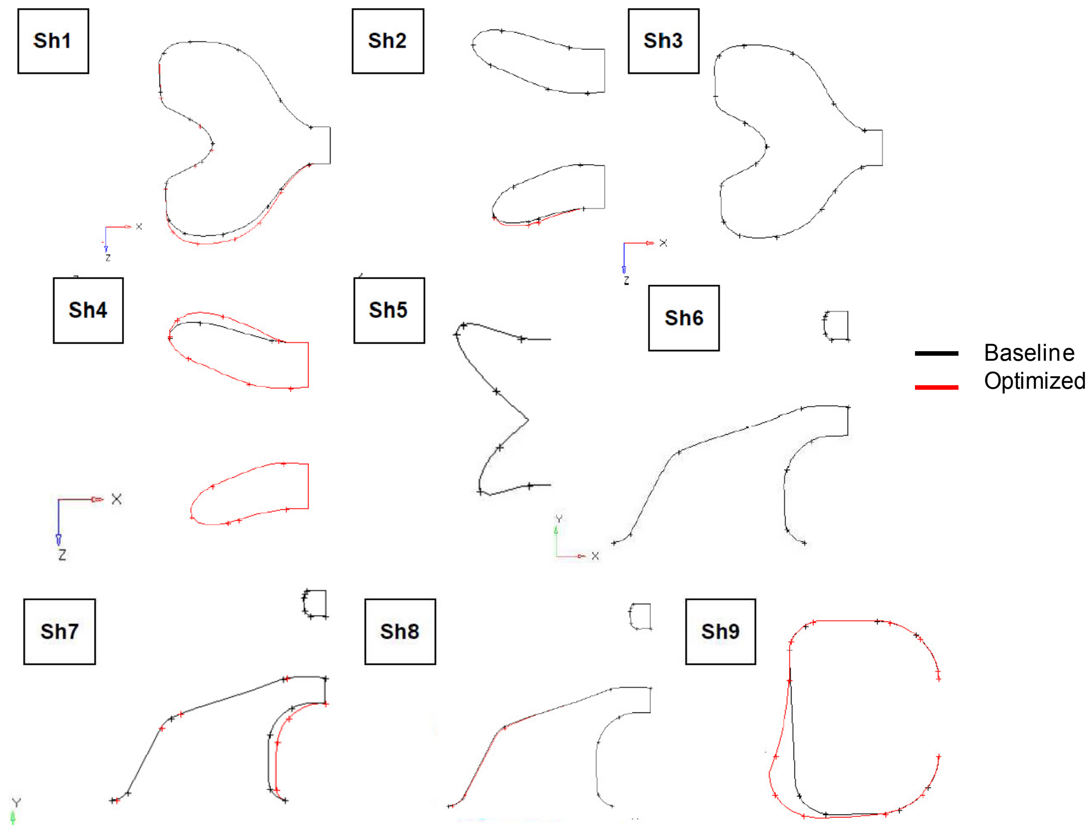


Figure 7.34: Optimized solution geometrical configuration: comparison with the baseline intake geometry.

the flow field, and therefore a reduction of the vortex flow, as will be illustrated later on. On the other hand, deformation of the *sh*4 shape acts in the direction of locally widen the intake duct section, and this has beneficial effects on both total pressure losses reduction and flow distortion requirements. Moreover, the optimized solution features a more smoothed elbow curvature (*sh*7) with respect to baseline. Finally, a remarkable rotation of the whole air intake around the point "A" depicted in Figure 7.34 is given to the optimized configuration: once again, this has a beneficial impact on the flow field inside the intake duct, since it reduces flow separation in the entry region occurring in forward flight condition. In Table 7.13, the values of the objective functions of the optimized solution are reported and compared with the baseline. As apparent, a remarkable reduction of the total pressure losses along the intake duct was obtained with the optimized solution, especially in hover conditions; in addition, regarding the flow distortion at the AIP, a large reduction of the $DC(60)$ at the engine face was achieved in forward flight and, to a lesser extent, in hover conditions, even though the flow distortion was not directly included in the objective function, but rather treated using a penalty function (please read section 7.32).

	Baseline configuration	Forward Flight	Hovering Flight
Total pressure drop reduction with respect to baseline [%]	\	10.85%	23.4%
DC60 reduction with respect to baseline [%]	\	19.6%	8.29%

Table 7.13: Trial intake optimization results: Total pressure drop and DC60 reduction with respect to the baseline geometry

In Figure 7.35 and Figure 7.36 the total pressure contours over a series of transversal sections along the optimized intake duct are depicted for both the hovering and forward flight conditions and they are compared with the baseline. As apparent, in hover conditions the region of low total pressure occurring in the final portion of the intake duct is less extended in the optimized geometry, and total pressure losses are less severe. Furthermore, also in forward flight conditions

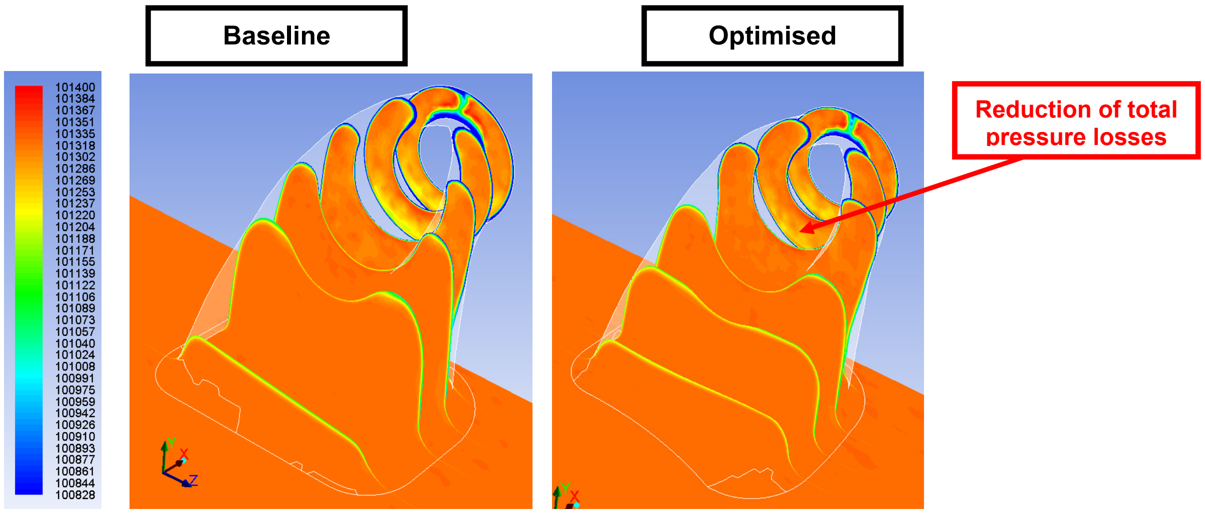


Figure 7.35: Contours of total pressure ([Pa]) over a series of transversal sections along the intake duct in hovering condition: comparison of the baseline (on the left) and optimized (on the right) solutions.

a reduction of the total pressure losses with respect to the baseline is evidenced, especially in the upper portion of the intake duct and towards the AIP. As expected, also the flow behavior over the AIP is improved in the optimized solution with respect to the baseline, for both hovering and forward flight conditions, as apparent from Figure 7.37 and Figure 7.38: in fact, the total pressure field is much more uniform over the engine face in both the considered flight conditions, and the most severe total pressure drops are eliminated in the optimized geometry. The more favorable behavior of the optimized intake duct is confirmed also by the visualization of the streamlines path, illustrated in Figure 7.39 and Figure 7.40 for hovering and forward flight conditions respectively. While no appreciable differences are evidenced in the streamlines behavior with respect to the baseline in hovering, a remarkable reduction of the vortex flow occurring in the second

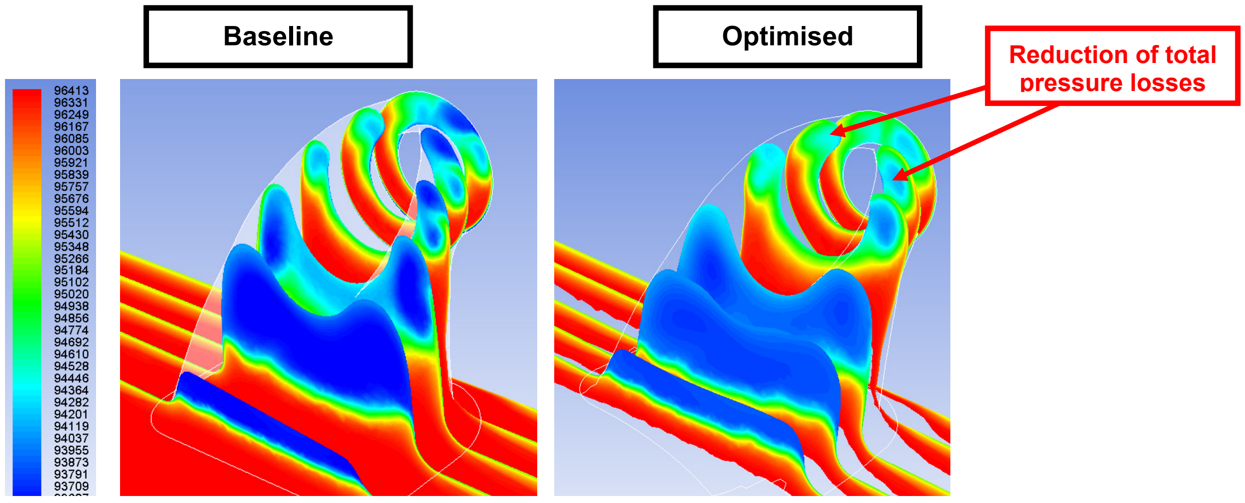


Figure 7.36: Contours of total pressure ([Pa]) over a series of transversal sections along the intake duct in forward flight condition: comparison of the baseline (on the left) and optimized (on the right) solutions.

bend of the S-shaped duct is achieved in forward flight conditions, due to the more streamlined shape of the optimized geometry with respect to the baseline. Actually, this beneficial effect is propagated to the AIP, as illustrated in Figure 7.38. Finally, the comparisons of the baseline and optimized DC60 profiles over the AIP for both hovering and forward flight conditions are reported in Figure 7.41 and Figure 7.42 respectively. While no appreciable modifications are shown in hover, a remarkable reduction of the maximum DC60 value is apparent in forward flight conditions with respect to the baseline (see Table 7.12): this is in fact very beneficial from the compressor performance point of view.

7.12 Reverse Engineering Procedure in CATIA® V5

From the optimization run the values of the design variables of the selected geometry were derived. Those values were introduced within Altair Hypermesh® in order to obtain the morphed mesh of the optimal geometry. Then, the mesh was exported in *stl* format: specifically, only the modified components were exported, being the dummy duct left unchanged.

The reverse engineering process results straightforward and it is qualitatively described in the following.

First, the file **.stl* created in Altair Hypermesh® was read within CATIA® V5 using of the Import function under the *Digitalize Shape Editor* module.

Second, the *Quick Surface Reconstruction* module was used in order reconstruct the surfaces from the *stl* model. In particular, a value of 1000 for *Surface Details* option was selected, and the *Auto Tangency* option was activated before launching the reconstruction process. This allowed to properly reconstruct the surfaces, and

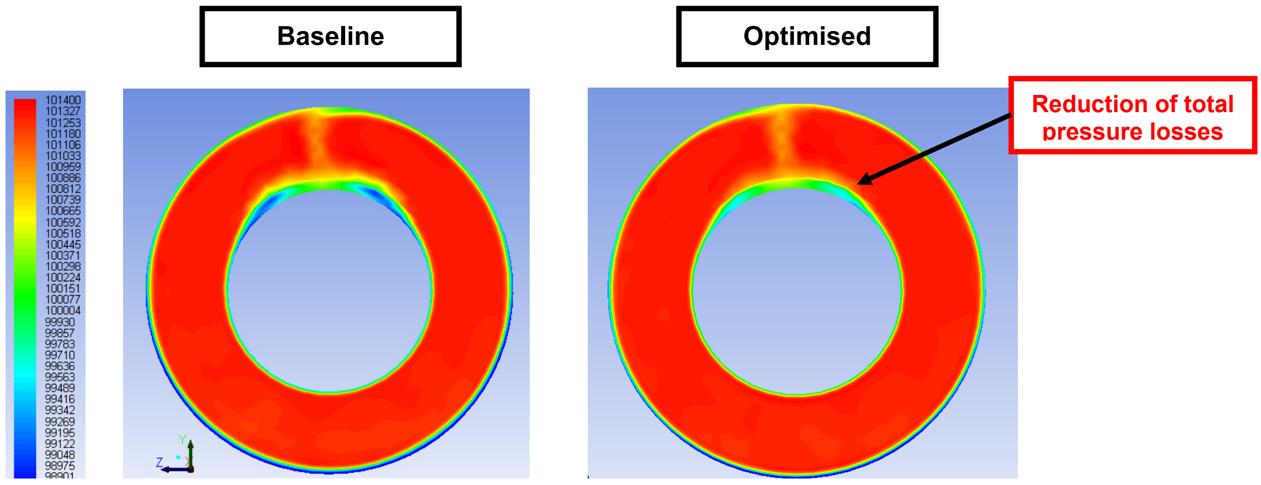


Figure 7.37: Contours of total pressure ([Pa]) over a series of transversal sections along the intake duct in hovering condition: comparison of the baseline (on the left) and optimized (on the right) solutions.

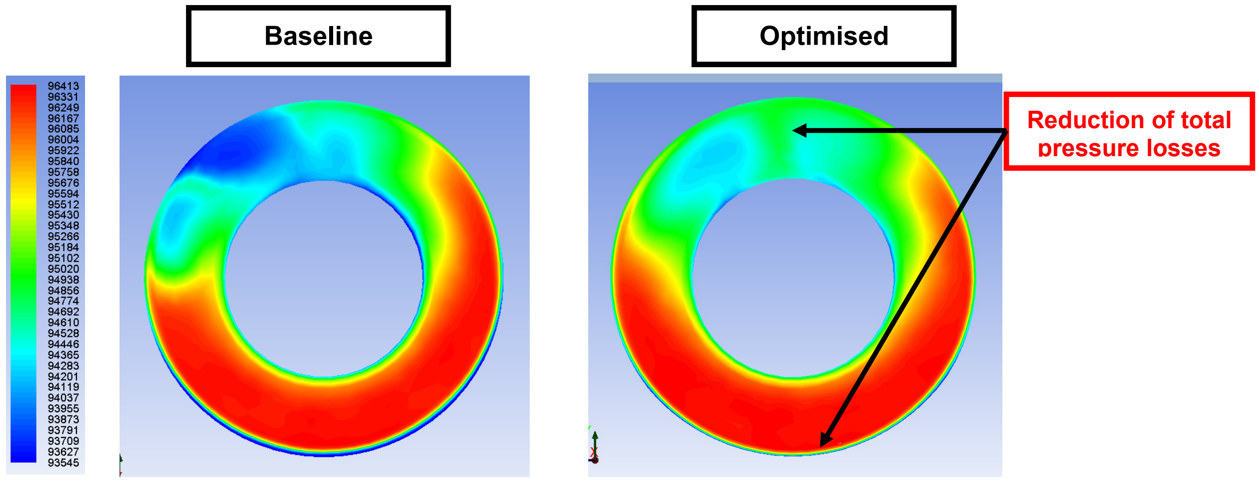


Figure 7.38: Contours of total pressure ([Pa]) over a series of transversal sections along the intake duct in forward flight conditions: comparison of the baseline (on the left) and optimized (on the right) solutions.

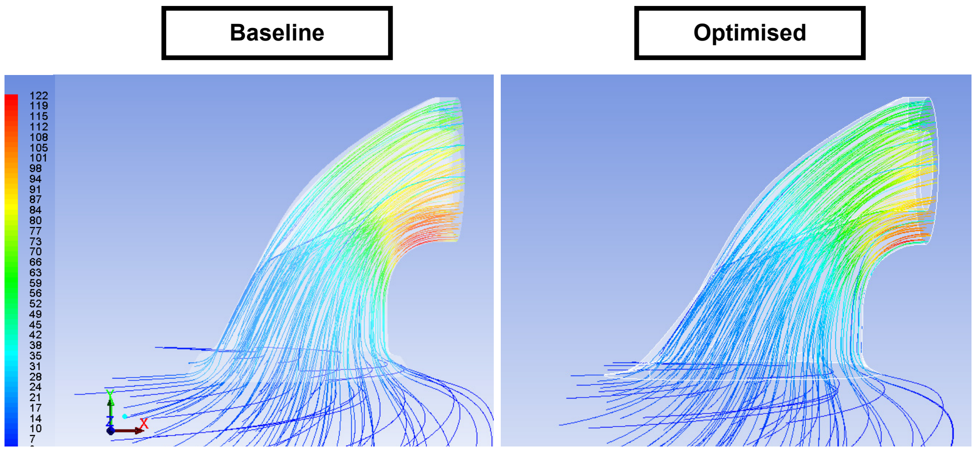


Figure 7.39: Streamlines pattern inside the air intake in hover: comparison of baseline (on the left) and optimized (on the right) configurations.

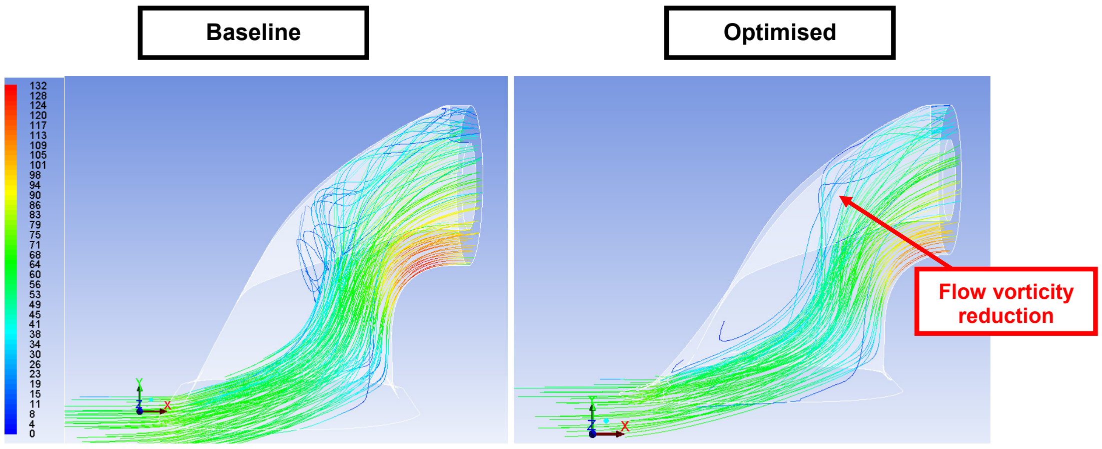


Figure 7.40: Streamlines pattern inside the air intake in forward flight: comparison of baseline (on the left) and optimized (on the right) configurations.

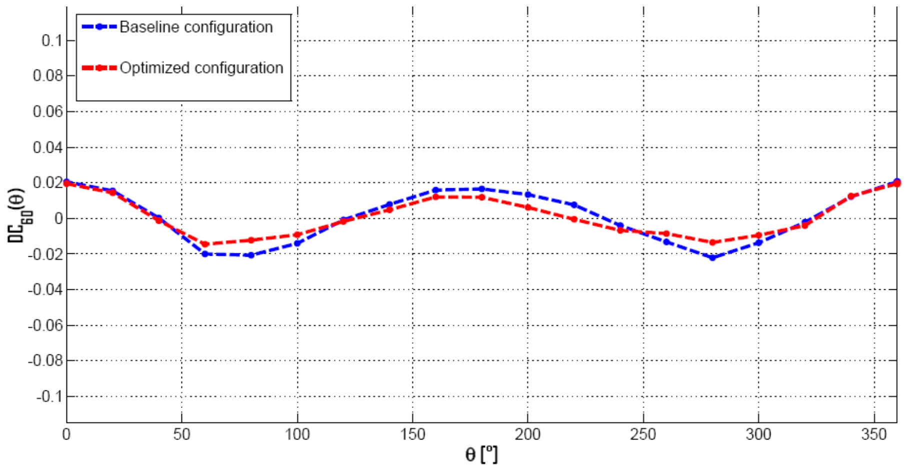


Figure 7.41: Comparison of baseline and optimized DC60 distribution in hovering.

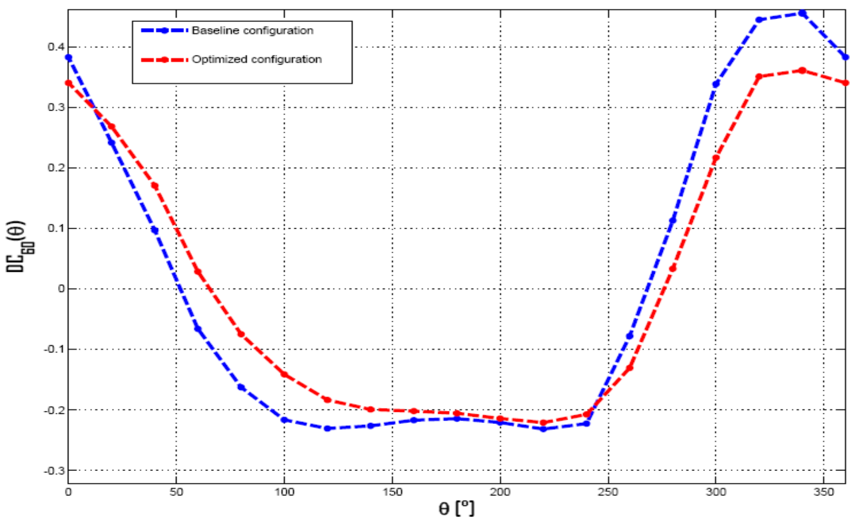


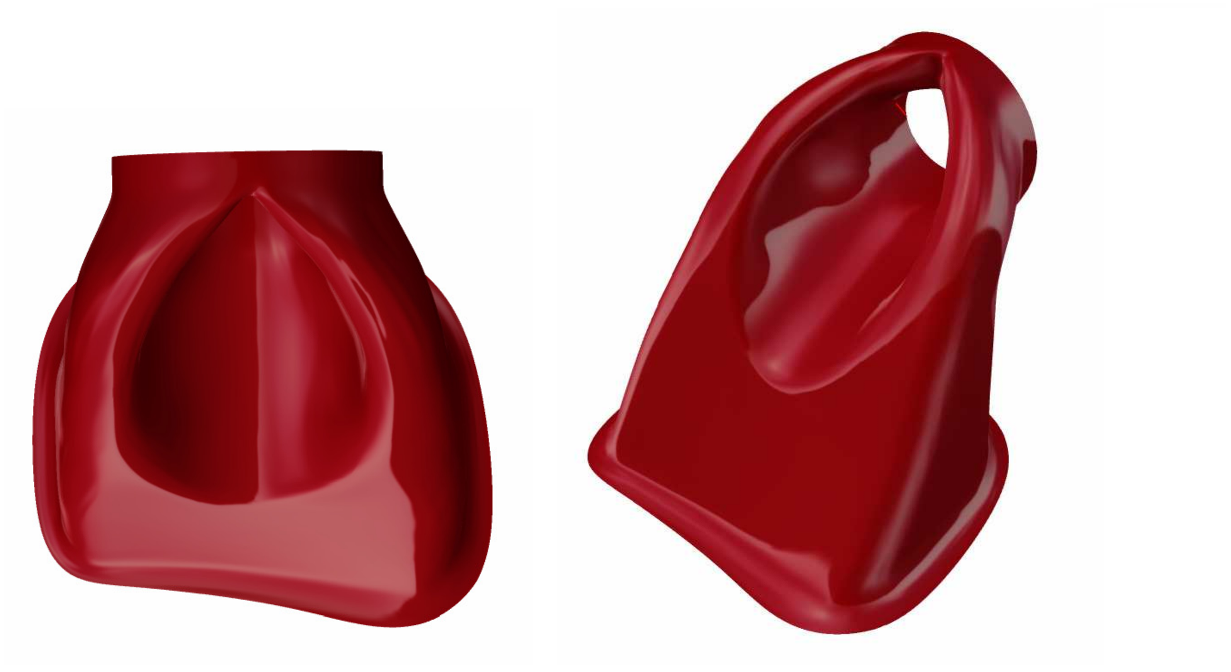
Figure 7.42: Comparison of baseline and optimized DC60 distribution in forward flight.

to capture all the needed details.

The upper and isometric views of the optimized geometry are depicted in Figure 7.43.

Once the optimized model has been reconstructed, in order to evaluate the geometrical differences between the optimized and the baseline configurations, a comparison was performed. It consisted in an overlap of the two CAD models.

The superimposition is depicted in Figure 7.44.



(a) Upper view (b) Isometric view

Figure 7.43: The final optimized intake geometry.

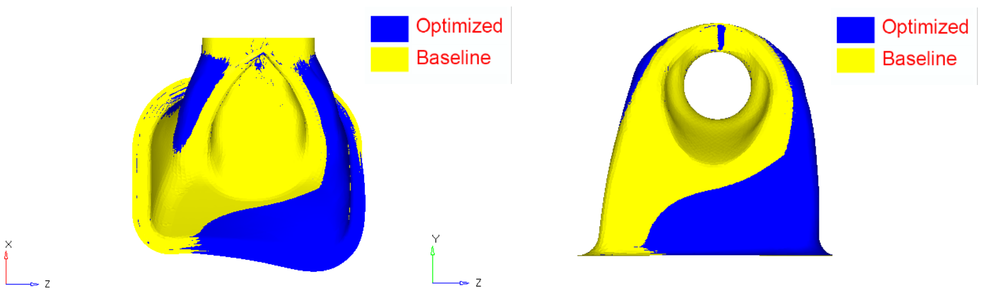


Figure 7.44: Superimposition of the baseline and the optimized intake CAD model.

Part II

Test case B:
Aerodynamic shape optimization of the frontal region of the ERICA tiltrotor using Ansys Fluent® as the CFD solver

7.13 Introduction

The present work deals with the ERICA tiltrotor nose shape optimization aimed at reducing its baseline aerodynamic drag. To this purpose, several steps were performed in order to obtain reliable results while reducing the overall optimization time as much as possible.

First, a study of drag sensitivity to the mesh quality was carried out (as described in chapter 7.19) in order to find out a model capable of reproducing drag characteristics of the tiltrotor nose with an acceptable accuracy while featuring the smallest number of 3D elements. As a matter of fact, decreasing the number of grid elements results in a reduction of calculation time, but, at the same time, the model capacity to properly reproduce the aerodynamic behaviour of this component is progressively deteriorated. Hence, the goal of this first step was to identify a suitable model resulting from a trade-off between these two aspects.

The second step consisted in a D.O.E. analysis over the selected CFD model, aimed at determining which factors were most influential on the response, that is, which design variables most influenced the nose drag characteristics. To this purpose, a Student analysis was carried out on the independent variables. The shape parameterization, the D.O.E. analysis and the Student test are illustrated and discussed in chapter 7.22.

Finally, the nose aerodynamic shape optimization was performed, as discussed in detail in chapter 7.23. As regards the optimization engine, the GeDEAII algorithm was selected, whose characteristics and performance were described during chapter 2.

7.14 Optimization Procedure

In this chapter, the whole sequence of operations necessary to carry out the optimization study is described. As already mentioned, the optimization procedure was not performed in a single step, but rather multiple steps were required to obtain reliable results, while reducing the overall optimization time as much as possible. Each operation will be fully analyzed in the next chapters.

The whole sequence of operations carried out to address a multi-objective optimization problem is shown in Figure 7.45. It is worth mentioning that each block was carried out independently from the others.

The upper portion of the flow chart shows the set of preliminary operations that need to be accomplished before starting the successive automatic loops. The procedure started from the elaboration of a baseline geometry CAD model of the tiltrotor fuselage provided by AgustaWestland in CATIA® format. The following steps focused on the construction of the baseline CFD model: first of all, the surface mesh was generated using CATIA® surface meshing tool, then the volume mesh was created by means of Ansys TGrid® and finally the CFD case was set up with the commercial CFD code Ansys Fluent® which is the standard code for CFD simulation adopted at AgustaWestland.

Next, a study on aerodynamic drag sensitivity to the grid features was carried out as described in detail in chapter 7.19. In fact, it is well known that cross-sectional area dimensions, virtual wind tunnel box length and mesh refinement level greatly influence the computed drag. A lot of combinations of these parameters were tested to obtain a model characterized by an acceptable accuracy while featuring the least possible number of cells.

Once this study was performed, the selected model was "frozen" and imported into Altair HyperMesh® environment for the geometrical parameterization.

A D.O.E. study was then carried out with the aim of exploring the design space, and a Student analysis was undertaken which made it possible to decrease the number of independent design variables; hence, the total time needed for the next GeDEAII driven optimization procedure could be reduced.

Once the independent variables to be retained were identified, the optimization process could be addressed, which was managed by GeDEAII as the master program.

Here some words are spent about the followed morphing strategy in this work. In details, it was decided to parameterize the surface mesh only, and to re-mesh every time the volume mesh, for the reasons explained below. Actually, the direct volume mesh morphing may cause boundary layer prisms distortions, leading to a deformed mesh featuring elements having negative Jacobian, even for small displacements. Instead, this choice allowed us to overcome this morphing limitation: it is now possible to morph only the surface mesh and then re-mesh the volume at each objective function/s evaluation, with the only drawback of a minimal increased overall optimization time.

Moreover, the surface mesh morphing, and then the volume re-meshing, allowed to increase the morphing capabilities and therefore to enlarge the optimization search space.

The lower part of the flow chart in Figure 7.45 describes the results extraction and elaboration. The output of the automatic optimization loop is the optimal combination of the design variables of the individuals lying on the Pareto frontier. At this point, the designer can identify the individuals of interest among the multiple optimal solutions of the Pareto front and the meshed geometry of each selected individual can be reconstructed using HyperMorph®.

The final step is the reverse engineering process, which allows obtaining the deformed surface starting from the mesh; this process can be performed using CATIA®.

7.15 The object of the optimization

The scope of this work consisted in the optimization of ERICA nose, whose $1/8th$ scaled wind-tunnel model is depicted in Figure 7.46. To this purpose, a CFD-based optimization procedure was implemented, which was introduced in chapter 7.14, and will be described in detail in the following chapters.

CFD simulations were carried out on the components depicted in Figure 7.47. In

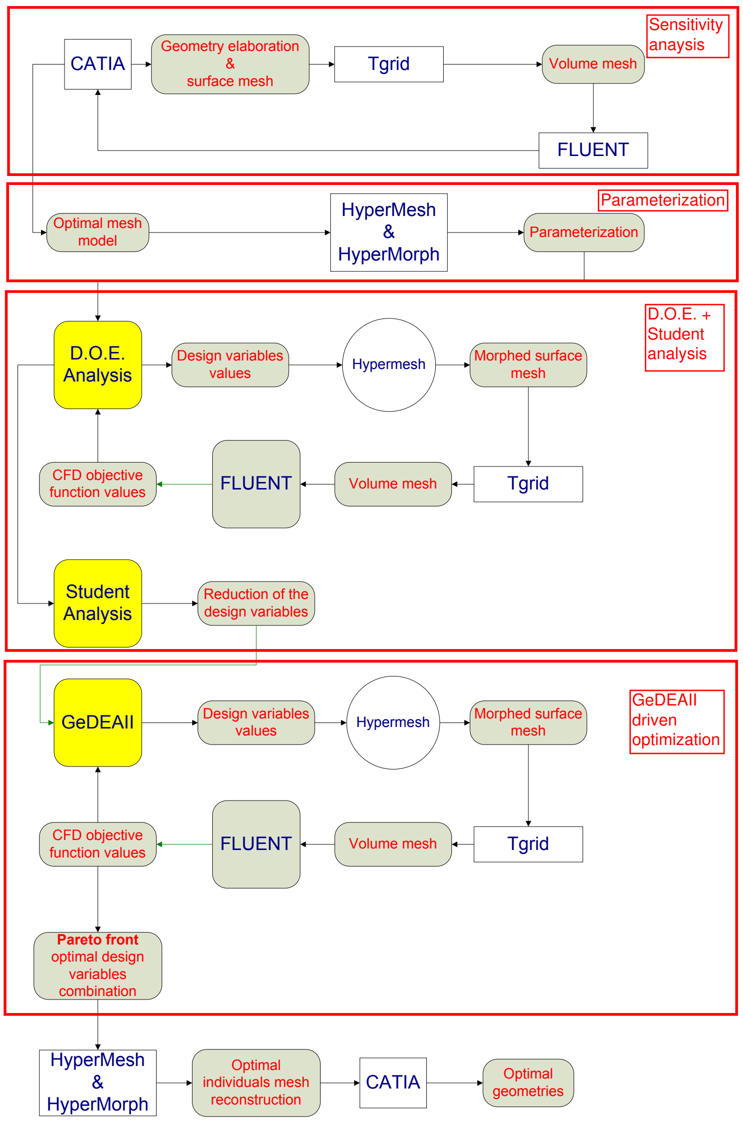


Figure 7.45: Flow chart of the complete optimization procedure.

addition to these components, a tail cone (Figure 7.48) was added to the simulated geometry in order to avoid flow separations downstream the cylindrical portion of the fuselage which could detrimentally affect convergence behavior of numerical simulations.

As far as the optimization target is concerned, a reduction of 2% in the C_d of the baseline ERICA nose was required. The reference flight conditions are summarized in Table 7.14.

The reader is referred to the Acronyms chapter for the definition of fuselage incidence.

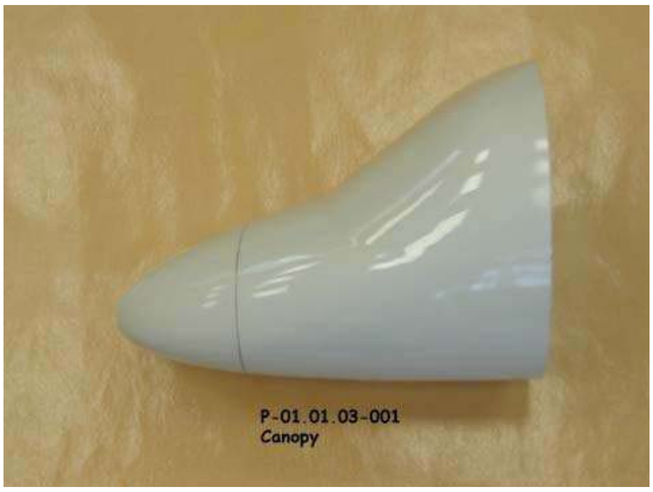


Figure 7.46: ERICA nose geometry (taken from [74])

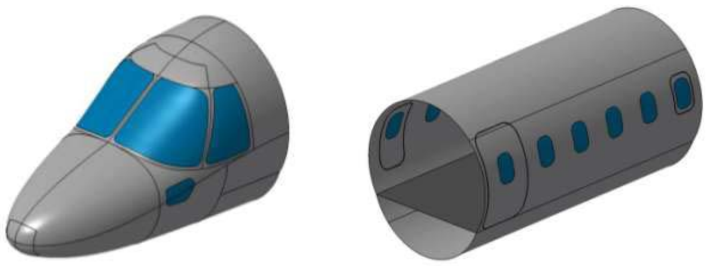


Figure 7.47: ERICA nose (on the right) and cylindrical portion of the fuselage (on the left) (taken from [74])

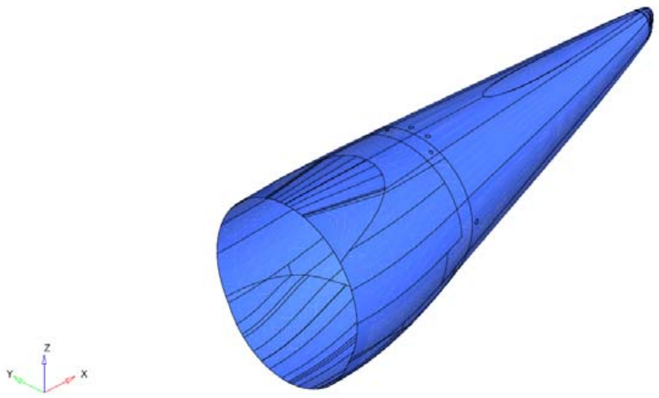


Figure 7.48: ERICA tail cone (taken from [74])

Flight Condition	True air speed, $[Kts]$	True air speed, $[m/s]$	Pressure altitude, $[m]$	Static pressure, $[Pa]$	OAT $[C]$	Density $[kg/m^3]$	Speed of sound $[m/s]$	Fuselage incidence $[deg]$	Nacelle attitude $[deg]$	Outer wing setting angle $[deg]$	Rotor RPM	One rotor thrust, $[N]$
Forward Flight cruise	*350*	*180.06*	*7500*	*38251*	*-33.75*	*0.556623*	*310.175*	*-1.97*	*0.00*	*0.00*	*425.8*	*12431*

Table 7.14: Flight Conditions for Nose optimization (taken from [74])

7.16 Geometry elaboration

As already mentioned, the tiltrotor model was made up of three components, specifically the canopy, the fuselage and the tail cone, depicted in Figure 7.47 and Figure 7.48 respectively. The aircraft fuselage was inserted into a virtual wind tunnel, whose dimensions were varied during the sensitivity analysis, as will be described later. A sketch of the wind tunnel box is depicted in Figure 7.49.

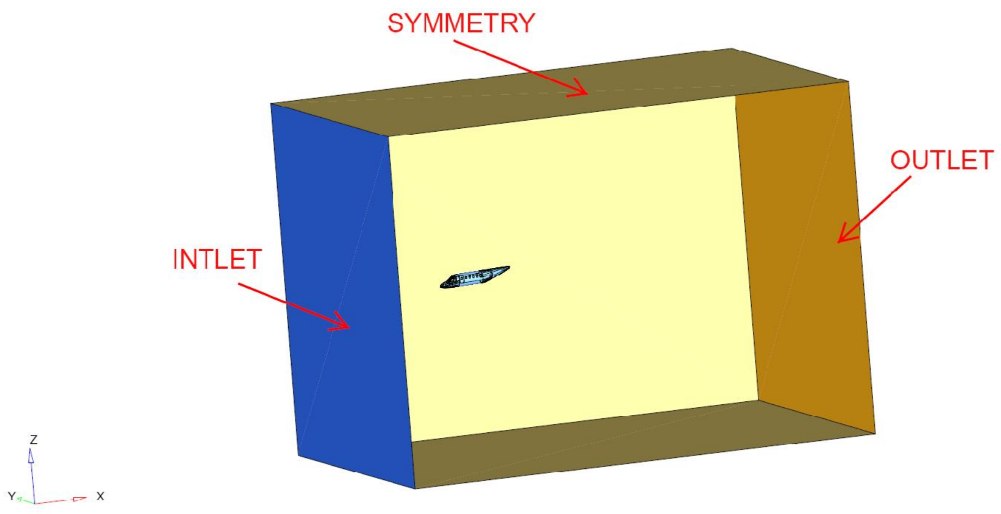


Figure 7.49: Virtual wind-tunnel layout with the tiltrotor fuselage.

In CATIA® environment, the model surfaces were organized into six distinct components, namely:

1. **Inlet**, which is depicted in Figure 7.49;
2. **Outlet**, which is depicted in Figure 7.49;
3. **Symmetry**, which is depicted in Figure 7.49;
4. **Nose**, which corresponds to the canopy surface, depicted in Figure 7.46;

5. **Fuselage**, which corresponds to the cylindrical portion of the fuselage surface, depicted in Figure 7.47;

6. **Tail cone**, which corresponds to the tail cone surface, depicted in Figure 7.48.

7.17 Mesh generation

As far as the mesh generation is concerned, the software used to carry out the meshing operations was respectively:

1. CATIA® surface meshing tool for the surface mesh generation;

2. Ansys Tgrid® for the volume mesh generation.

7.17.1 Superficial mesh

The CATIA® surface meshing tool can be used to mesh the geometry surfaces directly from the CATIA® format, so no import/export operations are necessary to accomplish this task. Moreover, the surface mesh obtained from the CATIA® meshing tool is entirely interfaced with the CATIA® geometry. This characteristic is important in an industrial context, because a modification of the baseline geometry can be directly transformed into a correspondent surface mesh modification simply by updating the mesh.

The surface mesh was more refined over the tiltrotor nose, which was the object of optimization, whereas the superficial grid over the cylindrical portion of the fuselage was slightly coarser; finally, the tail cone surface mesh was the coarsest one. The choice of adopting different mesh sizes was motivated by the attempt to contain the total amount of surface elements (and consequently the number of 3D cells) as much as possible.

The transition between two zones with different mesh size was accomplished by properly selecting CATIA® Advanced meshing tool parameters. This feature is a really important characteristic of a CFD mesh, and it is fundamental for a good CFD solution convergence. Two different surface mesh distributions on the ERICA surface were investigated during the sensitivity analysis, as will be illustrated in chapter 7.19. The finally selected model was characterized by a number of triangular elements equal to 175.191; the corresponding target element size (expressed in $[m]$) and the number of superficial elements for each of the fuselage components are summarized in Table 7.15.

A sketch of the finally selected surface mesh over the nose component is reported in Figure 7.50, while the superficial mesh over the virtual wind tunnel walls is illustrated in Figure 7.51. Some words are worth spending as regards the quality criteria accomplished during superficial mesh generation. Actually, in order to achieve a good mesh quality it was necessary to define and set proper quality parameters before starting the meshing operations; these parameters were then checked during the mesh generation. AgustaWestland quality requirements can

	Number of elements	Target element size $[mm]$
Inlet	1200	3500
Outlet	1200	3500
Symmetry	6512	3500
Canopy	73648	30
Fuselage	76994	40
Tail cone	15636	80

Table 7.15: Finally selected number of elements and target element size for each fuselage component

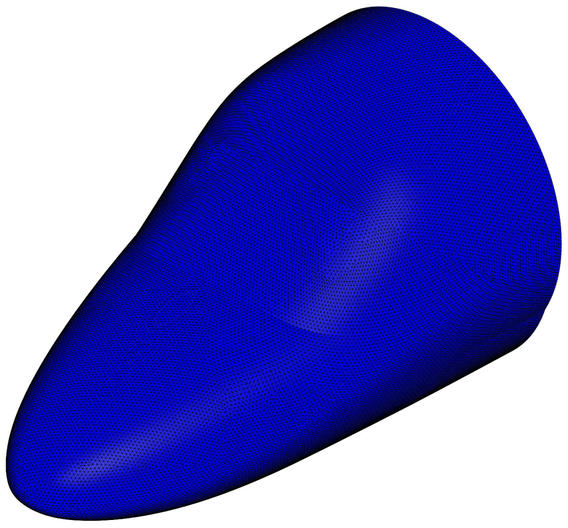


Figure 7.50: Superficial mesh over ERICA nose.

be met by imposing proper limits on *Skewness* and *Aspect-Ratio* within CATIA®. These parameters can be defined as follows:

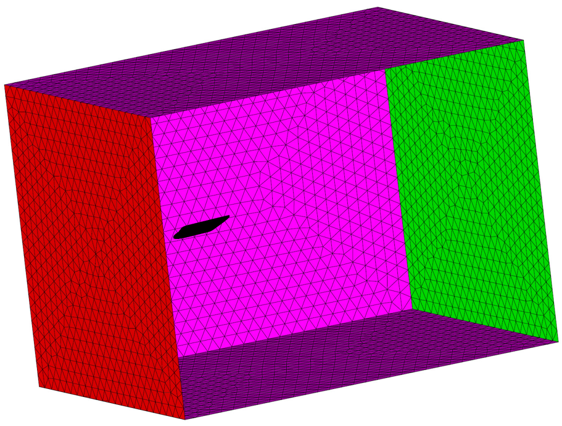


Figure 7.51: Superficial mesh over the virtual wind tunnel walls.

- **Skewness**: it is a quantitative measure of the element distortion with respect to the ideal element shape (equilateral triangle for triangular surface mesh).

  Skewness is calculated within CATIA® in the following way:

$$Q = 1 - \frac{S}{S'} \tag{7.10}$$

  where Q is the Skewness, S is the ideal element surface, and S' is the actual element surface.

- **Aspect-Ratio**: it is the ratio between the maximum and minimum length among the sides of the element; for instance, the Aspect-Ratio AR of a triangular element of sides l_1, l_2, l_3 is defined as:

$$AR = \frac{max\,(l_1, l_2, l_3)}{min\,(l_1, l_2, l_3)} \tag{7.11}$$

The parameters limits specified for the ERICA model superficial mesh are summarized in Table 7.16. The mesh statistics of the finally selected superficial mesh with

	Best value	Poor at	Bad at	Worst value
Skewness	0	0.3	0.5	1
Aspect-Ratio	1	1.5	3	$1 * 10^6$

Table 7.16: Quality parameters limits specified for ERICA surface mesh.

respect to the above mentioned criteria are summarized in Table 7.17 and Figure 7.52.

	Optimal elements	Poor elements	Bad elements	Worst element	Mean value
Skewness	175191 (100 %)	3 (0.00 %)	2 (0.00 %)	0.539	0.034
Aspect-Ratio	175168 (99.98 %)	28 (0.02 %)	0 (0.00 %)	2.369	1.144

Table 7.17: Mesh statistic for the finally selected ERICA superficial grid.

As apparent, AgustaWestland quality criteria were fulfilled with wide margins. Moreover, all the bad elements were located near the tail cone component, in the region adjacent to the fuselage surface, so that they were expected not to affect the simulated drag values of the nose, which was the object of the optimization.

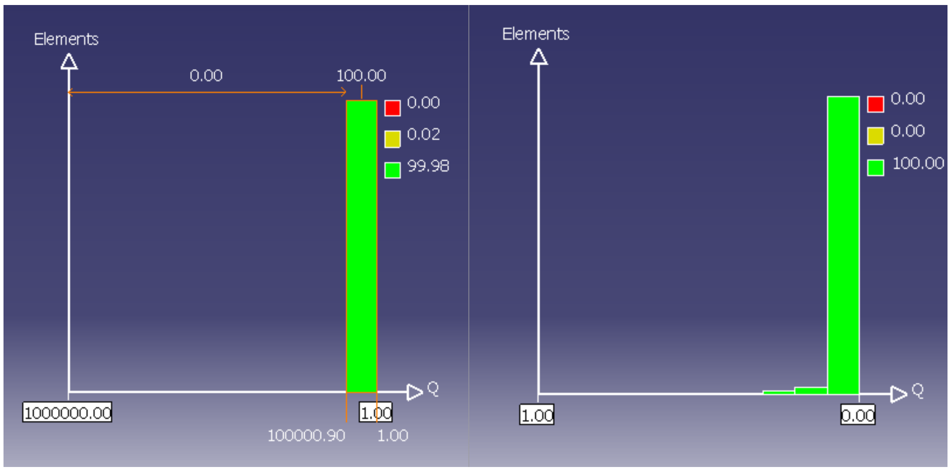


Figure 7.52: Histograms representing ERICA surface mesh quality statistics: *Aspect-Ratio* (on the left); *Skewness* (on the right).

7.17.2 Volume grid

Starting from the surface mesh described above, the volume mesh was generated by means of the CFD meshing tool Ansys Tgrid®. Actually, the surface mesh created within CATIA® can be saved in a Nastran compatible format (.dat) which can be easily read by Tgrid®.

In Tgrid® environment, the volume between the tiltrotor surface and the wind-tunnel walls was filled with tetrahedral elements, while the boundary layer region was modelled using prismatic cells. In Table 7.18, the Tgrid® settings selected to generate the internal volume mesh are presented.

Ten prismatic layers were built over the tiltrotor fuselage in order to ensure that the simulated boundary layer was entirely contained within the prismatic layers over the whole fuselage, with the exception of regions of separated flow. Moreover,

Components with boundary layer	Canopy, Fuselage, Tail cone
Components without boundary layer	Inlet, Outlet, Symmetry
B.L. Offset method	Uniform
B.L. Growth method	Geometric
B.L. Number of layer	10
B.L. First height	1
B.L. Growth rate	1.2
Tri/Tet Improve surface mesh option	Enabled
Tri/Tet Refinment method	Adv/front
Tri/Tet Cell size function	Geometric
Tri/Tet Growth rate	1.2
Tri/Tet Refinement regions	yes

Table 7.18: Ansys Tgrid® settings for volume mesh generation.

a height of 1 mm was given to the first prismatic layer in order to meet the y+ requirements for the wall function calculation during the CFD simulation ([7]). An important aspect that has to be pointed out is the use of local refinement regions, which allowed a local improvement of the volume mesh around the ERICA model, in particular near the nose region, while a coarser tetrahedral mesh could be used in the remaining internal regions.

As a result, a really good mesh quality was obtained in the regions of interest (that is, the nose and the cylindrical portion of the fuselage), with the minimum number of 3D elements. A longitudinal view of the whole volume mesh is depicted in Figure 7.53, while Figure 7.54 illustrates a close-up of the mesh near the tiltrotor surface, where the refinement regions around the fuselage can be clearly appreciated. Finally, a particular of the boundary layer over the canopy is reported in Figure 7.55.

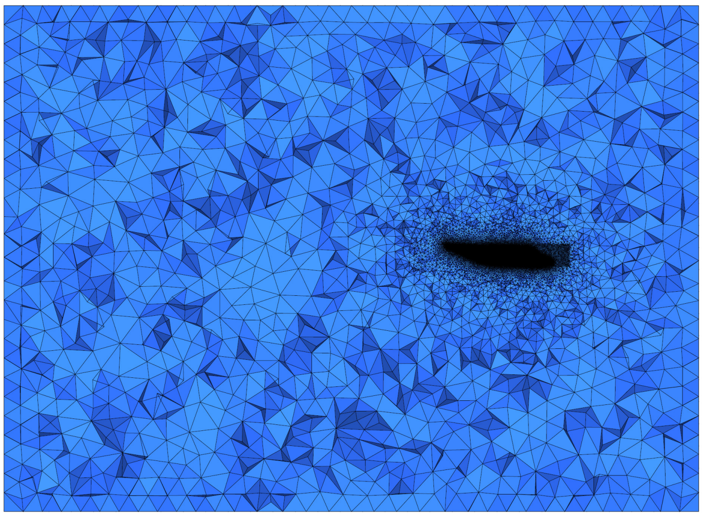


Figure 7.53: Longitudinal view of the whole volumetric mesh around the tiltrotor fuselage.

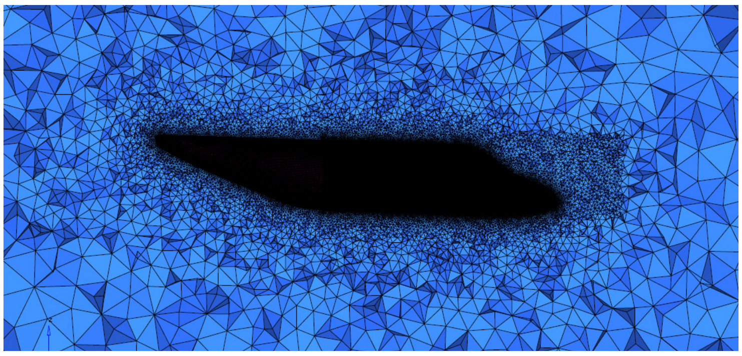


Figure 7.54: Close-up of the volume mesh near the fuselage.

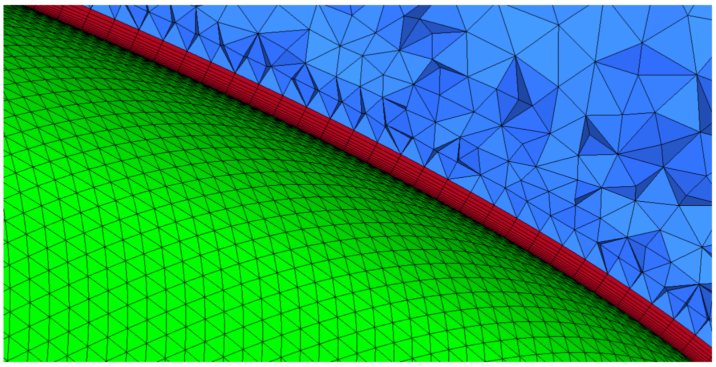


Figure 7.55: Close-up of the volume mesh: boundary layer over the canopy.

7.18 Fluid-dynamic model set up

A pressure-based solver type with absolute velocity formulation and steady approach was adopted for the tiltrotor simulations. The $\kappa - \omega$ SST model was selected for turbulence handling. The air was treated as an ideal gas having constant specific heats, which automatically enables the equation energy resolution. This makes it possible to include the compressibility effects in the numerical simulations: in fact, as it can be deduced from Table 7.14, the flight condition selected for optimization features a moderately high Mach number.

In Table 7.19, the air properties assigned for the ERICA nose optimization are summarized. As far as the solution algorithm is concerned, a SIMPLE scheme was adopted, which solves the pressure and moment equations separately. A third order MUSCL discretization scheme was selected for all the variables, since it guarantees a high accuracy of the numerical solution, due to its potential to improve spatial accuracy by reducing numerical diffusion, particularly for complex three-dimensional flows, while not negatively affecting the total time requested for simulations when compared to the second order upwind scheme. In particular, for the gradient spatial discretization, the Green-Gauss node based discretization scheme was chosen instead of the more common Green-Gauss cell based or Least squares cell based schemes, because it is more suitable for unstructured tetrahedral mesh [7], as is the case for the tiltrotor fuselage simulation.

Gas thermo-dynamic model	Ideal Gas
Cp Specific Heat $[J/kg \cdot K]$	Constant: 1006.43
Thermal conductivity $[W/m \cdot K]$	Constant: 0.0242
Viscosity variation law	Sutherland: three coefficients methods
μ_0 *Reference viscosity* $[kg/m \cdot s]$	$1.716 \cdot e^{-5}$
T_0 *Reference temperature* $[K]$	273.11
S Effective temperature $[K]$	110.56

Table 7.19: Air properties for ERICA nose optimization.

Solver type	Pressure-based steady-state
Pressure-velocity coupling scheme	SIMPLE
Spatial discretization schemes	
Gradient	Green-Gauss node based
Pressure	Second order
Density	Third-order Upwind
Momentum	Third-order Upwind
Turbulent kinetic energy	Third-order Upwind
Specific dissipation rate	Third-order Upwind
Energy	Third-order Upwind

Table 7.20: Solver settings and discretization schemes.

The selected solver settings and discretization schemes are summarized in Table 7.20.

7.18.1 Boundary and operating conditions

The boundary conditions for the optimization study were selected according to the operating flight conditions reported in Table 7.14.

A total pressure condition was imposed on the wind tunnel inlet, while a static pressure was assigned over the outlet section. Over the lateral walls of the wind tunnel, together with the top and bottom surfaces, a symmetry condition was imposed. All the other surfaces were treated as hydraulically smooth, adiabatic walls. The operating reference condition was set to the reference static value for the prescribed flight condition in Table 7.14, $p = 38251.4[Pa]$.

The pressure inlet boundary condition imposed on the inlet boundary zone requires the specification of both the gauge total pressure and the total temperature.

These quantities were determined applying the ideal gas laws to the data of Table 7.14; in fact, from the static temperature (T_∞ in [K], corresponding to the OAT given in Table 7.14) and the velocity (V_∞ , the True air speed) the operating Mach number (Ma_∞) was calculated as:

$$Ma_\infty = \frac{V_\infty}{\sqrt{kRT_\infty}} \tag{7.12}$$

where k is the specific heats ratio (1.4 for dry air) and R is the gas constant (287 $[J/(kg \cdot K)]$ for dry air). The resulting operating Mach number is 0.5806. The following equations were used to calculate the total pressure and total temperature at the inlet:

$$P_T = P_\infty \left(1 + \frac{k-1}{2} Ma_\infty^2\right)^{\frac{k-1}{k}} \tag{7.13}$$

$$T_T = T_\infty \left(1 + \frac{k-1}{2} Ma_\infty^2\right) \tag{7.14}$$

being P_∞ the static pressure given in Table 7.14. Subtracting the value of the reference pressure to the total pressure calculated with Equation 7.13, we obtained a gauge total pressure of 9811.29 [Pa], while the resulting total temperature from Equation 7.14 was 255.54 [K].

Table 7.21 summarizes the settings for the pressure inlet boundary condition.

Gauge total pressure $[Pa]$	9811.29
Supersonic/initial gauge pressure $[Pa]$	0
Direction specification method	normal to boundary
Turbulence specification method	Intensity and length scale
Turbulent intensity $[\%]$	1
Turbulent length scale $[m]$	0.5
Total temperature $[K]$	255.54

Table 7.21: Pressure inlet specification at inlet.

The pressure outlet boundary condition requires the specification of the gauge static pressure and the back-flow total temperature at the outlet surfaces: a relative zero gauge pressure (corresponding to the undisturbed value) and a backflow total temperature equal to the inlet static temperature were chosen. Table 7.22 summarizes the settings for the pressure outlet boundary condition on the outlet surfaces. Regarding the turbulence specification method, a turbulence intensity of 1% was selected. As far as the turbulent length scale is concerned, the final selected value 0.5 [m] was the result of an iterative calculation process. In fact, the turbulent length scale can be defined as $\sqrt{\frac{\kappa}{\omega}}$. Hence, the values of κ and ω were iteratively computed at both inlet and outlet on a CFD model characterized by an initial turbulent length scale of 1 [m]. However, some tests were performed that allowed to verify that the tiltrotor nose drag was quite insensitive to variations of the turbulent length scale, at least for reasonable values of this parameter.

Gauge pressure $[Pa]$	0
Direction specification method	normal to boundary
Turbulence specification method	Intensity and length scale
Turbulent intensity $[\%]$	1
Turbulent length scale $[m]$	0.5
Backflow Total temperature $[K]$	239.4

Table 7.22: Pressure-outlet specification at outlet.

7.19 Grid sensitivity analysis

In this section, a preliminary sensitivity study which was carried out in order to identify a suitable mesh to be used during the nose optimization is described. The final aim was to obtain a reliable CFD model while saving computational time and resources.

To this purpose, the following strategy was followed:

1. First, the wind tunnel cross-sectional area influence on drag characteristics was investigated, the superficial mesh and the box length being kept fixed; the nose drag was computed via a surface integral of both pressure and viscous drag.

2. Once the proper cross-sectional area was selected, the influence of the box length was analysed.

3. Next, the superficial mesh impact on aerodynamic performances was investigated and a mesh size was selected, resulting from a trade-off between the accuracy in predicting aerodynamic drag and the requested CFD calculation time, which greatly depends on the mesh size.

4. Finally, the influence of the volume mesh refinement on the objective function value was considered, and the final model was selected.

As far as the wind tunnel cross-sectional area is concerned, a square section was adopted and several dimensions were investigated, while the box length was kept fixed to 84 [m]. Moreover, the superficial mesh over the tiltrotor surface was held unchanged, with an element size of 30 [mm] over the canopy, 40 [mm] over the cylindrical portion of the fuselage and 80 [mm] on the tail cone.

The analyzed configurations are summarized in Table 7.23, where the nose drag values are reported for each of the investigated cross-sectional dimensions.

In Figure 7.56, a graphical representation of the nose drag sensitivity to cross-sectional area is illustrated. As apparent, the nose drag increases with increasing cross-sectional area, (due mainly to the pressure drag component) until a plateau is reached. The drag value appears to be stabilized for an 80X80 $[m^2]$ cross section, so this model was selected and submitted to further sensitivity analyses.

Wind tunnel dimensions $[m]$	Nose pressure DRAG $[N]$	Nose vosicous DRAG $[N]$	Total nose DRAG $[N]$
30x30x84	357	666	1024
40x40x84	526	665	1192
60x60x84	627	663	1291
80x80x84	672	663	1336
100x100x84	664	663	1327

Table 7.23: Drag sensitivity to wind tunnel cross-sectional area dimensions.

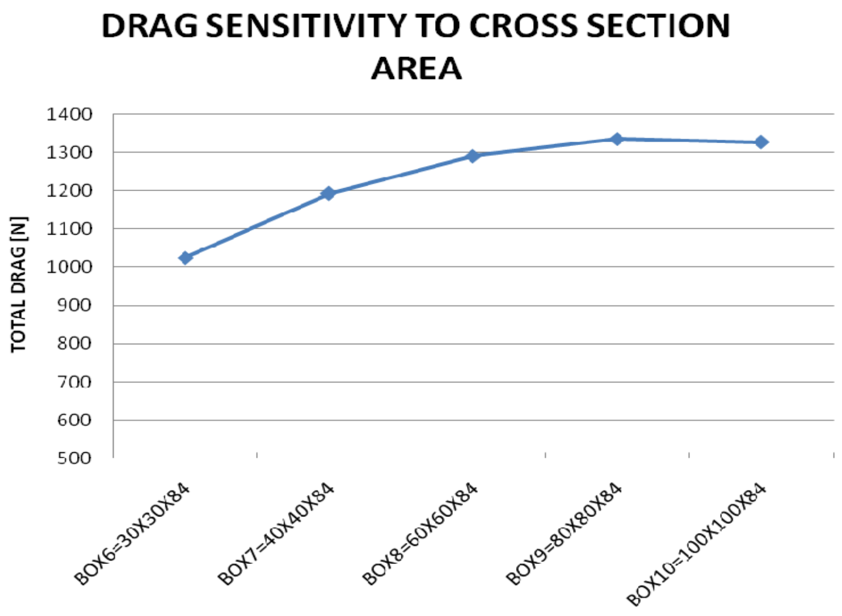


Figure 7.56: Nose drag sensitivity to cross-sectional area.

Next, the influence of the box length was investigated. Specifically, starting from a wind-tunnel length equal to 84 $[m]$, the box length was increased, in order to better reproduce the undisturbed flow at the domain exit, and the effect on the nose drag was monitored. The variations in box length were shown to slightly affect nose overall drag values, as apparent from Table 7.24 and Figure 7.57: actually, variations of nose drag are in the order of 10 $[N]$ within the considered range of box lengths. In light of this, it was decided to select the model with the smallest number of elements among the four ones investigated, which is the 80x80x110 $[m^3]$. It was characterized by a total number of elements equal to 2.7 millions.

The third step of sensitivity analysis concerned the superficial mesh refinement level. To this purpose, starting from the above mentioned element sizes, the mesh was refined and the effect on nose drag was investigated. Specifically, the new superficial mesh was characterized by an element size of 20 [mm] over the canopy, 30 [mm] over the cylindrical portion of the fuselage and 80 [mm] on the tail cone. The comparison between the two models is summarized in Table 7.25: in

Wind tunnel dimensions $[m]$	Nose pressure DRAG $[N]$	Nose viscous DRAG $[N]$	Total nose DRAG $[N]$
BOX4=80x80x110 $[m^3]$ Length=110 m, 22 m upstream the fuselage	643	663	1316
BOX4=80x80x125 $[m^3]$ Length=125 m, 40 m upstream the fuselage	636	660	1299
BOX4=80x80x145 $[m^3]$ Length=145 m, 60 m upstream the fuselage	638	662	1300
BOX4=80x80x180 $[m^3]$ Length=180 m, 80 m upstream the fuselage	646	660	1306

Table 7.24: Drag sensitivity to wind-tunnel box length.

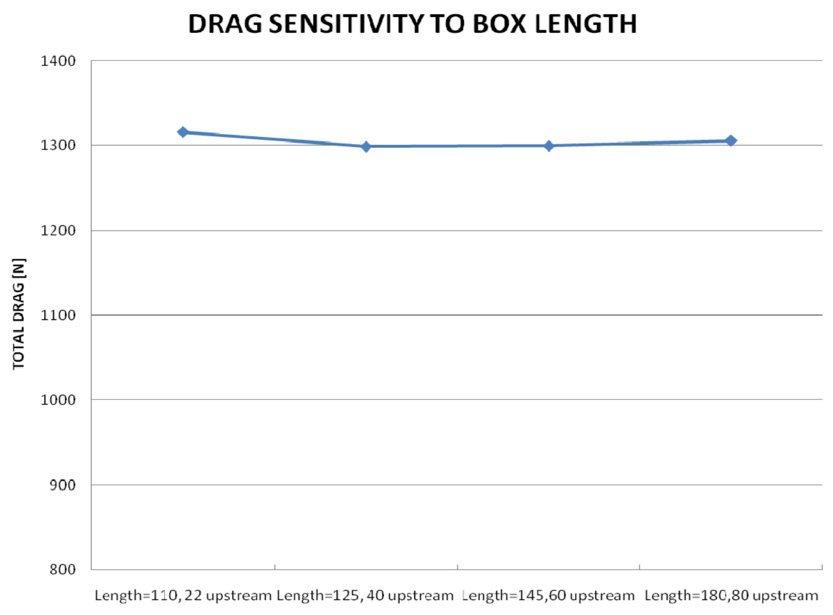


Figure 7.57: Nose drag sensitivity to wind-tunnel box length.

the first column the superficial grid refinement is reported, with the first number representing the mesh size of the nose component, the second one the mesh size of the fuselage component, and the third one the mesh size of the tail component. As

Superficial mesh refinement $[m]$	Nose pressure DRAG $[N]$	Nose viscous DRAG $[N]$	Total nose DRAG $[N]$
30-40-80 [mm]	643	663	1316
20-30-40 [mm]	644	668	1318

Table 7.25: Drag sensitivity to superficial mesh size.

apparent, the nose drag change is nearly negligible when refining the superficial mesh over the nose and cylindrical fuselage. On the other hand, with the refined superficial mesh the total number of volume cells rises up to 4.5 millions of elements. Hence, due to computational reasons, the less refined superficial mesh was retained, with 30 [mm], 40 [mm] and 80 [mm] element size over the nose, cylindrical fuselage and tail cone respectively.

Finally, the effects of the volume grid refinement were investigated. It was realized that the volume grid refinement has a great influence on the nose drag. In particular, drag was found to decrease with decreasing growth rate of tetrahedral elements, which governs the transition of the elements size from the inner region of the domain (near to the prismatic layers) to the external region. Hence, a sensitivity analysis was performed, in order to highlight the influence of this parameter on the drag value, keeping the wind tunnel dimensions fixed to 80x80x110 [m^3] and holding the superficial grid element size unchanged to the 30-40-80 [mm] combination. Results are reported in Table 7.26.

Volume mesh refinement	Nose pressure DRAG [N]	Nose viscous DRAG [N]	Total nose DRAG [N]
Growth Ratio: 1.6	643	663	1316
Growth Ratio: 1.2	253	665	918

Table 7.26: Volume mesh refinement influence on the nose drag characteristics.

As apparent, while the viscous drag is nearly insensitive to the volume grid refinement, the pressure component decreases with increasing refinement. Given that the number of elements in the latter configuration rises up to 4.4 millions, a further refinement was not taken in consideration, because a further increase in the amount of mesh elements would have become prohibitive from the computational time point of view. Moreover, the validation of the model with growth ratio equal to 1.2 gave satisfactory results, as will be illustrated in chapter 7.20, so this model was retained for the optimization study.

Hence, the finally selected model, which was then used in the following validation analysis, is the 80x80x110 [m^3] wind tunnel box, 30-40-80 [mm] superficial element size, with volumetric growth ratio equal to 1.2. Once the model was selected, the y+ characteristics of the associated CFD solution were investigated.

In Figure 7.58, the y+ distribution over the tiltrotor fuselage is illustrated.

As apparent, the selected parameters for grid generation were shown to guarantee that the non-dimensional mesh thickness at the fuselage surface fell within the discretization levels ($y+ = 30 \div 300$) suggested for the standard wall functions implemented in the conventional turbulence models to work properly, in particular over the nose region, where the fluid-dynamic quantities necessary for the optimization objective function evaluation are calculated.

7.20 Validation of the CFD model

Some experimental data on a non-powered 1/8 scaled ERICA tilt-rotor modular model were available from a series of low speed wind tunnel tests performed in the framework of WP4.4 of the NICETRIP Project (FP6/Aeronautics project AIP5-CT-2006-030944).

The experimental campaign was held at Politecnico di Milano and split into three major phases, in which different model configurations were tested. An exhaustive analysis of the acquired aerodynamic coefficients of both the whole

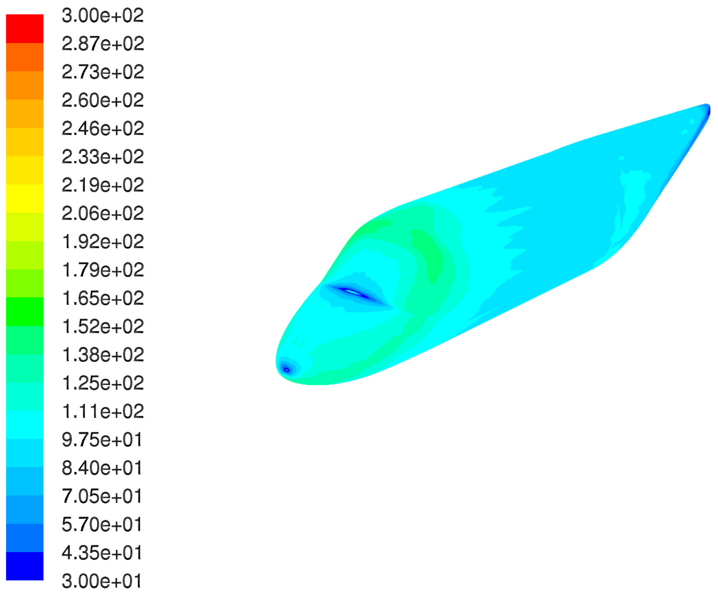


Figure 7.58: Contours of wall y+ over the ERICA fuselage.

aircraft and the isolated components was carried out in [73]. Specifically, the contribution of each component to the model global aerodynamic coefficients was highlighted, with the main objective of investigating the aircraft drag breakdown. Actually, the above mentioned experimental observations refer to specified values of Reynolds and Mach numbers typical of the wind tunnel environment and of the scaled model dimensions. In [73], an extrapolation of the wind tunnel measured data on the model scaled tilt-rotor was carried out in order to infer some basic aerodynamic characteristics of the full scale aircraft: viscosity and compressibility effects were accounted for by means of some empirical correlations published in the literature. Starting from the experimental measurements, the lift and drag coefficients of the main aircraft components in airplane mode were derived at typical full scale cruise Reynolds and Mach numbers. Then, their cumulative effect on the global aerodynamic coefficients of the aircraft was evaluated.

In this chapter, a validation of the tiltrotor CFD model selected for the optimization runs was performed against wind tunnel data over the bare fuselage. Specifically, the geometry and mesh were scaled down to the actual 1/8 wind tunnel model dimensions and the fluid dynamic simulation was carried out at the experimental conditions, with an incidence angle equal to that selected for the optimization runs. Then, the simulated fuselage drag was compared with the experimental acquisitions. As will be illustrated in the following, a good agreement of the calculated drag with experiments was found at wind tunnel conditions. When taking into account the effects of both full-scale Reynolds and Mach number at real flight conditions on the aerodynamic coefficients, this simulation served as a validation of the numerical model to be used in the optimization phase. Actually, being the fuselage drag at wind tunnel conditions captured with a satisfactory accuracy, the CFD model was judged capable of properly reproducing the tiltrotor aerodynamic behaviour even at the full-scale conditions proper of the optimization runs.

In Table 7.27, the flowfield characteristics of the selected test case for validation at model-scaled conditions are summarized, in terms of flow static pressure and temperature, Reynolds number referred to the wing mean aerodynamic chord and Mach number.

Static pressure $[Pa]$	97956
Static temperature $[K]$	294.18
Air density $[kg/m^3]$	1.16
Air speed $[m/s]$	44.85
$Re_{wingchord}$	8.73751E+05
Mach	0.13

Table 7.27: Flowfield characteristics at wind tunnel conditions.

The same fluid dynamic set up of the optimization runs was used for the model validation. In particular, a pressure-based solver type with absolute velocity formulation and steady approach was selected for the simulations. The $\kappa - \omega$ SST model was chosen as the viscous model with turbulence intensity equal to 1%, and the air was treated as an ideal gas having constant specific heats. The boundary conditions were as follows: a total pressure condition was imposed on the wind tunnel inlet, while a static pressure was assigned over the outlet section, the actual values of both depending on the data of Table 7.27: to all the lateral surfaces of the wind tunnel and to its top and bottom surfaces a symmetry condition was applied. The fuselage surface was treated as a hydraulically smooth, adiabatic wall.

The adopted boundary conditions for the model-scaled case are summarized in Table 7.28.

		Model Scale
INLET	total pressure $[Pa]$	99119.72
	static pressure $[Pa]$	97956
	total temperature $[K]$	295.2
OUTLET	static pressure $[Pa]$	97956
	backflow total temperature $[K]$	295.2

Table 7.28: Boundary conditions on the fluid domain for the validation case.

As far as the solution algorithm is concerned, a SIMPLE scheme was adopted, along with a third order MUSCL discretization scheme for all the variables. A normalized residual on the continuity equation around $1 * 10^{-6}$ was obtained at the end of the simulation run and both the fuselage lift and drag coefficients reached stabilized values.

In Table 7.29, the fuselage drag coefficients of the model-scaled tiltrotor coming from the simulations is reported and compared to the experimental one. The drag was predicted with a satisfactory accuracy, leading to a percentage error of 9 % with respect to the wind tunnel value, which is considered good for the optimization purposes.

In Figure 7.59, the experimental drag curve of the tiltrotor fuselage is depicted, along with the obtained CFD results.

	Experiment at $\alpha = -1.99[deg]$	**Simulation at** $\alpha = -1.97[deg]$	% **deviation**
Model-scaled Drag coefficient	0.01883	0.017	-9[%]
Reference Area= 0.578 $[m^2]$			

Table 7.29: Model-scaled tiltrotor fuselage drag coefficient: simulations vs. experiments.

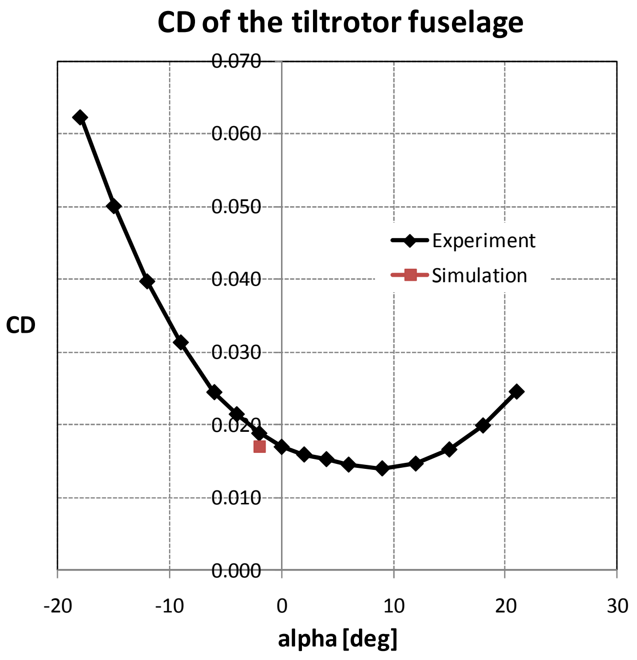


Figure 7.59: Model-scaled tiltrotor fuselage drag curve: simulations vs. experiments.

Given that the correlation with model-scaled experiment was accurately captured, the CFD model was judged accurate enough also when transposing the results at the higher Reynolds and Mach number proper of the operating flight conditions, at which the nose optimization was carried out.

7.21 Parameterization

Once the baseline CFD solution was available, the geometry parameterization was carried out. This operation is of outstanding importance in the optimization process and it can be performed using various techniques. For the scope of the present work, the commercial software HyperMesh® was chosen as the parameterization tool, due to the really strong and versatile capabilities of the morphing tool HyperMorph®. The choice of this software is mainly justified with its effectiveness and ease of use, which allows the user to build up complicated parametric models in a short time by means of a graphical user interface. Moreover, the parameterization procedure in HyperMorph® is very general, in the sense that it is independent from the peculiar model (either FEM or CFD) which is the object of the optimization analysis. Therefore, whatever the geometry, the user is always allowed to use the same morphing strategies, again saving a lot of working time. Both those characteristics are fundamental in an industrial context, where time is always an essential issue.

The strategy chosen to correctly perform the parameterization can be summarized with the following steps:

1. Importation of the baseline case file (i.e., the model 30-40-80, 80x80x110, growth ratio 1.2) into HyperMesh® in *Nastran* Format.

2. Creation of the desired shapes, starting from the baseline geometry, through the morphing techniques available within HyperMorph®. For CFD studies, the more suitable strategy is the domains-handles approach to perform localized deformation, and the morph-volumes approach to satisfy global morphing requirements.

3. Saving of the generated shapes; with this operation, HyperMorph® stores the current handles − nodes perturbations allowing the user to apply them to the undeformed model with any given scaling factor. With this method, the scaling factor of any generated shape can be dealt with as a design variable by an optimization algorithm.

4. Declaration of the created shapes as design variables for the HyperStudy optimization analysis, following the guidelines presented in [4].

5. Saving of the current parameterized model into an .hm file, which will be then exploited for the batch-mode parameterization.

6. Creation of the HyperMesh® command file. An example of such a command file can be found in Appendix C.3; its function will be described in more detail in chapter 7.22.

7.21.1 Morphing Constraints

The only component to be optimized in the present study was the tiltrotor nose: hence, both the cylindrical portion of the fuselage and the tail cone were kept unchanged. In other words, all the nodes over both the fuselage and the tail cone surface (purple dots in Figure 7.60) were constrained to stay fixed.

Even though it slightly increased the computational resources required during the mesh deformation operations, the introduction of the morphing constraints was mandatory to obtain a suitable optimized geometry, conformal to non-aerodynamic requirements, such as manufacturing and structural constraints.

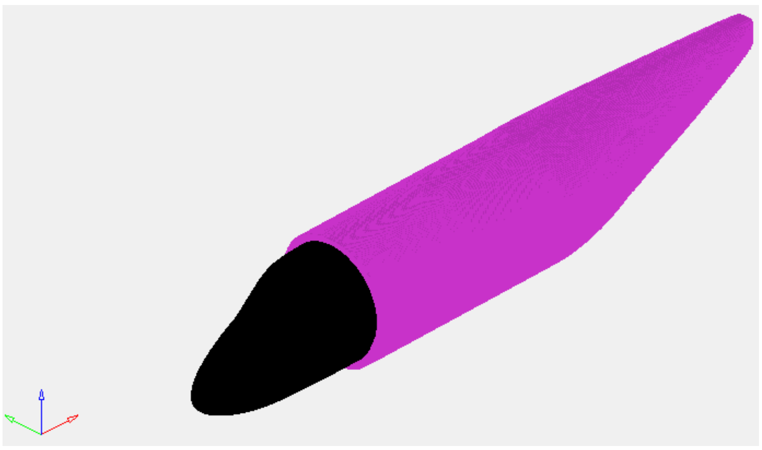


Figure 7.60: Fixed nodes over the fuselage and tail surfaces.

In addition, tangency constraints between the nose component and the cylindrical portion of the fuselage were introduced, by means of the *Set Biasing* option made available within Hypermorph® environment.

7.21.2 Design Variables Definition

In the following, a brief description of each design parameter taken into consideration for the nose optimization study is provided.

Obviously, some parameters are symmetrical with respect to the ERICA main axis; thereby they will be presented together, since they were deformed simultaneously during both D.O.E. analysis and the optimization study.

A total amount of nine design variables were considered. However, after the Student analysis only eight of them were shown to be influential on the nose drag, and they were retained for the optimization procedure.

All the design variables, with the exception of the *sh4*, were built by means of the *morph-volumes* approach in order to satisfy global morphing requirements. Moreover, with this approach more streamlined shapes of the nose could be achieved than those obtainable with the domains-handles approach. The following design variables were identified for the nose optimization:

1. ***sh1-sh2***: this was a symmetrical shape, which consisted of a rotation of the first eight handles of the morph volume block around two symmetrical axes, as illustrated in Figure 7.61; in this figure, the baseline shape along with the morph volumes created around the nose are presented. The initial deformation range assigned to this variable was equal to ±30 [deg];

2. ***sh3***: this shape consisted of the nose stretching along the negative direction of the x-axis. The deformation range was given a value of $0 \div 0.5[m]$. It is shown in Figure 7.62;

3. ***sh4***: this shape consisted of an initial deformation along z-axis in the nose region near the windscreen (Figure 7.63). It was given an initial deformation range of ±0.15 [m], due to pilot visibility constraints;

4. ***sh5***: this shape consisted of the rotation of two morph-volume handles, located at the interface between the nose and the cylindrical portion of the fuselage, as illustrated in Figure 7.64. It was given an initial deformation range of ±5 [deg], due to pilot visibility reasons.

5. ***sh6-sh7***: this was a symmetrical shape, consisting of a rotation of two morph-volume handles along two symmetrical axes, located at the interface between the nose and fuselage (Figure 7.65). It was given an initial deformation range of ±20 [deg], due to pilot visibility reasons;

6. ***sh8-sh9***: this shape was similar to the *sh1-sh2* one, but a wider portion of the canopy was involved in this case: in fact, the rotational axes were moved upstream (Figure 7.66).The initial deformation range assigned to this variable was equal to ±15 [deg];

7. ***sh10-sh11***: this was a symmetrical shape, conceived to deform the nose region along y-axis, as shown in Figure 7.67. It was given an initial deformation range of ±0.3[m];

8. ***sh12-sh13***: this shape was similar to the previous one, but it regarded a portion of the nose located downstream. The initially selected deformation range was equal to ±0.1 [m]. It is shown in Figure 7.68 (the arrows are small when compared to the ones of the previous shape, since they are proportional to the shape displacement);

9. ***sh14-sh15***: this was a shape located at an intermediate position between the previous ones, that is *sh10-sh11* and *sh12-sh13*, as apparent in Figure 7.69. It was given an initial deformation range of ±0.2 [m].

7.22 D.O.E. and Student analysis

Once both the sensitivity analysis and the geometry parameterization were completed, the Design of Experiment study loop was performed.

Design of Experiments (D.O.E.) can be defined as a test or a series of tests in which the input variables of a process or system are changed so that the reasons for changes in the output response can be identified and observed.

In this work, D.O.E. was applied with the aim of exploring the design space and determining which factors were most influential on the response. To this purpose, a Student analysis was performed exploiting the results of D.O.E.

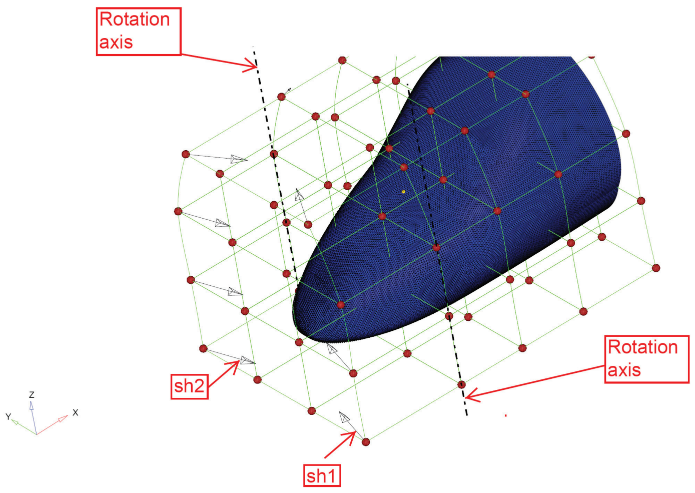


Figure 7.61: Shape *sh1-sh2* definition.

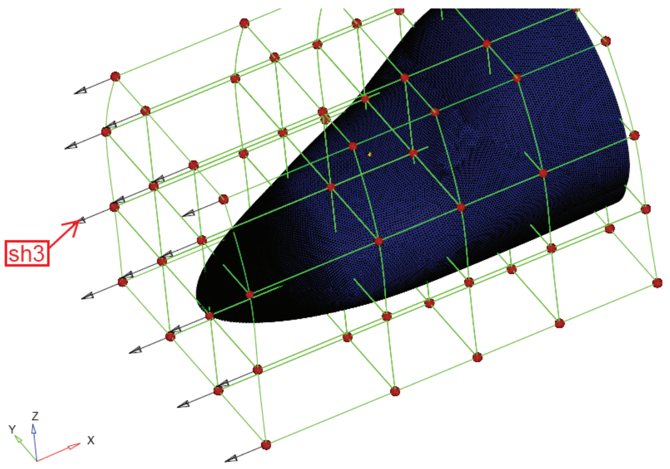


Figure 7.62: Shape *sh3* definition.

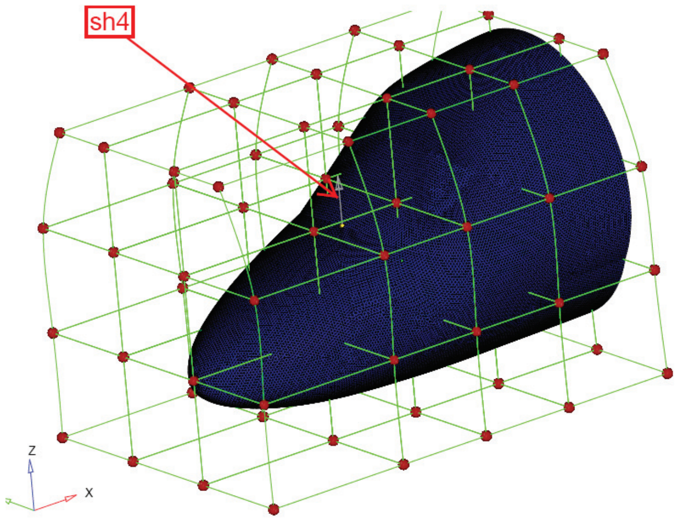


Figure 7.63: Shape *sh4* definition.

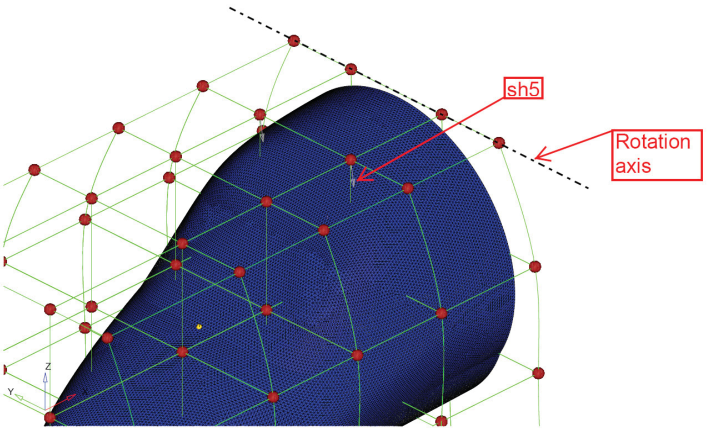


Figure 7.64: Shape *sh5* definition.

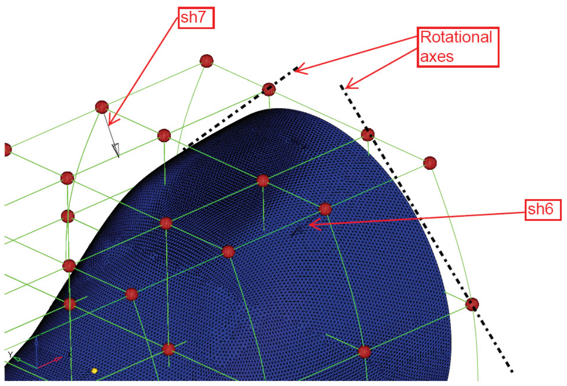


Figure 7.65: Shape *sh6*-*sh7* definition.

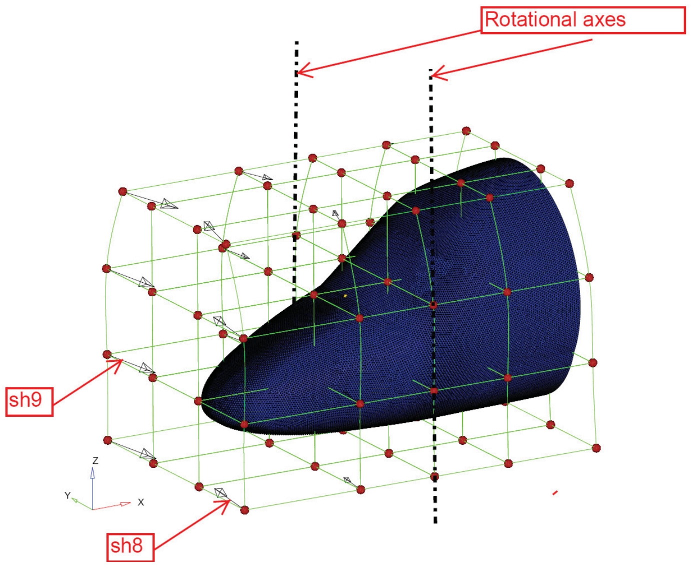


Figure 7.66: Shape *sh8-sh9* definition.

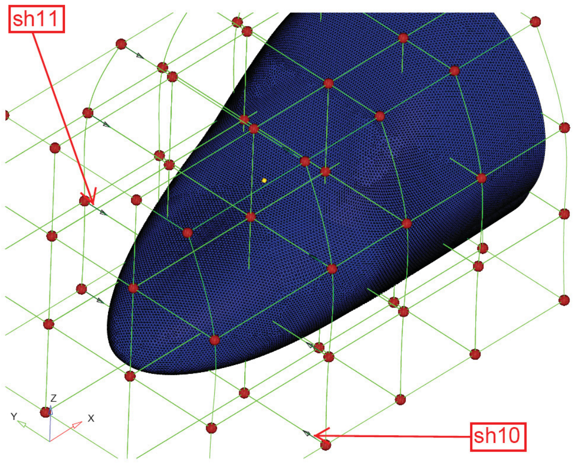


Figure 7.67: Shape *sh10-sh11* definition.

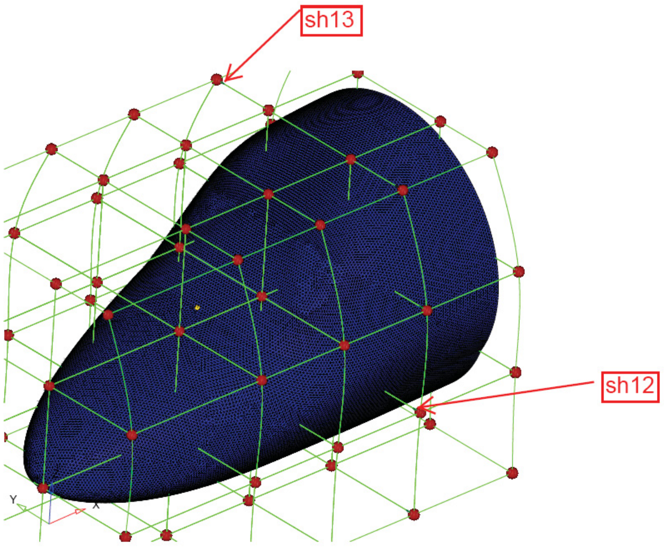


Figure 7.68: Shape *sh12-sh13* definition.

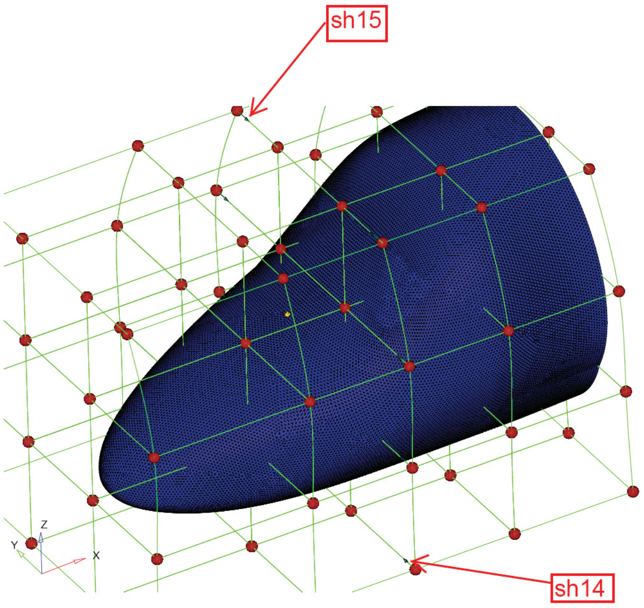


Figure 7.69: Shape *sh14-sh15* definition.

The basic operations performed within the D.O.E. loop are illustrated in the central part of the flow chart reported in Figure 7.45. Each program involved in the loop is interfaced and synchronized with the others so that, when one program is running, the following one waits until the preceding execution is complete. This allows to generate an input/output chain, in which each program uses the output of the preceding software as its own input.

The input/output chain illustrated in Figure 7.45 works as follows:

1. The D.O.E. Matlab® script, compiled in Matlab® environment using a dedicated command, creates the actual values of the design variables to be processed. A Latin Hypercube D.O.E. was selected in this work, which samples the entire design space into equal-probability regions, while allowing the designer to decide the amount of samples to be performed. A total number of 63 samples was selected, in order to keep the overall computational time to a reasonable level.

2. The design variables set are then imported, one at a time, into HyperMesh®, which, in turn, performs the shape parameterization according to the received values by means of its dedicated tool HyperMorph®. To this purpose, a tcl file *Command.tcl* was prepared to carry out these operations. The command that allows to launch HyperMesh® in batch mode in Windows® environment is the following:

   ```
   \emph{[path]/hmbatch.exe -c<Command.tcl>
   ```

   where `hmbatch.exe` is the batch-mode release of HyperMesh® (executable on Windows® platform), which performs shape parameterization in batch-mode, whereas `Command.tcl` is the name of the command file. HyperMesh® opens a previously created <filename.hm> file, containing the superficial mesh, along with the design variables built using HyperMorph® (please see chapter 7.21); then, the superficial deformations are carried out, and a Nastran format file (<filename.dat>) containing the deformed superficial mesh is created.

3. <filename.dat> is then transferred to Ansys Tgrid® software, which creates the volume mesh according to the instructions contained in a Ansys Tgrid® journal file, *Tgrid-batch.jou*, which can be found in Appendix C.4. As a result, a <filename.cas> is built, which is then submitted to Ansys Fluent® calculations;

4. Ansys Fluent® performs the CFD run and returns the objective function values to the D.O.E. code, which, in turn, provides Altair HyperMesh® with the next set of design variables; then, the loop starts again. A dedicated journal file, *Fluent-batch.jou*, is used to launch Ansys Fluent® in batch mode: the journal file can be found in Appendix C.5.

In the present work, each variable was given a normalized deformation range equal to ±1, where +1 corresponds to the upper variable value and -1 to the lower variable value (the true range of each variable was indicated in chapter 7.21).

Once all the calculations have been performed, the sets of the design variables, along with the correspondent values of the nose drag (computed in Ansys Fluent® environment), were transferred to the Matlab® *Student.m* file. *Student.m* file, which supervises the entire Student analysis block, can be found in Appendix C.7.

Actually, the most influential factors on the objective function were determined using a statistical analysis based on the *t-Student* parameter:

$$t = \frac{|x_1 - x_2|}{\sigma} \tag{7.15}$$

being x_1 the mean value of the objective function for the upper set, namely the set of configurations where the investigated variable is given the upper value (i.e., from the mean value, 0, to the upper value, +1); x_2 is the mean value of the objective function for the lower set, (i.e., the set of configurations where the investigated variable is given the lower values, from 0 to -1), and σ the standard deviation, defined as follows:

$$\sigma = \sqrt{\frac{\left(\sum_{i=1}^{n_1} (x_{1i} - \bar{x_1})^2 + \sum_{i=1}^{n_2} (x_{2i} - \bar{x_2})^2\right)(n_1 + n_2)}{(n_1 + n_2 - 2)\, n_1 n_2}} \tag{7.16}$$

where x_{1i} and x_{2i} are the objective functions of the i_{th} configuration of the upper and lower set respectively, and n_1 and n_2 are the cardinalities of the upper and lower configuration sets.

As it is well known, the t-Student parameter assesses whether the means of two groups are statistically different from each other, and it is defined in such a way that the bigger its value, the higher the difference between the two populations, and hence the higher the response variation caused by the corresponding parameter. Therefore, a design variable with an associated high value of t-Student parameter is expected to have a higher influence on the response than a variable with a lower Student parameter. The results of the Student analysis on the tiltrotor nose drag are reported in Figure 7.70.

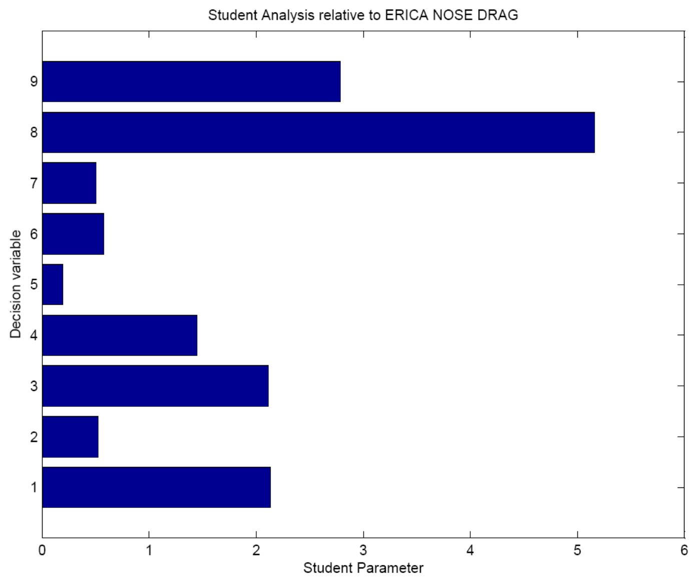


Figure 7.70: Student analysis on the design variables of ERICA nose total drag.

As apparent, the fifth design variable, corresponding to shapes *sh6-sh7* (please see chapter 7.21) is expected to have a small influence on the ERICA nose total drag. Thereby, it was decided to discard it in the following optimization study. Moreover, since the Student analysis showed that the initial selected deformation ranges could yield non-streamlined shapes, it was decided to modify the variation range of each variable. Specifically, the variables were given a non-symmetric range and the allowed deformations were reduced with respect to the initial ones. Finally, different deformation ranges were assigned to each variable, based on the Student analysis results.

The newly defined deformation ranges are summarized in Table 7.30.

Design variable	**Baseline values**	**Lower range**	**Upper range**
sh1-sh2	0	-0.2	1
sh3	0	0	0.5
sh4	0	-0.2	0.5
sh5	0	0	0.2
sh6-sh7	0	0	0
sh8-sh9	0	-0.2	0.5
sh10-sh11	0	-0.2	0.5
sh12-sh13	0	-0.2	0.5
sh14-sh15	0	-0.2	0.8

Table 7.30: Design variables deformation ranges.

7.23 Optimization study: discussion of results

Once the design variables to be retained and the proper deformation ranges were identified through both the D.O.E. analysis and the Student test, the optimization study was performed, using the GeDEAII algorithm.

The basic operations proper of this step of the optimization procedure are illustrated in the lower part of the flow chart reported in Figure 7.45, that is the GeDEAII driven optimization block. Actually, they are very similar to the operations performed during the D.O.E. analysis, with the exception of the block master code (highlighted in yellow in Figure 7.45). In fact, the supervising code in this phase is the GeDEAII algorithm.

The Matlab® function which performs all the synchronization instructions is called *Command-function.m*, and can be found in Appendix C.2. The remainder of the GeDEAII code was left unchanged.

The involved softwares are Ansys Fluent® for the CFD calculations and the objective function computation, Ansys Tgrid® for the mesh generation, and Altair Hypermesh® as the parameterization tool. The total number of individuals per each generation was set to 12, and the initial randomly generated population evolved for 3 generations, which were considered sufficient for a single-objective optimization. The optimization convergence history is depicted in Figure 7.71.

The evolution process resulted in an optimized configuration characterized by

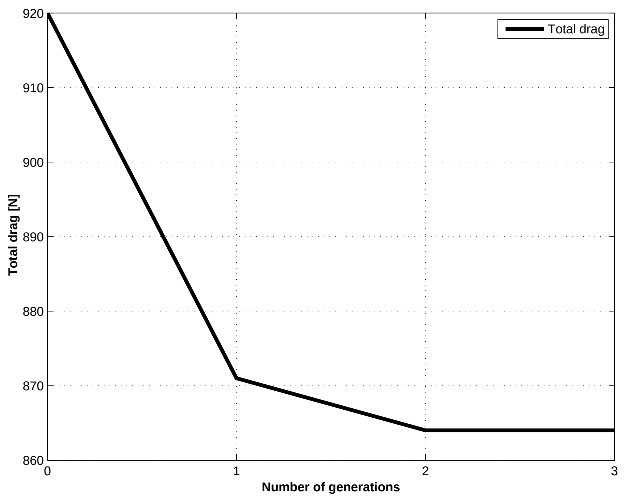
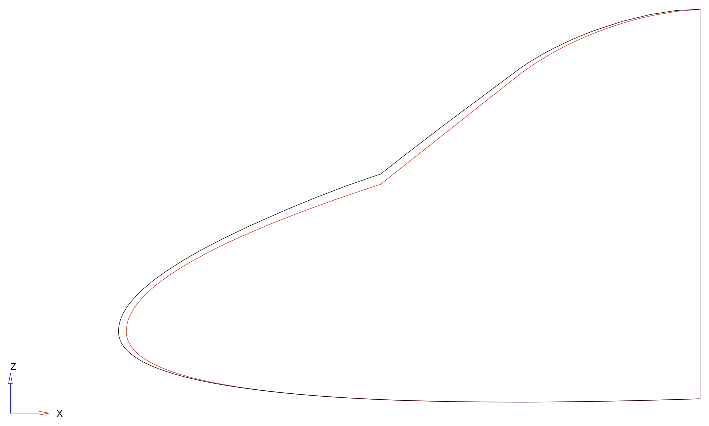


Figure 7.71: Convergence history of the GeDEAII driven optimization.

a drag reduction of nearly 6%, when compared to the baseline configuration. A comparison between the baseline and the optimized geometries is depicted in Figure 7.72, where a sketch of both the configurations on both a top and a longitudinal view is represented.



The objective function values for both the baseline and the optimized nose geometries are reported in Table 7.31, where the total drag, along with the pressure and viscous components are reported. As apparent, the optimization has led to a significant reduction of the pressure drag, while keeping the viscous component substantially unchanged. The design variables values of the optimized configuration are summarized in Table 7.32, where both the absolute and normalized values are reported (one has to remember that a normalized value equal to 1 means that the corresponding variable assumes the upper allowed value of the range). Except for design variables *sh3* and *sh8*-*sh9*, all the other variables were given a value near

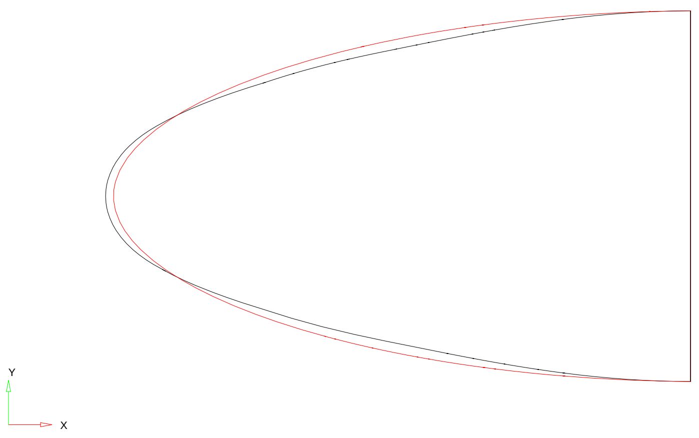


Figure 7.72: Geometrical comparison between the baseline and the optimized nose configurations: longitudinal view (top) and top view (bottom).

	Total nose drag [N]	Pressure component [N]	Viscous component [N]
Baseline	918	253	665
Optimized	862	192	669

Table 7.31: Objective function values for the baseline and optimized nose configurations.

to the maximum allowed in the optimized configuration. In particular, *sh4*, *sh5*, *sh10-sh11*, *sh12-sh13* and *sh14-sh15*, were equal to the upper value in their range. This is due to the fact that the above mentioned variables had a positive effect on the reduction of pressure drag, since they acted in the sense of reducing the cross sections of the nose component. It is also worth considering that parameter

	sh1-sh2	sh3	sh4	sh5	sh6-sh7	sh8-sh9	sh10-sh11	sh12-sh13	sh14-sh15
Absolute	0.9077	0.1172	0.5	0.2	0	0.255	0.5	0.4876	0.8
Normalized	0.9231	0.2344	1	1	0	0.65	1	0.98	1

Table 7.32: Absolute and normalized design variables values of the optimized configuration.

sh3, which governs the nose deformation along the longitudinal x axis, takes an intermediate value between lower and upper limits. This can be explained as long as a compromise between pressure and skin friction drag components is to be achieved when the minimum drag is searched for. The pressure coefficient and skin friction coefficient contours over both the baseline and optimized nose geometries are illustrated in Figure 7.73 and Figure 7.74 respectively. A series of cuts over both the baseline and the optimized geometry were performed along the planes depicted in Figure 7.75 in order to better analyze the pressure and viscous drag

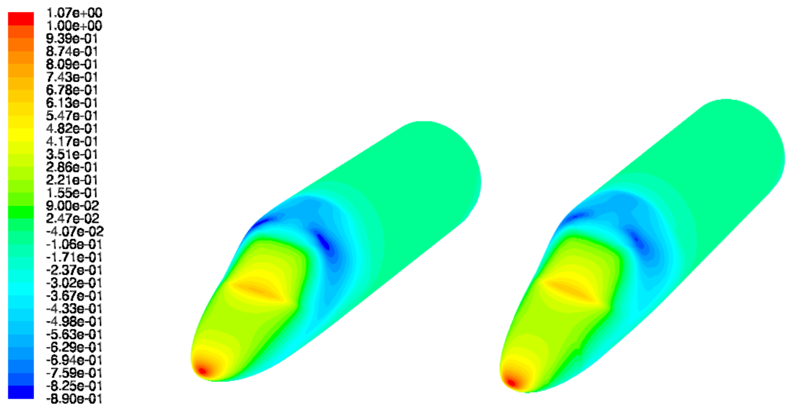


Figure 7.73: Pressure coefficient contours over the baseline configuration (left) and the optimized configuration (right).

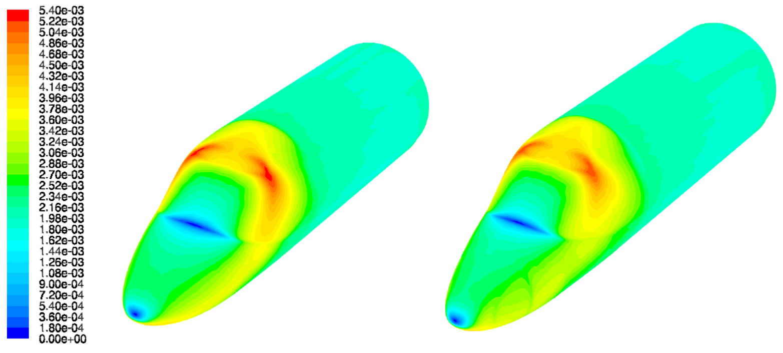


Figure 7.74: Skin friction coefficient contours over baseline configuration (left) and the optimized configuration (right).

components on the fuselage nose and identify the reasons of improvement of the objective function featured by the optimized configuration. To this purpose, the local values of both pressure and viscous drag of the optimal solution over the above mentioned planes are illustrated in Figure 32 and Figure 33 respectively, and compared with the baseline geometry. These values were obtained using the following formulae:

$$D_p(x) = pDA_x \quad (local\ pressure\ drag) \tag{7.17}$$

$$D_v(x) = \tau_x \left(DA_y + DA_z\right) \quad (local\ skin-friction\ drag) \tag{7.18}$$

where p is the local value of pressure, DA_x the local surface Area projection normal to the x-axis, τ is the local shear stress (resolved along its x-axis component), and DA_y and DA_z the local surface Area projection normal to the y-axis and z-axis, respectively. As apparent, the zones where pressure and viscous drag are generated can be deduced by analyzing the behaviour of D_p and D_v. From Figure

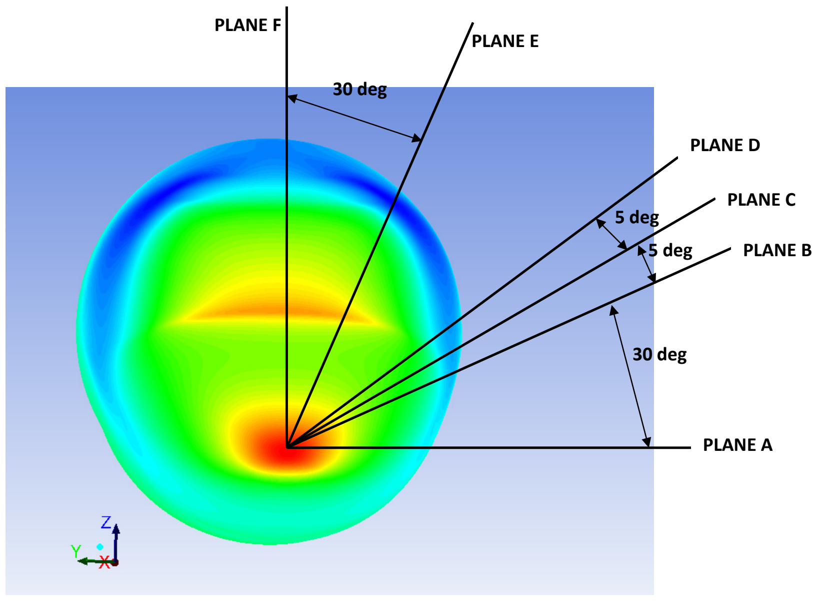


Figure 7.75: Visualization of the planes selected for pressure and viscous drag analysis.

7.76, it can be argued that the main difference between original and optimized configuration, in terms of pressure drag, originates close to planes "C" and "D", where smaller values in the pressure drag can be found between x=1.5 m and x=3 m on the pressure side of the nose. On the contrary, in plane "F" the optimized solution exhibits slightly higher values of the pressure drag between x=1.5 m and x=3 m. However this does not affect the overall pressure drag, which is in fact reduced. On the other hand, from Figure 7.77, it can be deduced that the original and optimized configuration are almost equivalent in terms of viscous drag. In fact, the curves over the planes "A" and "E" are almost coincident, while on plane "F" the optimized solution exhibits higher values of the viscous drag between x=1.5 m

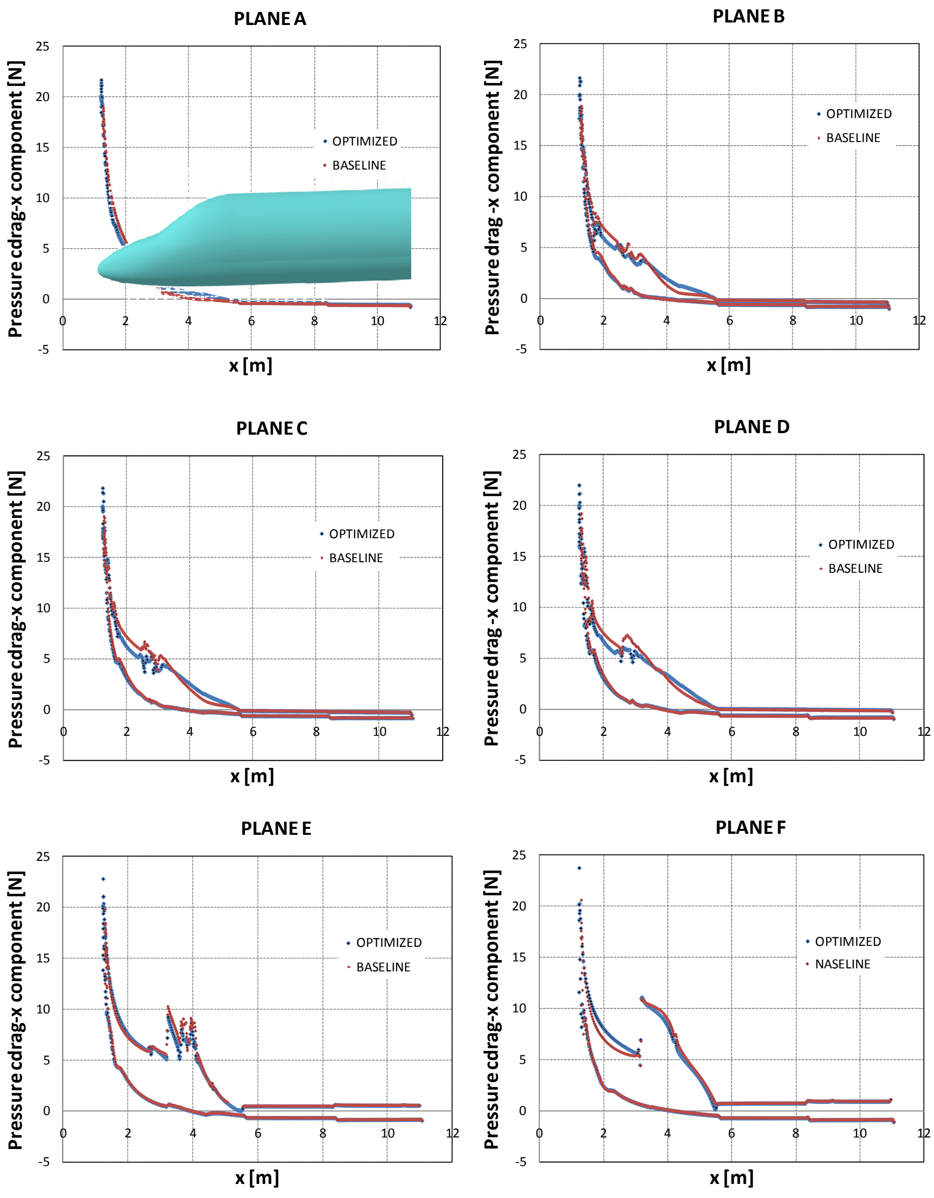


Figure 7.76: Pressure drag component over the selected planes: comparison of baseline and optimized configurations.

and x=3 m. However, over planes "B", "C" and "D" the viscous component of the optimized solution is smaller than the baseline between x=1.5 m and x=4 m and slightly higher between x=4m and x=5.5 m. This results in an overall compensation of the viscous drag, which is in fact nearly unchanged from the baseline to the optimized solution.

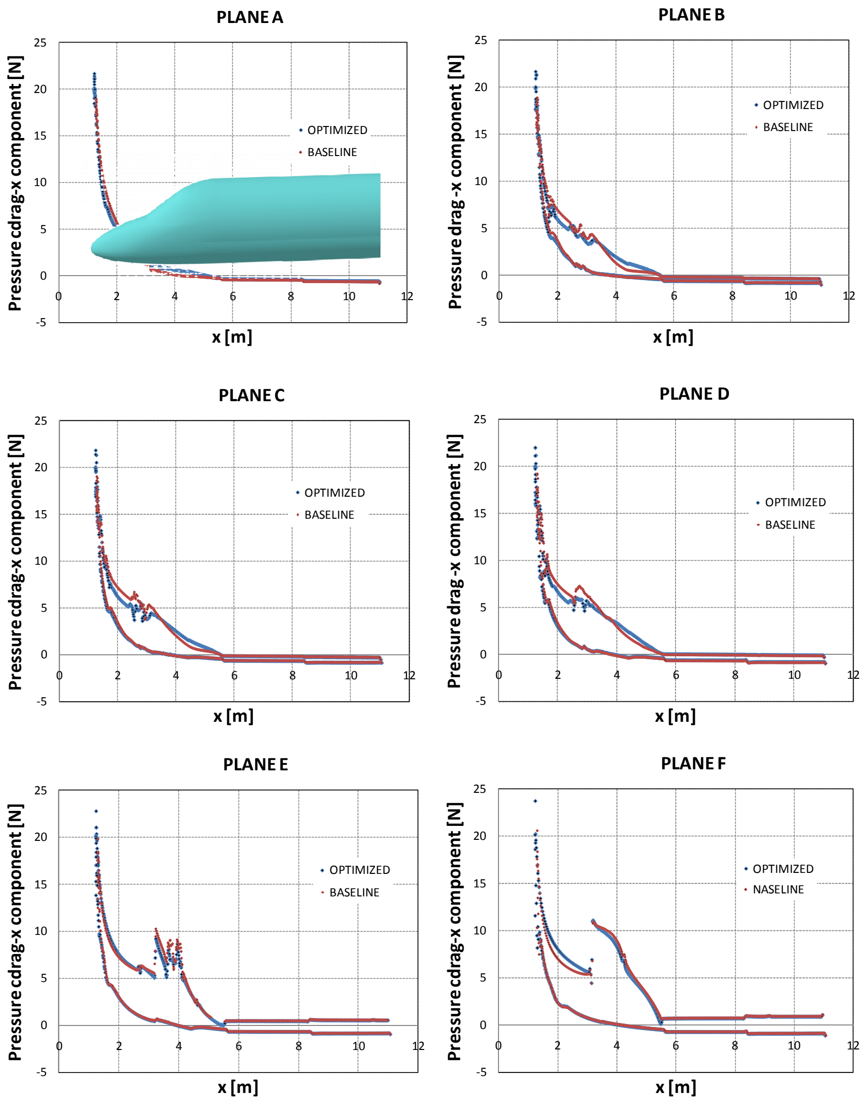


Figure 7.77: Viscous drag component over the selected planes: comparison of baseline and optimized configurations.

7.24 Reverse Engineering Procedure in CATIA® V5

From the optimization run the values of the design variables that minimize the nose drag were derived. Those values were introduced within Altair Hypermesh® in order to obtain the morphed mesh of the optimal geometry. Then, the mesh was exported in *stl* format: specifically, only the modified components were exported, being the cylindrical portion of the fuselage and the tail cone kept unchanged.

Moreover, since the geometry is symmetrical with respect to the zx plane, only half of the model was reconstructed. The reverse engineering process is qualitatively described in the following.

The file **.stl* created in Altair Hypermesh® was read within CATIA® V5 by using the Import function under the *Digitalize Shape Editor* module. The reader is referred to Figure 7.49 for the orientation of the coordinate axes used in the following.

Several planes normal to the x-axis needed to be created in order to extract the pertinent geometry sections, their number depending on the complexity of the original model. In this case, four planes were generated, whose coordinates in the x direction are summarized in Table 7.33. In particular, plane 5 represents the

	Plane 1	**Plane 2**	**Plane 3**	**Plane 4**	**Plane 5**	**Plane 6**	**Plane 7**	**Plane 8**	**Plane 9**
Absolute origin	1420	1700	2100	2700	3165	3500	4265	5065	5500

Table 7.33: x-coordinate of the generated planes normal to x-axis.

discontinuity interface between the nose and the windscreen. In addition to planes normal to the x direction, three more planes were created:

- Plane 10: it derives from a rotation of -1.97 deg (equal to the fuselage incidence) of the xy plane around the y-axis and passes through the point at the top of the nose;
- Plane 11: it is a clockwise rotation of 20° of plane 10 around the x-axis;
- Plane 12: it is a clockwise rotation of 45° of plane 10 around the x-axis;

First, the *Planar Section* tool was used in order to extract sections of the *stl* model over the above mentioned planes. Then, using the *Curve from Scan* tool, the pertinent curves were generated with a defined tolerance and with the least possible number of segments of the least possible order. The tolerance value must be accurately selected, since a too low value may detrimentally affect the quality of the generated curves and consequently of the derived surfaces. During curve creation, the user may select particularly meaningful points, for instance the point at the top of the nose or the intersection between plane 5 and the *zx* plane. In the following, the curves generated from sections 1 to 9 will be referred to as *group 1*, while those generated from planes 10 to 12 will be referred to as *group 2*.

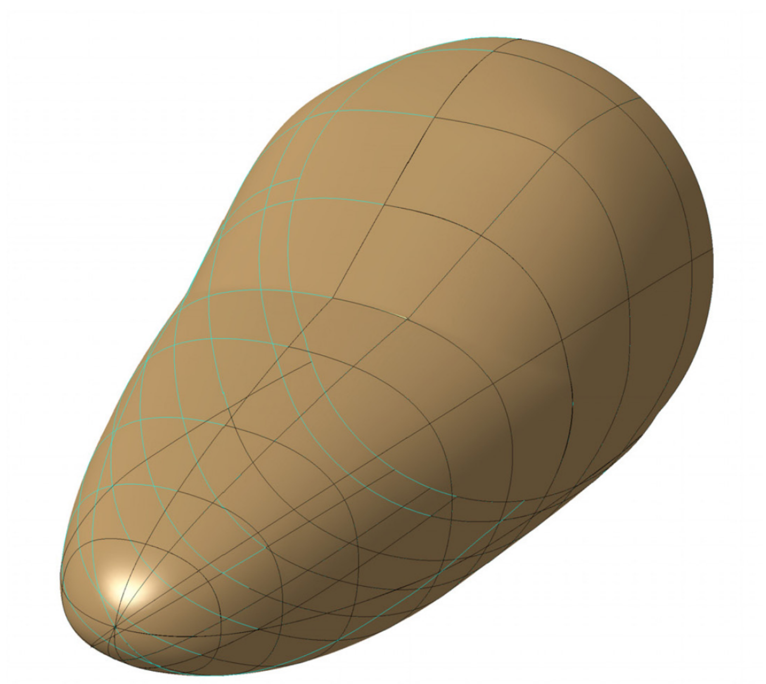


Figure 7.78: The reverse engineering procedure: cyan curves represent the *group 1*, while black curves represent the *group 2*.

The *group 1* curves were then split using planes zx, 10, 11 and 12: this operation is represented in Figure 7.78.

In addition to these operations, the region between the nose and the windscreen was handled in a peculiar way.

As a matter of fact, in order to respect the original variation of curvature between the nose and the windscreen surfaces, some points were added to these curves. A particular of this operation regarding curve generated using plane 11 is depicted in Figure 7.79. A spline was created using the above mentioned points, that is represented in Figure 7.80 (colored in red), along with the other construction lines. The tool *Multi-Section Surface* in the *Generative Shape Design* module was finally exploited to create surfaces from the previously generated curves. As mentioned before, only half of the model was recreated, therefore the surfaces in the symmetry zone have to be tangent to the y-axis. Triangular parts near the top of the nose were filled in such a way to guarantee tangency to the previously created surfaces. The complete model was finally obtained using symmetry. The upper and front views of the optimized geometry are depicted in Figure 7.81. Once the optimized model has been reconstructed, in order to evaluate the geometrical differences between the optimized and the baseline configurations, a comparison was performed. It consisted in an overlap of the two CAD models. The superimposition of the baseline and the optimized CAD models is depicted in Figure 7.82.

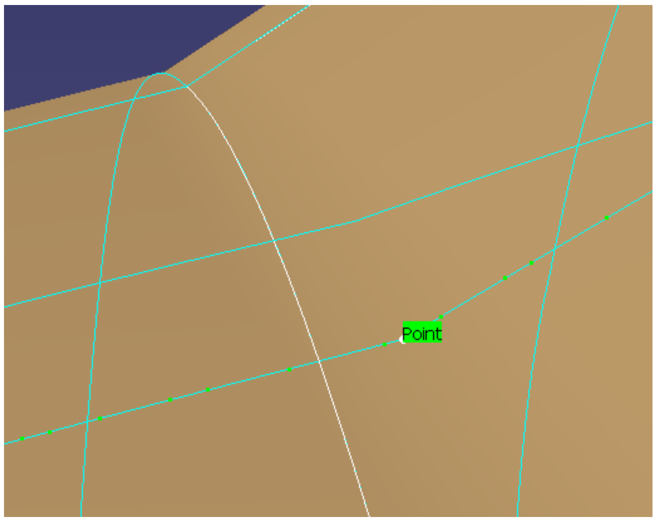


Figure 7.79: The point introduced during the curve creation in order to respect the original variation of the curvature.

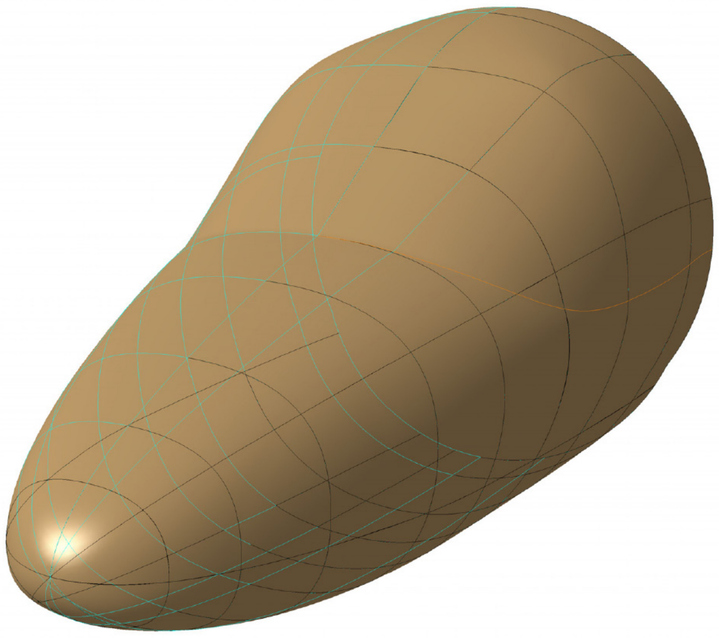


Figure 7.80: The cockpit construction lines.

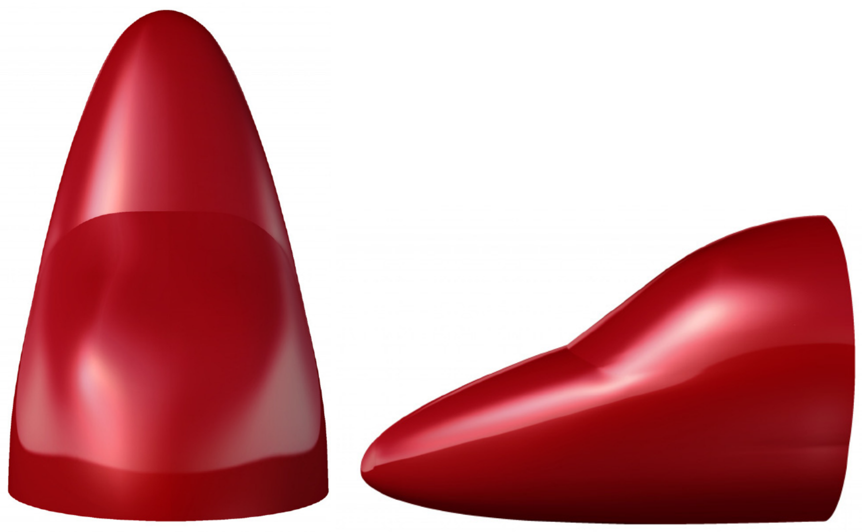


Figure 7.81: The final optimized nose geometry.

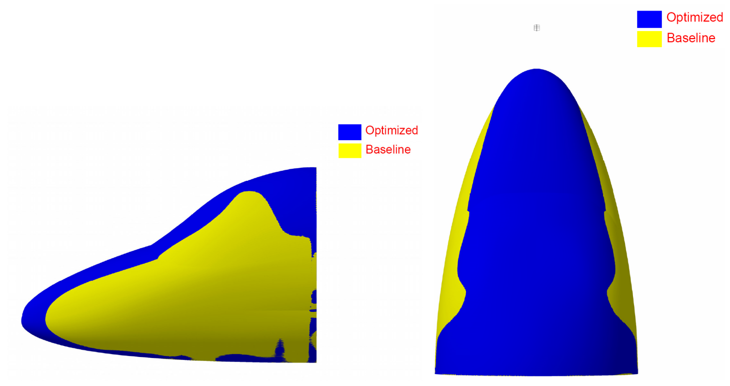


Figure 7.82: Superimposition of the baseline and the optimized CAD models.

Part III

Test case C: Aerodynamic shape optimization of the frontal region of the ERICA tiltrotor using OpenFOAM® as the CFD solver

7.25 Introduction

The present work deals with the ERICA tiltrotor nose shape optimization aimed at reducing its baseline aerodynamic drag. Unlike the optimization work presented in Part II, now a different CAD and mesh model will be addressed.

OpenFOAM® CFD open-source code was exploited to calculate aerodynamic behavior of the model that underwent the optimization process.

Moreover, the visibility constraints were strictly taken into account via a procedure described in Appendix B.

Several steps were performed in order to obtain reliable results while reducing the overall optimization time as much as possible. First, the baseline model was built up. To do this, the superficial mesh was first built in CATIA® environment, and the parameterization performed with Altair Hypermesh®. Finally, the related *.hm* file was built. The latter was subsequently managed in batch-mode via a script similar to that presented in Appendix C.3.

The second step consisted in the nose aerodynamic shape optimization driven by GeDEAII presented in Chapter 2.

As it was said above, this optimization test case featured an open-source CFD code, that is OpenFOAM®.

Both the features and the framework of this powerful solver are presented in Appendix D so as to make the understanding of this test case simpler.

7.26 Optimization Procedure

In this chapter, the whole sequence of operations necessary to carry out the optimization study is described. As already mentioned, the optimization procedure was not performed in a single step, but rather multiple steps were required to obtain reliable results, while reducing the overall optimization time as much as possible.

The whole sequence of operations carried out to address this work is shown in Figure 7.83. It is worth mentioning that each block was carried out independently from the others. The upper portion of the flow chart shows the set of preliminary operations that need to be accomplished before starting the successive automatic loops.

The procedure started from the elaboration of a baseline geometry CAD model of the tiltrotor fuselage provided by AgustaWestland in CATIA® format. The following steps focused on the construction of the baseline CFD model: first of all, the surface mesh was generated using CATIA® surface meshing tool, then the volume mesh was created by means of Ansys TGrid® and finally the baseline CFD case was set up with the open-source CFD code OpenFOAM which is a new CFD open-source code adopted at AgustaWestland for external aerodynamic characterization of helicopter and tiltrotor fuselage components.

Once the baseline CFD solution was available, the geometry parameterization was carried out. This operation is of outstanding importance in the optimization

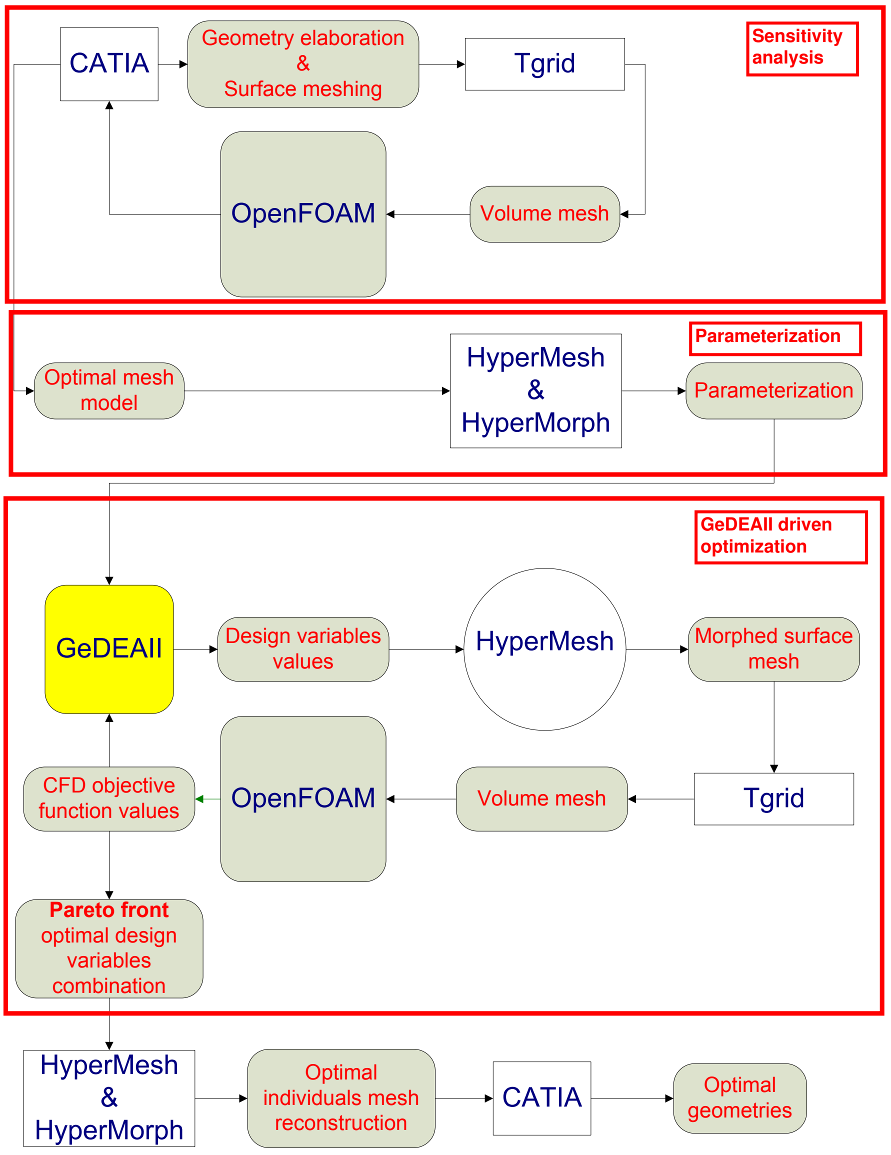


Figure 7.83: Flow chart of the complete optimization procedure accommodating Open-FOAM as the CFD colver.

process and it can be performed using various techniques. For the scope of the present work, the commercial software HyperMesh® was chosen as the parameterization tool, due to the really strong and versatile capabilities of the morphing tool HyperMorph®.

Here some words are spent about the followed morphing strategy in this work. In details, it was decided to parameterize the surface mesh only, and to re-mesh every time the volume mesh, for the reasons already explained in section 7.14.

After having identified the independent variables to be taken into account, the optimization process could be addressed, which was managed by GeDEAII as the master program.

The lower part of the flow chart in Figure 7.83 describes the results extraction and elaboration. The output of the automatic optimization loop is the optimal combination of the design variables of the individuals lying on the Pareto frontier. At this point, the designer can identify the individuals of interest among the multiple optimal solutions of the Pareto front and the meshed geometry of each selected individual can be reconstructed using HyperMorph®.

For the sake of brevity, the reverse engineering process, which allows obtaining the deformed surface starting from the mesh, is omitted in this test case. Nevertheless, it can be performed following the tips already given within sections 7.12 and 7.24.

The entire optimization loop was run under Linux/UNIX environment.

7.27 The object of the optimization

The scope of this work consisted in the optimization of the ERICA nose, whose full-scale CAD wind-tunnel model, made up of seven elements is depicted in Figures 7.84-7.90. Only the left sides of the CAD model are depicted in the following figures, since the simulations took into account only this part of the ERICA model. This was possible thanks to the symmetry of the investigated flow field.

- *nose*, the ERICA part that underwent the optimization process, visible in Figure 7.84;

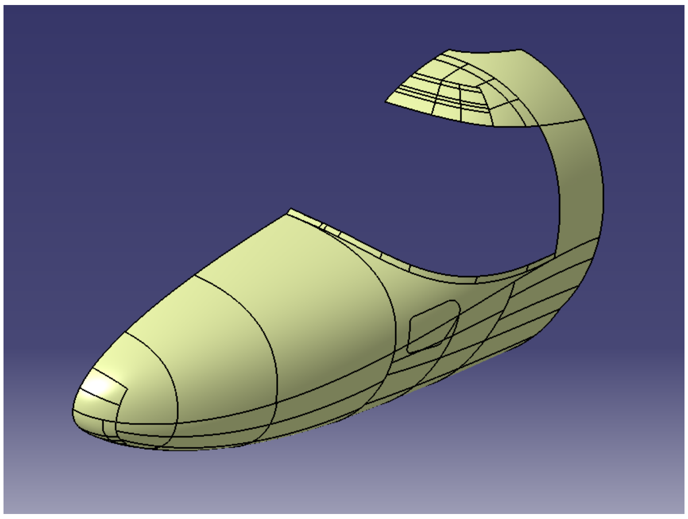


Figure 7.84: ERICA *nose* geometry in CATIA V5.

- *windscreen*, visible in Figure 7.85;
- *fuselage*, visible in Figure 7.86;
- *wf-fearing*, visible in Figure 7.87;

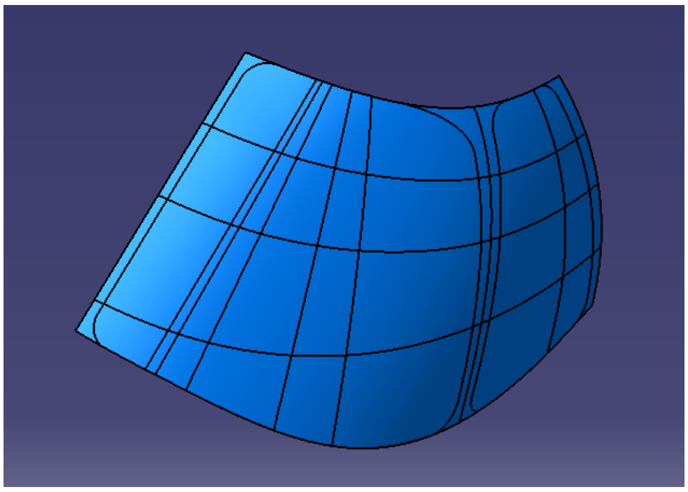


Figure 7.85: ERICA *windscreen* geometry in CATIA V5.

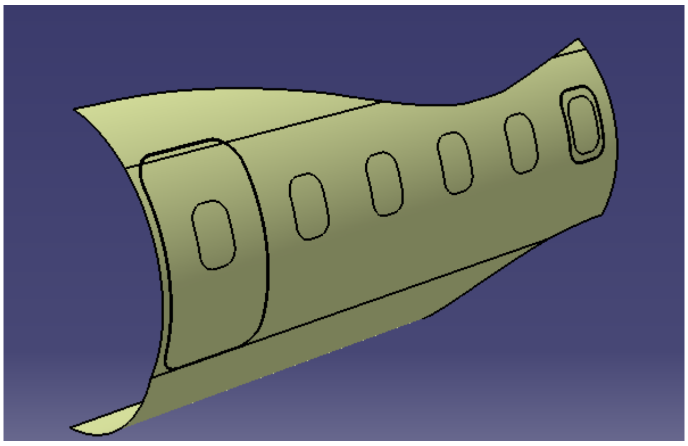


Figure 7.86: ERICA *fuselage* geometry in CATIA V5.

- *wing-sx*, visible in Figure 7.88;
- *sponson-sx*, visible in Figure 7.89;
- *tail*, visible in Figure 7.90;

To this purpose, a CFD-based optimization procedure was implemented, which was introduced in chapter 7.26, and will be described in detail in the following chapters. The entire model was then inserted into a virtual wind tunnel, which is made up of six elements: *inlet*, *outlet*, *symmetry*, *sx*, *up* e *down*. It is visibile in Figure 7.91.

Box features a length of 192.8 meters, it is 58.8 [m] deep, and 117.2 [m] high.

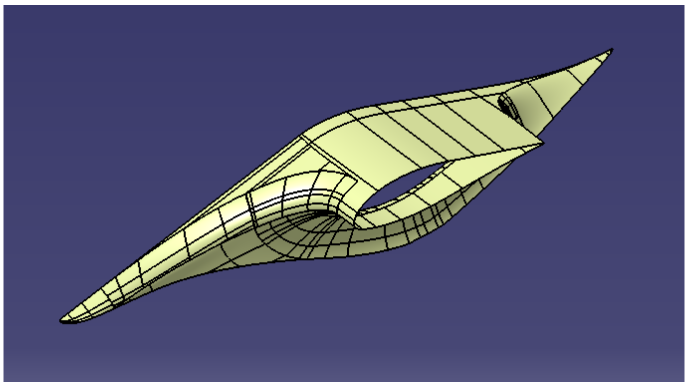


Figure 7.87: ERICA *wf-fearing* geometry in CATIA V5.

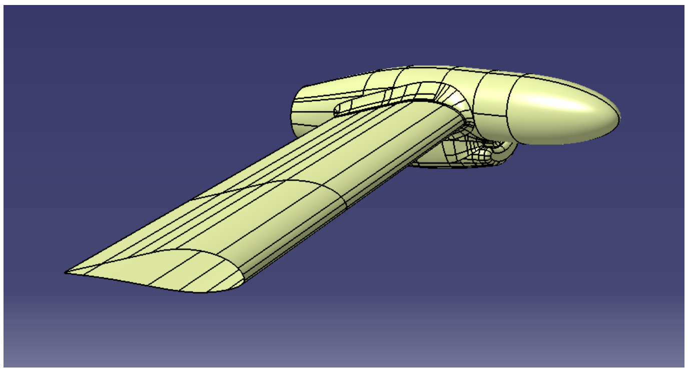


Figure 7.88: ERICA *wing-sx* geometry in CATIA V5.

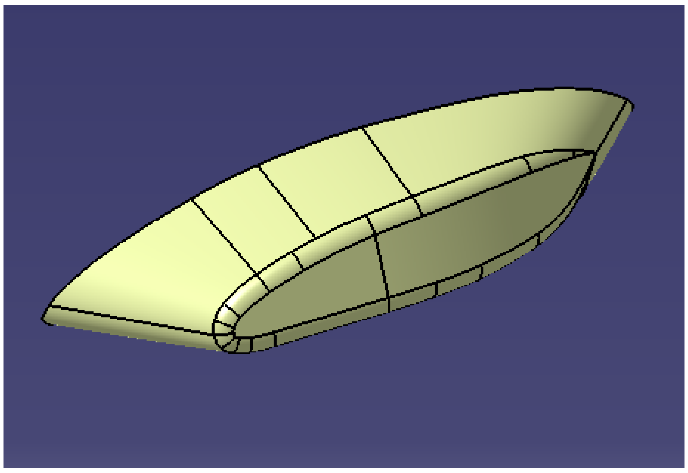


Figure 7.89: ERICA *sponson-sx* geometry in CATIA V5.

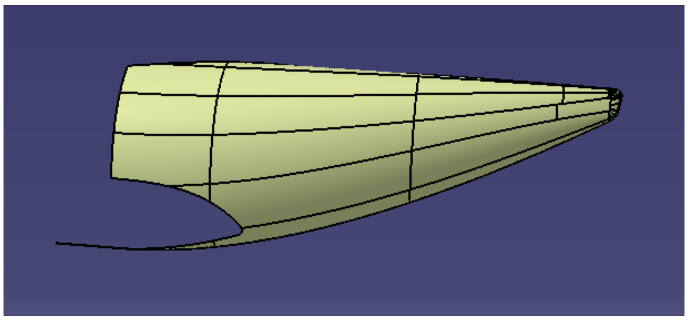


Figure 7.90: ERICA *tail* geometry in CATIA V5.

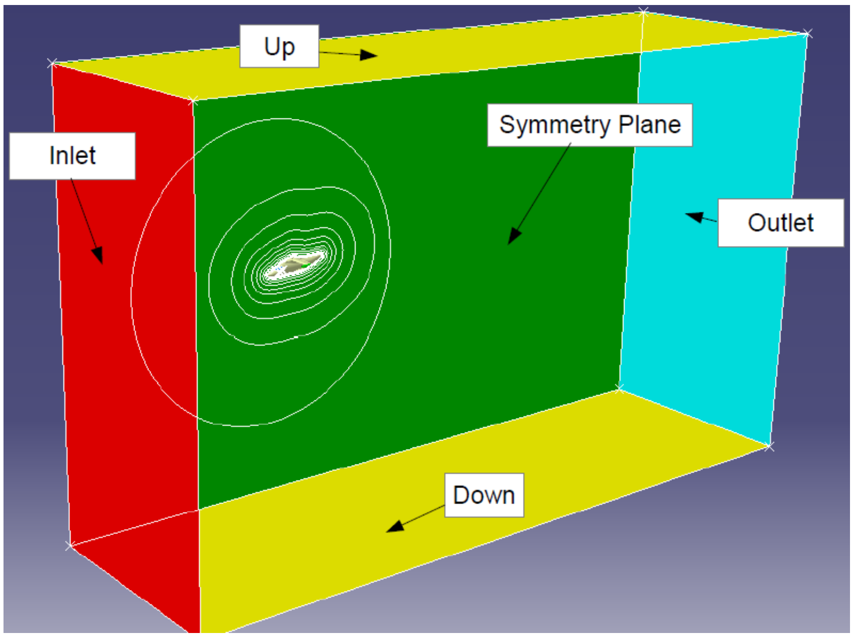


Figure 7.91: Wind tunnel cad model.

7.28 Mesh generation

As far as the mesh generation is concerned, the softwares used to carry out the meshing operations were respectively:

1. CATIA® surface meshing tool for the surface mesh generation;

2. Ansys Tgrid® for the volume mesh generation.

7.28.1 Superficial mesh

The CATIA® surface meshing tool can be used to mesh the geometry surfaces directly from the CATIA® format, so no import/export operations are necessary to accomplish this task. Moreover, the surface mesh obtained from the CATIA® meshing tool is entirely interfaced with the CATIA® geometry. This characteristic is important in an industrial context, because a modification of the baseline geometry can be directly transformed into a correspondent surface mesh modification simply by updating the mesh.

The surface mesh was more refined over the tiltrotor nose, which was the object of optimization, whereas the superficial grid over the cylindrical portion of the fuselage was slightly coarser; finally, the tail cone surface mesh was the coarsest one. The choice of adopting different mesh sizes was motivated by the attempt to contain the total amount of surface elements (and consequently the number of 3D cells) as much as possible.

Surface mesh is visible in Figure 7.92 where the entire wind tunnel box is visible. Moreover, in Figure 7.93 *nose* and *windscreen* superficial mesh models are depicted.

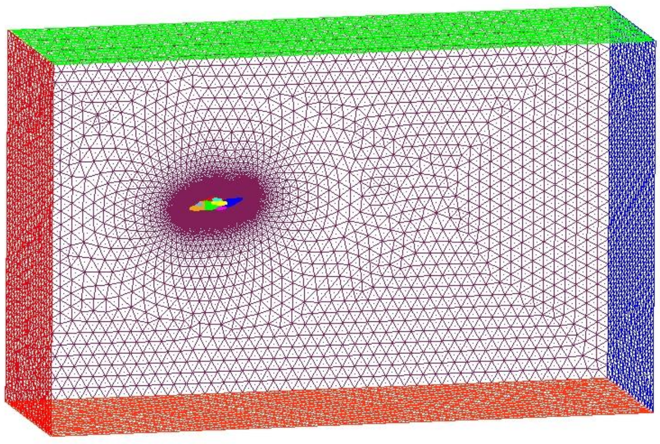


Figure 7.92: Superficial mesh model: wind tunnel component.

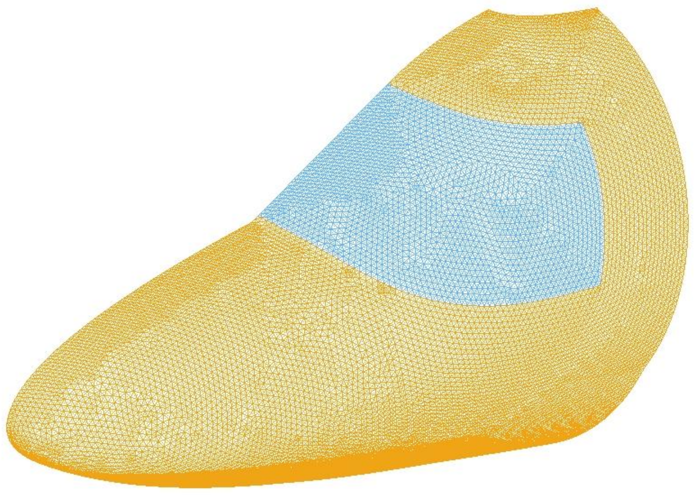


Figure 7.93: Superficial mesh model: *nose* component.

Some words are worth spending as regards the quality criteria accomplished during superficial mesh generation. Actually, in order to achieve a good mesh quality it was necessary to define and set proper quality parameters before starting the meshing operations; these parameters were then checked during the mesh generation. AgustaWestland quality requirements can be met by imposing proper limits on *Skewness* and *Aspect-Ratio* within CATIA®.

These parameters can be defined as follows:

- **Skewness**: it is a quantitative measure of the element distortion with respect to the ideal element shape (equilateral triangle for triangular surface mesh).

  Skewness is calculated within CATIA® in the following way:

$$Q = 1 - \frac{S}{S'} \tag{7.19}$$

  where Q is the Skewness, S is the ideal element surface, and S' is the actual element surface.

- **Aspect-Ratio**: it is the ratio between the maximum and minimum length among the sides of the element; for instance, the Aspect-Ratio AR of a triangular element of sides l_1, l_2, l_3 is defined as:

$$AR = \frac{max\,(l_1, l_2, l_3)}{min\,(l_1, l_2, l_3)} \tag{7.20}$$

The parameters limits specified for the ERICA model superficial mesh are summarized in Table 7.34. The mesh statistics of the finally selected superficial mesh with

	Best value	Poor at	Bad at	Worst value
Skewness	0	0.3	0.5	1
Aspect-Ratio	1	1.5	3	$1 * 10^6$

Table 7.34: Quality parameters limits specified for ERICA surface mesh.

respect to the above mentioned criteria are summarized in Table 7.35 and Figure 7.94.

	Optimal elements	Poor elements	Bad elements	Worst element	Mean value
Skewness	238018 (99.95 %)	114 (0.00 %)	0 (0.00 %)	0.565	0.038
Aspect-Ratio	238132 (100 %)	0 (0.00 %)	0 (0.00 %)	2.34	1.157

Table 7.35: Mesh statistic for the finally selected ERICA superficial grid.

As apparent, AgustaWestland quality criteria were fulfilled with wide margins. Moreover, these values are on the line with those presented in section 7.17.1.

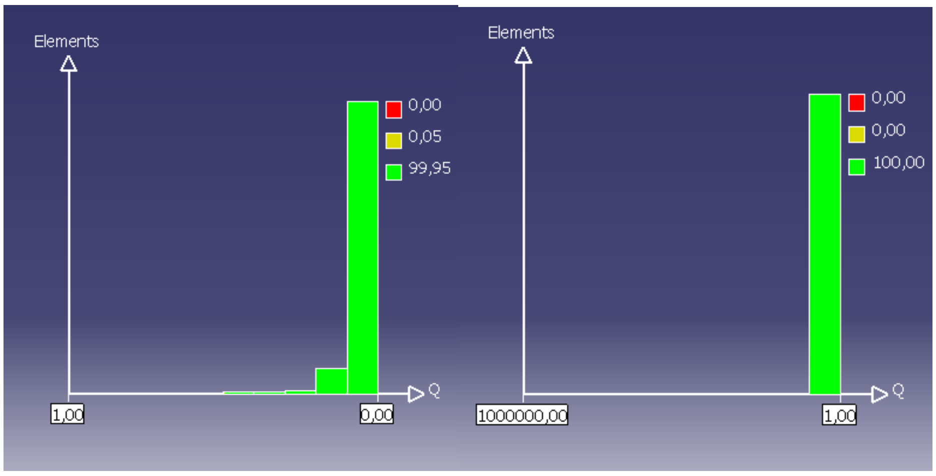


Figure 7.94: Histograms representing ERICA surface mesh quality statistics: *Skewness* (on the left); *Aspect-Ratio* (on the right).

7.28.2 Volume grid

Starting from the surface mesh described above, the volume mesh was generated by means of the CFD meshing tool Ansys Tgrid®. Actually, the surface mesh created within CATIA® can be saved in a Nastran compatible format (.dat) which can be easily read by Tgrid®.

In Tgrid® environment, the volume between the tiltrotor surface and the wind-tunnel walls was filled with tetrahedral elements, while the boundary layer region was modelled using prismatic cells. In Table 7.36, the Tgrid® settings selected to generate the internal volume mesh are presented.

Components with boundary layer	nose, fuselage, fearing, tail, wing, sponson
Components without boundary layer	inlet, outlet, symmetry, up, down,sx
B.L. Offset method	Uniform
B.L. Growth method	Geometric
B.L. Number of layer	10
B.L. First height	0.15
B.L. Growth rate	1.2
Tri/Tet Improve surface mesh option	Enabled
Tri/Tet Refinment method	Adv/front
Tri/Tet Cell size function	Geometric
Tri/Tet Growth rate	1.2
Tri/Tet Refinement regions	yes (*box-fuselage*, *box-wing*)

Table 7.36: Ansys Tgrid® settings for volume mesh generation.

Ten prismatic layers were built over the tiltrotor fuselage in order to ensure that the simulated boundary layer was entirely contained within the prismatic layers over the whole fuselage, with the exception of regions of separated flow. Moreover, a height of 0.15 mm was given to the first prismatic layer in order to meet the y+ requirements for the wall function calculation during the CFD simulation ([7]).

An important aspect that has to be pointed out is the use of local refinement regions, which allowed a local improvement of the volume mesh around the ERICA model, in particular near the nose region, while a coarser tetrahedral mesh could be used in the remaining internal regions.

As a result, a really good mesh quality was obtained in the regions of interest (that is, the nose and the cylindrical portion of the fuselage), with the minimum number of 3D elements. A longitudinal view of the whole volume mesh is depicted in Figure 7.95, while Figure 7.96 illustrates a close-up of the mesh near the tiltrotor surface, where the refinement regions around the fuselage can be clearly appreciated. Finally, a particular of the boundary layer over the canopy is reported

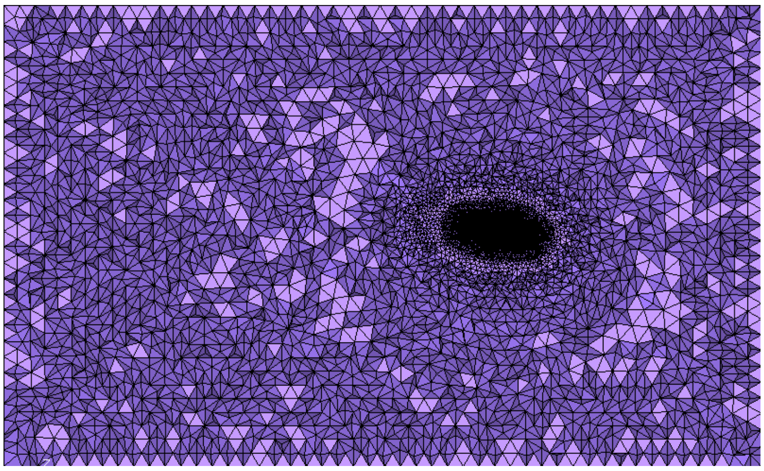


Figure 7.95: Longitudinal view of the whole volumetric mesh around the tiltrotor fuselage.

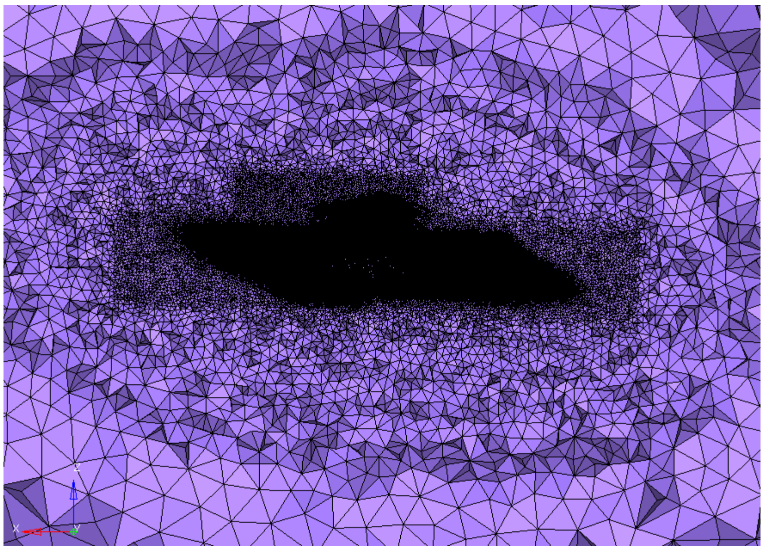


Figure 7.96: Close-up of the volume mesh near the fuselage.

in Figure 7.97.

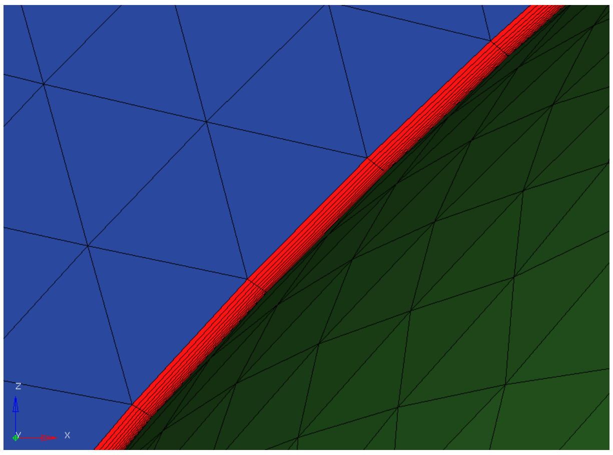


Figure 7.97: Close-up of the volume mesh: boundary layer over the canopy.

7.29 Fluid-dynamic model set up

A pressure-based solver type with absolute velocity formulation and steady approach was adopted for the tiltrotor simulations. The $\kappa - \omega$ SST model was selected for turbulence handling. The air was treated as an ideal gas having constant specific heats, which automatically enables the equation energy resolution. This made it possible to include the compressibility effects in the numerical simulations: in fact, as it can be deduced from Table 7.40, the flight condition selected for optimization features a moderately high Mach number.

In Table 7.37, the air properties assigned for the ERICA nose optimization are summarized. These parameters were stored into the *thermophysicalProperties* file presented in Appendix D.6.3. The OpenFOAM format of this file is presented in the following:

```
thermoType hPsiThermo<pureMixture<sutherlandTransport
<specieThermo<hConstThermo<perfectGas>>>>>;
mixture
{
specie
{
nMoles 1;
molWeight 28.9;
}
thermodynamics
{
Cp 1007;
Hf 0;
}
transport
{
```

```
As 1.4792e-06; Ts 116;
}
}
```

As far as the solution algorithm is concerned, a SIMPLEC scheme was adopted, which solves the pressure and moment equations separately. In details, the *rhoSimplecFoam* solver was addressed for this work purposes, which is a steady-state SIMPLEC solver for laminar or turbulent RANS flow of compressible fluids. It showed high-level performance in terms of accuracy and robustness.

The CFD calculations started with 1*st* order discretization schemes. The selected 1*st* order discretization schemes are summarized in Table 7.38, which is a excerpt of the *fvSchemes* and *fvSolution* files used during OpenFOAM simulation. Relaxation factors were initially given low values, in order to make more stable the iterative search of the solution.

Gas thermo-dynamic model	Ideal Gas
Cp Specific Heat $[J/kg \cdot K]$	Constant: 1007
Thermal conductivity $[W/m \cdot K]$	Constant: 0.0242
Viscosity variation law	Sutherland: two coefficients methods
μ_0 *Reference viscosity* $[kg/m \cdot s]$	$1.4792 \cdot e^{-6}$
T_0 *Reference temperature* $[K]$	116
S Effective temperature $[K]$	110.56

Table 7.37: Air properties for ERICA nose optimization.

After 500 iterations, discretization schemes were set to a $1st - 2nd$ order, in order to increase the accuracy of the solution. Moreover, relaxation factors slightly increased to speed up the simulations.

7.29.1 Boundary and operating conditions

The boundary conditions for the optimization study were selected according to the operating flight conditions reported in Table 7.40. In particular, *turbulentIntensityKineticEnergyInlet* OpenFOAM boundary condition allows to calculate the turbulent kinetic energy by means of the following relation:

$$\kappa = \frac{3}{2} u_{avg} I^2 \tag{7.21}$$

that is, it is calculated from the intensity provided as a fraction of the mean velocity. A value equal to 5% was chosen for the turbulent intensity I.

[18]This boundary condition is not present in the official OpenFOAM release. As a matter of fact, it is a home-made boundary condition which allows to calculate the turbulent frequency at the inlet boundary by using the turbulent viscosity ratio, $\frac{\mu_t}{\mu}$, which is the ratio between the turbulent viscosity, μ_t, and the molecular dynamic viscosity, μ. It proved to be an effective and robust boundary condition.

gradSchemes	
{	
default	Gauss linear;
}	
divSchemes	
default	Gauss upwind;
div((muEff*dev2(T(grad(U)))))	Gauss linear;
}	
interpolationSchemes	
{	
default	linear;
div(U,p)	linear phi;
UD	linear phid;
}	
relaxationFactors	
{	
p	0.3;
rho	0.01;
U	0.7;
h	0.7;
k	0.5;
omega	0.5;
}	
relaxationFactors0	
{	
p	0.3;
rho	0.01;
U	0.7;
h	0.7;
k	0.5;
omega	0.5;
}	

Table 7.38: 1*st* order discretization schemes and solver settings.

gradSchemes	
{	
default	cellLimited Gauss linear 1;
grad(p)	Gauss linear;
grad(U)	Gauss linear;
grad(h)	Gauss linear;
}	
divSchemes	
{	
div(phi,U)	Gauss linearUpwind grad(U);
div((muEff*dev2(T(grad(U)))))	Gauss linear;
div(U,p)	Gauss linearUpwind grad(p);
div(phi,h)	Gauss linearUpwind grad(h);
div(phi,omega)	Gauss linearUpwind grad(omega);
div(phi,k)	Gauss linearUpwind grad(k);
}	
relaxationFactors	
{	
p	1;
rho	1;
U	0.9;
h	0.95;
k	0.9;
omega	0.9;
}	
relaxationFactors0	
{	
p	0.3;
rho	0.1;
U	0.7;
h	0.7;
k	0.7;
omega	0.7;
}	

Table 7.39: $1st - 2nd$ order discretization schemes and solver settings.

		OpenFOAM boundary condition	
Variable	Value	Inlet	Outlet
U	154 m/s	pressureInletVelocity	pressureInletOutletVelocity
p	38251 Pa	totalPressure	fixedValue
T	239.4 K	totalTemperature	inletOutletTotalTemperature
k	88.935 $\frac{m^2}{s^2}$	turbulentIntensityKineticEnergyInlet	inletOutlet
Ω	320693 s^{-1}	turbulentEVRFrequencyInlet[18]	inletOutlet

Table 7.40: Operating conditions and OpenFOAM boundary conditions settings.

As regards the fuselage walls, the following boundary conditions were set. For a deeper insight into the OpenFOAM boundary condition settings at the walls, readers are referred to Appendix D.5.

Variabile	Fuselage, nose, wf-fearing, tail, sponson-sx, wing-sx
U	fixedValue
p	zeroGradient
T	zeroGradient
k	fixedValue
Ω	compressible::omegaWallFunction

Table 7.41: Boundary conditions set onto fuselage walls.

7.30 Fluid dynamic characterization of the baseline geometry

Results reported in this section refer to a flow configuration featuring a fuselage incidence of +0.159°. In order to reach an accurate solution, 1500 iterations were needed. During the first 800 iterations, first order discretization schemes were exploited. Residuals history is depicted in Figures 7.98 and 7.99. After that, an automatic switch to first-second order schemes was performed.

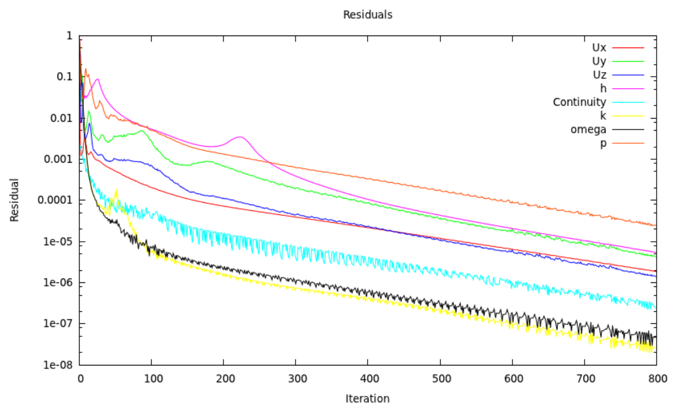


Figure 7.98: First order residuals with low under relaxation factors.

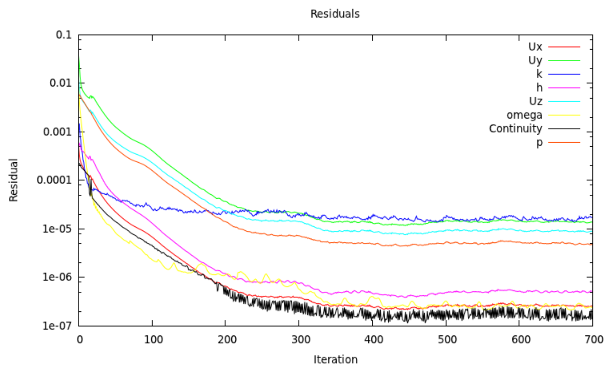


Figure 7.99: First-second order residuals with high under relaxation factors.

In Table 7.42, the aerodynamic forces acting on the whole ERICA model are presented. Their viscous (F_v) and pressure components (F_p) are reported as well.

Forces	[N]	Forces	[N]
F_{px}	7225.56	F_x	10001.98
F_{vx}	2776.42		
F_{pz}	85710.66	F_z	85831,05
F_{vz}	120.39		

Tabella 7.42: Aerodynamic forces acting onto the entire baseline model.

Convergence behaviours concerning the lift coefficient CL, the drag coefficient CD and the moment coefficient CM are shown in Figures 7.100, 7.101 e 7.102, whereas their values are reported in Table 7.43.

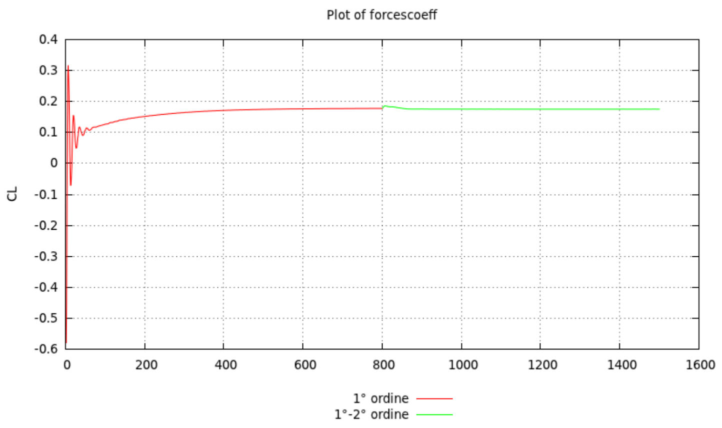


Figura 7.100: Convergence behaviour of CL.

In Figures 7.103 and 7.104 pressure coefficient C_p and skin friction coefficient C_f are shown respectively, with a particular emphasis on the nose region.

In Figures 7.105 and 7.106 *streamlines* are shown on a longitudinal plane.

Friction lines (shear stress magnitude) are shown in Figure 7.107 over the nose surface.

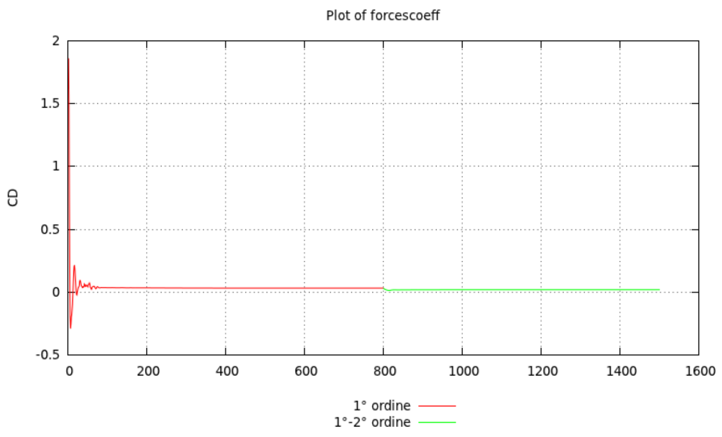


Figura 7.101: Convergence behaviour of CD.

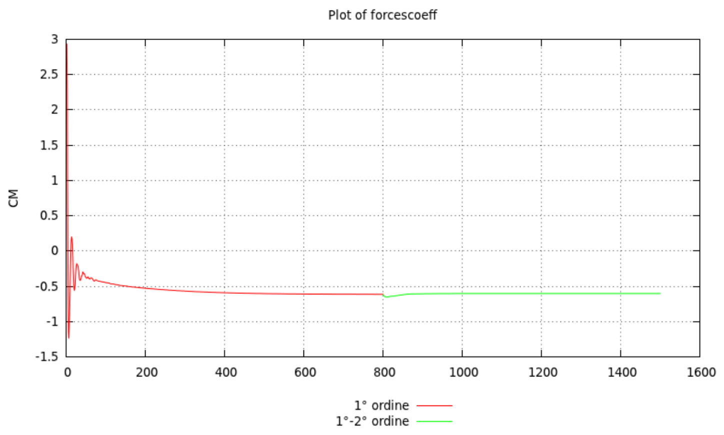


Figura 7.102: Convergence behaviour of CM.

Coefficient	Value
CL	0.3502
CD	0.0408
CM	-0.1731

Tabella 7.43: CL, CD and CM coefficients for the *baseline* geometry.

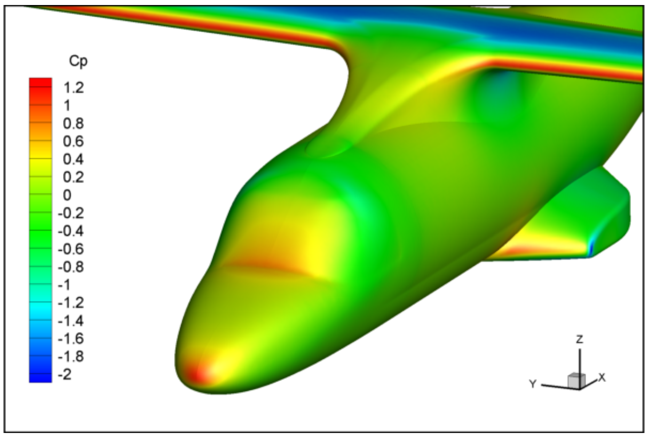


Figure 7.103: Pressure coefficient contours C_p for the *baseline* geometry.

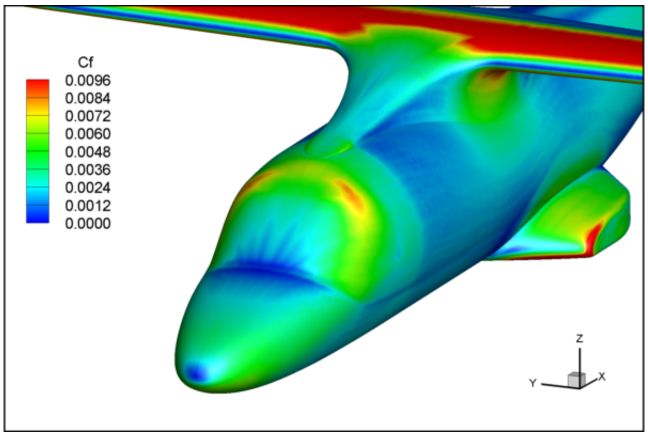


Figure 7.104: Skin friction coefficient C_f for the *baseline* case.

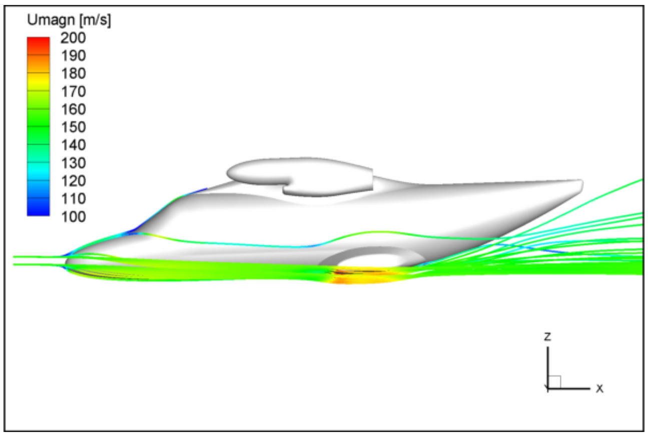


Figure 7.105: Streamlines along a longitudinal plane. *Baseline* geometry.

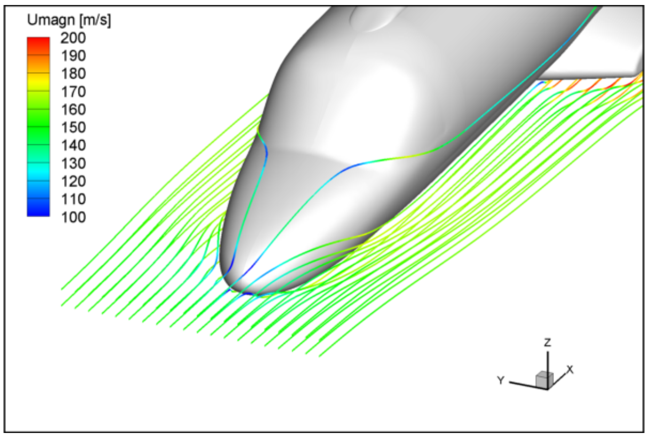


Figure 7.106: Streamlines in the nose region. *Baseline* geometry.

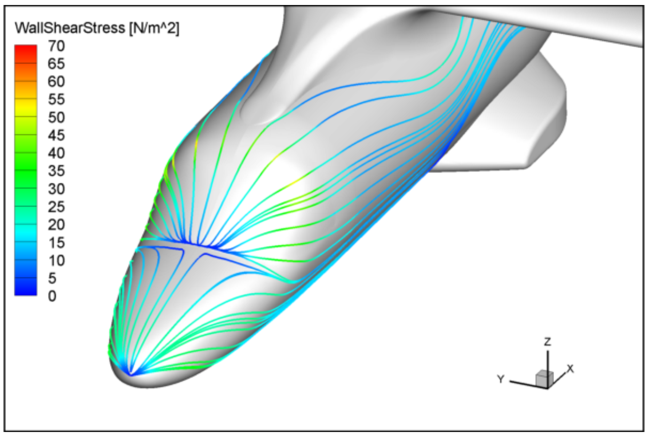


Figure 7.107: Friction lines over the *nose* for the *baseline* geometry.

In Figure 7.108, the y+ distribution over the tiltrotor fuselage is illustrated.

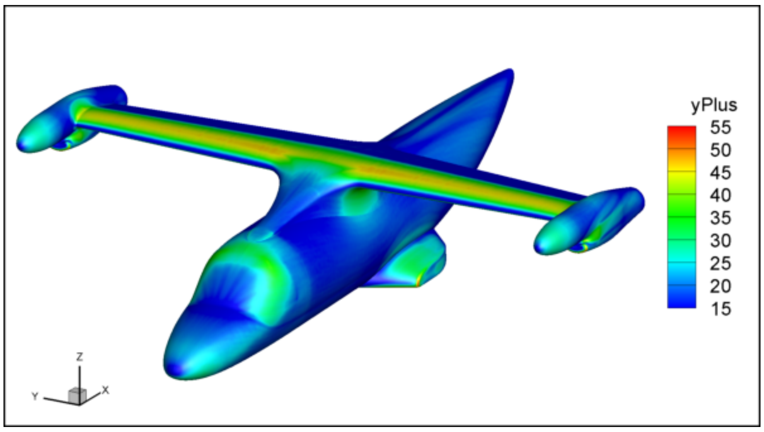


Figure 7.108: Contours of wall y+ over the ERICA fuselage.

As apparent, the selected parameters for grid generation were shown to guarantee that the non-dimensional mesh thickness at the fuselage surface fell within the lower range of the the discretization levels ($y+ = 30 \div 300$) suggested for the standard wall functions implemented in the conventional turbulence models to work properly.

7.31 Parameterization

Once the baseline CFD solution was analyzed, the geometry parameterization was carried out. This operation is of outstanding importance in the optimization process and it can be performed using various techniques. For the scope of the present work, the commercial software Altair HyperMesh® was adopted as the parameterization tool, due to the really versatile capabilities of the morphing tool HyperMorph. The choice of this software is mainly justified with its effectiveness and ease of use, which allows the user to build up complex parametric models in a relatively easy way by means of a dedicated graphical user interface. Moreover, the parameterization procedure in HyperMorph is very general: specifically, it is independent from the peculiar model (either FEM or CFD) which is the object of the optimization analysis. Therefore, whatever the geometry, the user is always allowed to use the same morphing strategies, resulting in a remarkable saving of the required working time for parameterization. Actually, both the above mentioned characteristics are fundamental in an industrial context, where time is always an essential issue. The geometry parameterization was carried out in a series of steps:

1. First of all, the baseline case file was imported into HyperMesh® in Nastran® format.

2. The desired shapes were generated, starting from the baseline geometry, through the morphing techniques available within HyperMorph®. For the present work, the domains-handles approach was addressed, which allowed

large deformations, required for a proper design space exploration, whereas guaranteeing streamlined shapes.

3. The generated shapes were then saved: with this operation, HyperMorph stores the current handles/nodes perturbations, allowing the user to apply them to the baseline model with any given scaling factor. Using this approach, the scaling factor of any generated shape can be dealt with as a design variable by an optimization algorithm.

4. The parameterized model was saved into an *.hm* file, which was then used for the batch-mode parameterization.

5. Finally, the HyperMesh® command file was created. The latter constituted the input file of the *Command.tcl* script described in Appendix C.3.

7.31.1 Morphing Constraints

The introduction of morphing constraints was mandatory to obtain a suitable optimized geometry for industrial applications. The only component to be optimized in the present study was the tiltrotor nose: hence, both the cylindrical portion of the fuselage, the wing-fuselage junction, the wing themselves, the nacelles, the sponsons and the tail cone were kept unchanged. In other words, all the nodes over both the fuselage and the tail cone surface (highlighted in red in Figure 7.109) were constrained to stay fixed.

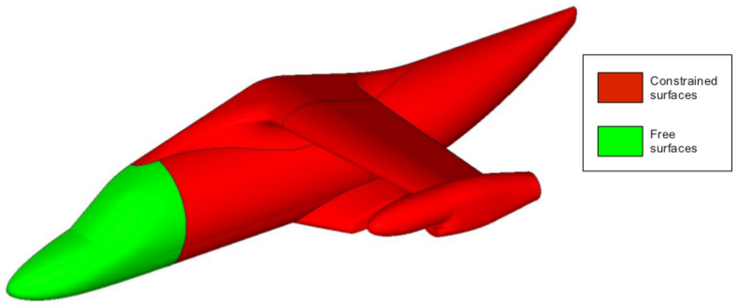


Figure 7.109: Fixed (shown in red) and free (shown in green) surfaces.

Moreover, tangency between the morphed surfaces and the rest of the fuselage components was guaranteed. To do this, specific tangency constraints (please read the User Guide presented in [3]) were added to the model contained into the *.hm* file.

Even though it slightly increased the computational resources required during the mesh deformation operations, the introduction of the morphing constraints was mandatory to obtain a suitable optimized geometry, conformal to non-aerodynamic requirements, such as manufacturing and structural constraints.

7.31.2 Design Variables Definition

In the present section, a brief description of the design parameters used for the ERICA nose trial optimization is provided. A total amount of five design variables were considered. All the design variables were generated using the so called domains-handles approach in order to satisfy global morphing requirements. In particular, this approach allows for the application of mesh nodes displacements within a geometrical region (domain) by changing the location of specific, user defined, control points (handles) ([3]). Once applied, the nodes displacements can be saved as perturbation vectors and then they can be re-applied to the baseline model with any given scaling factor. Actually, the morphed geometry results from a linear combination of the user defined shapes multiplied by their own scaling factor:

$$v = \sum_{i=1}^{5} \alpha_i Sh_i \tag{7.22}$$

where:

- v is the global displacement vector;
- $Sh_i$is the i^{th} basic shape;
- α_i is the i^{th} shape scaling factor and it is actually generated by the optimization algorithm GeDEAII for each analyzed individual.

For the present application, the following ranges were defined for the values of

- $\alpha_i \in [0,1], i = 1,2,3,4,5;$

Using this approach, a scaling factor equal to zero means that the morphed geometry is identical to the baseline one, while a scaling factor equal to one produces the maximum allowed displacement within the specified range. Finally, scaling factors equal to minus one produce the maximum allowed displacement but in the opposite direction with respect to the original definition of the shape modifications.

In the following, the adopted design variables for intake optimization are described. In Figure 7.110, a series of cut planes used to define the shape functions are illustrated, while the x-z or x-y sections of the intake model parametric shapes, applied with a basic scaling factor equal to one, are reported from Figure 7.111 to Figure 7.115; the corresponding handle displacements are visualized as well.

The selected shape functions are described hereafter:

1. **sh1**: this shape consists in an initial deformation along the x-axis of the blue handle illustrated in Figure 7.111, which is located on the transversal cut plane A; in this figure, the baseline shape along with the deformed one are presented. The initial deformation range assigned to this variable was equal to $\pm$ -300 [mm];

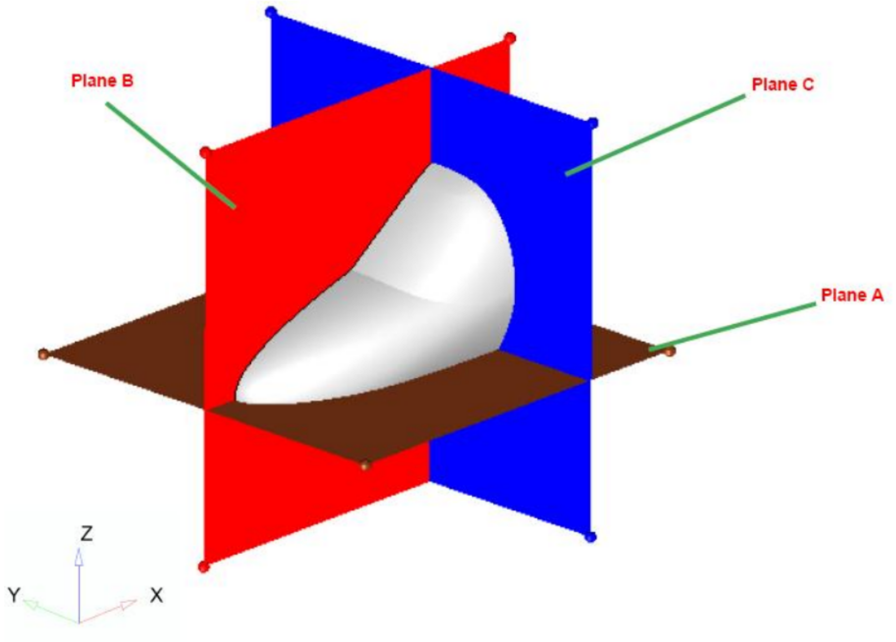


Figure 7.110: Cut planes used for the shape parameterization.

2. **sh2**: this shape consists in an initial deformation along the y-axis of the blue handle illustrated in Figure 7.112, which lies on the transversal cut plane A. The initial deformation range was given a value of ± 80 [mm].

3. **sh3**: this shape consists in an initial deformation along the x-axis of the blue handle illustrated in Figure 7.113, located on the transversal cut plane B. It was given an initial deformation range of ± 100 [mm];

4. **sh4**: this shape consists in an initial deformation along the x-axis of the blue handle illustrated in Figure 7.114, which lies on the transversal cut plane B. It was given an initial deformation range of ± 100 [mm];

5. **sh5**: this shape consists in an initial deformation along the x-axis of the blue handle illustrated in Figure 7.115. It was given an initial deformation range of ± 40 [mm].

During the definition of the parametric shapes, visibility requirements were taken into account. To this purposes, the *total vision envelope* was built for the baseline geometry. The vision envelope along with the procedure for constructing it are reported in Appendix B. As a matter of fact, the deformation ranges were decided so as to automatically satisfy these requirements. Obviously, this choice by far reduced the search space, thus affecting the potential reduction of the objective function.

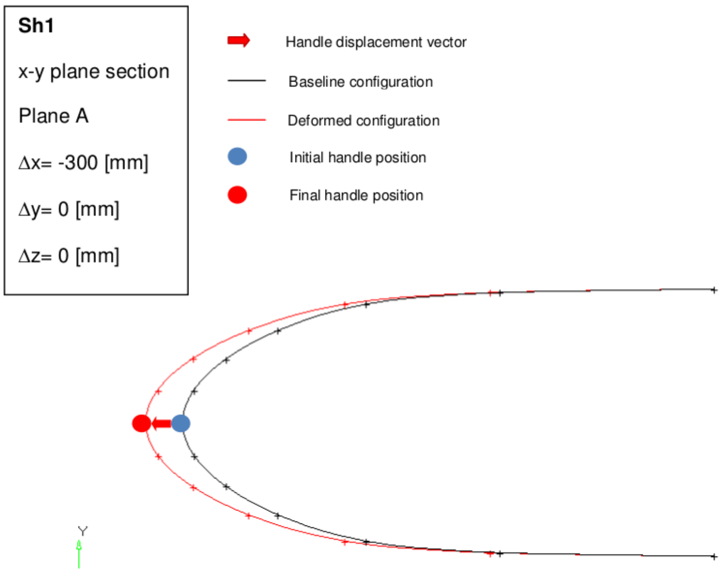


Figure 7.111: Parametric shape Sh1, applied to the intake model with scaling factor $\alpha = +1$.

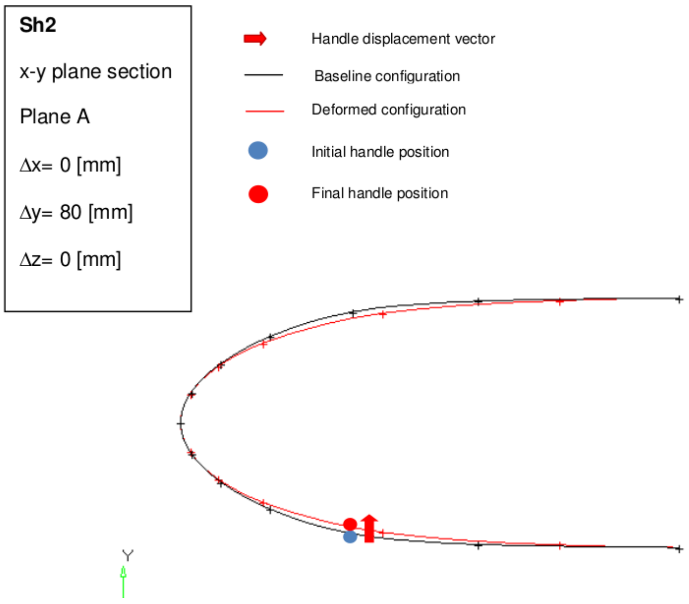


Figure 7.112: Parametric shape Sh2, applied to the intake model with scaling factor $\alpha = +1$.

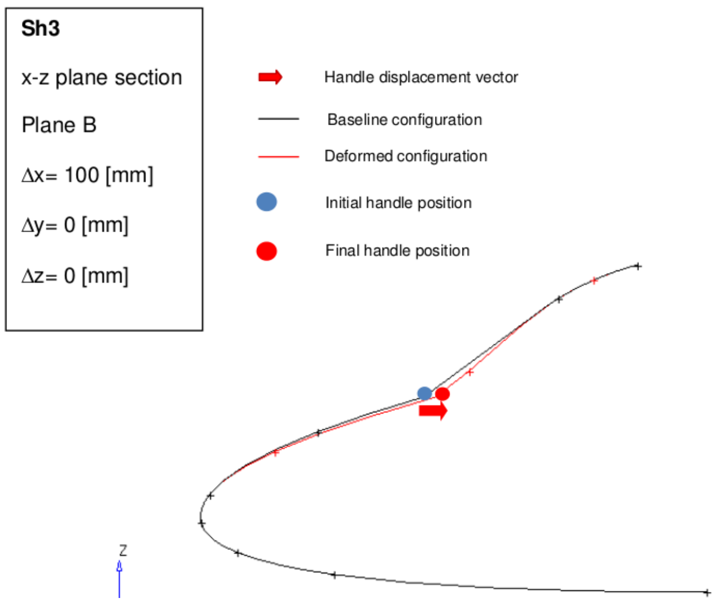


Figure 7.113: Parametric shape Sh3, applied to the intake model with scaling factor $\alpha = +1$.

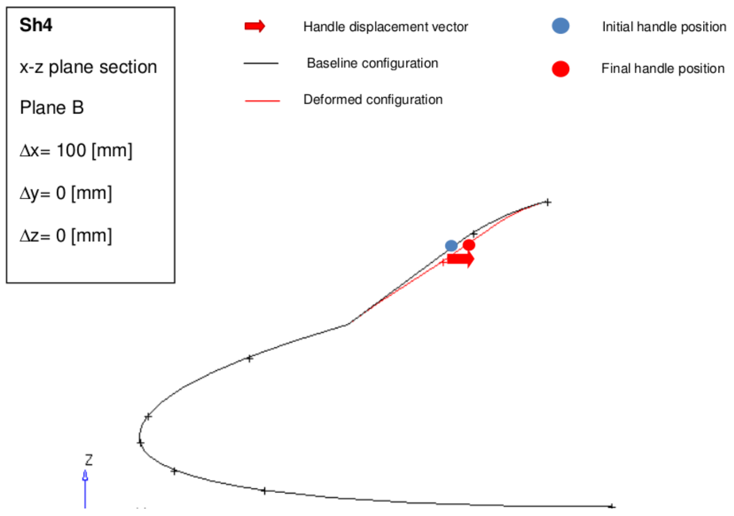


Figure 7.114: Parametric shape Sh4, applied to the intake model with scaling factor $\alpha = +1$.

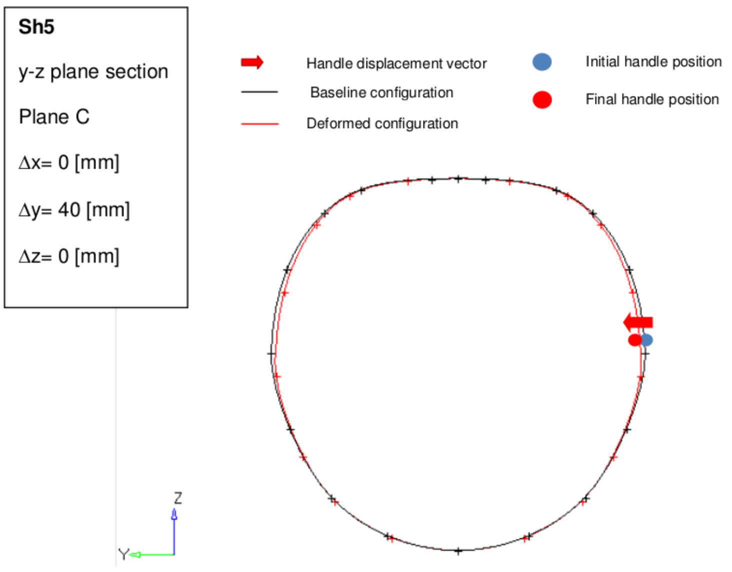


Figure 7.115: Parametric shape Sh5, applied to the intake model with scaling factor $\alpha = +1$.

7.32 Formulation of the intake optimization problem

Once the design CFD model and the parametric model for the nose geometry were built up, the successive step consisted in the GeDEAII-driven optimization. As stated above, the optimization problem considered was actually of the multi-point type, since the total pressure losses occurring in both forward flight and hover conditions needed to be minimized simultaneously. Moreover, the maximum $DC(60)$ at AIP was required to be kept to reasonable levels in both the considered flight conditions. From a mathematical point of view, the optimization problem can be expressed in the following way:

$$Minimize\,[F(\mathbf{x})] \tag{7.23}$$

where $[F(\mathbf{x})]$ is defined as follows:

$$[F(\mathbf{x})] = Overall \quad drag \quad force \tag{7.24}$$

and x is the design variable vector, consisting in the coefficients of the linear combination of the shape functions describing the morphed intake geometries (please read section 7.31.2):

$$x = [\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5] \tag{7.25}$$

subject to the following variables bounds:

- $\alpha_i \in [0,1], i = 1,2,3,4,5;$

As aforementioned, the overall ERICA drag force was monitored. The requirement of keeping the same baseline visibility in the optimized configuration was handled via a proper definition of the design variables ranges. In fact, the variations guaranteed that almost each investigated configuration would feature a wider Total vision envelope (please read Appendix B for a deeper insight into this topic) than that characterizing the baseline configuration. In the case the calculated angles were lower or equal than the baseline ones, the parameterized configuration was automatically replaced by a new one. This replacement was performed until the visibility requirements were met. Moreover, the number of individuals per generation was set to 12, while the number of generations for this trial optimization run was set to 4.

7.33 Discussion of results

The trial intake optimization results are discussed in this section. All the instructions to properly interface the codes involved into the optimization loop were encoded within the script *Command-function.m* similar to the one presented within Appendix C.2. The softwares used were HyperMesh11® for the surface meshing and parameterization (by means of the *Hypermorph* tool), TGrid13® for the volume meshing and OpenFOAM® as the CFD solver.

In Figures 7.116, 7.117, 7.118 and 7.119 the comparison between the baseline and the optimized nose geometries is shown.

First of all, one can see that the frontal region was extended, whereas the windscreen region was moved downstream so as to augment the pilot visibility.

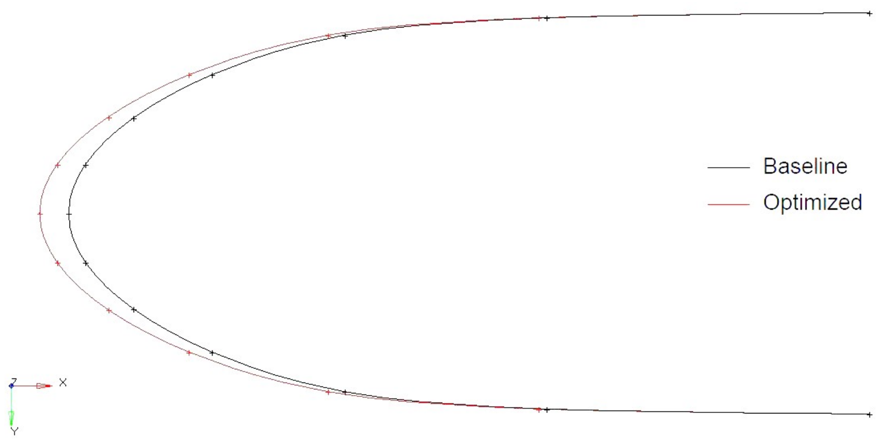


Figure 7.116: Comparison between the *baseline* (black line) and optimized (red line) configurations. *Sh1* shape.

The objective function values are reported in table 7.44, where the CD values are shown along with the overall aerodynamic forces decomposed into their viscous and pressure terms. Results refer to the left part of the investigated ERICA model[19].

As one can see, the optimization process has brought about a pressure drag reduction, while keeping the viscous component essentially unchanged.

[19]Thanks to the flow field symmetry, only the left part was analyzed. This choice allowed to reduce the overall computational time. Results presented in Table 7.44 refer to the left part only.

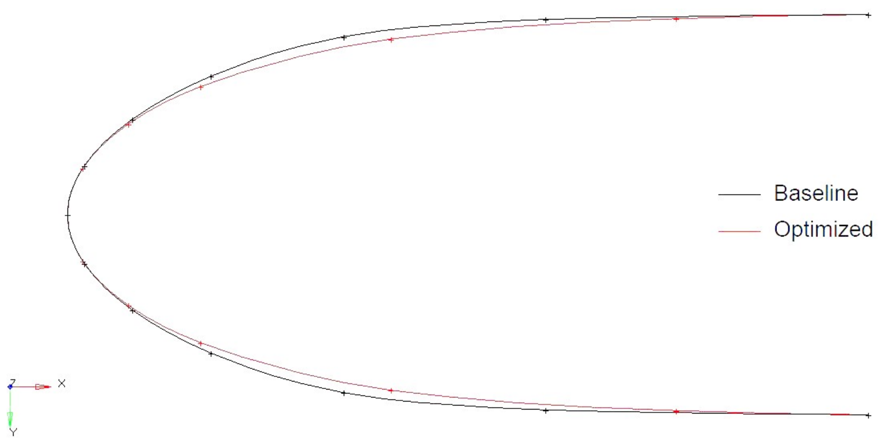


Figure 7.117: Comparison between the *baseline* (black line) and optimized (red line) configurations. *Sh2* shape.

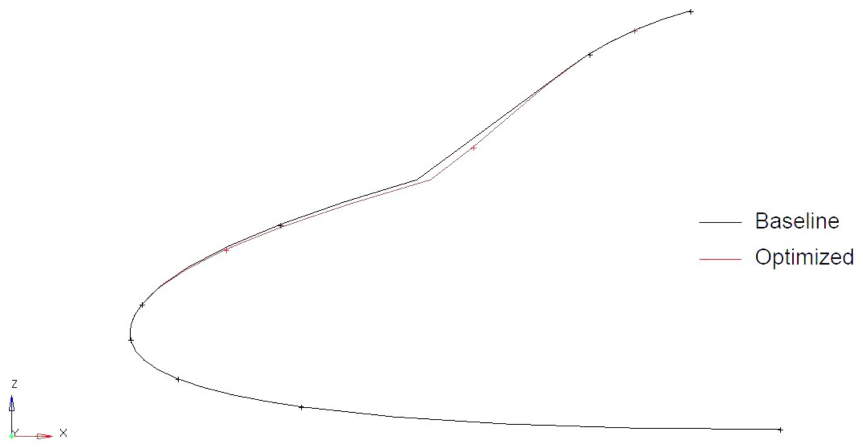


Figure 7.118: Comparison between the *baseline* (black line) and optimized (red line) configurations. *Sh3* shape.

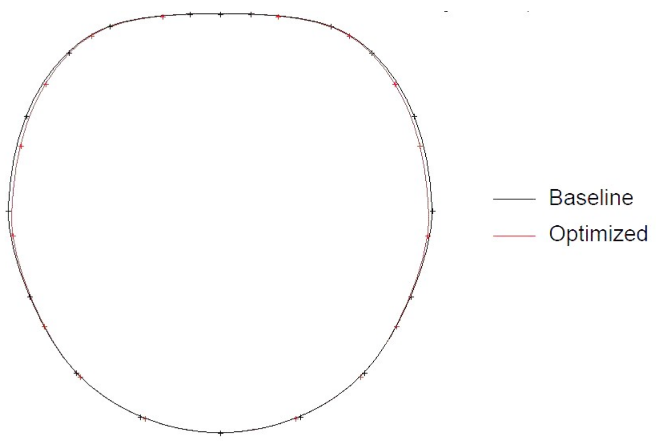


Figure 7.119: omparison between the *baseline* (black line) and optimized (red line) configurations. *Sh5* shape.

	CD	Total drag	Pressure drag	Viscous drag
Baseline	0.020403	5000.98	3612.78	1388.20
Optimized	0.020120	4931.68	3545.70	1385.98
difference %	1.39	1.39	1.86	0.16

Table 7.44: CD comparison and *drag* force [N] between the *baseline* and the optimized *configuration*.

Design variables values along with the related shapes displacements in [*mm*] are reported in table 7.45:

	Sh1	Sh2	Sh3	Sh4	Sh5
Design Variables	0.51986	1.0	0.90458	0.0	0.71689
Shape Displacement [mm]	-155.95	80	90.458	0.0	28.676

Table 7.45: Design Variables and shapes displacements

Shape *Sh1* consisted in an upstream nose displacement, *Sh2* shape narrowed the nose lateral walls up to the maximum allowable value, *Sh5* shape was deformed nearly up to the maximum value.

All these deformations, reported in Table 7.45, caused a reduction of the pressure drag, by means of a reduction of the frontal nose section.

Sh3 shape deformed the lower windscreen edge up to 90[mm] while keeping undeformed the upper edge.

This led to a visibility increase as shown in Figure 7.120. The tips given in Appendix B were followed to built the following graph.

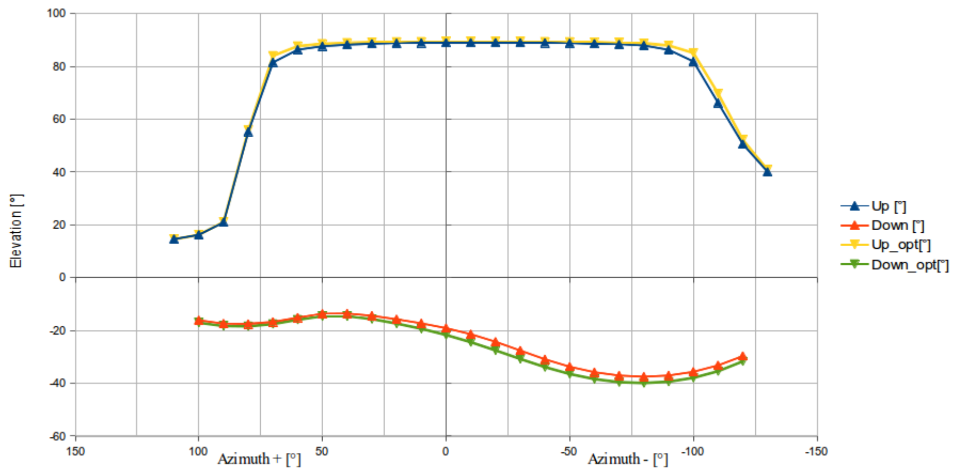


Figure 7.120: Rectilinear graph. Comparison between *baseline* e *optimized* configurations.

According to the values assumed by the design variables, the greater increase concerns the *Down* angles.

Now the comparison of the *Cp* (*pressure coefficient*) and *Cf* (*skin friction coefficient*) between baseline and optimized nose configurations is performed.

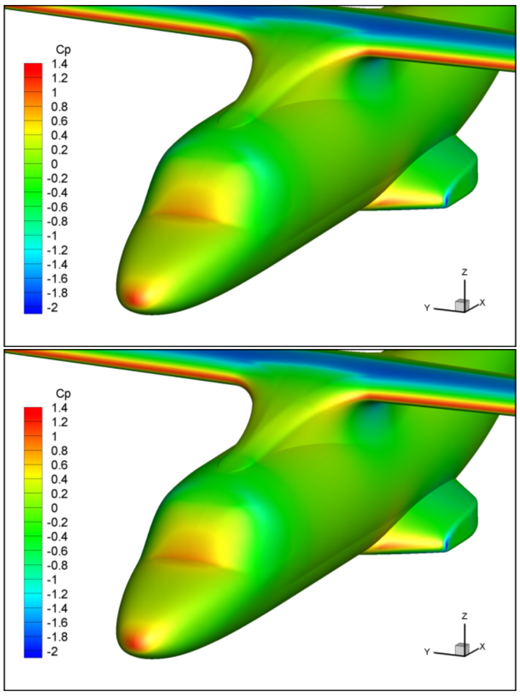


Figure 7.121: Comparison of *Cp* between *baseline* (UP) and *optimized* (DOWN) configurations

As one can see, there is no substantial difference as regards *Cp* and *Cf* values: this is due to the fact that the optimization has not yet reached a substantial drag reduction achievable by means of the GeDEAII driven optimization loop.

Now some graphs are reported, showing the distribution of pressure and shear stresses, along some intersection lines between *nose*, *fuselage* and *wf-fearing* components and the cut planes showed in Figure 7.123.

Figure 7.124 shows pressure profile of the baseline and the optimized configurations along the intersection line with the "A" plane.

As one can see, between x=1 e x=3 [m] the optimized configuration features a pressure reduction. Figures 7.125, 7.126, 7.127 and 7.128 show the pressure distribution of the baseline and optimized configurations along the intersection line with the planes "B", "C", "D" and "E", respectively.

In all the previous figures, one can observe that the optimized pressure profile features lower values then the baseline one, even if to a small extent. Pressure forces acting over the frontal region walls are lower in the optimized configuration when compared to the baseline one, whereas over the *windscreen* a small pressure increase occurs: *Sh3* shape made the windscreen somewhat more perpendicular, hence more exposed to the flow.

Figure 7.129 shows the wall shear stress τ profile along the intersection line between the fusalege and plane "A".

Figures 7.130, 7.131, 7.132 and 7.133 show the distribution of shear stress along the intersection between the fuselage and Planes "B", "C", "D" and "E", respectively.

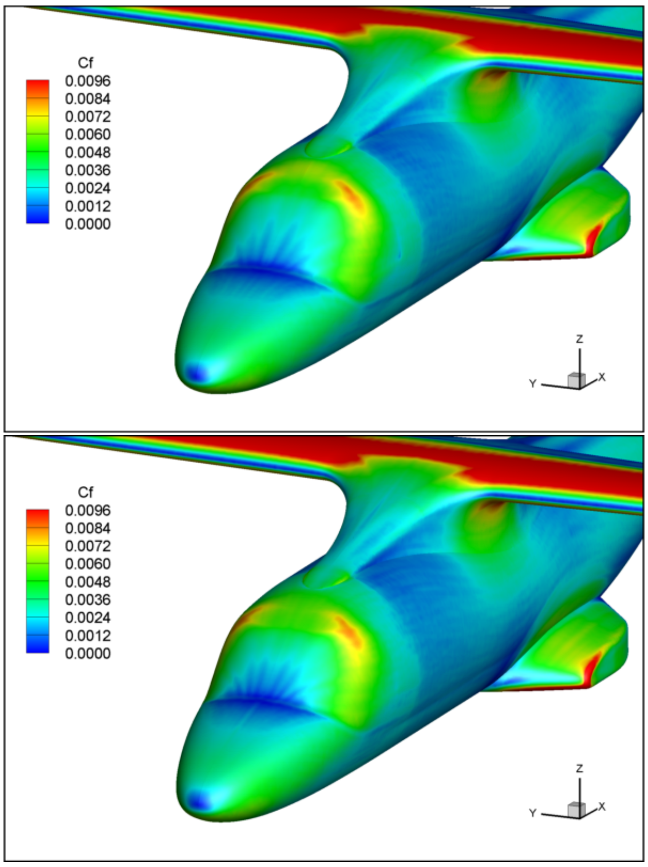


Figure 7.122: Comparison of *Cf* between *baseline* (UP) and *optimized* (DOWN) configurations

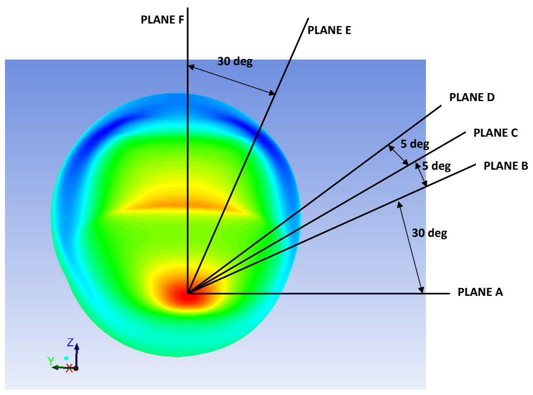


Figure 7.123: Visualization of the cut planes selected for pressure and viscous drag analysis.

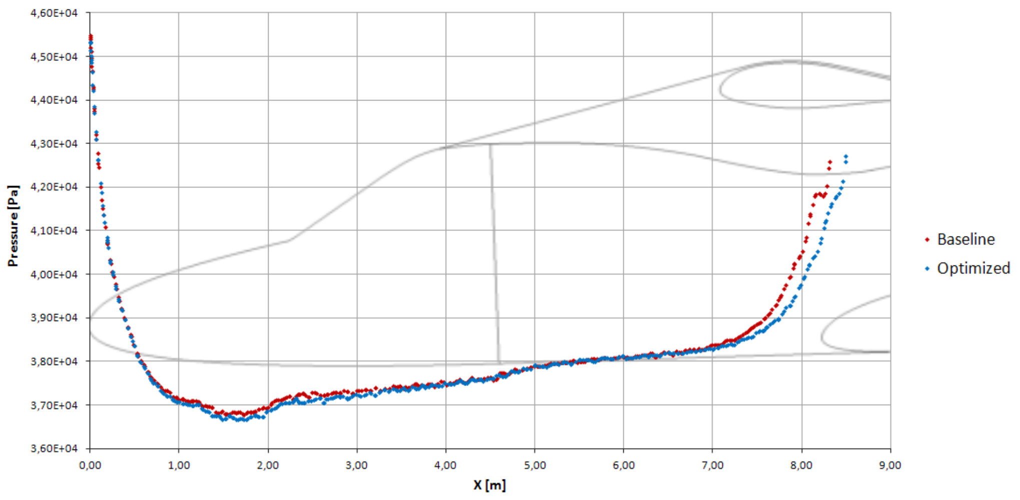


Figure 7.124: Pressure profile over Plane A.

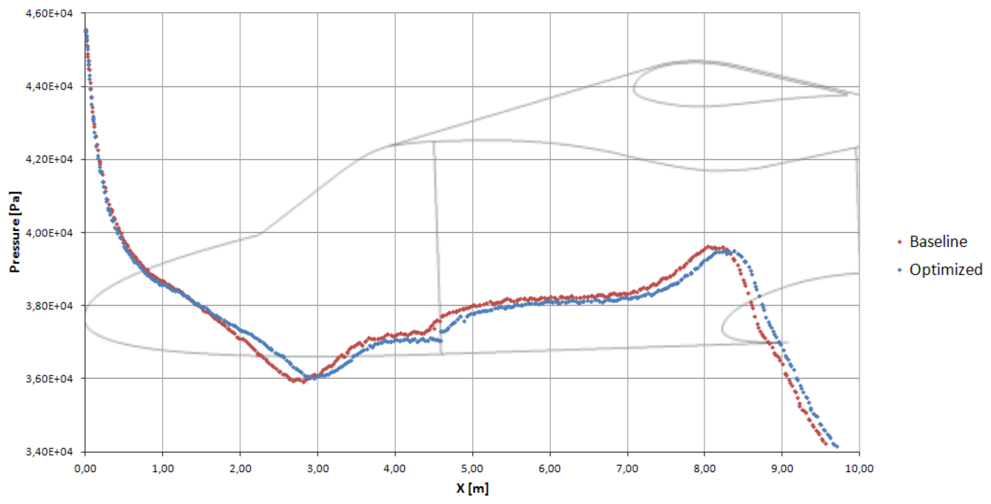


Figure 7.125: Pressure profile over Plane B.

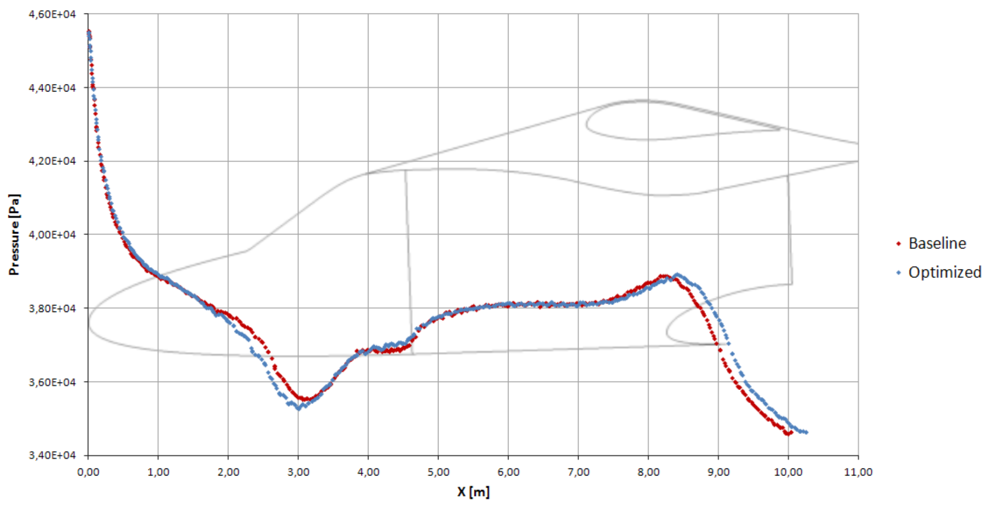


Figure 7.126: Pressure profile over Plane C.

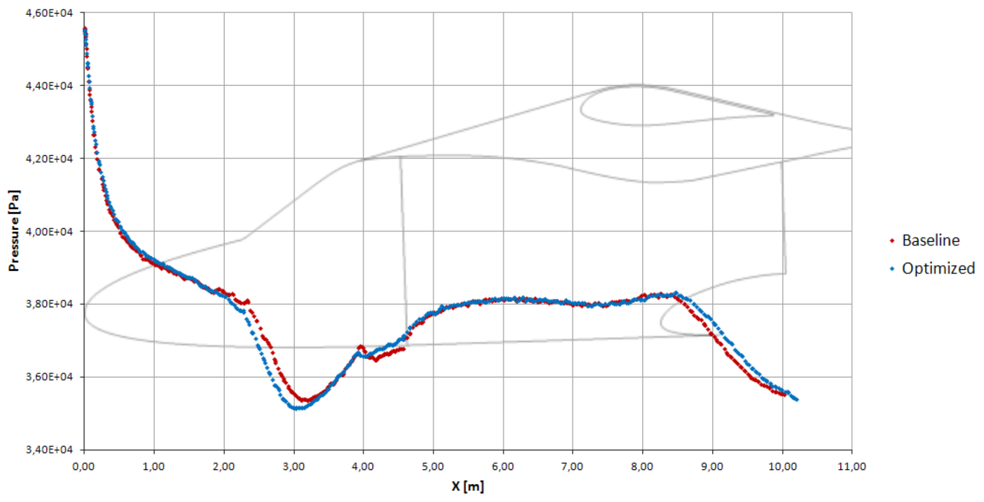


Figure 7.127: Pressure profile over Plane D.

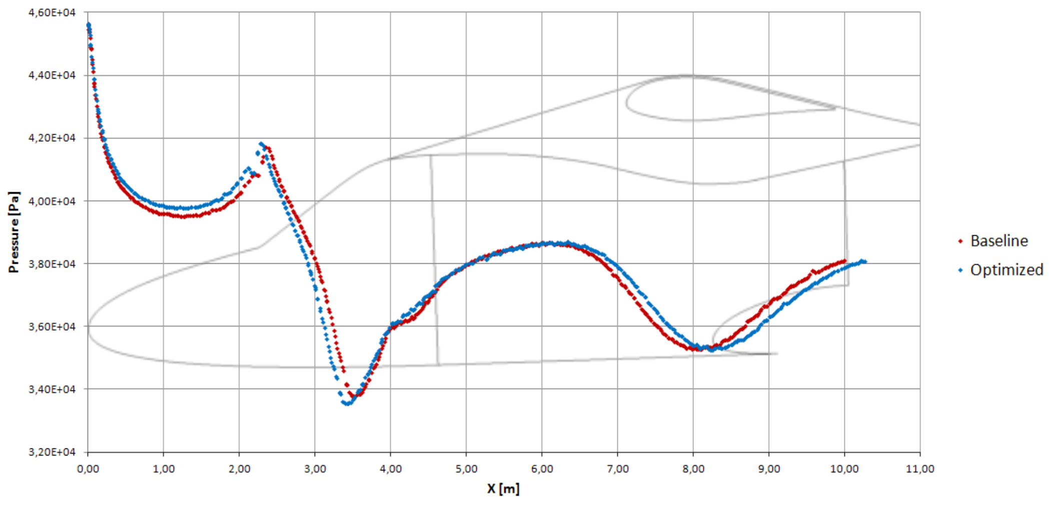


Figure 7.128: Pressure profile over Plane E.

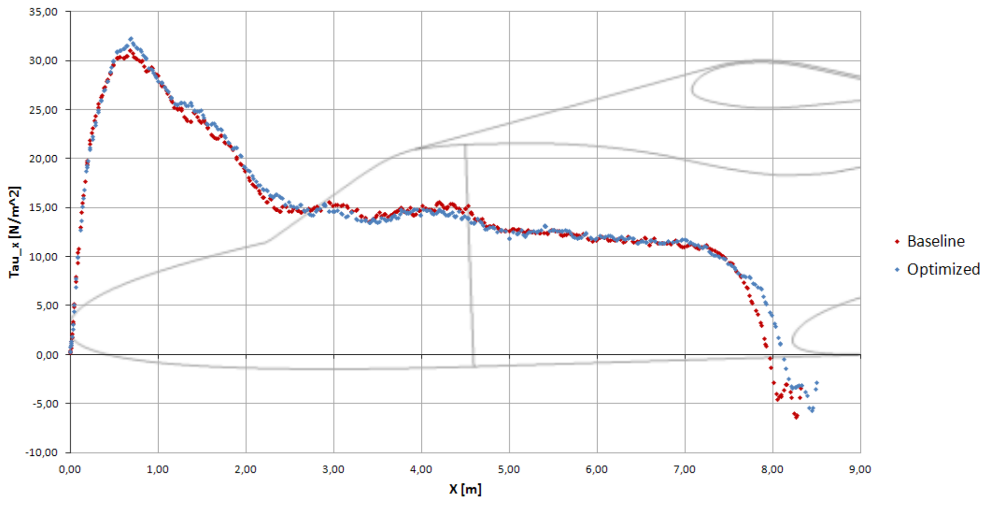


Figure 7.129: τ profile along plane A.

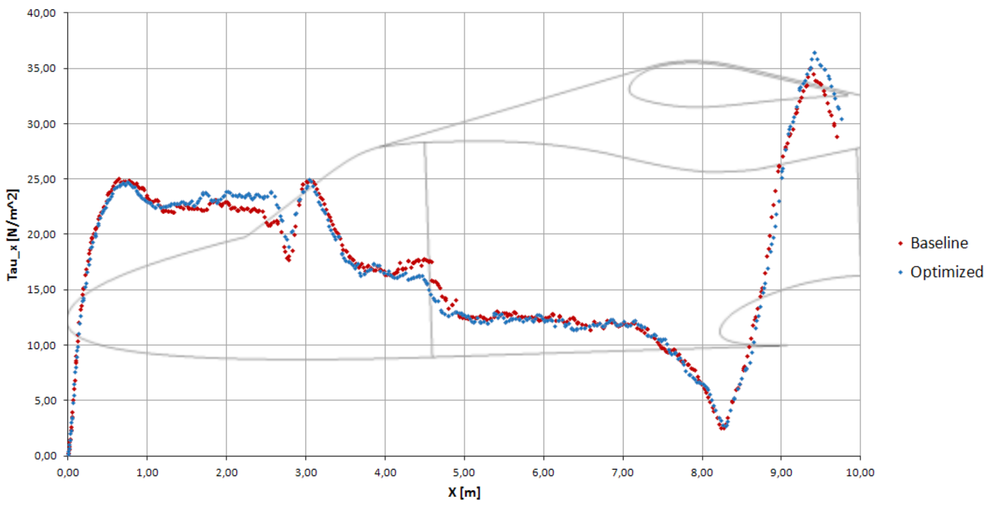


Figure 7.130: τ profile along plane B.

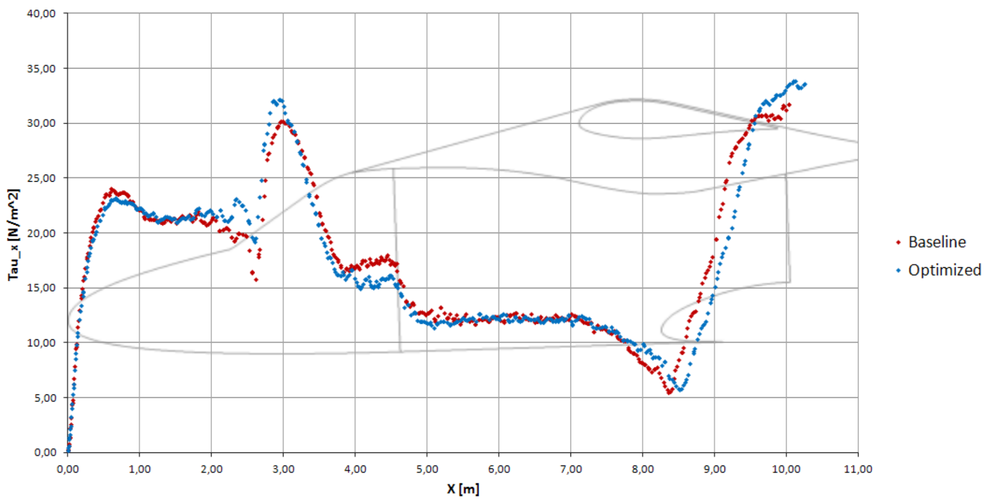


Figure 7.131: τ profile along plane C.

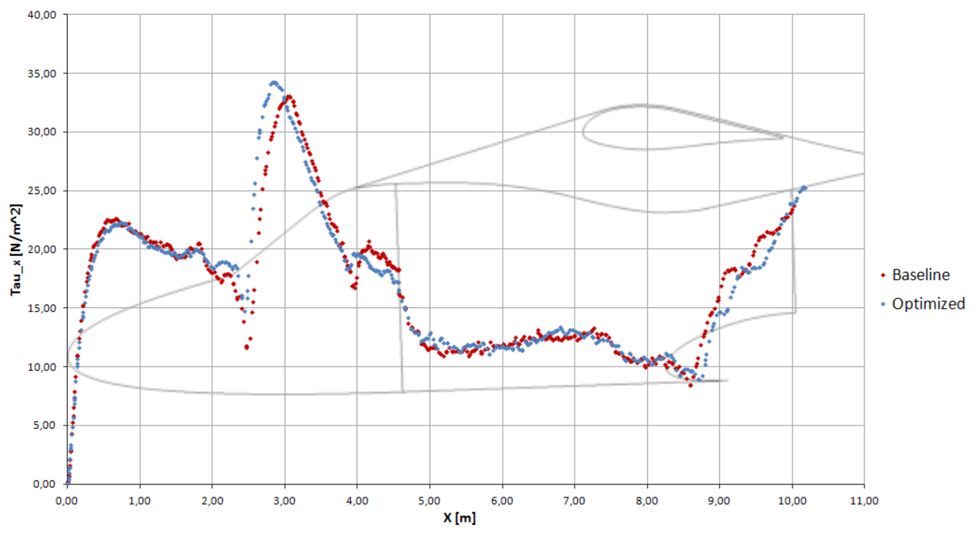


Figure 7.132: τ profile along plane D.

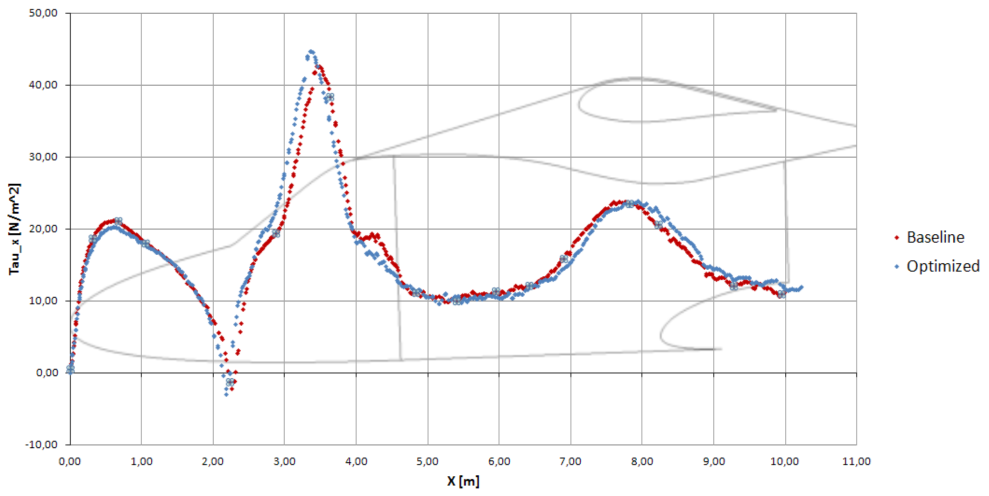


Figure 7.133: τ profile along plane E.

Unlike the pressure forces, the *optimized* configuration features greater wall shear stress then the *baseline* one. However, the reduction of pressure force characterizing the optimized configuration prevails over the increase of wall shear stress, and a total drag reduction is achieved with the optimized geometry.

In conclusion, even though the results in terms of drag force reduction are not so impressive, this optimization run demonstrate that OpenFOAM can effectively take the place of the CFD solver within the optimization loop. Moreover, in order to increase the drag force reduction, a variation range widening is expected to give beneficial effects. A further improvement concerns the boundary conditions settings. In fact, this is a key issue when considering the application of OpenFOAM as the CFD solver, since the results obtained strongly depend on the boundary conditions chosen for the calculation.

Conclusions

In this Thesis, the computer-based automatic optimization loop for the aerodynamic shape optimization of helicopter and rotorcraft components has been discussed.

The first few chapters of the Thesis were intentionally given a didactic/theoretical layout. In fact, this choice allowed to conceptually introduce the reader the basilar optimization features, which were then exploited during the optimization test problems described within the results chapter. In particular, the first chapter dealt with the optimization theory, whereas the second chapter introduced the GeDEAII features and performance. A chapter was then dedicated to the commonly used parameterization techniques when dealing with external aerodynamics problems. A brief introduction to CFD was reported for completeness. An entire chapter concerned the principal passive and active techniques available for immediate or potential usage on modern rotary wing aircrafts to obtain external drag reduction.

Before introducing the results in Chapter 7, a chapter was dedicated to the Intake aerodynamics, whose concepts were recalled in the following.

The automatic optimization loop involved several softwares of different nature which are typically used separately and which needed to be correctly interfaced. The key software was the optimization engine, which managed all the instructions needed to reach the optimum/optima, depending on whether mono or multi-objectives problem were considered, respectively.

The shape parameterization was performed by means of a dedicated free-form deformation software, whereas the CFD mesh was built via a dedicated commercial code. Test cases presented in this Thesis involved both commercial and open-source CFD codes, that is Fluent® and OpenFOAM®, respectively. This choice was motivated by the intention of AgustaWestland, which funded this project, to start studying the capabilities of this open-source code, in order to introduce it gradually into the official optimization procedure for external aerodynamic optimization.

Optimization loops involving Ansys Fluent® as the CFD solver, presented in Part I and II of Chapter 7 were developed specifically to work under Windows® environment, whereas the one presented in Part III of Chapter 7 employing OpenFOAM® as the CFD solver was entirely designed to work under Linux/UNIX environment.

Optimization loop has been designed to work in both Windows and UNIX environments. This feature has a twofold motivation:

- To give the possibility to a potential user to perform optimization runs on workstations under Windows environment;
- To give the possibility to a company to deeply exploit HPC capabilities, typically by means of UNIX-based clusters.

All the instructions needed to correctly perform the optimization run were encoded in Matlab language, and collected into a single script, to reduce as much as possible the complexity of the loop.

Presented test cases concerned the shape optimization of the ERICA tilt rotor nose region and the engine air intake ♯1 of the AW101 helicopter. During the three test cases, GeDEAII, the Evolutionary algorithm designed and perfected during my PhD course, constituted the optimization engine. Achieved results demonstrate the strength of the algorithm under severe optimization environments. Test cases intentionally featured different optimization difficulties, such as the presence of functional constraints, handled via the penalty function approach, or the presence of geometrical constraints, which made harder the search procedure of the optimization algorithm. As a matter of fact, these difficulties are common when real engineering optimization problems are concerned. Moreover, they prove further the effectiveness, the strength and time-saving characteristics of the presented optimization loop.

As far as future works is concerned, it can be stated that the optimization loop can be improved in almost all its constitutive parts. That is, each software involved in the optimization loop can be improved singularly. This would actually translate into an enhancement of the overall optimization performance.

In particular, enhanced CFD techniques can be exploited, such as RANS unsteady simulations or LES (Large Eddy Simulation) calculations. As a matter of fact, fluid dynamic accuracy is a key issue when analyzing the optimization results.

Moreover, it is underlined here that the whole optimization flowcharts presented in Figures 7.1, 7.45 and 7.83 wouldn't require to be modified to accommodate these improvements brought about to the CFD solver. If this was the case, one would only have to change the block concerning the CFD solver, and be careful to properly link the output of the preceding software as the input of the new one.

A potential development consists in making the optimization procedure loop entirely based on open-source softwares, which is a challenge from both the research and the industrial point of view. In fact, from one hand this would allow the designer to deeply and totally customize each software features. From the other hand, this would allow to reduce further the design costs. As far as the free-form parameterization tool is concerned, one could built an in-house code, and validate it against the Altair HyperMesh® software. As regards 3D mesh generation, GMSH ([16]), a finite element mesh generator with built-in pre- and post-processing facilities, may be used in place of the commercial code Ansys Tgrid® . GMSH can be run either interactively using the graphical user interface or in ASCII text files using GMSH's own scripting language. The last feature would allow to easily integrate it into the automatic optimization loops presented in this Thesis.

The real issue to be faced concerns the replacement of the state-of-the-art commercial codes with other free softwares, without a loss of performance, which would directly translate into a reduction of product quality. A lot of work must be done to perform this shift, first by testing singularly the performance of each open-source software, and finally by proving the effectiveness of whole open source software-based optimization loop.

Appendices

Appendix A

$DC(60)$ computation Procedure

A standard function to evaluate this parameter does not exist in Ansys Fluent®, so a dedicated Matlab routine has been used for this non-standard post-processing operation. This function provided also an automatic $DC(60)$ extraction procedure for the subsequent optimization activities.
The first step is to extract from Ansys Fluent® a series of data fields needed to evaluate the $DC(60)$ coefficient ([13]). The first of those fields regards the coordinates x, y and z of the cell centroids. The second one regards the areas of the cells constituting the patch over which the $DC(60)$ must be calculated. The third one constitutes the total pressure data of each cell. All these tasks were accomplished by means of a unique Ansys Fluent® command, present in a dedicated Ansys Fluent® journal file. The latter contains all the instruction to properly set up the CFD boundary condition following the tips expressed within Part I of Chapter 7.

An excerpt about the instructions needed to extract these fields is reported here for the sake of clarity:

```
/file/write-profile

File.prof

aip

()

total-pressure

face-area-magnitude

()
```

After having extracted these needed informations, they are stored into the

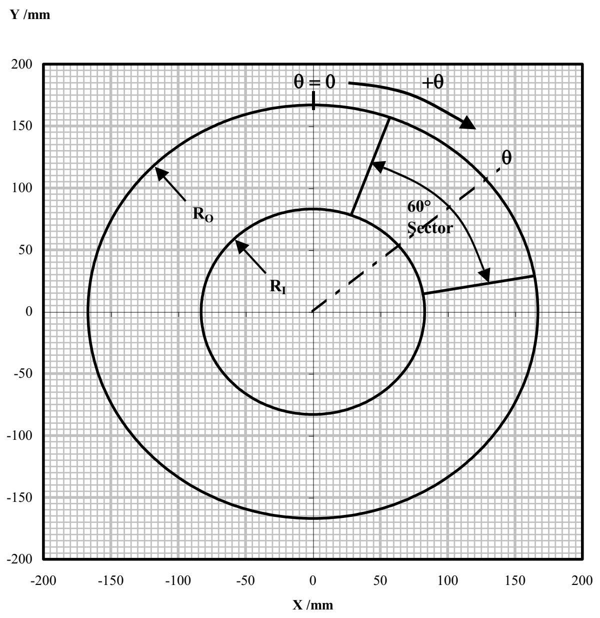


Figure A.1: View of AIP from Downstream Showing 60°Sector for $DC(60)$ Calculation at Angle. (taken from [37])

File.prof file, which is then imported into Matlab environment and processed for the purpose of $DC(60)$ calculation, via a dedicated script.

In particular, the Rolls-Royce Distortion Parameter, DC60MAX, is calculated, following the guidelines given in [13]. This parameter defines the circumferential distortion for a 60° sector of the AIP. In particular, according to the Rolls Royce convention, angles are assumed positive in clockwise direction, in a downstream-to-upstream view, as depicted in Figure A.1. The area weighted average total pressure in any 60° sector of the AIP can be calculated from the summation of the face areas, A_n, and values for total pressure, H_n, for the n faces in that sector,

1. $H_n' = \frac{\sum A_n H_n}{\sum H_n}$ in [Pa], where H_n is in [Pa] from Ansys Fluent® and,

2. $A_n = \sqrt{A_{xn}^2 + A_{yn}^2 + A_{zn}^2}$

It is suggested that since the faces are small there is no need for special treatment of faces which cross the boundaries of the 60°sectors; those faces whose centroid is within the sector are included in the sum, and those whose centroid is outside the centroid are not included. For any given angle, the distortion parameter is defined as:

$$DC60(\theta) = \frac{H_1^{'} - H_\theta^{'}}{q_1^{'}} \tag{A.1}$$

where the area weighted total pressure ($H_1^{'}$) and the weighted dynamic pressures ($q_1^{'}$) are in the same units. (the pedex 1 refers to the whole AIP section, whereas the pedex θ refers to the investigated 60 [deg] sector.)

Appendix B

Methodology for assessing compliance with the visibility requirements

Visibility requirements are to be satisfied during certification campaign. For this reason, it is vital to establish if a particular aircraft configuration meet these requirements. In order to establish whether a particular parameterized configuration provides adequate external vision from within the aircrew station or not, a rectilinear graph showing the *total vision plot* must be built up. The vision requirements set forth in this paragraph are applicable to the ERICA configuration introduced in the results section present within the Part III of Chapter 7 and are given relative to the longitudinal fuselage reference line. Total vision envelope of the baseline ERICA configuration is depicted in Figure B.1.

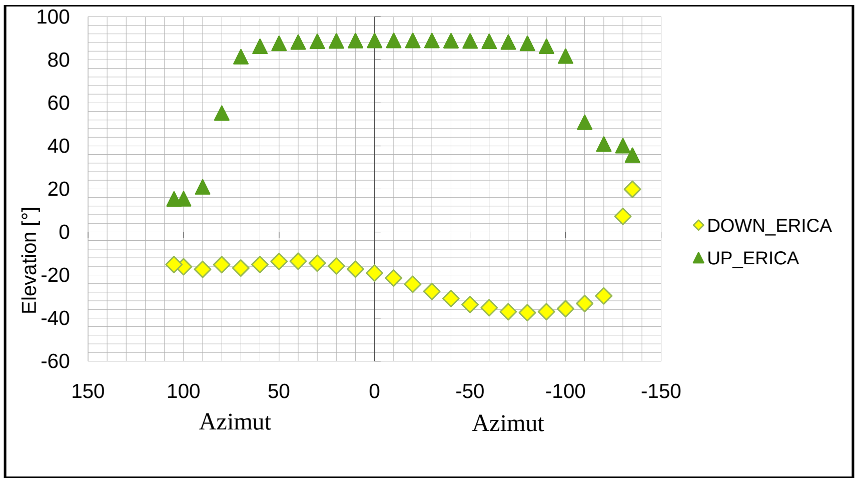


Figure B.1: Total vision envelope of the baseline ERICA configuration.

Figure B.1 shows the *total vision envelope* (plus and minus 110° in azimuth and plus and minus 90° in elevation), referred to the baseline configuration of the ERICA. The plots shall reflect the unobstructed vision area, which is defined as that area of transparent material through which vision is unobstructed by structure, edge bonding material, or any other material, which prohibits clear vision. Convention for azimuth angles is shown in B.2. The reference plane, from

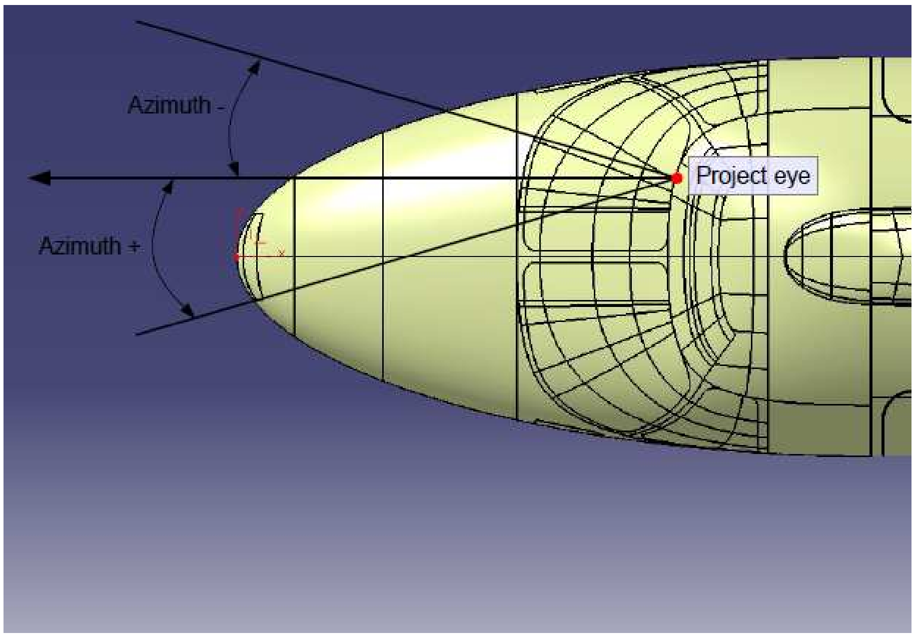


Figure B.2: Assumed azimuth angles convention.

which the vision angles are specified, shall be the pilot's horizontal vision plane (or line) with respect to the specific aircraft longitudinal fuselage reference line (see Figure B.2). The zero reference in azimuth shall be straight ahead of the design eye position.

A subroutine written in *Tcl*[1] script language was created in order to find, for each azimuth coordinate, the intersection between the iso-azimuth lines and the windscreen edges. Based on these values, the Up and Down angles were then calculated via a Matlab script. Definition of Up angles and Down angles is clearly visible in Figure B.3.

Up angle is defined as the angle ranging from the horizontal vision plane to the intersection with the upper windscreen edge (at that particular azimuth angle). Down angle is defined as the angle ranging from the horizontal vision plane to the intersection with the lower windscreen edge (once again, at that particular azimuth angle).

It follows that for each azimuth angle, two elevation angles can be defined, that is the Up and the Down angles. The envelope of these angles constitute the *total vision plot* of the investigated aircraft configuration, and it has been already

[1]*Tcl* (Tool Command Language), commonly pronounced "tickle", is a scripting language like Perl, JavaScript, Born, and Korn. It can be used coupled with HyperMesh, by calling HyperMesh commands directly from a Tcl script.

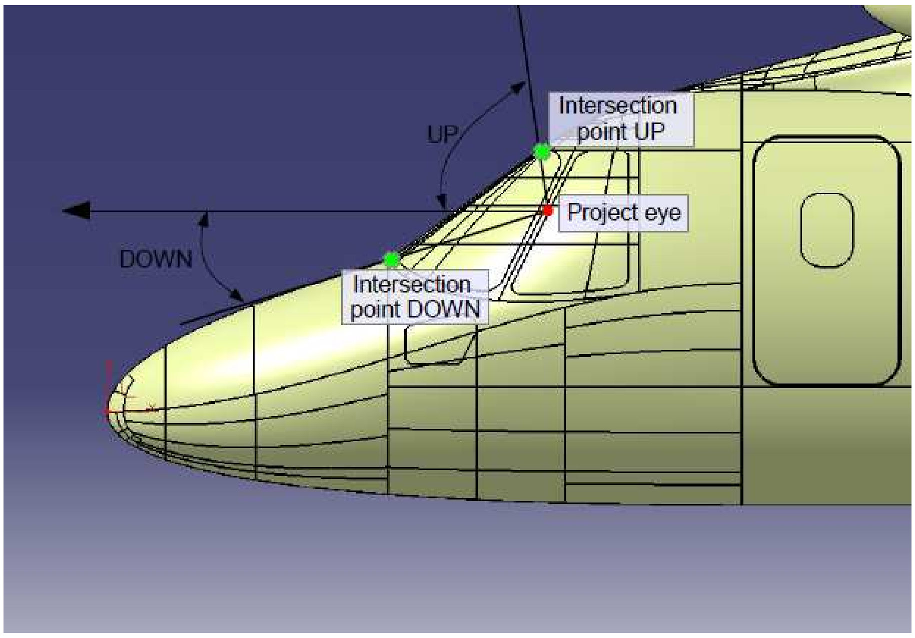


Figure B.3: Assumed elevation angles convention.

presented in Figure B.1.

The zero reference in azimuth is coincident with the pilot eye (as suggested in [ref]). As far as the ERICA nose optimization problem is concerned, it is located at x=2976 [mm], y=525[mm] and z=1358[mm].

During the optimization run, the calculated Up and Down angles related to the deformed configurations were compared to those reported in Table B.1.

Azimuth [°]	Up [°]	Down [°]	Azimuth [°]	Up [°]	Down [°]
-130	40.07		0	88.96	19.11
-120	50.56	29.67	10	88.89	17.29
-110	66.16	33.19	20	88.77	15.76
-100	81.88	35.64	30	88.57	14.40
-90	86.29	37.03	40	88.23	13.57
-80	87.92	37.48	50	87.65	13.66
-70	88.29	37.05	60	86.34	15.11
-60	88.60	35.80	70	81.54	16.74
-50	88.79	33.73	80	55.25	17.54
-40	88.90	30.91	90	20.94	17.37
-30	88.97	27.60	100	16.24	16.11
-20	89.00	24.27	110	14.61	
-10	88.99	21.40			

Table B.1: *Up* and *down* angles as a function of *azimuth* angles for the ERICA baseline configuration.

In the case the calculated angles were lower or equal than the baseline ones, the parameterized configuration was automatically replaced by a new one. This replacement was performed until the visibility requirements were met. This ensured the convergence to an optimal solution able to satisfy automatically the visibility requirements.

Appendix C

Automatic loop optimization files

C.1 *GeDEAII.m*

GeDEAII.m is the Matlab® script related to the multi-objective evolutionary algorithm GeDEAII. It supervises the entire "GEDEAII driven optimization" block depicted in Figures 7.45 and 7.1. This script contains all the instruction needed to perform the optimization running described in Part I, II and III of chapter 7. The GeDEAII flow chart is depicted in Figure C.1. A deeper insight into the GeDEAII framework can be found in Chapter 2, in particular in section 2.2.

In order to start the optimization process, i.e. the "GeDEAII driven optimization", *GeDEAII.m* file must be compiled into an executable (specifically, *GeDEAII.exe* on a Windows® platform, *GeDEAII* on a Linux/UNIX platform) format. Compiling can be performed by means of MATLAB® Compiler using of the following command from the Matlab prompt:

```
mcc -m GeDEA.m
```

Any changes in the *GeDEA.m* Matlab script are possible; however, in this case a re-compiling of the script according to the guidelines given above is needed. Compiling is possible, inside the MATLAB environment, on all desktop or laptop Windows and/or Linux 64-bit based.

For further details about GeDEAII features, readers are referred to Chapter 2.

C.2 *Command-function.m*

Command-function.m is a Matlab® routine needed to interface the GeDEAII core and the other softwares involved in the optimization run. It takes as input from the genetic algorithm core the actual values of the design variables and give back to it the correspondent values of the objective functions, providing the synchronization between GeDEAII and Hypermesh®, Tgrid® and the CFD solver.

It was used in both the D.O.E. analysis and in the optimization process described in Part II of Chapter 7. However, similar files were used during optimization works presented in Part I and III.

In details, it performs the following tasks:

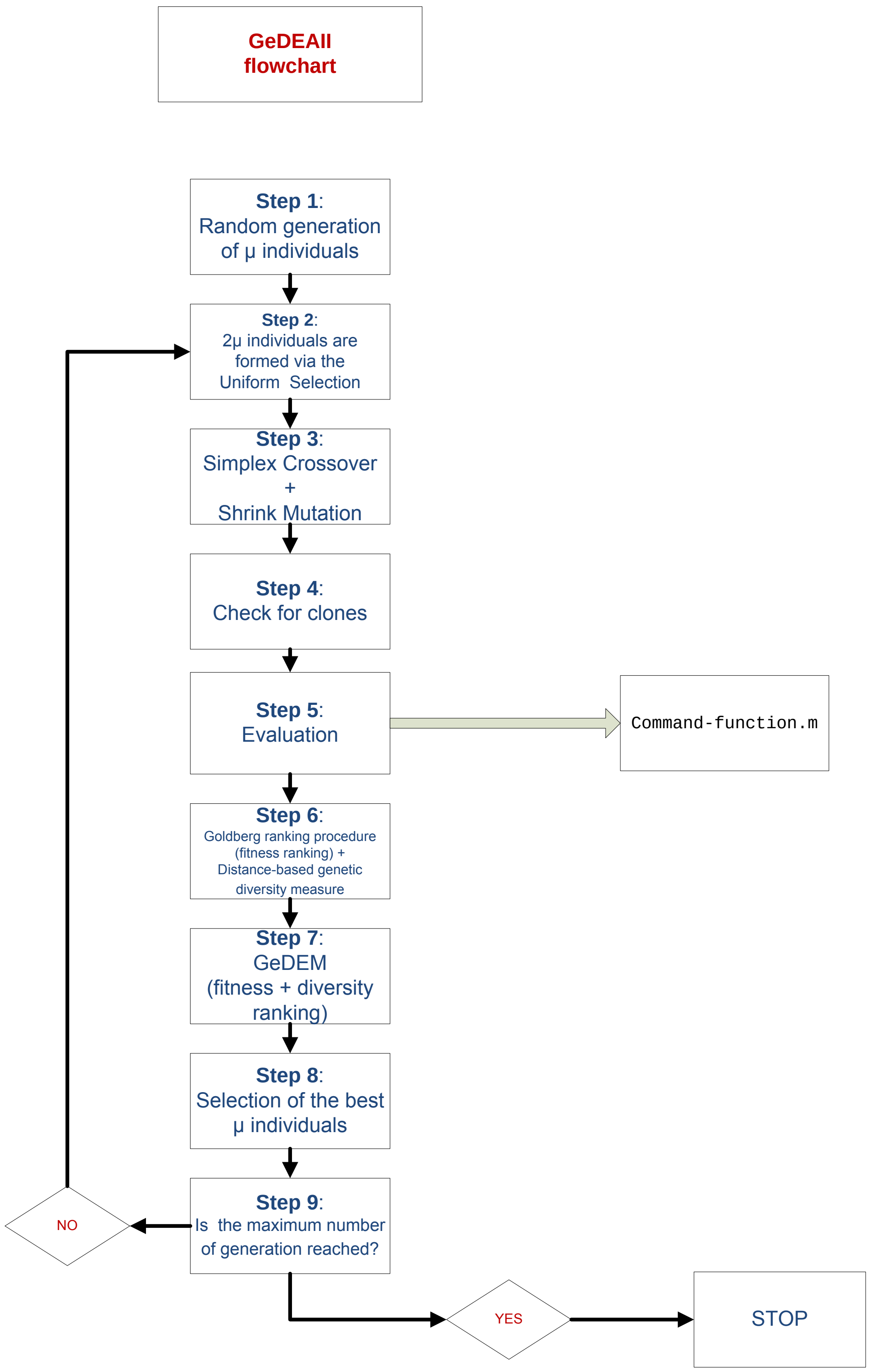

GeDEAII flowchart
Step 1: Random generation of μ individuals
Step 2: 2μ individuals are formed via the Uniform Selection
Step 3: Simplex Crossover + Shrink Mutation
Step 4: Check for clones
Step 5: Evaluation
Command-function.m
Step 6: Goldberg ranking procedure (fitness ranking) + Distance-based genetic diversity measure
Step 7: GeDEM (fitness + diversity ranking)
Step 8: Selection of the best μ individuals
Step 9: Is the maximum number of generation reached?
NO
YES
STOP

Figure C.1: Flow chart of the GeDEAII code.

1. It builds the *design_variables.des* file containing the set of design variables characterizing the current geometry;
2. It launches Altair HyperMesh® in batch mode, which, in turn, performs the shape parameterization itself according to the values read in *design_variables.des* file;
3. It then launches Ansys Tgrid® which, in turn, creates the volume mesh;
4. After that, it launches Ansys Fluent® which, in turn, performs the CFD simulation;
5. It finally reads the objective function/s (created by means of Ansys Fluent® or OpenFOAM® post-processing internal tools) related to the geometry previously created;

Belove, an excerpt of this file in its original script language is presented.

```
function [objvalue,objvalue_total,objvalue_no_tail]=command-function(var_value)
global path;
global nparam;
global c;
global command_mio_ultima_parte;
global command_mio_prima_parte;
global command_mio;
global command_deformed;
global Fluent_batch;
global Altair_batch;
global gt;
path_Fluent='"C:\Program Files\ANSYS Inc\v120\fluent\ntbin\win64\fluent.exe"'
pathtxt=...
'"C:\Users\Administrator\Desktop\Claudio\Optimization\OptimizerInput.txt"'
[parametro,valore] = textread(pathtxt,' %s %n',2,'delimiter','=');
CPU=valore(1);
CPU=num2str(CPU);
gt=valore(2);
objvalue=[path,'objvalue.data'];
objvalue_total=[path,'objvalue_total.data'];
objvalue_no_tail=[path,'objvalue_no_tail.data'];
path_Tgrid='"C:\Fluent.Inc\ntbin\ntx86\tgrid.exe"';
Tgrid=...
[path_Tgrid ' -r5.0.3 3d ' '-hidden' ' -i "C:\Tgrid-batch.jou" &'];
file_log2=[path, 'Tgrid_mio2.log'];
pathFluentjournal=...
'"C:\Users\Administrator\Desktop\Claudio\Optimization\Fluent-batch.jou"';
fluent=[path_Fluent ' 3ddp -t' CPU  ' -hidden' ' -i' pathFluentjournal '&'];
path_Altair='C:\Altairwin64\hw10.0\hm\bin\win64\hmbatch.exe';
```

```
Fluent_batch=[path,'Fluent_batch.jou'];
DELIMITER = ',';
HEADERLINES = 3000;
fileToRead1=command_mio;
rawData1 = importdata(fileToRead1,DELIMITER,HEADERLINES);
size1=size(rawData1);
raw=size1(1);
f=0;
var_value_true(1)= var_value(1);
var_value_true(2)= var_value(1);
var_value_true(3)= var_value(2);
var_value_true(4)= var_value(3);
var_value_true(5)= var_value(4);
var_value_true(6)= 0;
var_value_true(7)= 0;
var_value_true(8)= var_value(5);
var_value_true(9)= var_value(5);
var_value_true(10)= var_value(6);
var_value_true(11)= var_value(6);
var_value_true(12)= var_value(7);
var_value_true(13)= var_value(7);
var_value_true(14)= var_value(8);
var_value_true(15)= var_value(8);
openfile=...
'C:/Users/Administrator/Desktop/Claudio/Optimization/design_variables.des';
fid778=fopen(openfile,'wt+');
[m,n]=size(v);
for j = 1 : n
  v2=v(1,j);
  v1=num2str(v2);
  fseek(fid778,0,'eof');
  fprintf(fid778,'%s',v1);
  fprintf(fid778,'\n');
end
fclose(fid778);

b=[path_Altair ' -c"' command_deformed '"'];
[status,result] = dos(b,'-echo');

[status,result] =dos(Tgrid,'-echo');
fid344=fopen(file_log2);
if fid344 ~= -1
 fclose(fid344);
end
```

```
while fid344==-1
     pause(1);
     fid344=fopen(file_log2);
     if fid344 ~= -1
      fclose(fid344);
     end
 end
 pause(5);
[status,result] = dos(fluent,'-echo');
fid345=fopen(objvalue);
if fid345 ~= -1
  fclose(fid345);
end
controlllll1=0;
tic;
while fid345==-1
     pause(1);
     fid345=fopen(objvalue);
       if fid345 ~= -1
        fclose(fid345);
       end
       elapsed=toc;
       if fid345 == -1 && (elapsed>=21600)
           controlllll1=1;
           dos('pskill cx392.exe');
           break
       end
end
pause(5);
if controlllll1==0;
 [string,pressure,viscous,total] =...
 textread(objvalue,'%s %f %f %f',1,'headerlines',14,'delimiter',' ');
 [string,pressure_total,viscous_total,total_total] =...
 textread(objvalue_total,'%s %f %f %f',1,'headerlines',18,'delimiter',' ');
 [string,pressure_no_tail,viscous_no_tail,total_no_tail] =...
 textread(objvalue_no_tail,'%s %f %f %f',1,'headerlines',16,'delimiter',' ');
 objvalue=[pressure,viscous,total];
 objvalue_total=[pressure_total,viscous_total,total_total];
 objvalue_no_tail=[pressure_no_tail,viscous_no_tail,total_no_tail];
else
 objvalue(1:3)=0;
 objvalue_total(1:3)=0;
 objvalue_no_tail(1:3)=0;
 end
delete('*deformed*');
delete('nt.log*');
delete('mio*.log*');
delete('mio.log*');
delete('mio2.log*');
delete('*.dat*');
delete('*.cas*');
delete('*Fluent*.bat*');
delete('*Fluent.log*');
delete('*fluent.log*');
delete('*cleanup*');
delete('Tgrid_mio*');
delete('command.cmf');
delete('command');
clc;
return
```

C.3 *Command.tcl*

This file contains all the instructions to correctly perform the shape parameterization in Altair Hypermesh® environment. In details, it performs the following instructions:

1. It opens the Hypermesh .hm file, in order to read the superficial mesh and the parameterized model previously created;
2. It performs the shape parameterization itself, by reading the design variables values previously stored in the *design_variables.des* file;
3. It smooths the deformed superficial mesh by means of the internal algorithms present in Altair Hypermesh® ([3]);
4. It exports the deformed superficial mesh model into a format readable by Ansys Tgrid®.

In Figures 7.1, 7.45 and 7.83 it corresponds to the *Hypermesh* circle.

It is written in TCL programming language.

Belove, an excerpt of the *Command.tcl* used during AW101 ♯1 intake described within Part I of Chapter 7 is presented. However, similar files were used during optimization works presented within Part II and III.

```
cd "C:/Users/Giovanni/Desktop/AW101"
set work_dir [pwd];
set path_altair_script "C:/Altairwin64/hw10.0/hm/scripts/cfd/cfd_comm_proc.tcl";
set cfd_aqa 1;
set OutPutFileName "C:/Users/Giovanni/Desktop/AW101/Left_intakefinal.cas";
set UpdateCasFileName "C:/Users/Giovanni/Desktop/AW101/Left_intakefinal_5.cas";
*templatefileset "C:/Altairwin64/hw10.0/templates/feoutput/general";
```

```
*readfile "$work_dir/file1200trim18000und3000_2_nofillet_10_5.hm";
*morphupdatedatabase;
*createmark elements 1 "all";
*createmark elements 2;
*elementtestjacobian elements 1 0 2 2 0 "2D Element Jacobian";
set checkjac [hm_getmark elements 2 1];
set checkjaccount [string length $checkjac];
if {$checkjaccount > 0} {
     set fileID5 [open "$work_dir/Negativevolumeiniziosexist.des" "w+"];
     puts $fileID5 $checkjac;
     close $fileID5;
}
*morphupdateparameter debug 0;
*morphupdateparameter qanegjac 0;
set parameters_number1 9;
set fileID [open "$work_dir/design_variables.des"];
set design {};
set var_name "sh";
*createmark domains 1 "all";
*morphreparam domains 1;
for {set i 1} {$i <= $parameters_number1} {incr i} {
gets $fileID i_design;
lappend design $i_design;
set var_index "$i";
append full_var_name $var_name $var_index;
*createmark shapes 1 "$full_var_name";
*morphshapeapply shapes 1 $i_design;
unset full_var_name;
}
unset design;
*displayall;
*createmark shapes 1 "all";
*deletemark shapes 1;
*createmark domains 1 "all";
*deletemark domains 1;
set listato "Intake Intake2 Intake3 IntakeCentral Intakeelbow
ConstrAIP Constrcentral Intakelip";
eval *createmark elems 1 "by comp name" $listato;
*movemark elems 1 "Intake";
*createmark elems 1 "by comp name" "Intake";
*createmark nodes 1;
set numbersmoothing 20;
for {set i 1} {$i <= $numbersmoothing} {incr i} {
 *marksmoothelements 1 1 1 50;
}
```

```
*createmark elems 1 "by comp name" "Far_out";
*createmark nodes 1;
set numbersmoothing 10;
for {set i 1} {$i <= $numbersmoothing} {incr i} {
 *marksmoothelements 1 1 1 10;
}
*createmark elems 1 "by comp name" "Inlet";
*createmark nodes 1;
set numbersmoothing 10;
for {set i 1} {$i <= $numbersmoothing} {incr i} {
 *marksmoothelements 1 1 3 10;
}
*createmark elems 1 "by comp name" "Outlet";
*createmark nodes 1;
set numbersmoothing 10;
for {set i 1} {$i <= $numbersmoothing} {incr i} {
 *marksmoothelements 1 1 1 10;
}
*createmark elems 1 "by comp name" "Simmetry"
*createmark nodes 1;
set numbersmoothing 10;
for {set i 1} {$i <= $numbersmoothing} {incr i} {
 *marksmoothelements 1 1 1 10;
}
*createmark elems 1 "by comp name" "Dummy"
*createmark nodes 1;
set numbersmoothing 50;
for {set i 1} {$i <= $numbersmoothing} {incr i} {
 *marksmoothelements 1 1 2 50;
}
*displaycollectorsall off,1,0;
*displayall;
set ::cfd_aqa 1;
#*evaltclstring "Fluent_save_format cas";
*evaltclscript "cfd_comm_proc.tcl" 0;
*evaltclstring "CfdCommProc_Fluent_save_msh" 0;
set fileID6 [open "$work_dir/ControlloAltair.des" "w+"];
puts $fileID6 $work_dir;
close $fileID6;
hm_exit;
```

C.4 *Tgrid-batch.jou*

This script is used to build the hybrid volume mesh in batch mode in Ansys Tgrid® environment. The volume mesh is finally stored into a *.cas* format file readable by Ansys Fluent®.

In Figures 7.1, 7.45 and 7.83 it corresponds to the *Tgrid* rectangle.

Belove, an excerpt of the *Tgrid-batch.jou* used during ERICA nose optimization described within Part II of Chapter 7 is presented. However, similar files were used during optimization works presented in I and III.

```
 /file/start-transcript
"C:\Users\Administrator\Desktop\Claudio\Optimization\Tgrid_mio.log"
q
q
q
/file/import/
nastran-surf-mesh
"C:\Users\Administrator\Desktop\Claudio\Optimization\box_9_30_40_80_def.dat"
,
q
q
q
/boundary/manage
;flip          ;mettere VIRGOLE
;group-4
;group-5
;group-6
()
q
q
q
/mesh/prism/controls/zone-specific-growth
apply-growth
s-4
s-5
s-6
()
uniform
geometric
10
1
1.2
no
q
q
```

```
q
q
/mesh/prism/controls/improve
improve-warp
yes
50
15
q
q
q
q
/mesh/prism/controls/improve
face-smooth
yes
q
q
q
q
/mesh/prism/controls/improve
node-smooth
yes
q
q
q
q
/mesh/tritet/controls/
skewness-method;;;;;  adv-front
must-improve-skewness
yes
max-skew-improve
yes
q
q
q
/mesh/tritet/controls/
refine-method
adv-front
q
q
q
/mesh/tritet/controls/adv-front-method
first-improve-params
0.4
15
0.85
```

```
10
yes
q
q
q
q
/mesh/tritet/controls/adv-front-method
second-improve-params
0.4
15
0.75
10
yes
q
q
q
q
/mesh/tritet/controls/
cell-size-function
geometric
1.2
q
q
q
/mesh/tritet/controls/improve-mesh
skewness-smooth
0.75
15
q
q
q
/mesh/tritet/controls/
improve
yes
q
q
q
/mesh/tritet/controls/
improve-surface-mesh
yes
q
q
q
/mesh/tritet/local-regions
define
```

```
mia3
800000
1000
5
5500
0
2500
14000
3500
3500
0
0
0
q
q
q
/mesh/tritet/local-regions
define
mia2
40000000
1500
5
10500
0
2500
26000
5000
5000
0
0
0
q
q
q
/mesh/tritet/local-regions
activate
mia2
q
q
q
/mesh/tritet/local-regions
activate
mia3
q
q
```

```
q
/mesh/auto-mesh
yes
pyramids
tritet
no
q
q
q
/file/write-case
"C:\Users\Administrator\Desktop\Claudio\Optimization\box_9_30_40_80_def.cas"
no
q
q
q
/file/stop-transcript
/file/start-transcript
"C:\Users\Administrator\Desktop\Claudio\Optimization\Tgrid_mio2.log"
/file/stop-transcript
q
q
exit
```

C.5 *Fluent-batch.jou*

This file allows to perform all the necessary instructions in Ansys Fluent® environment.

In the first part, mesh scaling along with boundary conditions setting is performed.

Next, the convergence strategy is described, until the residual criteria are fulfilled.

Finally, the .dat files containing all the information on ERICA nose drag are saved, which are then read by GeDEAII master code.

In both Figures 7.1 and 7.45 it corresponds to the *FLUENT* round corner square highlighted in gray.

Belove, an excerpt of the *Fluent-batch.jou* used during ERICA nose optimization described within Part II of chapter 7 is presented. However, a similar file was used during optimization work presented within Part I of chapter 7 .

```
 /file/start-transcript
"C:\Users\Administrator\Desktop\Claudio\Optimization\Fluent.log"
q
q
q
/file/read-case
```

```
"C:\Users\Administrator\Desktop\Claudio\Optimization\box_9_30_40_80_def.cas"
;file confirm-overwrite no
/grid/scale
0.001
0.001
0.001
q
q
q
/grid/reorder
reorder-zones
q
q
/grid/reorder
reorder-domain
q
q
/define/operating-conditions/gravity
no
q
q
q
/define/operating-conditions/operating-pressure
38251.4
q
q
q
/define/materials/change-create
air
air
yes
ideal-gas
yes
constant
1006.43
no
yes
sutherland
three-coefficient-method
1.716e-05
273.11
110.56
no
no
no
```

```
no
no
no
q
q
q
/define/models/energy
yes
yes
yes
yes
no
q
q
q
/define/models/viscous
kw-sst
yes
q
q
q
/define/boundary-conditions/zone-type
s-2
pressure-inlet
q
q
q
/define/boundary-conditions/zone-type
s-3
pressure-outlet
/define/boundary-conditions/zone-type
s-1
symmetry
q
q
q
/define/boundary-conditions/pressure-inlet
s-2
yes
no
9811.29
no
0
no
255.54
```

```
no
yes
no
yes
1
0.5
q
q
q
/define/boundary-conditions/pressure-outlet
s-3
no
0
no
239.4
no
yes
no
yes
1
0.5
no
no
q
q
q
/solve/set/gradient-scheme
;no
;no
yes
q
q
q
/solve/set/discretization-scheme
pressure
10
omega
0
mom
0
k
0
temperature
0
density
```

```
0
q
q
q
/solve/set/under-relaxation
mom
0.3
q
q
q
/solve/monitors/residual/convergence-criteria
0.00000005
0.000001
0.000001
0.000007
0.000007
0.000007
0.000007
q
q
q
/solve/monitors/residual/plot
yes
q
q
q
/solve/initialize/initialize-flow
q
q
q
/solve/set/equations
temperature
no
q
q
q
solve iterate 50
q
q
q
/solve/set/equations
temperature
yes
q
q
```

```
q
solve iterate 40
q
q
q
/solve/set/under-relaxation
mom
0.7
q
q
q
solve iterate 150
q
q
q
/solve/set/discretization-scheme
pressure
12
omega
6
mom
6
k
6
temperature
6
density
6
q
q
q
solve iterate 200
q
q
q
/solve/set/under-relaxation
pressure
0.6
q
q
q
solve iterate 1400
q
q
q
```

```
/report/forces
wall-forces
no
s-4
()
1
0
0
yes
"C:\Users\Administrator\Desktop\Claudio\Optimization\objvalue.data"
q
q
q
/report/forces
wall-forces
yes
1
0
0
yes
"C:\Users\Administrator\Desktop\Claudio\Optimization\objvalue_total.data"
q
q
q
/report/forces
wall-forces
no
s-4
s-5
()
1
0
0
yes
"C:\Users\Administrator\Desktop\Claudio\Optimization\objvalue_no_tail.data"
q
q
q
/file/stop-transcript
q
q
q
exit
yes
```

C.6 *OpenFOAM-autorun.sh*

This file allows to perform all the necessary instructions in OpenFOAM® environment.

In the first part, mesh importation and its scaling is performed.

Next, the convergence strategy is performed, until the residual criteria are fulfilled.

Finally, the files containing all the information on ERICA nose drag are saved, which are then read by GeDEAII master code.

In Figure 7.83 it corresponds to the *OpenFOAM* round corner square highlighted in gray.

Belove, an excerpt of the *OpenFOAM-autorun.sh* used during ERICA nose optimization described in Part III of Chapter 7 is presented.

```
#!/bin/sh
fluentMeshToFoam ERICA_FullScale_P_-1.970.msh -scale 0.001
OFprocessor=4
OFapplication=rhoSimplecFoam
logName=log
zStep1=800
zStep3=1500
################
# AUTO RUN CODE
################
sed -i 's/\(numberOfSubdomains \).*;/\1 '$OFprocessor';/g' system/decomposeParDict
decomposePar
rm system/fvSchemes
rm system/fvSolution
cp system/schema1o system/fvSchemes
cp system/solution1 system/fvSolution
sed -i 's/\(endTime \).*;/\1 '$zStep1';/g' system/controlDict
mpirun -np $OFprocessor $OFapplication -parallel | tee $logName
cp log log.1

# RUN 2
rm system/fvSchemes
rm system/fvSolution
cp system/schema2o system/fvSchemes
cp system/solution2 system/fvSolution
sed -i 's/\(endTime \).*;/\1 '$zStep3';/g' system/controlDict
mpirun -np $OFprocessor $OFapplication -parallel | tee $logName
cp log log.2
```

C.7 *Student.m*

This Matlab® file performs the Student analysis following the guidelines given in section 7.22 of Part II of Chapter 7.

```
function Student
k=1;
h=1;
g=1;
y=1;
u=0;
v=0;
sommatoria1=0;
sommatoria2=0;
numvar=[9];
a=1:numvar;
numrang=[2];
contatore_n1(1,numvar)=0;
contatore_n2(1,numvar)=0;

  totale='totale.mat';
  load(totale);
  for i=1:59
       for n=1:numvar
        RANGE(1,n)=min(x_current_total(:,n));
        RANGE(2,n)=max(x_current_total(:,n));
       end
  end
  for i=1:numvar
      media(i)=(RANGE(1,i)+RANGE(2,i))/2;
      %[sd,ds]=size(data1);
      for j=1:59
       if x_current_total(j,i)< media(i)
           x_current_total(j,i)=-1;
       elseif x_current_total(j,i) == media(i)
           x_current_total(j,i)=0;
       else
           x_current_total(j,i)=1;
       end
      end
  end
     c='DRAG';
     f_ob=num2str(c);

  cont2=59;
```

```
controllo2=[1];
controllo4=[1];
funct_min_tot(1,numvar)=0;
funct_max_tot(1,numvar)=0;
for j=1:numvar
 for i=1:cont2
       if x_current_total(i,j) ~= 1 %insieme -
           %cont3=cont_ob+numvar;
               controllo=[1];
               funct_min(k,g)=objvalue_total(i,3); %x1i
               funct_min_tot(j)=funct_min_tot(j) + funct_min(k,g);
               g=g+1;
               contatore_n1(j)=contatore_n1(j)+controllo;

        else
           controllo2=[0];
       end
       if x_current_total(i,j) ~= -1  %insieme + ~

               controllo=[1];
               funct_max(h,y)=objvalue_total(i,3); %x2i
               funct_max_tot(j)=funct_max_tot(j) + funct_max(h,y);
               y=y+1;
               contatore_n2(j)=contatore_n2(j)+controllo;

        else
           controllo2=[0];
       end
end
 k=k+1;
 h=h+1;
 g=1;
 y=1;
end
 for i=1:numvar
   funct_min_tot_media(i)=funct_min_tot(i)/contatore_n1(i);
 end
 for i=1:numvar
   funct_max_tot_media(i)=funct_max_tot(i)/contatore_n2(i);
 end
 for i=1:numvar
  numeratore_student(i)=abs(funct_min_tot_media(i)-funct_max_tot_media(i));
 end
 for i=1:numvar
  for j=1:contatore_n1(i)
```

```
                u(j)=((funct_min(i,j)-funct_min_tot_media(i))^2);
                sommatoria1 = sommatoria1+u(j);
     end
     for j=1:contatore_n2(i)
                v(j)=((funct_max(i,j)-funct_max_tot_media(i))^2);
                sommatoria2 = sommatoria2+v(j);
     end
     Standard_deviation(i)=
     sqrt(((sommatoria1+sommatoria2)*(contatore_n1(i)+contatore_n2(i)))/
     ((contatore_n1(i)+contatore_n2(i)-2)*contatore_n1(i)*contatore_n2(i)));
     STUDENT(i)=(numeratore_student(i)/Standard_deviation(i));
     sommatoria1=0;
     sommatoria2=0;
    end
  barh(a,STUDENT);
  d=['Student Analysis relative to ERICA NOSE','   ',f_ob];
  r=char(d);
  title (r);
  ylabel('Decision variable');
  xlabel('Student Parameter');
  figure(1);
  ggg=char(desc(1,foglio));
  bv=[ggg,'_',f_ob,'.jpeg'];
  GeneticAlgorithm.jpeg =...
  ['C:\Documents and Settings\Claudio\Desktop\Per Rita\',bv];
  print('-djpeg',GeneticAlgorithm.jpeg);
   k=1;
   h=1;
   g=1;
   y=1;
   u=0;
   v=0;
   contatore_n1(1,:)=0;
   contatore_n2(1,:)=0;
   funct_min_tot(1,:)=0;
   funct_max_tot(1,:)=0;
   sommatoria1=0;
   sommatoria2=0;
   Titolo=[f_ob];
   gt=(1+cont_ob*4)-4;
   gt_2=gt+1;
   ert=num2str(gt);
   ert2=num2str(gt_2);
   column=['A',ert];
   column_2=['A',ert2];
```

```
    pause(2);
    drawnow;

  RANGE(1,:)=0;
  RANGE(2,:)=0;
end
```

Appendix D

A technical overview of the open-source CFD package OpenFOAM®

D.1 *Introduction*

Nowadays, the open source CFD package OpenFOAM is undergoing at AgustaWestland an extensive numerical campaign in order to demonstrate its capabilities.

The reason why AgustaWestland is strongly motivated to introduce OpenFOAM into its CFD analyses is twofold:

1. OpenFOAM is an entirely open source package, which helps to reduce preliminary design costs.
2. One of the key advantages of OpenFOAM is its extensibility: the source code is accessible, and the architecture of OpenFOAM should make it easy to write a new solver or adapt existing ones. Moreover, this feature may allow the engineer to build new boundary conditions, in order to fulfill exactly both its needs and CFD simulations requirements.

OpenFOAM stands for "Open Source Field Operation and Manipulation"; it is the software being evaluated in the course of test case III. This chapter is intended to give the reader who is unfamiliar with it an idea of OpenFOAM; it is by no means an attempt to give the readers a complete documentation, but just to make simpler the understanding of test case III. For a deeper insight into OpenFOAM features and capabilities, readers are referred to [57] and [58].

Here, only the needed informations to understand the role OpenFOAM played in test case III, are presented.

D.2 *OpenFOAM framework*

OpenFOAM is on the one hand a C++ library, on the other hand a collection of *applications* (created using these libraries). The applications can be divided into two different categories: *solvers* and *utilities*, of which the former perform the actual calculations and the latter provide a range of functionalities for pre- and post-processing.

As far as the *solvers* is concerned, OpenFOAM covers an impressive range of applications with solvers ranging from a simple potential flow solver (*potential-Foam*) over incompressible steady-state (*simpleFoam*), transient laminar (*icoFoam*) turbulent (*turbFoam*) or dynamic mesh (*icoDyMFoam*) solvers, compressible steady-state (*rhoSimpleFoam*) or trans- and supersonic turbulent (sonic- TurbFoam) solvers to multiphase flow solvers (e. g., *interFoam*), LES solvers (*oodles*), combustion codes (*dieselEngineFoam*), electromagnetics (*mhdFoam*), solid stress analysis (*solidDisplace-mentFoam*) and even finance (*financialFoam*) solvers. Naturally, the focus in this thesis lies on the compressible flow steady-state solvers; OpenFOAM comes with several releases of them. For the purposes of the work presented in the chapter dedicated to the test case III, the *rhoSimplecFoam* solver was addressed. Further informations about this solver can be found in the dedicated section.

As far as *utilities* is concerned, they can basically be divided into supporting pre- and post-processing tasks. Utilities are usually called using

```
<utility> <root> <case> [ - optionalParameters ]
```

where `<utility>` is the name of the utility (e. g., *blockMesh*), `<root>` is the path to the root directory, and `<case>` is the path of the actual case, relative to the root directory.

Of the many pre-processing utilities that come with OpenFOAM, the following are used most often in the course of the test case presented in this thesis (see test case III):

- **fluentMeshToFoam** converts Ansys Fluent® meshes to OpenFOAM format. This is used for importing Ansys Fluent® format volume mesh created in Ansys Tgrid® environment.
- **transformPoints** transforms the mesh points in the polyMesh directory according to the translate, rotate and scale options.

The following post-processing utilities are the ones that offer functionality required for this thesis:

- **patchAverage** calculates the Area weighted average of the specified field over the specified patch;
- **yPlusRAS** calculates and reports yPlus for all wall patches, for the specified times when using RAS turbulence models.

Additional utilities used for mesh parallel decomposition used in this thesis are:

- **decomposePar** which automatically decomposes a mesh and fields of a case for parallel execution of OpenFOAM. The number of cores as well as the decomposition methods can be specified by the user into the *decomposeParDict* dictionary present in the *system* directory.

- **reconstructPar** reconstructs a mesh and fields of a case that is decomposed for parallel execution of OpenFOAM.

D.3 *OpenFOAM accessibility*

One of the key advantages of OpenFOAM is its extensibility: the source code is accessible, and the architecture of OpenFOAM should make it easy to write for example a new solver or adapt existing ones. A central theme of the OpenFOAM design is that the solver applications, written using the OpenFOAM classes, have a syntax that closely resembles the partial differential equations being solved. For example the equation

$$\frac{\partial \rho U}{\partial t} + \nabla \cdot \Phi U - \nabla \cdot \mu \nabla U = -\nabla p \tag{D.1}$$

is represented by the code

```
solve
 (
    fvm :: ddt ( rho , U )
  + fvm :: div ( phi , U )
  - fvm :: laplacian ( mu , U )
 ==
  - fvc : grad ( p )
 );
```

This and other requirements demand that the principal programming language of OpenFOAM has object-oriented features such as inheritance, template classes, virtual functions and operator overloading. These features are not available in many languages that purport to be object-orientated but actually have very limited object-orientated capability, such as FORTRAN-90. *C++*, however, possesses all these features while having the additional advantage that it is widely used with a standard specification so that reliable compilers are available that produce efficient executables. It is therefore the primary language of OpenFOAM.

D.4 *Model setup*

Every OpenFOAM case has a similar structure with slight differences stemming only from the particular choice of solver. The basic files needed to correctly set-up a CFD simulation are organized into three main folders corresponding to the following list:

1. A *0* (zero) directory containing individual files of data for particular fields. The data can be either, initial values and boundary conditions that the user must specify to define the problem, or results written to file by OpenFOAM. It is sufficient to say now that since steady state simulations start at time

$t = 0$, the initial conditions are usually stored in a directory named 0. For example, the velocity field U and pressure field p are initialized from files $0/U$ and $0/p$, respectively.

2. A *constant* directory that contains a full description of the case mesh in a subdirectory *polyMesh* and files specifying physical properties for the application concerned, e.g.*transportProperties*. In order to import the Ansys Fluent® format *.cas* file, the command `fluentMeshToFoam <meshFile>` was used, where `<meshFile>` is the name of the *.cas* format file, including the full or relative path. The *thermophysicalProperties* dictionary determines fluid model settings for the equation of state, whether cp is constant or not, what transport model should be used and so on. Typical files present in the current directory are:

    - *thermophysicalProperties*: file where all the needed informations about thermodynamic features are stored.
    - *turbulenceProperties*: file where it is specified whether the turbulence model is of *RAS* (*Reynold-Averaged Stress*) or *LES* (*Large Eddy Simulation*) type.
    - *RASProperties*: file where the turbulence model used for the present simulation is stored ($\kappa - \epsilon$, $\kappa - \omega$, $Spalart - Allmaras$...).
    - *trasportProperties*: file where all the transport properties of the fluid involved in the simulation are stored.
    - *polyMesh*: a subdirectory which is in turn divided into some directories such as:
        (a) *boundary*: file where the needed informations and names of patches where the boundary conditions have to be applied are stored.
        (b) *points, faces, owner, neighbour*: files concerning nodes, faces and mesh connectivity.

3. A *system* directory for setting parameters associated with the solution procedure itself. It contains at least the following 3 files:

    - *controlDict* where run control parameters are set including start/end time, time step and parameters for data output;
    - *fvSchemes* where discretization schemes used in the solution may be selected at run-time;
    - *fvSolution* where the equation solvers, tolerances and other algorithm controls are set for the run.

In Figure D.1, the basic structure of an OpenFOAM case is depicted.

D.5 *0* directory

As stated in section D.4, in *0* folder one can find several files related to the five thermodynamic variables:

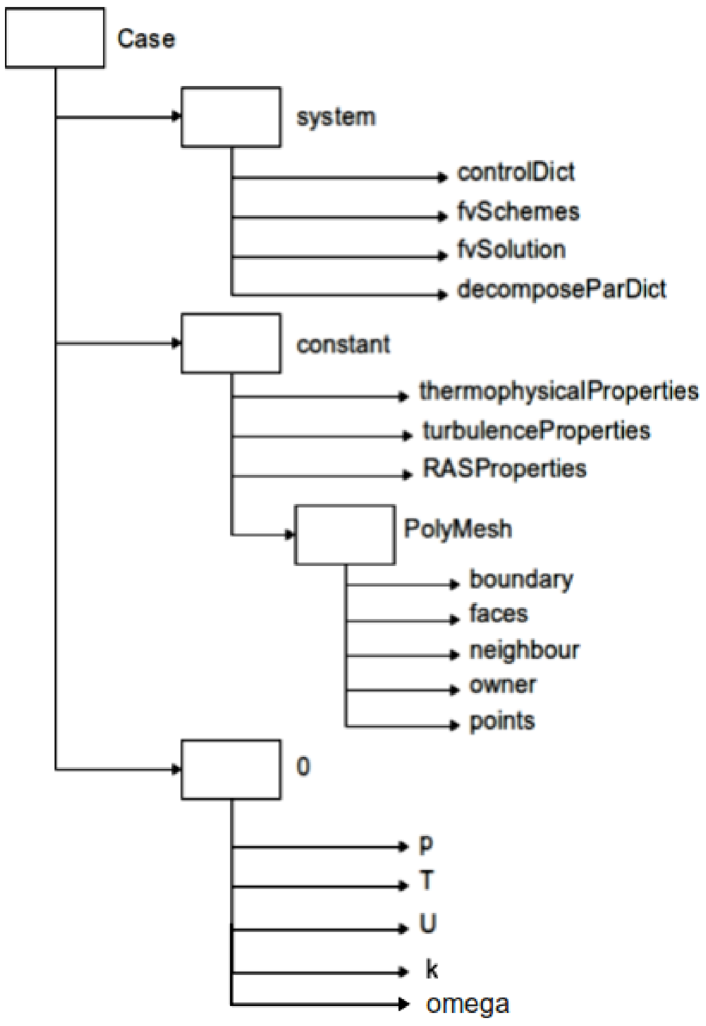


Figure D.1: Typical framework of an OpenFOAM case.

- p;
- U;
- T;
- κ;
- ω;

D.5.1 Variable *p*

Belove, the structure of *p* file is described in detail.

```
dimensions          [1 -1 -2 0 0 0 0];
```

In the *dimensions* entry, the units of measurement of the current variable referring to the System International (SI) is reported in the following order $[Kg\, m\, s\, K\, ...]$.

The pressure is specified in $[Pa]$ (Pascal):

$$1\,Pa = 1\,\frac{N}{m^2} = 1\,\frac{Kg \cdot m}{m^2 \cdot s^2} = 1\,\frac{Kg}{m \cdot s^2}$$

hence the unit of measurement for variable p in this file is *dimensions [1 -1 -2 0 0 0 0]”.*

```
internalField          uniform 98724;
```

The entry *“internalField”* defines the pressure values in each cell of the mesh model. In this way, the flow field related to the pressure is initialized in the entire domain. In this case, the pressure is uniformly initialized throughout the domain with the same value by means of the keyword “uniform” after *“internalField”*.

```
boundaryField
{
```

After this keyword, the boundary condition type and value of the variable is defined

---

```
inlet
{
    type totalPressure;
    p0 uniform 38251.4;
    gamma 1.4;
    U U;
    phi phi;
    rho rho;
    value $internalField;
}
```

Here, the boundary condition type for the *inlet* patch is defined, that is the entry wall of the wind tunnel. As regards pressure, the *totalPressure* b.c. type is used, which defines the total values of the analyzed variable: the total value is specified with *p0*, along with the specific heat ratio γ and the value of the static pressure on the considered patch *value.*

According to this boundary condition, fields *U*, *phi* e *rho* have not to be directly specified since they are automatically calculated by the solver.

The proper boundary condition for the *outlet* patch is specified belove, that is the exit wind tunnel wall. In the current work, a *Dirichlet boundary condition*, referred to as *type fixedValue*, is imposed for the pressure onto this patch until the end of the simulation. According to this boundary condition, a fixed value is imposed for the pressure on every face constituting the patch.

```
outlet
{
    type fixedValue;
    value $internalField;
}
```

Entries *fuselage*, *wf-fearing*, *nose*, *wing-sx*, *tail*, and *sponson-sx* represent the six patches of the ERICA fuselage where the pressure is specified via a *Neumann boundary condition*, referred to as *zeroGradient*, which imposes the normal gradient onto all the cells constituting the patch.

```
fuselage
{
    type zeroGradient;
}

wf-fearing
{
    type zeroGradient;
}

wing-sx
{
    type zeroGradient;
}

nose
{
    type zeroGradient;
}

tail
{
    type zeroGradient;
}

sponson-sx
{
    type zeroGradient;
}
```

Patches *symmetry-plane, up, down, sx* e *symmetry-plane-quad:25* refer to the lateral wind tunnel walls, where a *symmetry* boundary condition, referred to as *simmetryPlane*, is imposed.

```
symmetry-plane
{
    type symmetryPlane;
}
```

```
up
{
    type symmetryPlane;
}
down
{
    type symmetryPlane;
}
sx
{
    type symmetryPlane;
}
symmetry-plane-quad:25
{
    type symmetryPlane;
}
}
```

D.5.2 Variable *T*

For the sake of clarity, only the boundary conditions are introduced hereafter, not described above.

Onto the *inlet* patch, the same boundary condition specified for the variable *p* is defined, now implemented for the variable *T*.

```
dimensions          [0 0 0 1 0 0 0];
internalField         uniform 238.4;
boundaryField
{
inlet
{
    type totalTemperature;
    T0 uniform 239.4;
    gamma 1.4;
    U U;
    phi phi;
    rho rho;
    value $internalField;
}
```

As far as the outlet patch is concerned, the boundary condition referrd to as *inletOutletTotalTemperature* is defined, whose structure is quite similar to the one already presented for the *totalTemperature*, but in addition the *inletOutlet* entry is defined, which is a *zeroGradient* condition when flow is outwards, *fixedValue* when flow is inwards.

```
outlet
{
    type inletOutletTotalTemperature;
    T0 uniform 239.4;
    gamma 1.4;
    U U;
    phi phi;
    rho rho;
    value $internalField;
}
```

The boundary conditions set onto the lateral wind tunnel walls and onto the fuselage walls are the same defined in *p* file.

D.5.3 Variable *U*

The *U* file structure is reported belove:

```
dimensions        [0 1 -1 0 0 0 0];
internalField     uniform (154 0 0);
```

As one can see, the entry *internalField* defines the velocity field via its three-components reference system , that is *x,y,z*. In the current work, an entry velocity of 154 m/s was defined, with only the *x*-component being assigned with a non-zero value.

Onto the *inlet* patch, the boundary condition *pressureInletVelocity* is defined, which is used when the p variable is known at inlet, and U is evaluated from the flux, normal to the patch.

```
boundaryField
{
inlet
{
    type pressureInletVelocity;
    value $internalField;
}
```

At the outlet, the *pressureInletOutletVelocity* is imposed, which is a combination of *pressureInletVelocity* and *inletOutlet*. In this way, one tries to obtain the best integration between velocity and pressure at the outlet.

```
outlet
{
    type pressureInletOutletVelocity;
    inletValue $internalField;
    value $internalField;
}
```

As regards the fuselage walls, a *Dirichlet boundary condition*, referred to as *type fixedValue*, is imposed for the velocity U onto these patches until the end of the simulation. In particular, a no-slip boundary condition is reproduced, by specifying a zero-value for the velocity field.

Belove, an excerpt of U file concerning the fuselage wall boundary condition is reported.

```
fuselage
{
    type fixedValue;
    value uniform (0 0 0);
}
```

For the remaining parts of the fuselage, the same boundary condition is repeated, whereas for the lateral wind tunnel walls a symmetry boundary condition was used.

D.5.4 Variable Turbulence Kinetic Energy κ

Belove an excerpt of the file *k* is reported, with only the boundary conditions not yet mentioned so far: at the inlet, a fixed value for *k* is specified, by means of the boundary condition *fixedValue*, whereas at the outlet the *inletOutlet* boundary condition is used.

```
dimensions          [0 2 -2 0 0 0 0];
internalField         uniform 1.881938;
boundaryField
{
inlet
{
    type fixedValue;
    value $internalField;
}
outlet
{
    type inletOutlet;
    inletValue $internalField;
    value $internalField;
}
```

Onto the fuselage walls, one can impose a fixed value for the *k* field, assuming the fluid being at rest. Otherwise, the wall functions specified with the keyword *compressible::kqRWallFunction* can be imposed. Calculations performed with both the boundary conditions gave the same results.

```
fuselage
{
    type fixedValue;    //compressible::kqRWallFunction;
    value uniform 0;
}
```

Once again, for the lateral wind tunnel walls a symmetry boundary condition was used.

D.5.5 Variable Specific Dissipation Rate ω

Belove, an excerpt of the ω file is reported.

```
dimensions          [0 0 -1 0 0 0 0];
internalField         uniform 17.03824;
boundaryField
{
inlet
{
    type fixedValue;
    value $internalField;
}
outlet
{
    type inletOutlet;
    inletValue $internalField;
    value $internalField;
}
fuselage
{
    type compressible::omegaWallFunction;
    value uniform 0;
}
```

The ω variable requires the imposition of the wall functions onto the fuselage walls. For the purposes of this work, the *compressible::omegaWallFunction* was chosen, which proved to be quite robust, while being sufficiently accurate.

D.6 *Constant* directory

In this section, each file present in the *constant* folder will be described in details.

D.6.1 PolyMesh directory

In this folder, the only file modifiable by the user is the one referred to as *boundary*, whereas the other ones are automatically modified by OpenFOAM when the mesh is created (or imported) or adjusted during the simulations. In particular:

- *points:* represents the coordinate defining each cell vertices.
- *faces*: represents the list of the faces present within the mesh model.
- *owner e neighbour*: represents the way the cells are connected to each other.
- *boundary*: represents the patches where the boundary conditions are specifiable. Belove an excerpt of this file is reported.

```
13

(
wf-fearing
{
    type wall;
    nFaces 23217;
    startFace 12690203;
}
nose
{
    type wall;
    nFaces 20390;
    startFace 12713420;
}
fuselage
{
    type wall;
    nFaces 16282;
    startFace 12733810;
}
```

After *type* entry, the type of boundary condition is required; *nFaces* defines the number of faces of the boundary patch, while *startFace* is the index of the first face of the surface. These entries are created automatically once the mesh model has been created.

Belove, the boundary condition *symmetryPlane* is defined onto the lateral wind tunnel walls. This boundary condition is also useful when unbounded flows have to be simulated.

```
symmetry-plane
```

```
{
    type symmetryPlane;
    nFaces 60648;
    startFace 12896641;
}
```

For the inlet and outlet patches of the wind tunnel a generic *type patch* can be specified.

```
inlet
{
    type patch;
    nFaces 998;
    startFace 12957289;
}
outlet
{
    type patch;
    nFaces 1000;
    startFace 12958287;
}
```

Otherwise, a more suitable boundary condition can be defined, by means of the additional entry *physicalType inlet/outlet*:

```
inlet
{
    type patch;
    physicalType inlet;
    nFaces 998;
    startFace 12957289;
}
outlet
{
    type patch;
    physicalType outlet;
    nFaces 1000;
    startFace 12958287;
}
```

D.6.2 Files *turbulenceProperties* and *RASProperties*

In the *turbulenceProperties* and *RASProperties* files one specifies the type of turbulence model.

turbulenceProperties file looks as follow:

```
simulationType     RASModel;
```

Within this file one can choose between two type of simulations, that is RAS (Reynolds-averaged stress, also referred to as RANS as described in section 4.3.1) and LES (Large-Eddy Simulation).

File *RASProperties* is defined belove:

```
RASModel        kOmegaSST;
turbulence      on;
```

Within this file one can specify the turbulence model used for the RANS simulation.

D.6.3 File *thermophysicalProperties*

This file serves to specify the thermophysical properties of the fluid.

Its structure looks as follow (assuming valid the 2.0.0 release).

```
thermoType     hPsiThermo<pureMixture<sutherlandTransport<
                 specieThermo<hConstThermo<perfectGas>>>>>;
mixture
{
specie
{
    nMoles 1;
    molWeight 28.9;
}
thermodynamics
{
    Cp 1007;
    Hf 0;
}
transport
{
    As 1.4792e-06;
    Ts 116;
}
}
```

At first row, the entry *thermoType* determines the thermal model taken into account, in particular:

- *hPsiThermo*: General thermophysical model calculation based on enthalpy h and compressibility ψ;
- *pureMixture*: General thermophysical model calculation for passive gas mixtures (where no combustion phenomena occur);

- *sutherlandTransport*: Sutherland's formula for temperature-dependent transport properties. It derives the dynamic viscosity μ as a function of temperature T by means of the Sutherland's formula.

- *specieThermo*: Thermophysical properties of species, derived from Constant specific heat Cp, enthalpy h and entropy s.

- *hConstThermo*: Constant specific heat Cp model with evaluation of enthalpy h and entropy s.

- *perfectGas*: Perfect gas equation of state.

The remaining entries serve to specify the numerical values related to the features made in *thermoType* entry:

- *nMoles 1*: number of moles;

- *molWeight 28.9*: molecular weight;

- *Cp 1007:* Constant specific heat;

- *Hf 0*: heat of fusion;

- *As 1.4792e-06* e *Ts 116*: Sutherland's coefficients.

D.7 *system* directory

In this section, each file present in the *system* folder will be described in details.

D.7.1 File *controlDict*

The *controlDict* dictionary sets input parameters *essential* for the simulation run.

These entries are reported step-by-step belove. It is useful to underline here that all the instructions specified in the *controlDict* dictionary remain valid for both steady-state and transient simulations.

```
application     rhoSimplecFoam;
```

The entry *application* defines the solver chosen for the simulation. In test case III, the solver *rhoSimplecFoam* has been used.

The Entry *startFrom* determines the starting temporal folder (in this case, *latestTime* means that the numerical simulation starts from the last created temporal folder), whereas entry *starTime* defines the temporal folder (folder *0*).

```
startFrom      latestTime;
startTime      0;
```

The entry *stopAt* determines how the simulation stops: in this case, simulation stops at the prescribed iteration (runtime modifiable) defined after *endTime* entry, that is 3500.

```
stopAt      endTime;
endTime     3500;
deltaT      1;
```

deltaT entry determines the temporal step in the case of transient simulations, whereas it defines the iteration during steady-state calculations.

The two fields reported belove serve to decide when to save the results during the simulations. In this case, results are saved every 100 iterations.

```
writeControl      timeStep;
writeInterval     100;
```

```
writeFormat         ascii;
writePrecision      6;
writeCompression    compressed;
timeFormat          general;
timePrecision       6;
```

The entry *writeFormat* determines the saving format (in this case ASCII), *write-Precision* defines the number of digits to be retained, in *writeCompression* one can decide whether the results have to be compressed or not, *timeFormat* represents the choice of format of the naming of the time directories, *timePrecision* is an integer used in conjunction with *timeFormat* described above.

Within *controlDict* file, one can define some functions for the aerodynamic forces calculations, by loading some OpenFOAM internal libraries. In the following, the entries needed for calculating the aerodynamic forces acting onto the selected patches, are briefly explained.

```
functions
(
forces
{
type forces;
functionObjectLibs ("libforces.so");
```

The list *patches* defines the patches where the aerodynamic forces and/or coefficients must be calculated.

```
patches
(
fuselage
wf-fearing
wing-sx
nose
tail
sponson-sx
);
```

CofR defines the origin of the calculation of the aerodynamic moments along *x,y,z* axes: (0 0 0) means that this point is located exactly in the origin of the reference system.

```
CofR (0 0 0);
```

The two entries reported belove determine the saving frequency.

```
outputControl timeStep;
outputInterval 10;
```

rhoInf defines the reference density used for the forces calculation.

```
rhoInf 1.1607034104;
pName p;
UName U;
rhoName rhoInf;
}
```

D.7.2 File *decomposeParDict*

The method of parallel computing used by OpenFOAM is known as *domain decomposition*, in which the geometry and associated fields are broken into pieces and allocated to separate processors for solution. The process of parallel computation involves: decomposition of mesh and fields; running the application in parallel; and, post-processing the decomposed case as described in the following sections. The parallel running uses the public domain openMPI implementation of the standard message passing interface (MPI).

For this work purposes, the selected decomposition method was the *scotch*. *Scotch* decomposition requires no geometric input from the user and attempts to minimize the number of processor boundaries. The user can specify a weighting for the decomposition between processors, through an optional *processorWeights* keyword which can be useful on machines with differing performance among processors.

The *decomposeParDict* file collects the following set of parameters needed to properly decompose the domain.

```
numberOfSubdomains 4;
method scotch;
```

```
scotchCoeffs
{
}
```

numberOfSubdomains entry specifies the total number of subdomains the domain has to be decomposed in; the *scotchCoeffs* entry contains a list of weighting factors for allocation of cells to processors.

Another decomposition method is *Simple*, a geometric decomposition in which the domain is split into pieces by direction, e.g. 2 pieces in the x direction, 1 in y etc.

D.7.3 File *fvSchemes*

The *fvSchemes* dictionary in the system directory sets the numerical schemes for terms, such as derivatives in equations, that appear in applications being run. This section describes how to specify the schemes in the *fvSchemes* dictionary.

The terms that must typically be assigned a numerical scheme in *fvSchemes* range from derivatives, e.g. gradient ∇, and interpolations of values from one set of points to another one.

Below, the *fvSchemes* file refers to a case where simulation is performed using 1^{st} order discretization schemes. Main keywords used in fvSchemes are:

- ***ddtSchemes***

  ```
  ddtSchemes
  {
  default        steadyState;
  }
  ```

  The *ddtSchemes* field is defined within the *timeSchemes* environment, where discretization schemes for the first ($\frac{\partial}{\partial t}$) time derivative are defined. Discretization schemes for the second ($\frac{\partial^2}{\partial t^2}$) time derivative are defined within the *d2dt2Schemes* field.

  In Table D.1, available schemes for *ddtSchemes* are presented, whereas only *Euler* scheme is available for *d2dt2Schemes* (second order schemes).

Schemes	Description
Euler	First order, bounded, implicit
CrankNicholson	Second order, implicit, bounded
backward	Second order, implicit
steadyState	Does not solve for time derivatives

Table D.1: Discretization schemes available in *ddtSchemes*.

- ***interpolationSchemes***

  The *interpolationSchemes* sub-dictionary contains terms that are interpolations of values typically from cell centres to face centres. A selection of interpolation schemes in OpenFOAM are listed in Table D.2, being divided into four categories: one category of general schemes and three categories of schemes used primarily in conjunction with Gaussian discretization of convection (divergence) terms in fluid flow.

Interpolation schemes	Definition
Centred schemes	
linear	Linear interpolation (central differencing)
cubicCorrection	Cubic scheme
midPoint	Linear interpolation with symmetric weighting
Upwinded convection schemes	
upwind	Upwind differencing (1st order)
linearUpwind	Linear upwind differencing (1$st - 2nd$ order)
skewLinear	Linear with skewness correction
QUICK	2nd order scheme
TVD schemes	
limitedLinear	limited linear differencing
vanLeer	vanLeer limiter
MUSCL	MUSCL limiter
limitedCubic	Cubic limiter
NVD schemes	
SFCD	Self-filtered central differencing
Gamma ψ	Gamma differencing

Table D.2: Interpolation schemes.

- ***gradSchemes***

  gradSchemes

  {

  default *Gauss linear;*

  grad(p) *Gauss linear;*

  }

  The *gradSchemes* sub-dictionary contains gradient terms. The discretization scheme for each term can be selected from those listed in Table D.3.

Schema	Descrizione
Gauss <interpolationSchemes>	Second order, Gaussian integration
leastSquares	Second order, least squares
fourth	Fourth order, least squares
cellLimited <gradSchemes>	Cell limited version of one of the above schemes
faceLimited <gradSchemes>	Face limited version of one of the above schemes

Table D.3: Discretization schemes available in *gradSchemes*.

As one can see at the first line of Table D.3, (*Gauss <interpolationSchemes>*), after defining the gradient scheme as in the following

---

```
gradSchemes
{
default        Gauss ..... ;
grad(p)        Gauss ..... ;
}
```

the interpolation scheme must be specified, in our example *linear*:

```
gradSchemes
{comelico
default       Gauss linear;
grad(p)       Gauss linear;
}
```

Limited versions of any of the three base gradient schemes (*cellLimited* or *face-Limited*) can be selected, by preceding the discretization scheme by *cellLimited* (or *faceLimited*), e.g. a *cell limited Gauss* scheme.

```
grad(p)       cellLimited Gauss linear;
```

- ***divSchemes***

```
divSchemes
{
default                          Gauss upwind;
div(phi,U)                       Gauss upwind;
div((muEff*dev2(grad(U).T())))     Gauss linear;
}
```

  The *divSchemes* sub-dictionary contains divergence terms. The Gauss scheme is the only choice of discretization and requires a selection of the interpolation scheme for the dependent field. The interpolation scheme is selected from the full range of schemes in Table D.4, both general and convection-specific.

- ***snGradSchemes***

  The *snGradSchemes* sub-dictionary contains surface normal gradient terms. A surface normal gradient is evaluated at a cell face; it is the component, normal to the face, of the gradient of values at the centres of the two cells that the face connects with. A surface normal gradient may be specified in its

Scheme	Numerical behaviour
linear	Second order, unbounded
skewLinear	Second order, (more) unbounded, skewness correction
cubicCorrected	Fourth order, unbounded
upwind	First order, bounded
linearUpwind	First/second order, bounded
QUICK	First/second order, bounded
TVD schemes	First/second order, bounded
SFCD	Second order, bounded
NVD schemes	First/second order, bounded

Table D.4: Divergence schemes available in *divSchemes*.

Schema	Description
corrected	Explicit non-orthogonal correction
uncorrected	No non-orthogonal correction
limited ψ	Limited non-orthogonal correction
bounded	Bounded correction for positive scalars
fourth	Fourth order

Table D.5: Surface normal gradient schemes available in *snGradSchemes*.

own right and is also required to evaluate a Laplacian term using Gaussian integration. The available schemes are listed in Table D.5.

A *limited* scheme can be selected by specifying ψ term defined as described in Table D.6:

ψ	Description
0	corresponds to uncorrected
0.333	non-orthogonal correction $\leq 0.5 \times$orthogonal part
0.5	non-orthogonal correction $\leq$ orthogonal part
1	corresponds to corrected

Table D.6: Limiter coefficients available in limited schemes of *snGradSchemes*.

- ***laplacianSchemes***

  laplacianSchemes

  {

  default   Gauss linear corrected;

  }

  The *laplacianSchemes* sub-dictionary contains Laplacian terms. The Gauss scheme is the only choice of discretization and requires a selection of both an interpolation scheme for the diffusion coefficient, i.e. ν in our example, and a surface normal gradient scheme, i.e. ∇U. To summarize, the entries required are:

Gauss <interpolationScheme> <snGradScheme>$\longrightarrow$*Gauss linear corrected*

The interpolation scheme is selected from Table D.7, the typical choices being from the general schemes and, in most cases, linear.

Scheme	Numerical behaviour
corrected	Unbounded, second order, conservative
uncorrected	Bounded, first order, non-conservative
limited ψ	Blend of corrected and uncorrected
bounded	First order for bounded scalars
fourth	Unbounded, fourth order, conservative

Table D.7: Surface normal schemes available in *laplacianSchemes*.

Finally, *fvSchemes* allows the user to take into account a source term after solving a pressure equation:

```
fluxRequired
{
default      no;
p ;
}
```

D.7.4 File *fvSolution*

The equation solvers, tolerances and algorithms are controlled from the *fvSolution* dictionary in the system directory.

fvSolution contains a set of subdictionaries that are specific to the solver being run. These subdictionaries include *solvers*, *relaxationFactors*, *PISO* and *SIMPLE* which are briefly described in the remainder of this section.

The term *solvers* specify each linear-solver that is used for each discretized equation.

A second sub-dictionary of *fvSolution* that is often used in OpenFOAM is *relaxationFactors*, which controls under-relaxation, a technique used for improving stability of a computation, particularly in solving steady-state problems. Under-relaxation works by limiting the amount which a variable changes from one iteration to the next, either by modifying the solution matrix and source prior to solving for a field or by modifying the field directly. A relaxation factor α ranges from 0 to 1.

The limiting case where $\alpha = 0$ represents a solution which does not change at all with successive iterations. It corresponds to the more stable situation. On the other hand, the limiting case where $\alpha = 1$ represents a situation where no under relaxtion is applied. An optimum choice of α is one that is small enough to ensure stable computation but large enough to move the iterative process forward quickly.

The third subdictionary addresses PISO and SIMPLE algorithms. The pressure-implicit split-operator (PISO) or semi-implicit method for pressure-linked equations (SIMPLE) algorithms. These algorithms are iterative procedures for solving equations for velocity and pressure, PISO being used for transient problems and SIMPLE for steady-state. Hence, in test case III, the SIMPLE algorithm was addressed. Both algorithms are based on evaluating some initial solutions and then correcting them. SIMPLE only makes 1 correction whereas PISO requires more than 1, but typically not more than 4.

Below an example set of entries from the *fvSolution* dictionary required for the *rhoSimplecFoam* solver is reported. For the sake of clarity, the *p* solver is reported only.

```
solvers
{
p0
{
solver PBiCG;
preconditioner DILU;
tolerance 1e-08;
relTol 0.01;
}
p
{
solver GAMG;
tolerance 1e-08;
relTol 0.1;
smoother GaussSeidel;
nPreSweeps 0;
nPostSweeps 2;
nFinestSweeps 2;
cacheAgglomeration off;
nCellsInCoarsestLevel 20;
agglomerator faceAreaPair;
mergeLevels 1;
}
....
....
....
SIMPLE
{
```

```
nNonOrthogonalCorrectors 0;
pMin pMin [ 1 -1 -2 0 0 0 0 ] 100;
rhoMin rhoMin [ 1 -3 0 0 0 0 0 ] 0.1;
rhoMax rhoMax [ 1 -3 0 0 0 0 0 ] 2;
}
relaxationFactors
{
default 0.9;
p 0.9;
rho 0.9;
}
relaxationFactors0
p  0.3;
rho   0.1;
U  0.7;
h  0.7;
k  0.7;
omega  0.7;
}
```

Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor Prof. Ernesto Benini for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and was a source of inspiration.

Besides my advisor, I would like to thank Ing. Rita Ponza and Ing. Antonio Saporiti, for their encouragement, insightful comments, and hard questions.

I thank my fellow labmates, in particular Ing. Marco Manente, Ing. Marta Lazzarin, Ing. Davide Rondini, Ing. Andrea Garavello and M.Sc. candidate Enrico Fasolo for the stimulating discussions, and technical supports.

Warm thanks to all of the colleagues of the AgustaWestland Aerodynamics Office, for their precious guides and advices.

I would like to acknowledge whoever shared and will share with me the moments of my life.

Last but not the least, I would like to thank my family, for its moral (and financial!) support, you are my actual surplus value.

Claudio Comis Da Ronco
San Nicolò di Comelico, 2012

Bibliography

[1] AGARD. Aircraft drag prediction and reduction. *Advisory group for aerospace research and development, AGARD REPORT No.723. pp. 1*, 1985.

[2] R. B. Agrawal and K. Deb. Simulated binary crossover for continuous search space. *Complex Systems, 9, pp. 115-148.*, 1994.

[3] Altair Engineering. HyperMesh & BatchMesher User's Guide. 2010.

[4] Altair Engineering. Hyperstudy User's Guide. 2010.

[5] M. Amir. Application of piezoelectric actuators at subsonic speeds. *Journal of Aircraft*, 45(4):1419–1430, 2008.

[6] M. Amitay. Aerodynamic flow control using synthetic jet actuators. *Control-of-Fluid-Flow, Springer-Verlag, Berlin, Germany, ( ) pp. 45.*, 2006.

[7] Ansys Inc. FLUENT User's Guide, release 13.1. 2011.

[8] Ansys Inc. Tgrid User's Guide, release 13.1. 2011.

[9] T. Bäck. Selective pressure in evolutionary algorithms: A characterization of selection mechanisms. *In Proceedings of the First IEEE Conference on Evolutionary Computation*, pages 57–62, 1994.

[10] T. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, 1996.

[11] T. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

[12] G. K. Batchelor. *An Introduction to Fluid Dynamics*. February 2000.

[13] K.D. Baverstock. US 101. CFD Modeling Engine Intake Distortions and Total Pressure Losses. *Document No: EU54G0001W, Issue 3.*

[14] E. Benini and A. Toffolo. *Genetic Diversity as an Objective in Multi-Objective Evolutionary Algorithms*. Evolutionary Computation 11(2), 151-157, 2002.

[15] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler. *PISA - A Platform and Programming Language Independent Interface for Search Algorithms*. Springer, 2002.

[16] C. Geuzaine and J.F. Remacle. Gmsh Reference Manual. 2010.

[17] W. Calarese, W. Pan Crisler, and G. L. Gustafson. Afterbody drag reduction by vortex generators. *AIAA-85-0354.*, 1985.

[18] F.M. Catalano. On the effects of an installed propeller slipstream on wing aerodynamic characteristics. *Acta Polytechnica Vol. 44 No. 3, Czech Technical University Publishing House*, 2004.

[19] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[20] C. Comis Da Ronco. Aerodynamic noise in aero-engine LP turbines - Application of cfd techniques. *MSc. Thesis*, University of Padua, 2008.

[21] D. W. Corne, J. D. Knowles, and M. J. Oates. *The pareto envelope-based selection algorithm for multiobjective optimisation.* In M. S. et al. (Ed.), Parallel Problem Solving from Nature – PPSN VI, Berlin, pp. 839-848. Springer., 2000.

[22] K. Deb. *Multi-objective genetic algorithms: Problem difficulties and construction of test problems*. Evolutionary Computation, 7(3):205-230, 1999.

[23] K. Deb. *Multi-objective Optimization using Evolutionary Algorithms*. John Wiley& Sons, Chichester, UK., 2001.

[24] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. *A fast elitist nondominated sorting genetic algorithm for multi-objective optimization: NSGA-II.* In M. S. et al. (Ed.), Parallel Problem Solving from Nature – PPSN VI, Berlin, pp. 849-58. Springer., 2000.

[25] K. Deb and M. A. Goyal. Combined Genetic Adaptive Search (GeneAS) for Engineering Design. *Computer Science and Informatics*, 26:30–45, 1996.

[26] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable test problems for evolutionary multi-objective optimization. *Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology*, 2001.

[27] A. E. Eiben and T. Bäck. *Empirical Investigation of Multiparent Recombination Operators in Evolution Strategies*. Evolutionary Computation, 5(3), pp. 347-365, 1997.

[28] S. Farokhi. Aircraft propulsion. *John Wiley & Sons*, 2008.

[29] R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *The Computer Journal*, 7:149–154, 1964.

[30] D. B. Fogel. *Evolutionary computation: toward a new philosophy of machine intelligence*. IEEE Press, Piscataway, NJ, USA, 1995.

[31] C. M. Fonseca and P. J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In *Genetic Algorithms: Proceedings of the Fifth International Conference*. Morgan Kaufmann, 1993.

[32] C. M. Fonseca, L. Paquete, and M. L. Ibáñez. *An improved dimension-sweep algorithm for the hypervolume indicator*. IEEE Congress on Evolutionary Computation, pp. 1157-1163, Vancouver, Canada, 2006.

[33] Von Karman Institute for Fluid Dynamics. Flow control: Fundamentals, advances and applications. 2009.

[34] E. Gabriel. Drag estimation of v/stol aircraft. *Boeing Vertol Report D8-2194-1.*, 1968.

[35] M. Gad-el-Hak. *Flow Control*. Cambridge University Press, 2000.

[36] A. Gessow and G. Myers. *Aerodynamics of the Helicopter*. Frederick Ungar Publishing, New York, 1967.

[37] G. Giancamilli, F. Nannoni, and M. Cicale. Erica: the european advance tilt-rotor. *In European Rotorcraft Forum*, 2001.

[38] G. Godard and M. Stanislas. Control of a decelerating boundary layer. part 3: Optimization of round jets vortex generators. *Aerospace Science and Technology*, 10(6):455 – 464, 2006.

[39] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, 1 edition, jan 1989.

[40] A. A. Hassan, P. B. Martin, C. Tung, D. Cerchie, and J. Roth. Active flow control measurements and cfd on a transport helicopter fuselage. *AHS International, v 1, 61st Annual Forum Proceedings - AHS International, p 349-369*, 2005.

[41] J. Hefner and D. Bushnell. *An Overview of Concepts for Aircraft Drag Reduction*. Special Course on Concepts for Drag Reduction, AGARD Report No. 654, AGARD Special Course, von Karman Institute, 1977.

[42] A. L. Heyes and D. A. R. Smith. Modification of a wing tip vortex by vortex generators. *Aerospace Science and Technology*, 9(6):469–475, 2005.

[43] P. Hill and C. Peterson. *Mechanics and Thermodynamics of Propulsion*. Addison-Wesley Publishing Company, 1992.

[44] B.J. Holmes and C.J. Obara. Observations and implications of natural laminar flow on practical airplane surfaces. *Journal of Aircraft*, 20(12):993–1006, 1982.

[45] G. Hong, C. Lee, and Q. Ha. Effectiveness of synthetic jets enhanced by instability of tollmien-schlichting waves. *AIAA Paper*, 2002.

[46] R. Hooke and T. A. Jeeves. Direct search' solution of numerical and statistical problems. Journal of the ACM, 8(2):pp. 212-229, 1961.

[47] E. L. Houghton and A.E. Brock. *Aerodynamics for engineering students*. Edward Arnold, 2. ed., 1970.

[48] R. D. Joslin. Aircraft laminar flow control. *Annual Review of Fluid Mechanics*, 30(1):1–29, 1998.

[49] E. Kagambage and G. Dimitriadis. Design, manufacture and instalation of a scaled model of the nicetrip engine nacelle in the ulg wind tunnel. *Technical Report N SOUF-NICETRIP-08-01*, University of Liege, Departmente of Aerospace and Mecanics Aeroelasticity and Experimental Aerodynamics Lab, 2008.

[50] J. Kennedy and R. Eberhart. Particle swarm optimization. *Proceedings., IEEE International Conference on Neural Networks*, 4:1942–1948, August 1995.

[51] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science, Number 4598, 13 May 1983*, 220, 4598:671–680, 1983.

[52] J. Knowles, D. Corne, and K. Deb. *Multiobjective Problem Solving from Nature: From Concepts to Applications (Natural Computing Series)*. Springer-Verlag New York, Inc., 2006.

[53] D. Küchemann and J. Weber. *Aerodynamics of Propulsion*. New York: McGrawHill, 1993.

[54] F. Kursawe. *A variant of evolution strategies for vector optimization*. In Schwefel, H. P. and Manner, R., editors, *Parallel Problem Solving from Nature. 1st Workshop, PPSN I, volume 496 of Lecture Notes in Computer Science*, pages 193-197, Berlin, Germany. Springer-Verlag., 1991.

[55] H. J. Lamousin and W. N. Waggenspack Jr. Nurbs-based free-form deformations. *IEEE Computer Graphics and Applications*, 14:59–65, 1994.

[56] B. E. Launder and B. I. Sharma. Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc. *Letters in Heat and Mass Transfer*, pages 131–138, 1974.

[57] OpenCFD Limited. Openfoam: The open source cfd toolbox. Programmer's Guide Version 2.0.0. *Reading UK.*, 2011.

[58] OpenCFD Limited. Openfoam: The open source cfd toolbox. User Guide Version 2.0.0. *Reading UK.*, 2011.

[59] J.C. Lin, S.K.Robinson, R.J. McGhee, and W.O Valarezo. Separation control on high-lift airfoils via micro-vortex generators. *Journal of Aircraft*, 31(6):pp. 1317–1323, 1994.

[60] N. Marco and J.A. Desideri. Multilevel parametrization for aerodynamical optimization of 3d shapes. *INRIA Research Report no. 3622*, 1999.

[61] F. R. Menter. Zonal two-equation $\kappa - \omega$ turbulence models for aerodynamic flows. *AIAA Journal*, page 2906, 1993.

[62] R. Von Mises. Theory of flight. 1959.

[63] K. Mossi, N. Castro, R. Bryant, and P. Mane. Boundary condition effects on piezo-synthetic jets. *Integrated Ferroelectrics*, 71(1):257–266, 2005.

[64] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, January 1965.

[65] I. Ono and S. Kobayashi. *A Real-coded Genetic Algorithm for Function Optimization Using Unimodal Normal Distribution Crossover*. Proceedings of the Seventh International Conference on Genetic Algorithms, pp. 246-253., 1997.

[66] G. Pagnano. Il convertiplano erica: evoluzione della ricerca nei programmi quadro europei. *Agusta coordinamento R&T,*CRUI Viaggio della Ricerca in Italia, Milano, 2005.

[67] Politecnico di Milano. Nicetrip: Final complete wind tunnel test data base for aircraft sizing, suitable for use by the codes of the partners. *NICETRIP/POLIMI/WP4.TR007/4.0,* Appendix C, Company Restricted., 2008.

[68] S. B. Pope. *Turbulent Flows*. Cambridge University Press, Cambridge, UK, 2000.

[69] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, January 1964.

[70] K. Krishna Prasad, K.S. Choi, and T. V. Truong. *Emerging techniques in drag reduction*. John Wiley & Sons, 1996.

[71] C. Reid. The response of axial flow compressors to intake flow distortions. *ASME Paper 69-GT-29*, 1969.

[72] J. Reneaux and J. Preist. Control of the attachment line contamination. *2nd European Forum on Laminar Flow Technology, Bordeaux, France*, 1996.

[73] Green Rotorcraft Technical Report. Erica: analysis of the experimental aerodynamic data of the nicetrip wind tunnel test campaign (1/8 scale force model). *Report AgustaWestland TLRG0000K018, rev. A*, 2009.

[74] Green Rotorcraft Technical Report. Specification of geometrical constraints and of the design point for the optimisation task, together with the revised erica catia file. *Document N° CSJU/ITD GRC/RP/2.2.3/32023*, 2010.

[75] J.P. Robert. *Drag reduction: an industrial challenge. Special Course on Skin Friction Drag Reduction*. AGARD-R-. 786, Paper No. 2, 1992.

[76] H. H. Rosenbrock. An Automatic Method for Finding the Greatest or Least Value of a Function. *The Computer Journal*, 3(3):175–184, March 1960.

[77] RTO. Performance prediction and simulation of gas turbine engine operation. *Technical Report 44, NATO*, 2002.

[78] J. A. Samareh. A survey of shape parameterization techniques. *CEAS AIAA ICASE NASA Langley International Forum on Aeroelasticity and Structural Dynamics*, 1999.

[79] J. Seddon and L. Goldsmith. *Intake Aerodynamics*. Blackwell Science, 2 edition, 1999.

[80] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. *Proceedings SIGGRAPH '86*, 20:151–160, 1986.

[81] A. Seifert. Oscillatory blowing: a tool to delay boundary-layer separation. *AIAA-Journal, 31(11) pp. 2052-2060*, 1993.

[82] R.L. Simpson. Junction flows. *Annual Rev. Fluid Mech. 33, pp. 415-445*, 33:415–445, 2001.

[83] J. C. Spall. Introduction to stochastic search and optimization: Estimation, simulation, and control. *Journal of the American Statistical Association*, 99:1204–1205, December 2004.

[84] W. Spendley, G. R. Hext, and F. R. Himsworth. Sequential application of simplex designs in optimization and evolutionary operation. *Technometrics*, 4:441–461, 1962.

[85] E. Stanewsky. Adaptive wing and flow control technology. *Progress in Aerospace Sciences*, 37(7):583–667, 2001.

[86] O. Stein, A. M. Kempf, and J. Janicka. Les of the sydney swirl flame series: An initial investigation of the fluid dynamics. *Combustion Science and Technology*, 179:173–189(17), 2007.

[87] The Mathworks Inc. Genetic algorithm and direct search toolbox, theory guide,. *2_nd edition*, 2010.

[88] The Mathworks Inc. Optimization toolbox, theory guide,. *2_nd edition*, 2010.

[89] S. W. A. Thomas. Aircraft drag reduction technology - A summary. *in AGARD, Aircraft drag prediction and reduction*, 1985.

[90] S. Tsutsui and A. Ghosh. *A Study on the Effect of Multi-parent Recombination in Real Coded Genetic Algorithms. Proceedings of the 1998 IEEE ICEC*, pp. 828-833., 1998.

[91] S. Tsutsui, M. Yamamura, and T. Higuchi. *Multi-parent Recombination with Simplex Crossover in Real Coded Genetic Algorithms*. Procc. of the GECCO-99, pp. 657-644., 1999.

[92] J. Tu, G.H. Yeoh, and C. Liu. *Computational Fluid Dynamics: A Practical Approach*. Butterworth-Heinemann, Newton, MA, USA, 2007.

[93] A. Vecchio, M.Kurowski, and A. D'Alascio. Synthesis report on technology review of active devices for blunt fuselage drag reduction. (first draft version including technology review on implementation aspects of synthetic or pulsed jets). *CLEAN SKY - Green Rotorcraft ITD - GRC2 Deliverable GRC-D22.1-2A.*, 2008.

[94] H. K. Versteeg and W. Malalasekera. *An Introduction to Computational Fluid Dynamics*. Longman Scientific and Technical, 1995.

[95] P. Viswanath. Aircraft viscous drag reduction using riblets. *Progress in Aerospace Sciences*, 30(30):571–600, 2002.

[96] M.J. Walsh. Riblets as a viscous drag reduction technique. *AIAA Journal*, 21(4):485, 1983.

[97] C. Warsop, M. Hucker, and A. J. Press. Pulsed air-jet actuators for flow separation control. *Flow, Turbulence and Combustion*, 78(3-4):pp. 255–281, 2007.

[98] D. You and P. Moin. Large-eddy simulation of flow separation over an airfoil with synthetic jet control. *Annual Research Briefs*, Center for Turbulence Research:337–346, 2006.

[99] T. Young. Investigation of hybrid laminar flow control (hlfc) surfaces. *Aircraft Design*, 4(2-3):127–146, 2001.

[100] G. Zilli, , and M. Venturin. Metodi di line-search. *Lecture notes, Dipartimento di Metodi e Modelli Matematici Università di Padova*, 2006.

[101] G. Zilli, L. Bergamaschi, and M. Venturin. Metodi di ottimizzazione. *Lecture notes, Dipartimento di Metodi e Modelli Matematici Università di Padova*, 2008.

[102] E. Zitzler, K. Deb, and L. Thiele. *Comparison of multiobjective evolutionary algorithms: Empirical results*. Evolutionary Computation 8(2), 173-195., 2000.

[103] E. Zitzler and S. Künzli. *Indicator-Based Selection in Multiobjective Search*. In *Parallel Problem Solving from Nature (PPSN VIII)*, X. Yao et al., Eds. Berlin, Germany: Springer-Verlag, 2004, pp. 832-842., 2004.

[104] E. Zitzler, M. Laumanns, and L. Thiele. *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland,May 2001., 2001.