



NATIONAL TECHNICAL UNIVERSITY OF ATHENS

POSTGRADUATE STUDIES PROGRAM

"COMPUTATIONAL MECHANICS"

MASTER'S THESIS

**Multiphase CFD Simulation of suction of water by the engine of cars
subjected to water drive through**

Andreas NATSIS

Supervisors:

Prof. Andreas G. BOUDOUVIS

School of Chemical Engineering, NTUA, Greece

Dr. Alexander RAUFEISEN

TWT GmbH, Munich, Germany

September 2014

Dedicated to my parents

Acknowledgements

I would like to thank Professor Andreas Boudouvis for his trust on me and motivation during all this time. I would like to express my admiration towards his scientific excellence, his management skills on the department and his willingness to help the students in various ways. I would also like to mention, that without his contribution, this postgraduate program would be less attractive.

I am sincerely grateful to Niki M.E.P.E. and TWT GmbH for their support and for giving me the opportunity to gain such a valuable experience and knowledge. The working environment they provide is ideal and meets all the needs of an ambitious engineer. More precisely I would like to thank:

-Dr. Alexander Raufeisen for his guidance and help. His skills in inspiring people combined with his scientific knowledge are unique.

-Mr. Felix Schuttack and Mr. Christoph Ottmann. Whoever works in the same team with these highly talented and friendly engineers can only feel blessed.

-Dr. Dimitrios Sikoutris. His knowledge, experience, expertise and excellent cooking skills form a perfect example of a multitalented friend.

Moreover, I would like to thank my family and friends for their unconditional love and support. They always make it difficult for me to reciprocate.

Last but not least I am thankful to Miss Karolina Skiba for colouring my anxiety and long working hours with beautiful shades.

Abstract

“Multiphase CFD Simulation of suction of water by the engine of cars subjected to water drive through”

The work presented in this Master's thesis deals with multiphase flow through complex geometry. STAR-CCM+ and openFoam, two powerful CFD codes, are being used to simulate the water-drive-through of a passenger car. The aim of the simulations is to investigate the formation of waves generated from the velocity of the car while driving through shallow water. More precisely the critical velocity, during which the water reaches the air-suction system of the engine, is defined. The main objective is to build a methodology which will be able to predict the critical velocity for any passenger car. Additional experimental results should be used in the future to calibrate the simulation.

Automotive companies perform experimental tests to define the critical velocity. Low critical velocity could result in redesigning the air-suction system or neighbor parts. CFD simulations cost less, can be performed in earlier stages of the design process and can evaluate the performance of design variants.

The flow is considered two-phase (water, air), incompressible, immiscible and turbulent. The governing equations are Reynolds Averaged Navier-Stokes with a $k - \varepsilon$ turbulence model. The interphase is captured by using the Volume of Fluids (VOF) method. Moreover there are regions which are modelled as porous media according to the Darcy-Forchheimer formulation.

Both STAR-CCM+ and openFoam are used for all the steps of the simulation (meshing-preprocessing, processing and post processing). A comparison between the software products is made, not only regarding the results but also regarding their efficiency on meeting the requirements of such a methodology.

The results obtained from the analysis are comparable and realistic. Both openFoam and STAR-CCM+ predicted almost the same critical velocity for the car being simulated. Moreover the predicted flow portraits of the two software products match previous experimental results. The main difference is spotted on the aspect of robustness, in favor of STAR-CCM+. Both methodologies need to be calibrated and thoroughly validated with detailed experimental measurements.

This work is part of the research that N.I.K.I. and TWT GmbH are performing on the area of CFD simulations. TWT and N.I.K.I are long term partners with pioneer automotive companies such as BMW, Audi and Daimler.

Περίληψη

"Πολυφασική προσομοίωση προσπέλασης οχημάτων σε πλημμυρισμένους δρόμους"

Αυτή η μεταπτυχιακή εργασία πραγματεύεται την προσπέλαση οχημάτων σε πλημμυρισμένους δρόμους. Στόχος των διενεργούμενων προσομοιώσεων είναι ο καθορισμός του ύψους των κυμάτων που δημιουργούνται και αν φτάνουν την εισαγωγή αέρα του κινητήρα που θα είχε ως αποτέλεσμα την καταστροφή του. Τα λιμνάζοντα νερά εκτείνονται σε ύψος 40 εκατοστών ενώ το αυτοκίνητο επιταχύνει από μηδενική ταχύτητα μέχρι τα 13 χιλιόμετρα ανά ώρα. Η επίλυση γίνεται με δύο κώδικες, το STAR-CCM+ και το openFoam. Σκοπός της παρούσας εργασίας είναι ο σχεδιασμός μεθοδολογίας που να προβλέπει την κρίσιμη εκείνη ταχύτητα του αυτοκινήτου, κατά την οποία δημιουργείται τέτοιο κύμα που το νερό φτάνει στην εισαγωγή του κινητήρα.

Οι αυτοκινητοβιομηχανίες ενδιαφέρονται για την εύρεση της κρίσιμης ταχύτητας, καθώς σε περίπτωση που αυτή είναι πολύ χαμηλή προβαίνουν σε ανασχεδιασμό του συστήματος εισαγωγής του αέρα. Χρησιμοποιώντας κώδικες CFD μπορούν να προβλέπουν την ταχύτητά αυτή σε πρότερα στάδια παραγωγής, να αξιολογούν εναλλακτικά σχέδια και να προχωρούν σε βελτιστοποίηση του σχεδιασμού, σε μικρότερους χρόνους και μικρότερο κόστος σε σύγκριση με πειραματικές μετρήσεις.

Η ροή είναι διφασική (αέρας – νερό), ασυμπίεστη, μη αναμίξιμη και τυρβώδης. Το πλέγμα είναι μη δομημένο εξαέδρο με κύριο χαρακτηριστικό την πολυπλοκότητα της γεωμετρίας. Οι εξισώσεις που επιλύονται είναι: η εξίσωση της συνέχειας, οι Reynolds Averaged εξισώσεις Navier-Stokes (RANS) με μοντέλο τύρβης $k - \epsilon$ και η εξίσωση συνέχειας των φάσεων σύμφωνα με το μοντέλο VOF. Ακόμα υπάρχουν περιοχές στο πλέγμα που έχουν μοντελοποιηθεί ως πορώδεις περιοχές σύμφωνα με την εξίσωση *Darcy-Forchheimer*.

Και οι δύο κώδικες χρησιμοποιήθηκαν σε όλες της φάσεις που απαιτεί μία τέτοια προσομοίωση (meshing-preprocessing, processing and post processing). Πέραν της σύγκρισης των αποτελεσμάτων, οι δύο κώδικες συγκρίνονται με κριτήρια όπως η ευκολία χρήσης, ο χρόνος επίλυσης, οι περιορισμοί και οι δυνατότητες που προσφέρουν

Τα αποτελέσματα είναι συγκρίσιμα και ρεαλιστικά. Αμφότεροι οι κώδικες προέβλεψαν σχετικά ίσες μεταξύ τους κρίσιμες ταχύτητες. Οι κύριες διαφορές εντοπίζονται κυρίως στην ταχύτητα επίλυσης και στην ευκολία χειρισμού, με το STAR-CCM+ να επιδεικνύει πληθώρα δυνατοτήτων και εύκολων στην χρήση εργαλείων. Η αξιολόγηση των αποτελεσμάτων απαιτεί σύγκριση με πειραματικά αποτελέσματα.

Η εργασία αυτή αποτελεί μέρος της προσπάθειας των εταιριών N.I.K.I. και TWT GmbH να αναπτύξουν την μεθοδολογία προσομοίωσης της προσπέλασης αυτοκινήτων σε λιμνάζοντα νερά. Οι εταιρίες αυτές προσφέρουν τις υπηρεσίες τους σε θέματα CFD σε κολοσσούς της αυτοκινητοβιομηχανίας όπως BMW, Audi και Daimler.

Contents

1	Introduction	1
1.1	Water drive-through	1
1.2	Multiphase flows	2
1.3	Laminar and turbulent flow	3
1.4	Flow classification by Froude number	5
1.5	Free Surface Modeling Methods	6
1.6	Aim	10
2	Mathematical Formulation.....	12
2.1	Introduction	12
2.2	Governing equations.....	12
2.2.1	Momentum and mass conservation.....	12
2.2.2	Boussinesq approach	13
2.2.3	$k - \varepsilon$ turbulence model	13
2.2.4	Solution Procedures for the Two-Fluid Model	16
2.3	SIMPLE Solver Algorithm	17
2.4	Piso Solver algorithm	18
2.5	VOF	20
3	Computational Setup	23
3.1	Geometrical Setup and Mesh Generation in STAR-CCM+	24
3.2	Geometrical setup and mesh generation in openFoam.....	28
3.2.1	blockMesh	29
3.2.2	snappyHexMesh	29
3.2.3	checkMesh	30
3.2.4	Deleting bad cells.....	30
3.2.5	Decomposing.....	32
3.2.6	renumberMesh.....	32
3.3	Motion of the car	33
3.4	Boundary conditions	34
3.5	Initialization.....	35
3.6	Porous medium	36

3.7	Time discretization.....	37
3.8	Fluid properties	38
3.9	Gravity	38
3.10	Discretization schemes in openFoam	39
3.10.1	Time step and data output control	43
4	Results.....	44
4.1	Results with STAR-CCM+	45
4.2	Results with openFoam.....	48
4.3	Comparison	52
4.3.1	Critical velocity	53
4.3.2	Wave formation.....	53
4.3.3	Velocities.....	54
4.3.4	Volume fraction	55
5	Conclusions	57
5.1.1	Geometry handling	57
5.1.2	Mesh generation.....	57
5.1.1	Simulation setup.....	57
5.1.2	Processing	58
5.1.3	Post-processing.....	58
5.1.4	Results.....	58
5.2	Future work.....	59
	References.....	60

List of Figures

Figure 1.1 Water drive through of cars crossing a high-way	2
Figure 1.2 The positioning of the air intake system for a conventional car.....	2
Figure 1.3 Laminar Couette flow.....	4
Figure 1.4 a) unsteady flow observed from a stationary point of view. b) steady flow observed from a moving point of view.....	6
Figure 2.1 A typical CV and the notation used for Cartesian 2D grid	16
Figure 2.2 Flow chart of the SIMPLE algorithm.....	18
Figure 2.3 Flow chart of the PISO algorithm.....	19
Figure 2.4 Illustration of grids that are suitable (right) and unsuitable (left) for two phase flows using VOF model adapted from CD-adapco (2010).....	20
Figure 2.5 a) True interface b) Volume Fraction	22
Figure 3.1 The available external geometry of Aletis	23
Figure 3.2 The final geometry of Aletis used for the simulations.....	23
Figure 3.3 Geometry of the Wind Canal	24
Figure 3.4 Position of the suction system (red colour)	25
Figure 3.5 Position of the suction system (side view).....	25
Figure 3.6 Air-suction system.....	26
Figure 3.7 y-cut of the mesh ($y=0$)	26
Figure 3.8 z-cut of the mesh ($z=0.1$)	27
Figure 3.9 Close-up view of the mesh near the front of the car (y-cut)	27
Figure 3.10 Close-up view of the mesh near the front of the car (x-cut)	28
Figure 3.11 Close-up view of the mesh near the front of the car (y-cut)	30
Figure 3.12 a) original matrix b) reordered matrix according to Cuthill–McKee algorithm	33
Figure 3.13 Air filter of a 3-series BMW car.....	36
Figure 4.1 Clean air surface and dirty air surface of the air suction system.....	44
Figure 4.2 Surface averaged volume fraction of water in the entrance of the air-intake system.....	45
Figure 4.3 Iso-surface volume of fraction =0.5	46
Figure 4.4 Iso-surface volume of fraction =0.5 (view under the hood)	46
Figure 4.5 Volume fraction of Water in different velocities	46
Figure 4.6 Velocity magnitude distribution in a y-cut plane.....	47
Figure 4.7 Pressure distribution in a y-cut plane	47
Figure 4.8 Convective Courant number	48

List of Tables

Table 3.1 Interpolation schemes.....	40
Table 3.2 Behavior of interpolation schemes used in <i>divSchemes</i>	41
Table 3.3 Discretization schemes available in <i>gradSchemes</i>	41
Table 3.4 Behavior of surface normal schemes used in <i>laplacianSchemes</i>	42
Table 3.5 Surface normal gradient schemes.....	43
Table 4.1 Top view of iso-surfaces ($\alpha=0.5$) in different time steps.....	49
Table 4.2 Side view of the scalar field Volume Fraction of Water (y-cut, $y=0$).....	50
Table 4.3 Close side view of the isosurface ($\alpha = 0.5$).....	51
Table 4.4 Side view of velocity (Magnitude) distribution in different time steps.....	52
Table 4.5 Wave formation comparison between STAR-CCM+ and openFoam.....	54
Table 4.6 Comparison of velocity distribution (y-cut, $y=0$).....	55
Table 4.7 Comparison of volume fraction of water (y-cut, $y=0$).....	56

Nomenclature

Notation	Meaning
----------	---------

Fr	Froude number
------	---------------

V	Control volume
-----	----------------

g	Gravitational acceleration
-----	----------------------------

L	Length
-----	--------

Re	Reynolds number
------	-----------------

R	Hydraulic radius
-----	------------------

Ma	Mach number
------	-------------

x	Component of the three dimensional Cartesian coordinate system
-----	--

y	Component of the three dimensional Cartesian coordinate system
-----	--

z	Component of the three dimensional Cartesian coordinate system
-----	--

u	Component of the velocity vector in the x direction.
-----	--

v	Component of the velocity vector in the y direction.
-----	--

w	Component of the velocity vector in the z direction.
-----	--

$H(x, y, z)$	Height of the surface of the water
--------------	------------------------------------

t	time
-----	------

F	Fraction of fluid.
-----	--------------------

\mathbf{v}	Fluid velocity vector
--------------	-----------------------

S	Surface of the control volume
-----	-------------------------------

T	Viscous stress tensor
-----	-----------------------

p	Pressure
-----	----------

\mathbf{b}	Vector of all body forces applied on the control volume
--------------	---

\mathbf{I}	Identity matrix
--------------	-----------------

Greek letters

Notation	Meaning
----------	---------

ρ	Density
μ_{eff}	Effective dynamic viscosity of the fluid
μ_i	Turbulent viscosity
μ_t	Eddy viscosity
ν_t	Kinematic eddy viscosity
σ_k	Turbulent Prandtl number for k
σ_ϵ	Turbulent Prandtl number for ϵ
α_i	Volume fraction of the i^{th} phase, $\alpha_i = \frac{V_i}{V}$.
δ_{ij}	Kronecker delta
ϵ	Dissipation rate
Ω_{ij}	Rotation tensor

Abbreviations

Notation	Meaning
----------	---------

CFD	Computational Fluid Dynamics
CV	Control Volume
RANS	Reynolds Averaged Navier-Stokes
BC	Boundary Condition
VOF	Volume of fluid
VFW	Volume fraction of water

1 Introduction

Automotive industry struggles to design faster, stronger and more luxurious vehicles. On top of that their products should be able to overcome all environmental conditions. One of the adverse conditions a car can be subjected to is the water drive-through.

After heavy rain many roads end up full of water. The depth of the water is usually not deep enough for a car to drive through. The problem is that cars, while accelerating, they form waves which result, sometimes, in covering a big percentage of the car. The most sensitive part of the car is the air-intake system. If water reaches this area it is almost definite that the water will end up in the engine, causing a permanent damage.

It is very important for the automotive industry to define the critical velocity of the car, over which the formation of waves can end up in water suction from the engine. In order to give an answer to that, the water drive through is going to be simulated with two different software products (STAR-CCM+ and openFoam) and the results are going to be compared.

1.1 Water drive-through

Water drive through, is named the situation during which cars come across most frequently flood water. Even if the depth of the water is not high, the speed of the car can produce a bow wave which will result in the rise of the surface of the water. This can have a catastrophic impact on the car's engine. The engine's air intake on many cars is low down at the front of the car and it can take just an egg cupful of water in the combustion chamber to wreck an engine. Water doesn't compress and the piston in effect hits a wall, bending or breaking a con rod. In Figure 1.1 a flooded high-way with cars trying to cross it, is being illustrated. In this photo the wave in front of the car, which is caused because of its movement, can be observed.



Figure 1.1 Water drive through of cars crossing a high-way



Figure 1.2 The positioning of the air intake system for a conventional car

The figure 1.2 shows the position of the complete air suction system in a conventional car. The opening of air-inlet duct is located just behind the kidney grill. Due to this positioning of the air-inlet duct, the high water wave fills the inlet ducts and the air suction system very easily when a wave passes through.

1.2 Multiphase flows

Multiphase flow is a simultaneous flow of materials with different states or phases (i.e. gas, liquid or solid) or materials with different chemical properties but in the same state or phase (i.e. liquid - liquid systems such as oil droplets in water).

A particular important category of multiphase flows from an engineering prospective is two phase flows. In fluid mechanics, two-phase flow occurs in a system containing gas and liquid with an interphase separating the two phases.

The term "two-phase flow" covers an extremely wide range of flow patterns and regimes. It is useful to subdivide these into a small number of classes, to give a first impression of the physical processes involved. Two-phase flows are often broadly categorized by the physical states of the components and by the topology of the interfaces. Thus, a two-phase flow can be classified as gas-solid, gas-liquid, solid-liquid, or in the case of two immiscible liquids, liquid-liquid. Similarly, a flow can be broadly classified topologically as separated, dispersed or transitional (Bergles et. al. (1981), Hetstroni (1982), Ishii (1975)). This variety of combinations makes the design of industrial equipment for two-phase flow applications a very difficult task.

Since in the present thesis work, the interaction between water and air is going to be evaluated, only the gas-liquid flows are to be discussed in detail.

The development of a methodology that predicts in considerable detail and with sufficient accuracy the entire flow field of a flow and, in particular, two-phase flow is highly desirable. Such a methodology exist in form of Computational Fluid Dynamics (CFD). CFD is the analysis of engineering systems involving fluid flow, heat transfer and associated phenomena, such as two-phase flow, by means of computer-based simulation.

The development of CFD is part of a general trend in reducing expensive experimental studies and empirical correlations to more generally applicable and accurate mathematical models of engineering systems. Examples include computational solid mechanics (Kleiber (1998), Fung (2001)) and molecular dynamics simulations (Keil et. al. (1999)). Such methodologies can be used with increasing confidence in areas outside those studied experimentally. Moreover, a far broader in-depth insight into the process can be gained. Overviews of application of CFD in the process and associated industries are given in (Gosman (1983), Casey et. al. (1988), Trambouze (1996), Kuipers and Swaij (1998), Hjertager (1998), Keil et. al. (1999)). It should be noted that CFD and other numerical methods are still under development in many ways and will continue to be so for the foreseeable future. However, at their present state they go a long way towards meeting some of the requirements posed upon them. In addition, experimental and empirical methods are still being developed further. All of these methods are complimentary and benefit each other.

1.3 Laminar and turbulent flow

The velocity field in a drag-induced Couette flow between parallel flat plates remains constant on plane surfaces which are parallel to the flat plates. Individual plane laminae are sheared and thus slide over each other, the velocity being purely axial everywhere. Thus an element of fluid with a rectangular section at time $t = 0$ in a plane spanned by the direction of flow and the direction of shear is distorted into a parallelogram at time $t > 0$ (see Figure 1.3). Similarly, the velocity field in a pressure-induced Poiseuille Flow in a long pipe of circular cross-section normal to its axis remains constant on cylindrical surfaces which are concentric to the axis of the pipe. Individual cylindrical laminae slide over each other, the velocity again being purely axial everywhere. Smooth flows of such kinds are called laminar flows [see Richardson (1989), Rosenhead (1963), Shapiro (1964) and White (1974)] or, in the older literature, streamline flows and generally occur at low values of an appropriate dimensionless number, usually a Reynolds Number. It is to be distinguished from a rough or erratic Turbulent Flow, which is a high Reynolds number flow.

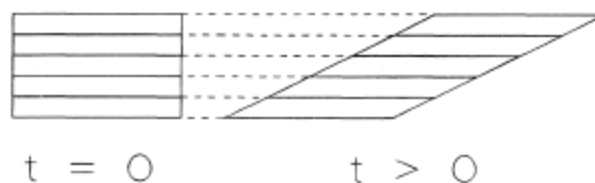


Figure 1.3 Laminar Couette flow

The origin of turbulence appears to be instability of the associated low Reynolds number laminar flow. Indeed, all laminar flows seem to be susceptible to instabilities leading eventually to turbulence (see Drazin & Reid (1981) for a discussion of the instability of several basic laminar flows).

In a turbulent flow, flow variables such as velocity and pressure fluctuate in space and time in an apparently random manner. No such fluctuations are observed in a laminar flow, though it should be noted that a laminar flow need not be steady, nor necessarily very smooth.

The details of the transition from laminar to turbulent flow are not well understood. For flow by a wall, however, the main steps as the Reynolds number is increased appear to comprise:

- an initial, often two-dimensional, instability which leads to:
- a Secondary Flow which is generally three-dimensional and itself unstable which leads to:
- a succession of increasingly complex secondary flows which are again themselves unstable and which lead to:
- further, almost without exception three-dimensional, flows which are themselves unstable, and so on, until:
- intense, local, three-dimensional fluctuations are produced which grow both in size and in number, merge and eventually:
- the flow becomes fully turbulent.

The exact point at which the flow ceases to be laminar is debatable and probably irrelevant.

It is important to note that an increase in Reynolds number occurs not only through an increase in speed. It also occurs through an increase in dimension or size. Thus, laminar flows tend to involve relatively slow motions of small objects. Turbulent flows, in contrast, tend to involve relatively fast motions of large objects. Some flows, particularly those associated with Boundary Layers, Jets and wakes, are laminar upstream and turbulent downstream. This is because the characteristic dimension involved in the Reynolds number is then the axial distance from the start of the flow. A typical example of this is smoke from a lighted cigarette. Near the cigarette, there is a smooth column of smoke: the flow is laminar. Further away, the column breaks down: the flow is turbulent.

The flow in an open-channel may be either laminar or turbulent, as all flows. The criterion for determining the type of flow is the Reynolds number, Re . For open-channel flow,

$$Re = \frac{\rho u R}{\mu} \quad (1.1)$$

Where u is the velocity of the flow and R is the hydraulic radius $R=A/P$. The analysis is not developed here but the open-channel limits for each type of flow become

Laminar: $Re < 500$

Turbulent: $Re > 1000$

In practice the limit for turbulent flow is not as well defined in channel as it is in pipes and so 2000 is often taken as the threshold for turbulent flow.

1.4 Flow classification by Froude number

Another very useful classification of the flow is by the dimensionless Froude number Fr which is the ratio of channel velocity to the speed of propagation of a small disturbance wave in the channel

$$Fr = \frac{V}{\sqrt{gL}} \quad (1.2)$$

where L is the water depth. The flow behaves differently depending on these three flow regimes:

- $Fr < 1.0$ Subcritical flow

Wave velocity > water velocity

Upstream levels affected by downstream controls

- $Fr = 1.0$ Critical flow
- $Fr > 1.0$ Supercritical flow

Wave velocity < water velocity

Upstream levels not affected by downstream controls

There is here a strong analogy with the three compressible flow regimes of the Mach number: subsonic ($Ma < 1$), sonic ($Ma = 1$) and supersonic ($Ma > 1$).

The Froude number is very useful when analyzing open-channel flow, due to the importance of both gravitational and inertia forces.

The Froude number is also known to be the ratio between the flow speed and the wave speed. The pressure in an open-channel is constant along the free surface, so if the flow is disturbed, a surface wave will appear, and not a pressure wave as shown in Figure 1.4. [4]

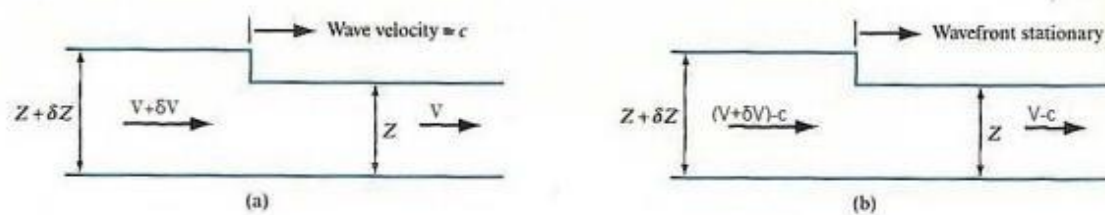


Figure 1.4 a) unsteady flow observed from a stationary point of view. b) steady flow observed from a moving point of view.

1.5 Free Surface Modeling Methods

An interface between a gas and liquid is often referred to as a free surface. The reason for the “free” designation arises from the large difference in the densities of the gas and liquid (e.g., the ratio of density for water to air is 1000). A low gas density means that its inertia can generally be ignored compared to that of the liquid. In this sense the liquid moves independently, or freely, with respect to the gas. The only influence of the gas is the pressure it exerts on the liquid surface. In other words, the gas-liquid surface is not constrained, but free.

Whatever the name, it should be obvious that the presence of a free or moving boundary introduces serious complications for any type of analysis. For all but the simplest of problems, it is necessary to resort to numerical solutions. Even then, free surfaces require the introduction of special methods to define their location, their movement, and their influence on a flow.

In the following discussion we will briefly review the types of numerical approaches that have been used to model free surfaces, indicating the advantages and disadvantages of each method. Regardless of the method employed, there are three essential features needed to properly model free surfaces:

1. A scheme is needed to describe the shape and location of a surface,
2. An algorithm is required to evolve the shape and location with time, and
3. Free-surface boundary conditions must be applied at the surface.

Lagrangian Grid Methods

Conceptually, the simplest means of defining and tracking a free surface is to construct a Lagrangian grid that is imbedded in and moves with the fluid. Many finite-element methods use this approach. Because the grid and fluid move together, the grid automatically tracks free surfaces.

At a surface it is necessary to modify the approximating equations to include the proper boundary conditions and to account for the fact that fluid exists only on one side of the boundary. If this is not done, asymmetries develop that eventually destroy the accuracy of a simulation.

The principal limitation of Lagrangian methods is that they cannot track surfaces that break apart or intersect. Even large amplitude surface motions can be difficult to track without introducing regridding techniques such as the Arbitrary-Lagrangian-Eulerian (ALE) method. The papers of Hirt, Cook and Butler (1970) and Hirt, Amsden and Cook (1974) can be consulted for early examples of these approaches.

The remaining free-surface methods discussed here use a fixed, Eulerian grid as the basis for computations so that more complicated surface motions may be treated.

Surface Height Method

Low amplitude sloshing, shallow water waves, and other free-surface motions in which the surface does not deviate too far from horizontal, can be described by the height, H , of the surface relative to some reference elevation. Time evolution of the height is governed by the kinematic equation. Equation (1.3) is a mathematical expression of the fact that the surface must move with the fluid:

$$\frac{\partial H}{\partial t} + u \frac{\partial H}{\partial x} + v \frac{\partial H}{\partial y} = w \quad (1.3)$$

where $H(x, y, z)$ is the height of the surface and (u, v, w) are the fluid velocities in the (x, y, z) directions.

Finite-difference approximations to this equation are easy to implement. Further, only the height values at a set of horizontal locations must be recorded so the memory requirements for a three-dimensional numerical solution are extremely small. Finally, the application of free-surface boundary conditions is also simplified by the condition on the surface that it remains nearly horizontal. Examples of this technique can be found in Nichols and Hirt (1971, 1975).

Marker-and-Cell (MAC) Method

The earliest numerical method devised for time-dependent, free-surface, and flow problems was the Marker-and-Cell (MAC) method, Harlow and Welch (1965). This scheme is based on a fixed, Eulerian grid of control volumes. The location of fluid within the grid is determined by a set of marker particles that move with the fluid, but otherwise have no volume, mass or other properties.

Grid cells containing markers are considered occupied by fluid, while those without markers are empty (or void). A free surface is defined to exist in any grid cell that contains particles and that also has at least one neighboring grid cell that is void. The location and orientation of the surface within the cell was not part of the original MAC method.

Evolution of surfaces was computed by moving the markers with locally interpolated fluid velocities. Some special treatments were required to define the fluid properties in newly filled grid cells and to cancel values in cells that are emptied.

The application of free-surface boundary conditions consisted of assigning the gas pressure to all surface cells. Also, velocity components were assigned to all locations on or immediately outside the surface in such a way as to approximate conditions of incompressibility and zero-surface shear stress.

The extraordinary success of the MAC method in solving a wide range of complicated free-surface flow problems is well documented in numerous publications. One reason for this success is that the markers do not track surfaces directly, but instead track fluid volumes. Surfaces are simply the boundaries of the volumes, and in this sense surfaces may appear, merge or disappear as volumes break apart or coalesce.

A variety of improvements have contributed to an increase in the accuracy and applicability of the original MAC method. For example, applying gas pressures at interpolated surface locations within cells improves the accuracy in problems driven by hydrostatic forces, while the inclusion of surface tension forces extends the method to a wider class of problems, Daly (1969) and Nichols and Hirt (1975).

In spite of its successes, the MAC method has been used primarily for two-dimensional simulations because it requires considerable memory and CPU time to accommodate the necessary number of marker particles. Typically, an average of about 16 markers in each grid cell is needed to ensure an accurate tracking of surfaces undergoing large deformations.

Another limitation of marker particles is that they don't do a very good job of following flow processes in regions involving converging/diverging flows. Markers are usually interpreted as tracking the centroids of small fluid elements. However, when those fluid elements get pulled into long convoluted strands, the markers may no longer be good indicators of the fluid configuration. This can be seen, for example, at flow stagnation points where markers pile up in one direction, but are drawn apart in a perpendicular direction. If they are pulled apart enough (i.e., further than one grid cell width) unphysical voids may develop in the flow.

Surface Marker Method

One way to limit the memory and CPU time consumption of markers is to keep marker particles only on surfaces and not in the interior of fluid regions. Of course, this removes the volume tracking property of the MAC method and requires additional logic to determine when and how surfaces break apart or coalesce.

In two dimensions the marker particles on a surface can be arranged in a linear order along the surface. This arrangement introduces several advantages, such as being able to maintain a uniform particle spacing and simplifying the computation of intersections between different surfaces. Surface markers also provide a convenient way to locate the surface within a grid cell for the application of boundary conditions.

Unfortunately, in three-dimensions there is no simple way to order particles on surfaces, and this leads to a major failing of the surface marker technique. Regions may exist where surfaces are expanding and no markers fill the space. Without markers the configuration of the surface is unknown, consequently there is no way to add markers, Nichols and Hirt (1975).

Volume-of-Fluid (VOF) Method

The last method to be discussed is based on the concept of a fluid volume fraction. The idea for this approach originated as a way to have the powerful volume-tracking feature of the MAC method without its large memory and CPU costs.

Within each grid cell (control volume) it is customary to retain only one value for each flow quantity (e.g., pressure, velocity, temperature, etc.) For this reason it makes little sense to retain more information for locating a free surface. Consequently the use of a single quantity, the fluid volume fraction in each grid cell, is consistent with the resolution of the other flow quantities.

If we know the amount of fluid in each cell it is possible to locate surfaces, as well as determine surface slopes and surface curvatures. Surfaces are easy to locate because they lie in cells partially filled with fluid or between cells full of fluid and cells that have no fluid.

Slopes and curvatures are computed by using the fluid volume fractions in neighboring cells. It is essential to remember that the volume fraction should be a step function, i.e., having a value of either one or zero. Knowing this, the volume fractions in neighboring cells can then be used to locate the position of fluid (and its slope and curvature) within a particular cell.

Free-surface boundary conditions must be applied as in the MAC method, i.e., assigning the proper gas pressure (plus equivalent surface tension pressure) as well as determining what velocity components outside the surface should be used to satisfy a zero shear-stress condition at the surface. In practice, it is sometimes simpler to assign velocity gradients instead of velocity components at surfaces.

Finally, to compute the time evolution of surfaces, a technique is needed to move volume fractions through a grid in such a way that the step-function nature of the distribution is retained. The basic kinematic equation for fluid fractions (equation (1.4)) is similar to that for the height-function method (equation (1.3)), where F is the fraction of fluid function:

$$\frac{\partial F}{\partial t} + u \frac{\partial F}{\partial x} + v \frac{\partial F}{\partial y} + w \frac{\partial F}{\partial z} = 0 \quad (1.4)$$

A straightforward numerical approximation cannot be used to model this equation because numerical diffusion and dispersion errors destroy the sharp, step-function nature of the F distribution.

It is easy to accurately model the solution to this equation in one dimension such that the F distribution retains its zero or one values. Imagine fluid is filling a column of cells from bottom to top. At some instant the fluid interface is in the middle region of a cell whose neighbor below is filled and whose neighbor above is empty. The fluid orientation in the neighboring cells means the interface must be located above the bottom of the cell by an amount equal to the fluid fraction in the cell. Then the computation of how much fluid to move into the empty cell above can be modified to first allow the empty region of the surface-containing cell to fill before transmitting fluid on to the next cell.

In two or three dimensions a similar procedure of using information from neighboring cells can be used, but it is not possible to be as accurate as in the one-dimensional case. The problem with more than one dimension is that an exact determination of the shape and location of the surface cannot be made. Nevertheless, this technique can be made to work well as evidenced by the large number of successful applications that have been completed using the VOF method, Nichols and Hirt (1975), (1980) and (1981).

The VOF method has lived up to its goal of providing a method that is as powerful as the MAC method without the overhead of that method. Its use of volume tracking as opposed to surface-tracking function means that it is robust enough to handle the breakup and coalescence of fluid masses. Further, because it uses a continuous function it does not suffer from the lack of divisibility that discrete particles exhibit.

1.6 Aim

In this work two software products, STAR-CCM+ and openFoam, are being used in order for the water drive-through of a passenger car to be simulated. The main aim of the present work is to investigate whether it is possible to perform such a simulation in openFoam and create a methodology able to tackle all the possible issues. openFoam is a powerful tool but it has never been used for such a simulation. The simulation is already set up in STAR-CCM+ and several cars have been simulated. The setup of the simulation in STAR-CCM+ and its results are going to be used as a guide and validation data.

The boundary specification in openFoam is not a trivial issue as there are not specific boundary conditions proposed for such a simulation. The cooling modules and the filter in the air-suction system have to be simulated as porous media. The generation of the mesh is a difficult task, because of the geometrical complexity and the generation of artificial waves. The post-processing in so big simulations is always a challenge. In STAR-CCM+ the procedure is almost automated. Moreover the

results are being obtained while the simulation is running, resulting in a huge save of memory in the disc. Similar functionality should be achieved also in openFoam.

In the first chapter an introduction is being made in order for the reader to familiarize himself with the problem. In the second chapter the governing equations are being presented and an explanation of the Volume of Fluid (VOF) model is being attempted. The third chapter describes the computational setup of the problem both in STAR-CCM+ and openFoam. The differences between the two software products are being discussed and highlighted. In the fourth chapter the results are being presented. The results of the two software products are being compared, while the most important criterion is the height of the wave in front of the car. The last chapter contains the conclusions and some proposals for future work.

2 Mathematical Formulation

2.1 Introduction

In the present thesis commercial CFD software STAR-CCM+ and open source software openFoam are being used. Below some principals based on which these software products work are briefly explained. First, basic flow equations applicable to the problem are explained. Second the VOF model is interpreted. At the end of this chapter, the SIMPLE and the PISO algorithms are being presented. More information about these techniques can be found in large body of work by Demirdzic et al. (1993) and Demirdzic and Musferija (1995) and comprehensive manual that comes with STAR-CCM+ and the book by Ferziger et al.(2002). Regarding openFoam more information can be found in www.openfoam.org and regarding interFoam in the thesis of Damian (2009)

2.2 Governing equations

2.2.1 Momentum and mass conservation

In this section, the basic flow equations are presented. Basic flow equation for this problem are integral form of Navier Stokes equations which include continuity and momentum equations. The continuity equation is a statement of mass conservation (equation (2.1)). The momentum equation arises from applying Newton's second law to fluid motion, together with the assumption that the fluid stress is the sum of a diffusing viscous term (proportional to the gradient of velocity) and a pressure term.

$$\frac{d}{dt} \int_V \rho v dV + \int_S \rho (\mathbf{v} - \mathbf{v}_b) \cdot d\mathbf{a} = 0 \quad (2.1)$$

$$\frac{d}{dt} \int_V \rho \mathbf{v} dV + \int_S \rho \mathbf{v} \otimes (\mathbf{v} - \mathbf{v}_b) \cdot d\mathbf{a} = \int_S (\mathbf{T} - p\mathbf{I}) \cdot d\mathbf{a} + \int_V \rho \mathbf{b} dV \quad (2.2)$$

In these equations ρ is the fluid density, V is the control volume bounded by closed surface S , \mathbf{v} is fluid velocity vector, \mathbf{v}_b is the velocity of CV surface, t is time, \mathbf{T} is viscous stress tensor and \mathbf{b} is vector of all body forces. Viscous stress tensor is defined as:

$$\mathbf{T} = \mu_{eff} [\nabla \mathbf{v} + \nabla \mathbf{v}^T - \frac{2}{3} (\nabla \cdot \mathbf{v}) \mathbf{I}] \quad (2.3)$$

Where μ_{eff} is effective dynamic viscosity of the fluid, which represents the sum of laminar and turbulent viscosities. The Above equations are solved using Segregated Flow Model available in STAR-

CCM+ software and openFoam. The Segregated Flow model solves the flow equations (one for each component of velocity, and one for pressure) in a segregated, or uncoupled, manner. The linkage between the momentum and continuity equations is achieved via a predictor-corrector approach which includes a collocated variable arrangement and a Rhie-and-Chow-type pressure-velocity coupling combined with a SIMPLE-type algorithm and a PISO algorithm in openFoam. In STAR-CCM+ software the Segregated Flow solver contains two other solvers: velocity solver and pressure solver. The velocity solver solves the discretized momentum equation to obtain the intermediate velocity field. The pressure solver solves the discrete equation for pressure correction, and updates the pressure field (CD-adapco 2010).

In study of wave motion and for many engineering applications, viscous effects of the fluid are negligible in the most part of the medium, due to the large dimensions of the structures and intensity of the motion of the water particles. This can be used, together with the assumption of irrotational flow, to simplify the problem of defining a wave theory to some extent as it allows the use of velocity potentials in the fundamental differential equations which reduces the number of fluid unknowns at the expense of increasing the order of the equations (potential flow problem) (Gerhart et al. (1992)). Generally this turns the continuity equation to a Laplacian of the velocity potential that should be solved under the imposition of linear and nonlinear boundary conditions. This is an approach that is not being followed in the current work.

2.2.2 Boussinesq approach

The Reynolds averaged approach to turbulence modeling requires that the Reynolds stresses are appropriately modeled (Hinze (1975)). A common method employs the Boussinesq hypothesis to relate the Reynolds stresses to the mean velocity gradient:

$$-\overline{p u_i' u_j'} = \mu_i \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \left(p k + \mu_i \frac{\partial u_k}{\partial x_k} \right) \delta_{ij} \quad (2.4)$$

The Boussinesq hypothesis is used in the $k - \varepsilon$ turbulence model. The advantage of this model is relatively low computational cost associated with the computation of turbulent viscosity μ_i .

2.2.3 $k - \varepsilon$ turbulence model

Two-equation turbulence models allow the determination of both, a turbulent length and time scale by solving two separate transport equations. The standard $k - \varepsilon$ model is quite widely used in industries. Robustness, economy and reasonable accuracy for a wide range of turbulent flows explain its popularity in industrial flow and heat transfer simulation (Ansys fluent theory guide 2011).

The standard $k - \varepsilon$ model is a model based on model transport equations for the turbulence kinetic energy (k) and its dissipation rate (ε). In the derivation of the $k - \varepsilon$ model, the assumption is that the flow is fully turbulent, and the effects of molecular viscosity are negligible.

The realizable $k - \varepsilon$ model differs from the standard $k - \varepsilon$ model in two ways: the realizable $k - \varepsilon$ model contains an alternative formulation for the turbulent viscosity and a modified transport equation for the dissipation rate ε , has been derived from an exact equation for the transport of the mean-square vorticity fluctuation. The term "realizable" means that the model satisfies certain mathematical constraints on the Reynold stresses, consistent with the physics of turbulent flows (Shabbir et al. (1995)).

The modeled transport equations for k and ε in the realizable model are:

$$\frac{\partial}{\partial t}(pk) + \frac{\partial}{\partial x_j}(pk u_j) = \frac{\partial}{\partial x_j} \left[\left(\mu + \frac{\mu_t}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right] + G_k + G_b - p\varepsilon - Y_M + S_k \quad (2.5)$$

$$\begin{aligned} \frac{\partial}{\partial t}(p\varepsilon) + \frac{\partial}{\partial x_j}(p\varepsilon u_j) &= \frac{\partial}{\partial x_j} \left[\left(\mu + \frac{\mu_t}{\sigma_\varepsilon} \right) \frac{\partial \varepsilon}{\partial x_j} \right] + \rho C_1 S_\varepsilon - \rho C_2 \frac{\varepsilon^2}{k + \sqrt{\nu \varepsilon}} \\ &+ C_{1\varepsilon} \frac{\varepsilon}{k} C_{2\varepsilon} G_b + S_\varepsilon \end{aligned} \quad (2.6)$$

Where:

$$C_1 = \max \left[0.43, \frac{\eta}{\eta + 5} \right], \eta = S \frac{k}{\varepsilon}, S = \sqrt{2S_{ij}S_{ij}} \quad (2.7)$$

The eddy viscosity is computed by

$$\mu_t = \rho C_\mu \frac{k^2}{\varepsilon} \quad (2.8)$$

In the above equations, S_k and S_ε are user defined source terms, G_k and G_b denote the generation of turbulence kinetic energy due to mean velocity gradient and buoyancy. Y_M denotes the contribution of fluctuating dilatation in compressible turbulence to the dissipation rate. σ_k and σ_ε represent turbulent Prandtl number for k and ε .

The other variables are calculated as

$$C_\mu = \frac{1}{A_0 + A_s \frac{kU^*}{\epsilon}} \quad (2.9)$$

$$U^* = \sqrt{S_{ij}S_{ij} + \check{\Omega}_{ij}\check{\Omega}_{ij}} \quad (2.10)$$

And

$$\check{\Omega}_{ij} = \Omega_{ij} - 2\epsilon_{ijk}\omega_k \quad (2.11)$$

$$\bar{\Omega}_{ij} = \check{\Omega}_{ij} - 2\epsilon_{ijk}\omega_k \quad (2.12)$$

Where $\bar{\Omega}_{ij}$ is the mean rate of rotation tensor viewed from the reference frame. The model constants A_0 and A_s are given by:

$$A_0 = 4.04, A_s = \sqrt{6}\cos\varphi \quad (2.13)$$

The model constants are given by:

$$C_{1\epsilon} = 1.44, \quad C_2 = 1.9, \quad \sigma_k = 1.0, \quad \sigma_\epsilon = 1.2 \quad (2.14)$$

The equations in the previous section are discretized according to Finite Volume method (FVM). In Finite Volume Method the solution domain is subdivided into a finite number of small cells called control volumes (CVs). Usually CVs are defined by a suitable grid and computational node is assigned to the CV center. All variations of FVM share the same discretization principals. They are different in relations between various locations within integration volume. The integral form of Navier Stokes equations are applied to each CV, as well as the solution domain as a whole. Summing all the equations for all CVs we obtain global conservation equation since surface integrals over inner CV faces cancel out. The final result is a set of linear algebraic equations with the total number of unknowns equal to the number of cells in the grid. Figure 2.1 show a typical 2D Cartesian control volume.

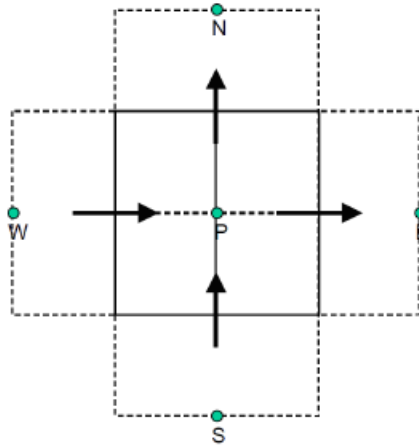


Figure 2.1 A typical CV and the notation used for Cartesian 2D grid

For both surface and volume integrals, it is most convenient to use midpoint rule approximations because they result in simple algebraic expressions that are second order accurate. Since the values at center of CVs are calculated in each time step this means to simply multiply the CV-center value by the CV-volume V . For the calculation of surface integral, further approximations are necessary since variables are not known at cell-face centers.

2.2.4 Solution Procedures for the Two-Fluid Model

Various solution procedures have been proposed to solve the set of coupled differential equations arising in the two-fluid models. Most of them use segregated approaches, in which the set of equations is solved sequentially. Some authors have proposed other solution techniques which utilize Riemann (Staedtke et al. (1998)) or partial block solution (Lathouwers and Van den Akker (1996), Lathouwers (1999)) techniques to handle the coupling between the equations in a more implicit manner. These techniques have not been used in this thesis, since they are often very demanding in terms of computational time, memory requirements and code complexity. Focusing on segregated approaches for two-phase flows, a number of different methodologies can be identified in the literature, e.g. IPSA (Spalding (1983)), BRITE (Politis (1989), Hill (1998)), ASTRID [376], two-phase ICE (Kashiwa et al. (1994)) and two-phase SOLA [Hirano and Tomiyama (1994)]. Among these there is little agreement concerning either the form of the equations to be solved or the solution procedure. However, certain similarities exist, for example, the use of the continuity equation in order to derive an equation for the pressure. The pressure change is then utilized to correct the velocities.

2.3 SIMPLE Solver Algorithm

SIMPLE is an acronym for Semi-Implicit Method for Pressure Linked Equations. SIMPLE algorithm is a widely used iterative numerical procedure to solve the Navier-Stokes equations. This method forms the basics of many commercial CFD packages. Application of SIMPLE algorithm to Navier Stokes equations in STAR-CCM+ includes the following steps (CD-adapco 2010):

1. Set the boundary conditions.
2. Compute reconstruction gradient of velocity and pressure.
3. Compute velocity and pressure gradients.
4. Solve the discretized momentum equation to create the intermediate velocity field v^* .
5. Compute uncorrected mass fluxes at faces.
6. Solve pressure correction equation to produce cell values of the pressure correction p' .
7. Update pressure field $p^{n+1} = p^n + \omega p'$ where ω is under relaxation factor for pressure.
8. Update boundary pressure correction
9. Correct face mass fluxes
10. Correct cell velocities
11. Update density due to pressure changes.
12. Free all temporary storage.

In Figure 2.2 the flow chart of the SIMPLE algorithm is being illustrated.

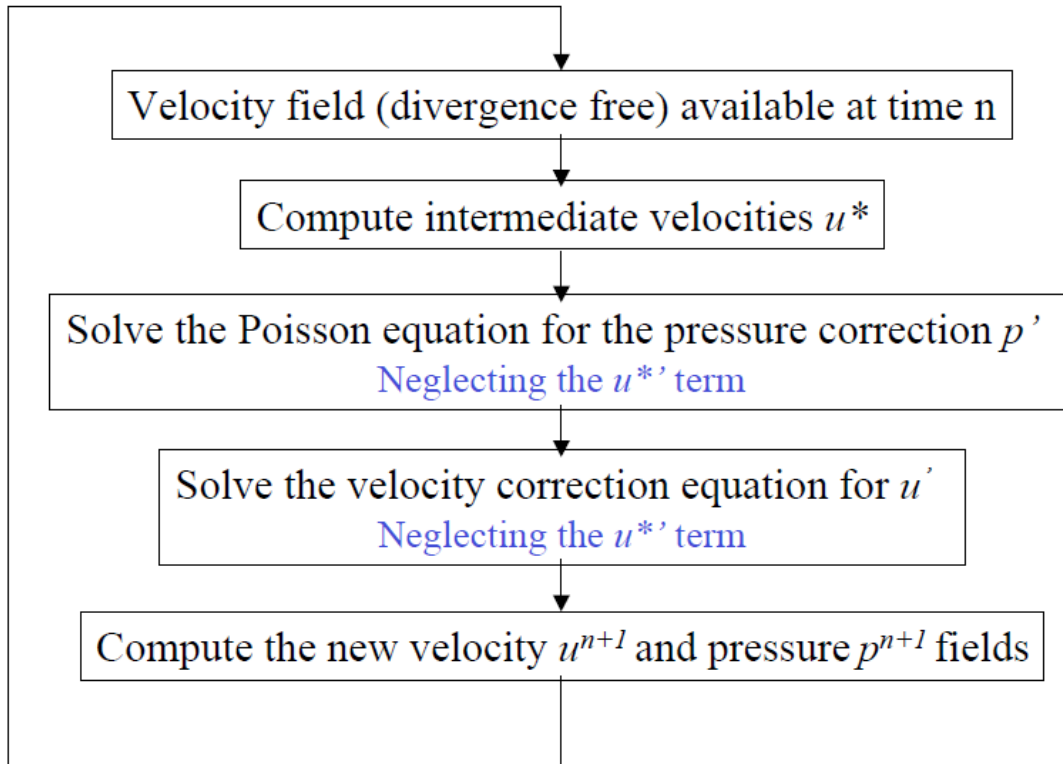


Figure 2.2 Flow chart of the SIMPLE algorithm

2.4 PISO Solver algorithm

PISO algorithm (Pressure implicit with splitting of operator) is an extension of SIMPLE algorithm. PISO is a pressure velocity calculation procedure for the Navier-Stokes equations developed originally for noniterative computation of unsteady compressible flow, but it has been adapted successfully on steady state problems (Issa (1986)).

PISO involves one predictor step and two corrector steps and the conservation of mass is designed to be satisfied within the predictor corrector steps.

The algorithm can be summarized as follows:

1. Set the boundary conditions.
2. Solve the discretized momentum equation to compute an intermediate velocity field.
3. Compute the mass fluxes at the cells faces.
4. Solve the pressure equation.
5. Correct the mass fluxes at the cell faces.
6. Correct the velocities on the basis of the new pressure field.
7. Update the boundary conditions.

8. Repeat from 3 for the prescribed number of times.
9. Increase the time step and repeat from 1.

As already seen for the SIMPLE algorithm, the steps 4 and 5 can be repeated for a prescribed number of time to correct for non-orthogonality. In Figure 2.3 the flow chart of the PISO algorithm is being illustrated.

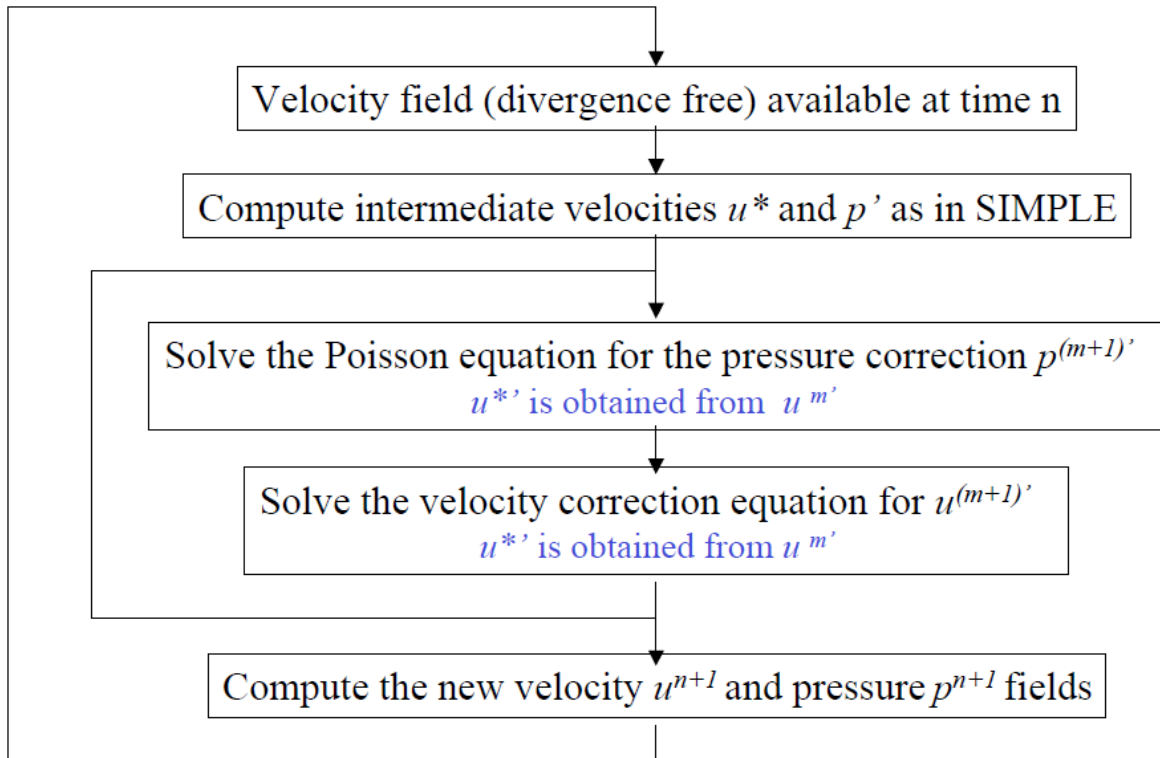


Figure 2.3 Flow chart of the PISO algorithm

In SIMPLE under-relaxation is required due to the neglect of u^{*}

$$u^{n+1} = u^* + a_u u' \quad (2.15)$$

$$p = p^n + a_p p' \quad (2.16)$$

There is an optimal relationship

$$a_p = 1 - a_u \quad (2.17)$$

PISO does not need under-relaxation, generally gives more stable results and takes less CPU time but is not suitable for all processes. For laminar backward facing step PISO is faster than SIMPLE but it is

slower for concerning flow through heated fin. If momentum and scalar equation have weak or no coupling then PISO is better than SIMPLEC. Finally PISO is suitable for irregular cells

2.5 VOF

VOF is the modeling method for the capture of the interphase which has been used in this work. VOF is a multiphase model which is well suited for simulation of flows where each phase constitutes a large structure, with relatively small total contact area between phases. The great advantage of VOF model is that it does not need to model inter-phase interactions therefore, it is computationally very efficient. However, it assumes that all phases within a partially filled cell, share the same velocity and pressure. For example if we have water and air in one cell, both are assumed to have the same pressure and velocity.

A good example of application of this method is in sloshing tanks. If the tank movement becomes very violent which result in breaking waves, large number of air bubbles and water droplets in the air, still VOF can be applied but needs very fine mesh in order to produce small modeling errors. VOF model is very sensitive to the grid used in simulation domain. Figure 2.4 shows proper and improper grid resolution that can be used to model air bubbles depending on their size in conjunction with VOF model.

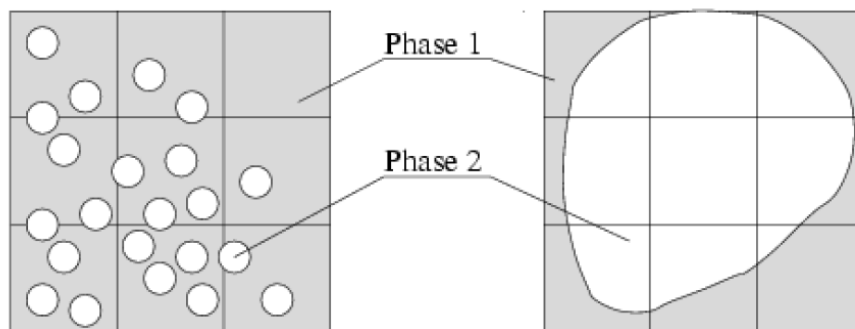


Figure 2.4 Illustration of grids that are suitable (right) and unsuitable (left) for two phase flows using VOF model adapted from CD-adapco (2010)

In VOF model spatial distribution of each phase at a given time is defined in terms of a variable called volume fraction α . Both fluids are treated as a single effective fluid whose properties vary in space according to the volume fraction of each phase, i.e.:

$$\rho = \sum_i \rho_i \alpha_i \quad (2.18)$$

$$\rho = \sum_i \mu_i \alpha_i \quad (2.19)$$

Where $\alpha_i = \frac{V_i}{V}$ is the volume fraction and ρ_i and μ_i are the density, and molecular viscosity of the i^{th} phase. For the case of two fluid mixture such as air and water mixture which is what we are dealing in this thesis we will have:

$$\rho = \rho_1 \alpha_1 + \rho_2 \alpha_2 = \rho_1 \alpha_1 + \rho_2 (1 - \alpha_1) \quad (2.20)$$

$$\mu = \mu_1 \alpha_1 + \mu_2 \alpha_2 = \mu_1 \alpha_1 + \mu_2 (1 - \alpha_1) \quad (2.21)$$

The transport of volume fraction α_i is described by the following conservation equation:

$$\frac{d}{dt} \int_V a_i dV + \int_S a_i (\mathbf{v} - \mathbf{v}_b) \cdot d\mathbf{a} = 0 \quad (2.22)$$

The discretization of transport equation (2.22) for a_i requires special care because a_i must be bound between zero and unity and the regions with partially filled cells should be as small as possible (Muzaferija and Peric (1998)). Equation (2.22) contains only convective fluxes and unsteady term. For time integration either fully implicit Euler method (for steady solution) or Crank-Nicolson (for unsteady solution) can be used. First order upwind scheme smears the interface too much and introduces artificial mixing of two fluids. Also since α must obey the bounds $0 < \alpha < 1$ one has to ensure that the scheme does not generate overshoots or undershoots. In addition, we have to ensure that convective flux out of one CV does not transport more of one fluid that is available in the donor cell. Also we have to take into account the interface orientation and local Courant number (Muzaferija and Peric (1998)). The sharpness between immiscible fluids is achieved by limiting the cell-face value to fall within shaded area of Normalized Variable Diagram (NVD) originally proposed by Leonard (1997) and shown in **Error! Reference source not found.**

This is the basis of the free surface tracking by volume of fluid method, and still corrections and considerations need to be applied on the scheme. Of the most notable one is the remedy proposed to avoid adverse effects of diffusion of phase fraction that results in smearing of the sharpness of the interface. Although equations are written for convection of phase fraction, there can be always the possibility of getting false diffusion (Versteeg and Malalasekera (2007)).

A possible remedy used in openFOAM is to introduce an extra term called Artificial Compression to the equation of phase fraction convection. Physically its role is to exert a pressure on the interface to keep it from dispersing. Thus the transport equation becomes (Rusche (2002)):

$$\frac{da}{dt} + \nabla \cdot (a\mathbf{u}) + \nabla \cdot (a(1-a)\mathbf{u}_r) = 0 \quad (2.23)$$

Where \mathbf{u}_r is a velocity field normal to the interface and applies the artificial compression on the surface. Based on the the term $(1-a)\mathbf{u}_r$, the region under the influence of compression velocity has phase fraction values other than 0 and 1. The artificial compression velocity in OpenFOAM is defined by the cAlpha value in the model setting. The zero value for cAlpha applies no compression velocity, and higher values than zero introduce the corresponding artificial velocities at the interface.

More details about these methods can be found in (Muzaferija and Peric (1998)) and (CD-adapco (2010)). Figure 2.5 shows how free surface is constructed from volume fraction α using VOF multiphase model.

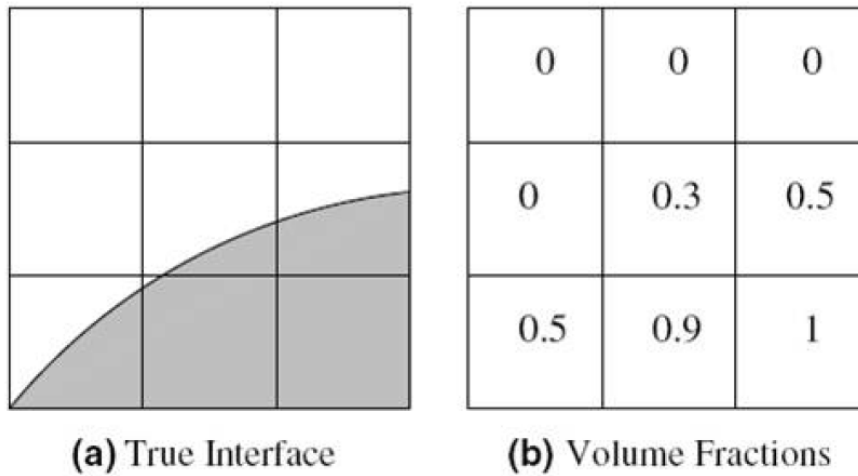


Figure 2.5 a) True interface b) Volume Fraction

3 Computational Setup

In this section the computational setup in STAR-CCM+ and openFoam is being presented. These two software products offer different capabilities so the setup was made as similar as possible. The biggest difference in terms of the solver is that STAR-CCM+ uses the SIMPLE algorithm as velocity pressure calculation procedure whereas openFoam PISO.

In order to perform the simulation, the geometry of the car Aletis was used. Aletis is a conceptual passenger car designed from TWT Gmbh and NIKI M.E.P.E. Unfortunately, although the car is fully designed, only the external geometry was available. In Figure 3.1 the external geometry of Aletis can be seen. In order to achieve the complexity under the hood of the car, a lot of CAD parts were imported. These CAD files were gathered from sites in the internet with free designs. Moreover a simplified version of cooling modules and air-suction system was designed.

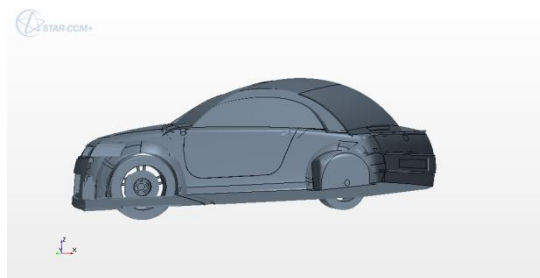


Figure 3.1 The available external geometry of Aletis

The final geometry (under the hood) is not a realistic one, but for the purposes of this work it serves its cause. In Figure 3.2 the final geometry of Aletis used for the simulations can be seen. Some parts are not visible in order for the under the hood geometry to be more observable.

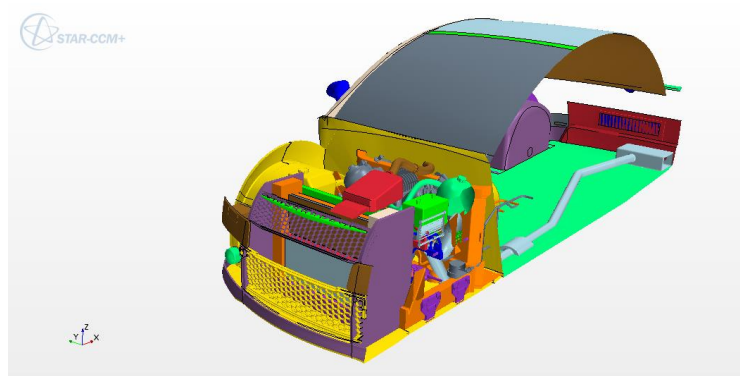


Figure 3.2 The final geometry of Aletis used for the simulations

3.1 Geometrical Setup and Mesh Generation in STAR-CCM+

STAR-CCM+ is a commercial software capable of tackling problems involving multi-physics and complex geometries. STAR-CCM+ has an established reputation for producing high-quality results in a single code with minimum user effort (<http://www.cd-adapco.com/products/star-ccm%2%AE>).

Designed to fit easily within engineering process, STAR-CCM+ makes it easy to entirely automate the simulation workflow and perform iterative design studies with minimal user interaction.

STAR-CCM+ gives the opportunity to manage the geometry very easily with many integrated tools. The post processing tools also help the engineer to process the results fast and with accuracy.

The Volume of Fluid (VOF) method in STAR-CCM+ is used to govern the air and water free surface evolution (Hirt and Nichols (1981), Muzaferija and Peric (1999)). The use of the VOF method for a typical sloshing problem implies the study of two fluids that must completely fill the tank domain. In this method, a passive scalar called volume fraction is assigned at each discretization cell, indicating the fraction of each phase in this particular cell. The scalar evolves in time with an additional transport equation in conjunction with the mass and momentum conservation laws. The VOF method is a widely used technique that has been applied to sloshing by many authors, like Kleefsman et al. (2005). STAR-CCM+ is a complete engineering software package able to solve the Navier–Stokes equations for laminar and turbulent flows and also able to move and deform meshes during the simulation. A semi-implicit method, known as SIMPLE algorithm, is used to handle the coupling of the Navier–Stokes equations and calculate the primitive variables of pressure and velocity at each time step.

The car is positioned in a distance of 8 meters from the Inlet and 14 from the outlet. The side walls are 6 meters away. In Figure 3.3 the geometry of the wind canal and the position of the car are being illustrated. The mesh consists of hexahedral cells apart from a small region (the air suction system) which consists of polyhedral cells. The cell size is 0.2 meters in the far field and 3mm close to the car.

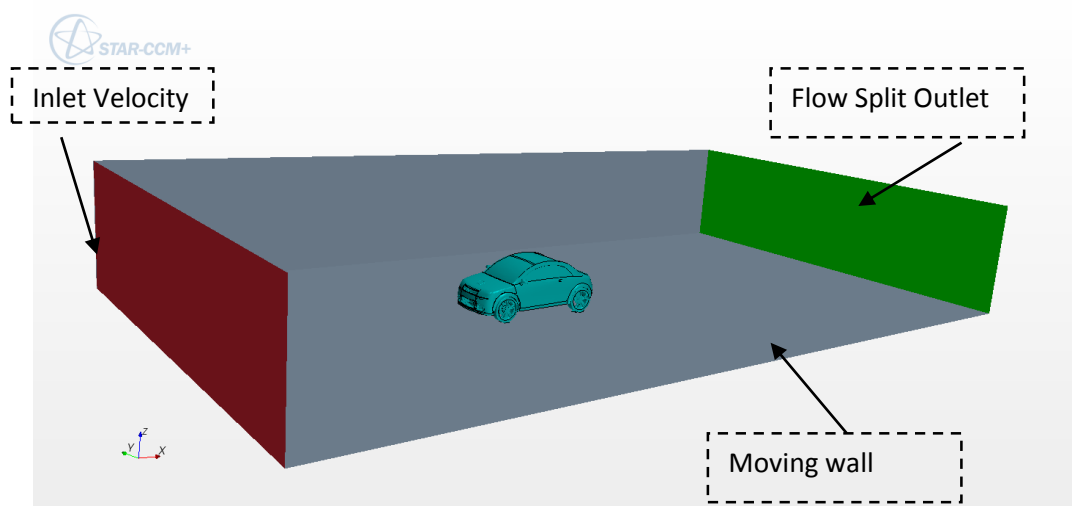


Figure 3.3 Geometry of the Wind Canal

In order to capture the gradients and the car geometry, the mesh should satisfy some prerequisites. It had to be very fine in regions such as around and in the front of the car and of course under the hood.

In Figure 3.4 and Figure 3.5 the positioning of the air suction system is shown. The inlet is close to the grids of the car (yellow color) and in front of the cooling system.

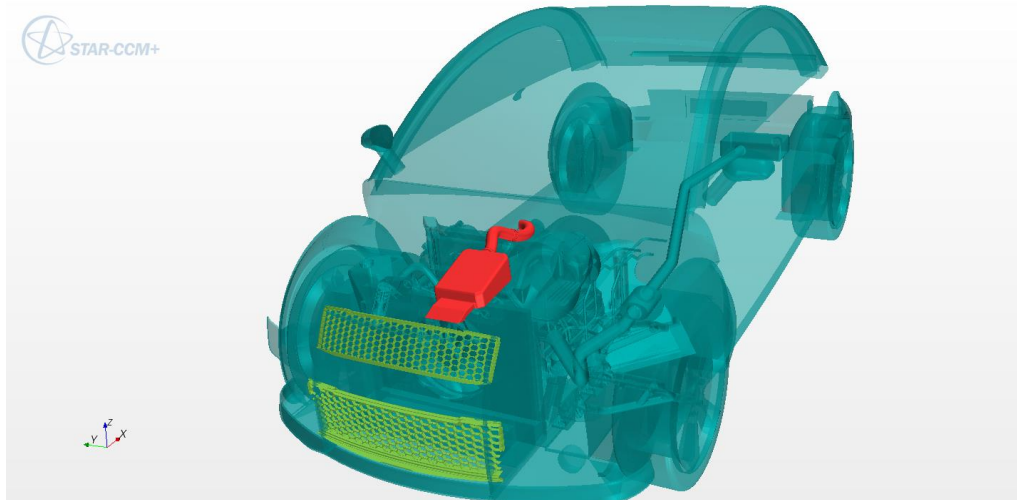


Figure 3.4 Position of the suction system (red colour)

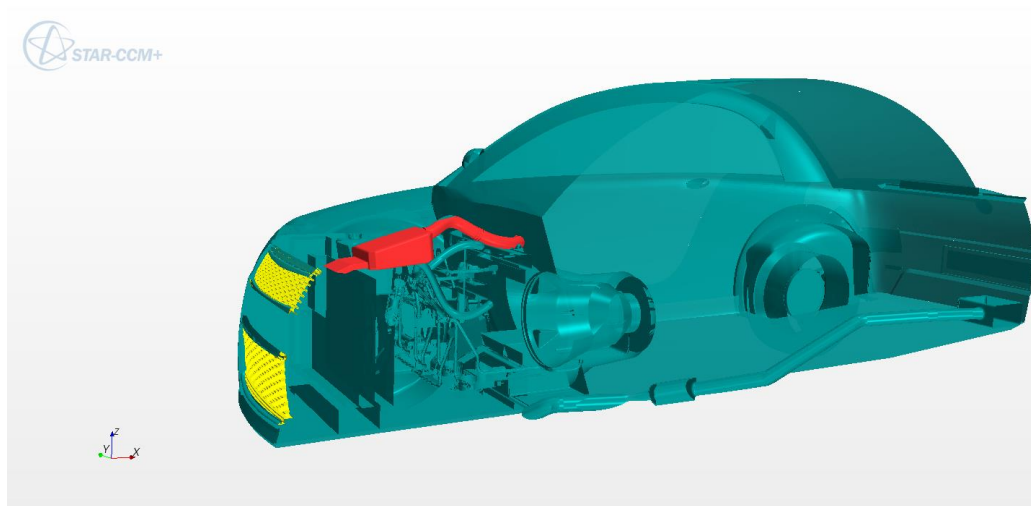


Figure 3.5 Position of the suction system (side view)

In Figure 3.6 the air suction system is being illustrated. The inlet sucks in fresh air which passes through the filter and ends up to the outlet. Through the outlet the air ends up in the engine. The suction of air from the engine is represented numerically by setting the outlet of the air-suction system as mass flow outlet.

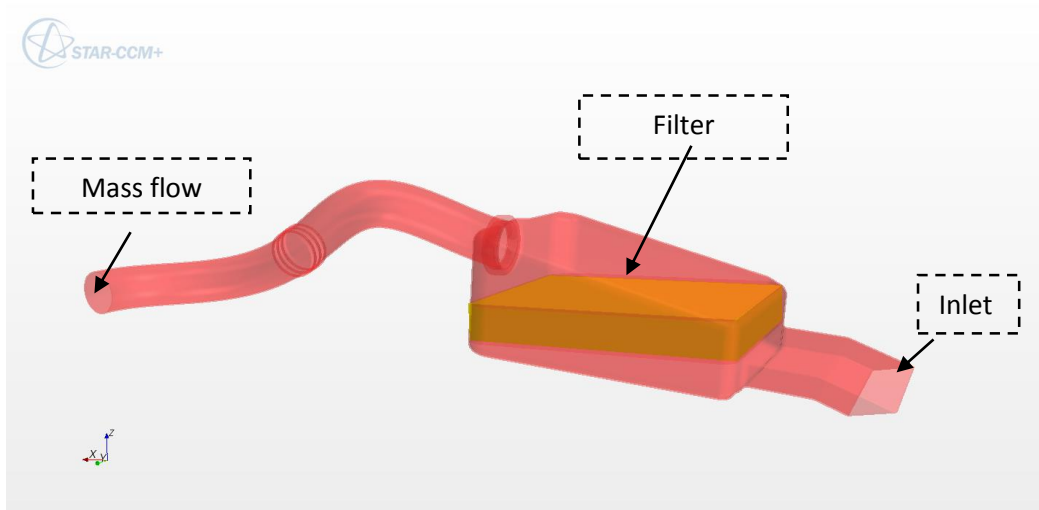


Figure 3.6 Air-suction system

In Figure 3.7 we can see an y _cut of the mesh ($y=0$) and in Figure 3.8 a z -cut ($z=0.1$) of the mesh. As it is obvious the mesh is coarser away from the car and denser close to it.

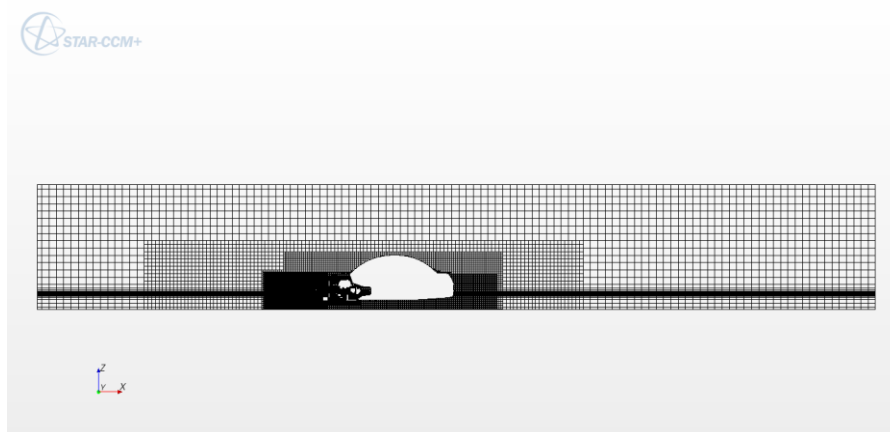


Figure 3.7 y -cut of the mesh ($y = 0$)

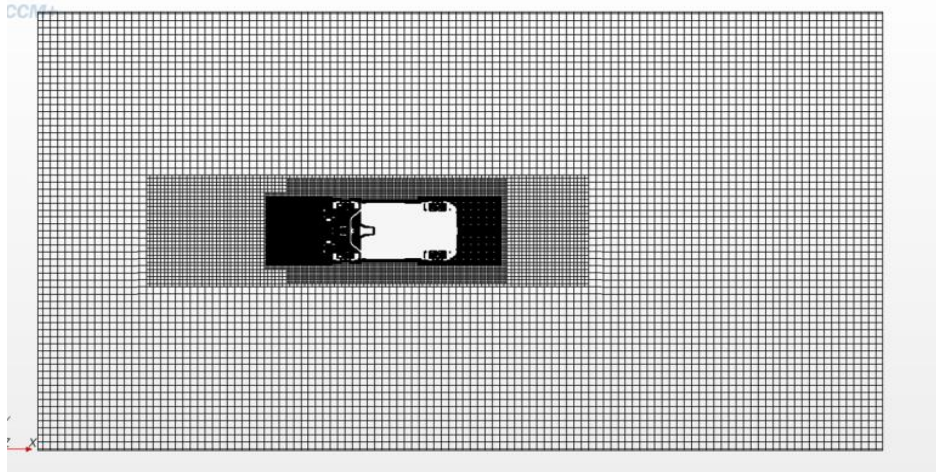


Figure 3.8 z-cut of the mesh ($z=0.1$)

In the mesh close to the car and under the hood is presented (y-cut). The mesh is dense enough to capture the geometry and the interface (air-water) correctly enough. In Figure 3.9 a y-cut and in Figure 3.10 an x-cut of the mesh under the hood of the car can be seen.

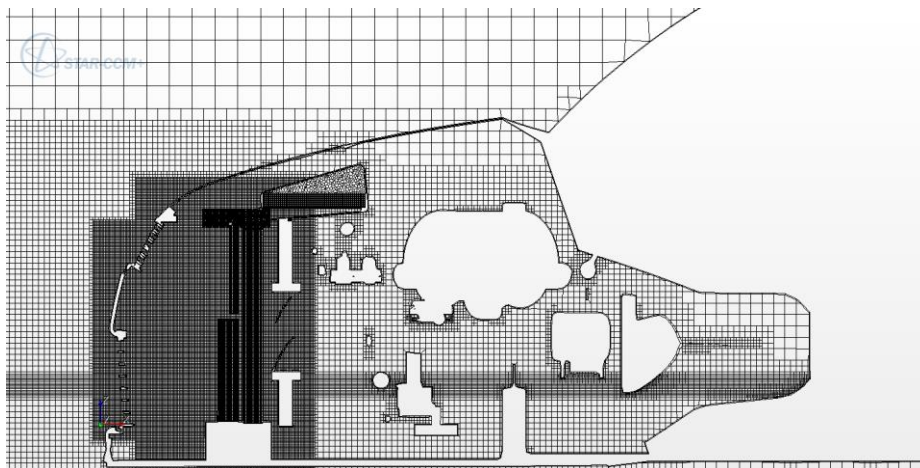


Figure 3.9 Close-up view of the mesh near the front of the car (y-cut)

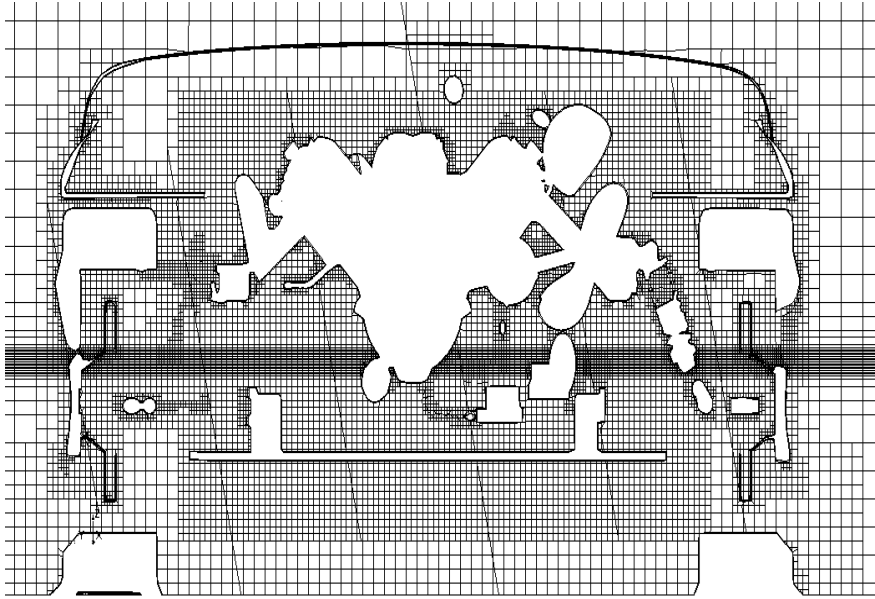


Figure 3.10 Close-up view of the mesh near the front of the car (x-cut)

3.2 Geometrical setup and mesh generation in openFoam

openFOAM CFD Toolbox is an open source CFD software package written in C++. The package includes a great number of solvers and tools appropriate for many CFD problems. For this work, the solver `porousInterFoam` has been used, which is a transient solver for incompressible, two phase, with porous regions flow. In this solver (as in most others), the discretization of the flow governing equations is based on the Finite Volume Method (FVM) with pressure and velocity solved by segregated methods. The solver implements the standard PISO (Pressure Implicit Splitting of Operators) method for pressure–velocity coupling. openFOAM can solve problems involving complex fluid flows with chemical reactions, heat transfer and turbulence, and can also work with solid dynamics and electromagnetics. openFOAM includes pre-processing and post-processing tools. It also offers the possibility to work in parallel, so that the user can take full advantage of the computer when working (<http://www.openfoam.com/>).

Moreover has its own specially designed data types and classes that enable user to manipulate field operation in a very effective way. For instance to define a velocity field over a specified grid points, one needs just to create an instance of a class called `volVectorField` for velocity. And this alleviates the need for accessing all data points while doing some operation on them.

openFOAM offers two tools for meshing. `blockMesh` and `snappyHexMesh` can be used in order to generate meshes that can capture complex geometries.

3.2.1 *blockMesh*

The *blockMesh* utility creates parametric meshes with grading and curved edges. The mesh is generated from a dictionary file named *blockMeshDict* located in the *constant/polyMesh* directory of a case. *blockMesh* reads this dictionary, generates the mesh and writes out the mesh data to points and faces, cells and boundary files in the same directory. .

The principle behind *blockMesh* is to decompose the domain geometry into a set of 1 or more three-dimensional, hexahedral blocks. Edges of the blocks can be straight lines, arcs or splines. The mesh is ostensibly specified as a number of cells in each direction of the block, sufficient information for *blockMesh* to generate the mesh data.

With *blockMesh* the size of the wind canal is being defined. In *openFoam* a longer canal is being used in order to succeed in having a more uniform flow in the outlet and near after the inlet. The size in the x direction is 45 meters (the car is positioned 15 meters after the inlet). The side wall are again 6 meters away from the car and the top wall is 6 meters higher.

3.2.2 *snappyHexMesh*

The *snappyHexMesh* utility generates 3-dimensional meshes containing hexahedral (hex) and split-hexahedral (split-hex) automatically from triangulated surface geometries in Stereolithography (STL) format. The mesh approximately conforms to the surface by iteratively refining a starting mesh and morphing the resulting split-hex mesh to the surface. An optional phase will shrink back the resulting mesh and insert cell layers. The specification of mesh refinement level is very flexible and the surface handling is robust with a pre-specified final mesh quality. It runs in parallel with a load balancing step every iteration.

The generated mesh consists of $7.9 \cdot 10^6$ cells. In regions far away from the car the cells are 40 cm cubes. Whereas near the car the cells are smaller than 1 cm. The mesh took 5,5 hours to be generated in 16 processors running in parallel.

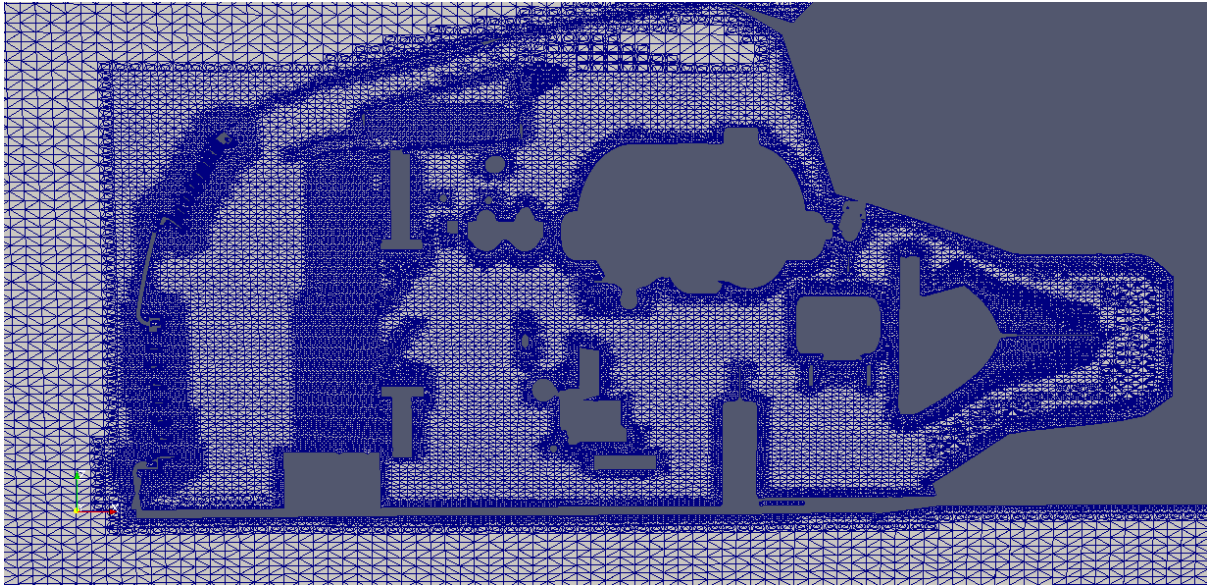


Figure 3.11 Close-up view of the mesh near the front of the car (y-cut)

3.2.3 *checkMesh*

By using this tool the user can check the validity of the mesh and if it respects user defined mesh quality controls. The maximum cell openness, the maximum aspect ratio, the mesh orthogonality and the maximum skewness of the mesh is being checked. The generated mesh failed in maximum skewness and in order for the solver not to crash, these cells had to be removed.

3.2.4 *Deleting bad cells*

All cells that are considered bad for OpenFOAM may impair the quality of the results, or might even prohibit obtaining a solution at all. One way of solving that problem is to delete these inappropriate cells. Usually they are located in very narrow places so deleting them will not dramatically affect the results of a typical aerodynamic case, but will help to avoid unphysical results in these particular cells. Deleting cells can be done by using the `setSet` and `subsetMesh` utilities. When `checkMesh` utility finds a particular type of inadmissible cells or faces it writes the description of them into the set file. The next step is to combine these sets of cells or sets of faces into one set of cells that can be deleted further on. Of course this operation should find for each face the cells that contain it. The responsible utility for that is `setSet`, the syntax of which is:

```
$ setSet [-batch file] [-latestTime] [-time time] [-parallel] [-constant] [-noVTK] [-noZero] [-case dir] [-region name]
```

When utility was run without -batch argument – it works in dialog mode. But a more efficient way of using this utility is by providing -batch file with commands. To delete skewFaces faces from the mesh the batch file batchForDeletingCells.batch was created.

```
cellSet cellsToDelete new
```

```
cellSet cellsToDelete add faceToCell skewFaces any
```

```
cellSet cellsToDelete invert
```

```
cellSet cellsToDelete subset
```

```
quit
```

The syntax of first four commands is:

```
<cellSet|faceSet|pointSet> <setName> <action> <source>
```

For example, by the third command in this file we add to cellSet “cellsToDelete” all cells that contain faces from skewFaces-set. Then the cell set “cellsToDelete” has to be inverted, in order for a new mesh to be generated from this set without undesirable cells. And then finally the new subset is created.

To feed this batch file to the setSet-utility the following command has to be typed:

```
$ setSet -batch batchForDeletingCells.batch
```

Now everything is ready to run subsetMesh on subset cellsToDelete. This utility will create a sub-mesh only containing cells from cellsToDelete.

```
$ subsetMesh cellsToDelete -overwrite
```

The new boundary patch was created after deleting cells : oldInternalFaces; it has type “empty” by default and this type has to be changed to “symmetryPlane” in order to reduce the influence of this boundary on the solution.

3.2.5 Decomposing

In order for the mesh to be generated in parallel the domain has to be split in the desired number of processors. This can be done in the decomposeParDict by using the following orders.

```
numberOfSubdomains 16;  
method simple;  
simpleCoeffs { n ( 8 2 1 );  
delta 0.001; }
```

The case was decomposed for 16 processors. Each processor had an operating Frequency of 3 GHz. The Ram memory was 48 Gb. Since the generated mesh was about $8 \cdot 10^6$ cells, the required RAM memory was about 10 GB.

3.2.6 renumberMesh

The performance of unstructured grids on workstations and distributed parallel computers is substantially affected by the efficiency of the memory hierarchy. This efficiency essentially depends on the order of computation and numbering of the grid. In order to renumber the cell list to reduce the bandwidth, the renumberMesh tool is being used. This tool exploits the Cuthill–McKee algorithm, which is an algorithm to permute a sparse matrix that has a symmetric sparsity pattern into a band matrix form with a small bandwidth.

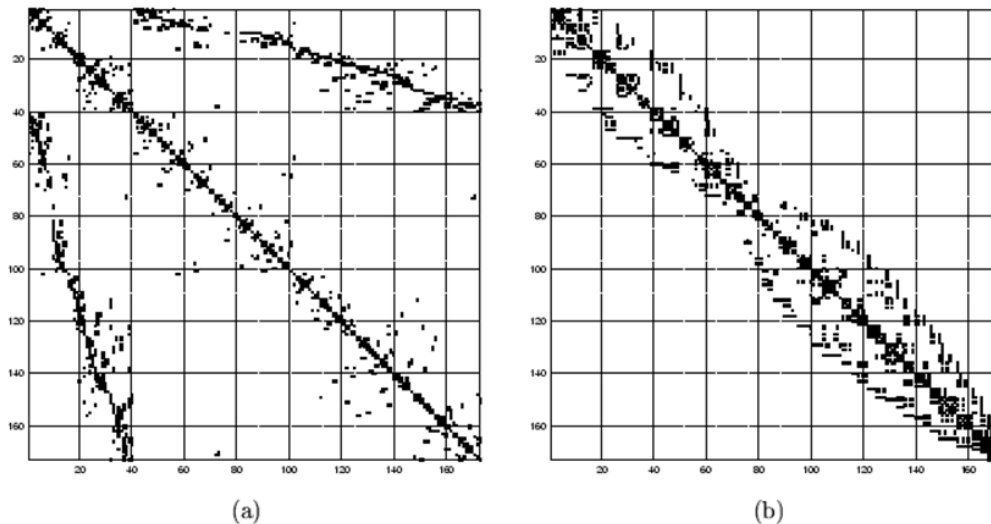


Figure 3.12 a) original matrix b) reordered matrix according to Cuthill–McKee algorithm

Indeed after the renumbering of the matrix the iterations were accomplished 5 times faster. This speed was not preserved after some hundreds of iterations for reasons that the author couldn't identify.

3.3 Motion of the car

In the experiments the water is still and the car drives through while accelerating from $V = 0\text{ m/s}$ with constant acceleration of about 0.5 m/s^2 until water is detected in the air-suction module. In the simulations, a simplification was made in order to minimize the computational time and take advantage of the available modeling techniques.

STAR-CCM+ offers overset mesh. An individual mesh is generated around the moving object (e.g. the car). This individual mesh can then be moved through a background mesh (e.g. wind canal). These two regions are connected via an overset interface. Since the simulation is unsteady this interface should be updated for every time step which would lead to an extremely high computational cost.

openFoam offers also a similar tool which is called interDyMFoam. This is a solver for 2 incompressible, isothermal immiscible fluids using a VOF (volume of fluid) phase-fraction based interface capturing approach, with optional mesh motion and mesh topology changes including adaptive re-meshing. But this technique would be also computationally expensive.

Instead of accelerating the car, the acceleration of the fluids was selected. According to the followed approach the car stands still and the water and the air are being accelerated. So in the inlet the velocity is being accelerated constantly at a value of 0.5 m/s^2 . In order to avoid wave formation due to different velocities in the water (Froude number) the whole water was accelerated through an x

component of gravity of $0.5m/s^2$. The air is being accelerated just because of the accelerating inlet velocity combined with the fact that both air and water are treated as incompressible fluids.

The advantages with this technique is that the computational cost is much lower while the results are close to the experiments.

3.4 Boundary conditions

STAR-CCM+ provides a “wave model” which can be used to simulate surface gravity waves and is used in conjunction with the VOF multiphase model. There are many wave types available such as flat wave, first order waves, fifth order waves etc. which allow to model waves with a specifies amplitude and periods.

In this case flat waves were used as the cars is supposed to drive through still water. The flat waves can be specified completely by providing the following details:

1. Height of the water
2. Velocity of the water. In our case it was not constant.
3. Velocity of the air (also not constant).
4. Density of the fluids

Inlet BC- With the help of the wave model the inlet boundary condition is a Dirichlet boundary condition for the velocity, the volume fraction was set constant according to the depth of the water and the pressure is extrapolated from the adjacent cells.

Outlet BC- The outlet was set flow split outlet. This boundary condition sets the velocity according to the adjacent cells while taking into account the mass preservation in the whole domain. The pressure and the volume fraction is extrapolated from the adjacent cells.

At turbocharger – the air-suction system has been modeled till the turbocharger inlet surface, as this surface is the end of the considered air-suction system. The region after this surface is not modeled since the water reaching the turbocharger inlet will eventually reach the engine. The turbocharger inlet was modeled as mass outflow BC. The air intake requirement of the engine at different speeds is known. So the velocity in this surface is set in a way that fulfills the required mass flow and the pressure is extrapolated from the adjacent cells.

In openFoam there is no such tool as the STAR-CCM+ wave model so the boundary conditions have to be set manually.

Inlet BC- in this boundary the velocity was posed as Dirichlet boundary condition while the pressure is extrapolated from the adjacent cells while respecting the inlet velocity. The volume fraction is set constant according to the depth of the water.

Outlet BC- the velocity is again set as Dirichlet boundary condition (the velocity is calculated by setting the desired volume rate in the boundary, this volume rate is equal to the inlet volume rate) and the pressure is treated the same way as the inlet BC too. The volume fraction is set as zero gradient.

The specification of boundary conditions was one of the most time consuming parts of this work. OpenFoam has not a similar simulation (with transient boundary conditions) in the Tutorials. Moreover the author didn't find any papers dealing with time dependent velocity boundary conditions combined with the interFoam solver. After a lot of tests the below table sums up the set of used boundary conditions.

	U	P_rgh	Epsilon	K	Alpha	nut
Inlet	uniform FixedValue	fixedFlux Pressure	fixedValue	fixedValue	calculated	zeroGradient
Outlet	uniform FixedValue	fixedFlux Pressure	zeroGradient	zeroGradient	zeroGradient	zeroGradient
Sidewalls	uniform FixedValue	fixedFlux Pressure	epsilon WallFunction	kqRWallFunction	zeroGradient	nutkWallFunction
Ground	uniform FixedValue	fixedFlux Pressure	epsilon WallFunction	kqRWallFunction	zeroGradient	nutkWallFunction
Atmosphere	uniform FixedValue	fixedFlux Pressure	epsilon WallFunction	kqRWallFunction	zeroGradient	nutkWallFunction
Filters	slip	fixedFlux Pressure	epsilon WallFunction	kqRWallFunction	zeroGradient	nutkWallFunction
Engine_inlet	massFlowRate*	fixedFlux Pressure	zeroGradient	zeroGradient	zeroGradient	zeroGradient
oldInternalFaces	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane

The main obstacle was that with any other combination of boundary conditions ,in the inlet and the outlet, the water entered the domain but either couldn't leave the domain through the outlet or it would leave with higher velocities, resulting in filling or emptying the domain.

The velocity type outlet boundary condition needs special treatment. In order to make the matrix less stiff the outlet was set far away from the car (30 meters away) in order to have an almost uniform flow.

3.5 Initialization

The initialization of the field variables velocity, pressure and volume fraction is very important for this kind of simulations.

The velocity was set as 0 all over the domain, the pressure 0 although in the water the pressure was calculated as hydrostatic automatically. The volume fraction of water was set 1 where water should be present and 0 where air should be present.

3.6 Porous medium

The filter of air-suction modules is made up of paper folded in a zig-zag way. The filter is able to prevent any dust particles or droplets to enter the engine. The air filter in the simulations is modeled as a porous medium. By this technique the macroscopic effect of the filter on the flow is retained, although the flow through the filter is not realistic. The effect of porous media on the flow is defined using lumped parameters, which are typically taken to be resistance coefficients for a source term in the momentum equation. In Figure 3.13 the air filter of BMW cars is illustrated.



Figure 3.13 Air filter of a 3-series BMW car

The source term is composed of two parts: a viscous loss term (first term on the right hand side of equation) and an inertial loss term (the second term on the right-hand side of the equation)

$$S_i = -\left(\frac{\mu}{\alpha} v_i + C \frac{1}{2} \rho |v| v_i\right) \quad (3.1)$$

Where S_i is the source term in the $i^{th}(x, y, or z)$ momentum equation. μ is the fluid molecular viscosity, α is permeability factor and C is the inertial resistance factor.

The values for the viscous and inertial resistance coefficients are obtained through experiments. The available experimental coefficients were for air. Applying these values, it was observed that the filter allowed the water to pass through with small resistance. In order to increase the resistance to water, the coefficient values were set phase dependent. So the water coefficients were set 1000 times higher than the air, the same ratio of water density to air density (Kothiwale (2013)).

The parts the cooling module such as the radiator and the condenser were modeled as porous media, like the filter.

While in STAR-CCM+ to specify a region as porous media is trivial, in openFoam there were some bugs to overcome. The basic porosity dictionary had to be modified in order for the solver to run. If not, when the solver porousInterFoam is executed, a fatal error appears (and similar error exists with versions 2.2.2 and 2.2.1).

In order for the problem to be tackled the dictionary is being modified. The variable thermo:mu is used where it should be only mu.

Add the line mu mu; in the DarcyForchheimer porosity dictionary defined in the constant/porosityProperties file. By doing so the muName keyword should be set to mu instead of thermo:mu in the solver.

```

24 DarcyForchheimerCoeffs
25 {
26 mu mu;
27 d d [0 -2 0 0 0 0] (7 e7 7e7 5e8 );
28 f f [0 -1 0 0 0 0] (0 0 0);

```

3.7 Time discretization

The current version of STAR-CCM+ allows the usage of VOF multiphase flow models only in conjunction with implicit schemes. The explicit schemes are dominated from the Courant number, which forces the simulation to have a small time step, but is considered to have less numerical diffusion and have a crisper interface. The greatest disadvantage is that it is difficult to converge in low quality meshes.

The Courant numbers for a one-dimensional case is defined as:

$$C = \frac{u\Delta t}{\Delta x} \quad (3.2)$$

Where C is the dimensionless Courant number, u is the magnitude of the velocity, Δt is the time step and Δx is the length interval.

Implicit schemes don't need to respect the Courant number. Thus greater time steps can be accomplished. However the numerical diffusion is higher. In STAR-CCM+ the implicit scheme was used because of the usage of the SIMPLE algorithm. On the other hand in openFoam the explicit scheme was used, denoted by the usage of PISO. This resulted in higher time steps in STAR-CCM+ than openFoam. CD-adapco claims that STAR-CCM+ can have good results even with very high Courant numbers (VOF simulations) whereas interFoam of openFoam needs a maximum Courant number of 1. Any higher value would result in crashing of the solver.

3.8 Fluid properties

The free surface is where the interaction between air and water is, and can often become unstable. Therefore an appropriate technique needs to be used to make sure it models this section correctly. Since this a multiphase situation the best approach is the coupled volume of fluid (VOF) method. This is an excellent way of modeling ships that produce breaking waves, because it can be used for two immiscible fluids where the interface position of these fluids is required to be calculated throughout the simulation (Zhang, Liu, et al. (2006)). The volume fraction is used to calculate the value of alpha. For values of 0 the fluid is air and for values of 1 the fluid is water. Anything between this is a mixture of the two and hence there will be an interface. The continuity equation for alpha is used to locate the interface by determining where alpha is changing at the fastest rate (Zhang, Liu, et al. (2006)).

The fluid properties are defined in the transportProperties dictionary. Water and air are defined in the sub dictionaries of respectively phase1 and phase2. For each phase the keyword transportModel is set to Newtonian. A Newtonian fluid is characterized by a constant kinematic viscosity which is kept unchanged with the rate of deformation. In this directory the word nu defines the kinematic viscosity and is set to 1.48e-05 for air and 1e-06 for water. Density, which is named as rho, is set to 1000 kg/m³ for water and 1 kg/m³ for air. The surface tension between air and water is defined as sigma and its value is specified to 0.07, adopted from the dam break tutorial.

3.9 Gravity

In OpenFOAM the gravity is a uniform vector field across the computational domain and can be set in the dictionary g. The absolute value of the gravity is -9.81 m/s² in the z-axis and 0.5 m/s² in the x-axis in order to accelerate the whole domain in accordance with the acceleration of the inlet velocity. The same setup was made in STAR-CCM+.

3.10 Discretization schemes in openFoam

The discretization schemes are set in a way to help the solver converge but minimize the diffusion. After a lot of tests the best combination is presented below.

ddtSchemes		default backward
gradSchemes		default faceLimited Gauss linear 1
divSchemes	div(rho*phi,U)	Gauss limitedLinearV 1
	div(phi,alpha)	Gauss vanLeer01
	div(phi*rb,alpha)	Gauss interfaceCompression
	div(phi,k)	Gauss upwind
	div(phi,epsilon)	Gauss upwind
	div(phi,R)	Gauss upwind
	div(R)	Gauss linear
	div(phi,nuTilda)	Gauss upwind
	div((muEff*dev(T(grad(U)))))	Gauss linear
laplacianSchemes		Default Gauss linear limited 0.5
interpolationSchemes		default linear
snGradSchemes		Default limited 0.5

The first time derivative terms are specified in the ddtSchemes sub-dictionary. The discretization scheme selected is the *backward*, which is a second order implicit scheme.

The *divSchemes* sub-dictionary contains divergence terms. Let us discuss the syntax of the entry in reference to a typical convection term found in fluid dynamics, which in OpenFOAM applications is commonly given the identifier `div(phi,U)`, where `phi` refers to the flux.

The `Gauss` scheme is the only choice of discretization and requires a selection of the interpolation scheme for the dependent field. To summarize, the entries required are:

```
Gauss <interpolationScheme>
```

The interpolation scheme is selected from the full range of schemes in Table 3.1, both general and convection-specific. The choice critically determines numerical behavior as described in Table 3.2. The syntax here for specifying convection-specific interpolation schemes *does not include the flux* as it is already known for the particular term, *i.e.* for `div(phi,U)`, we know the flux is `phi` so specifying it in the interpolation scheme would only invite an inconsistency.

Centered schemes	
linear	Linear interpolation (central differencing)
cubicCorrection	Cubic scheme
midPoint	Linear interpolation with symmetric weighting
Upwinded convection schemes	
upwind	Upwind differencing
linearUpwind	Linear upwind differencing
skewLinear	Linear with skewness correction
filteredLinear2	Linear with filtering for high-frequency ringing
TVD schemes	
limitedLinear	limited linear differencing
vanLeer	van Leer limiter
MUSCL	MUSCL limiter
limitedCubic	Cubic limiter
NVD schemes	
SFCD	Self-filtered central differencing
Gamma ψ	Gamma differencing

Table 3.1 Interpolation schemes.

Scheme	Numerical behavior
linear	Second order, unbounded
skewLinear	Second order, (more) unbounded, skewness correction

cubicCorrected	Fourth order, unbounded
upwind	First order, bounded
linearUpwind	First/second order, bounded
QUICK	First/second order, bounded
TVD schemes	First/second order, bounded
SFCD	Second order, bounded
NVD schemes	First/second order, bounded

Table 3.2 Behavior of interpolation schemes used in *divSchemes*

The *gradSchemes* sub-dictionary contains gradient terms. The discretization scheme for each term can be selected from those listed in Table 3.3

Discretization scheme	Description
Gauss <interpolationScheme>	Second order, Gaussian integration
leastSquares	Second order, least squares
fourth	Fourth order, least squares
cellLimited <gradScheme>	Cell limited version of one of the above schemes
faceLimited <gradScheme>	Face limited version of one of the above schemes

Table 3.3 Discretization schemes available in *gradSchemes*.

The Gauss keyword specifies the standard finite volume discretization of Gaussian integration which requires the interpolation of values from cell centers to face centers. Therefore, the Gauss entry must be followed by the choice of interpolation scheme from Table 3.1. It would be extremely unusual to select anything other than general interpolation schemes and in most cases the linear scheme is an effective choice, *e.g.*

`grad(p) Gauss linear;`

Limited versions of any of the 3 base gradient schemes — Gauss, leastSquares and fourth — can be selected by preceding the discretization scheme by cellLimited (or faceLimited), *e.g.* a cell limited Gauss scheme

`grad(p) cellLimited Gauss linear 1;`

The Gauss scheme is the only choice of discretization and requires a selection of both an interpolation scheme for the diffusion coefficient, *i.e.* ν in our example, and a surface normal gradient scheme. To summarize, the entries required are:

Gauss <interpolationScheme> <snGradScheme>

The interpolation scheme is selected from Table 3.1, the typical choices being from the general schemes and, in most cases, `linear`. The surface normal gradient scheme is selected from Table 3.5; the choice of scheme determines numerical behavior as described in Table 3.4. A typical entry for our example Laplacian term would be:

```
laplacian(nu,U) Gauss linear corrected;
```

Scheme	Numerical behavior
corrected	Unbounded, second order, conservative
uncorrected	Bounded, first order, non-conservative
limited ψ	Blend of corrected and uncorrected
bounded	First order for bounded scalars
fourth	Unbounded, fourth order, conservative

Table 3.4 Behavior of surface normal schemes used in *laplacianSchemes*.

The *snGradSchemes* sub-dictionary contains surface normal gradient terms. A surface normal gradient is evaluated at a cell face; it is the component, normal to the face, of the gradient of values at the centers of the 2 cells that the face connects. A surface normal gradient may be specified in its own right and is also required to evaluate a Laplacian term using Gaussian integration.

The available schemes are listed in Table 3.5 and are specified by simply quoting the keyword and entry, with the exception of `limited` which requires a coefficient ψ , $0 \leq \psi \leq 1$.

$$\psi = \begin{cases} 0 & \text{corresponds to **uncorrected**,} \\ 0.333 & \text{non-orthogonal correction} \leq 0.5 \times \text{orthogonal part,} \\ 0.5 & \text{non-orthogonal correction} \leq \text{orthogonal part,} \\ 1 & \text{corresponds to **corrected**.} \end{cases}$$

A limited scheme with $\psi = 0.5$ is therefore specified as default by:

```
default limited 0.5;
```

Scheme	Description
corrected	Explicit non-orthogonal correction
uncorrected	No non-orthogonal correction
limited ψ	Limited non-orthogonal correction
bounded	Bounded correction for positive scalars
fourth	Fourth order

Table 3.5 Surface normal gradient schemes.

3.10.1 Time step and data output control

According to the OpenFOAM foundation, the surface tracking algorithm in interFoam is more sensitive to the Courant number than other models. It is recommended that the Courant number is equal or less than 0.5 in the interface. In the controlDict file the time adjustments can be made. Here it is not useful to have a fixed time step because the propagation of the velocity is not easily predicted, so adjustTimeStep is set to yes. maxAlphaCo sets the maximum Courant number applied to alpha1 and is set to 0.5. maxCo is the same as the previous keyword but it is applied to other fields such as p_rgh and U, and is also set to 0.5.

The maximum Courant number is a result of a small cell and a high velocity. maxDeltaT determines the upper limit to the time step, and it is set to 1. The keyword writeInterval sets the time when the results are written, even if in OpenFOAM the calculations are performed at arbitrary time steps, and is set to 0.05. writeControl is set to adjustableRunTime to allow this. Then the startFrom keyword is set to startTime, and startTime is set to 0, so the first fields data input is read from the 0/ directory. writeFormat is set to the default value ascii, but it could also be written in the binary format by typing binary. The keyword writeFormat is set to 6 and writeCompression is set to compressed, because the data files are big. Finally, the timeFormat is set to general and timePrecision is set to 6 which are default values for OpenFOAM.

4 Results

From an engineering point of view, the difficulties in dealing with multiphase flows is that the mass, momentum and energy transfer rates and processes are quite sensitive to the geometric distribution or topology of the components in the flow. The nature of flow within each phase will depend on that geometric distribution. Hence there is a complicated two way coupling between the flow in of the phases and the geometry of the flow. This complexity due to two way coupling is a major challenge in the study of multiphase flows.

Before going into the results, a quick note on the reporting techniques and surfaces used for analysis. Mainly two surfaces have been considered in the analysis. These are the surface just after the inlet of the air suction system and before the filter (Dirty air) and the surface just after the filter (Clean air). The region of air-suction system after the filter is a critical region since any traces of water in this region will travel to the engine.

The velocity of the car at the moment that the surface before the filter is filled with water at a percentage of more than 30 per cent, is being defined as the critical velocity. If water reaches this surface, because of the suction of air from the engine, it is certain that it will end up in the engine.

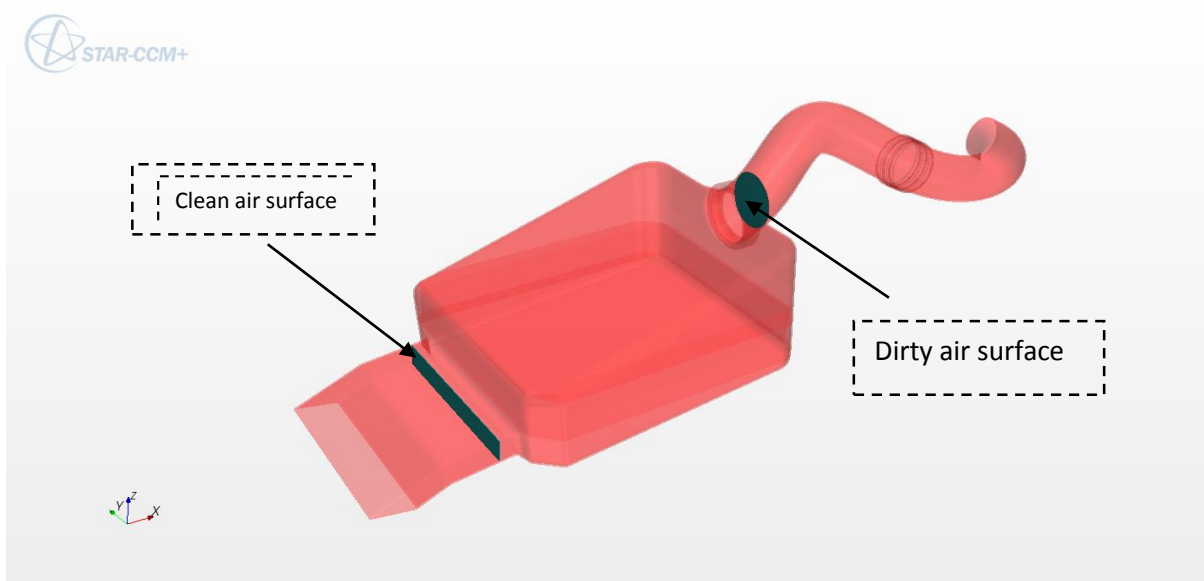


Figure 4.1 Clean air surface and dirty air surface of the air suction system.

The evaluation method that has been used for analysis is the surface average which is represented by

$$\text{Surface average} = \frac{1}{a} \int \phi da \quad (4.1)$$

Where a is the surface area.

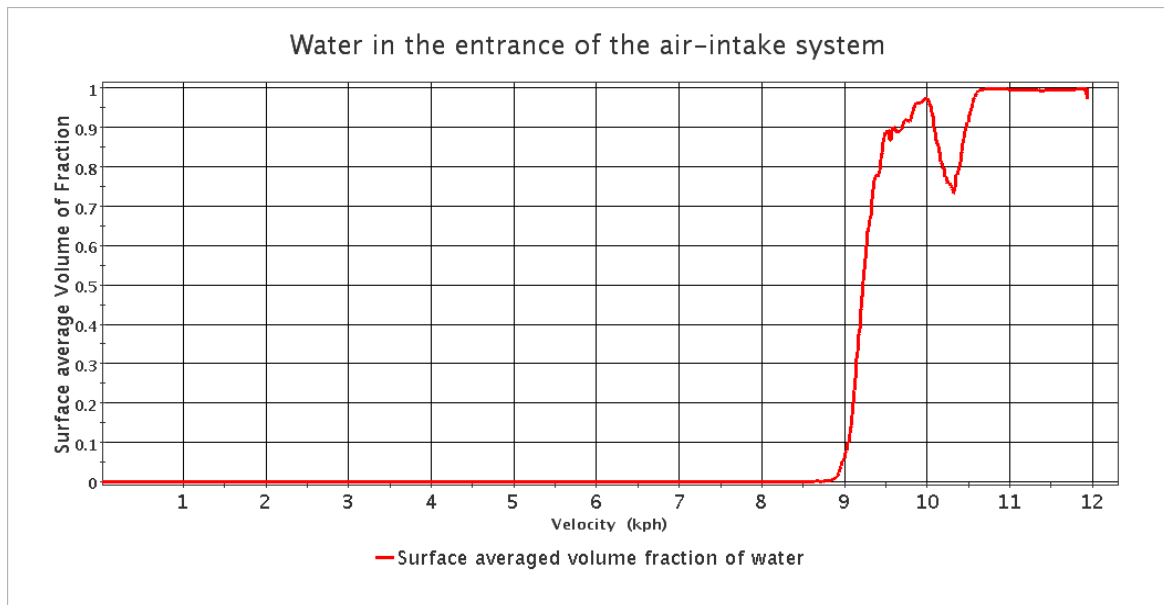


Figure 4.2 Surface averaged volume fraction of water in the entrance of the air-intake system

The scalars that have been used in the analysis are pressure, volume fraction of water and velocity. Surface average of volume of fraction represents the ratio of area that contains cells in specified volume fraction range weighted with respective volume fraction values to the total surface area. Surface average value 1 represents the surface which is completely filled with volume fraction of water of value 1 and surface average value of 0 represents there are no cells with volume fraction in specified range on the surface.

In Figure 4.2 it can be seen that the surface before the filter is filled with water when the car's speed is almost 9.2 km/h. It could be said that the critical velocity for this car, when travelling in 40 cm deep water is 9.2 km/h. At this speed water reaches the engine resulting in its permanent damage.

4.1 Results with STAR-CCM+

In Figure 4.3 and 4.4 the iso-surface created for volume of Fraction = 0.5 can be seen. The value 0.5 has been considered as the one which indicated the interface between water and air. The velocity of the water has reached almost 12 km/h and big wave is formed in front of the car. This wave has covered the front side of the car and obviously has reached the air-suction system.

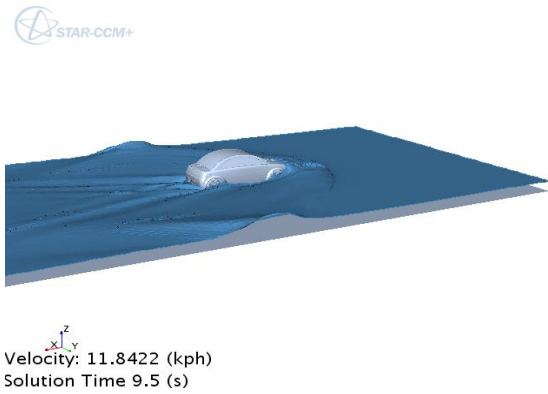


Figure 4.3 Iso-surface volume of fraction =0.5

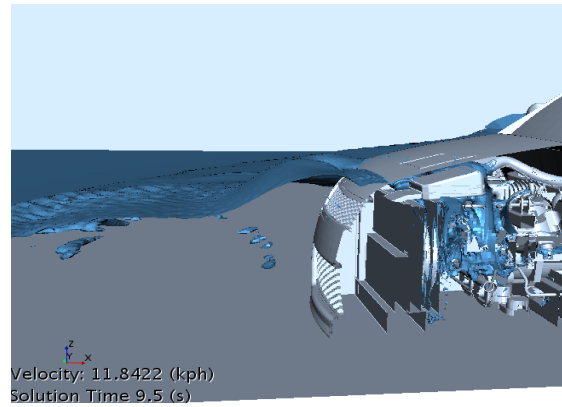


Figure 4.4 Iso-surface volume of fraction =0.5
(view under the hood)

In Figure 4.5 the volume fraction of water in a y normal plane (origin $y=0$) is being illustrated. The wave in front of the car gets bigger as the velocity of the water increases. In these photos one can identify the porous behavior of the cooling system as it decelerates the water acting almost as a wall.

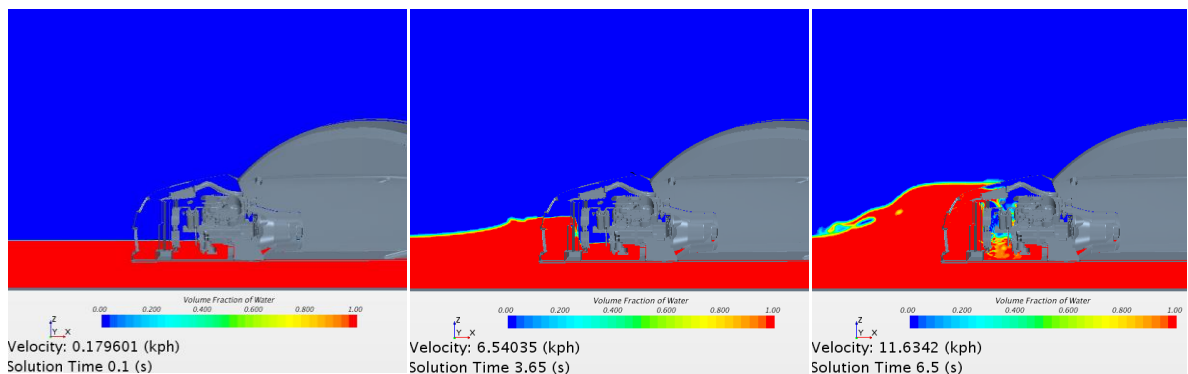


Figure 4.5 Volume fraction of Water in different velocities

In Figure 4.6 the velocity (magnitude) distribution is being illustrated. One can indicate the almost zero velocity in the cooling systems (they have been modeled as porous media). The low velocity close to the surface of the wave (according to Froude number that's the reason it is being formed)

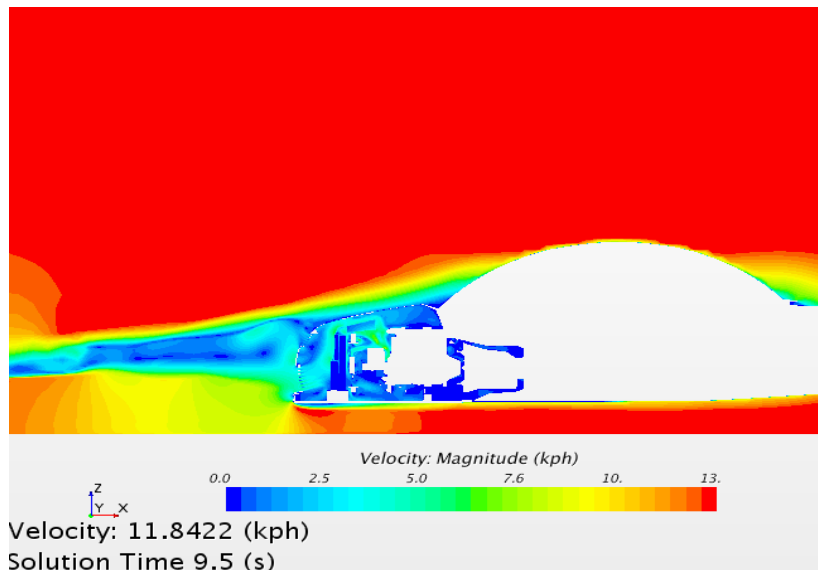


Figure 4.6 Velocity magnitude distribution in a y-cut plane

The pressure distribution in Figure 4.7 shows that the pressure is mostly affected by the depth of the water.

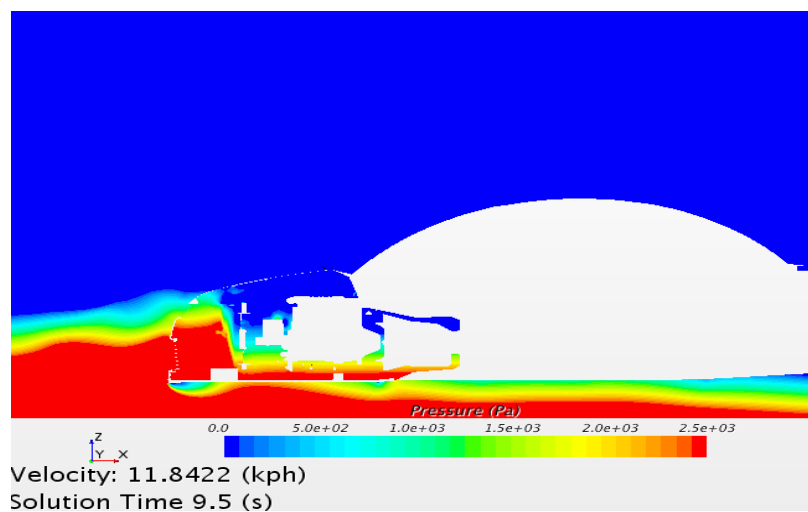


Figure 4.7 Pressure distribution in a y-cut plane

In Figure 4.8 the Courant number is being shown. The Courant number is more than one in a lot of cells but the simulation didn't diverge and instabilities didn't occur. The reason is that STAR-CCM+ use a semi-implicit scheme which allows to be stable in higher Courant numbers.

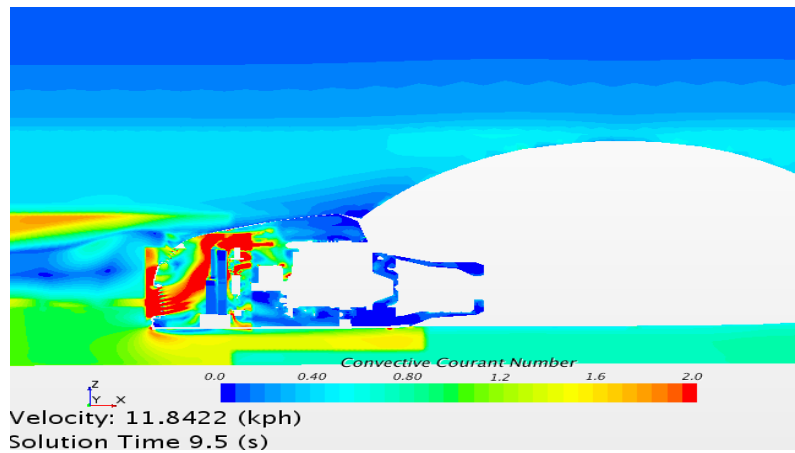


Figure 4.8 Convective Courant number

4.2 Results with openFoam

The results with openFoam were not thoroughly post processed. The reason was that the solver stopped crashing only a few days before of the deadline of this work and there was not enough time. Fortunately the last run did not crash and the results were good. The solver used to crash because the Courant number was increasing to higher values than 1. This was happening because of bad quality cells and not suitable boundary conditions. Especially the latter was addressed after a lot of effort.

In Table 4.1 different time steps of the isosurface of volume of fraction are being presented. The formation of waves is similar to STAR-CCM+

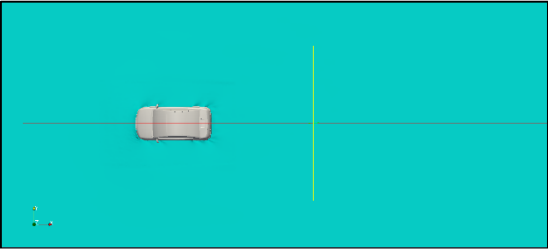
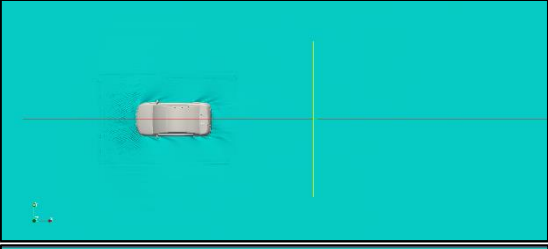
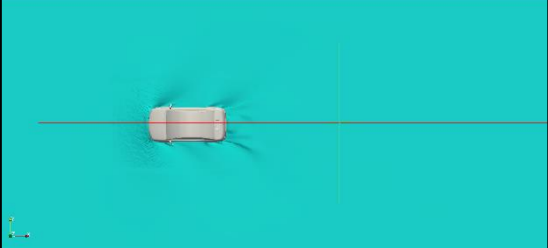
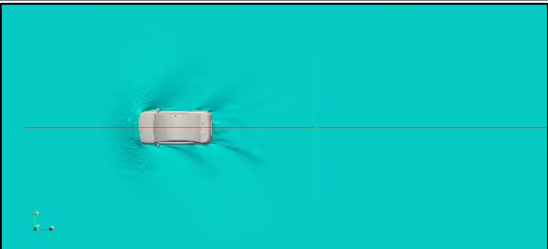
Physical time	Water velocity	Isosurface (0.5 Volume fraction of water)
2.2 sec	1.1 m/sec ²	
3.2 sec	1.6 m/sec ²	
4.4 sec	2.2 m/sec ²	
5.2 sec	2.6 m/sec ²	

Table 4.1 Top view of iso-surfaces ($\alpha=0.5$) in different time steps

In Table 4.2 the volume fraction of water as a scalar is being illustrated. The wave in front of the car gets bigger as the velocity gets higher. The interface between the phases is not precise. Perhaps a higher velocity compression and a finer grid would result in a crisper interface. The wave does not propagate in the upstream direction in comparison with STAR-CCM+. The reason may be the lower porosity in the cooling system falsely put in openFoam. The height of the wave though is very close to STAR-CCM+, making the results very promising.

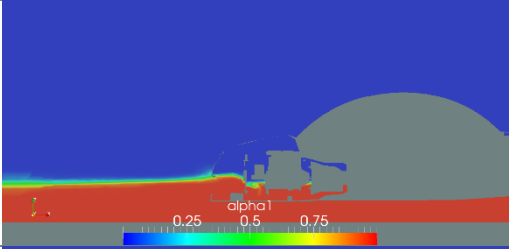
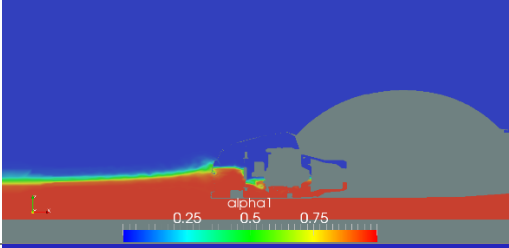
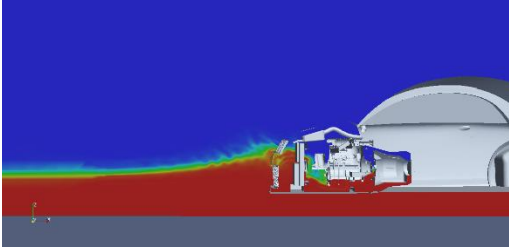
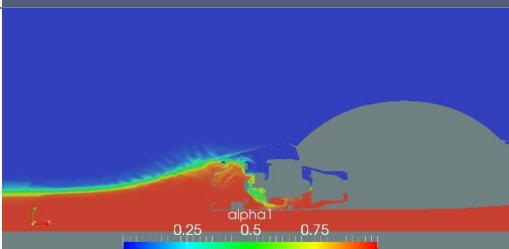
Physical time	Water velocity	Volume fraction of water
2.2 sec	1.1 m/sec ²	
3.2 sec	1.6 m/sec ²	
4.4 sec	2.2 m/sec ²	
5.2 sec	2.6 m/sec ²	

Table 4.2 Side view of the scalar field Volume Fraction of Water (y-cut, y=0)

In Table 4.3 the iso-surface of Volume of Fraction under the hood of the car can be seen. The surface of the water is not smooth because of the finer grid in this area. openFoam proves to be not very efficient when the grid changes size, resulting in creating false small deviations of the interphase. This more intense in higher velocities. The porous regions decelerate the fluid in lower rate as the porosity values are low.

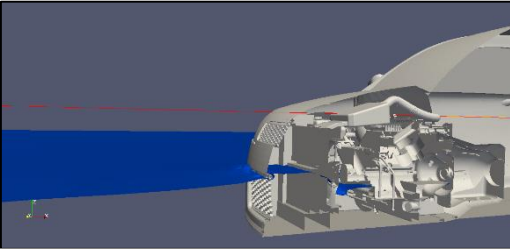
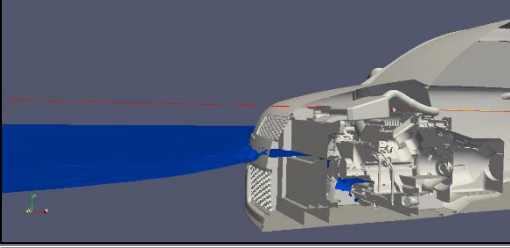
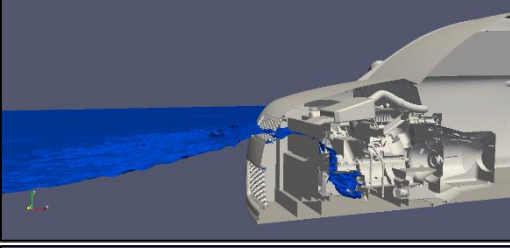
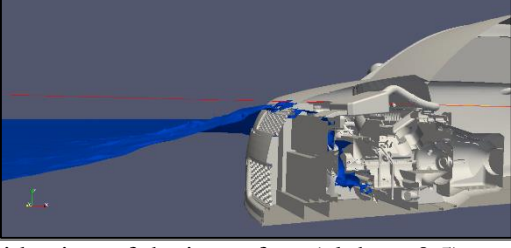
Physical time	Water velocity	Isosurface (0.5 Volume fraction of water)
2.2 sec	1.1 m/sec ²	
3.2 sec	1.6 m/sec ²	
4.4 sec	2.2 m/sec ²	
5.2 sec	2.6 m/sec ²	

Table 4.3 Close side view of the isosurface ($\alpha = 0.5$)

The velocity distribution (side view, $y=0$) in different time steps is shown in Table 4.4. the velocity in the interphase is the same for both air and the water because of the nature of VOF. VOF assumes same velocities for both phases in cells that are occupied by the two phases

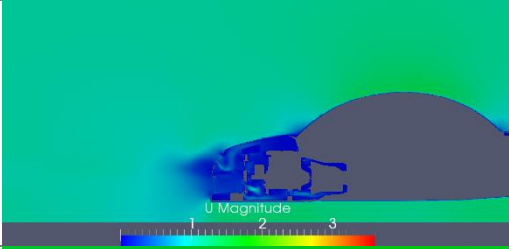
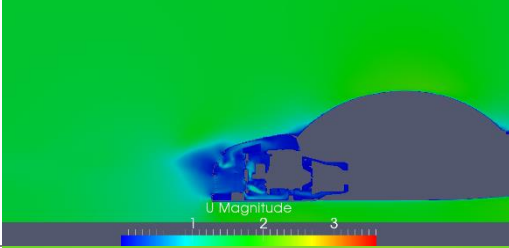
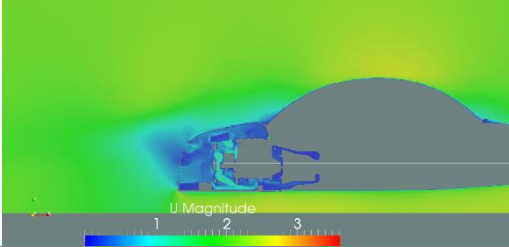
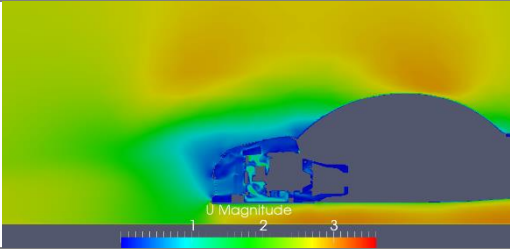
Physical time	Water velocity	Velocity (Magnitude) distribution, y-cut y=0
2.2 sec	1.1 m/sec ²	 A side-view velocity magnitude plot at 2.2 seconds. The flow is represented by a color gradient from blue (low velocity) to red (high velocity). A dark grey obstacle is visible on the right. A color scale at the bottom ranges from 1 to 3.
3.2 sec	1.6 m/sec ²	 A side-view velocity magnitude plot at 3.2 seconds. The flow field shows more developed structures around the obstacle compared to 2.2 seconds. The color scale at the bottom ranges from 1 to 3.
4.4 sec	2.2 m/sec ²	 A side-view velocity magnitude plot at 4.4 seconds. The flow structures are further developed and more turbulent. The color scale at the bottom ranges from 1 to 3.
5.2 sec	2.6 m/sec ²	 A side-view velocity magnitude plot at 5.2 seconds. The flow is highly turbulent with complex structures. The color scale at the bottom ranges from 1 to 3.

Table 4.4 Side view of velocity (Magnitude) distribution in different time steps

4.3 Comparison

A thorough comparison between the results of STAR-CCM+ and openFoam is not possible. The post processing of openFoam results is not the same with STAR-CCM+. This happened not only because of the different capabilities of the two software products, but because of lack of time. Nevertheless the compared results seem to be very close and encourage the researcher to believe that with further investigation could become almost identical.

4.3.1 Critical velocity

In terms of critical velocity the two software products gave almost the same answer. STAR-CCM+ predicted that the engine would fill with water when the car travels in a speed of 9.2 km/s, whereas openFoam in a speed of about 9.5 km/s. openFoam porosity values were lower and the suction of the air-intake system was deactivated. These reasons could explain the difference.

4.3.2 Wave formation

In Table 4.5 the formation of waves can be compared. The different graphics of the two software products make it difficult to recognize that the results are almost identical. A closer look though can confirm that the waves especially at the back side of the car are identical. In front of the car seem to be a little different. The different porosity values and the different smoothing of the iso-surfaces can be the reasons.

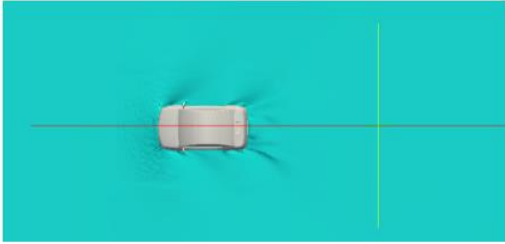
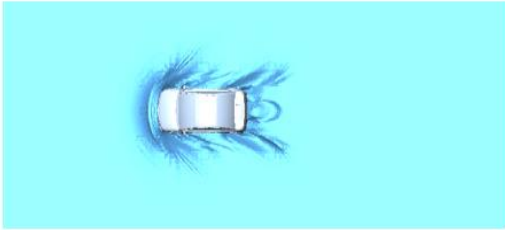
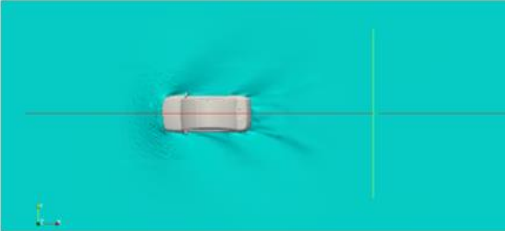
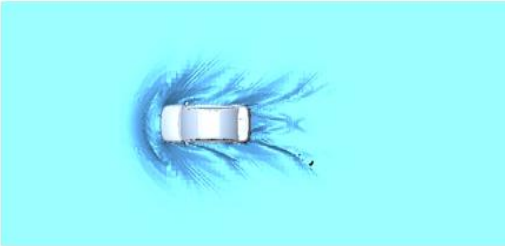
Physical time	Water velocity	Isosurface (0.5 Volume fraction of water)	
2.2 sec	1.1 m/sec ²		<u>openFoam</u>
			<u>Star-ccm+</u>
5.2 sec	2.6 m/sec ²		<u>openFoam</u>
			<u>Star-ccm+</u>

Table 4.5 Wave formation comparison between STAR-CCM+ and openFoam

4.3.3 Velocities

The comparison of velocity distribution (see Table 4.6) indicates that the results are very close. The greatest differences are spotted in the air suction system because in openFoam was not activated. Also the cooling modules present a great difference because of the different porosity values. The difference in graphics should be taken into consideration.

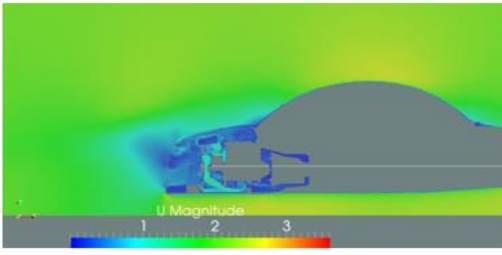
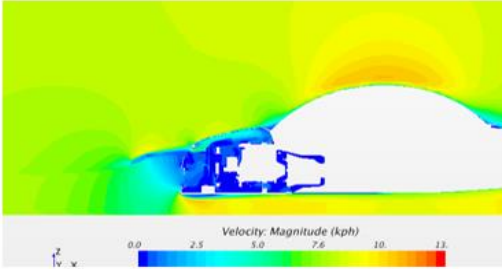
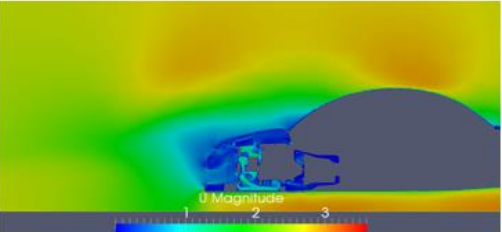
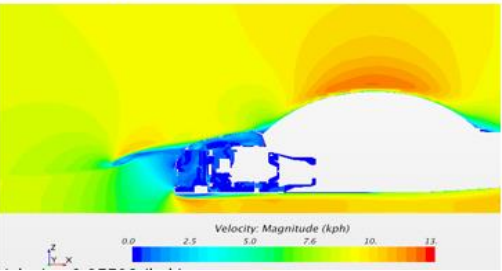
Physical time	Water velocity	Velocity (Magnitude) distribution, y-cut y=0	
2.2 sec	1.1 m/sec ²		<u>openFoam</u>
			<u>Star-ccm+</u>
5.2 sec	2.6 m/sec ²		<u>openFoam</u>
			<u>Star-ccm+</u>

Table 4.6 Comparison of velocity distribution (y-cut, y=0)

4.3.4 Volume fraction

The side view of volume fraction of water (Table 4.7) shows that the interphase in STAR-CCM+ is crisper than openFoam. This is because in the former the grid is a little finer, and the velocity compression functions better. Perhaps a higher value of compression in openFoam would result in crisper interphase.

Moreover the wave seems to travel also backwards STAR-CCM+, whereas in openFoam not. This could be explained because of the lower porosity values. In tests that were performed it was realized that openFoam can predict such kind of waves.

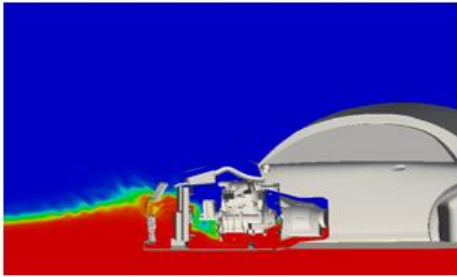
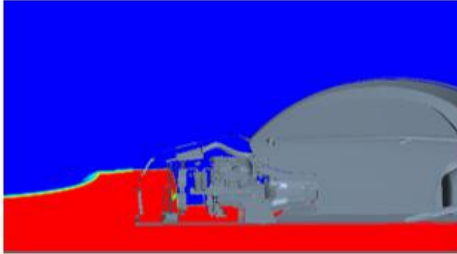
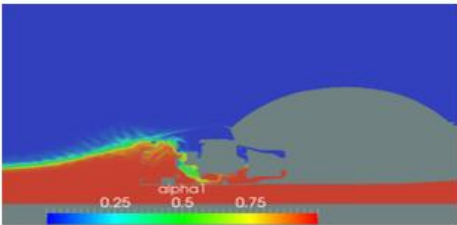
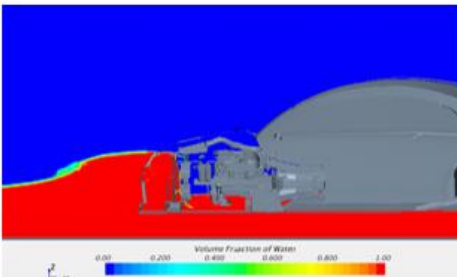
Physical time	Water velocity	volume fraction of water	
2.2 sec	1.1 m/sec ²		<u>openFoam</u>
			<u>Star-ccm+</u>
5.2 sec	2.6 m/sec ²		<u>openFoam</u>
			<u>Star-ccm+</u>

Table 4.7 Comparison of volume fraction of water (y-cut, y=0)

5 Conclusions

This work is the first try to create a methodology in openFoam capable of simulating the water drive through of conventional cars. The tackled obstacles were many but there are more to be tackled. The author believes that the results are promising and little more effort is required to make it possible for engineers to acquire reliable results regarding the critical velocity of cars driving through flooded roads.

5.1.1 Geometry handling

STAR-CCM+ provides the user with various tools to handle the geometry. It offers the possibility even to handle the wrapped and remeshed surface in a gui environment. This result is that the user can inspect and modify the geometry fast and easy.

On the contrary openFoam offers limited tools to handle the geometry. Through paraFoam the geometry can be visually inspected with limited accuracy. Through surfaceCheck one can check the geometric and topological quality of a surface. The offered tools to tackle any potential problems (e.g. non closed surface) are not efficient for large and complex geometries.

5.1.2 Mesh generation

The mesh generation in STAR-CCM+ is highly robust. The software provides the user with a variety of options and tools resulting in better capture of the geometry and a thorough manipulation of the refinement.

snappyHexMesh is a powerful tool but lacks on robustness and customization. The user has limited options for refinement control, cannot define the desired dimensions of the cells easily. On the other can run in parallel and is highly automatable.

5.1.1 Simulation setup

Both of the software products are highly automatable and make the setup procedure convenient. STAR-CCM+ offers better handling of the boundary conditions and the GUI makes it more user friendly.

5.1.2 Processing

STAR-CCM+ used an implicit solver while openFoam an explicit. The implicit solver of STAR-CCM+ has no time step constraint which makes the processing time less (the time step in STAR-CCM+ was set $5 * 10^{-3}$). On the contrary the explicit solver porousInterFoam is extremely sensitive in the Courant number which resulted in a variable time step dependent on the size of the cells and the velocities. This is a big drawback as it increases the computational cost significantly and makes it almost impossible to solve simulations with higher velocities and finer meshes. This was the reason that the mesh in openFoam was not so fine compared to STAR-CCM+ (also the mesh refinement in STAR-CCM+ was not high enough). The accuracy of explicit solvers does not compensate for this disadvantage.

5.1.3 Post-processing

The post processing in STAR-CCM+ is almost fun to do. The possibilities are almost limitless and the setting is extremely user friendly. paraFoam and openFoam offer limited tools and difficult to set. It takes a lot of experience and search to use the offered tools. Especially in large scale simulations becomes even more difficult. It should be mentioned though that the authors experience in ParaFoam is restricted.

5.1.4 Results

The results produced from STAR-CCM+ have been partially validated in the past with the help of experimental results, mainly focused on the formation of the wave and the critical velocity. The results of the simulation come into rough qualitative agreement with previous experimental results, in terms of flow portraits. Further investigation of the porosity values and the mass flow value in the engine should be made.

openFoam predicts similar critical velocity to STAR-CCM+. The formation of the wave in front of the car is not similar though. A possible answer is the lower porosity values set accidentally by the user. In general the results are promising, while taking into consideration that this is the first try of performing this kind of simulation.

5.2 Future work

Regarding openFoam many improvements should be made.

- At first a new simulation should be carried out with the right porosity values and the engine mass flow boundary activated. This could answer the question about the form of the wave and the absence of upstream propagation of the wave.
- Tests with higher interface compression should be made in order to investigate whether it is possible to make the interface crisper.
- A way should be found to overcome the very small time step in finer meshes. This could be met with semi implicit solvers. The LTSInterFoam is a possible solution, if the solver becomes able to handle porous regions. The other possibility is to create a multiregional simulation with dedicated solvers for each region.
- Essential is the development of a methodology to handle the geometry in openFoam. All the necessary geometry modifications were operated in STAR-CCM+ as the user did not find a way to make it in openFoam.
- Post processing in STAR-CCM+ is already automated while in openFoam is not. This could be done through already offered functions and libraries controlled from the ControlDict file.

References

Ansys fluent theory guide, (2011).version 14.0, November.

Ansys fluent theory guide, version 14.0, November 2011.

Bergles A. E., Collier J. G., Delhay J. M., Hewitt G. F. Mayinger and F. (1981). Two-phase Flow and Heat Transfer in the Power and Process Industries. Hemisphere Publ. Corp., Washington.

Casey M., Lang E., Mack R., Schlegel R. and Wehrli M. (1988). Applications of computational fluid dynamics for process engineering at Sulzer. Speedup J., 12(1): 43-51.

CIVE 2400: Fluid Mechanics Section 1. Fluid Flow in Pipes. School Of Mechanical Engineering. University Of Leeds online:http://www.efm.leeds.ac.uk/cive/cive2400/pipeflow2_2008a.pdf

Cive 2400: Fluid Mechanics Section 2. Open Channel Hydraulics. School Of Mechanical Engineering. University Of Leeds Online:

[Http://www.efm.leeds.ac.uk/Cive/Cive2400/OpenChannelhydraulics2.Pdf](http://www.efm.leeds.ac.uk/Cive/Cive2400/OpenChannelhydraulics2.Pdf)

CIVE 2400: Fluid Mechanics Section 2. Open Channel Hydraulics. School Of Mechanical Engineering. University Of Leeds online: www.efm.leeds.ac.uk

Coleman H. W., (1996). Proc. 1996 ASME Fluids Engineering Division Summer Meeting, San Diego, CA, USA, Jul 7-11.

Crowe C. T. et al., (1994). Numerical methods in multiphase flows: presented at the 1994 ASME Fluids Engineering Division Summer Meeting, Lake Tahoe, NV, USA, volume 185, Jun 19-23.

Daly, B.J., (1969). Numerical Study of the Effect of Surface Tension on Interface Instability, Phys. Fluids 12, 1340.

Damian S.M., (2009). Description and utilization of interFoam multiphase solver.

Douglas, J.F., Gasiorek, J.M., Swaffield, J. A. & Jack, L.B. (2005). Fluid Mechanics, Harlow, Person Prentice Hall.

Drazin, P. G. and Reid, W. H. (1981). Hydrodynamic Stability. Cambridge University Press. Cambridge.

Finnemore, E. J., Franzini, J.B. & Daugherty, R.L. (2002). Fluid Mechanics with Engineering Application. Boston. McGraw-Hill.

Fung Y. C. (2001). Classical and computational solid mechanics. World Scientific, Singapore.

Gerhart P. M., Gross R. J. and Hochstein J. I. (1992). Fundamentals of fluid mechanics, Addison-Wesley Publishing Company.

Gosman A. D., (1998). Developments in industrial computational fluid dynamics. Chem. Eng. Res. Des., 76(A2):153-161.

Harlow, F.H. and Welch, J.E., (1965). Numerical Calculation of Time-Dependent Viscous Incompressible Flow, Phys. Fluids 8, 2182.

Herrerias N., Izarra J., (2013). Two-Phase Pipe flow Simulations with openFoam. Master thesis, Norwegian University of Science and Technology.

Hetsroni G. (1982). Handbook of Multiphase Systems. Hemisphere Publ. Corp., Washington.

Hill D. P., (1998). The Computer Simulation of Dispersed Two-Phase Flows. PhD thesis, Imperial College, University of London.

Hirano M. and Tomiyama A., (1994). Numerical method for a two-fluid model based on the modified SOLA method. In Crowe et al., (1994), pages 125-129.

Hirt, C.W. and Nichols, B.D., (1981). Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries, J. Comp. Phys. 39, 201.

Hirt, C.W. and Nichols, B.D., (1981). Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries, J. Comp. Phys. 39, 201.

Hirt, C.W., Amsden, A.A., and Cook, J.L., (1974). An Arbitrary Lagrangian-Eulerian Computing Method for all Flow Speeds, J. Comp. Phys., 14, 227.

Hirt, C.W., Cook, J.L. and Butler, T.D., (1970). A Lagrangian Method for Calculating the Dynamics of an Incompressible Fluid with Free Surface, J. Comp. Phys. 5, 103.

Hjertager B. H., (1998). Computational fluid dynamics (CFD) analysis of multiphase chemical reactors. Trends Chem. Eng., 4:45-92.

<http://www.cd-adapco.com/products/star-ccm%C2%AE>

<http://www.openfoam.com/>

ICMF98, (1998). Proc. 3rd Int. Conf. on Multiphase Flow, Lyon, France (CDROM), June 8-12.

Ishii M. (1975). Thermo-Fluid Dynamic Theory of Two-Phase Flow. Eyrolles, Paris.

Issa R., (1986). Solution of the implicitly discretized fluid flow equations by operator-splitting. Journal of Computational Physics 62.

J. O. Hinze, (1975). Turbulence. McGraw-Hill Publishing Co., New York.

Kashiwa B. A., Padial N. T., Rauenzahn R. M. and Van der Heyden W. B., (1994). Cell-centered ICE method for multiphase flow simulations. In Crowe et al., (1994), pages 159-167.

- Keil F., Mackens W., Voss H. and Werther J., (1999). *Scientific Computing in Chemical Engineering II* {Computational Fluid Dynamics, Reaction Engineering and Molecular Properties. Springer-Verlag, Berlin.
- Kleefsman, K.M.T., Fekken, G., Veldman, A. E. P., Iwanowski, B., Buchner, B., (2005). A volume-of-fluid based simulation method for wave impact problems. *Journal of Computational Physics* 206, 363–393.
- Kleiber M., (1998). *Handbook of computational solid mechanics: survey and comparison of contemporary methods*. Springer-Verlag, Berlin.
- Kothiwale S., (2013). *Multiphase CFD simulation air-suction system and cooling system*, Aachen, Germany.
- Kuipers J. A. M. and van Swaaij W. P. M., (1998). Computational fluid dynamics applied to chemical reaction engineering. *Adv. Chem. Eng.*, 24:227-328.
- Lathouwers D. and Van Den Akker H. E. A., (1996). Numerical method for the solution of two-fluid model equations. In Coleman (1996), pages 121-126.
- Lathouwers D., (1999). *Modelling and Simulation of Turbulent Bubbly Flow*. PhD thesis, Technische Universiteit Delft.
- Muzaferija, S., Peric, M., (1999). Computation of free surface flow using interface tracking and interface capturing methods. In: Mahrenholtz, O., Markiewicz, M. (Eds.), *Nonlinear Water Wave Interaction Computational Mechanics Publications*, Southampton, pp.59–100. (Chapter2).
- Nichols, B.D. and Hirt, C.W., (1971). Calculating Three-Dimensional Free Surface Flows in the Vicinity of Submerged and Exposed Structures, *J. Comp. Phys.* 12, 234.
- Nichols, B.D. and Hirt, C.W., (1975). Methods for Calculating Multidimensional, Transient Free Surface Flows Past Bodies, *Proc. of the First International Conf. On Num. Ship Hydrodynamics*, Gaithersburg, ML, Oct. 20-23.
- Nichols, B.D. and Hirt, C.W., (1980). Numerical Simulation of BWR Vent-Clearing Hydrodynamics, *Nucl. Sci. Eng.* 73, 196.
- Politis S., (1989). *Prediction of Two-Phase Solid-Liquid Turbulent Flow Stirred Vessels*. PhD thesis, Imperial College, University of London.
- Richardson, S. M. (1989). *Fluid Mechanics*. Hemisphere. New York.
- Rosenhead, L. Ed. (1963). *Laminar Boundary Layers*. Clarendon Press. Oxford.
- Rusche H. (2002). *Computational Fluid Dynamics of Dispersed Two-Phase Flows at High Phase Fractions*, PhD Thesis, Imperial College of Science, Technology & Medicine, Department of Mechanical Engineering.

Shabbir A., Yang Z., Shih T.-H., Liou W. W. and Zhu J., (1995). Eddy Viscosity Model for High Reynolds Number Turbulent Flows – Model Development and Validation. *Computers Fluids*, 24(3):227-238.

Shabbir A., Yang Z., Shih T.-H., Liou W.W. and Zhu J., (1995). Eddy Viscosity Model for High Reynolds Number Turbulent Flows – Model Development and Validation. *Computers Fluids*, 24(3):227-238.

Shapiro, A. H. (1964). *Shape and Flow*. Heinemann. London.

Spalding D. B., (1983). Developments in the IPSA procedure for numerical computation of multiphase-flow phenomena with interphase slip, unequal temperatures, etc. In T. M. Shih, editor, *Numerical Properties and Methodologies in Heat Transfer: presented at the 2nd National Symp.*, Univ. Maryland, USA, pages 421-436, Washington, Sep 28-30, 1983. Hemisphere Publ. Corp.

Staedtke H., Franchello G. and Worth B., (1998). Towards a high resolution numerical simulation of transient two-phase flow. In *ICMF98 (1998)*.

Trambouze P., (1996). CFD applied to process engineering - An introduction. *Revue de L'Institut Francais du Petrole*, 51(2):199-203.

Versteeg, H. K. and Malalasekera, W. (2007). *An Introduction to Computational Fluid Dynamics, The Finite Volume Method*, Pearson Education.

White, F. M. (1974). *Viscous Fluid Flow*. McGraw-Hill. New York.

White, F. M. (2003). *FLUID MECHANICS (5TH EDITION)*. MCGRAW-HILL.

www.openfoam.org

Zhang, Zhi-rong, (2010). "Verification and Validation for RANS simulation of KCS container ship without/with propeller." *Journal of Hydrodynamics*, 22(5): 932-939.