Open source Computational Fluid Dynamics using OpenFOAM

Medina, H. J. , Beechook, A. , Saul, J. , Porter, S. , Aleksandrova, S. and Benjamin, S.

Open source Computational Fluid Dynamics using OpenFOAM

H. Medina[1]*, A. Beechook[1], J. Saul[1], S. Porter[1], S. Aleksandrova[1] and S. Benjamin[1]

Abstract
Computational Fluid Dynamics (CFD) is a tool that allows designers and engineers to readily evaluate the merits of a given design. It helps reducing the need to build and test prototypes predominantly at the early stages of the design process. CFD has been largely adopted by commercial aircraft manufacturers and is now considered an essential design tool. However, CFD is not as widely used for the design of light aircraft (for example, by amateur or enthusiast designers) mainly due to the cost associated with commercial CFD software. In recent years, a number of open-source CFD initiatives have emerged which have the potential to offer amateur or enthusiast light aircraft designers unrestricted access to CFD technology. In particular, the OpenFOAM project offers one of the most complete open-source CFD solutions and their software can be used without the need to purchase an expensive licence. Nonetheless, there are still some challenges to overcome in order to allow the widespread adoption of the CFD technology OpenFOAM offers by the light aircraft design community. In order to promote the use of OpenFOAM, an introduction of CFD using OpenFOAM (and some complementary open-source tools) will be presented, as well as practical examples and information to allow potential users to overcome some of the common challenges they are most likely to encounter.

Keywords
CFD, OpenFOAM, open source, aerospace, light aircraft, design, porous region

[1]*Centre for Mobility and Transport, School of Mechanical, Aerospace and Automotive Engineering, Coventry University, United Kingdom*
***Corresponding author**: h.medina@coventry.ac.uk or h.medina@aerofluids.org

Contents

1. Introduction

Computational Fluid Dynamics (CFD) is a tool used to predict the performance of engineering systems or sub-systems on which fluid flow (and other physical mechanisms e.g. heat transfer, combustions, two-phase flows, etc.) plays significant role. Consequently, it is not surprising that CFD is commonly used in a wide range of engineering sectors including the automotive, wind energy and aerospace industries. Regardless of the particular industrial application, CFD is used as a design tool to complement (and in some cases 'replace') traditional design methodologies involving extensive experimentation and/or prototyping. Despite its limitations[1], one of the advantages of incorporating CFD into the engineering design process is that it allows to assess the comparative merits of various design decisions relatively quickly and at a reduced cost (when compared to experimental programmes).

Examples of how CFD is aiding engineering design can be readily found. Della Vecchia et al. [3] used CFD in conjunction with MATLAB® to propose an optimised design of a new regional turboprop aircraft configuration for 90 passengers. The new design was developed starting from the typical configuration of existing turboprop aircraft for 70 passengers.

---

[1]Understanding the limitations of CFD is important before it can be reliably applied. This is a rather broad topic and it will not be covered in this paper. The interested reader is referred to [1] and [2]

This example illustrates how CFD can be used to develop new design concepts based on existing, and proven, designs. Furthermore, Della Vecchia's design resulted in an airframe configuration with improved aerodynamic performance. Boelens [4] used the $k - \omega$ turbulence model to study the effect that leading edge geometry features and flap gaps have on the performance of the X-31 aircraft at a range of angles of attack, including post-stall angles of attack. The CFD results were compared with experimental data and, in general, the agreement is shown to be acceptable. Nonetheless, the CFD simulations over-predicted the lift and pitching moment coefficients (particularly at higher angles of attack). Panagiotou et al. [5] developed an optimised winglet design for the MALE UAV using CFD. They used the Spallart-Allmaras turbulence model to predict how various winglet designs affected the lift-to-drag ratio characteristics of the UAV. The optimised design led to an increase in endurance of 10%.

In summary, these examples serve to illustrate how CFD is aiding engineers as they seek to develop new designs and/or improve the performance of existing aircraft. Although CFD can now be considered an established design tool and it is routinely employed in aircraft design,CFD is not widely incorporated into the design of light aircraft (manned), in particular, by amateur and enthusiast light aircraft designers. This trend could be attributed to many factors, which to the knowledge of the authors, have not been investigated and/or identified in the open literature. It could be argued that the cost of commercial packages as well as the need to undertake training to be able to confidently apply CFD to light aircraft design may be contributing factors. This paper aims to introduce the fundamentals of CFD using an open-source package (OpenFOAM) with the purpose to promote the use of CFD for light aircraft design.

2. Brief introduction to CFD

Computational Fluid Dynamics is essentially a method for solving a set of partial differential equations that represent a fluid system. These typically include equations representing the principles of conservation of mass, momentum and energy, as well as auxiliary equations to represent other physical phenomena or sources e.g. porous medium, heat exchangers, actuator discs, magnetic fields, etc. Additional transport equations can also be included within a CFD solution to model the transport of a given property such as species concentration in the case of combustion modelling, or turbulent quantities such as the turbulent kinetic energy k and its dissipation rate ε when modelling turbulence using the standard $k - \varepsilon$ model. Therefore, CFD can be used for a wide range of engineering applications provided suitable equations to capture the dominant physics are formulated and implemented into the CFD solver/code. In this section, a brief introduction to the most common techniques for modelling/simulating fluid flow and turbulence are presented, along with relevant and useful sources of further information for the interested reader.

2.1 Modelling approaches

In general, there are three main approaches used to model turbulence: (i) Direct Numerical Simulation (DNS), (ii) Large Eddy Simulation (LES), and (iii) Reynolds Averaged Navier-Stokes (RANS). In DNS, the Navier-Stokes equations are solved directly (i.e. without the need for a turbulence model) and all the turbulent scales must be captured by the computational grid or mesh. Turbulent flows are characterised by a wide range of time- and length-scales. For example, the airflow over an aircraft involves large scales (e.g. wing tip vortices, vortex shedding from the wing, etc.) and small scales (e.g. the flow within the boundary layer or within imperfections present on the aircraft's outer structure, such as gaps, steps, waviness, etc.). In DNS, all these structures must be resolved by the grid. Therefore, DNS is the most computationally demanding and expensive technique to model turbulence. At present, despite the rapid increase in computing power, DNS is mostly used as a research tool to study flows at low Reynolds numbers ($Re = UL/v$) and is not considered a feasible tool for the study of industrially relevant flow configurations or those involving high Reynolds flows.

Another approach used to study turbulence is Large Eddy Simulation (LES). As the name suggests, in LES the large eddies (or turbulent flow structures) are simulated. On the other hand, the small scales (i.e. those that fall within the inertial sub-range) are modelled using sub-grid scale models. The separation of scales is achieved using a spatial filter typically related to the grid size. Therefore, the definition of a "good" grid for LES is somewhat difficult to specify a priori. However, the separation of scales at the heart of the LES approach leads to a significant reduction in the number of cells needed to carry out a simulation when compared with DNS. Therefore, LES can be used to study industrially relevant configurations as it is significantly less computationally demanding than DNS whilst proving a wealth of information about the dominant flow structures and their temporal evolution. As a result, LES is now generally employed to investigate problems for which the transient features of the flow are important (such as aero-acoustics and combustion applications).

LES is generally too computationally demanding to be used as a design tool during the early stages of the engineering design process where typically only top-level performance estimations may be required or needed. To this end, the Reynolds-Averaged Navier-Stokes (RANS) approach has been widely used and it has underpinned aircraft design for decades. In RANS, the Navier-Stokes equations are Reynolds-averaged and the flow field is averaged after being decomposed into mean and fluctuating components. This process introduces a series of unknown quantities known as the Reynolds stresses which are modelled based on statistical information about the flow. Consequently, RANS models are used to incorporate the effects of turbulence into the solution and provide a means for determining the Reynolds stresses without the calculation of the detailed turbulent flow field.

Since RANS models are developed based on statistical information, they are typically fine-tuned for specific applications and there is a very large number of models available. The selection of an appropriate RANS model is one of the challenges faced when applying the RANS approach for engineering design. Additionally, since the Navier-Stokes equations have been averaged to derive the RANS equations, they can only provide solutions or information about the mean flow. However, RANS is orders of magnitude less computationally demanding than DNS and LES, and as a result, it can rapidly provide information about how design decisions may affect the performance of a given design. For this reason, RANS simulations are commonly used in industry despite its obvious limitations.

2.2 CFD workflow

In this section, the typical work flow that is used in CFD will be briefly introduced with the purpose of helping explain the current state-of-the-art (and ease of usage) of open source CFD software (in sections 4 and 5). Figure 1 illustrates in broad terms the typical steps that a user follows in order to complete a CFD study.
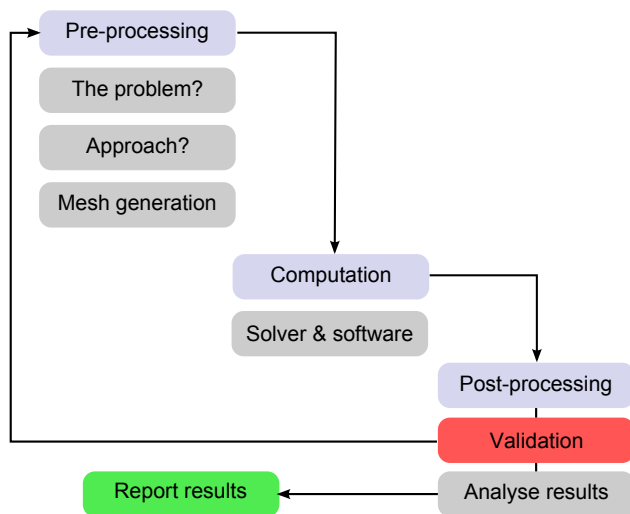

Figure 1. Typical work flow in CFD

The first, and arguably, the most challenging step is pre-processing. At the stage, the user evaluates the nature and dominant features of the problem and, if possible, makes assumptions in order to reduce complexity. Once the dominant physics are identified, the user will select the simulation approach (i.e. DNS, LES or RANS) - typically this choice is made based on computational resource availability. The next stage in the simulation work flow involves the spatial discretisation of the simulation domain by building a grid or mesh suitable to the chosen simulation approach (i.e. DNS, LES or RANS). It is worth emphasising that the meshing stage is perhaps the most critical and time consuming step in a typical CFD simulation and the difficulty of generating a 'good' mesh

increases with the complexity of the geometry of the problem. Also, the accuracy and validity of the final CFD solution is directly linked to the quality of the underlying mesh employed during the simulation. The final step in the pre-processing stage is to assign appropriate and representative boundary conditions. The choice of boundary conditions will depend on the choice of simulation approach and the physics that will be incorporated into the simulation. For a typical RANS simulation the user must provide suitable boundary conditions for the velocity, the pressure and the turbulence quantities used by the underlying turbulence models e.g. k and ε if the $k - \varepsilon$ model is used (where k is used to model the turbulent kinetic energy and ε models its dissipation rate). Ultimately, the choice of boundary conditions is model dependant and the user should gain familiarity with the turbulence model so that appropriate boundary conditions can be used, paying special attention to the requirement/s for wall and inlet boundaries.

The second phase in the CFD work flow is to carry out the calculations. To achieve this, the user must select an appropriate CFD solver/software/package. This choice can be influenced by various factors such as: software costs, licensing limitations, ease of use, solvers available, pre- and post-processing tools, software parallelisation, technical support, etc. Another decision the user will face is whether to use open source or commercial (typically without access to the source code) CFD software. Commercial packages have the distinct advantage of being generally well documented and offering technical support to users. However, to benefit from these advantages the cost of utilising commercial packages can be high. On the other hand, open source CFD software may be used at either no cost or at a lower cost when compared with commercial packages and the software developers may or may not offer technical support. This paper aims to promote and highlight the use of open source CFD packages, in particular OpenFOAM, to aid the design of light aircraft and their sub-systems. Nonetheless, there are certain advantages, as well as disadvantages when using open source software (summarised in table 1).

The third stage in the work flow involves post-processing (and validation) of the simulations results. Ideally, a validation study should be embedded within the work flow in order to assess the predictive capability of the software, the validity of any assumptions made and the accuracy/applicability of the chosen turbulence model. This stage is essential to be able to gain confidence or to interpret correctly the results produced by the CFD simulation/s. As part of the post-processing phase, the user will typically use the simulation results to calculate quantities of interest[2] (e.g. aerodynamic coefficients and/or forces) which can be used to assess the merits of a given design or designs. As a result of post-processing, the user may decide to refine the simulation parameters and re-define the pre-processing set up. Alternatively, if the user is satisfied with the results, the findings can be incorporated into the

---

[2]These are/should be typically pre-selected or identified during the pre-processing stage/s

design process (or communicated to others).

advantages	disadvantages
Low or no cost to use	Sometimes the software is distributed with complex licences
Source code made available	Source code may change
User customisable	Steep learning curve
User community support	Fees for technical support may apply
Simulations are typically configured using text files which allows to readily link the simulation set up to third party applications e.g. optimisation tools	Pre-processing and case configuration can be arduous at first (Graphical User Interface may not be available)

Table 1. Typical advantages and disadvantages of using open source CFD software

3. Introduction to OpenFOAM

In this section some of the key features that make OpenFOAM a competitive tool for CFD work, as well as a tool suitable for light aircraft design, are introduced. Also in this section, a brief introduction to the structure of a typical OpenFOAM simulation configuration case file is provided. Please note that only a selection of features, utilities and solvers will be discussed. This section is based on the OpenFOAM user guide [6] and it is not meant as a substitute.

3.1 Features overview
OpenFOAM is, in essence, a collection of libraries with functionality that allows for these libraries to be used to solve CFD problems. It is based on the finite volume method in which the computational domain is divided (discretised) into small volumes and the equations to be solved are re-written to conform to the finite volume approach (the interested reader is referred to Moukalled et al. [7]). At the heart of the OpenFOAM project lies a philosophy of code re-usability, ease of implementation and making the code available to all users to view, review and modify (under the GNU General Public License which ensures that the code remains free and open). As a result of this philosophy, users are allowed to freely modify and contribute to the development of the code (managed by the OpenFOAM Foundation) which has led to an extraordinary rapid pace in its development. For this reason, OpenFOAM is one of the most widely used open source CFD tools available. Also, since OpenFOAM has attracted a large user base with a wide range of scientific/industrial interests, the current default release version (3.0.0) offers users a significant number of features, ranging from solvers for incompressible and compressible flows to conjugate heat transfer, two-phase flows, laminar

flows and turbulence modelling (RANS, LES and DNS), etc. Another interesting feature of the OpenFOAM project is that the development team also offer (commercially) training and support, consultancy and custom code development which has also attracted the interest of industrial users. However, for the academic, amateur or enthusiast user, OpenFOAM offers a platform to perform CFD analysis at no cost, and thanks to a growing user base there is a vibrant user community that can help new users in their discovery of OpenFOAM (e.g. CFD Online [8]). Given the benefits of OpenFOAM outlined above, OpenFOAM has the potential to find a niche user base within the context of aiding individuals and/or organisations with an interest in light aircraft design.

3.2 Useful utilities
OpenFOAM is released with useful utilities to help the user throughout the entire CFD work flow, from pre-processing, mesh generation, mesh format conversion to post-processing of simulation results. Herein, a summary of utilities that may be of particular interest within the context of light aircraft design are highlighted in table 2. It is worth pointing out, in particular for the benefit of new/prospective users, that communication and interaction with OpenFOAM is via the terminal window using written commands i.e. using the name of the executable followed by any options available (that is, the name of utility and/or solver). In order to find out the options available for a particular utility, the user must enter in the command the name of the utility followed by '–help'.

3.3 Available fluid solvers
A solver is a computer executable that provides an algorithm to find the solution to a specific set of equations. Although there are many solvers available in OpenFOAM, only incompressible solvers will be presented since within the context of light aircraft design the airflow can generally be assumed to be incompressible. A list of the incompressible solvers (excluding incompressible multi-phase solvers) available in OpenFOAM along with a short description of their applicability is presented in table 3.

3.4 Case structure and configuration
The simulation case structure, set up and configuration of an OpenFOAM simulation is directly linked to the system of equations and domain regions (e.g. fluid or solid regions for conjugate heat transfer problems) used to represent the physical system to be modelled. In this section, for simplicity and relevance, the typical configuration and case structure for modelling incompressible and single-phase problems will be illustrated. The structure of a simulation case in OpenFOAM is shown in figure 2. This figure shows the top-level of the simulation folder. Within this folder, three additional directories are required to set up a simulation, namely, *0*, *constant* and *system*. In the *0* directory the boundary conditions for the simulation are defined and the user must include a series of text files; one for each one of the variables that define the physics of the problem e.g velocity, pressure and turbulence

Pre-processing	
checkMesh	Performs tests on the mesh and reports mesh quality statistics
applyBoundaryLayer	Generates a simple boundary layer ($1/7^{th}$ power-law) profile for the velocity and turbulence fields
Mesh generation	
blockMesh	Mesh generation tool using blocks. Recommended for simple geometries
snappyHexMesh	An automatic mesh generation tool that fits hexahedral elements to a surface describing the geometry. It is suitable for complex geometries.
Mesh conversion	
ideasUnvToFoam	Converts meshes saved in the .unv format to OpenFOAM format e.g. mesh files generated in SALOME
netgenNeutralToFoam	Converts meshes saved in the neutral file format generated by Netgen
Post-processing	
yPlusRAS	Calculates the value of y^+ for all wall boundaries
wallShearStress	Calculates the wall shear stress for all wall boundaries

Table 2. Useful utilities

Solver	Description
icoFoam	Transient solver for laminar incompressible flows. This solver can be applied to low Reynolds applications e.g. internal fluid systems, small UAVs and MAVs
simpleFoam	Steady state solver for incompressible flows using the SIMPLE algorithm. It can be used for both laminar and turbulent flows (using a background RANS turbulence model)
pisoFoam	Transient solver for incompressible flows using the PISO algorithm. It can be used in conjunction with both LES and RANS turbulence models
pimpleFoam	Transient solver for incompressible flows using the PIMPLE algorithm (a combination of the PISO and SIMPLE algorithms that allows to maintain numerical stability of the simulation whilst using larger time steps). It can be used in conjunction with both LES and RANS turbulence models

Table 3. Incompressible solvers available in OpenFOAM

3.5 Getting started with OpenFOAM

OpenFOAM is natively distributed for Linux operating systems and it can be downloaded on the OpenFOAM Foundation website [9]. There are versions available online that allow OpenFOAM to be installed in Windows, such as the Windows version of OpenFOAM distributed by CFD Support [10]. However, Windows versions may not be up to date and may have code bugs that have already been addressed in the Linux version. In regards to the Linux operating system, OpenFOAM provides binary installations for Ubuntu 14.04 LTS and Ubuntu 15.10. Ubuntu is, perhaps, one of the most popular Linux distributions available, and given its large user base, there is a wealth of community support for newcomers. Ultimately, there are other Linux distributions available for prospective OpenFOAM users to choose from (Debian, OpenSUSE, Fedora, Linux Mint, Korora, Mageia, Sparky Linux, Gentoo, etc.) based on their personal preference.

Once the prospective OpenFOAM user has a working version of Linux and installed OpenFOAM, there are a number of pre-configured incompressible simulation cases located at *<installation folder>/OpenFOAM <version>/tutorials /incompressible*. It is recommended to run the 'cavity' tutorial first since it is a relatively simple simulation and illustrates the operation of OpenFOAM. To run the tutorial, simply copy the tutorial case anywhere into the local *home* directory and enter *./Allrun* in the terminal ensuring that the current working directory corresponds to where the 'cavity' tutorial is located (the path of the current working directory can be checked by
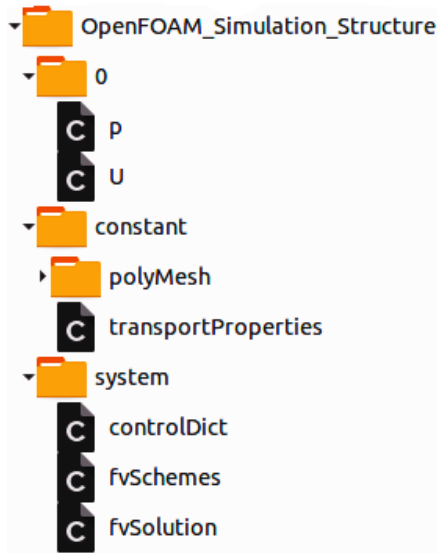
quantities (such as the turbulent kinetic energy, dissipation rate, transformed eddy viscosity, etc. depending on the turbulence model selected by the user). In the *constant* directory, the software requires entries for aspects of the simulation that remain unchanged through out the simulation e.g. the mesh definition, transport models and turbulence model selection. Also within the *constant* folder, a directory named *polyMesh* is present in which the mesh information is stored. Note that the majority of the text files required for this folder are automatically generated when a mesh is created or imported. However, the *boundary* file may need to be edited by the user in order to define base boundary types as detailed in the OpenFOAM user guide [6]. Finally, within the *system* folder, the user defines/controls the simulation by editing at least three text files i.e. *controlDict*, *fvSchemes* and *fvSolution* which are used to control the simulation, define how equations should be discretised and which numerical algorithms should be used to solve the given system of equations.

Figure 2. OpenFOAM simulation case structure

entering *pwd* in the terminal). Once the user understands the 'cavity' tutorial, more complex cases/tutorial can be explored.

Command	Description
pwd	Display current terminal address
cd	Change directory e.g. cd /home/User
cd ..	Change current working path to the parent directory
top	Display system processing currently running i.e. similar to the task manager in Windows
ls	Display contents at the current level

Table 4. Useful Linux terminal commands

4. The challenge of pre-processing

In its own right, pre-processing is challenging since it requires not only an understanding of how a given CFD package operates, but it also requires the user to have a solid understanding of the physics that are present in a given application. Additionally, a crucial step as part of a simulation pre-processing phase is to develop and generate a suitable mesh. Mesh generation, particularly for complex geometries, can be a time consuming activity even when highly specialised commercial grid generation software is used. Furthermore, mesh generation is the main bottleneck for open source CFD because the current state-of-the-art of open source mesh generation tools/software is falling behind commercial products. For example, OpenFOAM offers a very powerful mesh generation tool (*snappyHexMesh*). Although *snappyHexMesh* can generate high quality grids automatically, its implementation

requires significant amount of RAM memory and, from the usability viewpoint, it is difficult to configure it to generate high quality boundary layer grids (inflation layers). Fortunately, there are two reasonable capable open source grid generation tools worth highlighting.

Firstly, the Netgen [11] project provides an algorithm for generating unstructured tetrahedral meshes. It is released with a minimal graphical user interface and it can be used to generate grids for complex geometries. However, Netgen does not offer many features for building CAD geometries. A more complete open source pre-processing solution is provided by SALOME [12] which offers a geometry generation module to build CAD models. Additionally, SALOME includes a mesh generation module capable of producing both hexahedral and tetrahedral meshes, as well as hybrid grids. SALOME can export meshes in *.unv* format which include boundary information and can be easily converted into OpenFOAM format using the ideasUNVtoFoam utility shown in table 2. This makes SALOME an ideal open source pre-processor to use in conjunction with OpenFOAM. The main drawback of the SALOME platform is that it uses significant RAM memory when building tetrahedral meshes (since it uses the Netgen algorithm) and, to the authors knowledge, SALOME is not readily parallelised.

The text-based configuration approach used in OpenFOAM can be challenging to new users and involves a steep learning curve. OpenFOAM does not provide a native graphical user interface (GUI). However, engys Ltd. have developed a GUI for OpenFOAM known as HELYX-OS [13] which has been released as open source. It allows to set up and configure Open-FOAM simulations with ease from pre-processing to post-processing of results using ParaView (see section 5). HELYX-OS embeds the mesh generation utility *snappyHexMesh* and generates split-hex meshes from a user supplied surface geometry files e.g. *.stl* files. The graphical interface helps minimise some of the complexity of setting up the *snappyHexMesh* utility. It allows to create grids for complex geometries with relative ease and minimal user input. Although HELYX-OS does not support all the features available in OpenFOAM, it is, nonetheless, a useful tool for pre-processing and configuring OpenFOAM simulations which can help new users become familiar with the basic functionality of OpenFOAM.

5. Post-processing and ParaView

Post-processing of OpenFOAM results is facilitated thanks to the ability to export OpenFOAM simulation results to a wide range of third party applications (commercial or open source). The post-processing tools included in OpenFOAM are very powerful and allow sampling of points, lines, planes and surfaces though the simulation domain. The samples can be obtained once the simulation has been completed or during runtime. The utilities that are included in Open-FOAM are *probeLocations* and *sample* which require the definition of a 'dictionary file' located within the *system* folder. This 'dictionary file' is used to specify the sam-

pling needs of the user. A sample 'dictionary file' for these utilities can be found in *<installation folder>/OpenFOAM <version>/applications/utilities /postProcessing /sampling /probeLocations /probesDict* and *<installation folder> /Open-FOAM <version>/applications/utilities/postProcessing /sampling /sample/sampleDict*.
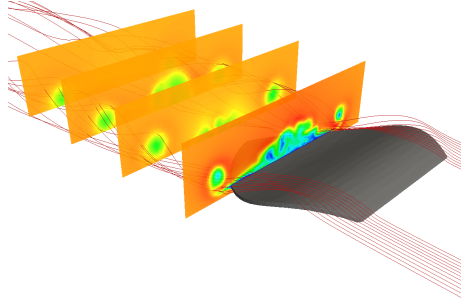

Figure 3. Example of the post-processing capabilities of ParaView showing the airflow over an MAV airframe (based on the simulation results from [14])

In addition to the post-processing tools developed by the OpenFOAM team, OpenFOAM is distributed with ParaView [15] which is a versatile post-processing open source software. ParaView has been in constant development since 2000 and is now a tool that rivals commercial packages. It can read OpenFOAM simulations natively and it offers a wide range of filters and data analysis functionality, ranging from data extraction to complex 3D rendering and animation. For example, the post-processing of the flow over a Micro-Aerial Vehicle (MAV) shown in figure 3 was prepared using ParaView and the simulation results presented in [14]. In summary, ParaView is essentially capable of accommodating the post-processing needs of most users.

# 6. Examples of CFD with OpenFOAM

In this section three examples to showcase the predictive capabilities of OpenFOAM are presented, including (i) Skin friction predictions over a flat plate at zero pressure gradient, (ii) Flow through a 1D porous medium, and (iii) Flow through a planar diffuser with downstream resistance. For succinctness, the results are presented avoiding unnecessary detail. The simulation set up and configuration parameters (including boundary conditions) can be found in the original publications referenced in each section.
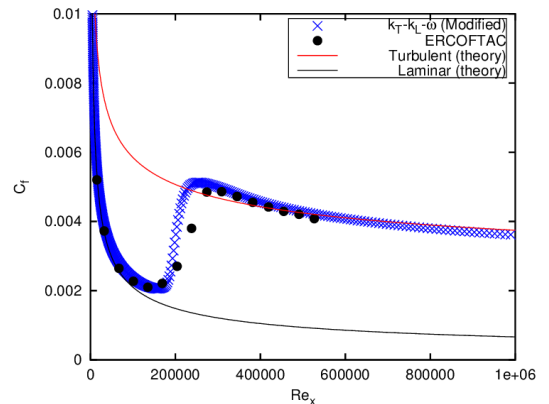
6.1 Flat plate at zero pressure gradient
OpenFOAM has been used to model the flow over a flat plate at zero pressure gradient using three turbulence models. Results for the predicted skin friction coefficient against the Reynolds number based on the distance from the leading edge (Re_x) are shown in figure 4. This figure also shows the theoretical laminar and turbulent skin friction distributions and the experimental results [16] corresponding to a freestream turbulence intensity of 3% (ERCOFTAC case T3A). The $k - \omega$ model [17] is set up in high-Reynolds mode (i.e. it uses
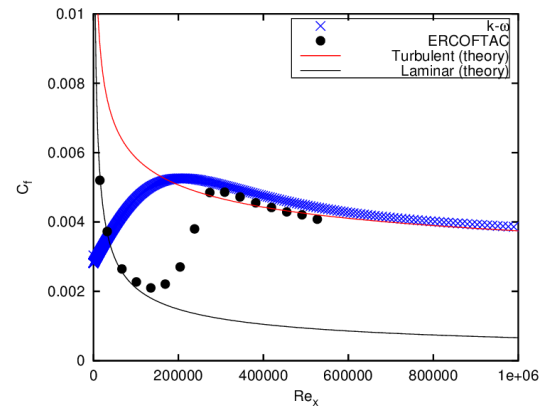
(a) $k - \omega$ model
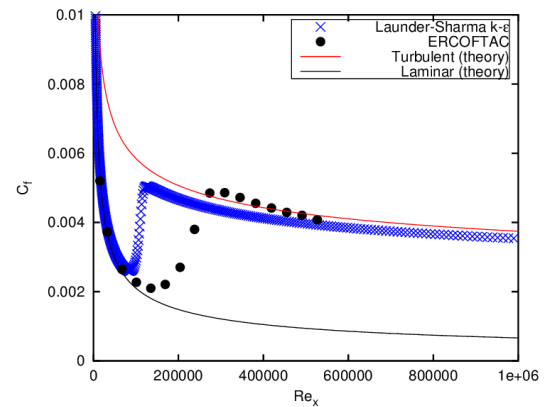
(b) Launder-Sharma $k - \varepsilon$ model

(c) $k - k_L - \omega$ model

Figure 4. Skin friction coefficient over a flat plate at zero pressure gradient and 3% turbulence intensity. The experimental results corresponding to the T3A case from ERCOFTAC [16]
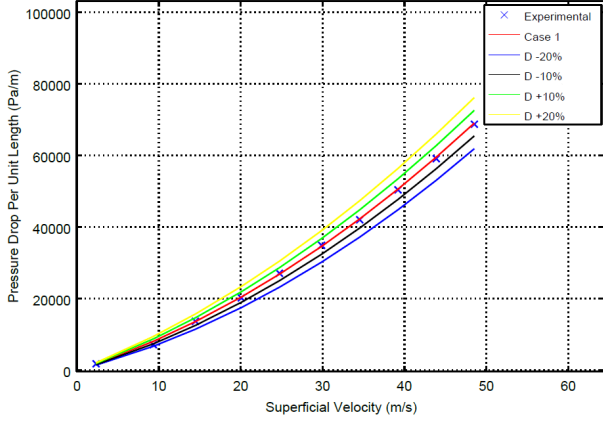
Figure 5. Pressure drop though a porous medium (experimental data extracted from [21])

Figure 6. Downstream velocity profile (experimental data extracted from [22])

wall functions). As expected, the skin friction distribution follows the theoretical turbulent profile (figure 4a) since the wall-function model is based on the logarithmic law of the wall for a turbulent boundary layer at zero pressure gradient. Figure 4b shows the results for the Launder-Sharma [18] low-Reynolds model (i.e. it uses damping functions to allow integration of the solution to the wall). This model shows an initial laminar skin friction solution with a sudden jump towards the theoretical turbulent boundary layer skin friction value. Finally, figure 4c shows the results from the $k - k_L - \omega$ model [19] following the implementation detailed in [20]. The $k - k_L - \omega$ model is a transition model that uses the concept of the laminar kinetic energy to model streamwise pre-transitional velocity fluctuations within the boundary layer to control the transition to turbulence. The results obtained using this model are in excellent agreement with the experimental results [16] (ERCOFTAC case T3A).

6.2 Pressure drop though a porous medium
OpenFOAM offers the flexibility of modelling incompressible flow through porous medium by incorporating the pressure drop due to the porous medium as a source term in the momentum equations. The Darcy-Forchheimer equation is used to model the pressure drop as a source defined as:

$$S_i = - \left(\mu D_{ij} + \frac{1}{2} \rho |U_{kk}| F_{ij} \right) U_i \qquad (1)$$

When the porous medium is permeable only in one direction equation 1 can be written as:

$$S = - \left(\mu D + \frac{1}{2} \rho U_s F \right) U_s \qquad (2)$$

The pressure drop due to a porous medium that is permeable only in one direction can be characterised experimentally by recoding the pressure drop against the superficial velocity, U_s (i.e. the velocity of the flow as it enters the porous
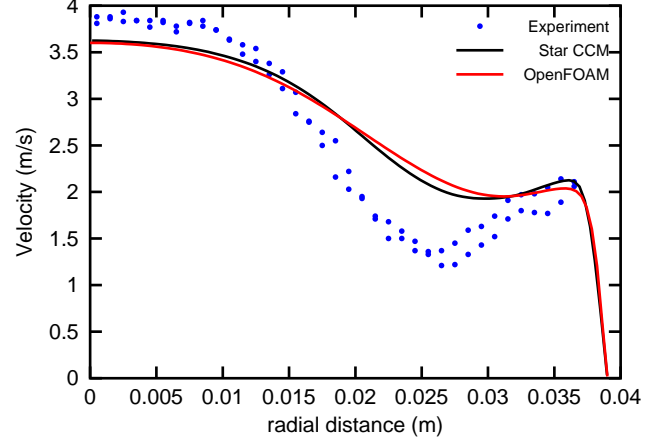
medium), and it can be expressed as shown in equation 3. Using equations 2 and 3, the relationship for α, β and D, F can be found (equations 4 and 5).

$$\frac{\Delta P}{L} = \alpha U_s + \beta U_s^2 \qquad (3)$$

$$\alpha = \mu D \qquad (4)$$

$$\beta = \frac{1}{2} \rho F \qquad (5)$$

Figure 5 shows simulations results based on a configuration similar to that presented in [21]. Case 1 in figure 5 shows that when OpenFOAM is configured using equations 4 and 5, as expected, it returns results that closely follow the experimental correlation used. This demonstrates that OpenFOAM has been correctly implemented to deal with simulations involving porous media using the Darcy-Forchheimer model (equation 1). Also, this figure shows the sensitivity of the porous medium model (equation 1) to changes in the viscous tensor D_{ij}.

6.3 2D diffuser with downstream porosity
In this section, a comparison between the predictions obtained using the commercial CFD package Star CCM+ and OpenFOAM is presented. The mesh used and computational set up is described in [22]. The simulations correspond to a planar diffuser with downstream porosity (permeable only in the streamwise direction). A comparison between the experimental and numerical streamwise velocity component profiles at 40mm downstream from the porous element is presented in figure 6. This figure shows that the porous medium model used (equation 1) is, perhaps, too simple for the numerical results to mimic the flow physics. However, figure 6 also shows that the velocity distributions predictions obtained using Start CCM+ and OpenFOAM are in excellent agreement.

This suggests that the predictive capabilities of OpenFOAM are similar to the commercial CFD software Star CCM+, at least for the configuration tested.

7. Conclusions

In this paper, the CFD software OpenFOAM is introduced along with other open source software that can be used to support the design of light aircraft through the entire CFD work flow. It has been highlighted that OpenFOAM is a powerful tool which offers an extensive range of features to model fluid flow, from incompressible to compressible flows, as well as multi-phase flows. OpenFOAM is distributed with a large number of models, including laminar and turbulence models within the RANS, LES and DNS simulation/modelling frameworks. Pre-processing has been highlighted as the bottleneck of the CFD workflow, in particular, when using OpenFOAM due to its steep learning curve. However, in order to promote and encourage the use of OpenFOAM for light aircraft design, this paper presented other open-source tools that can be used to reduce the difficulties associated with mesh generation and the configuration of OpenFOAM simulation cases. SALOME is a useful tool that can aid the geometry and mesh generation process and HELYX-OS can be used to configure or set up simulations in OpenFOAM. In regards to post-processing, it has been argued that ParaView (an open source post-processing package compatible with OpenFOAM) will satisfy the need of most users. Finally, the predictive capabilities of OpenFOAM are showcased for three test cases: (i) a flat plate at zero pressure gradient, (ii) the one-dimensional flow through a porous medium, and (iii) the flow through a planar diffuser with downstream porosity. For all these cases, OpenFOAM is benchmarked against experimental results and shows a good agreement, therefore, demonstrating the potential of OpenFOAM as a design tool that can be readily incorporated into the design of light aircraft.

References

[1] J. Blazek. *Computational Fluid Dynamics: Principles and Applications: Third Edition*. 2015.

[2] S. Biringen and C.-Y. Chow. *An introduction to computational fluid mechanics by example*. 2011.

[3] Pierluigi Della Vecchia and Fabrizio Nicolosi. Aerodynamic guidelines in the design and optimization of new regional turboprop aircraft. *Aerospace Science and Technology*, 38:88 – 104, 2014.

[4] O.J. Boelens. CFD analysis of the flow around the X-31 aircraft at high angle of attack. *Aerospace Science and Technology*, 20(1):38 – 51, 2012.

[5] P. Panagiotou, P. Kaparos, and K. Yakinthos. Winglet design and optimization for a MALE UAV using CFD. *Aerospace Science and Technology*, 39:190 – 205, 2014.

[6] CFD Direct. OpenFOAM user guide. http://cfd.direct/openfoam/user-guide/, 2015. [Online; accessed 26-October-2015].

[7] F. Moukalled, L. Mangani, and M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics*. Springer International Publishing, 2015.

[8] CFD Online. Welcome to CFD Online, serving the CFD community since 1994. http://www.cfd-online.com/, 2015. [Online; accessed 26-October-2015].

[9] OpenFOAM Foundation. OpenFOAM – free open source CFD. http://www.openfoam.org/, 2015. [Online; accessed 20-October-2015].

[10] CFD Support. Home Page. http://www.cfdsupport.com/index.html, 2015. [Online; accessed 27-October-2015].

[11] J. Schoberl. Netgen Mesh Generator. http://sourceforge.net/projects/netgen-mesher/, 2015. [Online; accessed 22-October-2015].

[12] Open Cascade. SALOME. http://www.salome-platform.org/, 2015. [Online; accessed 27-October-2015].

[13] engys. HELYX-OS. http://engys.com/products/helyx-os, 2015. [Online; accessed 27-October-2015].

[14] N. Madhavan and H. Medina. Pulsed-engine powered mav: Airframe configurations and wing morphing with cross-winds. Proceedings of the Aerodynamics Conference 2010. Royal Aeronautical Society, July 2010.

[15] Kitware. ParaView. http://www.paraview.org/, 2015. [Online; accessed 27-October-2015].

[16] J. Coupland. ERCOFTAC special interest group on laminar to turbulent transition and retransition: T3A and T3B. Technical Report A309514, ERCOFTAC, 1990.

[17] D.C. Wilcox. Re-assessment of the scale-determining equation for advanced turbulence models. *AIAA Journal*, 26(11):1299–1310, 1988.

[18] B.E. Launder and B.I. Sharma. Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc. *Letters in Heat and Mass Transfer*, 1(2):131–138, 1974.

[19] D.K. Walters and D. Cokljat. A three-equation eddy-viscosity model for reynolds-averaged navier-stokes simulations of transitional flow. *Journal of Fluids Engineering*, 130(12):1–14, 2008.

[20] H. Medina and J. Early. Modelling transition due to backward-facing steps using the laminar kinetic energy concept. *European Journal of Mechanics B/Fluids*, 44:60–68, 2014.

[21] S.S. Quadri, S.F. Benjamin, and C.A. Roberts. An experimental investigation of oblique entry pressure losses in automotive catalytic converters. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 223(11):2561–2569, 2009.

[22] S. Porter, A. Mat Yamin, S. Aleksandrova, S. Benjamin, C. A. Roberts, and J. Saul. An assessment of CFD applied to steady flow in a planar diffuser upstream of an automotive catalyst monolith. *SAE Int. J. Engines*, 7(4):1697–1704, 2014.