RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ MOSTEFA BENBOULAID BATNA 2

FACULTE DE TECHNOLOGIE
Département de Génie Mécanique
Filière : Aéronautique

MEMOIRE DE FIN D'ETUDES

Présenté
pour Obtenir le Diplôme de **Master**

SPÉCIALITÉ :
PROPULSION AÉRONAUTIQUE

Par
HALIMI Yazid
ZAOUCHE Abdelhakim

Thème

Influence d'une nouvelle forme de winglet sur l'écoulement autour d'une aile d'avion

Soutenu le 14/07/2019

Encadré par : Dr. Laïd MESSAOUDI

ANNEE UNIVERSITAIRE 2018/2019

ملخص

في هذه الدراسة لمعرفة تأثير شكل جديد من الجناح على التدفق حول جناح الطائرة أثناء الإقلاع والهدف من هذه الدراسة هو فهم الظواهر الديناميكية الهوائية الناجمة عن هذا التدفق معقد من خلال محاكاة البرمجيات الحرة ، فإن التدفق يعتبر ثابتًا ومضطربًا باستخدام نموذج الاضطراب الأنسب لهذا النوع من التدفق

Abstract

It is in this study to see the effect of a new form of wiglet on the flow around an aircraft wing during takeoff The objective of this study is the understanding of the aerodynamic phenomena brought into play by this flow complex through free software simulation Flow will be considered stationary and turbulent using the most appropriate turbulence model for this kind of flow

Résumé

Il s'agit dans cette étude de voir l'effet d'une nouvelle forme de winglet sur lécoulement autour d'une aile d'avion pendant le décollage L'objectif de cette étude est la compréhension des phénomènes aérodynamiques mis en jeux par cet écoulement complexe à travers la simulation par le logiciel libre L'écoulement sera considéré comme stationnaire et turbulent en utilisant le modèle de turbulence le plus adéquat pour ce genre d'écoulemen

Remerciement

En premier lieu, nous remercions notre Dieu qui nous ont menés sur la bonne voie en matière de science et de connaissance.
Nous voudrions remercier remercier mon promoteur ***Dr : Laïd MESSAOUDI*** *qui nous faisons profiter de son expérience et de ses précieux conseils tout au long de ce travail et dans le domaine de la recherche scientifique.*
Nous espérons que vous serez satisfait de ce travail et nous exprimons notre profonde gratitude et notre reconnaissance.
Nous remercions également les membres du jury qui seront amenés à juger ce modeste travail.
Nous tenons également à remercier tous les enseignants ayant contribué à notre formation depuis le tronc commun jusqu'à la dernière année de graduation

Dédicaces

A mes cher parents

A mes cher frères et mes cher sœurs

A toute ma famille

A tout mes amis et collègues

Yazid,Abdelhakim

Table des matières

LISTE DES FIGURES

LISTE DES TABLEAUX

Introduction générale

Introduction générale

L'origine de la conception des premiers modèles d'aéronefs fut initiée en 1808 par *Lord Cayley*, qui a introduit pour la première fois d'idée de l'aile fixe et des profils avec cambrure afin de générer une portance capable de maintenir l'appareil en l'air. Cependant la mise en oeuvre de ces idées doit attendre encore un siècle environ, avant l'apparition des premiers engins volants.
Les premiers travaux ont eu lieu avant 1884 par l'anglais *Hastrio Philips*, qui a progressé la forme des profils, en utilisant des séries de profils à double surface et des profils cambrés.
Il a fallu ensuite attendre, que les frères *Wright*, après leur première expérience sur les planeurs, se rendent compte en 1901 que beaucoup des données sur les profils ont été inadéquate avec l'expérience. C'est pour cela qu'ils ont construit leur propre soufflerie où ils ont essayé une centaine de profils pour finalement opter pour leur premier essai en vol pour un profil ayant une cambrure de 1/20 de la corde.

En 1920, l'aérodynamicien allemand, *L. Prandtl* élabora la théorie exacte de la sustentation de l'aile et l'application du comportement de l'écoulement d'air. C'est également lui qui introduisit la notion de la couche limite qui a été fortement développée plus tard.

L'aérodynamique est l'une des branches de la mécanique des fluides. Elle est spécialement réservée aux études de l'écoulement de l'air et plus pratiquement autour d'obstacles. C'est surtout autour de profils d'aile que ses applications deviennent de plus en plus étendues (turbomachines, ailes d'avions, ...etc.).

Le développement de l'aérodynamique a suivi celui d'autres sciences telles que l'informatique avec l'apparition d'ordinateurs de plus en plus puissant, malgré leur coût, les techniques expérimentales (essais en souffleries) et bien sûr, les mathématiques avec leurs grands progrès et contribution en techniques numériques pour la résolution en mécanique des fluides des équations de *Navier-Stokes* généralisées.

Dans la mécanique des fluides, un écoulement autour d'un corps de géométrie quelconque peut être générer par une distribution de singularités qui peuvent être localisées dans des positions assez spécifiques de telle sorte que la surface du corps devient une ligne de courant de l'écoulement. Ce dernier peut être finalisé en remplaçant la surface par une distribution continue d'une combinaison de sources-puits sous forme de maillage dense. Si le corps génère une force de sustentation, l'introduction de vortex est la plus souhaitée (Vortex scheet) afin de prévoir le phénomène de circulation dont l'intensité est fixée par la condition de Kutta car la détermination des caractéristiques aérodynamique d'une surface ainsi que les effets du sillage sur les extrémités

des ailes nécessitent en générale l'utilisation des méthodes numériques pour simuler et comparer les résultats de différentes conception et configuration d'aile.

La solution exacte ou approchée de l'équation intégrale dans la théorie des écoulements à potentiel continue pose plusieurs difficultés pour le calcul des champs d'écoulement autour de corps à profil quelconque. Cependant, une nouvelle approche pour solutionner l'équation à potentiel peut être accomplie par une méthode appelée méthode des panneaux (Panel Method ou méthode intégrale - méthode des singularités) qui transforme l'intensité de la singularité sur un élément spécifique en une fonction de courant ou potentiel, au lieu d'une distribution continue de singularités sur la surface du corps. La sommation des fonctions panneaux sur toute la surface permet ainsi de trouver un système d'équations algébriques facile à résoudre.

La simulation numérique de l'écoulement autour d'une aile d'avion munie de différentes winglets durant la phase de décollage a été déjà initiée par Ahmed Zekkour [1] en utilisant les logiciels commerciaux « *Ansys* » pour la conception et « *Star CCM+* » pour la simulation. Dans cette étude, 3 types de winglet on été étudiées et comparées avec celle d'une aile sans winglet : *Demi-circulaire*, *Whitcomb* et *Spiroid*. Un nouveau modèle dit « *DoubleSpiroid* » a été proposé par A. Zekkour et n'a pas été étudié par manque de temps.

Notre objectif est, d'une part, de reprendre cette étude numérique ainsi que la compréhension des phénomènes aérodynamiques mis en jeux par cet écoulement complexe mais avec des logiciels différents issus du monde libre « *Linux* ». D'autre part, de voir l'effet de la nouvelle forme de winglet sur l'écoulement autour de l'aile pendant le décollage.
Les outils gratuits que nous utiliserons dans ce travail sont: le logiciel de conception 3D paramétrique *« FreeCAD »*, le mailleur « *cfMesh »*, le solveur volumes finis « *OpenFOAM »* et tous les résultats seront présentés avec « *ParaView »*.

Ce mémoire est structuré de la manière suivante :

Un premier chapitre détaillant l'historique des Winglets ainsi que leur importance dans l'industrie aéronautique suivi de quelques notions succinctes sur l'aérodynamique et la CFD. Les outils utilisés dans nos simulations sont brièvement détaillés dans le second chapitre afin de bien mener les étapes de la simulation à travers les étapes du pré-processing, du processing et du post-processing. Le troisième chapitre est consacré à la conception du domaine de calcul à travers des macros-commandes en « *Python »* ainsi qu'au maillage qui est une étape primordiale sur laquelle repose nos résultats. En fin, le dernier chapitre regroupe les résultats des simulations des différentes configurations à travers les champs cinématique et dynamique.

Chapitre I

Notions d'aérodynamique

I.1- Introduction

Dans ce premier chapitre, nous allons détailler l'historique des Winglets ainsi que leur importance dans l'industrie aéronautique, puis nous verrons quelques notions succinctes sur l'aérodynamique et nous finirons par des notions sur la CFD.

I.2- Histoire et application des winglets

I.2.1- Définition

Les winglets sont des petites ailettes ajoutées à l'extrémité des ailes d'avions dans le but d'améliorer les performances sans trop augmenter les efforts sur leur structure. Il faut bien garder à l'esprit que du point de vue de la réduction de traînée induite, un winglet n'est pas réellement plus efficace qu'une simple extension de l'aile. Mais les winglets sont un excellent moyen de diminuer la traînée induite en minimisant les efforts sur la structure à l'emplanture de l'aile. Leur utilisation est particulièrement recommandée dans les cas où l'envergure doit être limitée par des contraintes structurales ou autres.

De plus la réduction de la traînée est essentielle pour réduire la consommation et donc la pollution des appareils. Par ailleurs, les winglets présentent l'avantage d'améliorer la stabilité du vol. Elles peuvent aussi être envisagées comme dispositifs annexes à ajouter à des avions anciens pour améliorer leurs performances sans que cela entraîne un chantier coûteux et sans que cela remette en cause les structures existantes.[2]

Figure I.1: *Idée de conception des winglets[2].*

I.2.2- Différents types de winglets

Au fil du temps, différents types et formes de winglets apparaîtront sur divers appareils avec des formes plus ou moins variées.

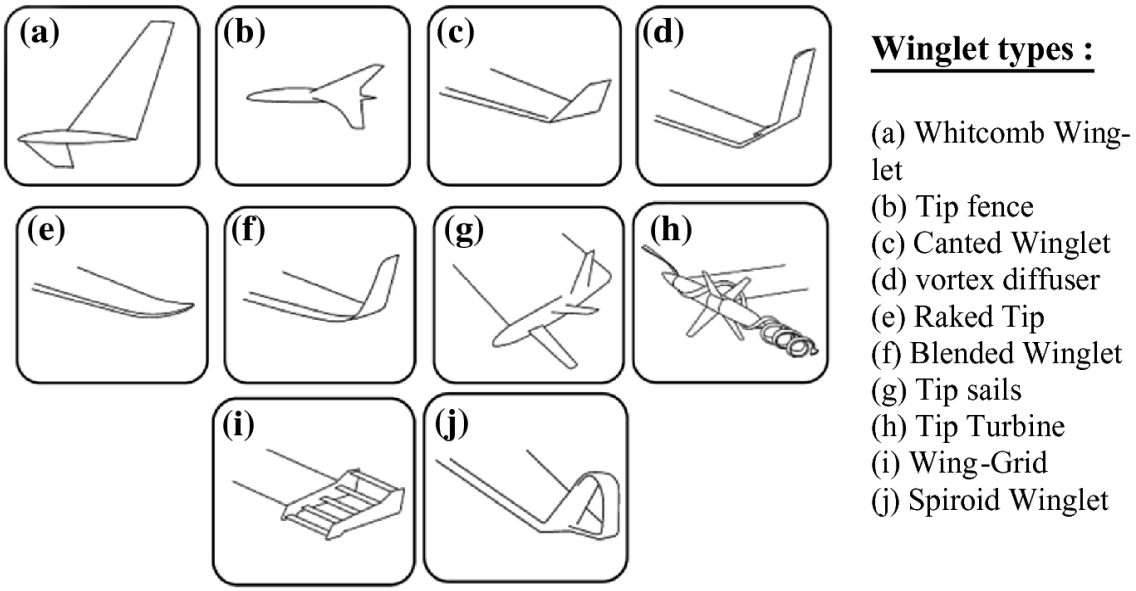


Figure I.2: *Différentes formes de winglets [3].*

Les quatre types de winglets que nous allons étudier dans notre travail sont :

(a) winglet *Demi-circulaire* ;
(b) winglet *Whitcomb* ;
(c) winglet *Spiroid* ;
(d) winglet *Double Spiroid*.

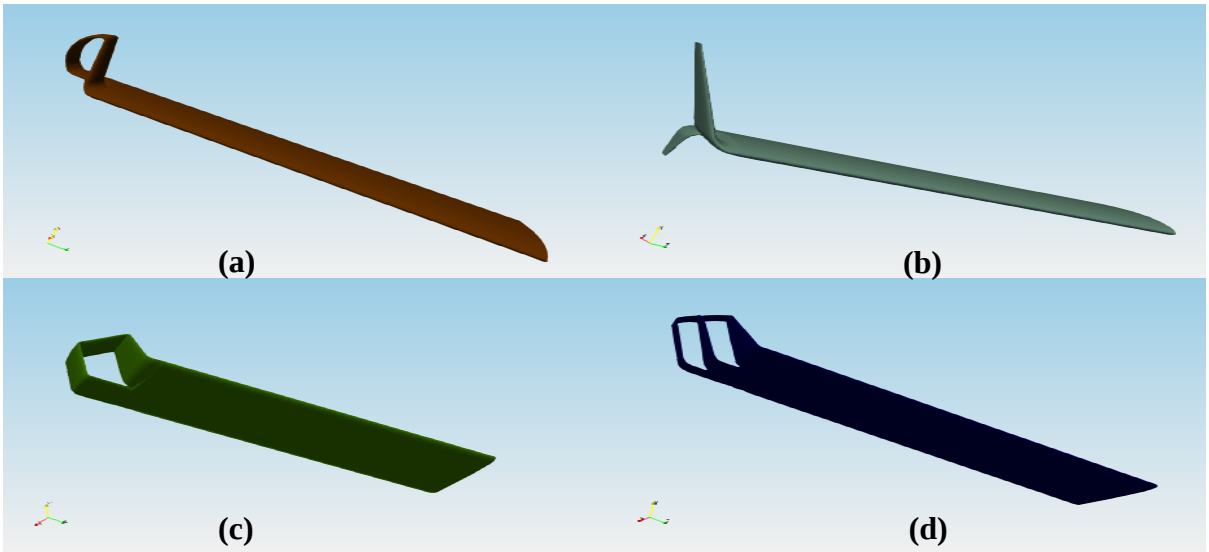


Figure I.3: *Différentes formes de winglets [2, 3].*

I.2.3- Historique

Ce n'est qu'en 1974 que l'on s'intéresse vraiment à réduire l'effet négatif de ces tourbillons en tentant de les supprimer ou de les réduire. C'est l'ingénieur Richard T. Withcomb du centre de recherche de la NASA qui va trouver une solution simple, issue de la nature, C'est en observant les vol de grands oiseaux tels que la cigogne ou bien encore la buse ou l'aigle qu'il découvre comment la nature a déjà reglé ce problème de tourbillons. En effet, à l'extrémité des ailes de la cigogne, on distingue des plumes qui se relèvent vers le haut (appelées rémiges). Ces plumes font office de barrière à l'air qui voudrait remonter sur l'intrados. Ces rémiges ont également la particularité d'être mobiles et de permettre à la cigogne de récupérer cette énergie des tourbillons pour en faire de la poussée grâce à un mouvement complexe de ces plumes. *Richard Whitcomb* décida donc de copier ce système sur un avion, en rajoutant une ailette fixe et verticale en bout d'aile. Ces ailettes, plus tard appelée «winglet» ou bien «sharklet» n'ont pas la capacité de traduire ces force en poussée mais permettent déjà de réduire considérablement ces tourbillons marginaux et donc la traînée qu'ils engendrent. Le premier avion à en bénéficier est un avion de l'US Air Force et de la NASA, un Boeing 707 d'essai.[2]

I.2.4- Application

L'application des winglets dans l'aviation civile n'a pas tardé puisqu'elle permettait des économies de carburant à une époque ou les prix du barils s'enflammaient. Les ingénieurs ont d'abord greffé des ailettes au bout des ailes de leurs appareils existants. Aujourd'hui, à l'heure du renouvellement opéré par les grands constructeurs que sont Airbus et Boeing, c'est toute l'aile qui est repensée pour économiser le carburant, en affinant celle-ci et en améliorant l'intégration de ces winglets, donnant parfois des résultats visuellement impressionnant.[2]

I.3- L'aérodynamique

I.3.1- Définition

L'aérodynamisme est une branche de la dynamique des fluides (elle même faisant partie de la mécanique des fluides). Elle consiste en l'étude de l'écoulement de l'air autour d'un objet et des conséquences qu'elle peut avoir sur ce dernier.[4]

1.3.2- L'aile

Un avion vole grâce à ses ailes et grâce à sa vitesse. Une aile comporte toujours deux faces : la face supérieure « *extrados* » a une forme bombée tandis que la face inférieure « *intrados* » est plutôt plane. La "pointe" avant est appelée bord d'attaque et l'extrémité arrière est appelée bord de fuite.[2]

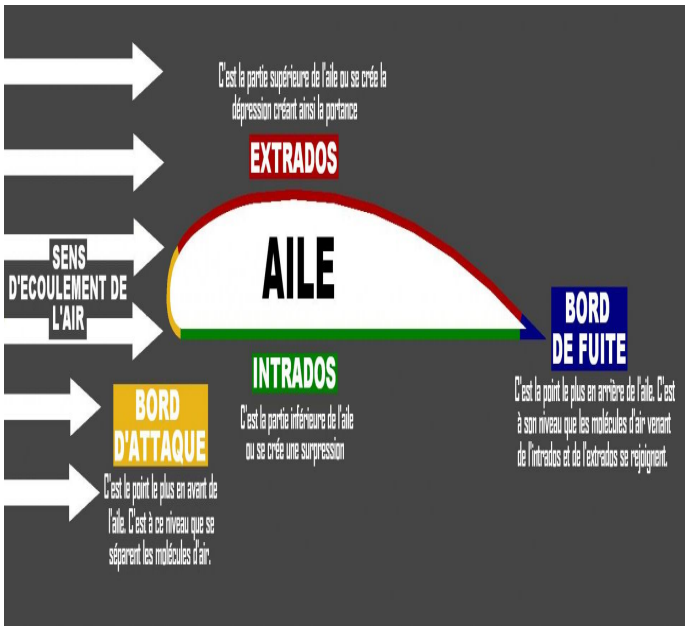


Figure I.4: *Composants d'un aile [2].*

Lorsque les ailes se déplacent vers l'avant l'air se divise en deux parties, l'une passant au dessus de l'aile, l'autre passant en dessous. Ces molécules d'air qui se séparent au niveau du bord d'attaque doivent se rejoindre au niveau du bord de fuite. Or la distance à parcourir pour les particules passant sur l'extrados est plus longue que pour celles passant sur l'intrados, elles doivent donc aller plus vite. Cela créé une surpression en dessous de l'aile et une dépression sur le dessus attirant celle ci vers le haut: c'est la *portance*.

I.3.3- Différents types d'aile

- L'**aile droite**, la plus utilisée dispose d'une excellente portance mais ne permet pas de voler à des vitesses élevées (au delà de 700 km/h). Elle est donc utilisée sur les avions de tourisme qui ne nécessitent pas de voler à de haute vitesse.

- L'**aile en flèche** permet au contraire d'atteindre des vitesses plus élevées mais perd un peu en portance. C'est pourquoi il faut une vitesse de décollage supérieure pour ce type d'aile. Le décrochage arrive donc bien plus tôt sur ce type d'aile.

- L'**aile en flèche inversée**, bien que très peu utilisée, offre de remarquables performances grâce à une manoeuvrabilité accrue. Toutefois, elle s'avère difficile à piloter.

- L'**aile à géometrie variable**. Le meilleur des deux mondes: combiner la portance élevée de l'aile droite en phase de décollage puis profiter de la faible trainée et des hautes vitesses que permet l'aile en flèche. Ceci ne va tout de même pas sans inconvénients puisque ce système est à l'origine d'un surpoids non négligeable, ce qui limite sa diffusion.

- L'**aile oblique**, elle pivote sur elle même en plein vol selon le même principe que l'aile à géometrie variable mais ce type de voilure se montre extrêmement délicate à piloter. Elle n'a jamais été appliquée sur un avion de série.

- L'**aile delta**, c'est-à-dire en forme de triangle permet notamment à portance égale une plus faible épaisseur puisque la surface est plus grande. Elle a également la forme du cone supersonique apparaissant lorsque l'avion franchit le mur du son, et donc reste toujours derrière celui ci ce qui a pour avantage d'éviter les turbulences. Elle est donc très utilisé sur les avions supersoniques. Elle ne permet malgré tout pas une très haute manoeuvrabilité.

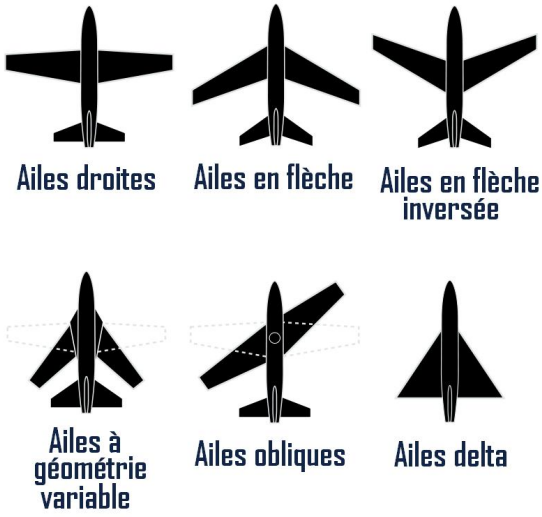


***Figure I.5**: Différentes formes d'ailes [2].*

I.3.4- Forces aérodynamiques

Comment un avion vole-t-il ? C'est la question que (presque) tout le monde s'est posé au moins une fois... mais tout le monde ne sais pas y répondre.

Un avion en vol symétrique est soumis à deux forces aérodynamiques:

- la portance, perpendiculaire à la vitesse d'avancement et dirigée vers le haut.
- la trainée, résistance à l'avancement parallèle à la vitesse et dirigée vers l'arrière.

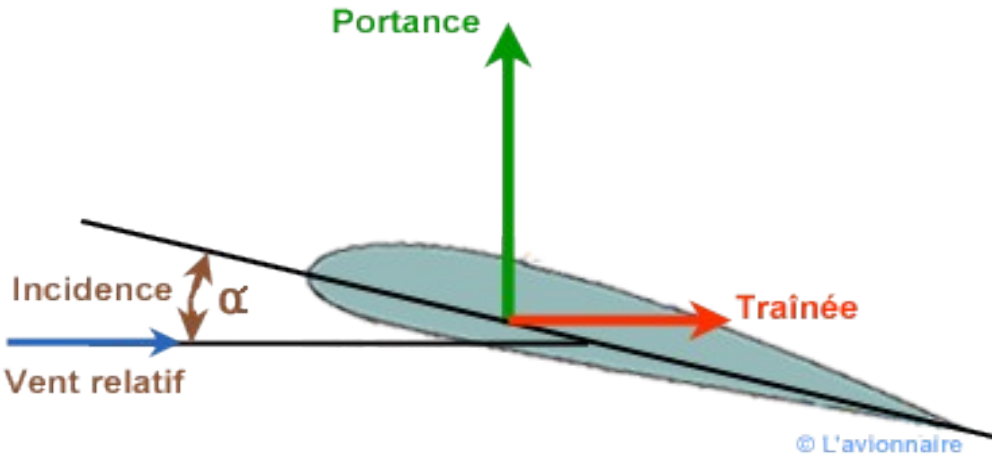


Figure I.6: *Forces aérodynamiques [5].*

I.3.4.1- Portance

La portance est une force qui vient compenser le poids de l'avion qui l'entraîne vers le bas. Celle-ci, à l'inverse, le "porte" donc exerce une force de sens opposé (mais de même direction).
En réaction à la quantité de mouvement de la masse d'air déviée dans un sens (vers le bas pour un profil porteur), l'aile est tirée dans l'autre sens (vers le haut). [2]

$$L = \frac{1}{2}.\rho.V^2.S.CL \quad \text{.....(I.1)}$$

avec: ρ: masse volumique du fluide en kg/m^3 ;

V: vitesse en m/s ;

S: surface de référence en m^2 ;

CL: coefficient de portance (Nombre sans dimension).

I.3.4.2- Traînée

La traînée est une des quatre principales forces aérodynamiques, elle s'oppose à la poussée. C'est la force exercée par l'air autour d'un objet en mouvement (avion, voiture, train...etc.) dans le sens opposé à sa direction.

L'air s'écoule de façon différente autour des objets selon leur forme. Un objet ayant une grande surface exposée à l'écoulement de l'air sera pénalisé. A l'opposé, un objet très profilé, comme une aile possédera une meilleure pénétration dans l'air. Cette pénétration se calcule grâce au coefficient de pénétration dans l'air appelé C_x qui permet de mesurer la traînée d'un objet en calculant son coefficient de traînée noté C_x. C'est un nombre sans unité. Il représente la force de résistance d'une surface. [2]

$$D = \frac{1}{2}.\rho.V^2.S.CD \quad \ldots\ldots \text{(I.2)}$$

avec: ρ: masse volumique du fluide en kg/m^3 ;

V: vitesse en m/s ;

S: surface de référence en m^2 ;

CD: coefficient de traînée (nombre adimensionnel).

I.3.4.3- Différentes formes de traînée

- **La trainée de frottement** qui résulte, comme l'indique son nom, du frottement de l'air sur l'objet. On favorise des surfaces lisses à des surfaces rugueuses ou irrégulières pour que les filets d'air "accrochent" le corps et ne se décollent pas.
- **La trainée de forme** qui provient du détachement des filets d'air de l'objet à l'arrière: lorsque l'objet avance, l'air s'écoule autour de lui en l' "accrochant" : or, à l'arrière de l'objet, les filets d'air ont tendance à se décrocher.
- **La traînée d'onde** qui a pour origine le passage aux vitesses supersoniques. En effet, lorsqu'un objet (généralement un avion ou une fusée) atteint la vitesse du son, une onde de choc se produit en avant de l'avion. Il se forme une surpression à l'avant de l'avion qui le ralentit considérablement, ce qui explique qu'il faut une énergie considérable pour atteindre de telles vitesses.

I.4- Turbulence de sillage

La turbulence de sillage dans son ensemble est due à l'existence de trois facteurs:

- les effets de souffle des hélices ou des réacteurs ;
- la turbulence induite par le fuselage et les aspérités génératrices de traînée ;
- les fameux tourbillons marginaux ou vortex contrarotatifs, prenant naissance aux extrémités des ailes. [2]

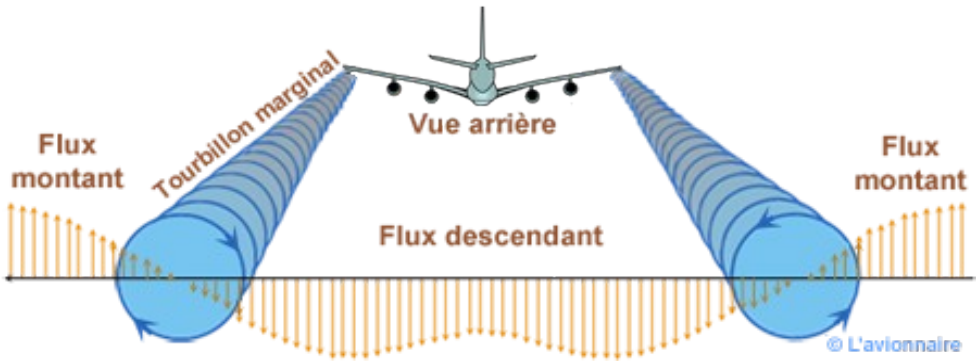


Figure I.7: *Sillage derrière l'avion [4].*

1.5 Régimes d'écoulement

Les expériences réalisées par *Reynolds* en1883 lors de l'écoulement d'un liquide dans une conduite cylindrique rectiligne dans laquelle arrive également un filet de liquide coloré, ont montré l'existence de deux régimes d'écoulement : régime laminaire et régime turbulent:

- **Régime laminaire:**

Si les filets fluides sont des lignes régulières, sensiblement parallèles entre elles, l'écoulement est dit laminaire.[2]

- **Régime turbulent:**

Si les filets fluides s'enchevêtrent, s'enroulent sur eux-mêmes, l'écoulement est dit turbulent.[6]
En utilisant divers fluides à viscosités différentes, en faisant varier le débit et le diamètre de la canalisation, *Reynolds* a montré que le paramètre qui permettait de déterminer si l'écoulement est laminaire ou turbulent est un nombre sans dimension $\boldsymbol{R_e}$ appelé nombre de *Reynolds* donné par l'expression suivante:

$$R_e = \frac{V \cdot d}{\nu}$$

où : ***V***: Vitesse moyenne d'écoulement à travers la section considérée en (m/s)

d: Diamètre de la conduite ou largeur de la veine fluide en (m).

υ: Viscosité cinématique du fluide (m^2/s).

Résultats empirique à titre indicatif:

- Si $R_e < 2000$ l'écoulement est laminaire.
- Si $R_e > 2000$ l'écoulement est turbulent.

I.6- Techniques CFD

La *CFD* ou « *Computational Fluid Dynamics* » pourrait se traduire par Simulation numérique de la Dynamique des Fluides. On peut simuler des phénomènes de transferts de chaleur et de masse et autres phénomènes tels que les réactions chimiques. Ces phénomènes sont souvent régis par les équations de *Navier-Stokes*. Pour des écoulements simples tels que l'écoulement stationnaire dans un tube circulaire ou celui de la couche limite sur une plaque plane, ces équations sont directement intégrées et résolues par méthode analytique. Dans le cas général, une méthode de discrétisation est appliquée pour l'approximation des équations aux dérivées partielles. Ces dernières sont remplacées par des équations algébriques qui sont résolues numériquement. Un maillage est généré dans un domaine d'étude qui entoure l'obstacle considéré (aile, avion, voiture, hélice, ...etc.) ou un domaine délimité par le contenant du fluide. Les équations algébriques sont résolues pour chaque nœud du maillage. Le plus souvent, une méthode itérative est appliquée. Les calculs sont exécutés à l'aide d'un code *CFD* [7]. En pratique, les simulations *CFD* s'effectuent en trois étapes : pre-processing, solver et post-processing. Au cours de la première étape, «pre-processing», on construit la géométrie et on définit le domaine d'étude. Le maillage est ensuite généré à l'intérieur de ce domaine. Ensuite, on spécifie les propriétés physiques du fluide et on choisit les modèles à appliquer, l'algorithme de calcul, les méthodes d'interpolation et autres schémas. La deuxième étape, «Solver», c'est la phase de calcul, ou de résolution des équations algébriques. La troisième et dernière étape, «post-processing», c'est la phase de traitement et de visualisation des résultats.

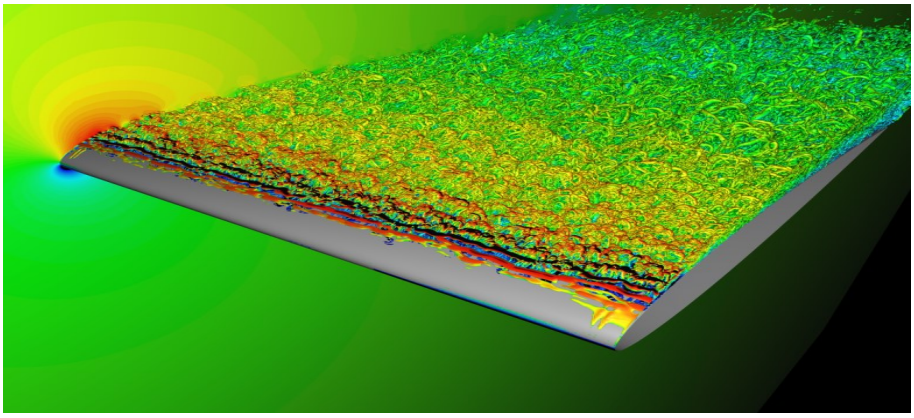


Figure I.8: *Simulation de l'écoulement autour d'une aile avec technique CFD [8].*

I.7 Conclusion

Dans ce chapitre, nous avons passé en revue quelques définitions et généralités sur les winglets et les ailes, son historique, ces applications et importance au niveau industriel, aussi quelque base d'aérodynamique et l'impact de cette technologie au niveau des performances des avions. Le chapitre suivant sera consacré aux outils que nous allons utiliser afin d'entreprendre la simulation de l'écoulement autour d'une aile avec différentes formes de winglets.

Chapitre II

Outils de simulation

II.1- Introduction

Nous allons donner, dans ce chapitre, les informations nécessaires tous les les outils que nous avons utilisé durant nos simulations. Nous présenterons, en premier lieu, le logiciel de conception 3D paramétrique « *FreeCAD* » et nous insisterons sur certains détails très importants concernant le nouveau atelier « *cfdOF* » inclus à partir de la version 0.17 et qui permet de lancer le maillage et les simulations à partir de « *FreeCAD* ». Nous présenterons aussi le logiciel de maillage « *cfMesh* » ainsi que le solveur « *openFOAM* ». nous terminerons finalement par le logiciel « *ParaView*» qui nous permet de présenter tous nos résultats.

II.2- FreeCAD

II.2.1- Définition

« *FreeCAD* » est une application de modélisation paramétrique 3D CAD/CAE. Elle est principalement destinée à la conception mécanique, mais sert aussi à toutes les utilisations telles que la modélisation des objets de précision et le contrôle de l'historique de la modélisation 3D. Il actuellement très utilisé avec les imprimantes 3D.[9]

II.2.2- Explorer FreeCAD

Le concept principal de l'interface de « *FreeCAD* » est qu'il est composé d'ateliers (*Workbench*). Un atelier est une collection d'outils adaptés pour une tâche spécifique, comme travailler avec des maillages, faire du dessin 2D, ou faire des esquisses avec contraintes. Les outils inclus dans chaque atelier peuvent être personnalisés : on peut ajouter des outils provenant d'autres ateliers ou même créer nos propres outils à travers des « *macros* ». Les points de départ largement utilisés après l'atelier « *Squetcher* » sont les ateliers « *PartDesign* » et « *Part* ».

L'interface de « *FreeCAD* » est composée de (Fig.II.1):

1. La vue 3D, affichant le contenu du document.
2. L'arborescence montrant l'historique et la hiérarchie de la construction de tous les objets dans le document.
3. L'éditeur de propriétés, qui permet d'afficher et de modifier les propriétés des objets sélectionnés.
4. La vue rapport (ou fenêtre de sortie) dans laquelle sont affichées les messages, les avertissements et les erreurs.
5. La console *Python*, où sont affichées toutes les commandes exécutées par « *FreeCAD* » et que nous pouvons réutiliser dans les macros sous forme de code *Python*.

6. Le sélecteur d'atelier qui permet de sélectionnez l'atelier à activer.

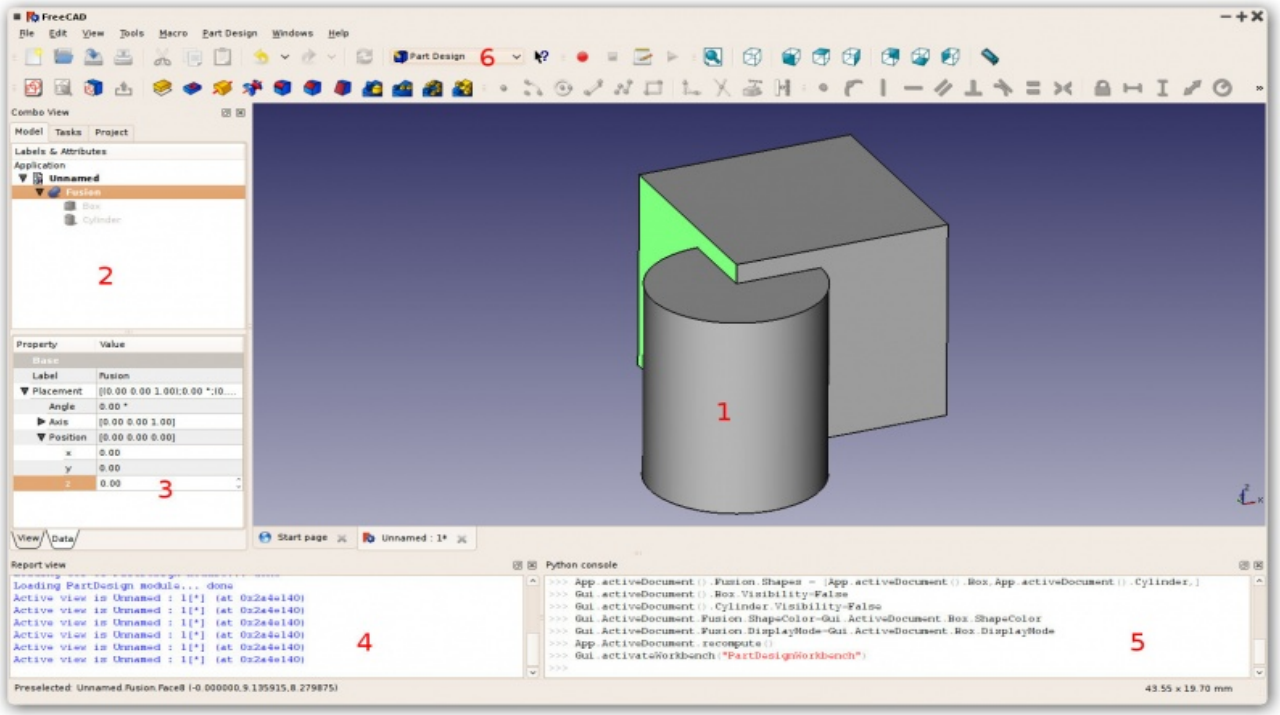


***Figure II.1**: Interface de travail de FreeCAD.[9]*

II.2.3- Ateliers de FreeCAD

A l'instar de nombreuses applications de conception modernes telles que *Revit* ou *CATIA*, « *FreeCAD* » est basé sur le concept d'Atelier. Un atelier peut être considéré comme un ensemble d'outils spécialement regroupés pour une tâche donnée. Dans un atelier de fabrication de meubles traditionnels, vous disposerez d'une table de travail pour la personne qui travaille le bois, d'une autre pour celui qui travaille avec des pièces métalliques et peut-être d'une troisième pour celui qui monte toutes les pièces ensemble.[9]

Les ateliers suivants sont disponibles sur toutes les installations « *FreeCAD* »:

1. **L'atelier Arch** pour concevoir des éléments architecturaux.
2. **L'atelier Draft** contient des outils 2D ainsi que des opérations de CAO 2D et 3D de base.
3. **L'atelier Drawing** a été utilisé pour afficher votre travail 3D sur une feuille de dessin 2D mais est désormais obsolète. Il est toujours nécessaire de lire les anciens fichiers « *FreeCAD* » contenant un objet Drawing créé à l'origine avec cet atelier. Voir l'atelier TechDraw, qui est le remplaçant plus évolué.
4. **L'atelier FEM** permet l'analyse numérique par la méthode des éléments finis.
5. **L'atelier Image** pour travailler avec des images matricielles.

6. **L'atelier Inspection** met à votre disposition des outils spécifiques à l'analyse de formes. Il est toujours en développement.
7. **L'atelier Mesh Design** pour travailler avec des maillages.
8. **L'atelier OpenSCAD** offre l'interopérabilité avec OpenSCAD et permet la réparation de l'historique de modèles créés par géométrie de construction de solides (CSG en anglais).
9. **L'atelier Pièce** pour travailler avec des pièces de CAO.
10. **L'atelier Part Design** pour construire des pièces à partir d'esquisses.
11. **L'atelier Path** est utilisé pour générer des instruction G-Code. Il est toujours en développement.
12. **L'atelier Plot** permet de modifier et d'enregistrer les sorties créées à partir d'autres modules et outils.
13. **L'atelier Points** permet de travailler sur des nuages de points.
14. **L'atelier Raytracing** pour le lancer de rayons (rendu ou images de synthèse).
15. **L'atelier Reverse Engineering** (rétro-ingénierie) a pour but de fournir des outils spécifiques à la conversion de formes/solides/maillages en des fonctions paramétriques compatibles avec « *FreeCAD* ».
16. **L'atelier Robot** pour étudier les mouvements de robots industriels.
17. **L'atelier Ship** travaille sur des entités de navire qui doivent être créés à partir de géométrie fournie.
18. **L'atelier Sketcher** pour créer des esquisses contraintes géométriquement.
19. **L'atelier Spreadsheet** sert a créer et manipuler des données dans une feuille de calcul.
20. **L'atelier Start Center** permet d'aller rapidement à l'un des ateliers les plus communs.
21. **L'atelier Surface** fournit des outils pour créer et modifier des surfaces. Il est similaire au générateur de forme de Part Face à partir d'arêtes.
22. **L'atelier TechDraw** est le successeur le plus avancé et le plus riche en fonctionnalités de l'atelier Drawing.
23. **L'atelier Test Framework** (banc de test) permet de déboguer « *FreeCAD* ».
24. **L'atelier Web** fournit un navigateur en lieu et place de la vue 3D de « *FreeCAD* ».

II.2.4 Atelier PartDesign (conception de pièce)

L'atelier *PartDesign* (conception de pièce) fournit des outils avancés pour la modélisation de pièces complexes et solides. Il est principalement axé sur la création de pièces mécaniques pouvant être fabriquées et assemblées dans un produit fini. Néanmoins, les solides créés peuvent en général être utilisés à d'autres fins, telles que la conception architecturale, l'analyse par éléments finis ou l'usinage et l'impression 3D.[9]

Les outils *PartDesign* sont situés dans le menu *PartDesign* qui apparaît lorsque cet atelier est chargé (Fig.II.2).

1. - Outils structure.
2. - Outils d'assistance PartDesign.
3. - Outils de modélisation PartDesign.

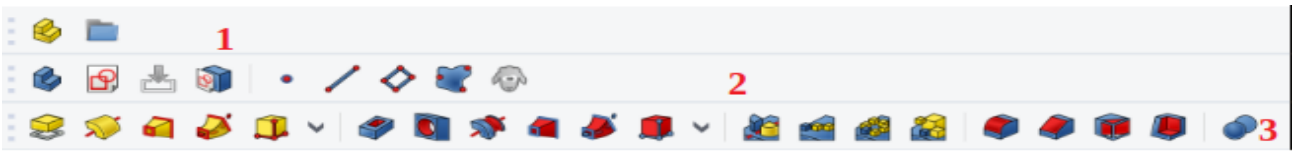


Figure II.2: *Outils PartDesign.*

II.2.5- Atelier Sketcher

L'atelier Sketcher permet de créer des géométries 2D nommées esquisses, qui seront principalement utilisées par l'atelier PartDesign, l'atelier Architectural et d'autres ateliers. En général, la géométrie 2D est le point de départ de la plupart des modèles de CAO - une esquisse 2D peut être extrudée en une forme 3D; d'autres esquisses peuvent être utilisées pour créer des fonctions comme des cavités, des crêtes ou encore des extrusions qui s'ajoutent aux formes 3D précédemment construites. Avec les opérations Booléennes sur des solides définies dans l'atelier Part, l'atelier *Sketcher* est au cœur de la conception 3D solide.

L'atelier *Sketcher* met à l'avant les «contraintes» qui permettent de définir des formes 2D selon des critères géométriques précis. Un solveur mathématique calcule le niveau de contrainte de l'esquisse et permet l'exploration interactive des degrés de liberté. [9]

Les outils de l'atelier Sketcher sont affichés lorsque ce dernier est chargé (Fig.II.3) :

1. Géométries d'esquisse.
2. Contraintes d'esquisse.
3. Outils d'esquisse.
4. Sketcher Outils B-spline.

5. Sketcher espace virtuel.

Figure II.3: *Ooutils de l'atelier Sketcher.*

II.2.6- Atelier Part 

Les capacités de modélisation de solides de « *FreeCAD* » sont basées sur le noyau Open Cascade Technology (OCCT) , un système de CAO de niveau professionnel qui offre une création et une manipulation avancées de la géométrie 3D. L'atelier *Part* est une couche située au-dessus des bibliothèques OCCT, qui permet à l'utilisateur d'accéder aux primitives et fonctions géométriques OCCT. Toutes les fonctions de dessin 2D et 3D de chaque atelier (*Draft*, [*Sketcher* Workbench/fr| Sketcher], *PartDesign*, ...etc.), sont basées sur ces fonctions exposées par l'atelier *Part*. Par conséquent, ce dernier est considéré comme le composant central des capacités de modélisation de « *FreeCAD* ». [9]

Les outils de l'atelier *Part* sont sont affichés lorsque ce dernier est chargé (Fig.II.4) :

1. Primitives.
2. Modifier les objets.
3. Outils des opérations.
4. Outils des mesures.

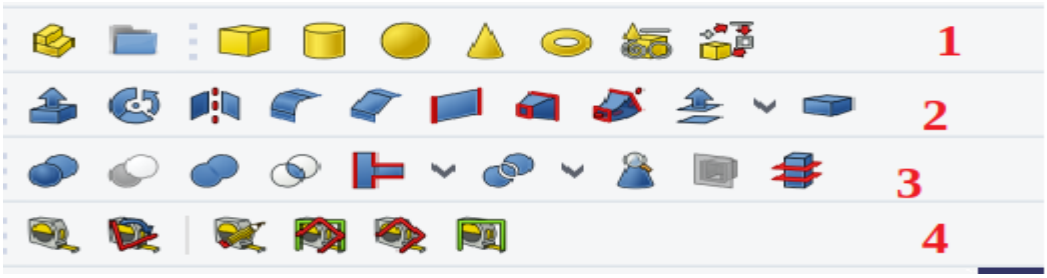


Figure II.4: *Outils de l'atelier Part.*

II.3- Atelier «*cfdOF*»

II.3.1- Définition

L'atelier *cfdOF* est un atelier de sumilation et analyses dans FreeCAD (Fig.II.5).

Cet atelier a pour objectif d'aider les utilisateurs à configurer et à exécuter des analyses CFD basées sur le solveur OpenFOAM à l'intérieur de « *FreeCAD* ». Il guide l'utilisateur dans la sélection de la physique appropriée, en spécifiant les propriétés du matériau, en générant un maillage, en attribuant des conditions aux limites et en définissant les paramètres du solveur avant d'exécuter la simulation. Les meilleures pratiques sont choisies pour maximiser la stabilité des solveurs.[10]

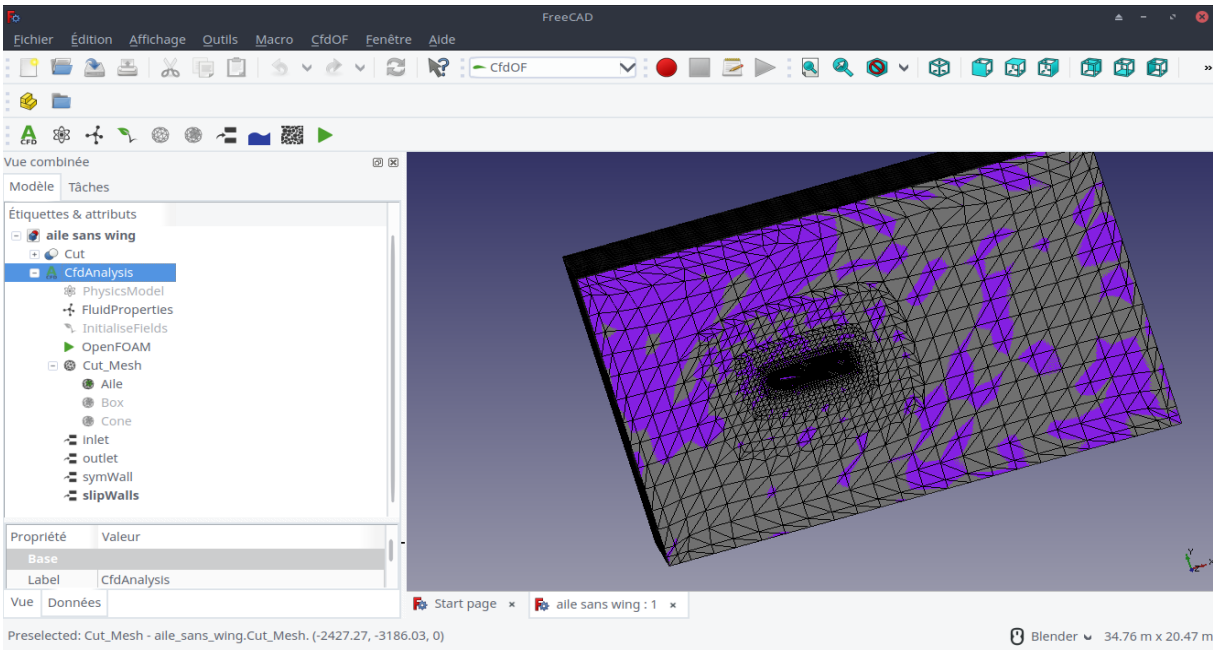


***Figure II.5**: Atelier cfdOF dans FreeCAD.*

II.3.2- Caractéristiques

Flux laminaire incompressible.

Écoulement à surface libre incompressible.

Débit compressible à haute vitesse (HiSA).

Base de données matérielle de base.

Initialisation du flux avec un solveur de potentiel.

Maillage cartésien de cellules coupées avec couches limites (cfMesh).

Maillage cartésien de cellules coupées avec un support poreux (snappyHexMesh).

Maillage tétraédrique à l'aide de GMSH .

Post-traitement à l'aide de *ParaView*.

Régions poreuses et chicanes poreuses.

Fonctionne sous Windows 7-10 et Linux.

Tests unitaires.

Extension à un écoulement turbulent en utilisant RANS (k-w SST).

Nouveau constructeur de cas utilisant une structure de modèle extensible.

Macro script

II.3.3- Paramètres de l'atelier cfdOF

L'atelier *cfdOF* fonctionne selon les étapes suivantes (Fig.II.6):

1-Créer un conteneur d'analyse avec le solveur *cfdOF*.

2-Sélectionner le modèle physique laminare ou turbulent «RANS», compressible ou incompressible.

3-Ajouter des propriétés de fluide (viscosité et densité de fluide).

4-Initialiser les variables de flux internes en fonction du modèle physique sélectionné.

5-Créer un maillage en utilisant *cfMesh*, *snappyHexMesh* ou *gmesh.*

6-Créer des régions de maillage et régions de raffinement.

7-Créer des couches limites.

8-Sélectionner et créer une zone d'initialisation.

9-Sélectionner et créer une zone poreuse.

10-Editer les propriétés et lancer le solveur.

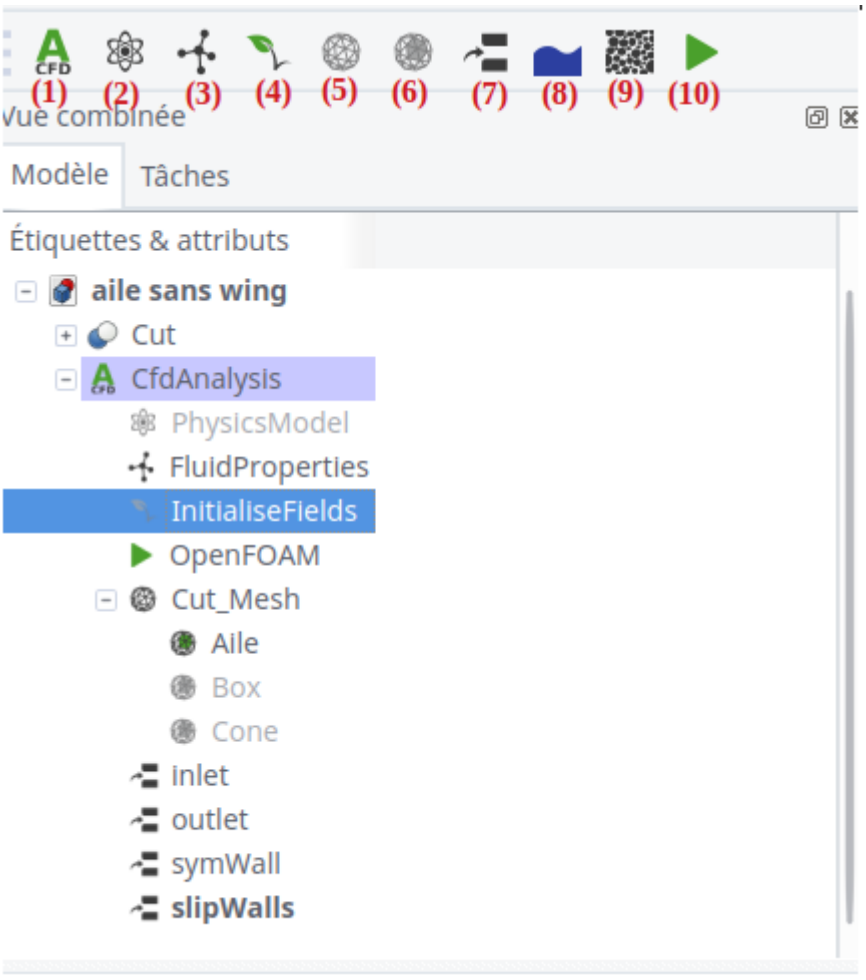


***Figure II.6**: Paramètres de l'atelier cfdOF.*

II.3.4- Installation de l'atelier cfdOF

Avant d'installer c*fdOF*, le plan de travail Plot doit d'abord être installé dans « *FreeCAD* » à l'aide du gestionnaire d'*addon*:

- Lancer « *FreeCAD* » .
- Sélectionnez Outils | Gestionnaire d'addons ...
- Sélectionnez Tracé dans la liste des ateliers, puis cliquez sur "Installer / mettre à jour".
- Redémarrez « *FreeCAD* ».
- Répétez l'opération ci-dessus pour l'atelier "*CfdOF*".
- Pour l'installation de dépendances, voir ci-dessous (Fig.II.7)[9]

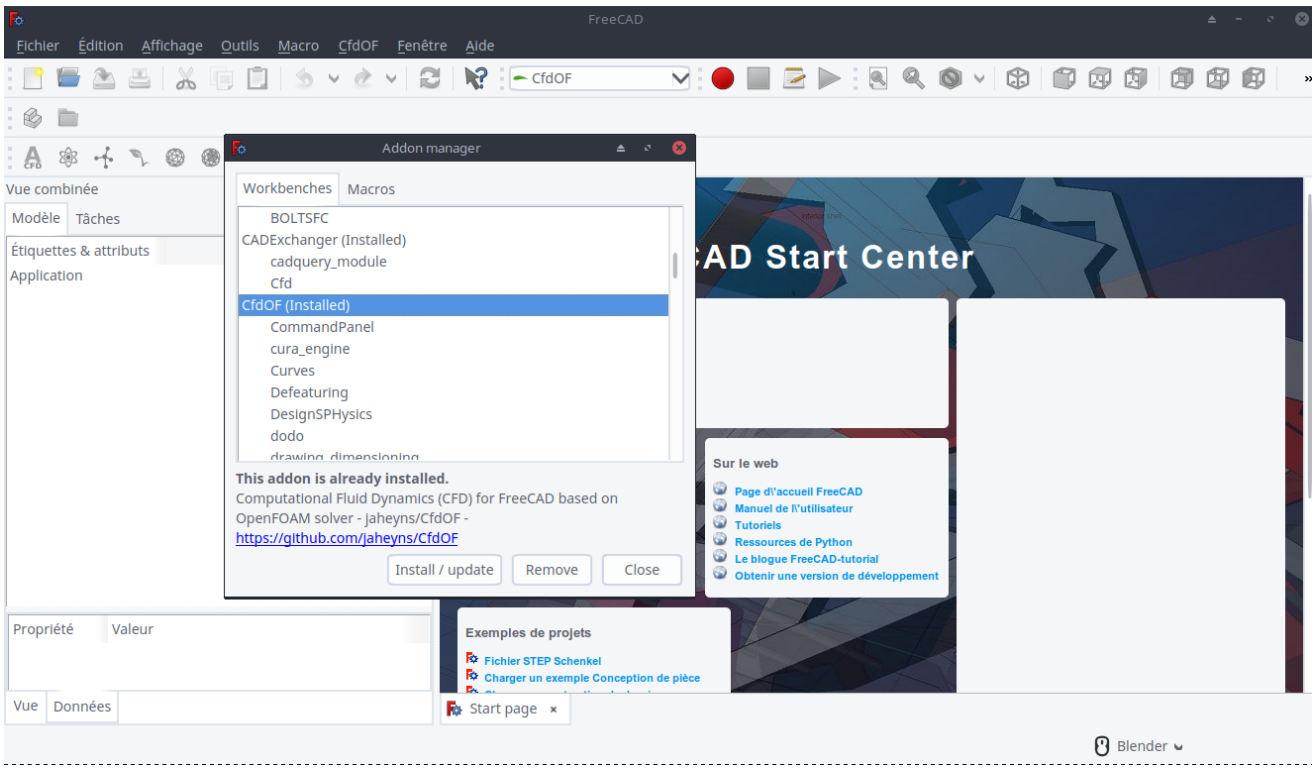


***Figure II.7**: Installation de l'atelier cfdOF.*

II.4 Mailleur cfMesh

II.4.1 Définition

cfMesh est un outil de maillage permettant d'avoir différent type de maillage. Il est développé par la société "*Creative Fields*". Ce mailleur est Open source, gratuit et compatible avec *OpenFOAM*. Il s'agit d'un mailleur parallèle assez rapide qui permet de générer des mailles cartésiennes (2D et 3D), tétraédriques et polyhédrales. Le fichier de la géométrie à mailler doit être au format stéréolithographie (.stl).[11]

2.4.2 Capacité de cfMesh

- **Géométrie:** au format stéréolithographie (.stl). Les patchs des condition aux limites doivent être définie dans le fichiers STL.
- **Raffinement :** cfMesh permet le raffinement du maillage localement à l'aide des patchs et de géométrie primitive.
- **Espace mince:** cfMesh permet de garder ou de supprimer les mailles dans les petits gap.
- **Inflation sur les surfaces:** Nombre de couches, rapport d'épaisseur, épaisseur maximale.
- **Modification du maillage:** cfMesh permet d'effectuer des transformations topologiques et géométriques sur le maillage.
- **Parallélisation:** cfMesh permet la parallélisation du maillage à l'aide de OpenMPI.

Les schémas suivantes montrent la structure de *cfMesh* (Fig.II.8-9):

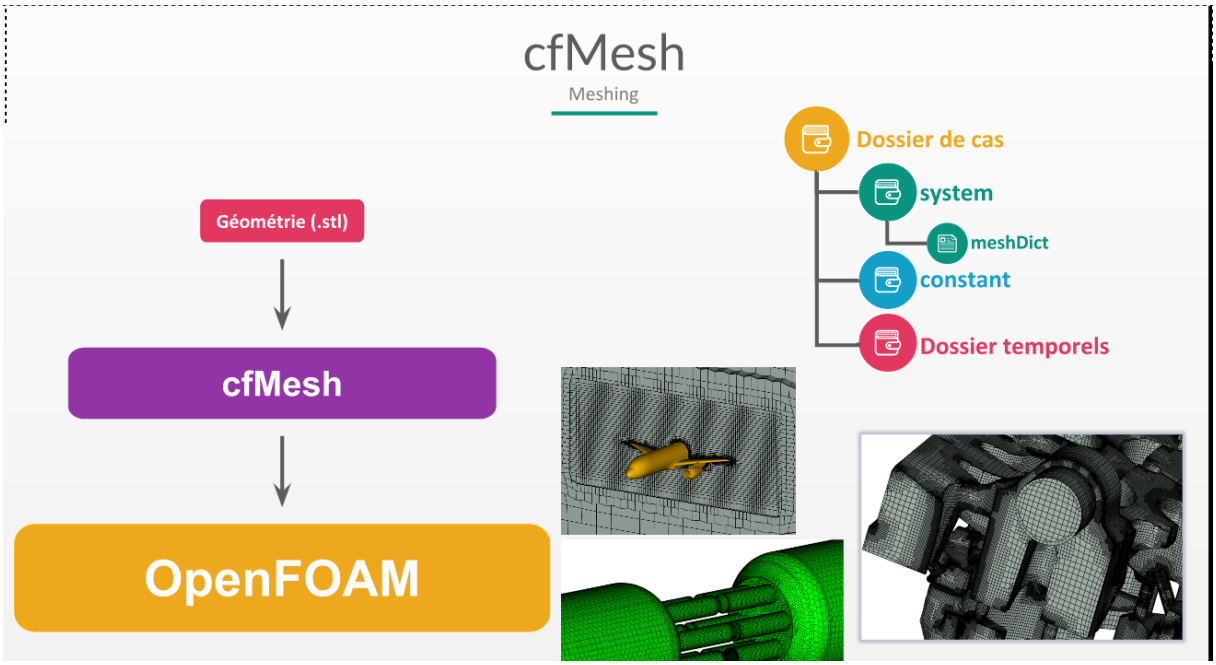


Figure II.8: *Schéma de la structure de cfMesh [11].*

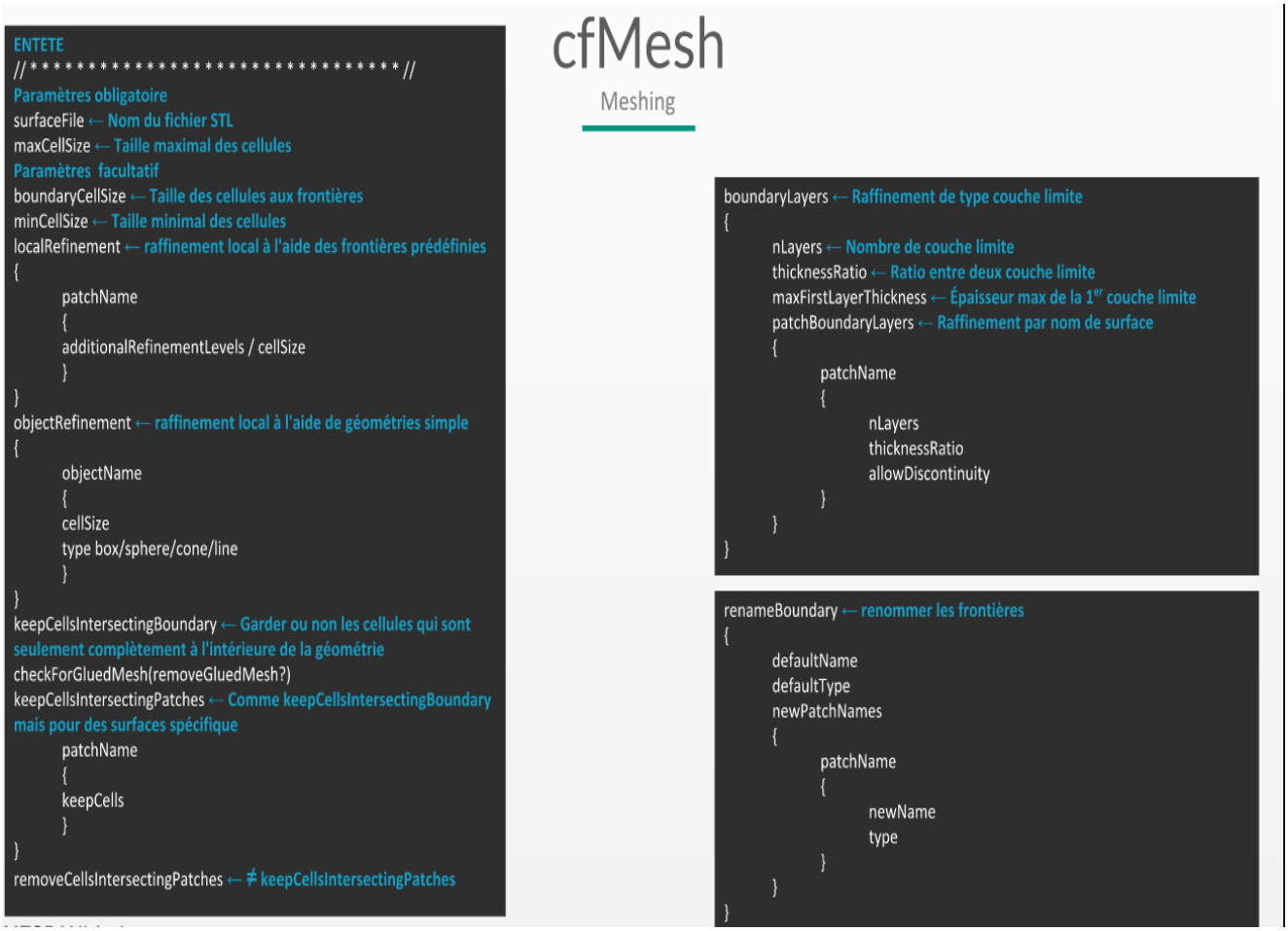


Figure II.9: *Exemple de fichier cfMesh [11].*

II.5- Solveur openFOAM

II.5.1- Définition

OpenFOAM (***O****pen* ***F****ield* ***O****peration* ***A****nd* ***M****anipulation*) est un solveur multi-physique (CFD, mécanique des solides, éléctromagnétisme, finance, ...etc) principalement orienté vers la mécanique des fluides, il a été conçu en 1989 à l'*Imperial College London*, et il est dédié à la résolution des équations aux dérivées partielles par la méthode des volumes finis.[12]

Il s'agit d'un code open-source développé en C++ (programmation orienté objet), conçu comme une boite à outils, il contient plus de 200 programmes (des pré-processeurs, des solveurs et des outils de post-traitement), il est réutilisable et modifiable à souhait, et l'utilisateur peut par ce biais programmer un nouveau solveur correspondant à son cas d'étude. Une autre caractéristique du

logiciel est qu'il ne possède pas une interface graphique (comme *Fluent*, *Code_Saturne*,.. etc), toute la simulation numérique et le conditionnement du cas se fait dans des fichiers textes.

Pour télécharger le logiciel, il suffit d'aller au site suivant : *http://www.openfoam.org/download/* et suivre la démarche qui est bien détaillée.[12]

II.5.2- Fonctionnalités d'OpenFOAM

OpenFOAM est une bibliothèque rapide à mettre en place, large et évolutive, qui ne cesse jamais d'évoluer, puisque chaque utilisateur peut créer son propre solveur et son propre modèle. Ce code est un outil inédit qui a plusieurs fonctionnalités. Il permet à la fois de générer des maillages (ou de les importer), de faire de la CFD et offrir des outils pour la visualisation et le post-traitement des résultats.

- Génération du maillage: Ce logiciel permet de générer des maillages structurés à partir de :

- *blockMesh*: générateur de maillage multi-block.
- *snappyHexMesh*: Cet outil est très important et permet un raffinement de maillage automatique. *OpenFOAM* est capable d'utiliser des maillages faits avec d'autres mailleurs tels que *Salome*, *Ansys*, *FreeCAD*, *CFX*, *Star-CCM+*.

- Calcul CFD: *OpenFOAM* en tant que code de calcul permet de résoudre un problème de mécanique des fluides via la méthode des volumes finis, quelque soit le problème étudié (stationnaire ou instationnaire, compressible ou incompressible, mono-phasique ou multi-phasique ...etc), en choisissant un solveur bien adapté au problème, en modifiant un solveur déjà existant ou même en créant un nouveau solveur qui décrit le mieux le problème étudié.

- Post-traitement des résultats: *OpenFOAM* fournit aussi des outils de post-traitement et de visualisation des résultats, le plus utilisé est *paraFoam*. Outil d'une grande utilité et puissance, qui permet de visualiser facilement les grandeurs qui intéressent l'utilisateur, leur évolution suivant n'importe quelle variables, et extraire les données désirées.
Après avoir présenté brièvement ce code de calcul ainsi que ses principales caractéristiques, nous allons par la suite expliquer en détail le fonctionnement de ce code, et les différents étapes à suivre pour simuler un problème en mécanique des fluides avec ce logiciel.[12]

II.5.3- Structure générale d'un cas OpenFOAM

Le code *OpenFOAM* en général est une bibliothèque C++, qui permet de créer des exécutables appelés *applications*.

Ce Code de calcul dispose d'énormes applications pré-compilées qui correspondent à toute catégorie de problèmes à étudier. L'une des questions fondamentales à se poser pour étudier un cas avec *OpenFOAM*, est celle de l'application (ou solveur) à utiliser. L'utilisateur n'a pas besoin de choisir les équations à résoudre, mais plutôt la bonne application à adopter.

Les applications sont divisées en deux catégories :

Solvers : Conçus pour simuler un problème donné.

Utilities : Destinées pour la manipulation des données.

Les équations à résoudre sont codées dans ces applications.

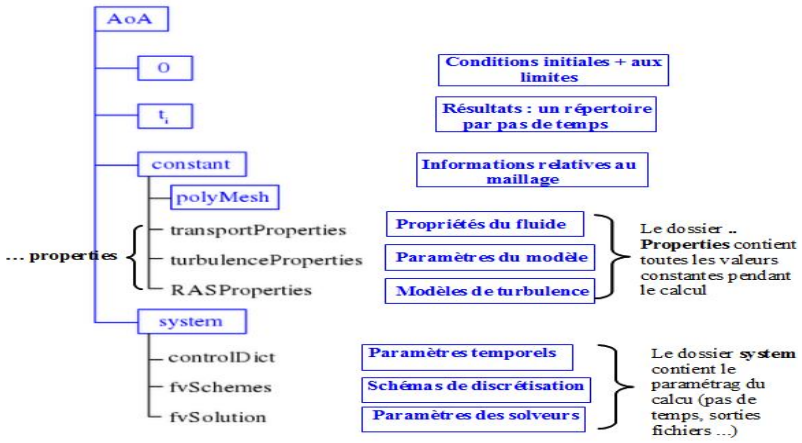


Figure II.10: *Schéma de la structure générale d'un cas OpenFOAM.[12]*

II.5.4- Description des dossiers

II.5.4.1- Dossier « 0 »

Ce dossier contient les valeurs initiales et les conditions limites des différentes grandeurs. Les conditions aux limites sous *OpenFOAM* est le point le plus délicat. Leur rôle dans la modélisation n'est pas simplement dans la géométrie, mais représentent une partie intégrale de la solution. Dans cette partie nous allons nous focaliser sur ce point, et nous discuterons la façon dont elles sont traitées sous *OpenFOAM*. Chaque grandeur à résoudre (U, p, k ...) par le solveur doit être initialisée sur toutes les zones du cas ainsi qu'à l'intérieur du domaine. [12]

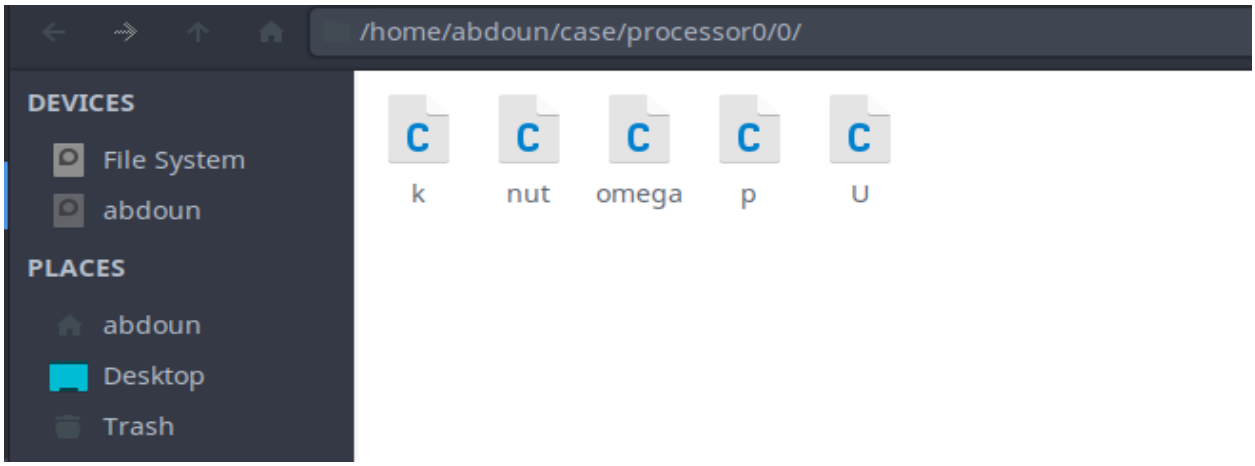


Figure II.11: *Dossier « 0 ».*

II.5.4.2- Dossier « constant »

Ce dossier contient les paramètres nécessaires pour le maillage et les constantes du problème (propriétés du fluide, ...etc). La génération du maillage se fait dans le sous-dossier *polyMesh* et la définition des constantes du problème se fait dans des fichiers properties.

Les fichiers qui contiennent les propriétés physiques, thermiques et énergétiques de notre système, par exemple: *transportProperties*, mixtureProperties, *thermophysicalModel*, *...etc.*

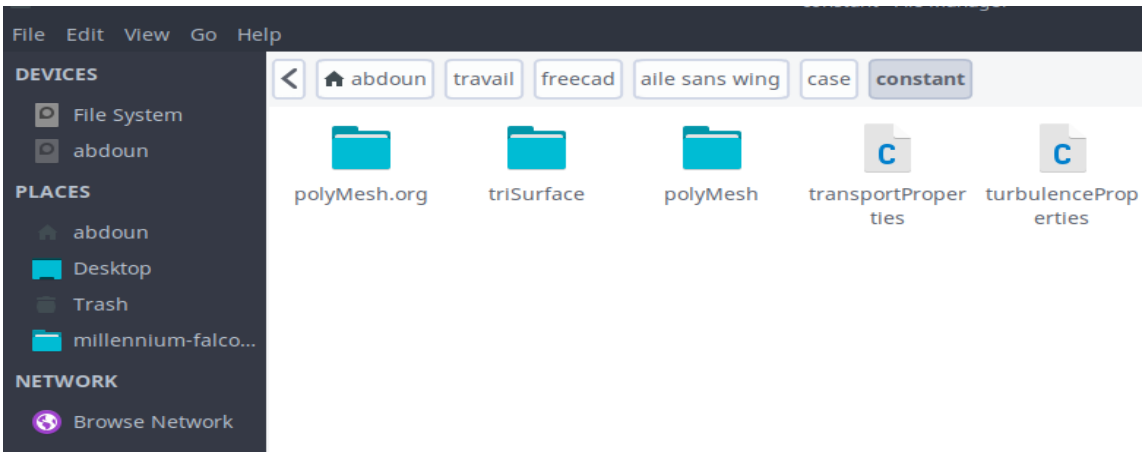


Figure II.12: *Dossier « constant ».*

La modélisation de la turbulence est définie dans le fichier *turbulenceProperties* dans lequel on doit spécifier le modèle adopté (RANS, LES) qui sont définis dans le *userGuide* de *OpenFOAM*.[12]

II.5.4.3- Dossier « system »

Ce dossier contient trois fichiers indispensables pour lancer une simulation par *OpenFOAM:*

- Le fichier c*ontrolDict* dans lequel est
- Le fichier *fvSchemes* dans lequel
- Le fichier *fvSolution* dans lequel

II.5.5- Outil de visualisation et Post-traitement

Le principal outil de post-traitement fourni par *OpenFOAM* est *paraFoam*. Cet outil open source est en fait un script qui lance le module de lecture fourni par *OpenFOAM*. Il est exécuté en tapant dans un terminal la commande *paraFoam* au niveau du dossier du cas en question.

OpenFOAM permet aussi d'accéder aux résidus au cours des itérations pour chaque grandeur à résoudre . Pour ceci, il suffit de taper la commande suivante dans le dossier du cas à étudier :

> **pyFoamPlotWatcher log.simpleFoam**

Dans ce cas nous avons mis à titre d'exemple le nom du solveur PlotWatcher , dans un autre cas, il faut simplement mettre le nom du solveur correspondant.

II.6- ParaView

II.6.1- Définition

« ParaView » est un outil open-source multi-plateforme, d'analyse des données et de visualisation. Les utilisateurs peuvent rapidement construire des visualisations pour analyser leurs données en utilisant des techniques qualitatives et quantitatives. L'exploration des données peut être effectuée de manière interactive en 3D ou par l'utilisation de la programmation sous forme de script en Python. « *ParaView* » a été développé pour analyser de très grands ensembles de données en utilisant les ressources de calcul à mémoire distribuée. Il peut être exécuté sur des supercalculateurs pour analyser des données de l'ordre du terra, ainsi que sur les ordinateurs portables pour des données plus petites.[11]

II.6.2- Fonctionnalités

- Prend en charge plusieurs plates-formes (binaires et source).
- Champs de visualisation et d'animation 2D et 3D scalaire, vecteur et tenseur.
- Analyse et manipulation des données de façons qualitatives et quantitatives.
- Interface graphique utilisateur intuitive et interactive.
- Rendu parallèle et distribué.

- Scriptable en Python.
- Prise en charge des plugins en XML et C ++ (VTK).

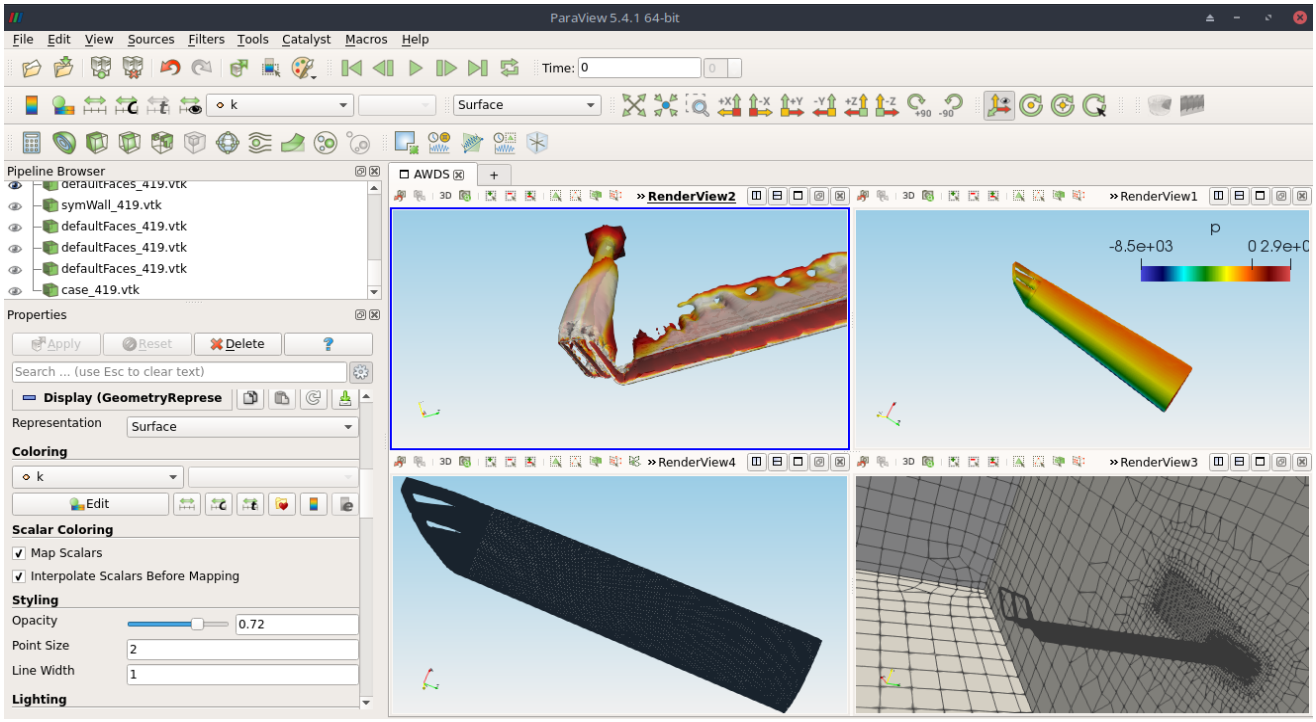


***Figure II.13**: Interface de ParaView.*

II.6.3- Formats supporté

« ParaView » supporte les formats les plus utililisés dans le domaine scientifique. A titre indicatif, nous avons :

boundary, cas, case, cosmo, cube, csv, dem, d3plot, e, ex, ex2v2, exo, exoii, foam, g, gen, gadget2, hierarchy, inp, isdyna, k, mha, mhd, nc, ncdf, netcdf, nhdr, nrrd, obj, particles, pdb, pht, ply, png, pop, pvd, pvti, pvtk, pvtp, pvtr, pvts, pvtu, raw, res, sesame, sos, spcth, stl, tec, tiff, tp, vpc, vrml, vthb, vti, vtk, vtm, vtmb, vtp, vtr, vts, vtu, wind, wrl, xdmf, xmf, xyz.[11]

II.6.4- Avantages

- Tutoriel instructif (avec des exemples de fichiers), documentation en ligne.
- Interface moderne et intuitive.
- Offre une variété de représentations et de filtres.
- Facilement accessibles des informations.
- Supporte l'hébergement à distance.

- Supporte les scripts Python pour écrire des marcos.
- Développement actif.
- Gratuit.

II.6.5- Explotation des résultats

Le meilleur moyen d'utiliser les résultats de la simulation avec *openFOAM* est de taper dans un terminal au niveau du dossier principal du cas à étudier :

> **foamToVTK -latestTime**

Après cette commande un répertoire VTK est créé et il contient toutes les données de la simulation à visualiser à travers des fichier *.vtk qui est un format très utilisé par « ParaView ».

II.6.6- Prise en main

Ouvrir « *ParaView* », menu File/Open, regarder le nombre de format de fichier lus au niveau de Files of type. Dans le panneau 'Properties', section 'properties', cocher toutes les variables puis cliquer sur 'Apply'. Dans le panneau 'Information', retrouver des informations sur le dataset
Dans les raccourcis des options d'affichage [13] :

modifier les valeurs des 2 listes déroulantes: la variable à afficher (Solid Color, Density, ...) ou le type de représentation (Outline, Wireframe, Points, Surface, …).
Tester les mouvements de la caméra
Pour aller plus loin : afficher et modifier le centre de rotation des données[13] 

II.6.7- La Color Map

La Color Map sert à traduire des valeurs scalaires en couleur et transparence via une fonction de transfert. C'est l'objet Lookup Tablede VTK. 

L'effet visuel est très dépendant des valeurs choisies pour cette lookup table.
Choisir comme mode de représentation 'surface', avec par exemple, la densité.

Utiliser le raccourci pour afficher l'éditeur de la color map.

Tester d'autres jeux de couleur préexistant :

Activer/désactiver la transparence («enable opacity»)

Tester la modification de la fonction de transfert pour les couleurs et l'opacité.

Diminuer le nombre de couleur (à 10 par exp).[13]

II.6.8- Organisation des écrans

Les panneaux de contrôle, accessibles via le menu «View».

Le panneau properties : Bien distinguer les 3 parties (on peut les replier): Properties, Display, View.

Retrouver les raccourcis utilisés en

Afficher les axes.

Afficher/enlever des panneaux. Par exemple, memory inspecteur, color map éditeur, ...

Les «layouts» : Représentation des données, sous différentes formes, appelées 'Views' comme pour les panneaux de contrôle —> Attention à la confusion.

Ajouter une vue via un nouvel onglet : cliquer sur le Layout #1 et choisir la vues 'SpreadSheet View' (la dernière) - ou la Slice View.

Ajouter une vue, sur le même écran : utiliser et ajouter un Render view.

En plus : Lier les caméras des rendus d'un layout : Click bouton droit souris sur une des 2 images, 'Link Camera' puis cliquer sur la deuxième image.[13]

II.6.9- Filtres

Les filtres les plus commun sont dans dans la barre d'outil:

Les filtres se paramètrent dans le panneau properties, section properties.

Menu Filters/Alphabetical --> Liste de tous les filtres. La fonction de recherche est pratique.

Pour ne pas avoir à cliquer sur Apply pour chaque modification, on peut activer le bouton auto Apply

Pour effacer tout le pipeline (filtre + source) : Edit/Reset Session [13]

Les filtres supplémentaires sont:

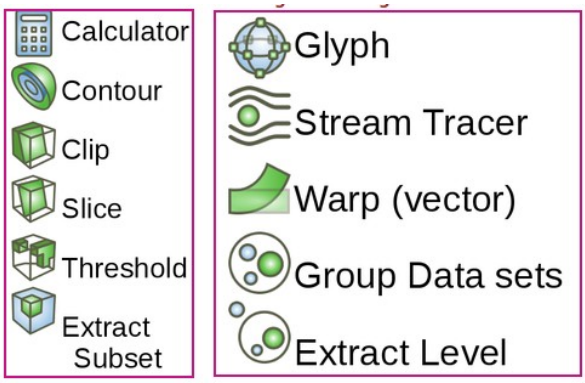


Figure II.14: *Filtres supplémentaires de « ParaView ».*

II.6.10- Animations et films

Ouvrir l'Animation View : Menu 'View'/'Animation View'[13]

Dans la list-box en bas sélectionner 'Caméra', puis cliquer sur le + pour ajouter une animation.

Laisser les autres options par défaut (ou pas).

Décocher si besoin l'animation 'Time Keeper - Time', et relancer l'animation.

Pour enregistrer une animation : menu 'File' / 'Save Animation'

Autre type d'animation : Variation de contour, déplacement d'objet le long de ligne de champ, … [13]

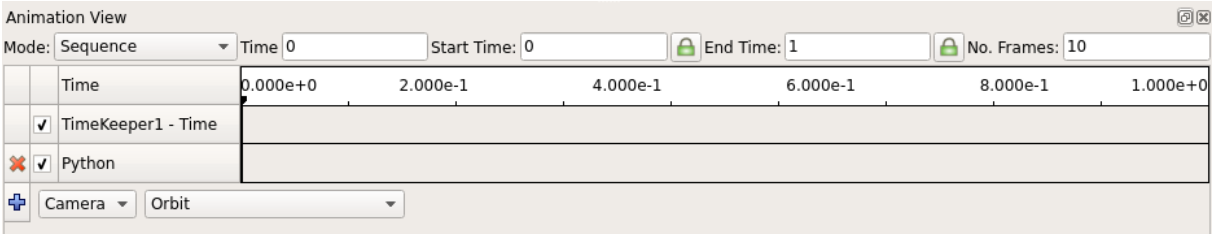


Figure II.15: *Barre d'animations et films.*

II.7- Conclusion

Dans ce chapitre, nous avons passé en revue tous les outils que nous allons utiliser pour mener à bien nos simulations. Ces logiciels, codes et utilitaires seront utilisés et appliqués à nos cinq cas de configurations afin d'étudier l'écoulement autour d'une aile muni de plusieurs formes de winglets. Ceci fera l'objet du prochain chapitre.

Chapitre III

Conception et simulation

III.1- Introduction

Dans ce chapitre, nous allons commencer par la conception des winglets utilisées ainsi que du domaine de calcul. Ensuite nous aborderons le maillage de ce domaine en indiquant les étapes essentielles, les qualités des différents maillages et les difficultés rencontrés durant cette étape cruciale. Nous finirons ce chapitre par la simulation avec *cfdOF* de l'écoulement autour des différentes configurations.

III.2- Conception des winglets

Il a été prévu, dans le plan de notre travail de concevoir les quatre types de winglets avec le logiciel « *FreeCAD* ». Malheureusement, par manque de temps, nous n'avons pas pu le faire et nous avons utilisé directement ceux conçues par A. Zakkour [1] mais sous forme de fichiers « *.stl ». Ce type de format nous a posé un grand problème au niveau de sa lecture par « *FreeCAD* » et nous avons perdus beaucoup de temps. Finalement nous avons résolus ce problème par la conversion de ces fichier à l'extension « *.stp » qui est complètement reconnue par « *FreeCAD* ». L'exemple ci-dessous (fig.III.1) nous montre les détails des deux formats.

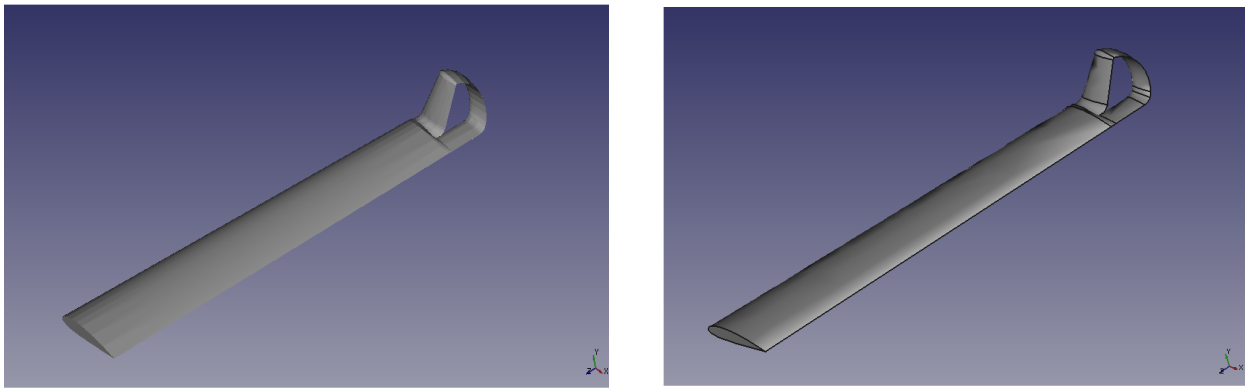


Figure III.1: *Aile avec winglet Demi-circulaire : format STL (gauche) et format STP (droite).*

III.3- Conception de l'aile

Comme nous ne disposons pas du fichier STL de l'aile seule, les données de conception ont été relevées à partir de notre article de référence [1] sur lequel nous nous sommes basé (Fig.III.2). Ces dimensions nous ont permis de concevoir l'aile avec le logiciel « *FreeCAD* » tel qu'il montré sur la figure (Fig.III.3) ci-dessous :

Les désignations et les formes des différentes configurations sont mentionnées respectivement dans le tableau (Tab. III.1) et la figure (Fig.III.4) ci-dessous :

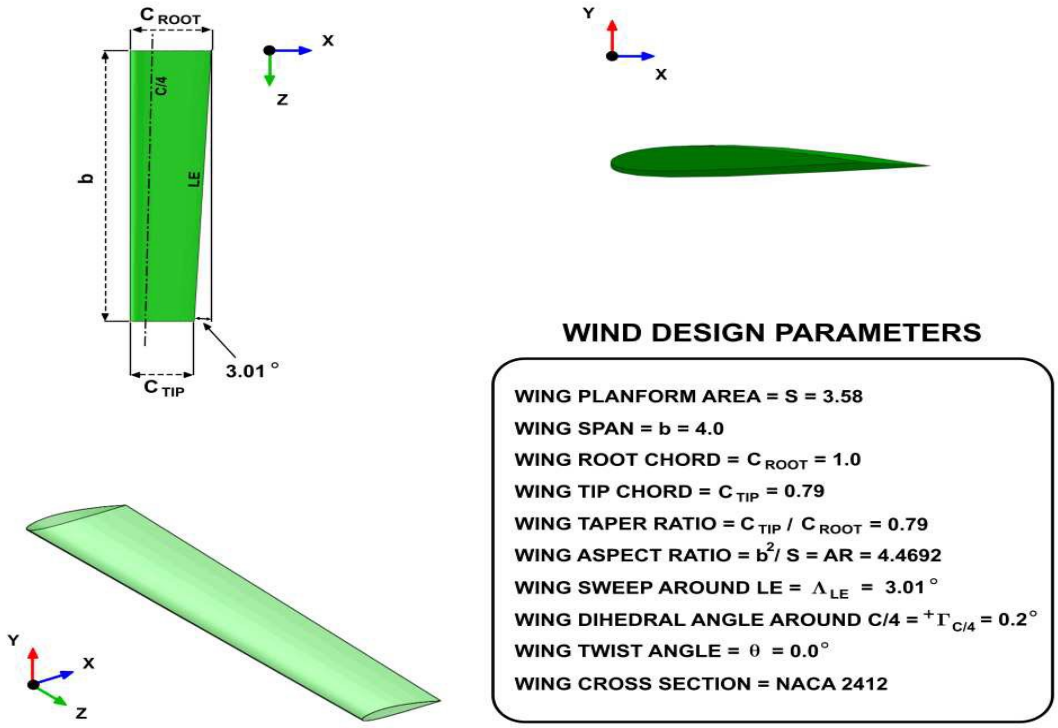


Figure III.2: *Données de conception de l'aile [3].*

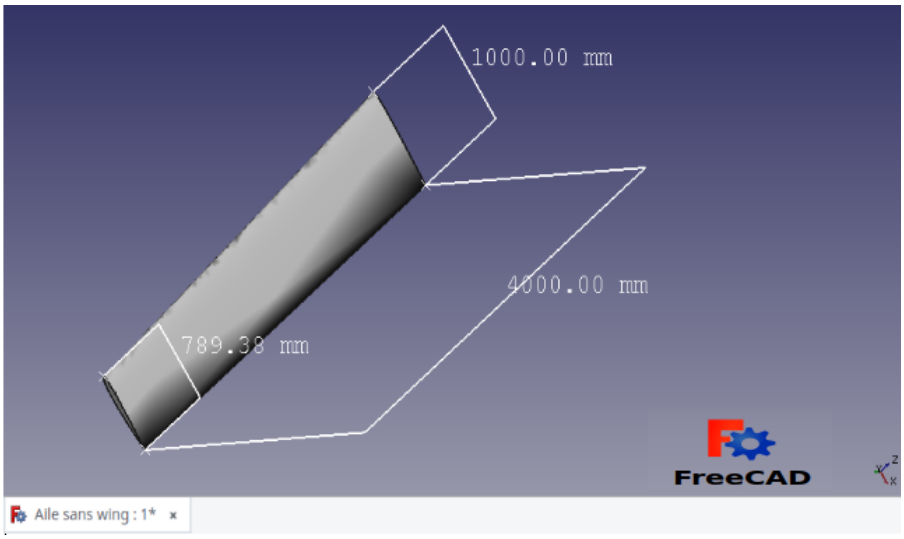


Figure III.3: *Réalisation de l'aile avec « FreeCAD ».*

Abréviation	***Désignation***
AWDC	Aile avec winglet *Demi-circulaire*
AWS	Aile avec winglet *Spiroidal*
AWDS	Aile avec winglet *Double Spiroidal*
AWW	Aile avec winglet *Whitcomb*
ASW	Aile Sans *Winglet*

Tableau III.1: *Abréviations et désignations.*

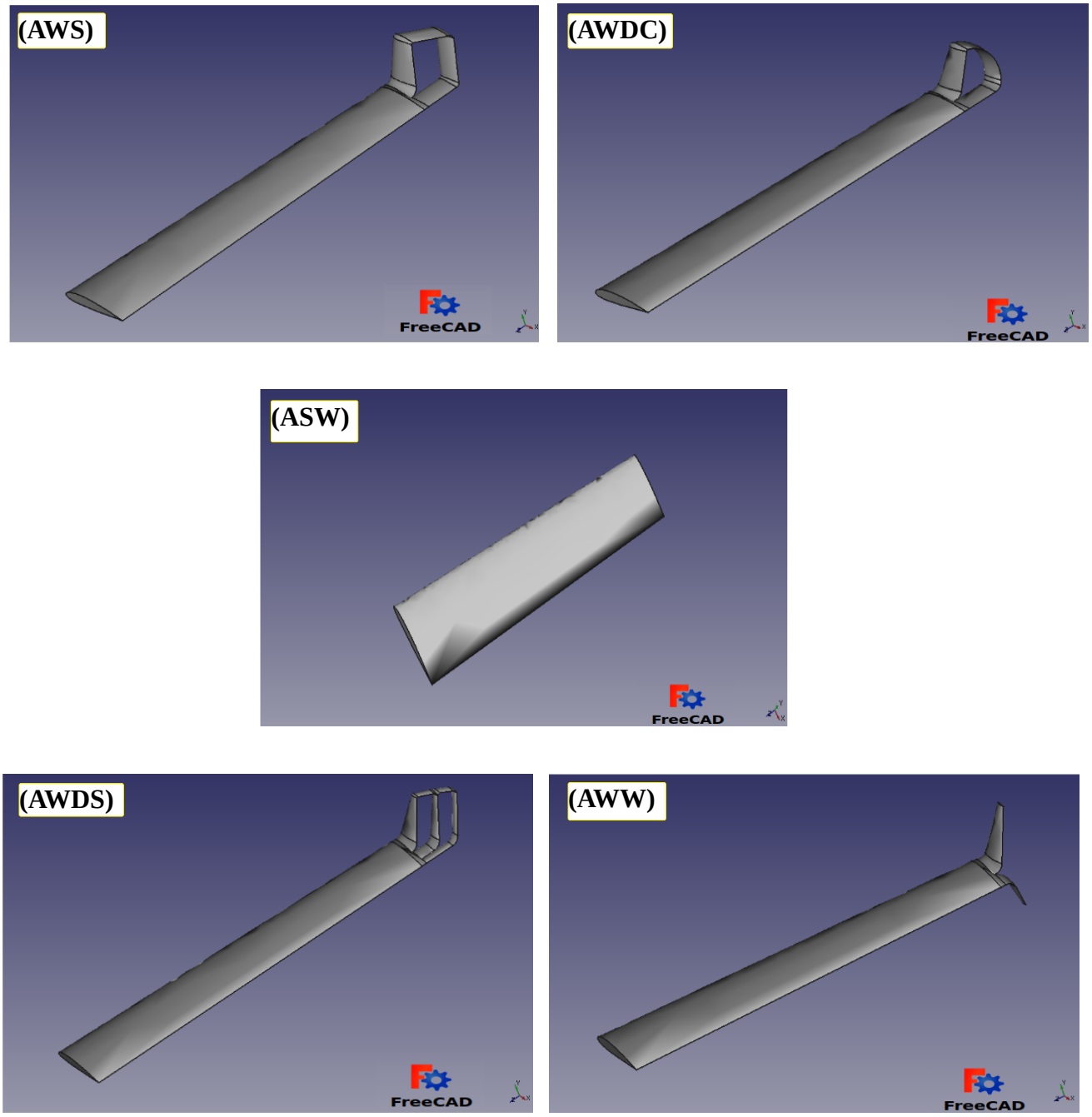


Figure III.4: *Formes des différentes winglets.*

Il est à noter que toutes les configurations des ailes sont calées à 8° (angle d'attaque) par rapport au plan horizontal de l'avion, ceci correspond à la phase de décollage où les tourbillons atteignent des valeurs maximales.

III.4- Domaine de calcul et volumes de contrôle

Le domaine de calcul avec lequel nous avons travaillé est le même pour tous les cas. La figure (Fig.III.5) ci-dessous présente les dimensions de ce domaine ainsi que les types de conditions aux limites utilisées.

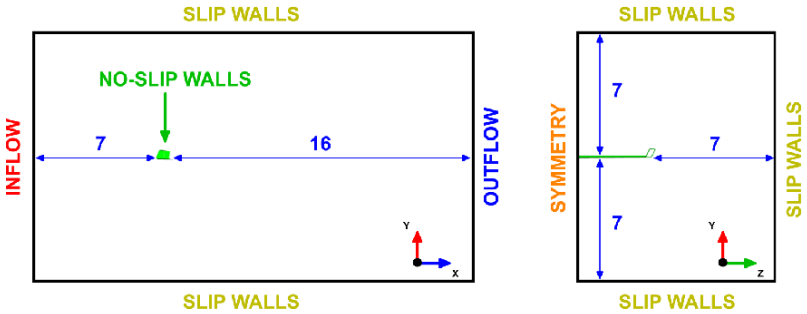


Figure III.5: *Domaine de calcul (distances en m).[3]*

Le domaine de calcul étant très grand pour être maillé avec la même finesse, il est alors nécessaire de faire recours aux volumes de contrôles. Nous avons utilisé pour tous nos cas de simulations deux volumes de contrôles. Le premier, de forme parallélépipédique, situé juste derrière l'aile afin de capter le sillage de l'aile et le second, de forme conique de langueur 12 m, englobant et derrière la winglet afin de capter les modifications apportées par la winglet sur l'écoulement (Fig.III.6).

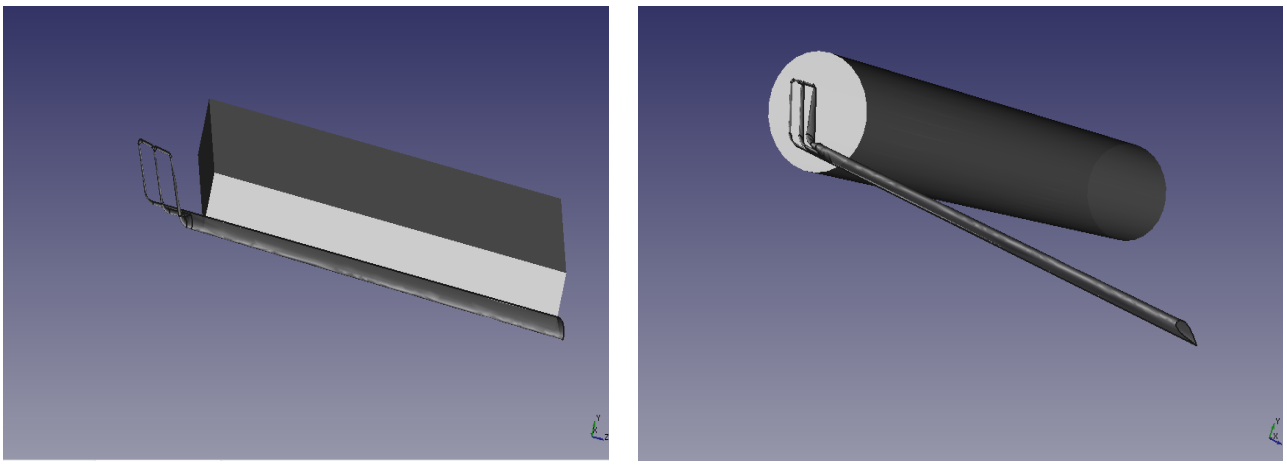


Figure III.6: *Volumes de contrôle derrière l'aile et la winglet*

III.5- Macros-commandes et Python

Une *macro* est un moyen pratique et facile d'automatiser une série de commandes dans « *FreeCad* ». Il suffit d'enregistrer une série de commandes puis de les sauvegarder dans un fichier. Une fois cet enregistrement (macro) sauvé, son exécution se fera d'une manière automatique. [9]
Ces macros sont en réalité une liste de commandes écrites en langage *Python* que l'on peut également modifier pour créer des scripts très complexes (en créant des boucles par exemple).

Alors que les scripts *Python* ont normalement pour extension .py, les macros « *FreeCAD* » doivent avoir comme extension « .FCMacro ». Une collection de macros écrites par des utilisateurs expérimentés se trouve dans la page Macros (Fig.III.7).
Les commandes, qui servent à créer des macros, se trouvent sur la barre d'outils : 

boutons: *Enregistrement*, *Arrêt* de l'enregistrement, *Édition* de la macro et *Exécution* de la macro.[9]

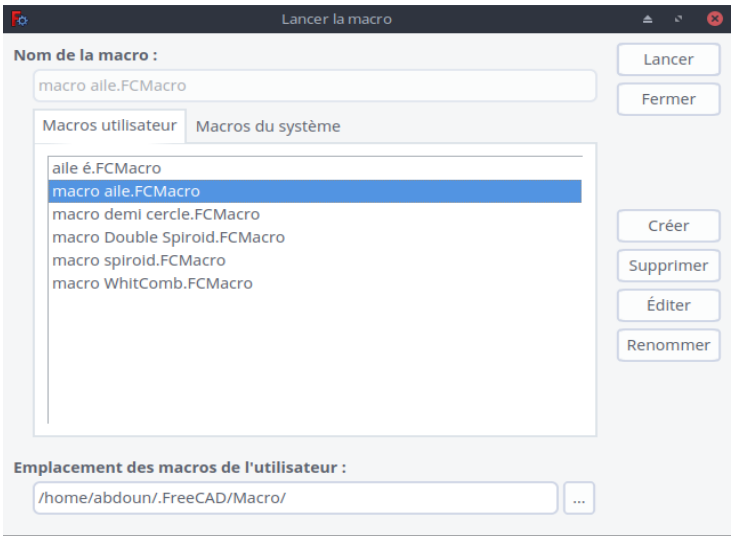


Figure III.7: *Interface de gestion des macros.*

Python est un langage de programmation très simple à utiliser et très facile à apprendre. Il est open-source, multi-plateforme et peut être utilisé pour un large éventail d'applications, de la programmation de simples scripts sur console à des programmes très complexes. Mais l'une de ses utilisations les plus répandues est comme langage de script, car il est facilement intégrable dans d'autres applications. C'est exactement de cette manière qu'il est utilisée dans « *FreeCAD* ». A partir de la console python, ou à partir de vos propres scripts, vous pouvez programmer « *FreeCAD* » et lui faire exécuter des commandes très complexes pour lesquelles il n'y a pas encore d'outils disponibles dans l'interface graphique. [9]

Nous avons programmé, pour les cinq configurations, cinq macros en langage *Python* et nous présentons un exemple de la configuration ***AWDS*** dans l'annex (A1).

III.6- Moyens de simulation

Comme moyens de calculs, nous avons utilisé un micro-ordinateur relativement moyen comparés à ceux utilisés en aérodynamique. Nous utilisons un processeur i7 à 4 cœurs et 8 Gb de RAM. Ces moyens sont insuffisant pour aboutir à des résultats précis mais suffisants à des fins pédagogiques.

III.7- Maillages

Nous avons utilisé le mailleur *cfMesh* de type cartésien à travers l'atelier *cfdOF* dans *« FreeCAD »*. En cas de modifications, les commandes utilisées dans le terminal sont : *Allmesh* et *cartesianMesh*. Les paramètres du maillage sont mentionnés dans le tableau (III.2) ci-dessous :

Mailleur		*Cartésien*
Taille de l'élément de base		*1 m*
Epaisseur de raffinement		*40 mm*
Paramètres de raffinement(cfMesh) taille relative de l'élément	Winglet	*0,015*
	Aile	*0,31*
	Cône	*0,63*
	Boîte	*0,63*
Nombre de couches limites		*15*
Taux d'expansion		*1,10*
Première hauteur de cellule		*4,8 mm*

Tableau III.2: *Paramètres du maillage.*

Nous avons passé beaucoup de temps à essayer de trouver le bon maillage puisque les résultats des simulations en dépendent. Nous avons rencontré beaucoup de problèmes dans les configurations qui sont différentes vue la forme des winglets n'est pas la même. Nous montrons ci-dessous (Fig.III.8-13) un exemple de maillage non réussi et les maillages acceptables avec lesquels nous avons travaillé et trouvé des résultats cohérents. Les détails de ces maillages sont mentionnés dans le tableau (Tab.III.3).

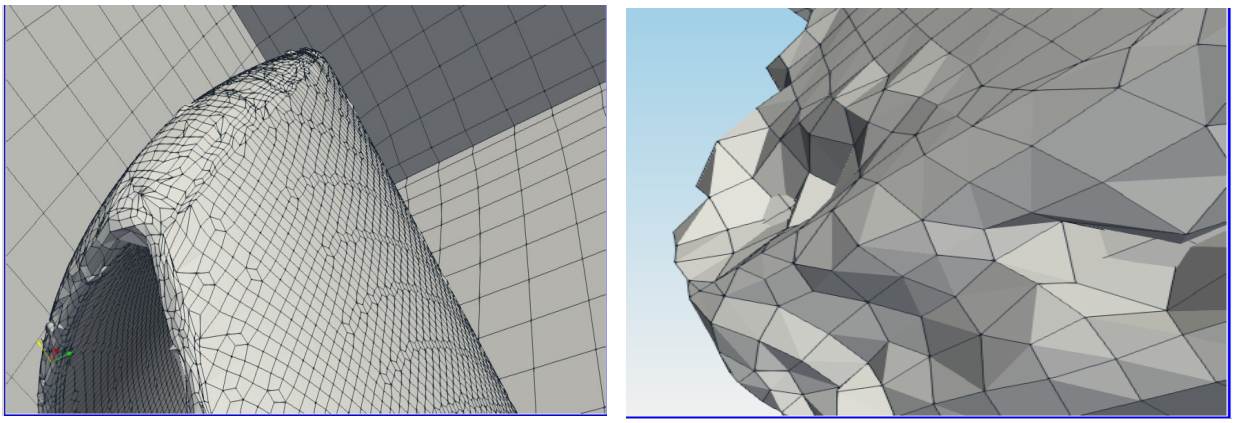


Figure III.8: *Maillage non réussit.*

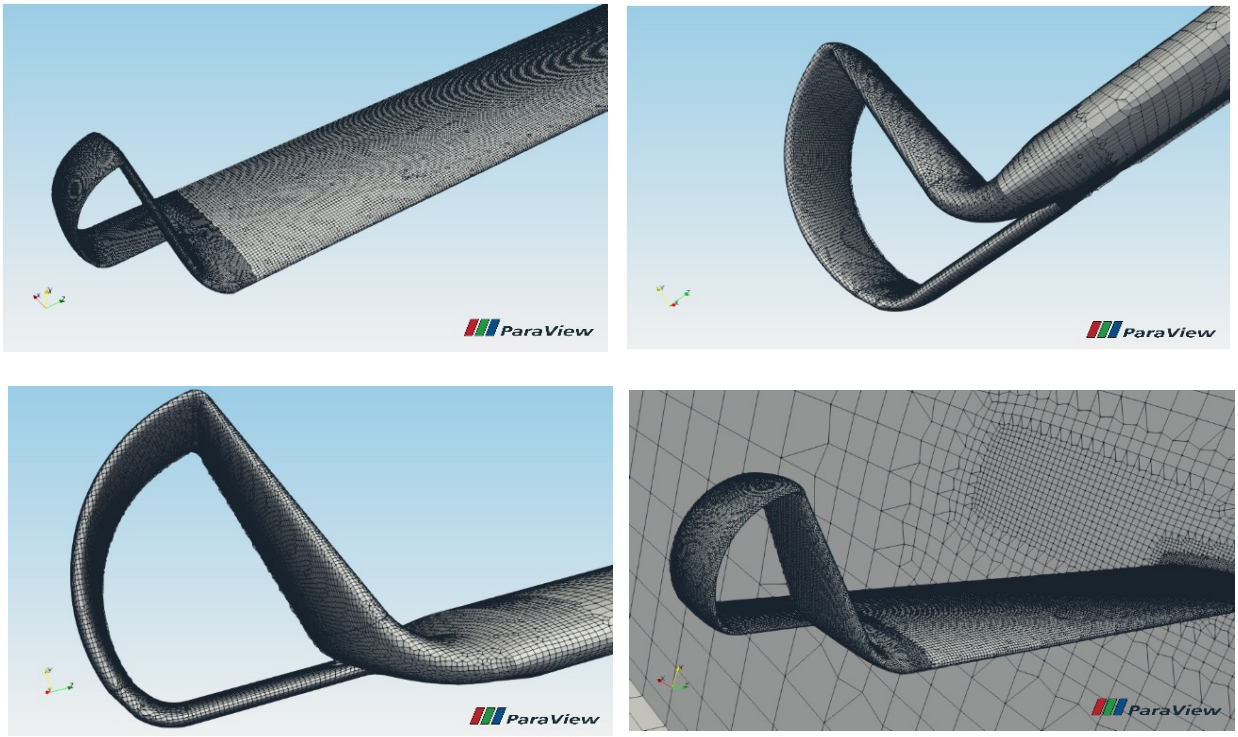


Figure III.9: *Maillage de la configuration AWDC.*

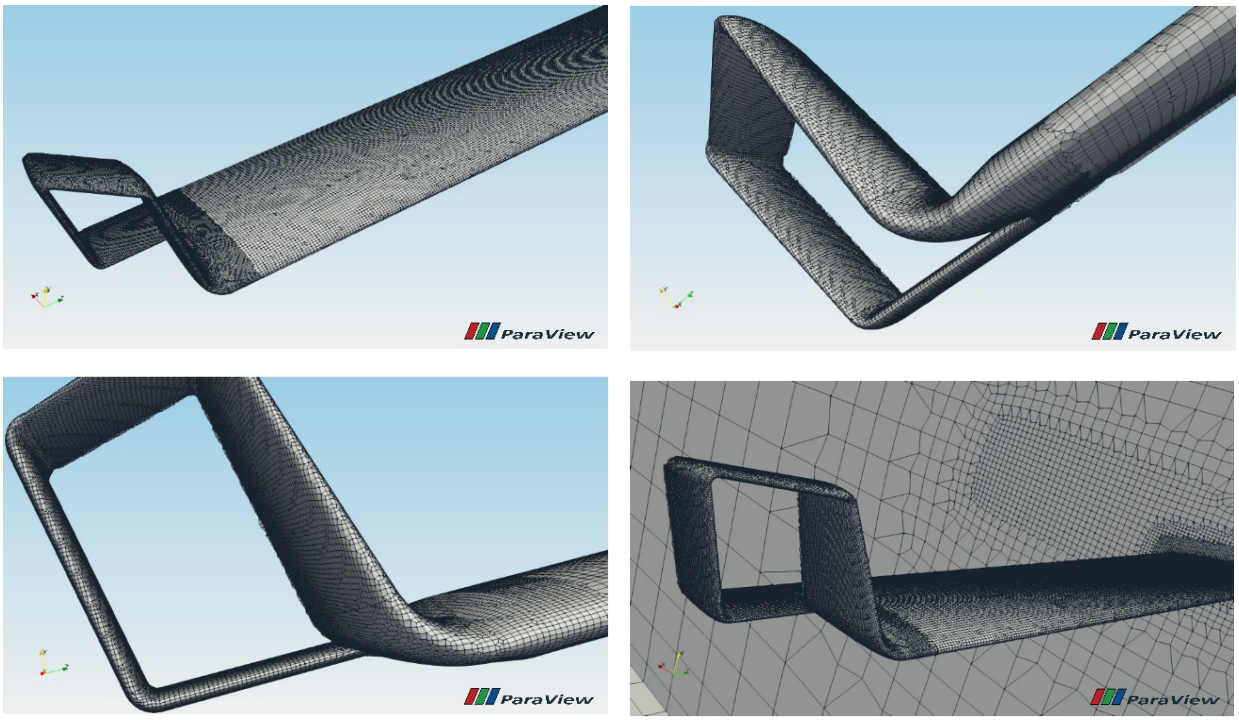


Figure III.10: *Maillage de la configuration AWS.*

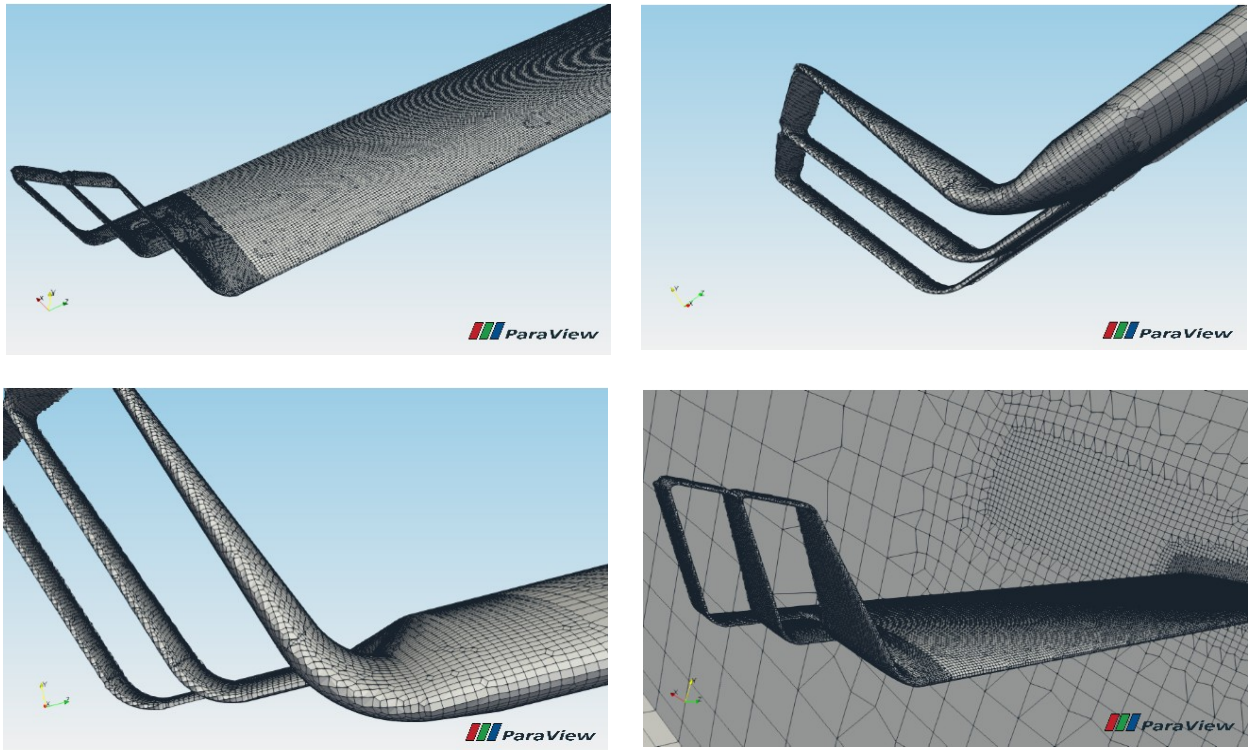


Figure III.11: *Maillage de la configuration AWDS.*

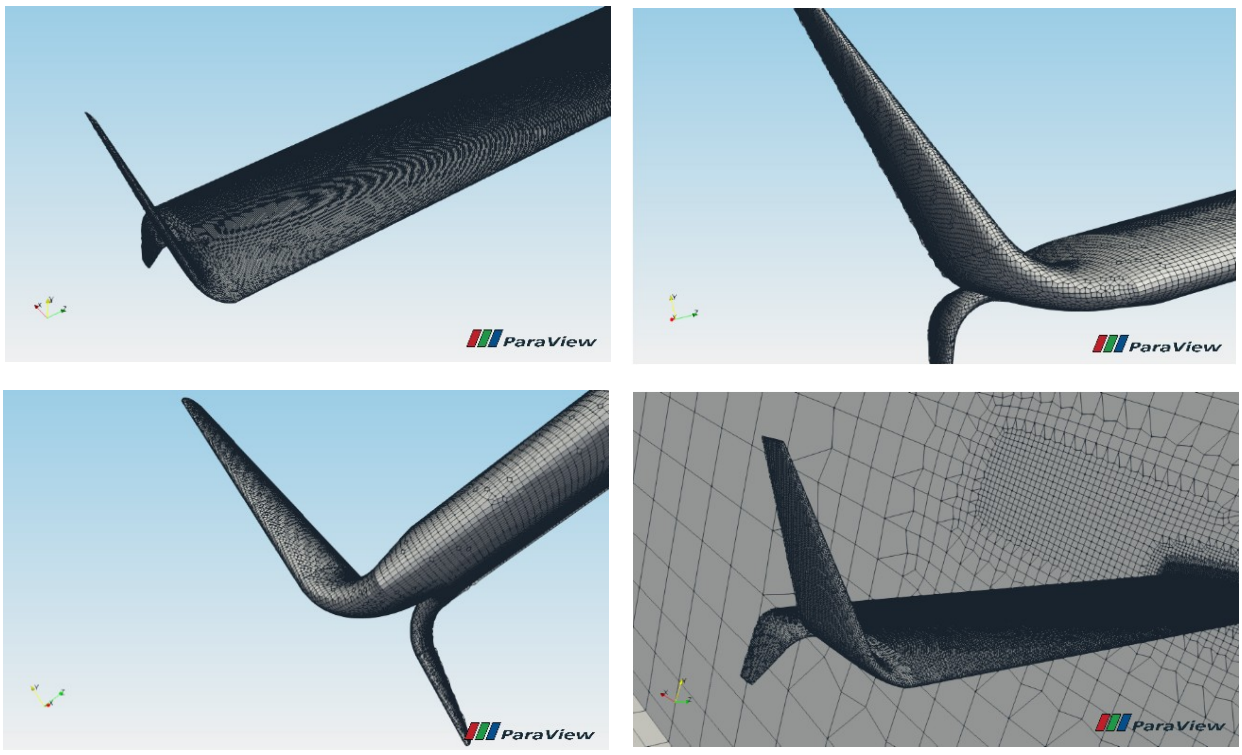


Figure III.12: *Maillage de la configuration AWW.*

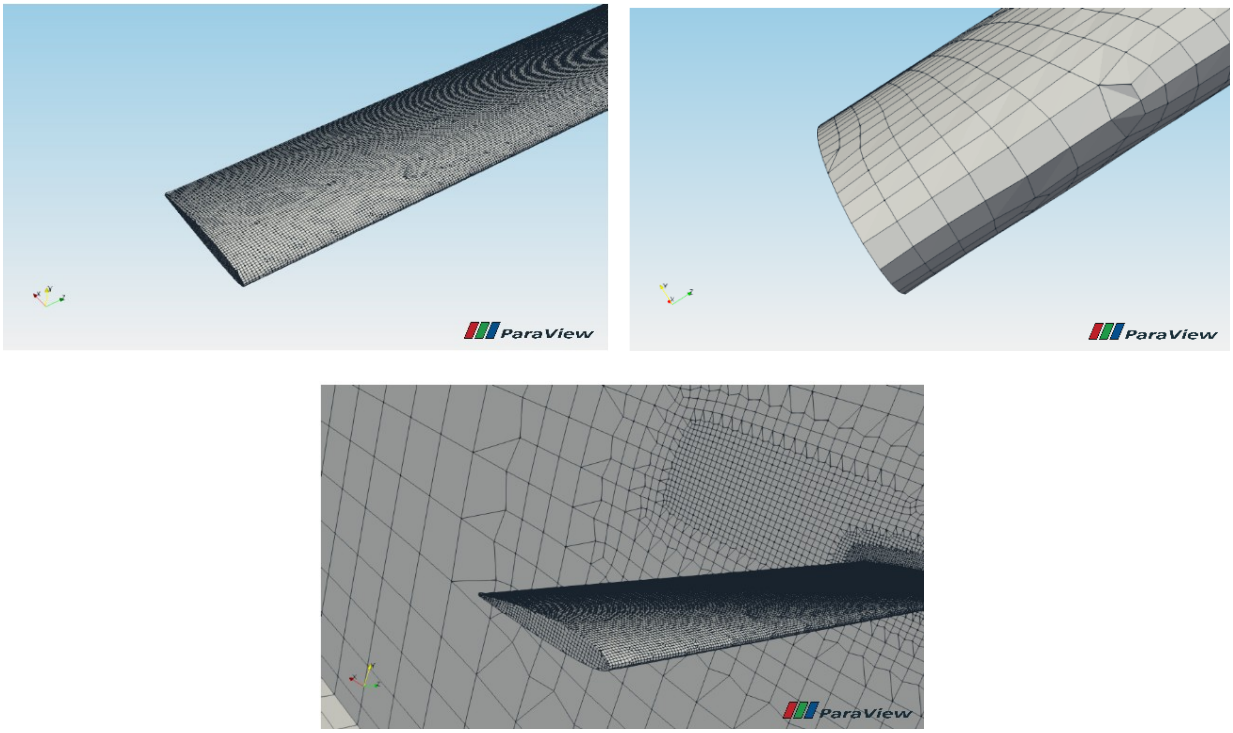


***Figure III.13**: Maillage de la configuration ASW.*

Cas	Nbre de cellules	Nbre de nœuds	Taille sur disque (MB)	Temps de maillage (mn)
ASW	*1 154 219*	*951 235*	*200*	*3*
AWDC	*1 847 032*	*1 534 941*	*320*	*6*
AWW	*4 737 475*	*4 109 619*	*830*	*14*
AWS	*2 123 405*	*1 771 770*	*300*	*14*
AWDS	*1 996 474*	*1 657 524*	*340*	*31*

***Tableau III.3**: Caractéristiques des maillages des différentes configurations.*

III.8- Conditions aux limites

Les conditions aux limites que nous avons introduis dans le logiciel « *FreeCAD* » dans l'atelier *cfdOF* et que nous avons utilisé pour nos différentes simulations sont mentionnés dans le tableau (III.4) suivant :

Modèle physique	Ecoulement	Incompressible visqueux
	Turbulent	RANS
	Model	kOmegaSST
Propriétés du fluide	Fluide	Air
	Densité	1.225 kg/m^3
	Viscousité dynamique	1.8e-05 kg/(m*s)
Vitesse (pendant le décolage)		70 m/s

Tableau III.4: *Conditions aux limites.*

III.9- Modifications

Dans cette partie, nous allons parler des modifications dans les fichiers de *meshDict* de *cfMesh* et *controlDict* d'*openFOAM*. Dans le premier, nous pouvons modifier toutes les données de maillage et la géométrie, de la zone de raffinement (box, cone) ainsi que du nombre de couches limites.

Dans le second fichier *controlDict*, nous avons ajouter une fonction pour lire un autre fichier dans le dossier « system » grâce à la fonction #include "forceCoeffs". Elle permet de lire le fichier *forceCoeffs* qui contient les données relatifs au calcul des coefficients de traînée et de portance.

Le fichier *controlDict* permet de contrôler le temps de début et de fin du calcul, les pas de calcul et de sauvegarde des résultats et l'application ou solveur qui doit être lancé (*simpleFoam*). L'exemple ci-dessous nous montre les détails de ce fichier:

```
/*---------------------------*- C++ -*----------------------------*\
|  =========                 |                                     |
| \\      /   F ield         | OpenFOAM: The Open Source CFD Toolbox |
|  \\    /    O peration     | Version:  4.x                       |
|   \\  /     A nd           | Web:      www.OpenFOAM.org          |
|    \\/      M anipulation  |                                     |
\*-------------------------------------------------------------*/
FoamFile
{
   version    2.0;
   format     ascii;
   class      dictionary;
   object     controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
application          simpleFoam;
startFrom            startTime;
startTime            0;
```

```
stopAt          endTime;
endTime         1500.0;
deltaT          1.0;
writeControl    timeStep;
writeInterval   50;
purgeWrite      0;
writeFormat     ascii;
writePrecision  8;
writeCompression uncompressed;
timeFormat      general;
timePrecision   6;
runTimeModifiable true;
libs
(
   // Needed for availability of porous baffle boundary in potentialFoam
   "libturbulenceModels.so"
);
functions
{
 #include "forceCoeffs"
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
```

Dans le dossier « constant/polyMesh », on trouve le fichier *boundary* qui contient toutes les faces du domaine du calcul et de l'aile. Nous modifions leurs noms et leurs types selon les conditions aux limites (Tab.III.5).

Emplacement	nom	Renommer	Type
Faces du domaine	face0	*inlet*	*Patch*
	face1	*face1*	*slipwall*
	face2	*symwall*	*symmetry*
	face3	*face3*	*slipwall*
	face4	*face4*	*slipwall*
	face 5	*outlet*	*patch*
faces de l'aile + winglet	le reste des faces	*defaultFaces*	*wall*

Tableau III.5: *Modifications dans le fichier « boundary ».*

Le fichier *createPatchDict* nous permet de regrouper automatiquement les faces par type, comme le montre l'exemple ci-dessous:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.x                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      createPatchDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
pointSync false;
// Patches to create.
patches
(
    {
        name defaultFaces;
        patchInfo
        {
            type wall;
        }
        constructFrom patches;
        patches ( face6 face7 face8 face9 face10 face11 face12 face13 face14 );
    }
 {
        name slipWalls;
        patchInfo
        {
            type wall;
        }
        constructFrom patches;
        patches ( face1 face3 face4 );
    }
    {
        name inlet;
        patchInfo
        {
            type patch;
```

```
        }
        constructFrom patches;
        patches ( face0 );
    }
    {
        name symWall;
        patchInfo
        {
            type symmetry;
        }
        constructFrom patches;
        patches ( face2 );
    }
{
        name outlet;
        patchInfo
        {
            type patch;
        }
        constructFrom patches;
        patches ( face5 );
    }
);
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
```

III.10- Simulations

Maintenant que toutes les étapes ont été expliquées, nous allons montrer un seule exemple parmi les cinq configurations sur lesquelles nous avons travaillé. Le cas *AWDS* sera détaillé pas à pas jusqu'à la fin de la simulation en annexe (A1). Cette méthode est celle que nous avons suivie au début de notre travail. Mais après et afin de nous faciliter la tache, nous avons programmé, pour les cinq configurations, cinq macros en langage *Python* et nous présentons un exemple de la configuration ***AWDS*** dans l'annex (A2).

Une fois la macro (ou programme script Python) exécutée, toutes les étapes de conception (importation du fichier, calage de l'aile à 8°, création du domaine de calcul, opération booléenne et création des zones de raffinement), de maillage (raffinement de l'aile, du box et du cone, couches) et d'analyse CFD (modèle physique, propriétés du fluide, initialisation de l'écoulement, conditions aux limites) sont exécutées automatiquement sans aucune intervention de l'utilisateur.

De ce fait, toute modification (CL par exemple) se fera au niveau de la macro avant lancement.

Nous présentons, sur les figures (Fig.III.14-18) ci-dessous les résultats des cinq macros :

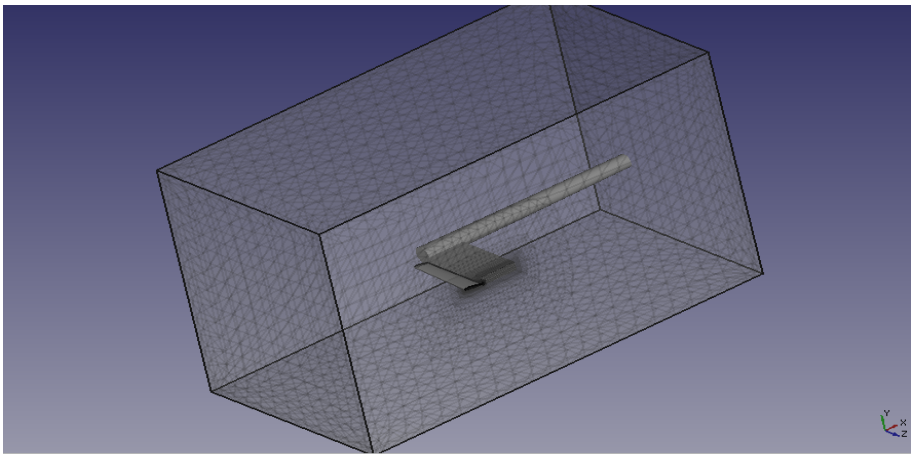


Figure III.14: *Conception de la configuration ASW.*

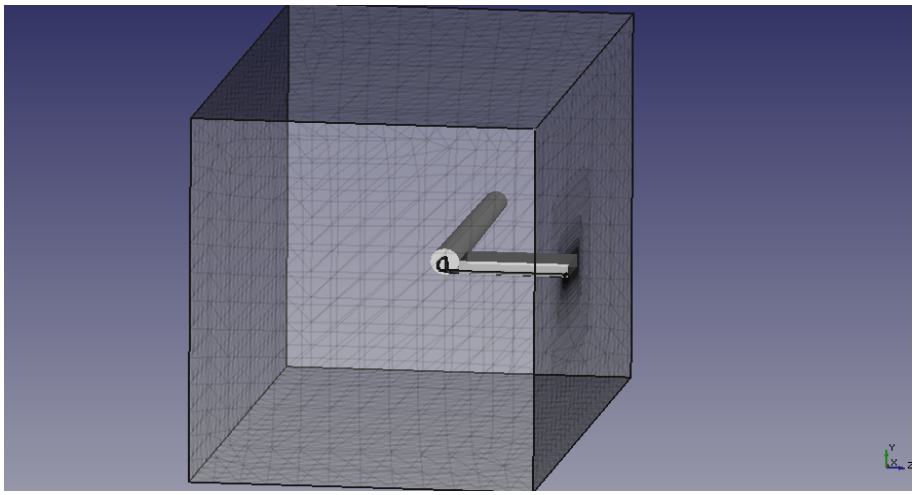


Figure III.15: *Conception de la configuration AWDC.*

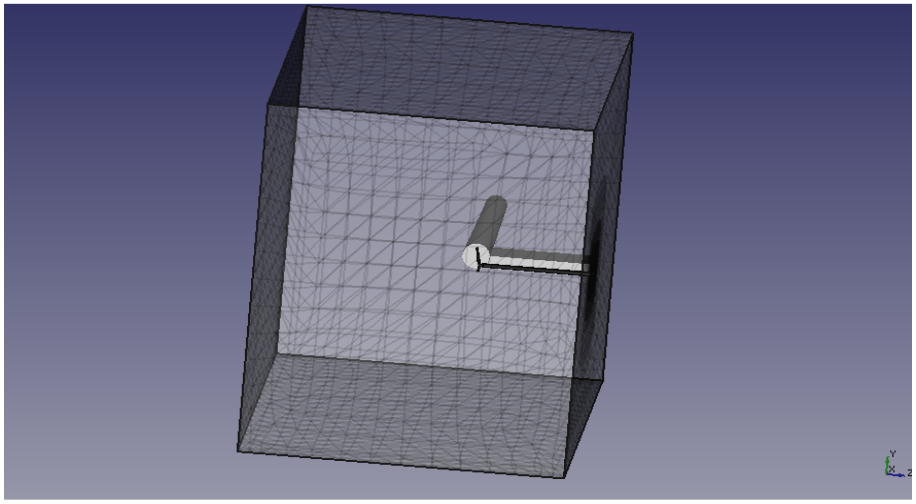


Figure III.16: *Conception de la configuration AWW.*

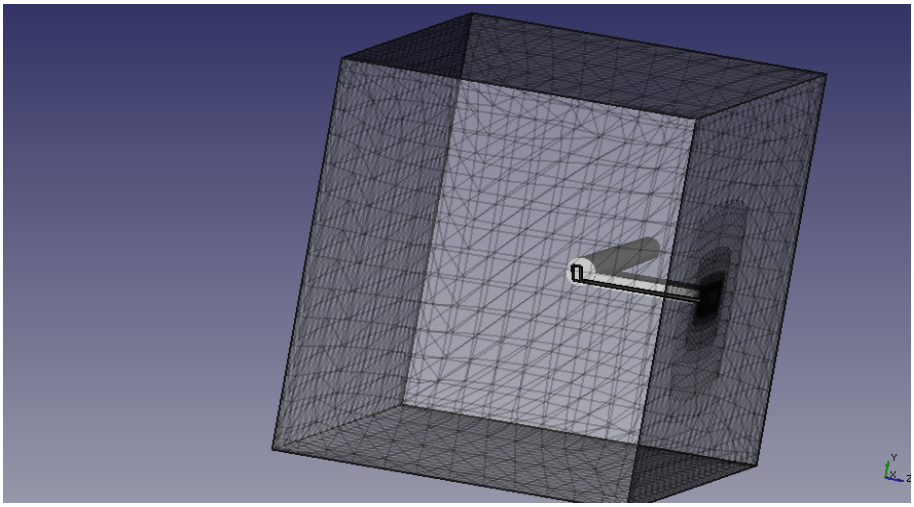


Figure III.17: *Conception de la configuration AWS.*

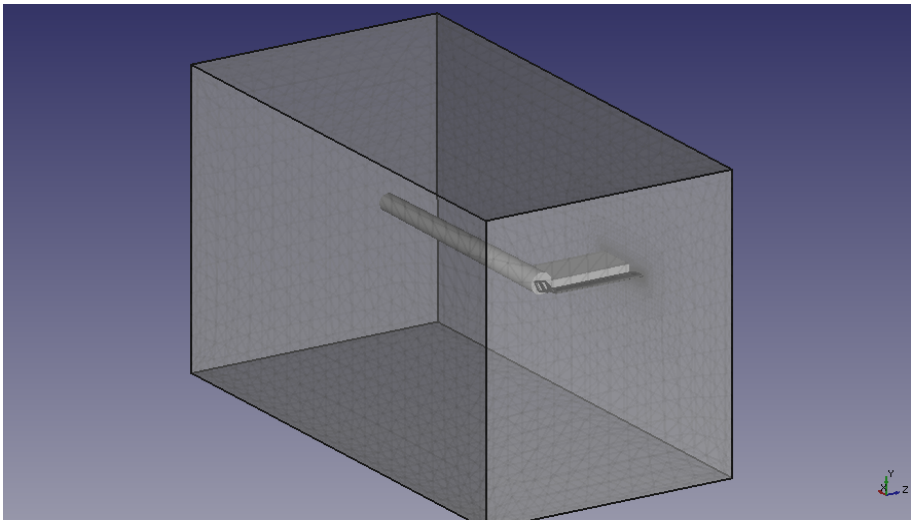


Figure III.18: *Conception de la configuration AWDS.*

Nous présentons, sur la figure (Fig.III.19) ci-dessous les résidus des calculs des différentes simulations sur lesquelles figure le temps de calcul de chacune des configurations :

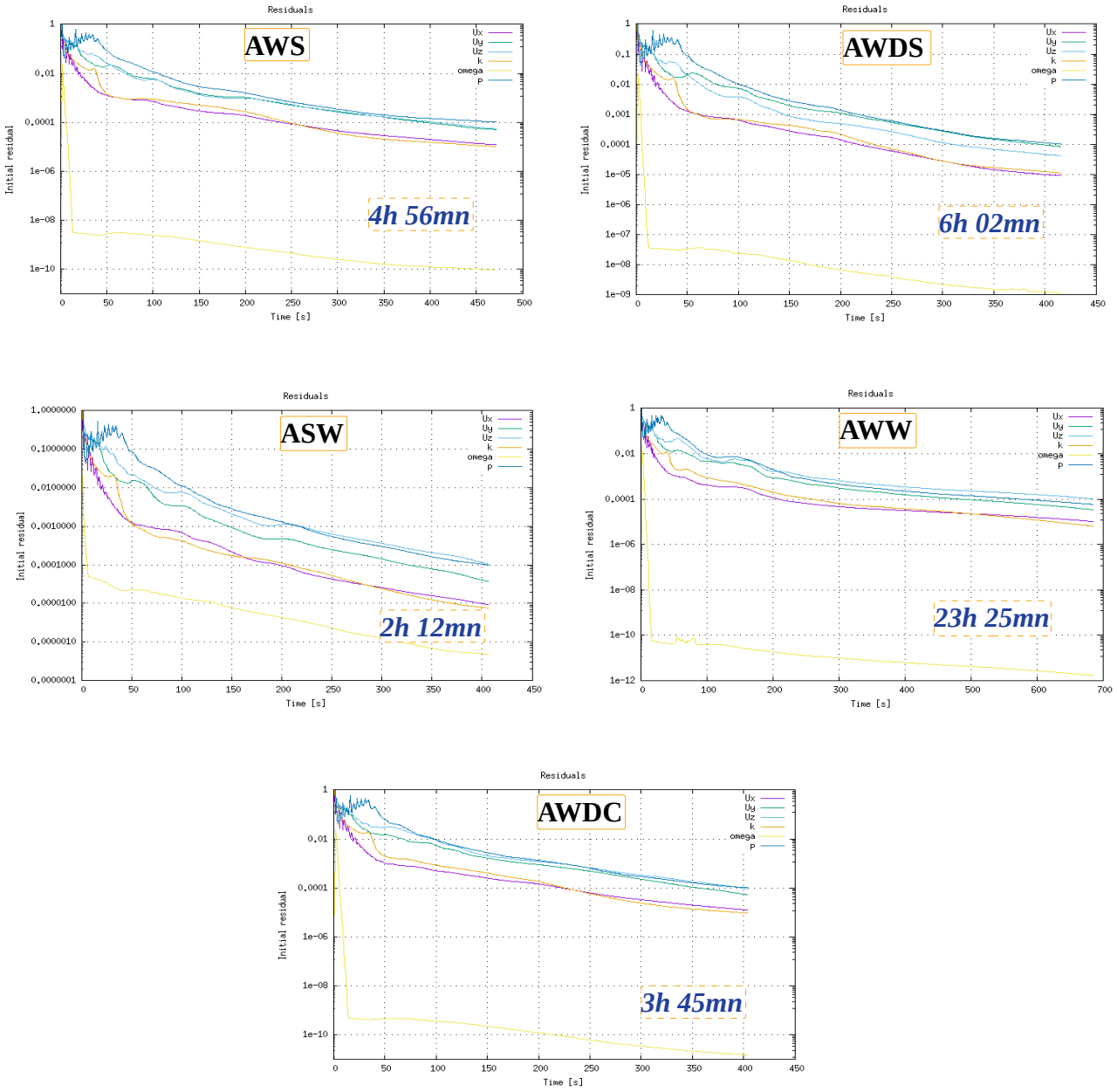


Figure III.19: *Résidus et temps de calculs des différentes configurations.*

Il est important de noter qu'avant de passer à l'exploitation des résultats avec « *ParaView* », nous devons lancer dans un terminal et au niveau du dossier principal les commandes suivantes qui nous permettrons d'exploiter les résultats à travers les fonctions existantes dans l'utilitaire *postProcess* à sa voir : le Q-criterion, lambda2 et la vorticity.

Ces commande sont:

- Q-critirion ====> postProcess -func Q
- Lambda ====> postProcess -func Lambda2
- Vorticité ====> postProcess -func vortisity

Maintenant, nous pouvons créer le fichier de résultats VTK qui doit être exploité par « *ParaView* » en utilisant la commande :

- > foamToVTK -latestTime

III.11- Conclusion

Au bout du compte, après tous les efforts et les nombreux échecs et en fonction des moyens dont nous disposons, nous avons obtenu des résultats acceptables. Nous avons beaucoup appris des programmes que nous avons utilisés dans notre expérience (*FreeCAD*, *paraView*, *cfMesh, Python* et *openFOAM*). Nous avons réussi à mailler toutes les configurations avec un maillage suffisant pour exploiter les résultats des simulations qui seront exposés dans le chapitre suivant.

Chapitre IV

Résultats et interprétations

IV.1- Introduction

Ce chapitre constitue le fruit de nos efforts considérables et de nos nombreuses tentatives de simulations pour lesquelles nous avons obtenu des résultats acceptables. Nous utiliseront uniquement le logiciel de visualisation « *ParaView* » et nous présenterons les résultats les plus significatifs des champs cinématique et dynamique pour chaque configuration.

Il est à noter que les résultats que nous présentons ne sont pas précis mais qui restent acceptables vis-à-vis des moyens informatiques dont nous disposons. En effet, les maillages utilisés dans la référence [2] sont de l'ordre de 14.10^6 de cellules, chose que nous pouvons atteindre mais le temps de simulation va être très élevé et qui est inacceptable pour mener à terme notre projet.

IV.2- Résumé des simulations

Nous présentons, dans le tableau (Tab. IV.1), les informations nécessaires relatives aux différentes configurations que nous avons étudié.

Cas	N^{bre} d'iterations	Temps de calcul	Taille sur disque [GB]
ASW	*410*	*02 h 12 min*	*0,671*
AWDC	*408*	*03 h 45 min*	*1,1*
AWW	*690*	*23 h 25 min*	*2,3*
AWS	*475*	*04 h 56 min*	*2,5*
AWDS	*419*	*06 h 02 min*	*1,4*

Tableau IV.1: *Comparaison entre les différentes configurations.*

IV.3- Coefficients aérodynamiques

La force de portance F_L et la force de traînée F_D sont calculées en intégrant les contraintes de pression et de cisaillement de la paroi sur la surface de l'aile pour chaque cas; ensuite, les coefficients de portance C_L et de traînée C_D sont calculés par les formules (I.1) et (I.2). Le tableau (IV.1) ci-dessous nous donne les valeurs de ces forces ainsi que le rapport des coefficients dans chaque cas de figure.

Cas	Portance F_L [N]	Trainée F_D [N]	C_L/C_D
ASW	11095,8	746,768	14,858
AWDC	12807,6	800,482	15,999
AWW	11543,8	**538,489**	**21,437**
AWS	**12889,2**	778,342	16,559
AWDS	12302,7	875,824	14,047

Tableau IV.2: *Coefficients de traînée et de portance.*

On note que le cas *AWDS* présente le rapport le plus faible et que le cas *AWW* est le plus optimal (meilleure performance de la winglet) puisqu'il présente le plus grand rapport C_L/C_D.

IV.4- Caractéristiques cinématiques et dynamiques de l'écoulement

IV.4.1- Pressions autour de l'aile

La pression influe sur le comportement de l'aile surtout si on parle du C_L et C_D la surpression et la dépression autour du profil d'aile se traduisent en forces qui sont appliquées sur l'aile.

Nous représentons (Fig.IV.1-5) le champ de pression autour de l'aile pour toutes les configurations.

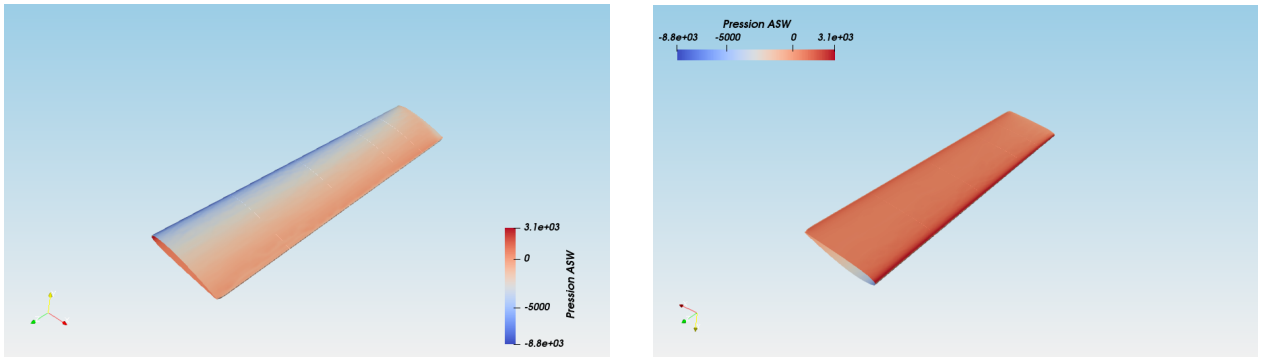


Figure IV.1: *Répartition de pression autour l'aile. Cas ASW.*

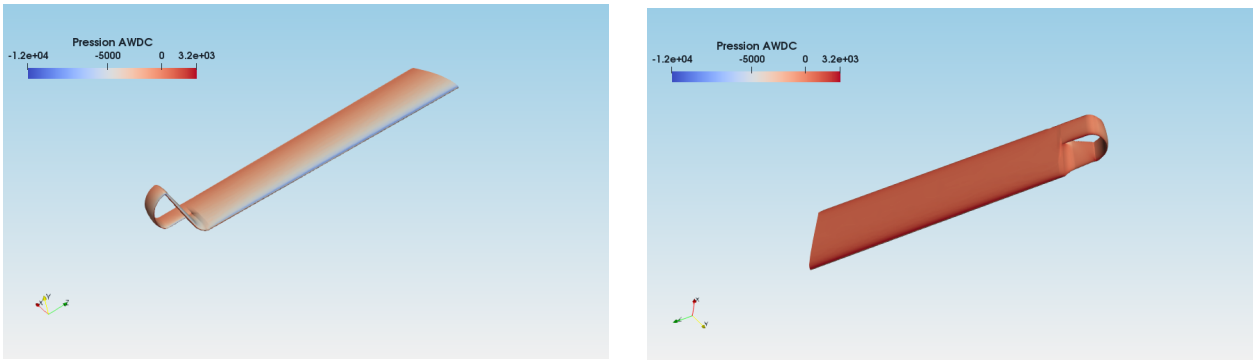


Figure IV.2: *Répartition de pression autour l'aile. Cas AWDC.*

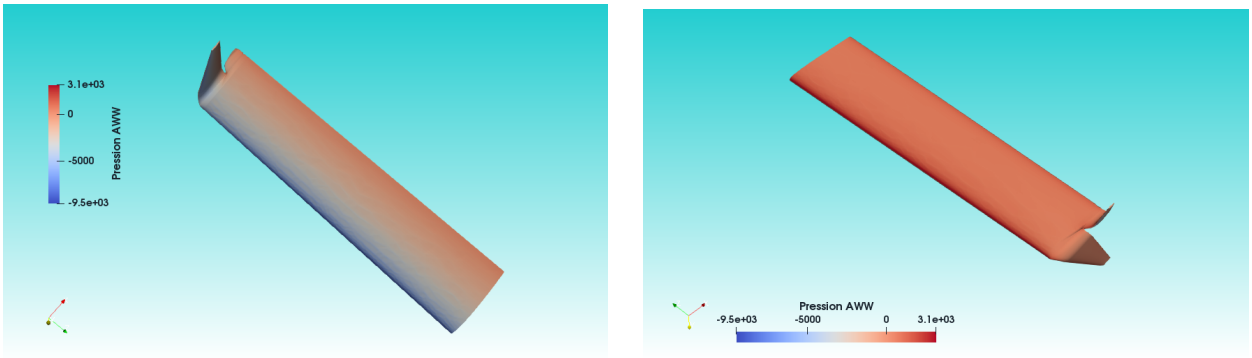


Figure IV.3: *Répartition de pression autour l'aile. Cas AWW.*

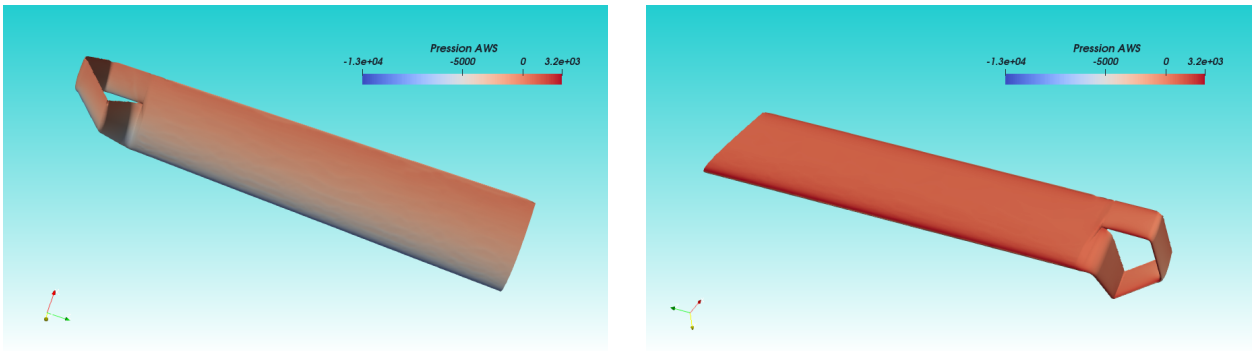


Figure IV.4: *Répartition de pression autour l'aile. Cas AWS.*

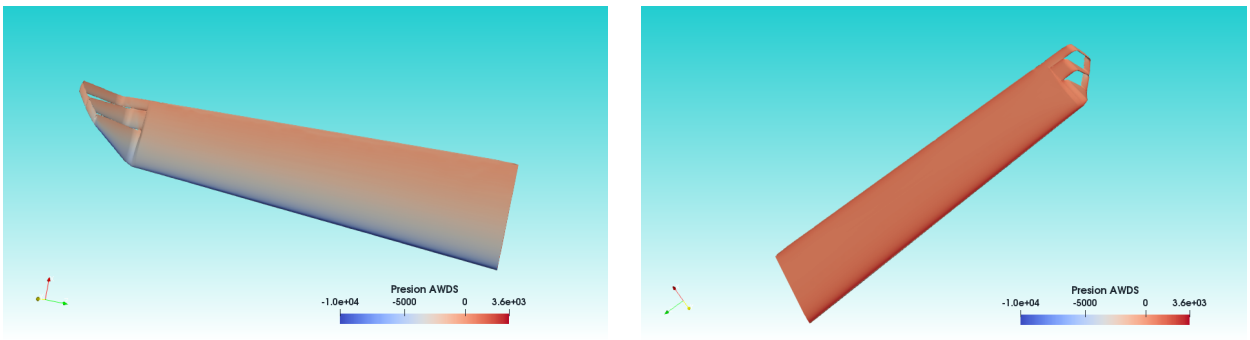


Figure IV.5: *Répartition de pression autour l'aile. Cas AWDS.*

On remarque bien la dépression au niveau de l'extrados et la surpression au niveau de l'intrados, ce qui crée la portance.

Les courbes ci-dessous (IV.7-11) représentent la répartition de pression autour de l'aile de chaque cas pour différentes valeurs de l'envergure Z (Z = -1 m ; Z = -2 m ; Z = -3 m ; Z = -3,5 m) (Fig.IV.6).

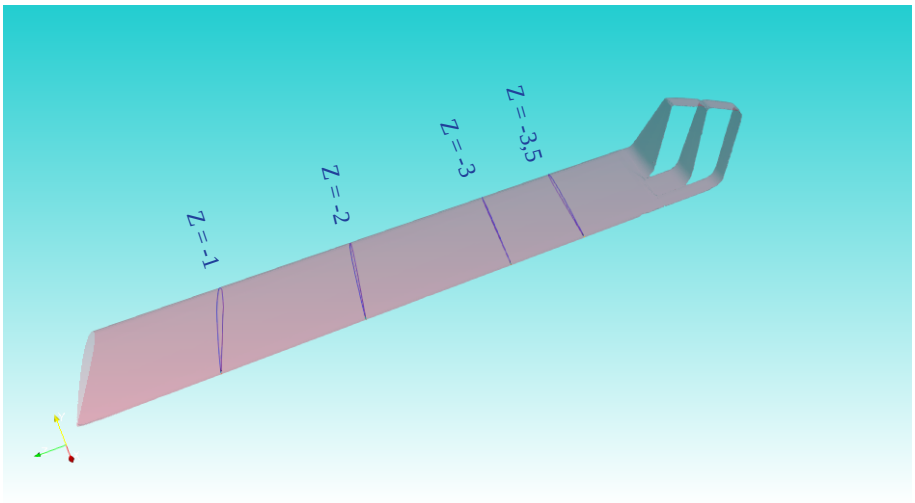


Figure IV.6: *Positions de relevée de la répartition de pression le long de l'aile.*

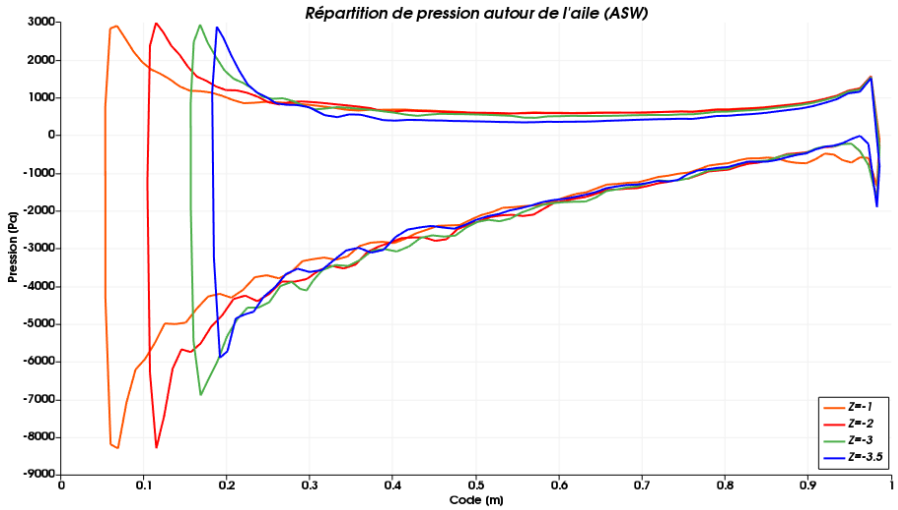


Figure IV.7: *Répartition de pression autour du profile l'aile. Cas ASW.*

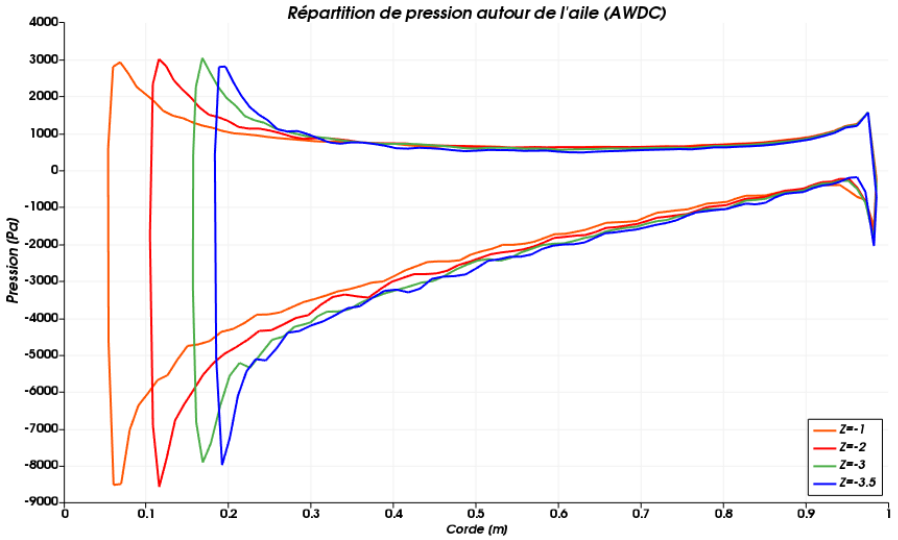


Figure IV.8: *Répartition de pression autour du profile l'aile. Cas AWDC.*

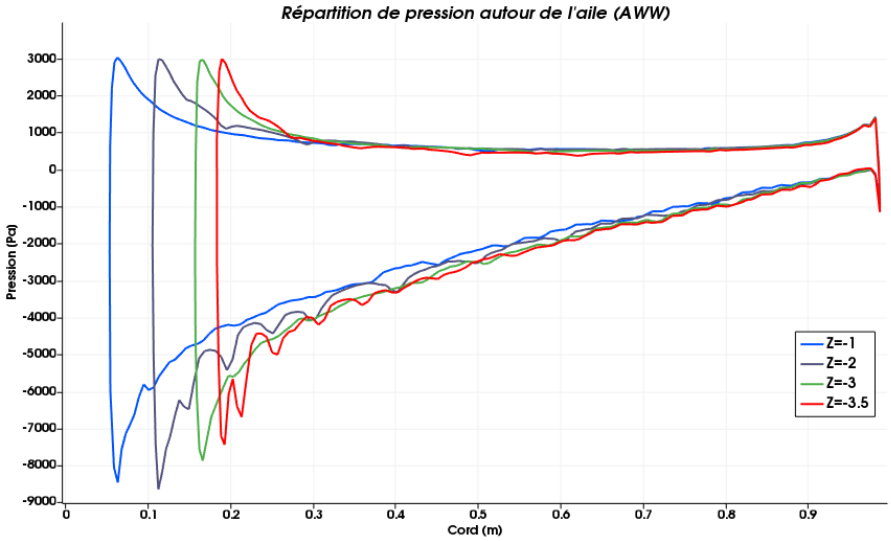


***Figure IV.9**: Répartition de pression autour du profile l'aile. Cas AWW.*

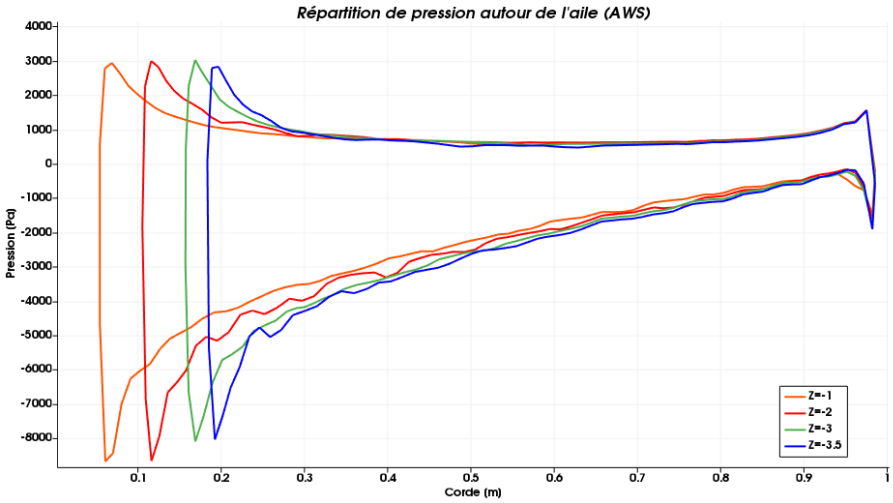


***Figure IV.10**: Répartition de pression autour du profile l'aile. Cas AWS.*

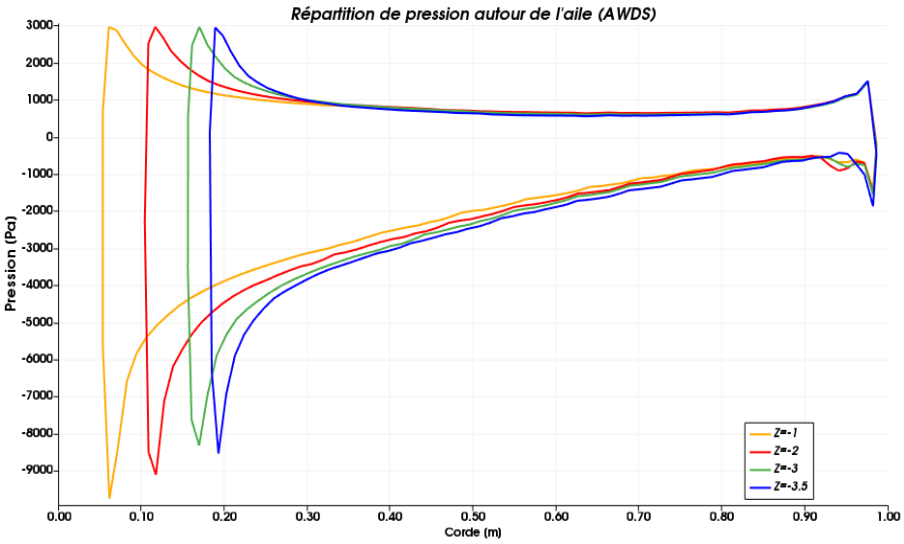


***Figure IV.11**: Répartition de pression autour du profile l'aile. Cas AWDS.*

Nous voyons que, les courbes sembles les mêmes mais il y a quelque différence entre eux et cela peut se remarquer en représentant toutes ces courbes sur le même graphique et à une même position de l'envergure (Fig.IV.12-15).

Nous remarquons qu'au niveau de l'intrados les différentes configurations son presque les mêmes. Par contre, au niveau de l'extrados, il y a une différence dans les courbes.

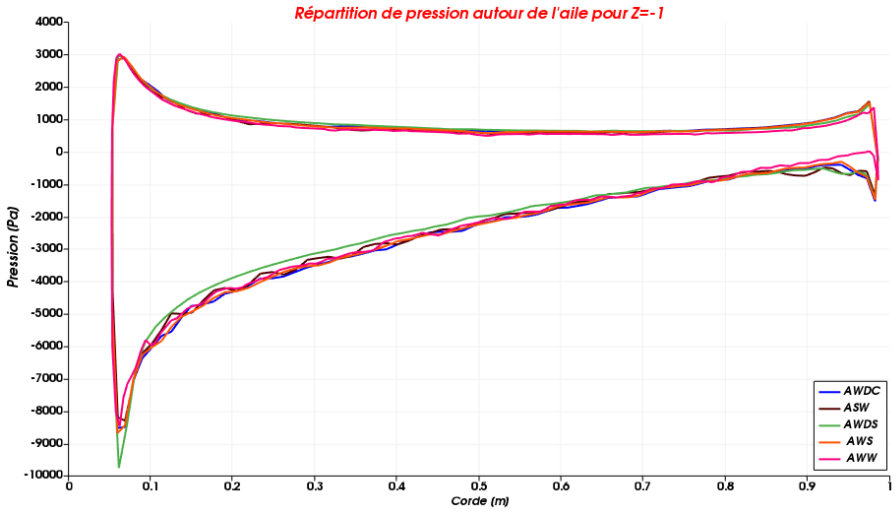


***Figure IV.12**: Répartition de pression autour du profile l'aile à z = -1 m.*

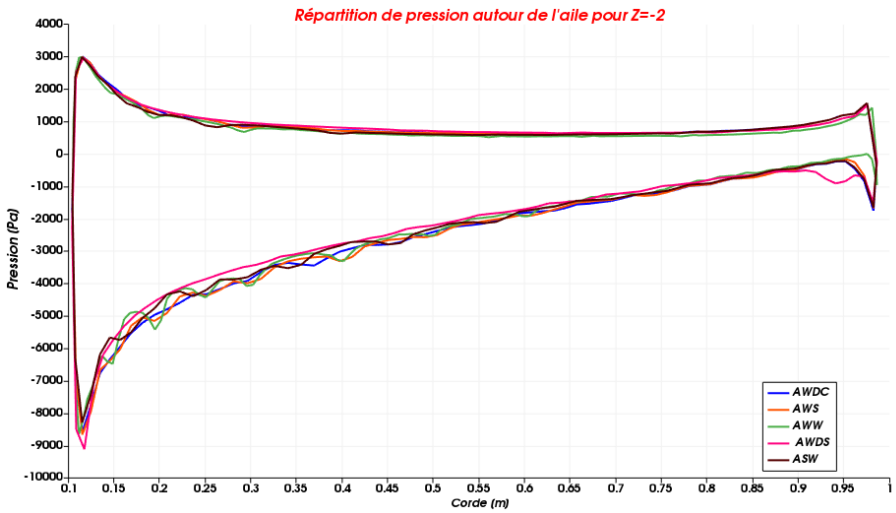


***Figure IV.13**: Répartition de pression autour du profile l'aile à z = -2 m.*

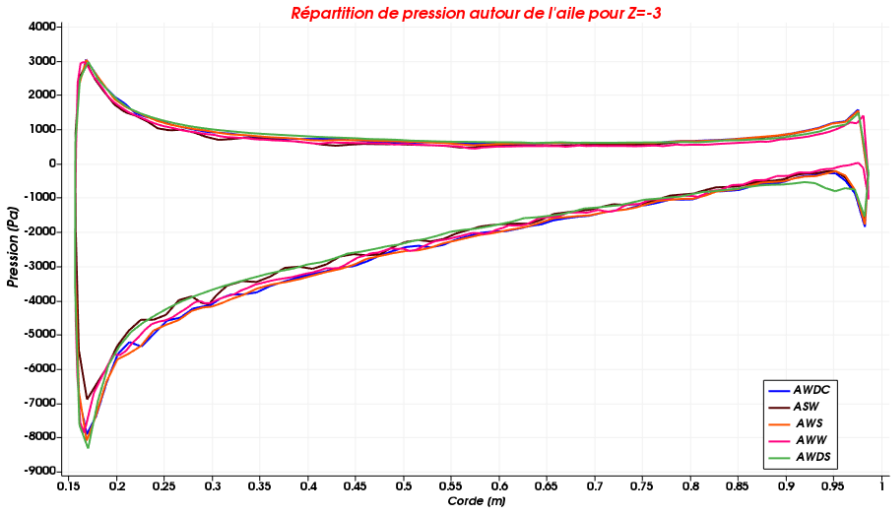


Figure IV.14: *Répartition de pression autour du profile l'aile à z = -3 m.*

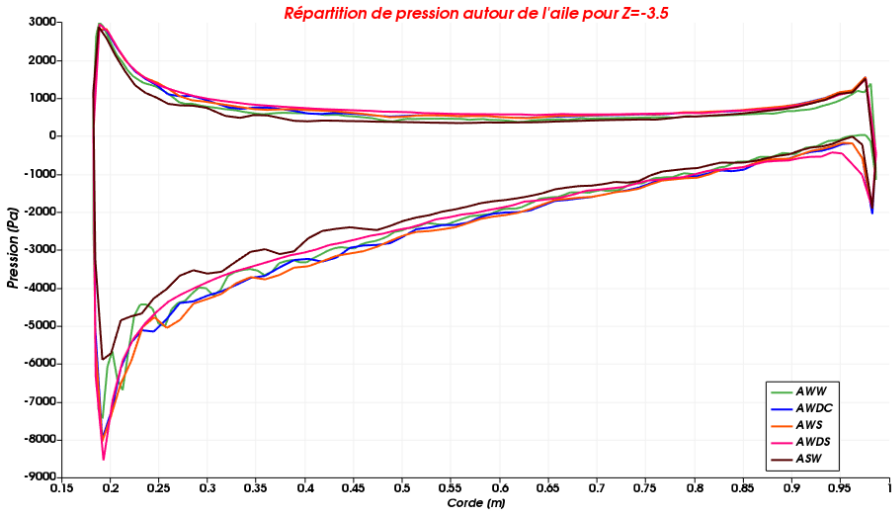


Figure IV.15: *Répartition de pression autour du profile l'aile à z = -3,5 m.*

On note que dans les courbes précédentes, le cas *AWDS* est préférable à un autre car il apporte plus de portance. Le cas *ASW* est, pa contre, le cas le plus défavorable puisqu'il présente le minimum de portance par rapport aux autres.

IV.4.2- Lignes de courants

Une ligne de courant est une courbe de l'espace décrivant un fluide en mouvement et qui, à tout instant, possède en tout point une tangente parallèle à la vitesse des particules du fluide.

Les figures ci-dessous (Fig.IV.16-20) nous montrent les lignes de courant autour de l'aile et des winglets pour les 5 configurations.

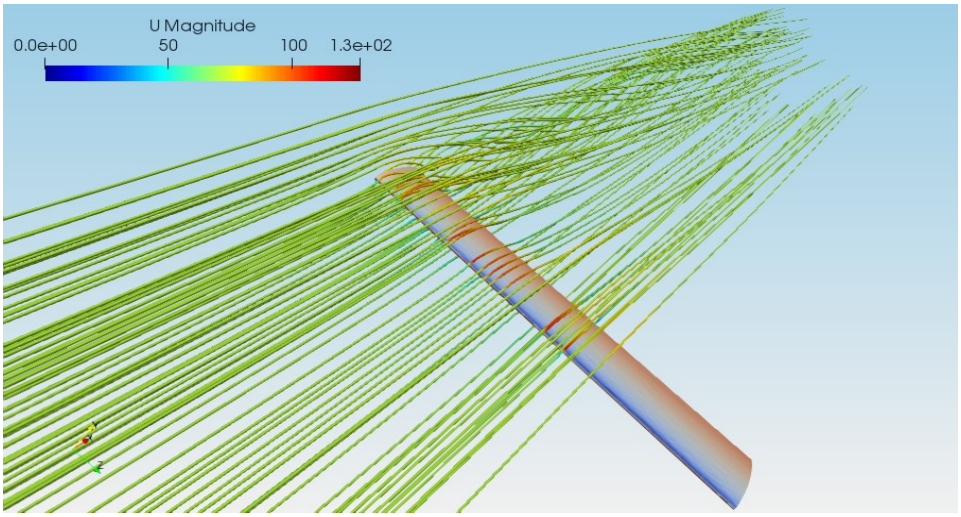


Figure IV.16: *Lignes de courant. Cas ASW.*

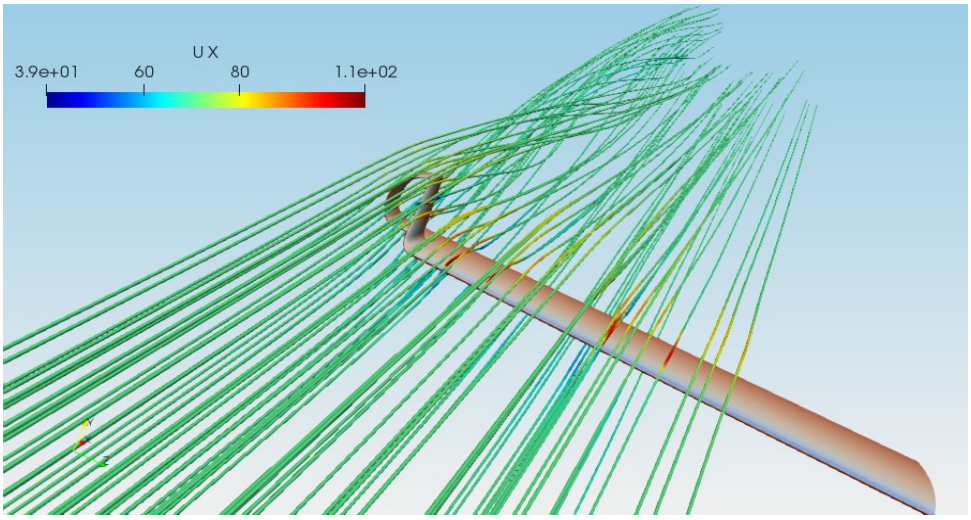


Figure IV.17: *Lignes de courant. Cas AWDC.*

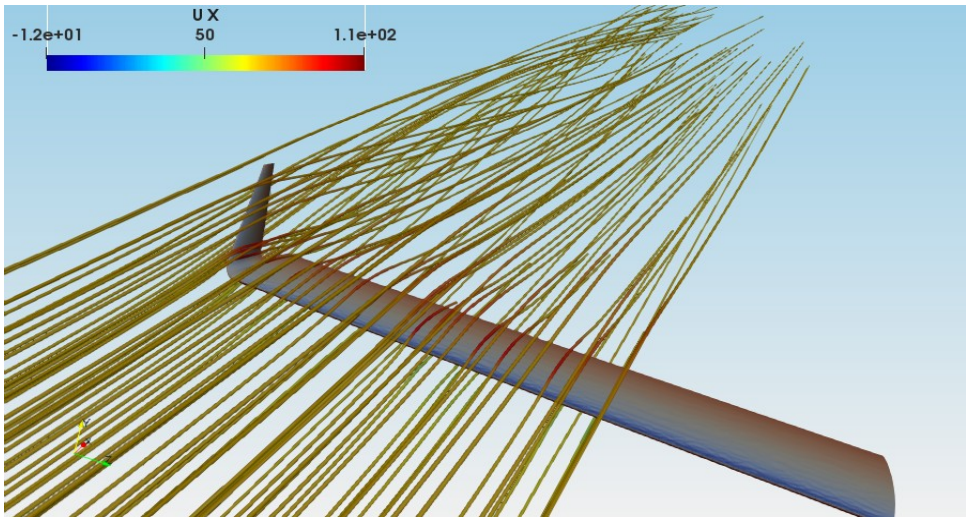


Figure IV.18: *Lignes de courant. Cas AWW.*

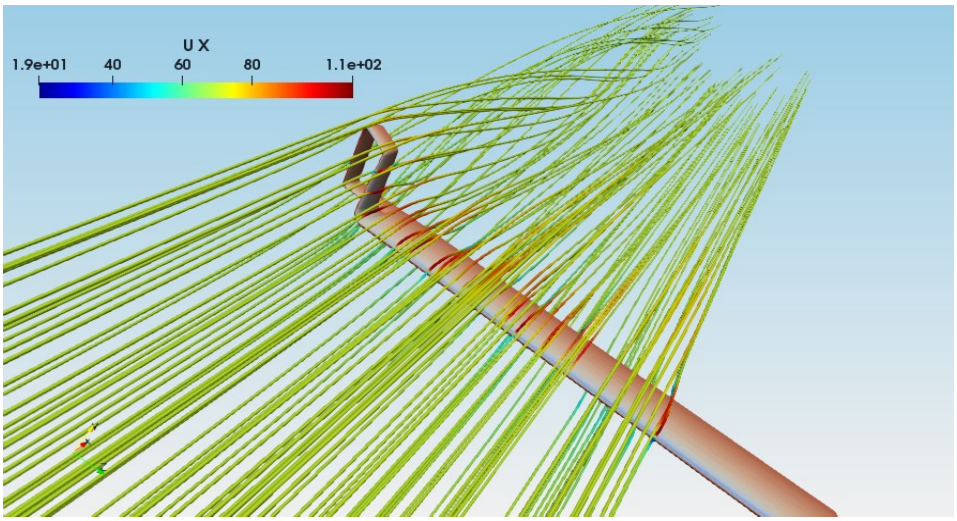


Figure IV.19: *Lignes de courant. Cas AWS.*

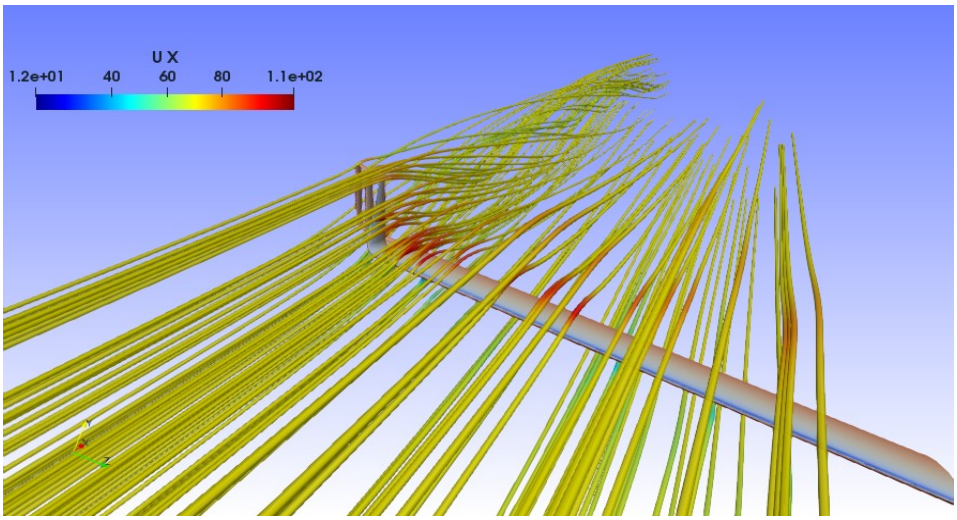


Figure IV.20: *Lignes de courant. Cas AWDS.*

Nous remarquons, à partir de ces lignes de courant autour de la winglet, que le cas *AWW* est celui qui présente le moins de perturbations derrière cette dernière. Le cas le plus défavorable étant, bien sûr, celui sans winglet *ASW*.

Pour le cas *AWDS*, nous ne pouvons affirmer, à partir des lignes de courant, s'il est meilleur ou non que le cas *AWS*.

IV.4.3- Energie cinétique turbulente

Les figures (Fig.IV.21-25), présentent la variation de l'énergie cinétique turbulent derrière la winglet pour chaque configuration.

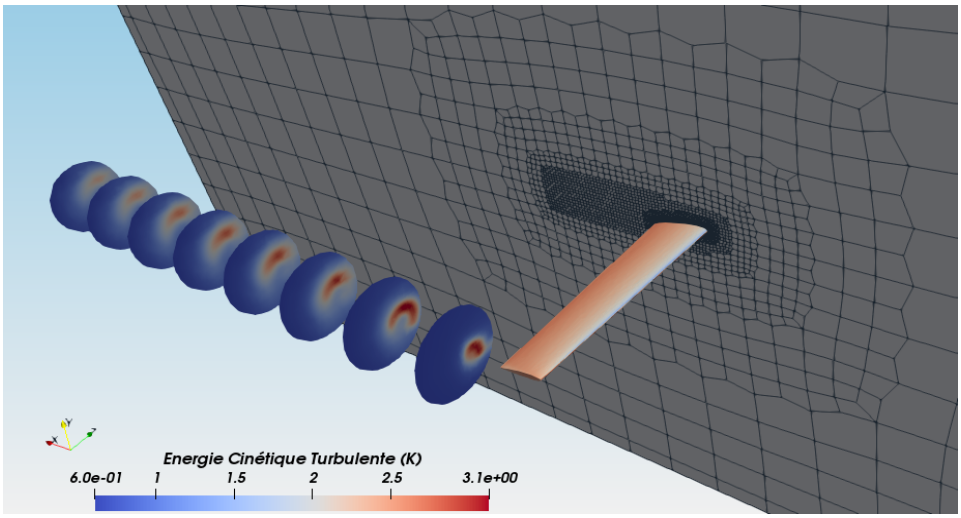


Figure IV.21: *Energie cinétique turbulente. Cas ASW.*

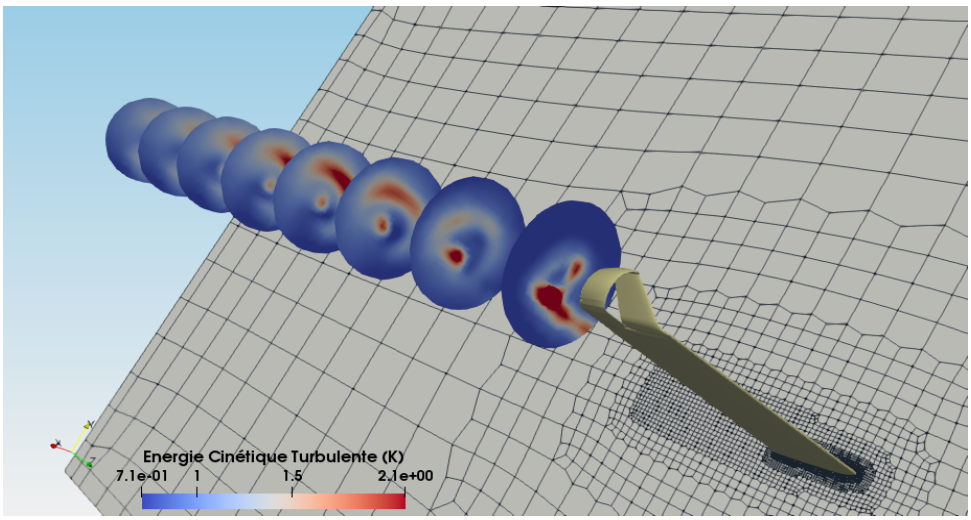


Figure IV.22: *Energie cinétique turbulente. Cas AWDC.*

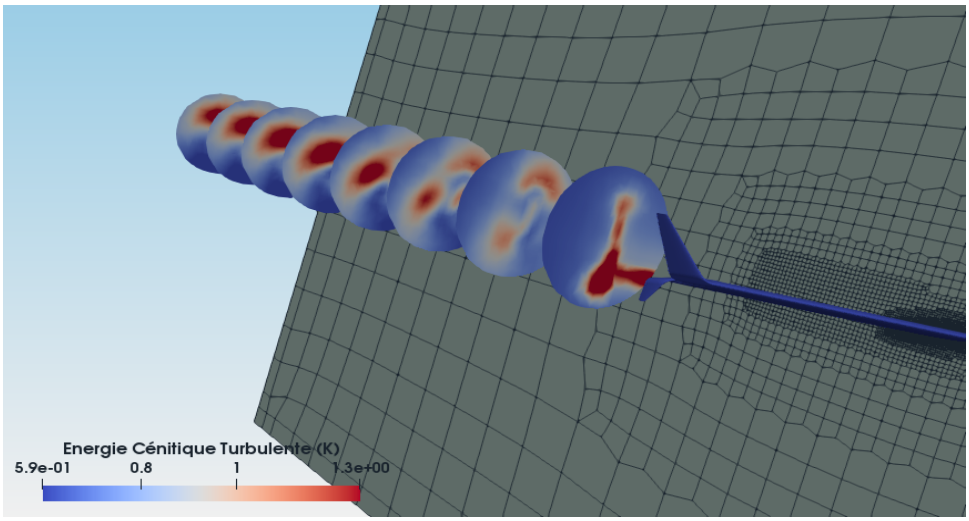


Figure IV.23: *Energie cinétique turbulente. Cas AWW.*

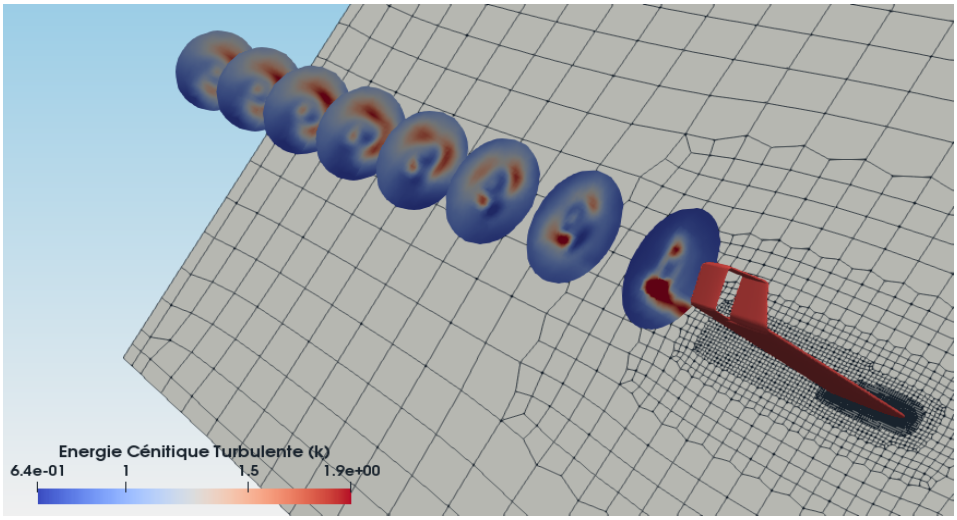


Figure IV.24: *Energie cinétique turbulente. Cas AWS.*

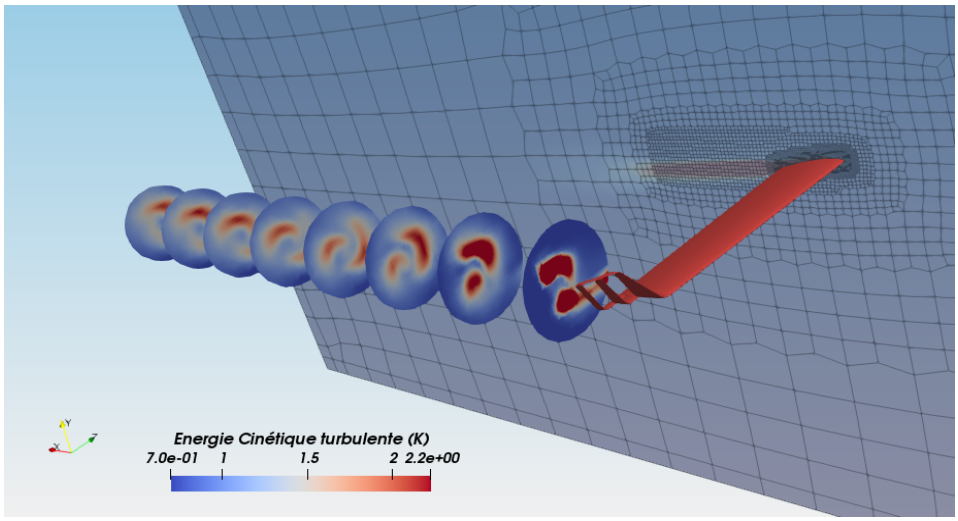


Figure IV.25: *Energie cinétique turbulente. Cas AWDS.*

IV.4.4- Vorticité

Les figures (Fig.IV.26-30), présentent la variation de l'énergie cinétique turbulent derrière la winglet pour chaque configuration.

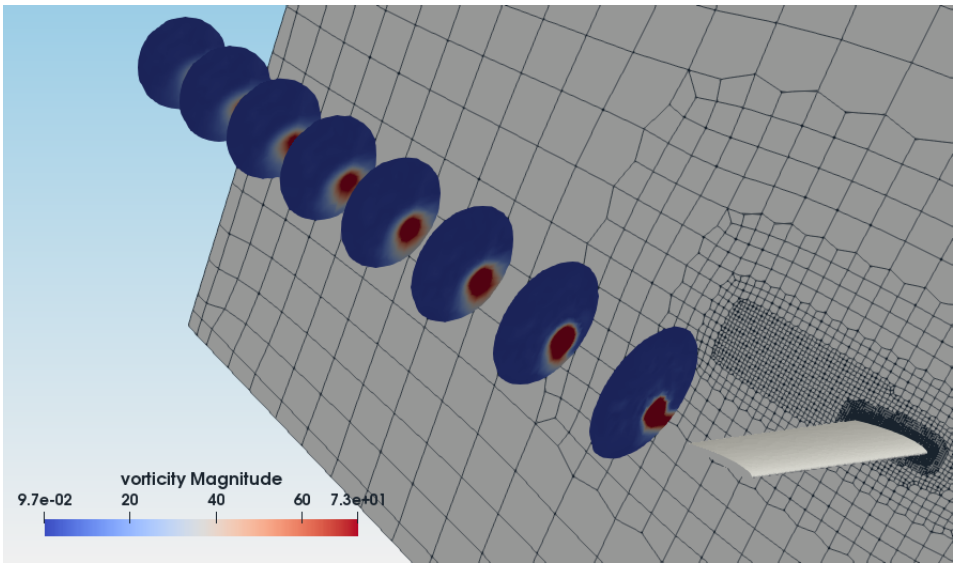


Figure IV.26: *Vorticité derrière la winglet. Cas ASW.*

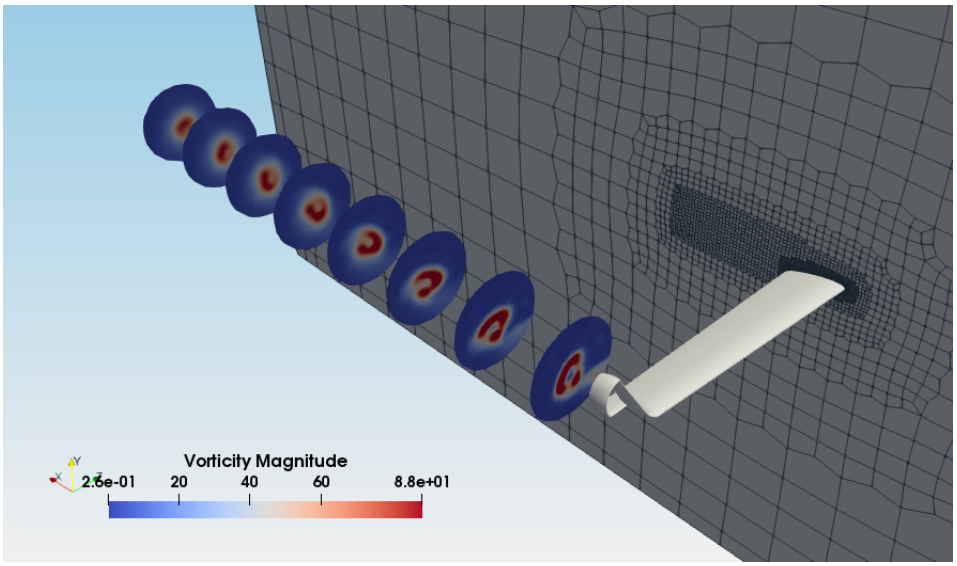


Figure IV.27: *Vorticité derrière la winglet. Cas AWDC.*

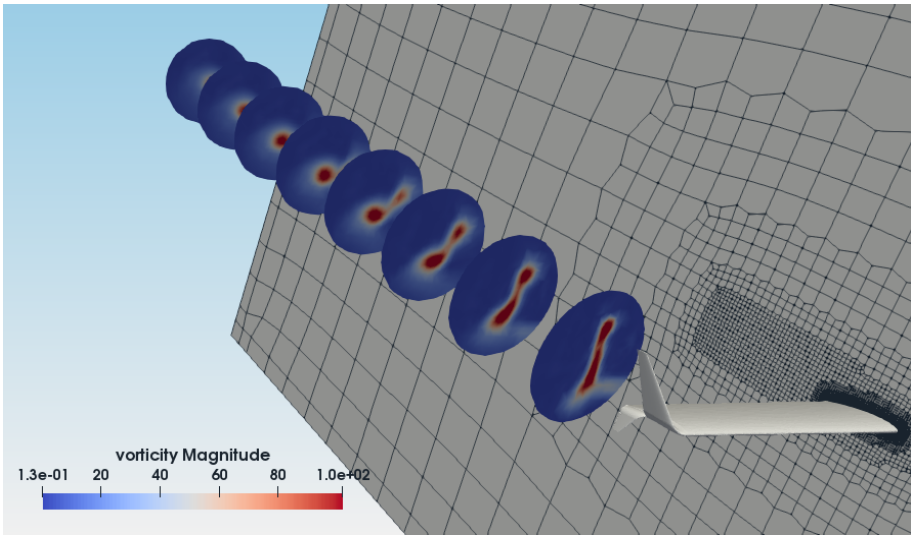


***Figure IV.28**: Vorticité derrière la winglet. Cas AWW.*

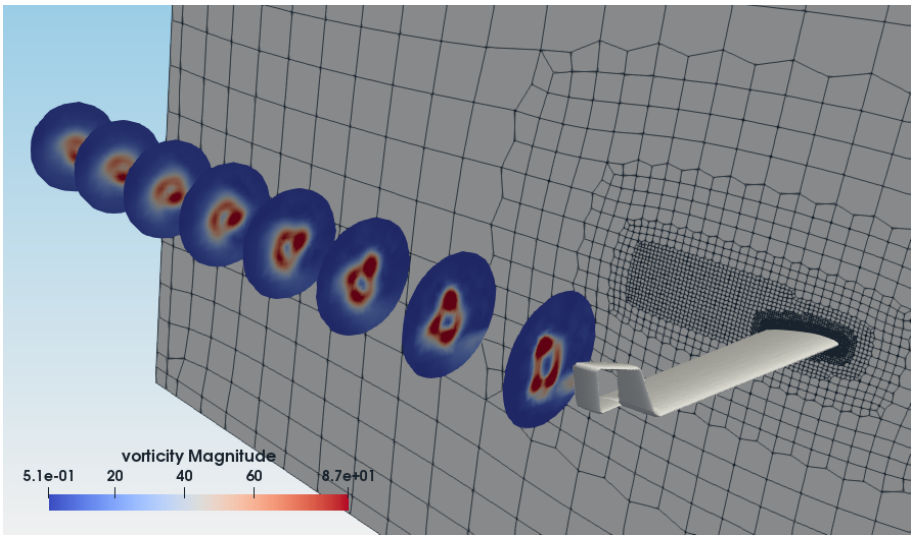


***Figure IV.29**: Vorticité derrière la winglet. Cas AWS.*

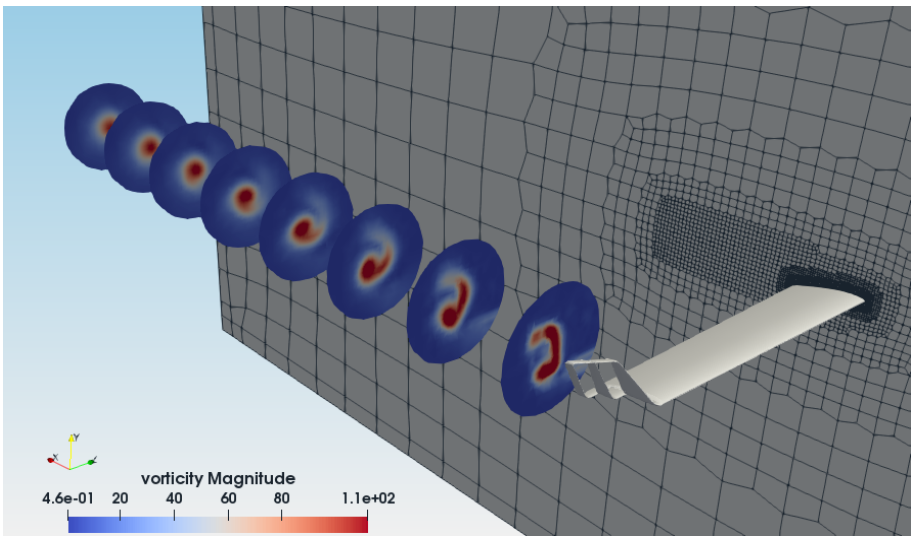


***Figure IV.30**: Vorticité derrière la winglet. Cas AWDS.*

Nous remarquons, que les configurations avec winglets génèrent des tourbillons de plus faibles intensités que celui dans le cas sans winglet. Ceci est bien visible puisque nous avons utilisé la même échelle pour les cinq cas pour pouvoir les comparer. On peut alors conclure que l'écoulement derrière l'avion muni de winglets sera moins perturbé que celui sans winglet puisque les tourbillons n'ont pas assez d'énergie pour subsister. En effet, ceci permettra de minimiser le temps de décollage entre deux avions successifs.

IV.4.5- Q-Criterion

Le Q-Criterion est un nouveau critère de visualisation des écoulements turbulents, il définit la vorticité en tant qu'une même région de fluide avec un second invariant positif de ∇u. Ce critère rajoute aussi une seconde condition sur la pression, limitant cette dernière à être inférieure à la pression ambiante dans le vortex. Ce critère représente donc un équilibre entre le taux de déformation et l'intensité des tourbillons définissant ainsi les tourbillons comme des surfaces dont l'intensité est supérieure à celle des taux de déformations. En utilisant la même échelle pour les différentes configurations, nous aurons les figures suivantes (IV.31-35):

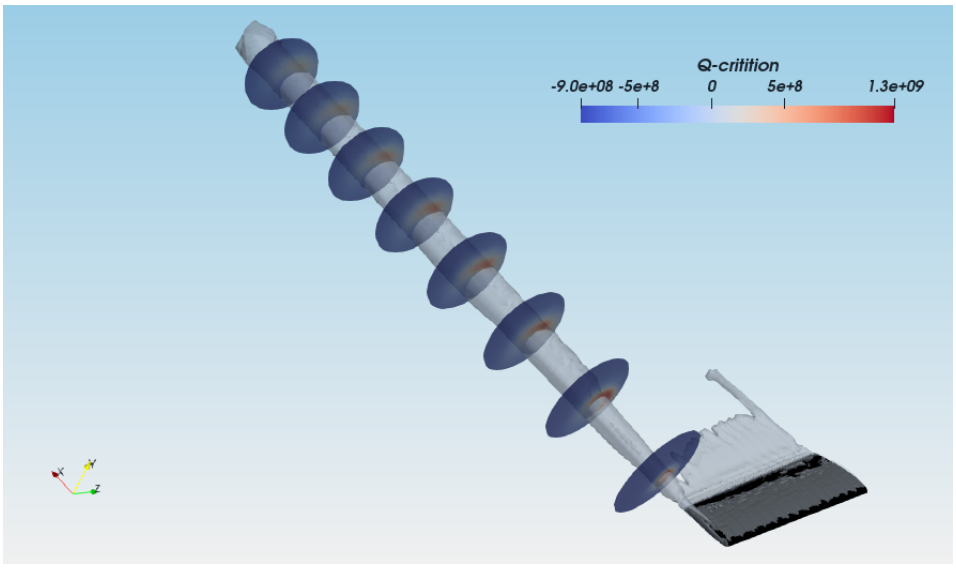


Figure IV.31: *Q-Criterion. Cas ASW.*

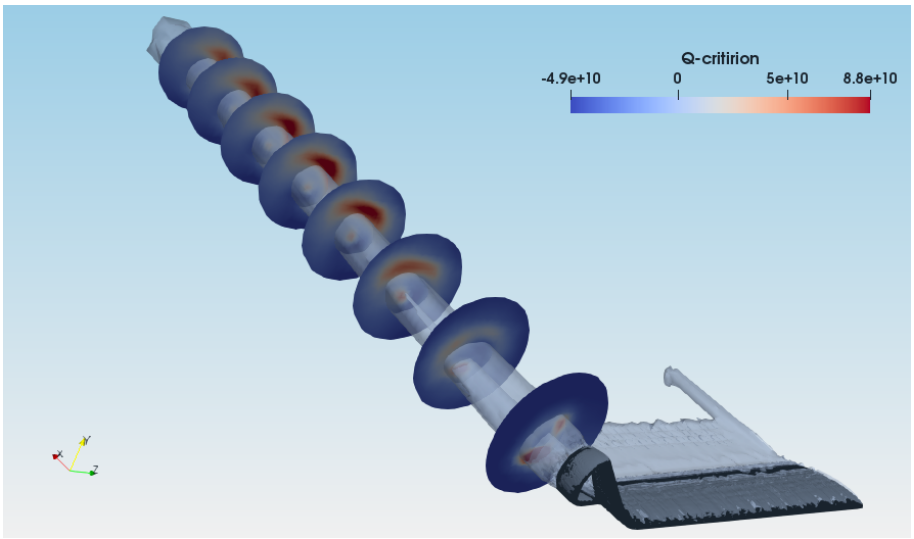


Figure IV.32: *Q-Criterion. Cas AWDC.*

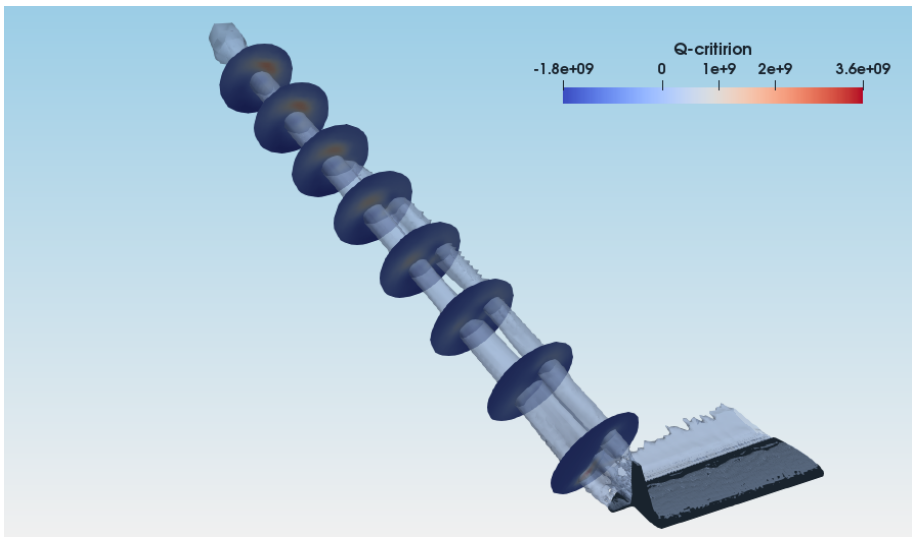


Figure IV.33: *Q-Criterion. Cas AWW.*

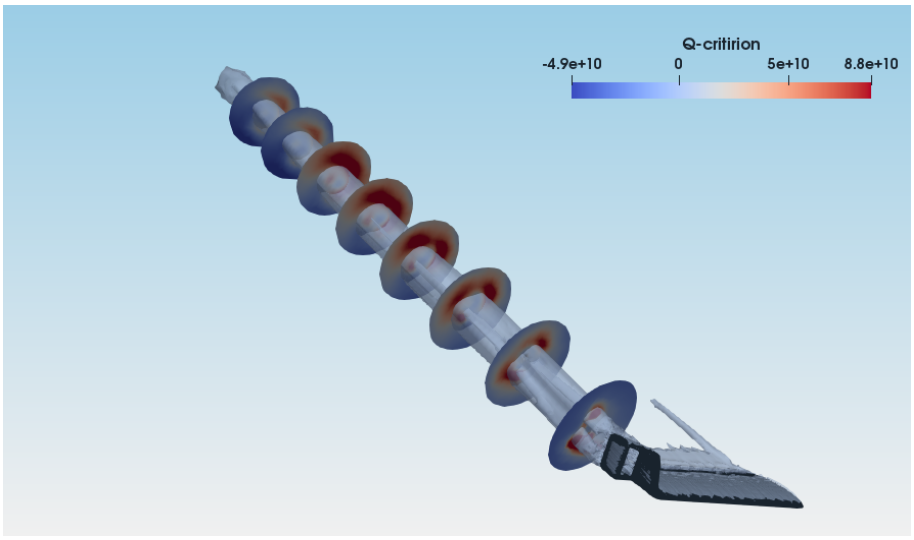


Figure IV.34: *Q-Criterion. Cas AWS.*

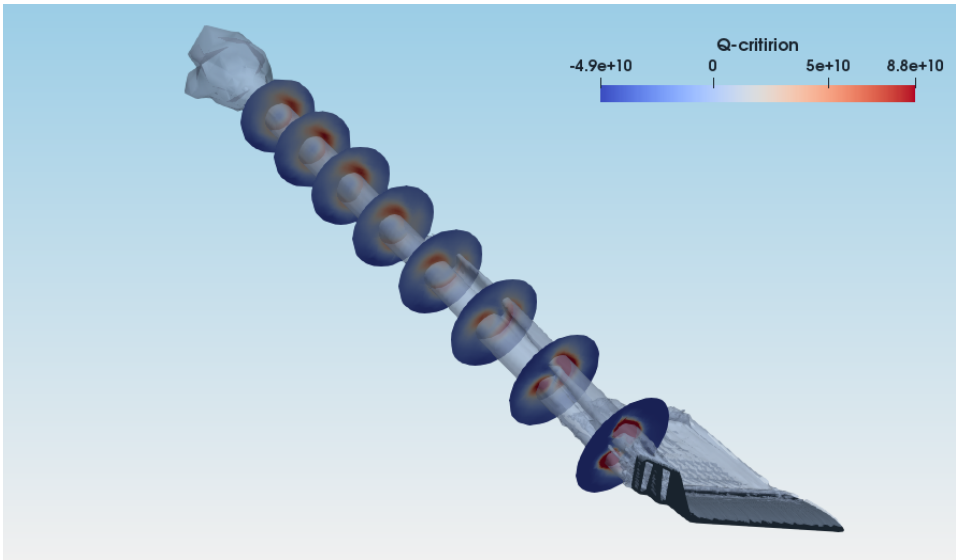


***Figure IV.35**: Q-Criterion. Cas AWDS.*

IV.5- Conclusion

Vu que nous n'avons pas utilisé les mêmes qualités de maillages que ceux de la référence [1], nous ne pouvons alors comparer nos résultats avec. Par contre, en fonction des résultats de nos simulations, il est claire que le cas *AWS* est meilleur que celui avec double spiroid *AWDS* puisque ce dernier entraîne une plus grande traînée en plus des perturbations plus accentuées derrière la winglet.

Conclusion générale

Conclusion générale

Dans cette étude, nous avons d'abord compris les difficultés à simuler une géométrie complexe dans un domaine d'étude assez grand. Nous avons bien assimilé les étapes à suivre pour mener à bien une simulation numérique.Il faut citer que nous n'avons pas réalisé facilement toutes ces conceptions et maillages. La difficulté était l'utilisation de logiciel FreeCAD en Mars 2019 était la première fois que nous utilisons. Nous avons fait de petits TP sur les programmes pendant 20 jours pour apprendre certaines choses qui nous ont aidés à commencer notre expérience grâce à notre encadreur Dr:Messaoudi.L et nos efforts avec lui.

Les maillages obtenus sont qualifié de « bon maillage ». Nous sommes ensuite passés à la simulation de 5 configurations dans le cas du décollage de l'avion avec le logiciel FreeCAD .Le modèle de turbulence que nous avons utilisé est celui de KomegaSST le seul modèle dans lequel le programme fonctionne. Les résultats obtenus ont été validés partiellement avec l'article que nous avons pris comme référence[1].

Les résultats que nous avons trouvés nous confirment l'importance de l'utilisation des winglets. En effet, la présence de cette dernière en bout d'aile modifie la structure de l'écoulement en transportant les tourbillons vers l'extrémité de la winglet puis contribue à la portance de l'aile et diminue la traînée.

Nous recommandons d'utiliser des moyens robustes pour pouvoir examiner en détail la structure des tourbillons autour de l'aile et derrière la winglet. Aussi, la validation expérimentale est à préconiser dans ce cas en utilisant des modèles réduits dans la soufflerie du laboratoire d'aérodynamique.

Références

Références

[1] ZEKKOUR.Ahmed « Influence des différentes formes de winglets sur l'écoulement autour d'une aile d'avion » 2016

[2] Aérodynamique et biomimétique, la cigogne et winglets , « http: //aerodynamique.e-monsite.com/pages/le-biomimetisme-dans-l'aerodynamisme/ » 2015.

[3] A. B. a. Joel E. Guerrero a, Dario Maestro a, « biomimetic spiroid winglets for lift and drag control » 2011.

[4] « L'aérodynamique », Wikipedia. 2008

[5] Vioux daniel «https://www.lavionnaire.fr/AerodynPortance.php»,2010

[6] Mr Riadh Ben Hamouda « Dynamique des fluides incompressibles réels »2015

[7] D. G. Ouahiba, « Les techniques cfd» 2011.

[8] Sean Thielen,« New Generation of CFD Software Positions France's Aerospace Lab for Aerodynamics Breakthroughs »,2018

[9] « DOCUMENTATION https://www.freecadweb.org/ » 2019

[10] Join GitHub today « CfdOF: A Computational fluid dynamics (CFD) workbench for FreeCAD »2019

[11] MESBAHI Amine « Free CFD Workflow » ,2019

[12] BEN ABBES Amira,SOUBSOL David,HASSI Tarik « Validation du solveur open-source OpenFOAM pour la résolution de problèmes thermiques »2012/2013 ENSEEIHT, Département Hydraulique et Mécanique des fluides

[13] Caroline Bligny « ParaView – Démo/TP » février 2019

Annexes

A1- Macro pour simulation de configuration AWDS

```
==================================
# La geometrie
==================================

import FreeCAD
import Part

FreeCAD.open(u"/home/abdoun/PFE-Aero/22/DoubleSpiroid.FCStd")
App.ActiveDocument.addObject("Part::Box","Box")
App.ActiveDocument.ActiveObject.Label = "Cube"
App.ActiveDocument.recompute()
FreeCAD.getDocument("DoubleSpiroid").getObject("Box").Length = '23000 mm'
FreeCAD.getDocument("DoubleSpiroid").getObject("Box").Width = '14000 mm'
FreeCAD.getDocument("DoubleSpiroid").getObject("Box").Height = '12000 mm'
FreeCAD.getDocument("DoubleSpiroid").getObject("Box").Placement =
App.Placement(App.Vector(-7000,-7000,-12000),App.Rotation(App.Vector(0,0,1),0))
FreeCAD.getDocument("DoubleSpiroid").getObject("Part__Feature008").Placement =
App.Placement(App.Vector(0,0,0),App.Rotation(App.Vector(0,0,1),-8))
App.activeDocument().addObject("Part::Cut","Cut")
App.activeDocument().Cut.Base = App.activeDocument().Box
App.activeDocument().Cut.Tool = App.activeDocument().Part__Feature008
Gui.activeDocument().Box.Visibility=False
Gui.activeDocument().Part__Feature008.Visibility=False
Gui.ActiveDocument.Cut.ShapeColor=Gui.ActiveDocument.Box.ShapeColor
Gui.ActiveDocument.Cut.DisplayMode=Gui.ActiveDocument.Box.DisplayMode
App.ActiveDocument.recompute()
u"/home/abdoun/PFEAero/22/DoubleSpiroid.FCStd""Part::Box","Box"

==================================
# L'analyse
==================================

import CfdAnalysis
analysis = CfdAnalysis.makeCfdAnalysis('CfdAnalysis')
import CfdPhysicsSelection
analysis.addObject(CfdPhysicsSelection.makeCfdPhysicsSelection())
import CfdFluidMaterial
analysis.addObject(CfdFluidMaterial.makeCfdFluidMaterial('FluidProperties'))
import CfdInitialiseFlowField
analysis.addObject(CfdInitialiseFlowField.makeCfdInitialFlowField())
import CfdSolverFoam
analysis.addObject(CfdSolverFoam.makeCfdSolverFoam())

## Modele physique

Gui.activeDocument().setEdit('PhysicsModel')
obj = FreeCAD.ActiveDocument.PhysicsModel
obj.Phase = 'Single'
obj.Flow = 'Incompressible'
```

```
obj.Thermal = 'None'
obj.Turbulence = 'RANS'
obj.TurbulenceModel = 'kOmegaSST'
obj.gx = '0 mm/s^2'
obj.gy = '-9,8e+03 mm/s^2'
obj.gz = '0 mm/s^2'

## Les proprietes de fluide

Gui.activeDocument().setEdit('FluidProperties')
mat = FreeCAD.ActiveDocument.FluidProperties
mat.Density = '1,2 kg/m^3'
mat.DynamicViscosity = '1,8e-05 kg/(m*s)'

## Initialiser les champs

Gui.activeDocument().setEdit('InitialiseFields')
init = FreeCAD.ActiveDocument.InitialiseFields.InitialVariables
init['PotentialFoam'] = True
init['UseInletUPValues'] = False
init['Ux'] = 0.0
init['Uy'] = 0.0
init['Uz'] = 0.0
init['Pressure'] = 0.0
init['alphas'] = {}
init['UseInletTemperatureValues'] = False
init['Temperature'] = 290.0
init['UseInletTurbulenceValues'] = False
init['omega'] = 0.994837673637
init['k'] = 0.01
init['Inlet'] = ' '
FreeCAD.ActiveDocument.InitialiseFields.InitialVariables = init

================================
# Maillage
================================

import CfdMesh

# Maillage de domaine

CfdMesh.makeCfdMesh('Cut_Mesh')
App.ActiveDocument.ActiveObject.Part = App.ActiveDocument.Cut
FreeCAD.ActiveDocument.Cut_Mesh.CharacteristicLengthMax = '1000 mm'
FreeCAD.ActiveDocument.Cut_Mesh.MeshUtility = 'cfMesh'
FreeCAD.ActiveDocument.Cut_Mesh.ElementDimension = '3D'
FreeCAD.ActiveDocument.Cut_Mesh.CellsBetweenLevels = 3
FreeCAD.ActiveDocument.Cut_Mesh.EdgeRefinement = 0
FreeCAD.ActiveDocument.Cut_Mesh.PointInMesh = {'y': 0.0, 'x': 0.0, 'z': 0.0}
```

```
# Raffinement du maillage

import CfdMeshRegion

# Winglet

CfdMeshRegion.makeCfdMeshRegion(App.ActiveDocument.Cut_Mesh)
referenceList=[]
FreeCAD.ActiveDocument.MeshRegion.References = referenceList
referenceList.append(('Part__Feature008','Face3'))
referenceList.append(('Part__Feature008','Face4'))
referenceList.append(('Part__Feature008','Face5'))
referenceList.append(('Part__Feature008','Face6'))
referenceList.append(('Part__Feature008','Face7'))
referenceList.append(('Part__Feature008','Face8'))
referenceList.append(('Part__Feature008','Face9'))
referenceList.append(('Part__Feature008','Face10'))
referenceList.append(('Part__Feature008','Face11'))
referenceList.append(('Part__Feature008','Face12'))
referenceList.append(('Part__Feature008','Face13'))
referenceList.append(('Part__Feature008','Face14'))
referenceList.append(('Part__Feature008','Face15'))
referenceList.append(('Part__Feature008','Face16'))
referenceList.append(('Part__Feature008','Face17'))
referenceList.append(('Part__Feature008','Face18'))
FreeCAD.ActiveDocument.MeshRegion.RelativeLength = 0.008
FreeCAD.ActiveDocument.MeshRegion.RefinementThickness = '40 mm'
FreeCAD.ActiveDocument.MeshRegion.NumberLayers = 15
FreeCAD.ActiveDocument.MeshRegion.ExpansionRatio = 1.1
FreeCAD.ActiveDocument.MeshRegion.FirstLayerHeight = '4.8 mm'
FreeCAD.ActiveDocument.MeshRegion.RefinementLevel = 1
FreeCAD.ActiveDocument.MeshRegion.RegionEdgeRefinement = 1
FreeCAD.ActiveDocument.MeshRegion.Baffle = False
FreeCAD.ActiveDocument.MeshRegion.Internal = False
FreeCAD.ActiveDocument.MeshRegion.InternalRegion = {'Radius1': 0.001, 'Point1': {'y': 0, 'x':
0.0, 'z': 0}, 'Point2': {'y': 0, 'x': 0.0, 'z': 0}, 'Radius2': 0.001, 'Center': {'y': 0, 'x': 0.0, 'z': 0},
'BoxLengths': {'y': 0.001, 'x': 0.001, 'z': 0.001}, 'Type': 'Box', 'SphereRadius': 0.001}

# Aile

CfdMeshRegion.makeCfdMeshRegion(App.ActiveDocument.Cut_Mesh)
referenceList = []
referenceList.append(('Part__Feature008','Face1'))
FreeCAD.ActiveDocument.MeshRegion001.References = referenceList
FreeCAD.ActiveDocument.MeshRegion001.RelativeLength = 0.015
FreeCAD.ActiveDocument.MeshRegion001.RefinementThickness = '40 mm'
FreeCAD.ActiveDocument.MeshRegion001.NumberLayers = 15
FreeCAD.ActiveDocument.MeshRegion001.ExpansionRatio = 1.1
FreeCAD.ActiveDocument.MeshRegion001.FirstLayerHeight = '4.8 mm'
FreeCAD.ActiveDocument.MeshRegion001.RefinementLevel = 1
```

```
FreeCAD.ActiveDocument.MeshRegion001.RegionEdgeRefinement = 1
FreeCAD.ActiveDocument.MeshRegion001.Baffle = False
FreeCAD.ActiveDocument.MeshRegion001.Internal = False
FreeCAD.ActiveDocument.MeshRegion001.InternalRegion = {'Radius1': 0.001, 'Point1': {'y': 0, 'x': 0.0, 'z': 0}, 'Point2': {'y': 0, 'x': 0.0, 'z': 0}, 'Radius2': 0.001, 'Center': {'y': 0, 'x': 0.0, 'z': 0}, 'BoxLengths': {'y': 0.001, 'x': 0.001, 'z': 0.001}, 'Type': 'Box', 'SphereRadius': 0.001}

# Box

CfdMeshRegion.makeCfdMeshRegion(App.ActiveDocument.Cut_Mesh)
referenceList = [ ]
FreeCAD.ActiveDocument.MeshRegion002.References = referenceList
FreeCAD.ActiveDocument.MeshRegion002.RelativeLength = 0.031
FreeCAD.ActiveDocument.MeshRegion002.RefinementThickness = '0 mm'
FreeCAD.ActiveDocument.MeshRegion002.NumberLayers = 0
FreeCAD.ActiveDocument.MeshRegion002.ExpansionRatio = 0.0
FreeCAD.ActiveDocument.MeshRegion002.FirstLayerHeight = '0 mm'
FreeCAD.ActiveDocument.MeshRegion002.RefinementLevel = 1
FreeCAD.ActiveDocument.MeshRegion002.RegionEdgeRefinement = 1
FreeCAD.ActiveDocument.MeshRegion002.Baffle = False
FreeCAD.ActiveDocument.MeshRegion002.Internal = True
FreeCAD.ActiveDocument.MeshRegion002.InternalRegion = {'Radius1': 0.001, 'Point1': {'y': 0, 'x': 0.0, 'z': 0}, 'Point2': {'y': 0, 'x': 0.0, 'z': 0}, 'Radius2': 0.001, 'Center': {'y': 0, 'x': 2.05, 'z': -1.9}, 'BoxLengths': {'y': 0.5, 'x': 2.0, 'z': 3.8}, 'Type': 'Box', 'SphereRadius': 0.001}

# Cone

CfdMeshRegion.makeCfdMeshRegion(App.ActiveDocument.Cut_Mesh)
FreeCAD.ActiveDocument.MeshRegion003.References = referenceList
FreeCAD.ActiveDocument.MeshRegion003.RelativeLength = 0.031
FreeCAD.ActiveDocument.MeshRegion003.RefinementThickness = '0 mm'
FreeCAD.ActiveDocument.MeshRegion003.NumberLayers = 0
FreeCAD.ActiveDocument.MeshRegion003.ExpansionRatio = 0.0
FreeCAD.ActiveDocument.MeshRegion003.FirstLayerHeight = '0 mm'
FreeCAD.ActiveDocument.MeshRegion003.RefinementLevel = 1
FreeCAD.ActiveDocument.MeshRegion003.RegionEdgeRefinement = 1
FreeCAD.ActiveDocument.MeshRegion003.Baffle = False
FreeCAD.ActiveDocument.MeshRegion003.Internal = True
FreeCAD.ActiveDocument.MeshRegion003.InternalRegion = {'Radius1': 0.5, 'Point1': {'y': 0.2, 'x': 1.0, 'z': -4.3}, 'Point2': {'y': 0.2, 'x': 13.0, 'z': -4.3}, 'Radius2': 0.4, 'Center': {'y': 0, 'x': 0.0, 'z': 0}, 'BoxLengths': {'y': 0.001, 'x': 0.001, 'z': 0.001}, 'Type': u'Cone', 'SphereRadius': 0.001}


==================================
# Les conditions aux limit
==================================

import CfdFluidBoundary

# inlet ====> l'entrie
```

```
FemGui.getActiveAnalysis().addObject(CfdFluidBoundary.makeCfdFluidBoundary())
bc = FreeCAD.ActiveDocument.CfdFluidBoundary.BoundarySettings
bc['BoundaryType'] = 'inlet'
bc['BoundarySubtype'] = 'uniformVelocity'
bc['ThermalBoundaryType'] = 'zeroGradient'
bc['VelocityIsCartesian'] = True
bc['Ux'] = 70.0
bc['Uy'] = 0.0
bc['Uz'] = 0.0
bc['VelocityMag'] = 0.0
bc['DirectionFace'] = ' '
bc['ReverseNormal'] = True
bc['MassFlowRate'] = 0.0
bc['VolFlowRate'] = 0.0
bc['Pressure'] = 0.0
bc['SlipRatio'] = 0.0
bc['Temperature'] = 290.0
bc['HeatFlux'] = 0.0
bc['HeatTransferCoeff'] = 0.0
bc['TurbulenceInletSpecification'] = 'intensityAndLengthScale'
bc['TurbulentKineticEnergy'] = 0.01
bc['SpecificDissipationRate'] = 0.994837673637
bc['TurbulenceIntensity'] = 0.1
bc['TurbulenceLengthScale'] = 0.1
bc['PressureDropCoeff'] = 0.0
bc['ScreenWireDiameter'] = 0.0001
bc['ScreenSpacing'] = 0.0
bc['PorousBaffleMethod'] = 0
FreeCAD.ActiveDocument.CfdFluidBoundary.BoundarySettings = bc
FreeCAD.ActiveDocument.CfdFluidBoundary.Label = 'inlet'
FreeCAD.ActiveDocument.CfdFluidBoundary.References = []
FreeCAD.ActiveDocument.CfdFluidBoundary.References.append(('Box', 'Face1'))
FreeCAD.ActiveDocument.recompute()

# outlet ====> la sortie

FemGui.getActiveAnalysis().addObject(CfdFluidBoundary.makeCfdFluidBoundary())
bc = FreeCAD.ActiveDocument.CfdFluidBoundary001.BoundarySettings
bc['BoundaryType'] = 'outlet'
bc['BoundarySubtype'] = 'staticPressure'
bc['ThermalBoundaryType'] = 'zeroGradient'
bc['VelocityIsCartesian'] = True
bc['Ux'] = 0.0
bc['Uy'] = 0.0
bc['Uz'] = 0.0
bc['VelocityMag'] = 0.0
bc['DirectionFace'] = ' '
bc['ReverseNormal'] = False
bc['MassFlowRate'] = 0.0
```

```
bc['VolFlowRate'] = 0.0
bc['Pressure'] = 0.0
bc['SlipRatio'] = 0.0
bc['Temperature'] = 290.0
bc['HeatFlux'] = 0.0
bc['HeatTransferCoeff'] = 0.0
bc['TurbulenceInletSpecification'] = 'intensityAndLengthScale'
bc['TurbulentKineticEnergy'] = 0.01
bc['SpecificDissipationRate'] = 0.994837673637
bc['TurbulenceIntensity'] = 0.1
bc['TurbulenceLengthScale'] = 0.1
bc['PressureDropCoeff'] = 0.0
bc['ScreenWireDiameter'] = 0.0001
bc['ScreenSpacing'] = 0.0
bc['PorousBaffleMethod'] = 0
FreeCAD.ActiveDocument.CfdFluidBoundary001.BoundarySettings = bc
FreeCAD.ActiveDocument.CfdFluidBoundary001.Label = 'outlet'
FreeCAD.ActiveDocument.CfdFluidBoundary001.References = []
FreeCAD.ActiveDocument.CfdFluidBoundary001.References.append(('Box', 'Face2'))
FreeCAD.ActiveDocument.recompute()

# symmetry  ====> contrainte

FemGui.getActiveAnalysis().addObject(CfdFluidBoundary.makeCfdFluidBoundary())
bc = FreeCAD.ActiveDocument.CfdFluidBoundary002.BoundarySettings
bc['BoundaryType'] = 'constraint'
bc['BoundarySubtype'] = 'symmetry'
bc['ThermalBoundaryType'] = 'zeroGradient'
bc['VelocityIsCartesian'] = True
bc['Ux'] = 0.0
bc['Uy'] = 0.0
bc['Uz'] = 0.0
bc['VelocityMag'] = 0.0
bc['DirectionFace'] = ' '
bc['ReverseNormal'] = False
bc['MassFlowRate'] = 0.0
bc['VolFlowRate'] = 0.0
bc['Pressure'] = 0.0
bc['SlipRatio'] = 0.0
bc['Temperature'] = 290.0
bc['HeatFlux'] = 0.0
bc['HeatTransferCoeff'] = 0.0
bc['TurbulenceInletSpecification'] = 'intensityAndLengthScale'
bc['TurbulentKineticEnergy'] = 0.01
bc['SpecificDissipationRate'] = 0.994837673637
bc['TurbulenceIntensity'] = 0.1
bc['TurbulenceLengthScale'] = 0.1
bc['PressureDropCoeff'] = 0.0
bc['ScreenWireDiameter'] = 0.0001
bc['ScreenSpacing'] = 0.0
```

```
bc['PorousBaffleMethod'] = 0
FreeCAD.ActiveDocument.CfdFluidBoundary002.BoundarySettings = bc
FreeCAD.ActiveDocument.CfdFluidBoundary002.Label = 'constraint'
FreeCAD.ActiveDocument.CfdFluidBoundary002.References = []
FreeCAD.ActiveDocument.CfdFluidBoundary002.References.append(('Box', 'Face6'))
FreeCAD.ActiveDocument.recompute()

# wall slip ====> murs

FemGui.getActiveAnalysis().addObject(CfdFluidBoundary.makeCfdFluidBoundary())
bc = FreeCAD.ActiveDocument.CfdFluidBoundary003.BoundarySettings
bc['BoundaryType'] = 'wall'
bc['BoundarySubtype'] = 'slip'
bc['ThermalBoundaryType'] = 'zeroGradient'
bc['VelocityIsCartesian'] = True
bc['Ux'] = 0.0
bc['Uy'] = 0.0
bc['Uz'] = 0.0
bc['VelocityMag'] = 0.0
bc['DirectionFace'] = ' '
bc['ReverseNormal'] = False
bc['MassFlowRate'] = 0.0
bc['VolFlowRate'] = 0.0
bc['Pressure'] = 0.0
bc['SlipRatio'] = 0.0
bc['Temperature'] = 290.0
bc['HeatFlux'] = 0.0
bc['HeatTransferCoeff'] = 0.0
bc['TurbulenceInletSpecification'] = 'intensityAndLengthScale'
bc['TurbulentKineticEnergy'] = 0.01
bc['SpecificDissipationRate'] = 0.994837673637
bc['TurbulenceIntensity'] = 0.1
bc['TurbulenceLengthScale'] = 0.1
bc['PressureDropCoeff'] = 0.0
bc['ScreenWireDiameter'] = 0.0001
bc['ScreenSpacing'] = 0.0
bc['PorousBaffleMethod'] = 0
FreeCAD.ActiveDocument.CfdFluidBoundary003.BoundarySettings = bc
FreeCAD.ActiveDocument.CfdFluidBoundary003.Label = 'wall'
FreeCAD.ActiveDocument.CfdFluidBoundary003.References = []
FreeCAD.ActiveDocument.CfdFluidBoundary003.References.append(('Box', 'Face3'))
FreeCAD.ActiveDocument.CfdFluidBoundary003.References.append(('Box', 'Face4'))
FreeCAD.ActiveDocument.CfdFluidBoundary003.References.append(('Box', 'Face5'))
FreeCAD.ActiveDocument.recompute()

# end
```

A2- Tutoriel puor l'itulisation de cfd of :

Après avoir converti la format .STL de conseption de l'aile à un format .FCSTD , nous avons importer cette nouvelle dernière à FreeCAD ,nous avons incliné l'aile à 8° en position de décollage

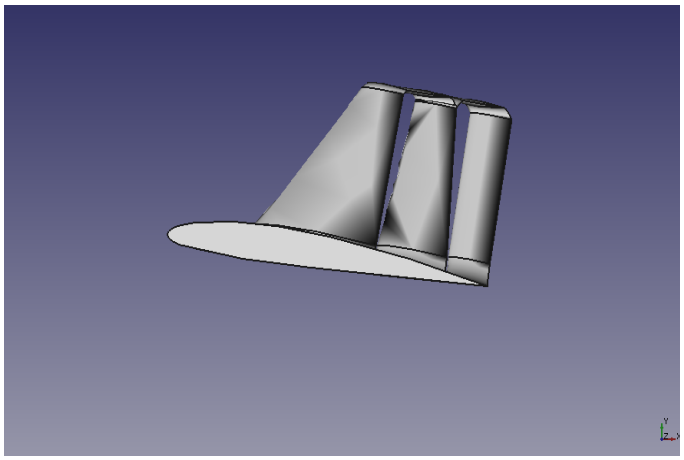


Figure A2.1: *AWDS incliné a 8°*

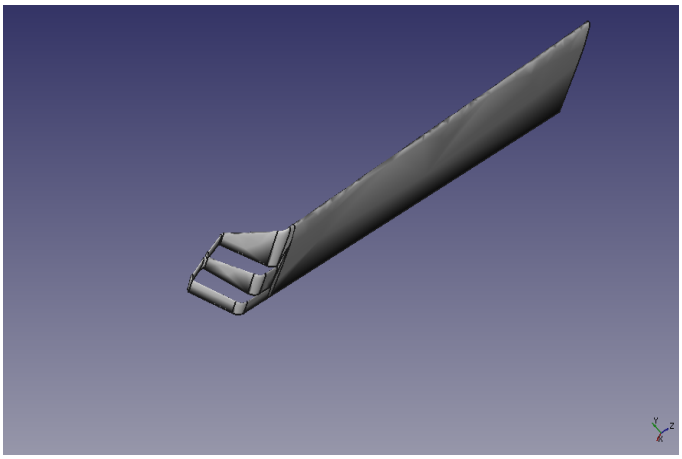


Figure A2.2: *AWDS incliné a 8°*

Dans l'atelier part nous avons créé un cube qu'est notre domain de calcul et en positionné ,ensuite nous avons faire une soustraction entre l'aile et le cube

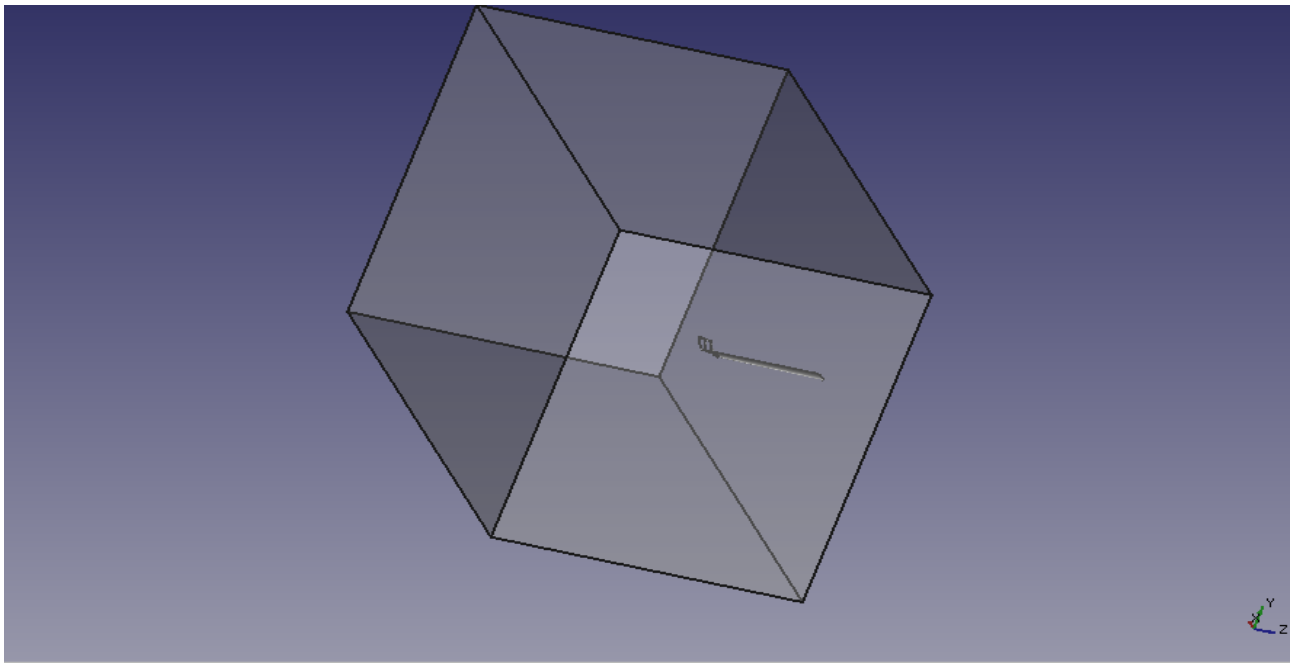


Figure A2.3: *soustraction entre l'aile et le cube*

Les conditions d'analyse

Dans l'atelier workbench cfd nous avons analysé notre domain de calcul par des conditions d'analyse qui presente le tabeau suivant :

Le modèle physique	Ecoulement	Incompressibel viscoux
	Turbulent	RNAS
	Model	KOmegaSST
Les propriétés de fluide	Le fluide	Air
	La densité	1.2 kg/m^3
	La viscousité dynamique	1.8e-05 kg/(m*s)
L'nitialisation des variables de flux internes	K	0.01 m^2/s^2
	Omega (w)	57 deg/s
La vitesse (la vitesse de l'avion dans la décolage)		70 m/s = 252 km/h

Tableau A2.1: *Conditions d'analyse.*

Maillages

Pour créer un maillage dans FreeCAD , d'abord nous sélectionner la soustraction que nous avons fait,ensuit nous appliquons sur créez maillage  qui ce trouve dans la bare d'outile de cfd of,nous choisir type de mailleur et la taille de l'élément de base .Dans noter travail nous avons utilisé le mailleur cfMesh de type cartésien .

Nous avons faire une raffinement de maillage de l'aile et la winglet chaque un est sa taille relative et nou avons crée les couches limits qui nous avons entrer des données de leur paramétres.

Nous avons utilisé deux volumes de contrôles. Le premier, de forme parallélépipédique, situé juste derrière l'aile afin de capter le sillage de l'aile et le deuxieme, de forme cône de langueur 12 m, englobant et derrière la winglet afin de capter les modifications apportées par la winglet sur l'écoulement.

Le tableau suivant presente tout les paramétres de maillage

Mailleur		Cartésien
Taille de l'élément de base		1 m
Epaisseur de raffinement		40 mm
Paramètres de raffinement(cfMesh) taille relative de l'élément	Winglet	0,015
	Aile	0,31
	Cône	0,63
	Boîte	0,63
Nombre de couches limites		15
Taux d'expansion		1,10
Première hauteur de cellule		4,8 mm

Tableau A2.2: *paramétres de maillage*

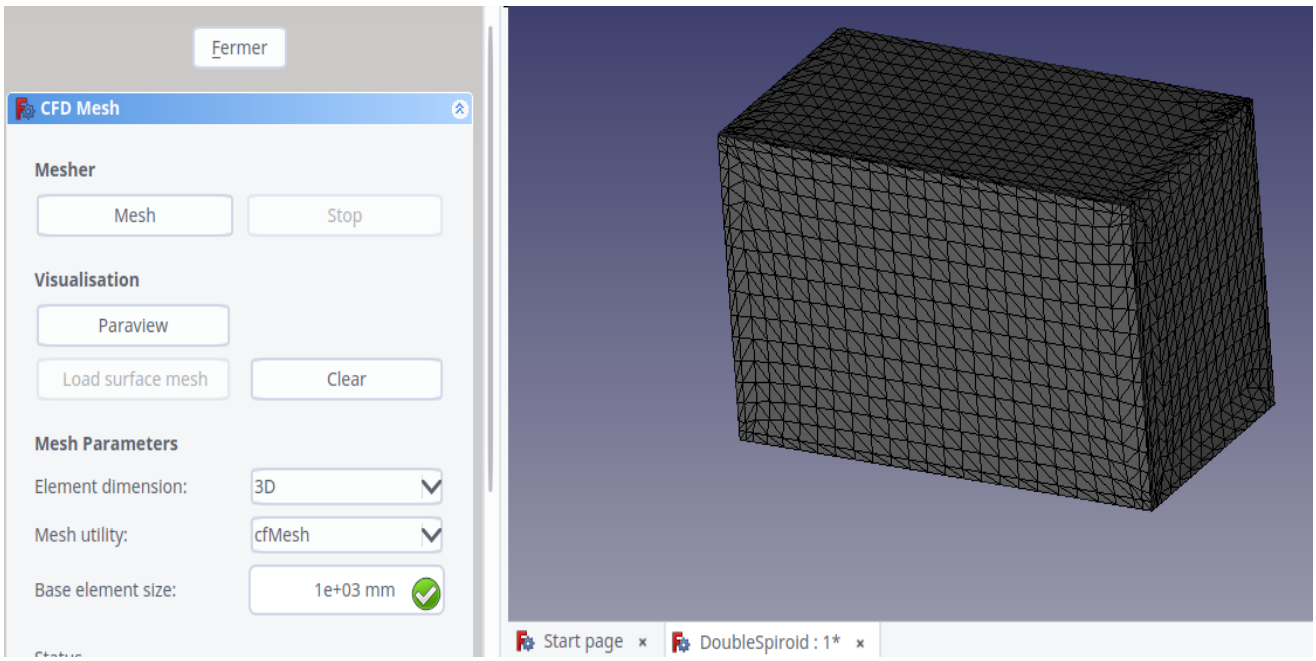


Figure A2.4: *Taille de l'élément de base de maillage*

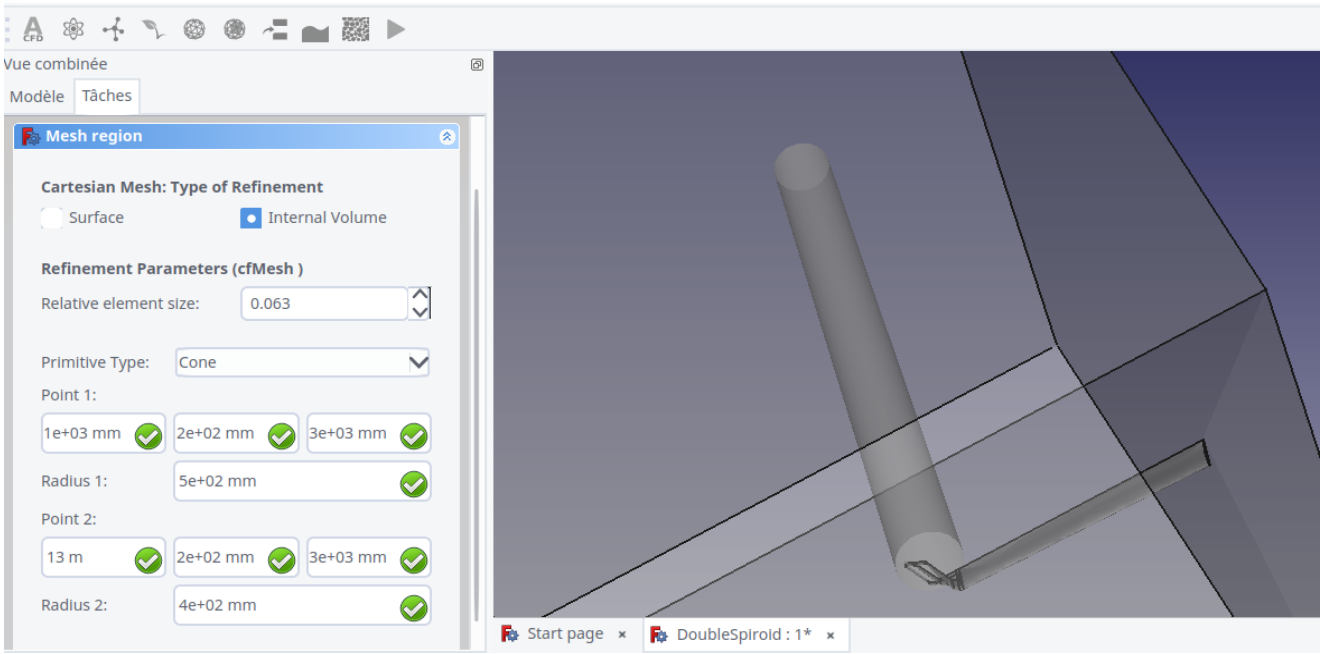


Figure A2.5:*volume de control derrière la winglet*

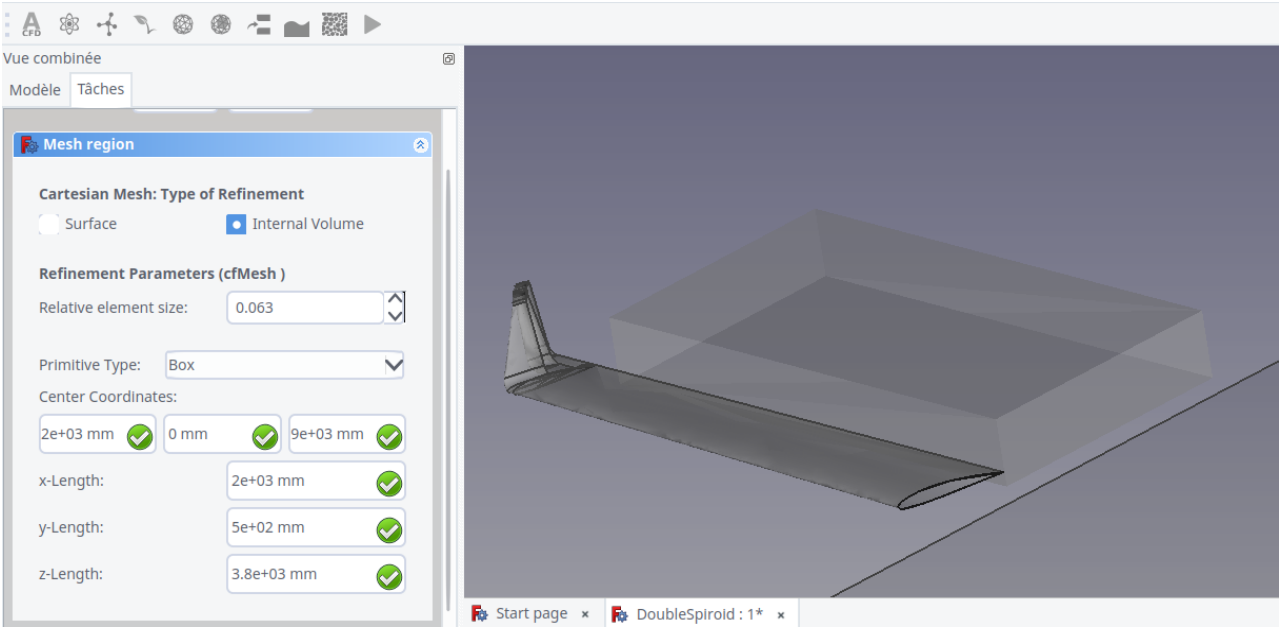


Figure A2.6: *volume de control derrière l'aile*

Après tout les étapes précédents nous lançons dans CFD mesh le maillage dans CFD mesh et nous trouve qu'il a crée un nouveau dossier dans notre repertoire d'enregistrement sous nom de meshCaseou ce trouve tout les données et les paraméteres du maillage

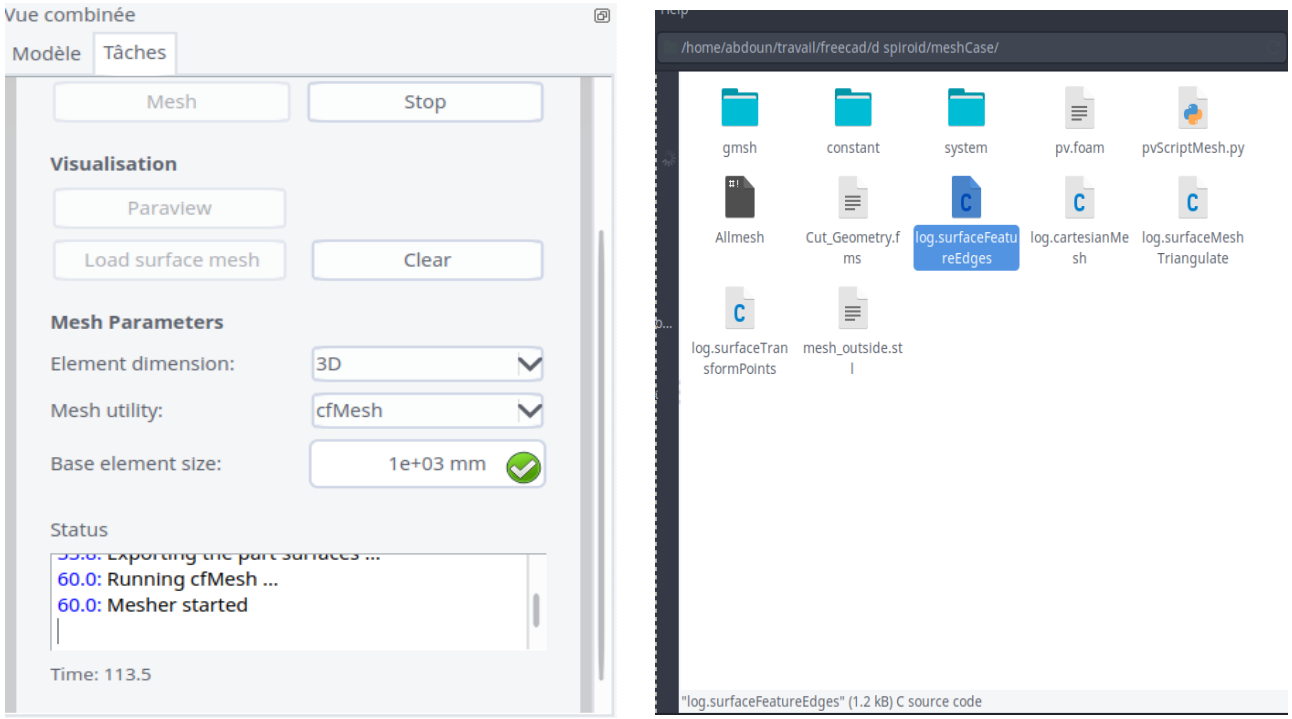


Figure A2.7: *le lancement de maillage et le dossier meshCase*

Nous allons appliquer toutes les étapes précédentes sur les autre configurasions des winglets ,nous pouvons les appliquer de deux manières par conception dans FreeCAD ou à travers les macro (voir l'annex)
Les résultats sont affichés dans le tableau et les images sont situées en dessous .

Configuration AWDS

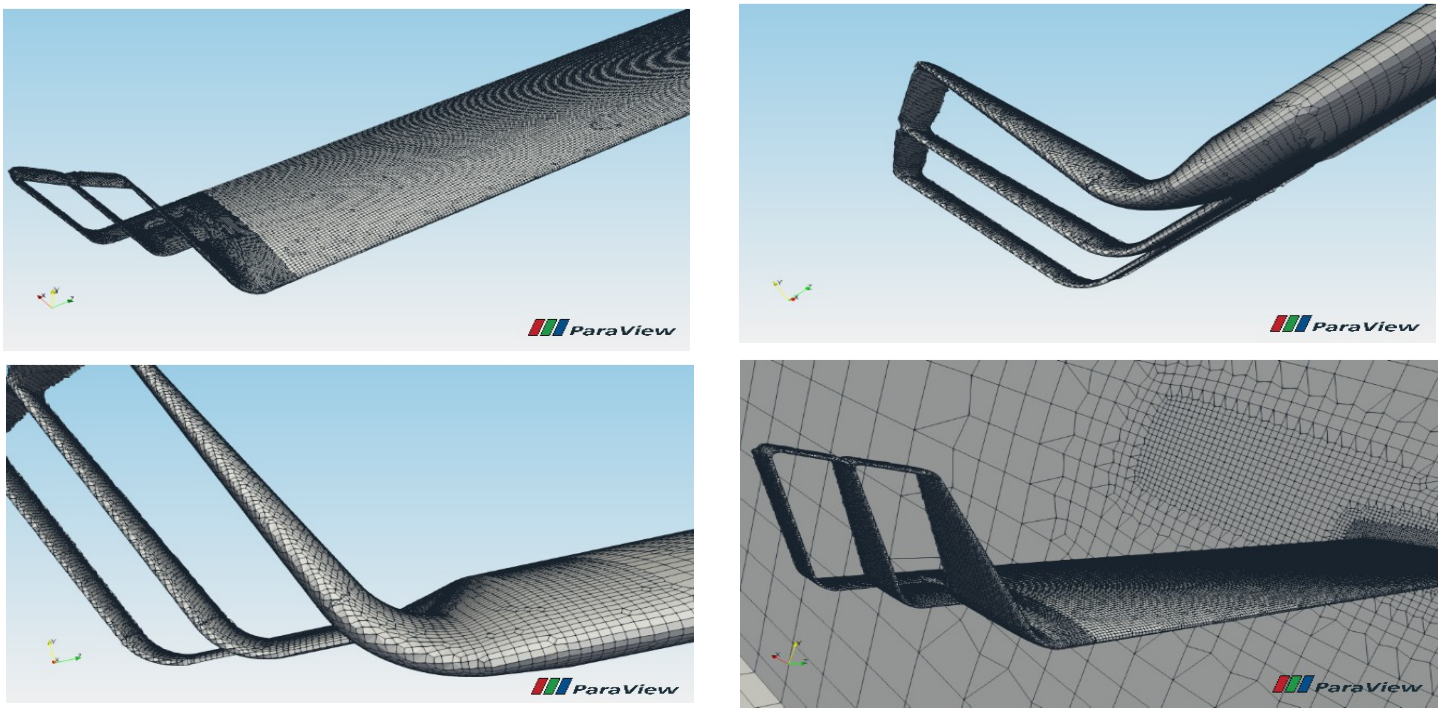


Figure A2.8: *La configuration AWDS*

Les conditions aux limites

Pour créer les conditions aux limites dans le d'atelier cfd of, nous appliquons sur « create au CFD boundary » a chaque fois que nous creons un type de condition au limit.pour l'entre nous avons sélectionné « inlet » et la face de l'entre et nous avons donné la valeur de vitesse ,pour la sortie nous avons sélectionné « outlet » ,la face,et nous donne la valeur de pression,et pour les contraintes « constrainte » ,de type « symmety » et nous selectioner la face ,enfin les auter faces nous pronder comme des murs par « wall » de type « slip » (glaissement) .

Le domaine de calcul avec lequel nous avons travaillé est le même pour tous les cas. La figure ci-dessous présente les dimensions de ce domaine.

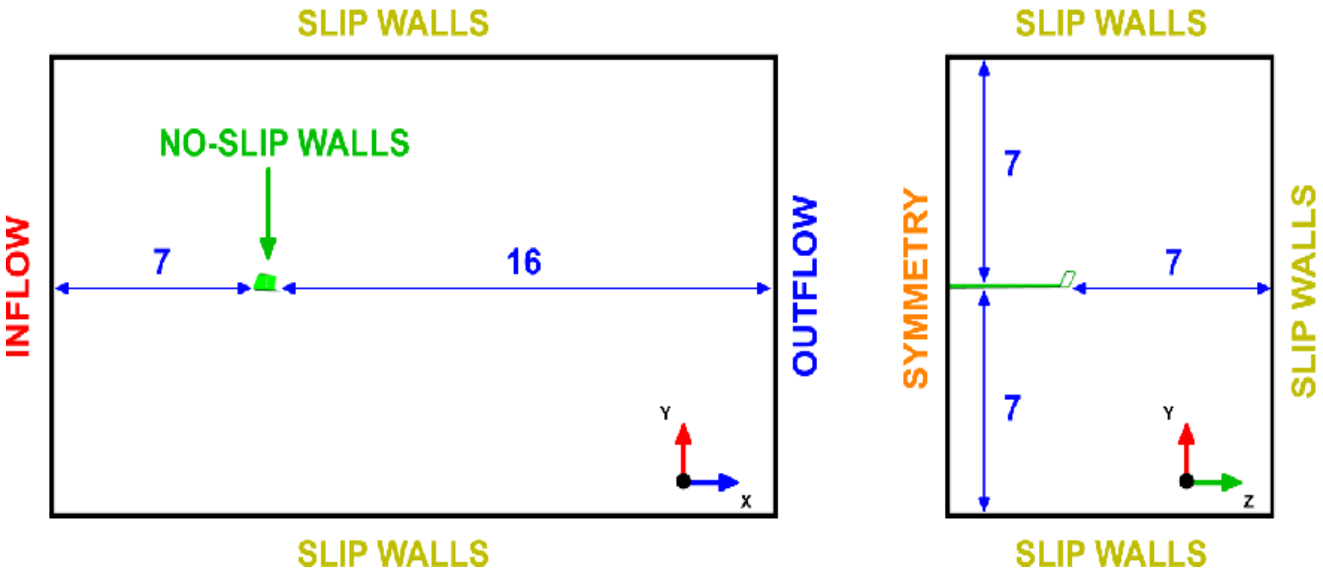


Figure A2.9: *Domaine de calcul et les conditions aux limites (les distances sont en mètres*

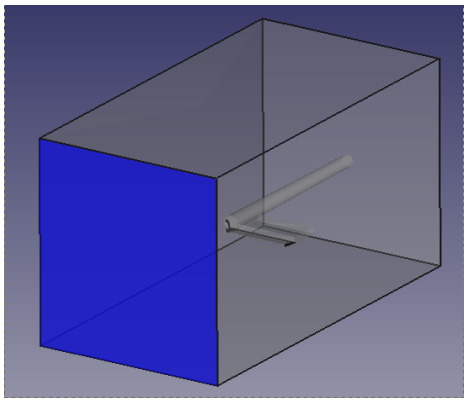
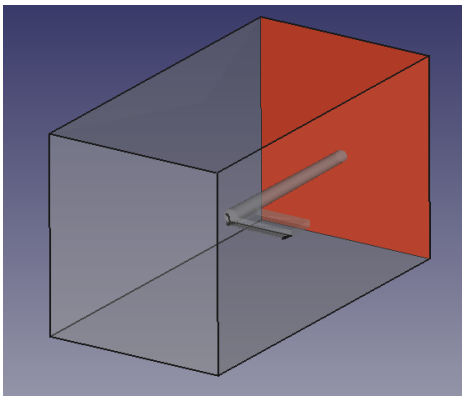
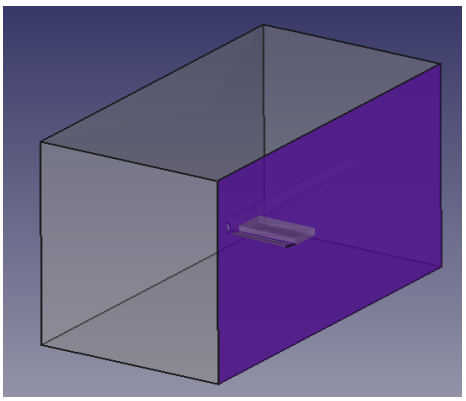
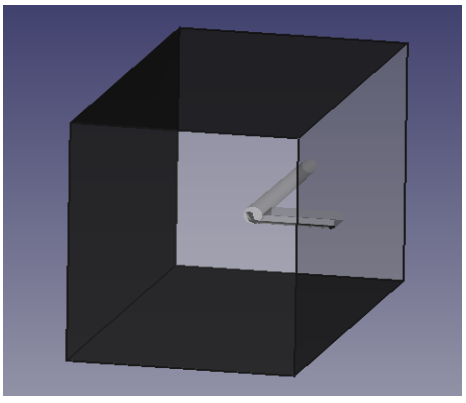
Conditions aux limites	La face qu'il représente
L'entreé (inlet)	
La sortie (outlet)	
La contrainte (symmetry)	
Les murs (slipwall)	

Tableau A2.3: *Les conditions aux limites.*

Après les étapes des conditions aux limite nous créons le dossier Case et nous lançons les calculs en appuyant sur « write » ensuit « Run »,ces étapes nous avouns faire quand nous appuyant sur la commande « OpenFOAM » OpenFOAM .

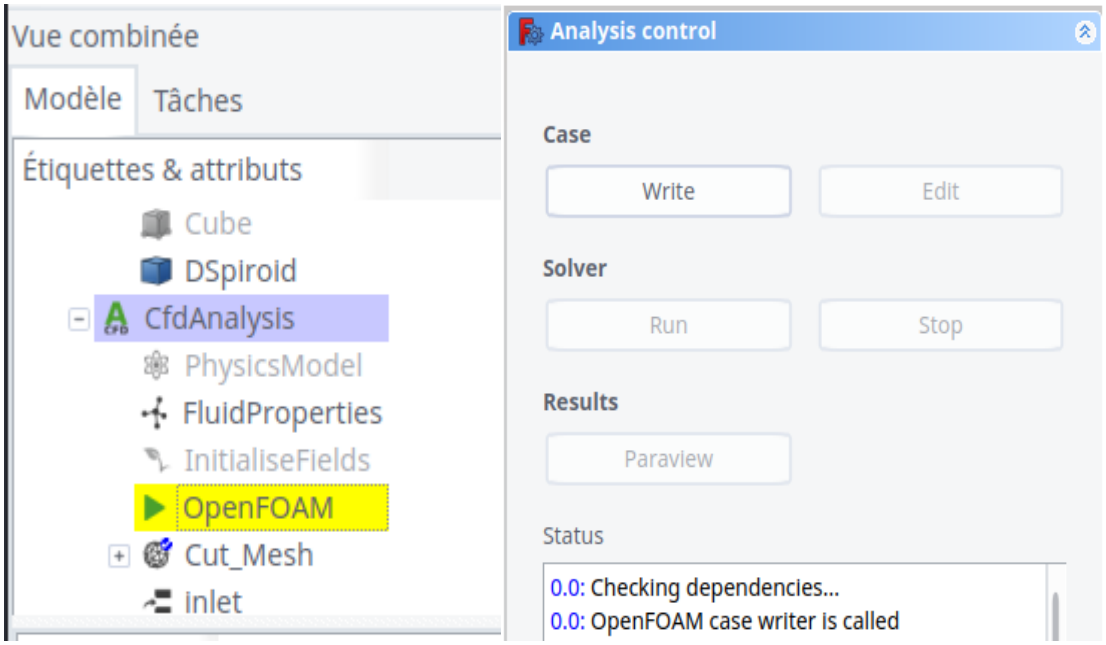


Figure A2.10: *Le dossier Case et lancement des calculs*

Macro

Ces commandes, qui servent à faire les macros, se trouvent sur la barre d'outils des macros: Sur la barre d'outils, il y a 4 boutons: Enregistrement, Arrêt de l'enregistrement, Édition de la macro et Exécuter la macro

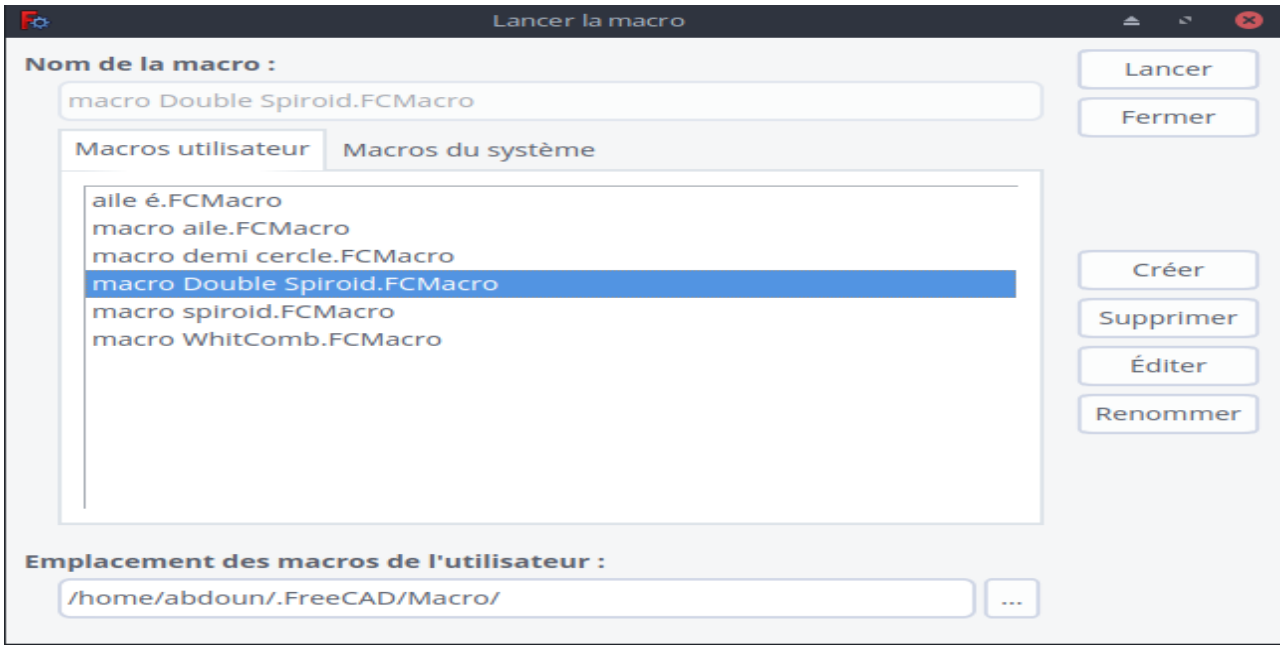


Figure A2.11 *:Interface listant les macros disponibles dans le système*

Modification cfMesh et OpenFAOM

Modification du dossier systeme

Dans cette partie, nous avons parlé des modifications dans les fichiers cfMesh (meshDict, controlDict), dans lesquelles nous pouvons modifier toutes les données de maillage et la géométrie de la zone de raffinemet (box , cone) , ainsi que permettre de modifier le nombre de couches limits dans ce fichier meshDict.

Dans le dossier systeme/ controlDict nous avons ajouter une fonction pour lire une outer fichier dans le dossier system et ce fonction est #include "forceCoeffs" ,elle permet de lire le fichier « forceCoeffs » que nous avons ajouté à partir du dossier « motorBike » et que ce fichier contient des données pour Soulever et traîner en englais « Lift » and « Drag » ,et les fonctions de la vitesse (magUInf) et la surface (Aref) .

Le fichier controlDict, nous pouvons spécifier le temps de début et de fin du calcul , le pas de souvegarder des résultats et l'application qui il sont utilisé (simpleFoam).Un tel exemple :

```
/*--------------------------------*- C++ -*----------------------------------*\
|  =========                 |                                                 |
| \\      /  F ield          | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration      | Version:  4.x                                   |
|   \\  /    A nd            | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation   |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
   version    2.0;
   format     ascii;
   class      dictionary;
   object     controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
application          simpleFoam;
startFrom            startTime;
startTime            0;
stopAt               endTime;
endTime              1500.0;
deltaT               1.0;
writeControl         timeStep;
writeInterval        50;
purgeWrite           0;
writeFormat          ascii;
writePrecision       8;
writeCompression     uncompressed;
timeFormat           general;
timePrecision        6;
runTimeModifiable    true;
```

```
libs
(
    // Needed for availability of porous baffle boundary in potentialFoam
    "libturbulenceModels.so"
);
functions
{
  #include "forceCoeffs"
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
```

Nous avons modifier aussi dans le ficher decomposeParDict la valeur qui donne par defaut par 4 pour creé les postprocess

Modification du dossier constant

Dans le dossier constant / polyMesh / ,nous trouvons le fichier « boundary », qui contient tous les faces du domaine du calcul et du l'aile , où nous modifions leurs noms et type selon sa fonction dans l'expérience.

Les éléments qui contient « nface » égale à 0 nous supprimer et nous conserver le reste des éléments,enfin nous modifier le nomber des feces qu'ils restent.

Nous avons fait le changement comme suit:

L'emplacement	Le nom	Renommer	Type
Les face de box	face0_0	inlet	Patch
	face1 _1	face1	slipwall
	face2 _2	symwall	symmetry
	face3_3	face3	slipwall
	face4_4	face4	slipwall
	face 5_5	outlet	patch
Les faces de l'aile et winglet	Le reste des faces	defaultFaces	wall

Tableau A2.4: *modifications des faces dans les fichiers de OpenFaom*

La meme chose puor le fichier meshMetaDict les faces qu'ils de type « wall »

Le fichier createPatchDict nous permet de regrouper automatiquement les faces par type,comme nous le voyons dans cet exemple:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.x                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      createPatchDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
pointSync false;
// Patches to create.
patches
(
    {
        name defaultFaces;
        patchInfo
        {
            type wall;
        }
        constructFrom patches;
        patches ( face6 face7 face8 face9 face10 face11 face12 face13 face14 );
    }
 {
        name slipWalls;
        patchInfo
        {
            type wall;
        }
        constructFrom patches;
        patches ( face1 face3 face4 );
    }
    {
        name inlet;
        patchInfo
        {
            type patch;
        }
        constructFrom patches;
        patches ( face0 );
    }
    {
        name symWall;
        patchInfo
```

```
        {
           type symmetry;
        }
        constructFrom patches;
        patches ( face2 );
    }
{
        name outlet;
        patchInfo
        {          type patch;
        }
        constructFrom patches;
        patches ( face5 );
    }
);
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
```

Modification du dossier 0

Dans le dossier 0 les modifications dans les facteurs U, k,omega et nut nous avons remplacé ces données de type de « defaultFace » dans nos fichiers par le type de «lowerWall» de fichier d'exempel pitzDaily.

Pour lancer les calcules dans openFaom nous utilisons Allrun aprés la modufication de les commandes pour fait des calcules en parallele après c'était en série .

En série par la commande :

```
# Update patch name and type
runCommand createPatch -overwrite
# Initialise flow
runCommand potentialFoam -initialiseUBCs
# Run application
runCommand simpleFoam
```

En parallele par la commande :

```
# Source tutorial run functions
. $WM_PROJECT_DIR/bin/tools/RunFunctions
# Update patch name and type
runCommand createPatch -overwrite
# Initialise flow
runParallel potentialFoam -initialiseUBCs
# Run application
runApplication decomposePar
runParallel $(getApplication)
runApplication reconstructPar
```

Nous avons lancés les calcul par la commande **>./Allrun** dans le terminal

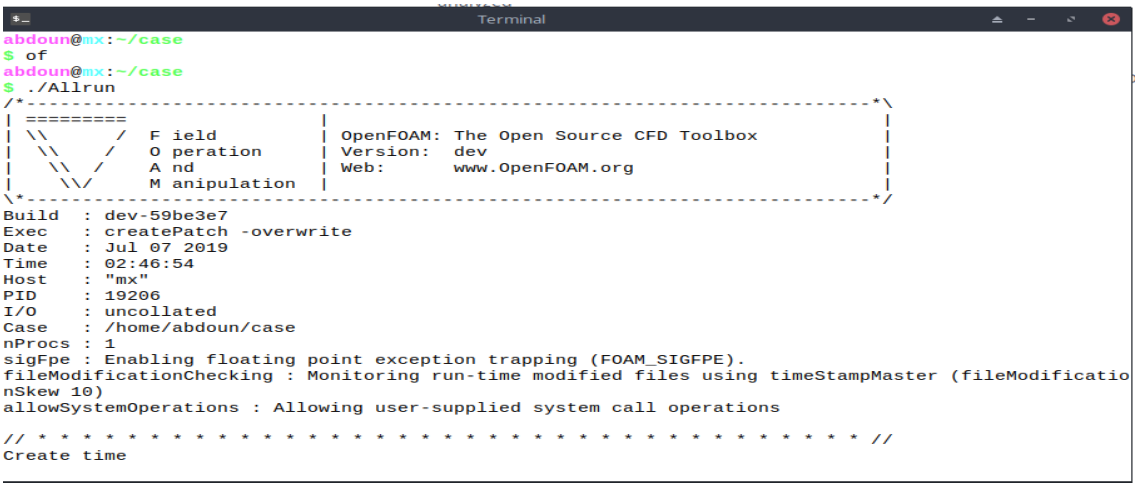


Figure A2.12: *La commande Allrun*

pour visualiser les itérations et les résultat en detail nous appliquons la commande >pyFoamPlotWatcher log.simpleFoam

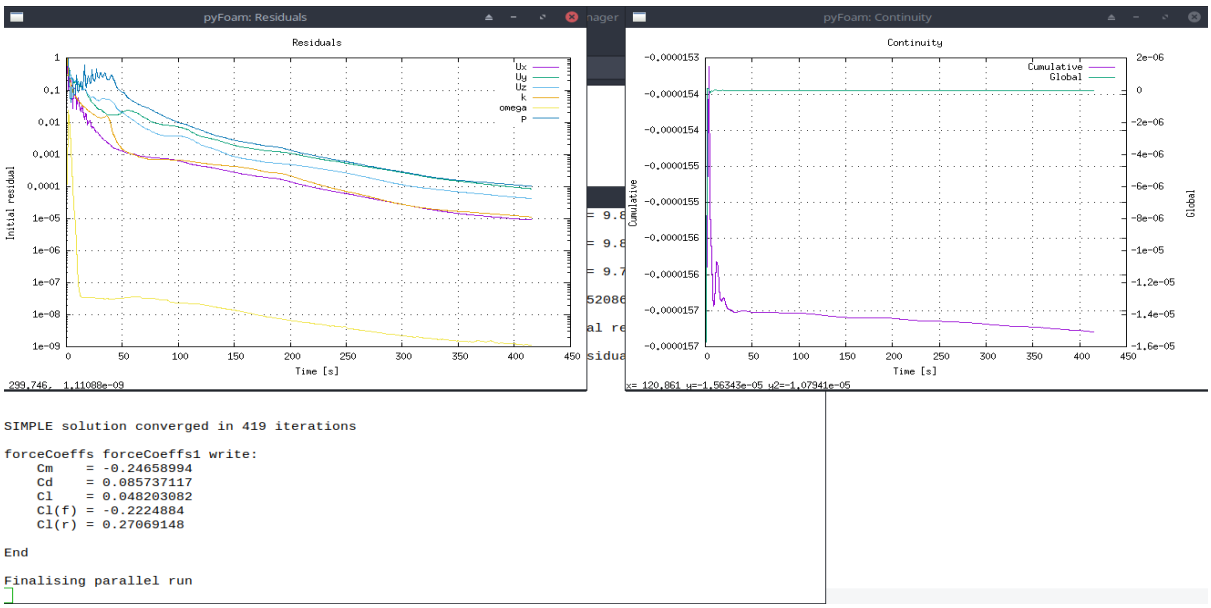


Figure A2.13: *La commande de visualisation et les itérations AWDS*

Après la fin de calcules nous en avons besoin des fonction de l'initialisation dans nos calculs que nous créons dans le dossier0 en terminal comme Q-ctitirion, lambda, vorticity.
les commande des fonctions dans le terminal est :

- Q-critirion ====> **postProcess -func Q**
- Lambda ====> **postProcess -func Lambda2**
- Vorticité ====> **postProcess -func vortisity**

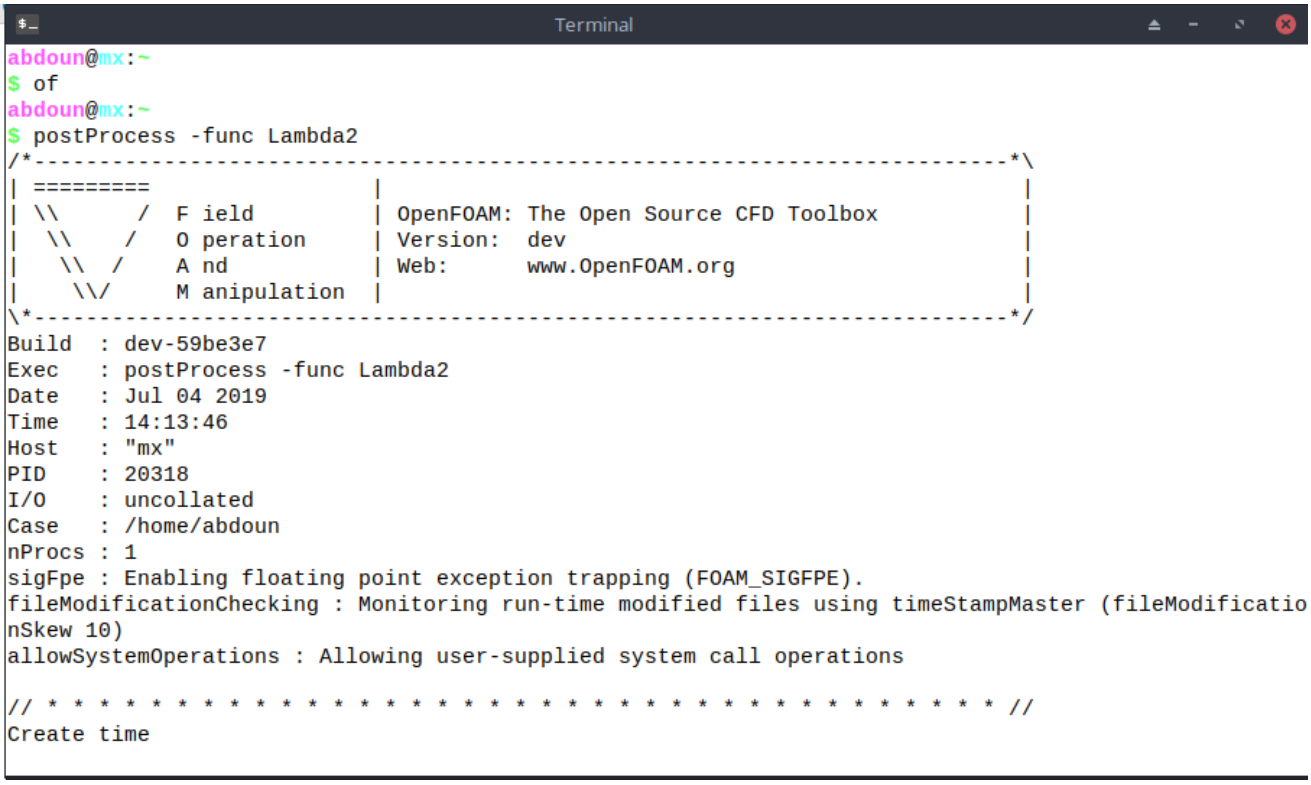


Figure A2.14: *la commande de fonctions Lambda2 dans le terminal*

aprés les étapes présédente nous avons crée VTK dans le dossier de cas par la commande >**foamToVTK -latestTime**

Figure A2.15: *créé le dossier VTK*

nous avons pratiqué tous les étapes précédant dans OpenFOAM pour tout les cas des winglets

A3- Macro pour les fichier de OpenFAOM

U :

```
*---------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.x                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
dimensions      [0 1 -1 0 0 0 0];
internalField   uniform (0.0 0.0 0.0);
boundaryField
{
 slipWalls
    {
        type        slip;
        value       $internalField;
    }
  inlet
    {
        // Fix all three components of velocity on inflow and only the normal component on
outflow,
        // in order to be well-posed if there are some faces on the patch which are actually
outflows.
        type              fixedNormalInletOutletVelocity;
        fixTangentialInflow yes;
        normalVelocity
        {
            type          fixedValue;
            value         uniform (70.0 0.0 0.0);
        }
        value             uniform (70.0 0.0 0.0);
    }
 symWall
```

```
    {
        type        symmetry;
        value       $internalField;
    }
 outlet
    {
     type        pressureInletOutletVelocity;
        value       $internalField;
    }
  defaultFaces
    {
        type        noSlip;
    }
}
// ************************************************************************* //
```

K :

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.x                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      k;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0.01;

boundaryField
{

    slipWalls
    {
        type        zeroGradient;
```

```
    }

    inlet
    {
        type       turbulentIntensityKineticEnergyInlet;
        intensity   0.01;
        value      $internalField;
    }

    symWall
    {
        type       symmetry;
        value      $internalField;
    }

    outlet
    {
        type       inletOutlet;
        inletValue  $internalField;
        value      $internalField;
    }

    defaultFaces
    {
        type          kqRWallFunction;
        value         uniform 0.01;
    }

}

// ************************************************************************* //
```

P :

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.x                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
```

```
    class       volScalarField;
    object      p;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
dimensions      [0 2 -2 0 0 0 0];
internalField   uniform 0.0;
boundaryField
{
 slipWalls
    {
        type        zeroGradient;
    }
 inlet
    {
        type        zeroGradient;
    }
 symWall
    {
        type        symmetry;
        value       $internalField;
    }
 outlet
    {
        type        fixedValue;
        value       uniform 0.0;
    }
defaultFaces
    {
        type        zeroGradient;
    }

}
// ************************************************************************* //
```

boundary :

```
/*---------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  dev                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       polyBoundaryMesh;
    location    "constant/polyMesh";
    object      boundary;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

5
(
    inlet
    {
        type            patch;
        nFaces          271;
        startFace       4618142;
    }
    symWall
    {
        type            symmetry;
        inGroups        1(symmetry);
        nFaces          6124;
        startFace       4618413;
    }
    outlet
    {
        type            patch;
        nFaces          256;
        startFace       4624537;
    }
    defaultFaces
    {
        type            wall;
        inGroups        1(wall);
        nFaces          61895;
```

```
        startFace       4624793;
    }
    slipWalls
    {
        type            wall;
        inGroups        1(wall);
        nFaces          1075;
        startFace       4686688;
    }
)

// ************************************************************************* //
```

ControlDict :

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.x                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     simpleFoam;
startFrom       startTime;
startTime       0;
stopAt          endTime;
endTime         1500.0;
deltaT          1.0;
writeControl    timeStep;
writeInterval   50;
purgeWrite      0;
writeFormat     ascii;
writePrecision  8;
writeCompression uncompressed;
timeFormat      general;
timePrecision   6;
runTimeModifiable true;
```

```
libs

(
   // Needed for availability of porous baffle boundary in potentialFoam
   "libturbulenceModels.so"
);
functions
{
 #include "forceCoeffs"
}
// ************************************************************************* //
```

meshDict

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
| \\    /   O peration     | Version:  4.x                                   |
|  \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|   \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
   version     4.0;
   format      ascii;
   class       dictionary;
   location    "system";
   object      meshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

surfaceFile      "Cut_Geometry.fms";
maxCellSize      1.0;
boundaryCellSize  1.0;
objectRefinements
{
   MeshRegion002
   {
      cellSize 0.063;

      type box;
      centre  (2.0
               0
               -1.9);
```

```
        lengthX  2.0;
        lengthY  0.5;
        lengthZ  3.8;
    }
    MeshRegion003
    {
        cellSize 0.063;

        type cone;
        p0   (1.0
              0.2
              -4.3);
        p1   (13.0
              0.2
              -4.3);
        radius0  0.5;
        radius1  0.4;
    }
}

surfaceMeshRefinement
{
    MeshRegion001
    {
        cellSize  0.031;
        surfaceFile "constant/triSurface/MeshRegion001.stl";
        refinementThickness 0.04;
    }
    MeshRegion
    {
        cellSize  0.015;
        surfaceFile "constant/triSurface/MeshRegion.stl";
        refinementThickness 0.04;
    }
}

boundaryLayers
{
    patchBoundaryLayers
    {
        "face16"
        {
            nLayers 15;
            thicknessRatio 1.1;
```

```
    maxFirstLayerThickness 0.0048;
}
"face17"
{
    nLayers 15;
    thicknessRatio 1.1;
    maxFirstLayerThickness 0.0048;
}
"face14"
{
    nLayers 15;
    thicknessRatio 1.1;
    maxFirstLayerThickness 0.0048;
}
"face15"
{
    nLayers 15;
    thicknessRatio 1.1;
    maxFirstLayerThickness 0.0048;
}
"face12"
{
    nLayers 15;
    thicknessRatio 1.1;
    maxFirstLayerThickness 0.0048;
}
"face13"
{
    nLayers 15;
    thicknessRatio 1.1;
    maxFirstLayerThickness 0.0048;
}
"face10"
{
    nLayers 15;
    thicknessRatio 1.1;
    maxFirstLayerThickness 0.0048;
}
"face11"
{
    nLayers 15;
    thicknessRatio 1.1;
    maxFirstLayerThickness 0.0048;
}
```

```
"face23"
{
   nLayers 15;
   thicknessRatio 1.1;
   maxFirstLayerThickness 0.0048;
}
"face22"
{
   nLayers 15;
   thicknessRatio 1.1;
   maxFirstLayerThickness 0.0048;
}
"face18"
{
   nLayers 15;
   thicknessRatio 1.1;
   maxFirstLayerThickness 0.0048;
}
"face19"
{
   nLayers 15;
   thicknessRatio 1.1;
   maxFirstLayerThickness 0.0048;
}
"face6"
{
   nLayers 15;
   thicknessRatio 1.1;
   maxFirstLayerThickness 0.0048;
}
"face7"
{
   nLayers 15;
   thicknessRatio 1.1;
   maxFirstLayerThickness 0.0048;
}
"face21"
{
   nLayers 15;
   thicknessRatio 1.1;
   maxFirstLayerThickness 0.0048;
}
"face20"
{
```

```
        nLayers 15;
        thicknessRatio 1.1;
        maxFirstLayerThickness 0.0048;
    }
    "face8"
    {
        nLayers 15;
        thicknessRatio 1.1;
        maxFirstLayerThickness 0.0048;
    }
    "face9"
    {
        nLayers 15;
        thicknessRatio 1.1;
        maxFirstLayerThickness 0.0048;
    }
  }
  optimiseLayer 1;
  untangleLayers 0;
 optimisationParameters
  {
    nSmoothNormals 3;
    maxNumIterations 5;
    featureSizeFactor 0.4;
    reCalculateNormals 1;
    relThicknessTol 0.1;
  }
}
// ************************************************************************* //
```