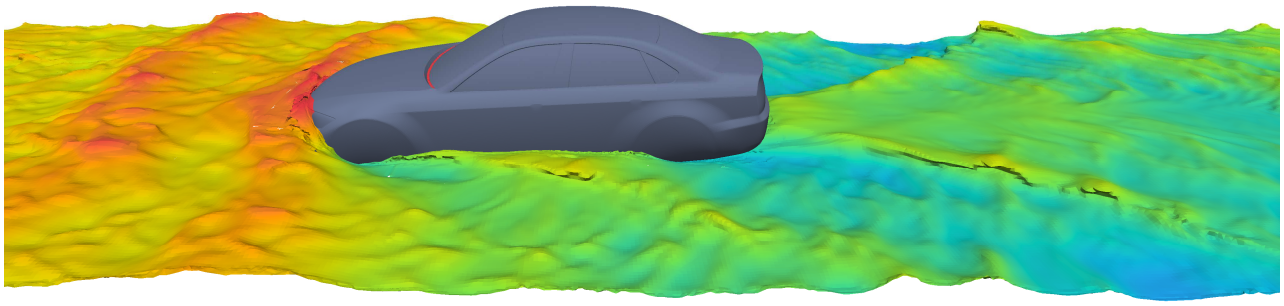
CHALMERS
UNIVERSITY OF TECHNOLOGY



CFD Modeling of Wading with Electric Vehicles

Development of a Simulation Method Using OpenFOAM

Master's thesis in Applied Mechanics

ELIN OLSSON

Department of Mechanics and Maritime Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2018

MASTER'S THESIS IN APPLIED MECHANICS

CFD Modeling of Wading with Electric Vehicles

Development of a Simulation Method Using OpenFOAM

ELIN OLSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mechanics and Maritime Sciences
Division of Fluid Mechanics
CHALMERS UNIVERSITY OF TECHNOLOGY

Gothenburg, Sweden 2018

CFD Modeling of Wading with Electric Vehicles
Development of a Simulation Method Using OpenFOAM
ELIN OLSSON

© ELIN OLSSON, 2018

Master's thesis 2018:04
Department of Mechanics and Maritime Sciences
Division of Fluid Mechanics
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone: +46 (0)31-772 1000

Cover:
External water surface obtained with the simplified vehicle geometry using OpenFOAM, visualized with an isosurface. On the surface, the water level contour is visible.

Chalmers Reproservice
Gothenburg, Sweden 2018

CFD Modeling of Wading with Electric Vehicles
Development of a Simulation Method Using OpenFOAM
Master's thesis in Applied Mechanics
ELIN OLSSON
Department of Mechanics and Maritime Sciences
Division of Fluid Mechanics
Chalmers University of Technology

Abstract

Due to increasing demands on the performance of electric vehicles (EVs), there is a desire to investigate the impact of tough situations on the vehicle, such as wading through water. As most previous wading studies are concerned with conventional internal combustion engine vehicles (ICEVs), automotive manufacturers are striving to expand their knowledge about wading with EVs. Wading is mainly examined through physical tests, but using computational fluid dynamics (CFD) for studying vehicle wading can provide a more rigorous examination as well as reduce cost and time required. One way to further decrease the cost is to use open source software.

In this study, a method for investigating wading with the open source CFD software OpenFOAM is developed. The generic DrivAer model is used, modified to resemble a model of a battery electric vehicle (BEV). The computer aided engineering (CAE) software ANSA is used for geometry cleanup and hexahedral volume meshing. The numerical method uses the volume of fluids (VOF) method for multiphase modeling and large eddy simulation (LES) for turbulence modeling. The study focuses on the water surface topology, water velocity, water flow through the engine compartment and forces acting on the car. Due to stability issues results are only obtained with a simplified vehicle geometry and not for the BEV model. Validation is done against a similar case in the commercial software STAR-CCM+. It is found that the results obtained for the water surface and velocity with the OpenFOAM method are similar to those obtained with STAR-CCM+, but that the lift forces on the car show large deviations.

To further investigate the effect of wading on EVs, simulations with the more detailed BEV geometry are conducted using STAR-CCM+. Results show that splashing is absent at low speed deep water wading, but that engine bay components are still exposed to large amounts of water. It is also found that the water pressure exerted on the battery pack and engine bay components at this velocity is insignificant.

It is concluded that the OpenFOAM method shows promising results, but further investigation of the solver settings is required in order to increase the stability when using complex geometries. Regarding wading with EVs, it is concluded that the impact of low speed wading on the vehicle is small. However, as the water ingress in the engine bay compartment is significant, the components here must be watertight or somehow protected.

Keywords: Electric vehicles, Wading, CFD, OpenFOAM, DrivAer, VOF, LES, STAR-CCM+

Preface

This report is the result of a Master's thesis project at the Department of Mechanics and Maritime Sciences at Chalmers University of Technology. The project was carried out at Technical Analysis at ÅF AB in Gothenburg between September 2017 and March 2018. The project was carried out in cooperation with NEVS AB in Trollhättan. The aim of the project was to investigate wading using open source CFD software, as well as further investigate wading with EVs. The project resulted in an expansion of the knowledge about wading with EVs as well as the outlines of a methodology for simulating wading with the open source software OpenFOAM.

Acknowledgements

I would like to thank those without whose help this project would not have been possible to conduct: First of all, I would like to thank my supervisor at ÅF Dr. Jan Östh, Group manager at ÅF Industry, for initiating the project and for all his time and support during the project.

I would also like to thank Dr. Mohammad El-Alti, Supervisor Aerodynamics and CFD at NEVS, for making this project possible, for valuable input regarding project aim, and for the opportunity to conduct numerical simulations at NEVS. Thanks also to Kristian Abdallah, Engineer CFD and Aerodynamics at NEVS, for helping me with running simulations, and to Daniel Egenvall, Design Engineer at NEVS, for designing components of the detailed BEV geometry.

Furthermore, I would like to thank everyone at the Fluid Mechanics Sections at ÅF Industry for great company and sharing of knowledge. Great thanks also to my examiner Professor Håkan Nilsson at Chalmers, and to Bercelay Niebles at Chalmers for help with the STAR-CCM+ license. Thanks also to Siemens and Beta CAE for providing student licenses to STAR-CCM+ and ANSA, respectively.

I would also like to thank my family for encouragement throughout my studies. Finally, thank you Max, for your endless support and patience.

Elin Olsson
Gothenburg, March 2018

Table of contents

Nomenclature

Acronyms

BEV	Battery Electric Vehicle
CAE	Computer Aided Engineering
CFD	Computational Fluid Dynamics
DNS	Direct Numerical Simulation
D-STAR	Detailed geometry STAR-CCM+ simulation
EV	Electric Vehicle
FCEV	Fuel-Cell Electric Vehicle
GUI	Graphical User Interface
ICEV	Internal Combustion Engine Vehicle
LES	Large Eddy Simulation
MAC	Marker And Cell
PHEV	Plug-in Hybrid Electric Vehicle
PID	Property Identifier
PISO	Pressure-Implicit with Splitting of Operators
PLDV	Passenger Light-Duty Vehicle
PMSM	Permanent Magnet Synchronous Machine
RANS	Reynolds-Average Navier-Stokes
SGS	Subgrid Scales
SIMPLE	Semi-Implicit Method for Pressure-Linked Equations
S-OF	Simplified geometry OpenFOAM simulation
S-STAR	Simplified geometry STAR-CCM+ simulation
TUM	Technical University of Munich
VOF	Volume Of Fluids

Physical quantities

A_{Ref}	reference area	m^2
$\bar{A}$	spatially filtered variable	-
A'	spatially unresolved variable	-
A^+	van Driest damping constant	-
C_l	lift coefficient	-
C_S	Smagorinsky coefficient	-
F_l	lift force	N
p	dynamic pressure	Pa
$\bar{S}_{ij}$	filtered rate-of-strain	1/s
$\|S\|$	characteristic filtered rate-of-strain	1/s
U	velocity magnitude	m/s
u_t	friction velocity	m/s
$\boldsymbol{u}$	velocity	m/s
y_w	distance to closest wall	m
y^+	dimensionless wall distance	-
α	volumetric phase fraction	-
Δ	filter length scale	m
δ_{ij}	Dirac delta function	-
μ	dynamic viscosity	Ns/m^2
ν	kinematic viscosity	m^2/s
ν_T	turbulent viscosity	m^2/s
ρ	density	kg/m^3
τ_{ij}	subgrid Reynolds stress	Pa
τ_w	wall shear stress	Pa
Co	Courant number	-

1

Introduction

The use of numerical simulations during the vehicle development process is increasing, especially in areas associated with time consuming and expensive testing procedures. One such area is vehicle wading, where the complex phenomena involved are difficult to study experimentally. Furthermore, there is a growing demand for vehicles using alternative energy sources, resulting in fast development of for example electric vehicles (EVs). Here, numerical studies of extreme situations like wading are crucial for reducing the development time. This chapter introduces the scope and context of the current project. It includes a background and summary of previous work which presents the relevance of the project, as well as the project purpose in terms of aims, goals and limitations. The chapter finishes with an outline of the report.

1.1 Background

With global warming and climate change as growing concerns, there is a desire to reduce the world's emissions of greenhouse gases. Emission reduction targets have been stated both nationally and internationally. For example, the Paris agreement on keeping global warming below 2 degrees celcius was signed by 125 countries in 2017 [1]. 23 percent of the global energy-related carbon dioxide emissions are caused by the transport sector [2]. The emission goals have made the automotive industry look for alternative energy sources such as biofuels and electricity. In 2016, the global number of EVs reached 2 million, and the global development taking place in the field suggests that the global stock of EVs can grow to 40 to 70 million in 2025 [3]. EVs here refer to electric passenger light-duty vehicles (PLDVs) and include battery electric vehicles (BEVs), plug-in hybrid electric vehicles (PHEVs) and fuel-cell electric vehicles (FCEVs). PLDVs refer to passenger cars and light trucks. The main issues for battery EV manufacturers are limited driving range, high battery cost and need for charging infrastructure [4]. However, progress within battery technology is made constantly. With growing popularity for EVs the manufacturers are also faced with increasing demands on the cars' performance. Customers not only seek small vehicles for inner-city transportation, but vehicles adapted for longer journeys and tougher conditions.

The term vehicle wading refers to the situation where a vehicle is driving through water. This occurs when a car is moving through deep puddles and floodings or crossing rivers. Wading can be categorized as either low speed driving through deep water or high speed driving through shallower depths, referred to as deep water wading and splash wading respectively. Figure 1.1.1 shows a car undertaking deep water wading. The amount

of previous studies on the impact of wading on EVs is limited, but some potentially problematic areas can be identified. The high-voltage system and battery pack found in many types of EVs can be hazardous in the case of water ingress. The vehicles are however equipped with various safety systems, for example to detect and avoid short circuiting in the high-voltage system. The battery pack is generally enclosed inside several layers of housing and sealing preventing from contact with water, although sensitive connections and pressure valves might still be exposed [5]. High pressure forces from the water can be problematic for battery pack components made out of weaker materials. Another problematic area is the air intake for the battery and motor cooling systems, where water ingress may occur. Most EV manufacturers use sealed liquid cooling systems instead of air cooling and can therefore omit air intakes and make the cooling system watertight [6]. A general problem occurring when a vehicle is entering deep water is the large lift force, which can affect the maneuvering.

Figure 1.1.1: *A car wading through deep water at a test facility. Figure obtained from [7], published with permission.*

Due to these problems, automotive manufacturers are concerned with expanding the knowledge within this field. Previous wading studies are mainly concerned with internal combustion engine vehicles (ICEVs), where flooding of the engine air intake and water damage on engine bay electronics are the main concerns [8]. But since the components in the propulsion system of an EV differ significantly from those of an ICEV these problems do not apply to the same extent.

Wading is mainly examined through physical tests where cars are driving at different speeds through water of different depths. Cameras outside and inside the car are used to film the water surface and water ingress in the engine bay. Water ingress can also be measured using porous materials whose water content is measured after the tests. Damage of under-body parts is investigated after the tests. However, none of these methods provide a complete understanding of the wading situation outside of and inside the vehicle. These tests also take place at a late stage in the design process. In case redesign of the vehicle is required after the wading tests time and cost increases severely for the development process.

Numerical simulations have reduced time consumption for many stages of the automotive

design process. Using computational fluid dynamics (CFD) for simulating vehicle wading could provide more rigorous examination of the wading phenomena as well as improve the cost efficiency of this part of the design process. Splash wading is a rapidly transient behaviour and causes the formation of small structures such as jets and droplets [8]. CFD simulations of such situations would require a very fine mesh as well as a short time step, making this a computationally expensive and complicated task. Therefore deep water wading is better suited for a numerical study and would produce better results using reasonable resources. Deep water wading will hereafter be denoted "wading".

1.2 Previous work

The amount of previous work regarding numerical studies of vehicle wading is limited. A number of studies, where a numerical method for investigating wading with STAR-CCM+ is developed, have been performed at Jaguar Land Rover by Khapane et al. [8, 9]. A first study investigates challenges related to the development of virtual wading investigations [9]. A simulation method is developed by first studying a simple block geometry both experimentally and numerically and thereafter continuing with a more complex car geometry. The investigation focuses on pressure on underbody components and the only presented results regard the fluid-structure interaction. A second study uses the developed model and present results for water entry in the engine bay as well as forces on the underbody components [8]. Focus lies on understanding the water ingress and splashing and some comparison is done with physical wading test results. However, the presented results are too case specific to be used for comparison in the current project.

A thorough study of design parameters affecting the engine bay water entry, also using STAR-CCM+, has been performed by Makana et al. [10]. The simulation setup is similar to the one used in the current project. The results of the study includes the path of the water in the engine bay area as well as the water surface outside of the car and are therefore valuable for this project.

An early study of vehicle wading was performed by Zheng et al. [11]. A detailed car model is used and simulations are also here conducted in STAR-CCM+. Results are presented for pressure and water volume fraction on the firewall of the car as well as water surface inside the engine bay area. No previous work on wading simulations with OpenFOAM has been found. Furthermore no studies regarding EV wading have been encountered.

1.3 Project aim

As mentioned in the previous section there exists only a limited amount of numerical studies of vehicle wading. Moreover, there is little information about the effect of wading on EVs and how it differs from ICEVs. This project aims to further investigate wading using CFD to learn more about the phenomenon and how to study it numerically. Focus will be on expanding the knowledge about EV wading with results obtained in this project. The wading study will focus on the water level outside the car, the water flow through the engine compartment and around the battery pack, and the forces acting on the car.

Additionally, a method for solving wading problems using the open source CFD software OpenFOAM will be developed and evaluated. This encompasses an analysis of different methods available and the selection of which to use, as well as an evaluation of the obtained results. With increasing trust in the software, OpenFOAM is gaining popularity also for industrial use. Since numerical analysis of vehicles is already affiliated with high costs, the possibility to use free software is of high interest for the automotive industry. The software is also highly flexible and adaptable to solve specific problems. In this project a simulation procedure for vehicle wading with OpenFOAM will be developed and used for conducting wading simulations. Simulations will also be conducted using the commercial software STAR-CCM+ and the results will be compared. The aim is to investigate the ability of the OpenFOAM method to simulate wading.

1.3.1 Project objectives

The project aim can be formulated into three objectives:

- Study wading and how it affects EVs and contribute with knowledge in this field.
- Develop a method for studying vehicle wading using the open source software OpenFOAM.
- Evaluate the ability of OpenFOAM to solve wading problems by comparing with results from STAR-CCM+, and previous studies.

1.4 Limitations

- One detailed geometry representing a model of a BEV and one simplified geometry will be used in this project. Improved designs will not be investigated.
- Only deep water wading at low speed with one depth and one velocity will be studied.
- The project time is limited to 20 weeks.
- Computational resources are limited to those available at ÅF and NEVS.

1.5 Outline

The introduction is followed by a description of BEVs and some BEV concepts that are interesting from a wading point of view in Chapter 2. Thereafter, the relevant theory for the numerical study is presented in Chapter 3. The geometry used in the simulations is described in Chapter 4. The OpenFOAM methodology is presented in Chapter 5 and the STAR-CCM+ methodology is presented in Chapter 6. In Chapter 7 the results of the simulations are presented, compared and discussed, followed by a summary and final conclusions together with recommendations for future work in Chapter 8.

2

BEV Concepts

The term EV covers a wide range of technologies, where BEV, PHEV and FCEV are the main types [12]. The different types of EVs differ in many ways, not least in stage of development. Most FCEVs are still at early demonstration stages while there already exist BEVs and PHEVs on the market that, depending on the customer's needs, are sufficient alternatives to ICEVs. Drawbacks such as shorter driving range, lack of charging infrastructure and higher costs are still present [4]. However, due to the increased focus on environmental issues major research is done within this field. The vehicle studied in this project is a BEV and therefore this chapter will focus on this type of EV.

A BEV is propelled by an electric motor, which is powered by a battery [12]. The key components of a BEV are the battery, power converter, electric motor and transmission, where the latter consists of gear and drive shaft [4]. An overview of the powertrain can be seen in Figure 2.0.1

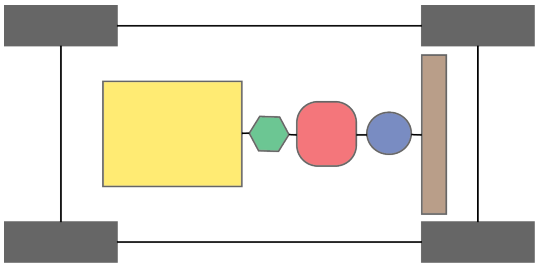


Figure 2.0.1: *Schematic of the driveline of a BEV, with battery (yellow), power converter (green), electric motor (red), gearbox (blue) and drive shaft (brown).*

Chemically stored energy is transported from the battery to the electric motor via the DC/AC power converter. The power from the motor is transferred to the drive shafts through a gear. The power converter is regulated by a control system which controls frequency and magnitude of the voltage depending on the type of motor and the driver's actions [13].

Different types of motors have been used in BEVs, however today the permanent magnet synchronous machine (PMSM) is most commonly used [6]. This motor type consists of permanent magnets on a rotor and a stator powered by an AC voltage source. The AC voltage source in this case consists of the battery and the power converter. In order to increase efficiency and reduce wear on components cooling of the motor is required. The dominating cooling method used for BEV motors is liquid cooling [6]. A cooling liquid consisting of almost equal parts water and ethylene glycol is used to prevent overheating,

freezing and corrosion. The cooling liquid is enclosed in a channel outside of the electric motor.

Cooling is also required for the battery pack to maintain optimal operating temperature. Operating outside of the optimal temperature range can worsen the performance and reduce the lifetime of the battery. Cooling also prevents fire in the battery pack. A few existing BEV models use air cooling and passive cooling. Liquid cooling is however most commonly used, using water or a blend of water and ethylene glycol [6]. The use of liquid cooling systems for both the motor and battery pack eliminates the need for air intakes, which reduces the risk of water entry in the engine bay in a wading situation.

All current BEVs use lithium-ion batteries of different types [6]. Multiple battery cells connected in series and sometimes also in parallel are required to provide the desired voltage and energy content [5]. These cells are located inside the battery pack. The battery pack, which often makes up a large part of the volume and weight of the vehicle, is commonly placed underneath the car to give a low center of gravity. To protect the battery cells from the sometimes harsh conditions outside of the car causing damage and aging, they are enclosed inside several layers of housing [5]. This housing is made water- and dust tight using advanced seals and connections. Therefore contact with water during wading should be impossible. However, the forces from the water on the battery pack during wading could be a risk. As the battery pack significantly contributes to the vehicle weight and size, there is a desire to use lightweight designs and materials for the housing [5]. If lighter materials are used the strength of the battery pack might be reduced, and high local water pressure could cause material failure.

3

Theory

In this chapter, the relevant theory for understanding the report is introduced. First, the wading situation and related forces are explained. Thereafter, the numerical methods used for modelling the flow are described.

3.1 Wading theory

Wading can physically be described as a multiphase flow situation involving two immiscible fluids, water and air. Each phase make up a large domain and between them a sharp interface is formed. The situation can therefore also be referred to as a separated two-phase free surface flow. Deep water wading is characterized by a large bow wave forming in front of the vehicle, leading to a higher water surface here compared to further back along the vehicle [8]. The height of the so called wading line is thus decreasing towards the vehicle's rear. Typical speeds are below 10 km/h and the initial water depth is up to 0.5 m. At such depths the wheels are partially submerged in water, and the wheel rotation causes splashing of water onto underbody components as well as into the engine bay area [8]. The above mentioned water behaviour is expected to occur in the wading simulations.

3.1.1 Lift force

The large volume of water displaced by the vehicle as it enters the water results in a significant lift force on the vehicle. This makes the vehicle more sensitive to side forces in for example river crossing situations. The lift force exerted on the vehicle is calculated as

$$F_l = 0.5 C_l \rho U^2 A_{Ref} \tag{3.1.1}$$

where C_l is the lift coefficient, ρ is the density, U is the magnitude of the flow velocity and A_{Ref} is the reference area (projected frontal area of the car) [14]. The lift coefficient is calculated numerically during the simulation. Previous vehicle aerodynamic experiments have shown that the ratio between the lift and the drag coefficient (lift/drag ratio) is about 0.23 [15]. The lift coefficient is therefore lower than the drag coefficient when only air flow is present in the domain. However, the experiments were conducted with higher velocities than in the current project. For wading simulations, a lift/drag ratio well above 1 is expected due to the lift force caused by the water.

3.2 Free surface methods

The two main groups of methods for solving problems in fluid dynamics are Lagrangian and Eulerian methods [16]. In Lagrangian methods the fluid properties are calculated in a coordinate system that is moving with the fluid particles. In this method each fluid particle is traced. In Eulerian methods the fluid properties are calculated in a fixed coordinate system, and the flow is observed at fixed coordinates. This requires the fluxes between the coordinates to be calculated. The fixed Eulerian coordinate system corresponds to the fixed mesh used in many numerical methods for fluid dynamics.

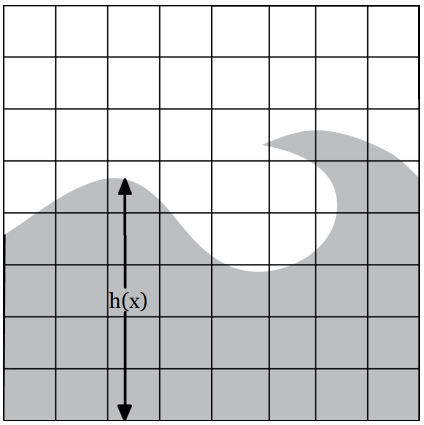
Separated free surface flows can be modelled using several different Eulerian methods. These can be classified as surface methods (interface-tracking) or volume methods (interface-capturing).

Surface methods explicitly track the interface between the phases. One surface method uses height functions to represent the surface with a height value at each surface coordinate and time. This method can however not have more than one height value for one coordinate point, making it impossible to represent complex waves or bubbles. Another surface method distributes connected marker particles on the interface and tracks these. The method is sensitive to the distance between the particles, and also struggles with complex surfaces like the height functions method [17]. Surface methods do in general give a sharp representation of the interface, but suffer from the mentioned drawbacks.

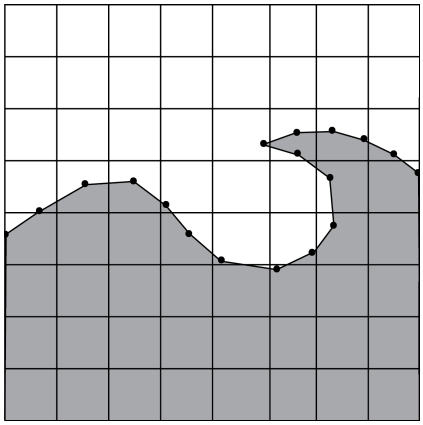
An alternative is to use volume methods, which instead track the two fluid volumes separated by the interface. This enables representation of complex surfaces, however to obtain a sharp surface resolution computationally expensive additional steps are required [18]. A common volume method is the marker and cell (MAC) method [18]. It tracks one of the fluid phases, which is represented by Lagrangian fluid particles that are distributed over the phase volume and moved through the Eulerian mesh. The vast number of tracked variables makes this method significantly more computationally expensive than the surface methods [17]. A computationally inexpensive option that still can handle complex surfaces is the volume of fluids (VOF) method, which tracks the volume fraction of one of the phases in each cell. The above described Eulerian surface and volume methods are illustrated in Figure 3.3.1.

3.3 VOF method

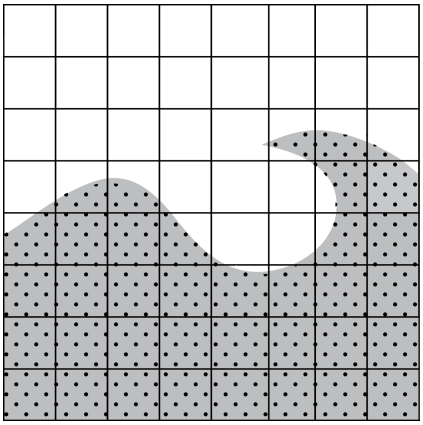
The VOF method is an Eularian volume tracking method where a step function is used to mark the location of the phases [17]. This work is limited to two-phase flows with water as the tracked phase and air as the second phase. The step function α marks the volume fraction of the tracked phase in a control volume, so that $\alpha = 1$ corresponds to a control volume entirely occupied by water and $\alpha = 0$ corresponds to a control volume not containing any water. The value of α is averaged in each mesh cell. The interface between the phases is found in cells where $0 < \alpha < 1$.



(a) *Surface method: height function tracking one value for the height of the surface at each surface coordinate.*



(b) *Surface method: connected marker particles distributed accross the surface.*



(c) *Volume method: MAC with marker particles distributed in the tracked phase volume.*

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0.1	0.4	0.5	0.3
0.2	0.5	0.5	0.1	0	0.1	0.7	0.9
0.8	1	1	0.7	0.1	0.2	0.8	1
1	1	1	1	0.9	0.9	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

(d) *Volume method: VOF with a value for the volume fraction of the tracked phase in each cell.*

Figure 3.3.1: *Illustrations of the Eulerian surface methods (interface-tracking) with height functions and marker particles and Eulerian volume methods (interface-capturing) MAC and VOF.*

The function α makes it possible to use only one set of equations to describe local flow properties in the entire flow domain, instead of one set of equations for each phase. Fluid properties are calculated using α as a weight. If for example ρ_{water} and ρ_{air} are the densities of the two phases, the density in the entire domain ρ can be described with [18]

$$\rho = \alpha\rho_{water} + (1 - \alpha)\rho_{air} \tag{3.3.1}$$

Similarly, for the dynamic viscosity μ

$$\mu = \alpha\mu_{water} + (1 - \alpha)\mu_{air} \tag{3.3.2}$$

3.3.1 Governing equations

The flow studied in this project is considered incompressible and the fluids are considered Newtonian. The continuity equation can then be expressed as

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{3.3.3}$$

where $\boldsymbol{u}$ is the fluid velocity. The momentum equation is expressed as

$$\frac{\partial u_i}{\partial t} + \frac{\partial (u_i u_j)}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial x_j \partial x_j} \tag{3.3.4}$$

where p is the dynamic pressure and ν is the kinematic viscosity.

The VOF method adds one governing equation for the transport of the volume fraction α, the advection equation [18]

$$\frac{\partial \alpha}{\partial t} + \frac{\partial (\alpha u_i)}{\partial x_i} = 0 \tag{3.3.5}$$

3.4 Turbulence modeling

Turbulent flows are characterised by being highly unsteady, having three-dimensional fluctuations and containing a wide range of time and length scales [14]. The vehicle wading situation leads to a flow that is turbulent, especially inside the engine bay and at the wheels. To account for the turbulence in the numerical simulation, a turbulence model is required. In so called direct numerical simulation (DNS), all present flow scales are resolved. DNS is therefore computationally expensive and require very fine discretization of the domain. This is not possible for large-scale engineering problems like wading. Using common Reynolds-averaged Navier-Stokes (RANS) turbulence models for free surface flows is popular, but give rise to problems with stability and overpredicted turbulence levels [19]. Large eddy simulation (LES) has shown promising results when used in combination with free surface methods, including VOF [19, 20].

3.4.1 LES

In LES the large scale eddies are directly resolved and the small scales of the flow are modelled to reduce the computational cost [21]. Since the large scale motions are responsible for the main transport of fluid properties, an accurate calculation of these is more important than capturing all small scale motions. In LES, a spatial filter is applied to determine which scales to resolve. The filtering decomposes a variable A into a resolved (filtered) component $\bar{A}$ and a residual non-resolved component A'. The scales not resolved by the grid are called subgrid scales (SGS) [21]. Figure 3.4.1 shows a schematic illustration of how large scale eddies are resolved by the grid whereas small scale eddies are not resolved.

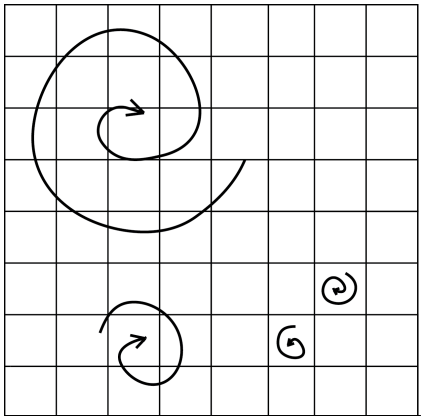


Figure 3.4.1: *Illustration of the spatial filtering in LES.*

Since the small scale motions are filtered out, the filtering leads to a volume average of the variables. Applying the volume averaging to the governing equations in Section 3.3.1 and rewriting leads to the LES governing equations. The filtered velocity field and dynamic pressure field are denoted $\bar{\boldsymbol{u}}$ and $\bar{p}$ respectively, giving the following continuity equation and momentum equation

$$\frac{\bar{u}_i}{\partial x_i} = 0 \tag{3.4.1}$$

$$\frac{\partial \bar{u}_i}{\partial t} + \frac{\partial(\bar{u}_i \bar{u}_j)}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \bar{p}}{\partial x_i} + \nu \frac{\partial^2 \bar{u}_i}{\partial x_j \partial x_j} - \frac{\partial \tau_{ij}}{\partial x_i} \tag{3.4.2}$$

where τ_{ij} is the subgrid Reynolds stress tensor or SGS stress. It takes into account the interactions of the smaller modelled scales and the larger resolved scales [22]. This term requires a model in order to solve the filtered governing equations.

3.4.2 The Smagorinsky model

The earliest and simplest model for the SGS stress is the Smagorinsky model [23]. It is a so called eddy-viscosity model in which the SGS stress is decomposed according to

$$\tau_{ij} - \frac{1}{3}\delta_{ij}\tau_{kk} = -2\nu_T \bar{S}_{ij} \tag{3.4.3}$$

where δ_{ij} is the Dirac delta function, ν_T is the turbulent viscosity (eddy viscosity) and $\bar{S}_{ij}$ is the filtered rate-of-strain tensor [21]. $\bar{S}_{ij}$ is expressed as

$$\bar{S}_{ij} = \frac{1}{2}\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right) \tag{3.4.4}$$

ν_T that appears in the model for the SGS stress also requires a model. This takes the following form

$$\nu_T = (C_S \Delta)^2 |S| \tag{3.4.5}$$

where C_S is the Smagorinsky coefficient, Δ is the filter length scale and $|S| = (2\bar{S}_{ij}\bar{S}_{ij})^{1/2}$ is the characteristic filtered rate-of-strain. Here, the simplicity of the model is clear with only one modelling parameter, C_S. Another advantage is its stability.

3.4.3 Wall treatment

The original Smagorinsky model gives inaccurate results in the near-wall region [24]. In the laminar region near the wall, ν_T is non-zero for Smagorinsky modelling resulting in too high dissipation. Therefore the van Driest damping function is added to the model, giving a new expression for ν_T:

$$\nu_T = (C_S\Delta)^2[1 - \exp(y^+/A^+)]|S| \tag{3.4.6}$$

where A^+ is a constant and y^+ is the dimensionless wall distance, calculated as

$$y^+ = \frac{u_t y_w}{\nu} = \frac{\sqrt{\tau_w/\rho}y_w}{\nu} \tag{3.4.7}$$

where u_t is the friction velocity, y_w is the distance to the closest wall and τ_w is the wall shear stress. This damping function is zero at the wall where $y = 0$.

A requirement for resolving the flow in the boundary layer is that the first grid point should be placed in the viscous sublayer, where the dimensionless wall distance is in the range $0 < y^+ < 1$ [21]. If that is fulfilled, the transport equations above are solved for all cells, also the cells adjacent to the wall. Another option is to use so called wall functions, when the grid near the wall is too coarse. This method applies robust boundary conditions at the first grid point to avoid solving the transport equations in the near-wall region [21]. The boundary conditions are calculated using the so called log-law in the region between the wall and the first grid point. This requires that the first grid point is placed inside the log-law region, where the flow is assumed to be independent of viscosity and $30 < y^+ < 200$. If the first grid point is instead placed in the viscous sublayer when using wall functions results will be inaccurate.

3.5 Courant number

The Courant number is defined as

$$Co = \frac{U\Delta t}{\Delta x} \tag{3.5.1}$$

and describes how many grid cells a fluid particle is transported across during one time step. By limiting the Courant number the stability of the solution can be improved.

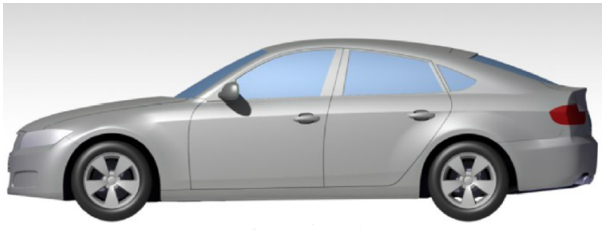
4

Geometry

In this chapter, the car geometry used in the current project is described, by presenting the original model together with modifications. Two geometries are described, one more detailed with wheels, side mirrors and open engine compartment, and one simplified without wheels and side mirrors and with an entirely closed, smooth body. The computational domain used for the numerical study is also described, as well as the method used for surface meshing.

4.1 DrivAer model

The car geometry used in this project is the generic DrivAer model developed by the Institute of Aerodynamics and Fluid Mechanics at the Technical University of Munich (TUM) [25]. It was developed as an alternative to the previous types of models used for aerodynamic investigation of cars. DrivAer is based on the simplified geometries of production cars from two major automotive companies. The model is available with different design configurations for many of its parts, for example the underbody and rear end. The model and different rear ends can be seen in Figure 4.1.1.



(a) *Fastback rear end.*

(b) *Notchback rear end (left) and estate back rear end (right).*

Figure 4.1.1: *DrivAer model with different rear ends. Figure obtained from [25], published with author's permission.*

Academic studies often use much simplified traditional models such as the Ahmed body. Even though there is a vast amount of research and data available for these models, they differ too much from a realistic car to make these results entirely applicable to automotive development. Automotive companies on the other hand use detailed models of their real production cars. These models can clearly produce realistic results, however they are difficult to access for the general public. Therefore, not enough research exists to enable validation. The open source geometry model DrivAer has so far proven to be a successful alternative to these models and several experimental and numerical studies have been conducted, see for example [26] and [27].

The model was developed for aerodynamic studies and most of the available work using DrivAer is focusing on air flow. The model is however also suitable for the study of water flow around the car and inside the engine bay.

4.1.1 Geometry modifications

The original DrivAer is a model of an ICEV. It is therefore necessary to perform modifications to the original DrivAer model to obtain a model of a BEV. The starting point for the modifications is the notchback DrivAer model with open engine compartment, smooth underbody and smooth wheels with open rims. Side mirrors and engine bay are also included. The main difference between a BEV and an ICEV is the electric driveline and battery used instead of the combustion engine driveline. A simplified electric driveline and battery pack is added to the model. The driveline consists of the electric motor, power converter, gearbox and drive shafts and can be seen together with the battery pack in Figure 4.1.2.

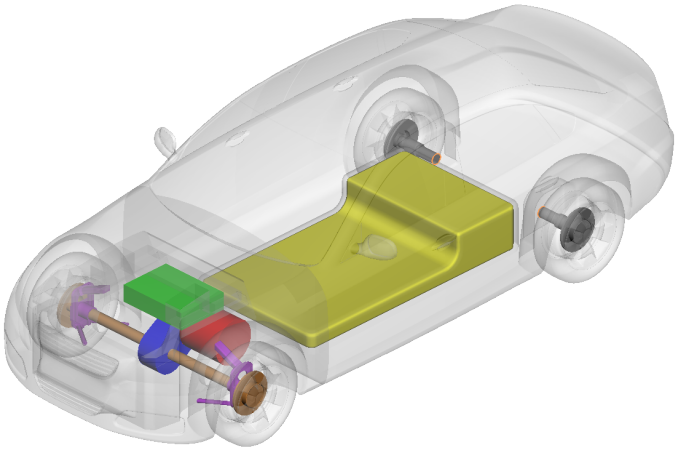


Figure 4.1.2: *Electric driveline consisting of battery pack (yellow) and electric driveline, consisting of electric motor (red), power converter (green), gearbox (blue), front and rear drive shafts (brown and grey) and suspension (purple) placed inside the transparent notchback DrivAer model.*

The motor, power converter and gearbox are placed inside the engine bay shown in

Figure 4.1.3. The engine bay included in the original DrivAer model is reconstructed to remove walls of zero thickness and sharp angles. The battery pack is placed under the car underbody. The smooth underbody included in the original DrivAer model is reconstructed to fit the battery pack and to form a firewall preventing water entry in the passenger compartment. To obtain a realistic flow from the engine bay area holes are added for the drive shafts in the front wheelhouses and a gap is left in the underbody in front of the firewall. The underbody and firewall can be seen in Figure 4.1.4.

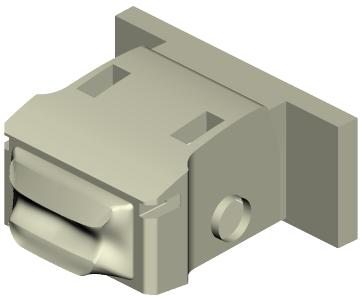


Figure 4.1.3: *Engine bay inside which the electric motor, power converter and gear box are located.*

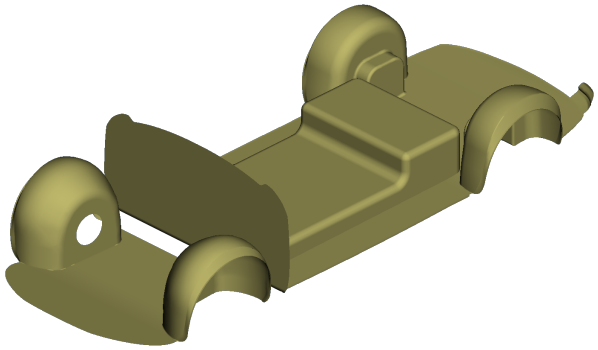


Figure 4.1.4: *Modified underbody with firewall. The drive shaft holes in the front wheelhouses and the gap in front of the firewall are clearly visible.*

Based on a previous study of the under-hood cooling flow of EVs the top grille is closed, leaving the lower grille as the only inlet on the vehicle front [28]. The grilles are directly connected to the engine bay. The complete DrivAer geometry used in the project can be seen in Figure 4.1.5. In this figure, parts of interest are highlighted.

After obtaining the geometry, cleanup of the model is required in order to prepare it for CFD simulations. The cleanup is done using the computer aided engineering (CAE) software ANSA. The procedure includes removing overlapping surfaces, removing long, thin needle surfaces and connecting adjacent edges to close gaps. If not removed, problems might arise near these features during the mesh generation. Some objects, including the firewall and power converter, require reconstruction as they are overlapping with other objects. The surfaces of the DrivAer model are of zero thickness.

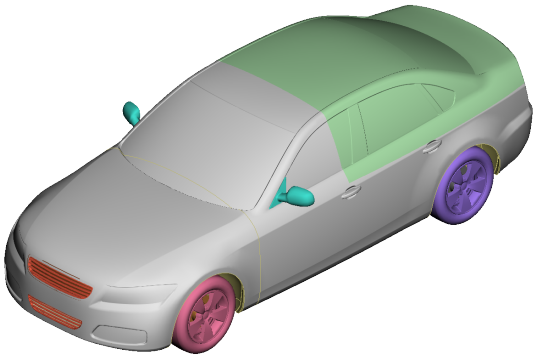


Figure 4.1.5: *Complete DrivAer geometry with notchback rear end. The front grilles (orange), wheels (pink and purple), side mirrors (light blue) and notchback rear end (light green) are highlighted.*

Zero-thickness walls which face the computational domain on both of its sides are problematic when generating the volume mesh. The geometry is therefore rebuilt to avoid such surfaces, as mentioned for the engine bay. The original model also contains several areas with sharp concave corners. In such areas it is difficult to obtain a volume mesh of good quality, therefore some parts are reconstructed to remove these areas. Geometry cleanup and rebuilding takes place during the meshing procedure as well, when problematic areas are detected.

4.1.2 Simplified geometry

Apart from the detailed DrivAer geometry described above, also a simplified car geometry is prepared. A simplified version of the DrivAer model with notchback rear end is available from TUM. This model is entirely closed and therefore has no engine bay. It has neither wheels nor side mirrors and the underbody is flat. Cleanup is preformed in the same way as for the detailed model, but as it is a simpler geometry with no modifications, the cleanup procedure is less extensive. The simplified geometry can be seen in Figure 4.1.6.

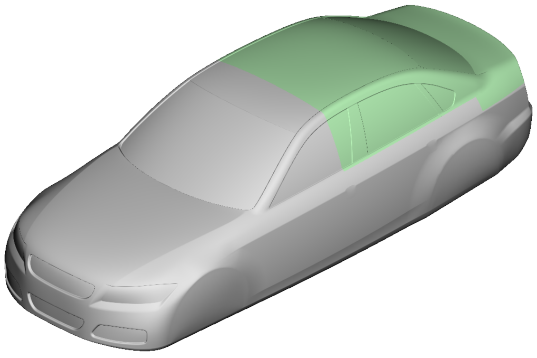


Figure 4.1.6: *Simplified DrivAer geometry with notchback rear end. The notchback rear end (light green) is highlighted.*

4.2 Computational domain

The computational domain is bounded by a rectangular box around the car. It is made long enough to fit interesting phenomena on the surface in front of and behind the car. Since this study is focused on the behaviour of the water surface inside and near the car, the height and width of the bounding box can be smaller than in aerodynamic studies. The tires of the detailed geometry are not deformed where they meet the road as would be the case in a real life situation. To avoid a sharp angle where the circular tire is touching the road a small domain is created around the bottom of each tire, see Figure 4.2.1. The bounding box can be seen in Figure 4.2.2. The inlet boundary of the bounding box is divided into one inlet for air and one for water, which is visible in Figure 4.2.2. A bounding box with the same dimensions is created for the simplified geometry.

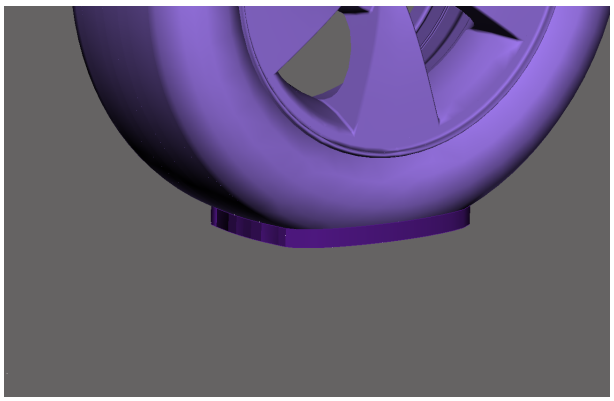


Figure 4.2.1: *The domain connecting the tires to the road and removing the sharp angle between tires and road.*

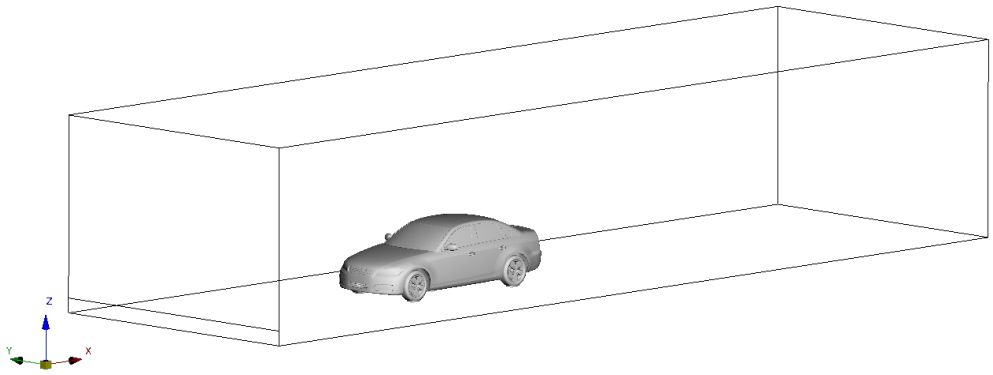


Figure 4.2.2: *The bounding box constituting the computational domain with the detailed car model inside it.*

4.3 Surface meshing

For the OpenFOAM simulations the volume mesh is prepared using ANSA and for the STAR-CCM+ simulations the volume meshing is prepared using the native mesher in

STAR-CCM+. Both volume meshing methods require an initial surface mesh on the car and the computational domain. This surface mesh is generated in ANSA. The surface mesh is generated in a similar manner for both the detailed and the simplified geometry. Different parts of the car are meshed separately to ensure that the local mesh has suitable size and is of good quality. First, mesh nodes are generated along the edges of the geometry using the ANSA function "Spacing - Auto CFD". For the car, the spacing is set to have a minimum value of 2 mm and a maximum value of 16 mm. The spacing is smaller for parts including small features, such as the lower grille, and larger on large flat surfaces of the car, such as the roof. A triangular surface mesh is then generated on the car surfaces using the ANSA function "Mesh generation - CFD".

Before generating a surface mesh on the bounding box, mesh refinement boxes are added to the geometry. It is desired that the volume mesh is finer in regions of interest and regions containing small features, such as around the car and around the water surface. One box is created around the car geometry and one box is created to contain the water volume. The cell size is set to 32 mm inside the larger box around the water volume and to 64 mm inside the smaller box around the car. The refinement boxes can be seen together with the detailed car model in Figure 4.3.1. For the bounding box the spacing is set to have a minimum value of 32 mm and a maximum value of 256 mm. A triangular surface mesh is generated on the bounding box in the same way as for the car and the cell sizes of the refinement boxes are reflected on this surface mesh.

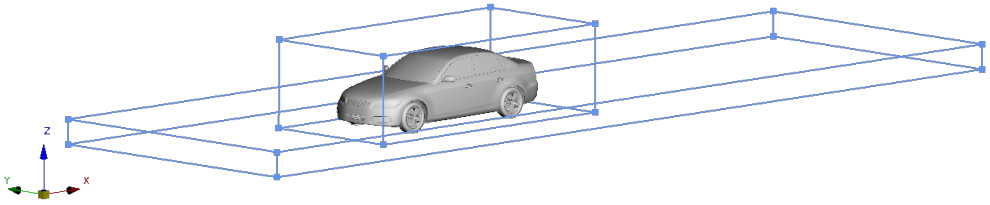


Figure 4.3.1: *The boxes used for refining the mesh around the car and the water domain together with the detailed car model.*

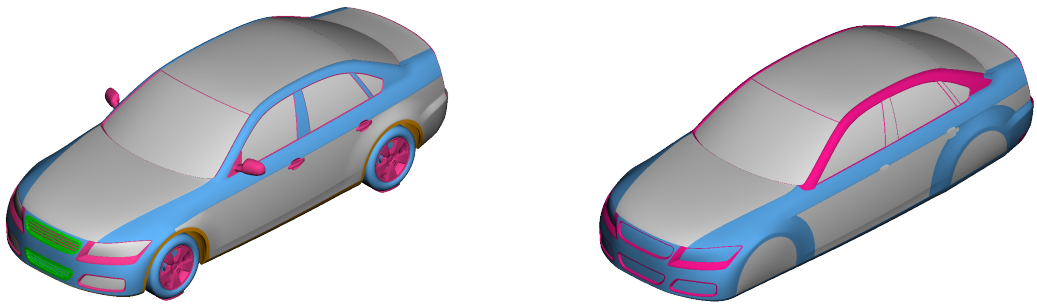
The quality of the surface mesh is checked with ANSA. For the surface mesh the quality criterion is that the cell skewness should be below 0.7. When this is fulfilled for all surface cells a good quality mesh is obtained. Automatic reshaping is used to improve violating cells. ANSA is also used to check that the mesh is free from intersecting cells, and that the surface mesh is watertight. The computational domain must be separated from the volumes outside of the domain by the surface mesh.

It is worth noting here that the quality of the surface mesh is less important for the volume generation in STAR-CCM+, where the surface mesh from ANSA only is used for detecting the surface and the computational domain.

4.4 Property regions

In ANSA so called property identifiers (PIDs) are used to assign different properties to different regions. The clean geometry is divided into regions with different PIDs to enable different boundary conditions and mesh refinement. The PID regions on the bounding box are based on boundary conditions and are also assigned a refinement level.

The entire surface of the car has the same boundary condition, wall, and the PID assignment on the car is only used for mesh refinement purposes in STAR-CCM+. The surface mesh generated in ANSA serves as the basis for the ANSA generated volume mesh, and therefore the PID assignment on the car is irrelevant to ANSA. However, the native mesher in STAR-CCM+ remeshes the surface and during this process the different PID regions on the surface are detected. The PID regions on the car are assigned a refinement level based on the size and curvature of the features to be captured. The refinement level is set as relative to a defined base size. The PID regions on the two car models can be seen in Figure 4.4.1.



(a) (b)

Figure 4.4.1: *PID region assignment on the (a) detailed and (b) simplified car model surfaces. The refinement level for the different regions are 100 % (grey), 75 % (orange), 50 % (blue), 25 % (pink) and 12.5 % (green) of the base size.*

5

OpenFOAM Methodology

This chapter presents the methodology used for conducting wading simulations with OpenFOAM. The methodology consists of volume meshing in ANSA, setup of the solver and selecting boundary and initial conditions. In OpenFOAM two simulation cases are prepared with identical solver settings and conditions, one with the more detailed BEV geometry and one with the simplified geometry. However, due to limitations in computational resources and project time, a simulation is only conducted with the simplified geometry. This simulation case is referred to as simulation S-OF (Simplified geometry OpenFOAM simulation).

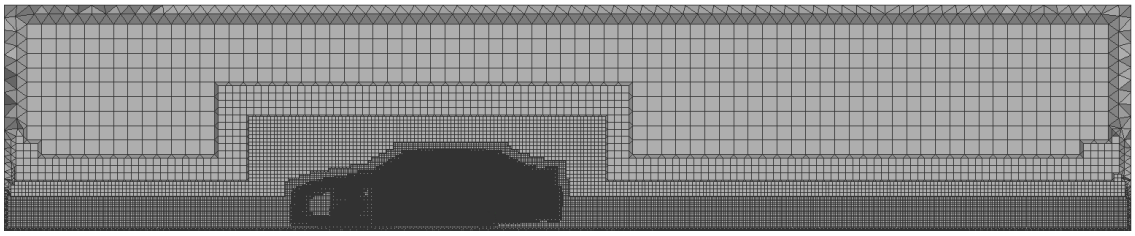
5.1 Volume meshing

After obtaining a good quality surface mesh on the geometry, a hexahedral volume mesh is generated in ANSA. OpenFOAM can operate on any type of mesh, however for free surface flows such as the flow in a wading simulation, hexahedral meshes are advantageous for reconstructing the surface during the post-processing [29]. The hexahedral mesh is generated using the ANSA function "Hexa interior". This mesher generates a structured hexahedral mesh in the main volumes, with pyramid and tetrahedral elements in transition zones between cell sizes and where the volume mesh connects to the surface mesh.

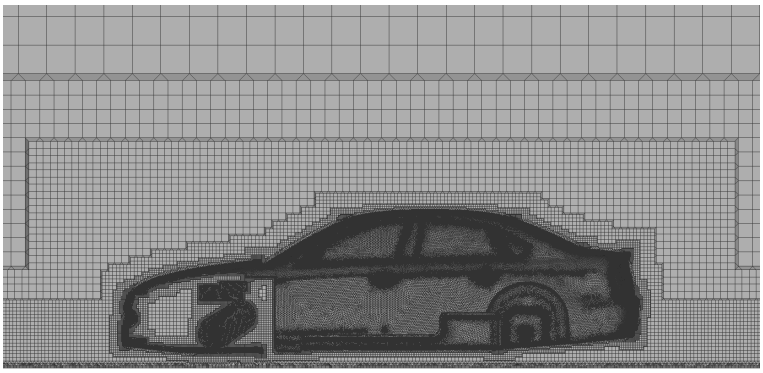
Before generating the hexahedral volume cells, prism cell layers are generated on the parts of the surface mesh corresponding to solid walls. This is to accurately capture the boundary layer at the walls. The prism layers consist of cells that are thin compared to their surface area. It is desired to have a first layer height in the range $0 < y^+ < 1$. Three prism layers are generated with layer height proportional to local surface cell size. These settings are chosen based on experience and the y^+ condition is controlled after the simulation. All surfaces of the car are considered solid walls. For the bounding box, layers are only generated on the road surface. It is due to the pyramids generated by "Hexa interior" at the surface that a triangular surface mesh and triangular prism layer cells are used.

Areas where the layer generation is problematic are automatically excluded. Here, the transition zone with pyramids connect directly to the triangular surface mesh and to the exposed sides of the layers. Before continuing with the volume mesh it is ensured that none of the cells in the prism layers are intersecting.

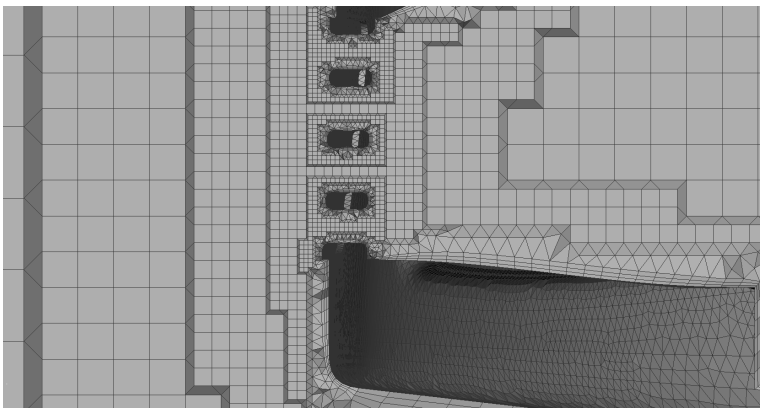
The final volume mesh for the detailed car geometry contains about 36 million cells and can be seen in Figure 5.1.1.



(a) *Entire volume mesh*



(b) *Zoom in of mesh around the car*



(c) *Zoom in of mesh around the lower grille*

Figure 5.1.1: *The hexahedral volume mesh with prism layers generated in ANSA for the detailed geometry, at the middle cross section plane in y-direction. Inside the car the triangular surface mesh is visible.*

The quality of the volume mesh is checked both in ANSA and in OpenFOAM after the mesh has been exported. The quality criteria for the volume mesh include that the cells should have positive volume, low aspect ratio, low skewness and low non-orthogonality. The obtained volume mesh contains a very low number of violating cells (below 400 in the ANSA check and below 200 in the OpenFOAM check) and is therefore considered a good mesh. No cells of zero or negative volume, which can lead to solver termination, are present.

As mentioned, the final volume mesh is exported from ANSA to OpenFOAM. It is possible to export the mesh directly in OpenFOAM format, making the process simple. It is also possible to prepare all case files required by OpenFOAM in ANSA. However, due to a

mismatch between the software versions, only the mesh files from ANSA are used whereas the rest are created in OpenFOAM.

5.1.1 Simplified geometry volume mesh

A volume mesh is generated in the same manner for the simplified car geometry in ANSA, using the same meshing function with the same cell sizes, mesh refinement boxes and prism layer settings. The mesh contains about 14.6 million cells and can be seen in Figure 5.1.2. Apart from not having a volume mesh inside the engine bay area, the volume mesh for the simplified geometry is very similar to the volume mesh for the detailed geometry. This is the geometry used for Simulation S-OF.

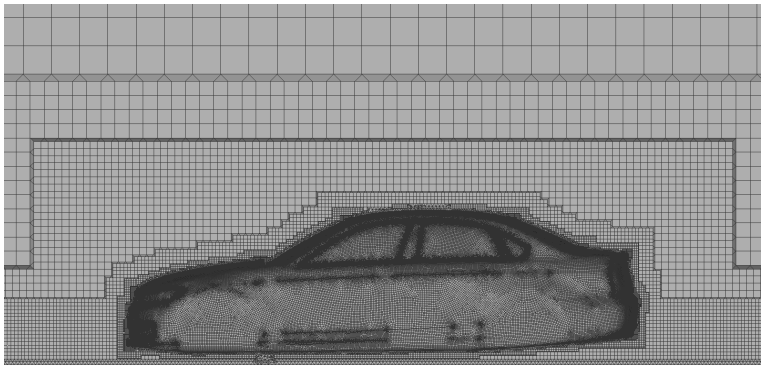


Figure 5.1.2: *The hexahedral volume mesh with prism layers generated in ANSA for the simplified car model. The figure is zoomed in around the car.*

5.2 Solver settings

For the simulation in OpenFOAM the solver interFoam is used. interFoam applies the VOF method and is used for incompressible, immiscible and isothermal two-phase flows. To couple the momentum and pressure solution this solver uses an algorithm called PIMPLE.

5.2.1 PIMPLE

The PIMPLE algorithm is a combination of the steady SIMPLE (semi-implicit method for pressure-linked equations) and the unsteady PISO (pressure-implicit with splitting of operators) algorithms [30]. These algorithms are used to calculate the pressure and velocity fields that fulfill the continuity equation, and other conditions specified for the case. PIMPLE is used for transient simulations and can handle larger Courant numbers than PISO. Inside the algorithm at each time step, the solver first calculates the new phase fraction field α. Thereafter it calculates the new velocity and pressure fields, $\boldsymbol{u}$ and p, using the momentum equation for the velocity and the Poisson equation for the pressure. Before moving on to the next time step a specified number of outer correction loops are used to ensure that the solution has converged for the current time step. One outer correction loop is used in the current simulation.

5.2.2 Transient solver and convergence

A fixed time step size of 1×10^{-4} s is used. In the smallest cells this can give very high Courant numbers (< 10). To obtain a stable and accurate solution with the interFoam solver it is desired to have a maximum Courant number of about 0.2 [31]. To enforce this it is possible to use a Courant number-based adjustable time step. However, as high Courant numbers are only observed in the smallest cells and the Courant number in a vast majority of the cells is much smaller than the limit, a fixed time step size of 1×10^{-4} s is considered acceptable.

In OpenFOAM the solvers and convergence criteria used for obtaining the solution for each field variable within each time step can be specified. Default solvers are used in this project and alternatives are not investigated. The convergence criterion is specified as an absolute or relative tolerance for the residual. Using a relative tolerance means that the residual must decrease a certain amount relative to its initial value, instead of decreasing to below an absolute value. Using a relative tolerance that leads to a higher value for the residual than the fixed value can decrease the simulation time but also lead to instability. For velocity and pressure, relative tolerance is used for all the solver iterations except for the final iteration during each time step, where an absolute tolerance is enforced. For the phase fraction field only absolute tolerance is used. More specific settings can be found in the fvSolution file in Appendix B.

5.2.3 Turbulence model

LES is used for modeling the turbulence. The Smagorinsky model is used for modeling the SGS stress, together with van Driest damping. The default settings for these models in OpenFOAM are used and can be seen in more detail in the turbulenceProperties file in Appendix B.

5.2.4 Discretization schemes

The discretization scheme used for each term in the transport equations can be specified in OpenFOAM. It is desired to use schemes that result in a stable and accurate solution. Further investigation of the numerical schemes is outside the scope of the current project and default schemes for the interFoam solver are used for all terms. Temporarily, alternative discretization schemes are required for the divergence terms due to stability issues. For accuracy second order spatial schemes are required for the divergence term, however these are less stable. First order spatial schemes are significantly more stable, but give less accurate results. A stable solution is obtained by initially using only first order spatial schemes and after some time steps replacing each one at a time for a second order scheme. All numerical schemes used can be found in the fvSchemes file in Appendix B.

5.3 Boundary conditions

The boundary conditions are prescribed for the surfaces of the bounding box and the car. In OpenFOAM the boundary conditions and initial conditions are provided in files for

each field variable that is solved for, in this case the phase fraction α, velocity $\boldsymbol{u}$, dynamic pressure p and turbulent viscosity ν_t. All boundary conditions can be seen in Table 5.3.1. Below, the boundary conditions are further explained.

Table 5.3.1: Boundary conditions used in OpenFOAM for each field variable, presented with type and value.

Boundary	α	$\boldsymbol{u}$	p	ν_t
inlet water	fixed value 1	fixed value 2 m/s	fixed flux press. 0 Pa	calculated
inlet air	fixed value 0	fixed value 2 m/s	fixed flux press. 0 Pa	calculated
outlet	inletOutlet	press. InletOulet vel.	total press. 0 Pa	calculated
top	inletOutlet	press. InletOulet vel.	total press 0 Pa	calculated
road	zero gradient	no slip	fixed flux press. 0 Pa	nut-k wall func.
car	zero gradient	no slip	fixed flux press. 0 Pa	nut-k wall func.
sides	symmetry	symmetry	symmetry	symmetry

The inlet boundary is divided into one inlet for the water and one for the air. The height of the water inlet boundary corresponds to the desired water level. The water level is set at the creation of the bounding box in ANSA to 0.3 m. The phase fraction is set to 1 at the water inlet and 0 at the air inlet, since water is the fluid tracked by α. The velocity is set to a fixed value of 2 m/s directed towards the car for both inlets, to simulate a car moving forward with this velocity inside the domain. For the inlet boundaries, the pressure boundary condition "fixed flux pressure" is used with the value 0. This corresponds to a zero gradient condition adapted for the presence of gravity and surface tension [32]. For the turbulent viscosity the "calculated" boundary condition is used. This means that the field value of the variable is used instead of a prescribed boundary condition [33].

Both the outlet and the top of the domain are set as open to the atmosphere with fixed atmospheric pressure. The combination of the "inletOutlet" condition for the phase fraction, the "pressure InletOutlet velocity" condition for velocity and the "total pressure" condition for pressure is used when the backflow into the domain is unknown. It uses a zero gradient condition for flow out of the domain while the inflow velocity is obtained based on the flux [32].

All surfaces of the car as well as the road are set as solid walls and are given a no slip

velocity condition, meaning that the velocity is zero at the walls. The "nut-k wall functions" condition used for the turbulent viscosity gives a value of the turbulent viscosity based on the turbulent kinetic energy k, using wall functions [33]. Wall functions are described in Section 3.4.3. The remaining left and right sides of the bounding box are given a symmetry condition.

5.4 Initial conditions

To reduce computational time and increase the stability of the solution, initial conditions are used to prescribe the field variables in the domain at the beginning of the first time step. A velocity of 2 m/s and a water level of 0.3 m is prescribed in the entire domain. Atmospheric pressure and no present turbulent viscosity is used as initial conditions for the remaining variables. The complete boundary and initial condition files are provided for each field variable in Appendix B.

6

STAR-CCM+ Methodology

This chapter presents the methodology used for conducting wading simulations with STAR-CCM+. The methodology consists of volume meshing, setup of the simulation and selecting boundary and initial conditions, all done in STAR-CCM+. Two simulation cases are prepared, one case using the simplified geometry and one case using the detailed BEV geometry. The cases also differ in the initial conditions applied to the domain. Apart from that, the settings and physics models used for the cases are identical. The simulation case using the simplified geometry is referred to as simulation S-STAR (Simplified geometry STAR-CCM+ simulation) and the simulation case using the detailed geometry is referred to as simulation D-STAR (Detailed geometry STAR-CCM+ simulation).

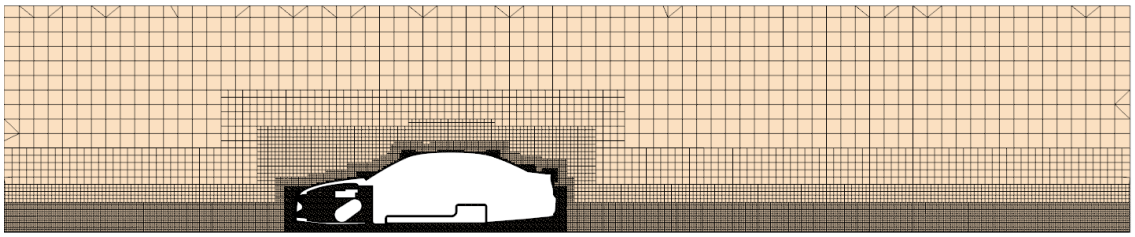
The purpose of simulation S-STAR is to obtain results using the same geometry and initial conditions as for the OpenFOAM case and thereafter compare these results. The purpose of simulation D-STAR is to study wading with EVs in more detail.

6.1 Volume meshing

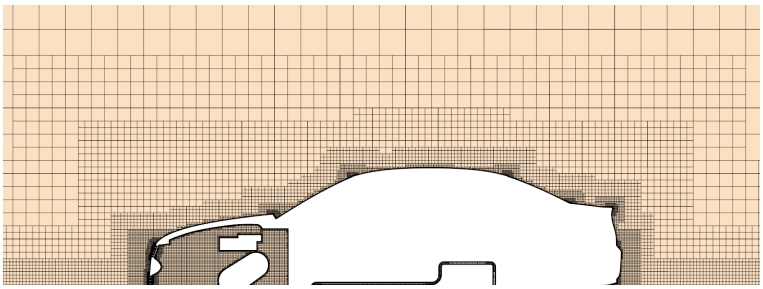
It is possible to create both a surface mesh and volume mesh of good quality directly in STAR-CCM+ using the graphical user interface (GUI). This is done with an automated procedure where key parameters are defined by the user. This way, a mesh that is approved by the CFD solver in STAR-CCM+ is obtained without compatibility problems. In order to detect the surface of the car and bounding box the initial surface mesh created in ANSA is imported to STAR-CCM+. The PID regions assigned in ANSA are recognized by STAR-CCM+. The target cell size for the surface cells are specified with a base size and a relative size for each PID. The base size is set to 16 mm with the smallest cells on the car taking 12.5 % of this value and the largest cells on the bounding box taking 1600 % of this value. The volumetric growth rate which controls the cell size away from the surface is also specified. For the prism cell layers on the car and the road the number of layers and total layer thickness is specified. 5 layers with a total thickness relative to the base size are used. As for the volume meshing in ANSA, the layer settings are based on experience rather than the y^+ condition. The layer settings are instead evaluated after conducting the simulations. For the volume mesh, the maximum cell size is set for the entire domain as well as the mesh refinement boxes. The same mesh refinement boxes and sizes as for the mesh generated in ANSA are used. An extra mesh refinement box with a maximum cell size of 16 mm is added around the lower part of the car to better capture the water surface here.

The volume meshing is done using a so called trimmed hexahedral mesher, which yields

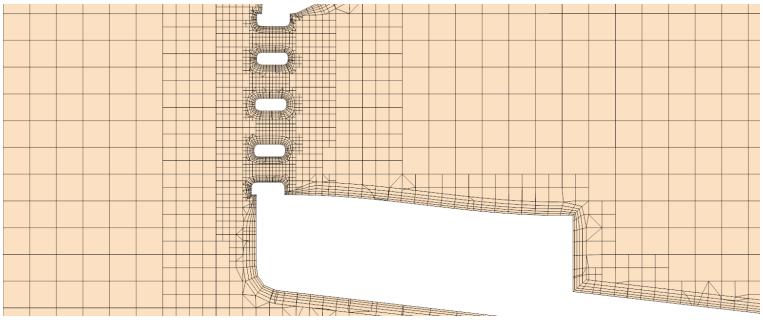
a hexahedral mesh with hanging nodes where cell refinement takes place. The volume mesh cells close to the surface mesh are trimmed to follow the shape of the surface. When executing the mesher in STAR-CCM+ a new surface mesh is generated, followed by the prism cell layers and volume mesh generation. The finished mesh requires no quality check. The final volume mesh generated for the detailed car geometry contains about 16 million cells and can be seen in Figure 6.1.1.



(a) *Entire volume mesh*



(b) *Zoom in of mesh around the car*



(c) *Zoom in of mesh around the lower grille*

Figure 6.1.1: *The trimmed hexahedral volume mesh with prism layers generated in STAR-CCM+ for the detailed geometry, at the middle cross section plane in y-direction.*

6.1.1 Simplified geometry volume mesh

In the same way as a volume mesh is generated for the detailed, a volume mesh is generated for the simplified car geometry with the STAR-CCM+ mesher. The settings used for the meshing are not altered and the generated volume mesh contains about 7.8 million cells. The volume mesh for the simplified geometry can be seen in Figure 6.1.2

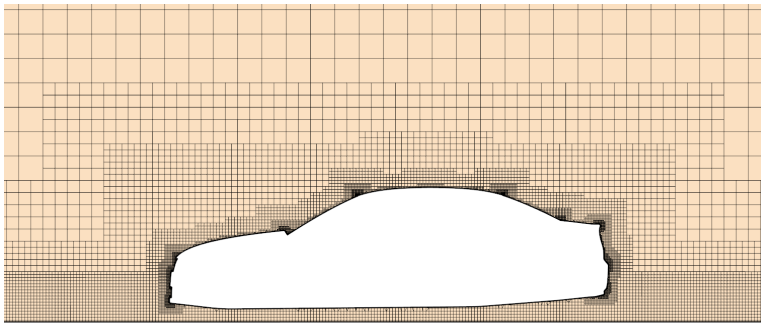


Figure 6.1.2: *The trimmed hexahedral volume mesh with prism layers generated in STAR-CCM+ for the simplified car model. The figure is zoomed in around the car.*

6.2 Simulation setup

After generating the volume mesh, the simulation is set up in STAR-CCM+ by continuing with using the GUI. First, all required physics models are selected, together with model specific settings. All selected physics models can be seen in Table 6.2.1. Below, the models are further explained and additional settings for the models are given.

Table 6.2.1: Physics models selected for the simulation in STAR-CCM+.

Three dimensional
Implicit unsteady
Eularian multiphase
Multiphase interaction
VOF
Multiphase equation of state
Segragated flow
Gradients
Turbulent
LES
Smagorinsky subgrid scale
Exact wall distance
All y+ wall treatment
Gravity
Segregated fluid isothermal

The flow is modelled in three dimensions and the time model is set to implicit unsteady,

meaning that an implicit solver is used for updating the solution at each time step. The Gradients model is selected automatically and is used for calculating the gradients in the transport equations. Default settings are used here. Segragated flow is selected, meaning that each of the the momentum equations and the continuity equations are solved separately. Under this model, the bounded-central discretization scheme is chosen for the convective term, based on recommendations for LES [34]. Since heat transfer is not modelled, the Segregated fluid isothermal model is selected which sets a constant temperature in the entire domain. A gravity model is also added to the simulation.

6.2.1 Multiphase modeling

To use the VOF method the Eularian multiphase models is selected. The Eularian multiphase model is used to create the present phases, in this case water and air. The properties of each phase is also set here. The Multiphase interactions model is selected automatically, however no interactions are specified. For the VOF model a sharpening factor exists. It is used to produce a sharp interface between the pases and is here set to 0.2 based on recommendations for free-surface flows where splashing occurs [34]. The Multiphase equation of state is automatically selected, and equation of state is selected for each phase. For water this is set to constant density and for air this is set to ideal gas.

6.2.2 Turbulence model

For turbulence modeling LES with the Smagorinsky subgrid model and van Driest damping is selected. The Exact wall distance model is used to calculate the exact wall distance for the wall treatment. For this, the All y^+ wall treatment model is used, which can treat cases both when the flow in the viscous boundary layer is resolved by the mesh and when it is not. When the boundary layer is resolved, i.e. $y^+ < 1$, the transport equations are solved for the entire region near the wall. When the y^+ value is higher, wall functions are used to model the turbulence, as described in Section 3.4.3.

6.2.3 Transient solver and convergence

When using the implicit unsteady solver in STAR-CCM+ time step size and inner iterations during each time step must be set. A fixed time step of 5×10^{-4} s is used together with 10 inner iterations. During initial simulations this is found to be enough for the residuals to converge within each time step. The time step size is chosen based on initial simulations to get a stable and fast solution converging within each time step, and the Courant number is not taken into account. The Courant number is however monitored during the simulation to see where it is much larger than 1. The simulation is run until the initial water level has reached the end of the domain and the water surface topology is unchanging.

6.3 Boundary conditions

Instead of prescribing a boundary condition for each field variable on each boundary as in OpenFOAM, only one boundary condition is given for each boundary in STAR-CCM+.

The inlet boundary is also here divided into one inlet for water and one for air, respectively. These boundaries are both set to velocity inlets with the velocity 2 m/s directed from the inlet. The volume fraction is set to 1 for the water phase fraction and 0 for the air phase on the water inlet boundary, and opposite on the air inlet boundary. The outlet and top boundaries are both set as pressure outlets with atmospheric pressure and allowed backflow. Both the road and the car surface are set as walls with no slip shear stress specification. The left and right sides of the domain are given the symmetry plane boundary condition.

6.4 Initial conditions

For simulation S-STAR, the same initial conditions as for simulation S-OF are used, i.e. a velocity of 2 m/s and a water level of 0.3 m inside the entire domain. These initial conditions are used to reduce computational time. For simulation D-STAR the domain is initially empty of water and both velocity, pressure and turbulence is set to zero in the entire domain. This way the effect of the air and water entering the domain at the inlet can be studied. It is also possible to observe the water entry in the engine bay area as the water level rises.

7

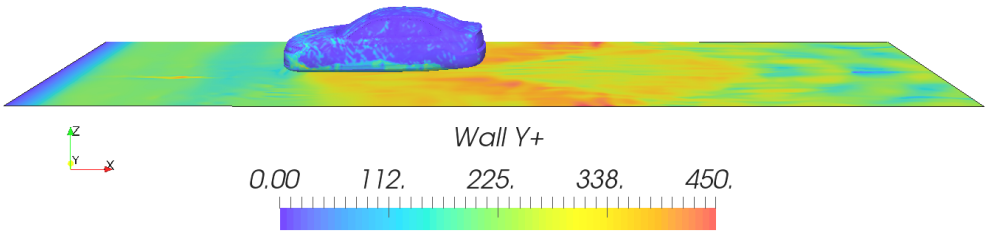
Results and Discussion

In this chapter the results obtained in the current project are presented and discussed. Results are presented for the three simulation cases S-OF, S-STAR and D-STAR. The results from simulation S-OF and simulation S-STAR are compared for validation of the OpenFOAM method. The results from simulation D-STAR are used to discuss the effect of wading on EVs. Post-processing of the OpenFOAM simulation was done with the open source software ParaView. For STAR-CCM+ post-processing tools are included in the software. All results were taken from the final time step. First, results are presented for the dimensionless wall distance and the Courant number to evaluate the mesh and solver settings. Then follows results for the water surface, lift forces and velocity for all simulations. Finally, results for engine bay water ingress and pressure forces are presented for the simulation S-STAR. The chapter ends with an evaluation of the two softwares used in this project.

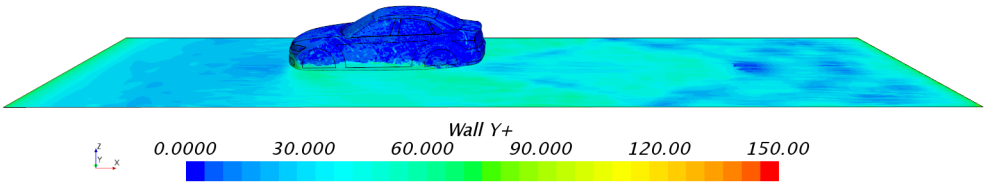
7.1 Dimensionless wall distance

Figure 7.1.1 shows the dimensionless wall distance y^+ on the road and the car surfaces for all simulations. It can be seen that the y^+ value on the car surface is higher in the part of the domain that is below water. For simulation S-OF very high numbers of y^+ are found both on the road and the lower part of the car. The majority of the cells on the road have a y^+ value between 200 and 450, while the cells on the lower part of the car have values between 0 and 230. Since wall functions were used for the OpenFOAM simulation it is desired to have all y^+ values in the log-law region, however cells with y^+ both above and below the range $30 < y^+ < 200$ are present. The cells with values above 200 will probably not affect the results remarkably, as the upper boundary of the log-law region is not distinct. But in order to improve the result it would be suitable to refine the mesh where the y^+ values are too high. The cells on the car where $y^+ < 30$ might give inaccurate results as the log-law does not apply in this region. To solve this problem, the mesh must be coarsened here, or improved wall functions which can handle low y^+ values must be used.

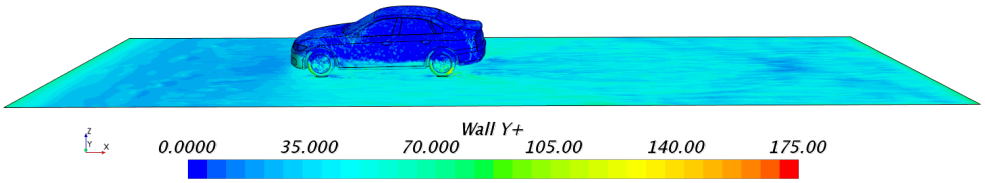
For both simulation S-STAR and simulation D-STAR, the y^+ value on the road and the lower part of the car is well below 200. On most parts, the y^+ value is within the log-law region. This means that wall functions are used for modelling the turbulence, since the "All y^+ wall treatment" model is used. However, for cells where the y^+ value is small enough this model enables the use of the transport equations also for the cells adjacent to the wall, giving accurate results in the entire near-wall region.



(a) *S-OF*



(b) *S-STAR*



(c) *D-STAR*

Figure 7.1.1: *Dimensionless wall distance y^+ on the car and the road for all simulations.*

When comparing the simulations S-OF and S-STAR, S-OF gives overall higher y^+ values. As the volume meshes used for these simulations have similar cell sizes in these regions, the higher value must be due to a higher wall shear stress in the OpenFOAM case.

7.2 Courant number

As described for both the OpenFOAM simulation in Section 5.2.2 and the STAR-CCM+ simulations in Section 6.2.3, the Courant number did not directly influence the choice of time step size. However, as very high Courant numbers can have a negative impact on the accuracy of the solution, the maximum Courant number of the domain was monitored during simulation. Also, at the end of the simulations the cells with the highest Courant numbers were detected. Table 7.2.1 shows the maximum Courant number of the domain for all simulations, as well as the number of cells with Courant numbers above certain values.

For simulation S-OF, the maximum observed Courant number is low and most cells have a Courant number below 0.2, as desired for a stable and accurate solution. The cells with the highest Courant numbers are found at the a-pillars and headlights. Simulation S-STAR gives a higher maximum Courant number and more cells where the $Co \approx 1$ criterion is violated. This is probably due to the larger time step used in the STAR-CCM+ simulations, which is five times larger than the time step size used in OpenFOAM. The cells with high Courant numbers are also here found at the a-pillars and headlights.

Table 7.2.1: Maximum Courant number and number of cells with Courant numbers above certain values, for all simulations.

Simulation	Max Co	$Co > 1$	$Co > 10$	$Co > 50$
S-OF	2.69	2.16×10^4 cells	-	-
S-STAR	32.2	2.96×10^5 cells	615 cells	-
D-STAR	222	4.15×10^5 cells	892 cells	10 cells

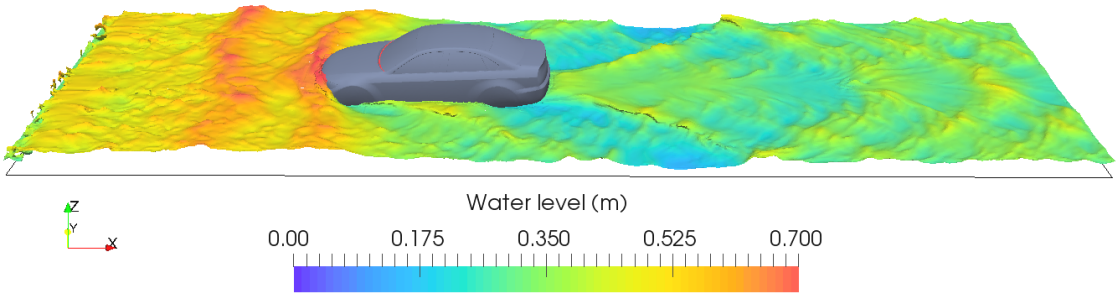
For simulation D-STAR, the maximum observed Courant number is much higher than for the other simulations, and the number of cells with $Co > 1$ is relatively very high. This can be explained by the higher total cell count for the detailed geometry mesh, and that the mesh refinement in the regions around the smaller features of the geometry give a much larger number of small cells. Smaller cells yield higher Courant numbers, according to Equation 3.5.1. When detecting the cells with $Co > 1$, they are found near the smaller features of the geometry, such as the lower grille, wheels and side mirrors. These features are not present in the simple geometry.

In Table 7.2.1 it can also be seen that number of cells with $Co > 10$ and $Co > 50$ for simulation D-STAR are relatively small, meaning that a majority of the cells have a Courant number not much higher than 1. Therefore, this should not have a major impact on the accuracy of the results. A smaller time step would have resulted in lower Courant numbers, however due to available computational resources and the time limitation of the current project, this was not implemented. This holds for all simulations.

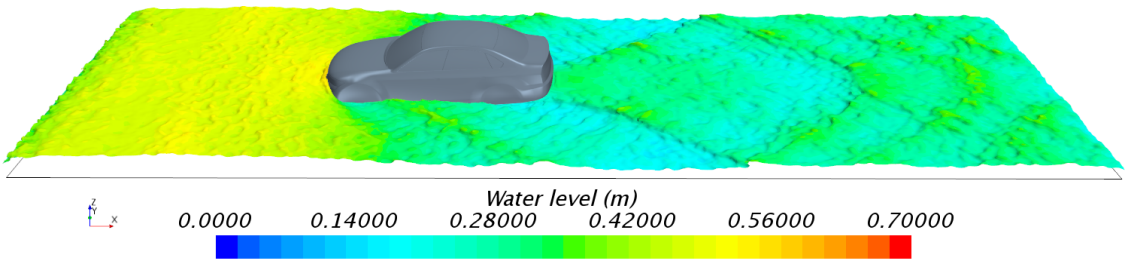
7.3 External water surface

Figure 7.3.1 shows the water surface in the domain visualized as an isosurface, for all simulations. A drawback of the VOF method is that since the surface is not explicitly tracked, no sharp surface exists between the two phases. Instead, the surface will be smeared, as the value of α gradually changes from 1 to 0 from the water domain to the air domain. The isosurface is the extracted surface where the water phase fraction takes the value 0.5. This is considered a reasonable approximation for the surface between the phases. In Figure 7.3.1, also the water level is presented on the isosurface.

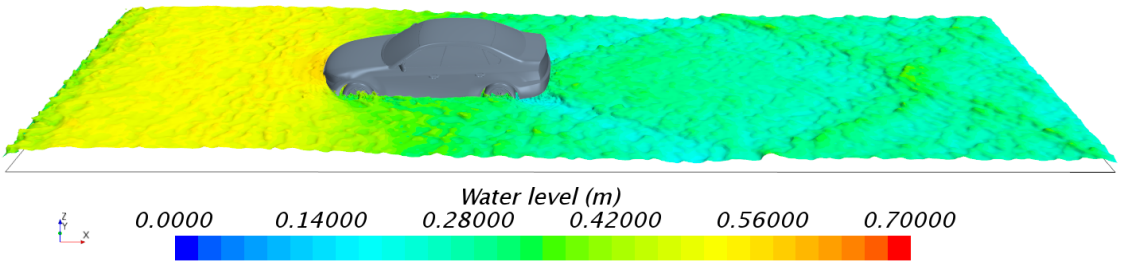
For all simulations it can be seen that the water level is higher in front of the car, caused by the constant inflow of water that meets the blunt car body. The expected bow wave is visible for simulation S-OF, but not as clear for the STAR-CCM+ simulations. However there is a distinct decrease of the water level just after the car front for all simulations. The lowest water level is found in two diagonal traces behind the car caused by the wake, and for simulation D-STAR it is also affected by the rear wheels. In the wake behind the car, the two streams from each side of the car appears to unite again. At the water inlet, the surface seems to bend upwards with a fast increase of the water level. This is due to the boundary condition for the phase fraction of water on the air inlet, which is set to zero. This forces the water surface to bend here, due to the accumulation of water in the domain in front of the car.



(a) *S-OF*



(b) *S-STAR*



(c) *D-STAR*

Figure 7.3.1: *Water surface visualized with an isosurface with water phase fraction value 0.5 for all simulations. On the surface, the water level contour is visible.*

While the water level topology is similar for all simulations, more violent level differences are found for simulation S-OF. The reason behind this is unknown and is probably due to differences in the solvers. The symmetry boundary condition applied at the left and right sides of the domain create an interference pattern behind the car. This is more similar for S-OF and S-STAR, due to similar geometry and initial conditions. An extra wave is also seen at the side of the car for these two simulations, due to the difference in the geometry. Since no water can pass through the lower grille for the simplified geometry, all water is forced under or to the sides of the car.

For simulation S-OF a small amount of water is visible just below the windshield. This was observed also for the STAR-CCM+ simulations, even if it is not visible in Figure 7.3.1. This is probably due to some numerical error, since no splashing occurs and no water flow going here is visible.

For simulation D-STAR, the domain was initially empty of water. The evolution of the water surface as water entered the domain can be seen in Appendix A.

A major aspect affecting all results above is the chosen reference frame of moving water and air instead of moving car. This affects both the shape of the water surface around the car, as well as the interface between air and water away from the car. A more realistic result can be obtained by using a moving car geometry entering a fixed water domain, together with using rotating wheels.

7.4 Lift and drag forces

Reference simulations were performed to investigate the lift/drag ratio with only airflow in the domain. The obtained lift force on the car geometry was then around 70 N. It also showed that the lift/drag ratio can vary from 50-100 for the low flow velocity used in the current project depending on the geometry. This result differs severely from aerodynamic studies where higher velocities are used, see Section 3.1.1. Therefore, the reference lift/drag ratio does not provide useful data for this case and only the lift and drag forces themselves can be compared. Table 7.4.1 shows the lift and drag force obtained for all simulations.

Table 7.4.1: Lift and drag force on the car geometry for all simulations.

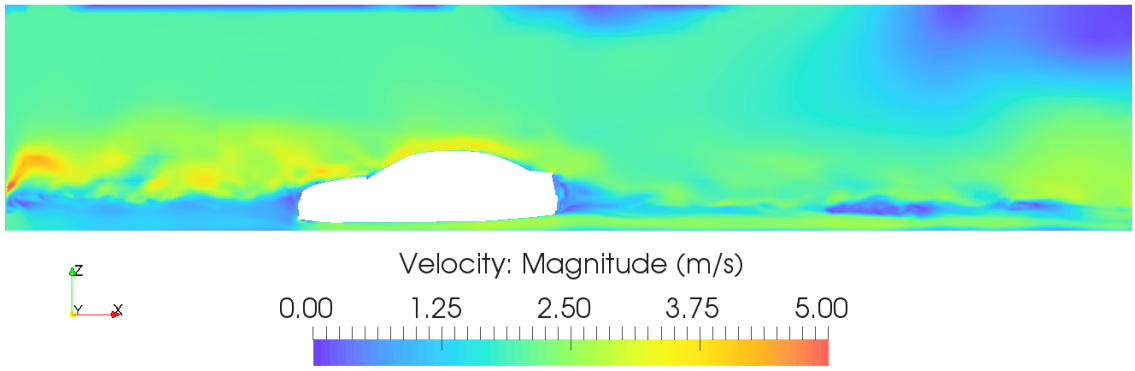
Simulation	Lift force (N)	Drag force (N)
S-OF	10930	1372
S-STAR	7667	777.3
D-STAR	6809	1349

Large variations of the forces are observed. The lift forces obtained for the STAR-CCM+ simulations are about 100 times larger than for the reference case with only air, showing that the lift force from the water is significant. The lift force reduces the gravitational effect on the car and makes it more sensitive to side forces. This could be problematic in the presence of currents, for example when crossing rivers or floodings. The higher drag force obtained for simulation D-STAR is due to the added features on this geometry such as wheels and side mirrors. Simulation S-OF shows both higher lift and drag force than simulation S-STAR, implying solver differences.

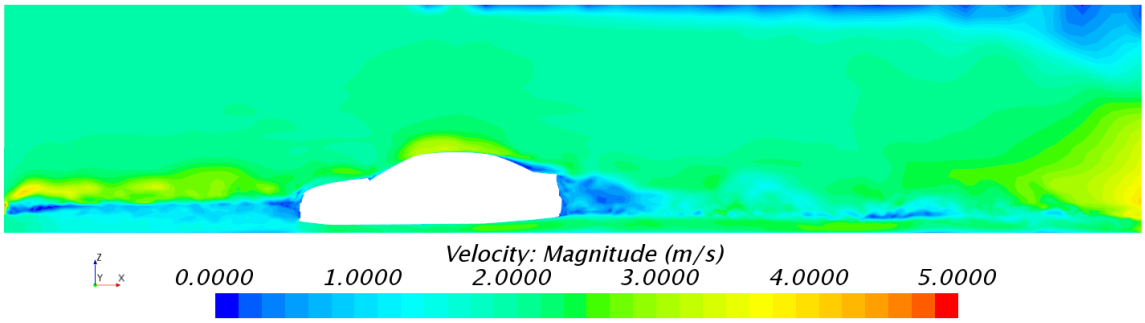
7.5 Velocity

Figure 7.5.1 shows the velocity magnitude in the entire domain at the middle cross section plane for all simulations. The velocity fields show a similar behaviour in several areas. The velocity is lower inside the water domain in front of the car, due to accumulation of the inflowing water in front of the blunt car body as discussed in Section 7.3. It can also be seen that the velocity is lowest for the water at and just below the surface. This is probably due to the accumulation of water where overshoot is forced upwards. Regions of low velocity are found near the surface also downstream of the car, and these are larger for the simulations with the simplified geometry. This is due to the different surface topology observed downstream of the car for the simulations with the simplified geometry,

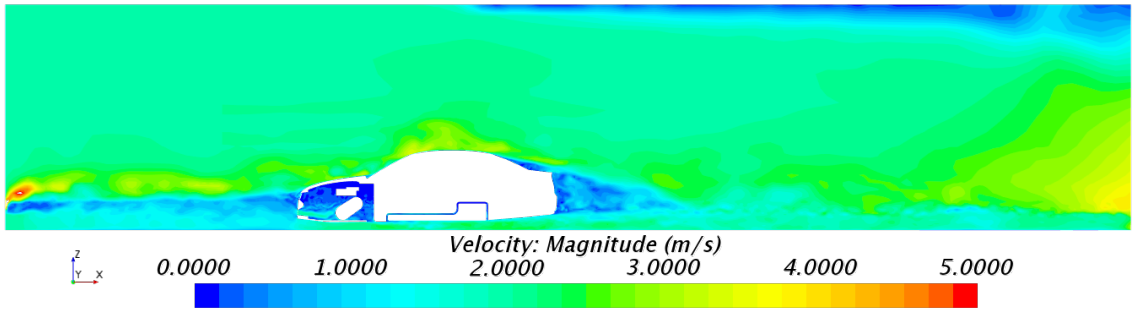
compared to the surface obtained with the detailed geometry. The water is accelerated after passing the car front as the flow is forced under the car. The water velocity field is further discussed with additional velocity planes later in this section.



(a) *S-OF*



(b) *S-STAR*



(c) *D-STAR*

Figure 7.5.1: *Velocity magnitude at the middle cross section plane in the y-direction for all simulations.*

A small region of high air velocity is present just above the water surface at the inlet. This is due to the bending of the water surface caused by the boundary condition at the air inlet, as discussed in Section 7.3. The inlet air is forced to pass the bent water surface and then reaches a high velocity immediately after the inlet. In the air domain a wake is formed behind the car.

When comparing the velocity field in Figure 7.5.1 for simulation S-OF and simulation

S-STAR, S-OF shows larger regions of deviating velocities both in the air and water domain whereas S-STAR shows a more uniform velocity field. Apart from that, the observed patterns are similar and the velocities in the different regions described above are of the same magnitudes for the two simulations. For simulation D-STAR, a slight increase of velocity is observed just after the lower grille as the water enters the engine bay area. A non-moving air domain is present in the upper half of the engine bay.

To further investigate the water velocity, the velocity magnitude was studied at six cross sections in the x-direction. The location of the cross section planes can be seen in Figure 7.5.2. The Figures 7.5.3, 7.5.4 and 7.5.5 show the velocity magnitude of the water phase at the six cross sections for all simulations.

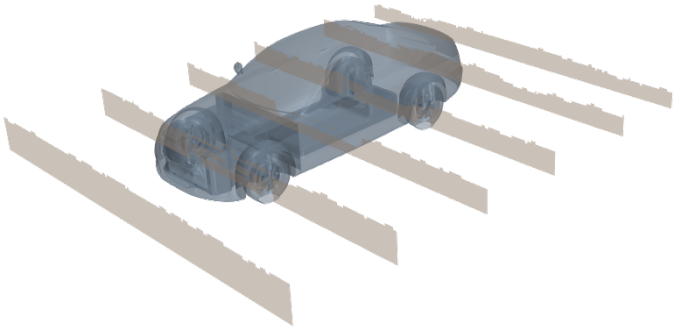
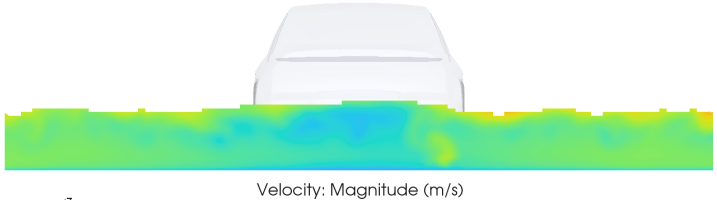


Figure 7.5.2: *Locations of the cross sections used in Figures 7.5.3, 7.5.4 and 7.5.5. The x-coordinates of the planes from front to rear are -1.5 m, 0.0 m, 1.5 m, 3.0 m, 4.3 m and 5.5 m.*

In Figures 7.5.3, 7.5.4 and 7.5.5 it is clear that the velocity of the water is lower in front of the car and that the water is accelerated when passing the car body. This acceleration is related to the decreased water depth, as mentioned in Section 7.3. At x=-1.5 m the velocity difference between the upper and lower part of the water domain right in front of the car body is clearly visible. This difference however disappears as the water passes the car and all water is accelerated. The difference in water level variation between simulation S-OF and simulation S-STAR mentioned in Section 7.3 is clearly visible. There are also differences in velocity magnitude between the cases.

For simulation D-STAR, regions of low velocity are found around the car wheels, battery pack and underbody for the locations x=0.0 m, x=1.5 m and x=3.0 m. These detailed features obstruct the water flow in a way that is not observed for the simplified geometry simulations.

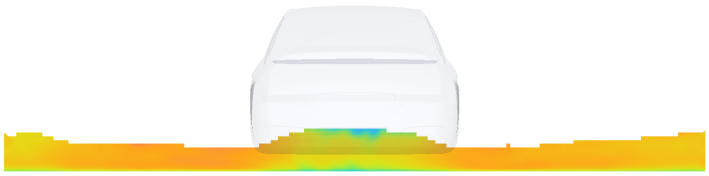


(a) x=-1.5 m

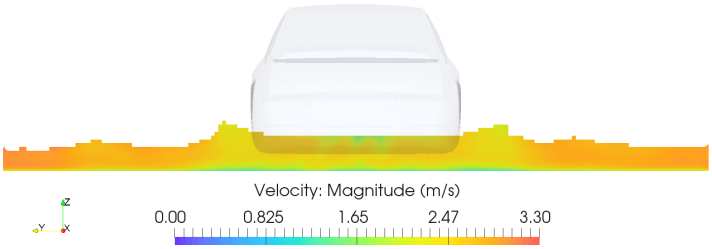
(b) x=0.0 m

(c) x=1.5 m

(d) x=3.0 m



(e) x=4.3 m



(f) x=5.5 m

Figure 7.5.3: *Velocity magnitude of the water phase at cross sections along the x-direction for simulation S-OF, seen from the inlet. The cross section locations can be seen in Figure 7.5.2.*

(a) x=-1.5 m

(b) x=0.0 m

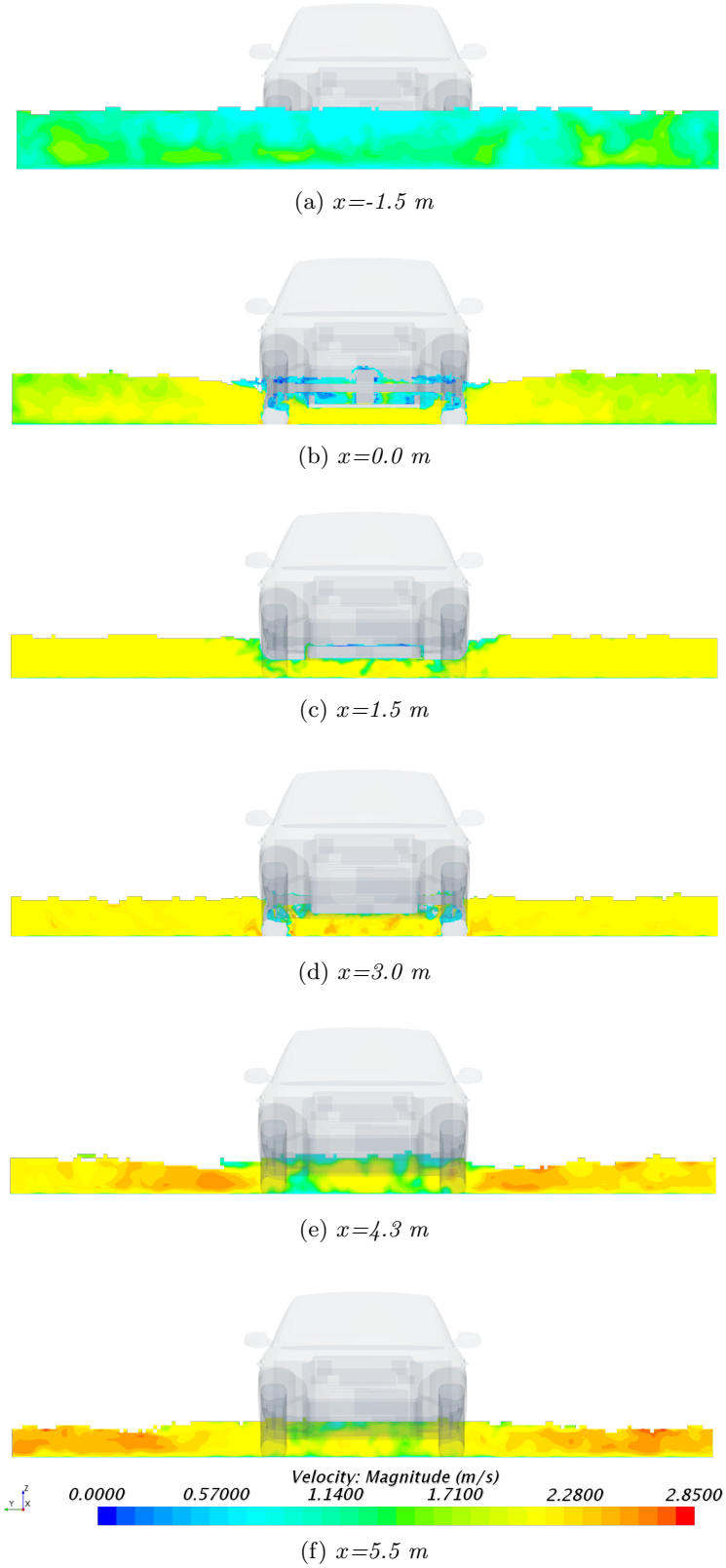
(c) x=1.5 m

(d) x=3.0 m

(e) x=4.3 m

(f) x=5.5 m

Figure 7.5.4: *Velocity magnitude of the water phase at cross sections along the x-direction for simulation S-STAR, seen from the inlet. The cross section locations can be seen in Figure 7.5.2.*



(a) x=-1.5 m

(b) x=0.0 m

(c) x=1.5 m

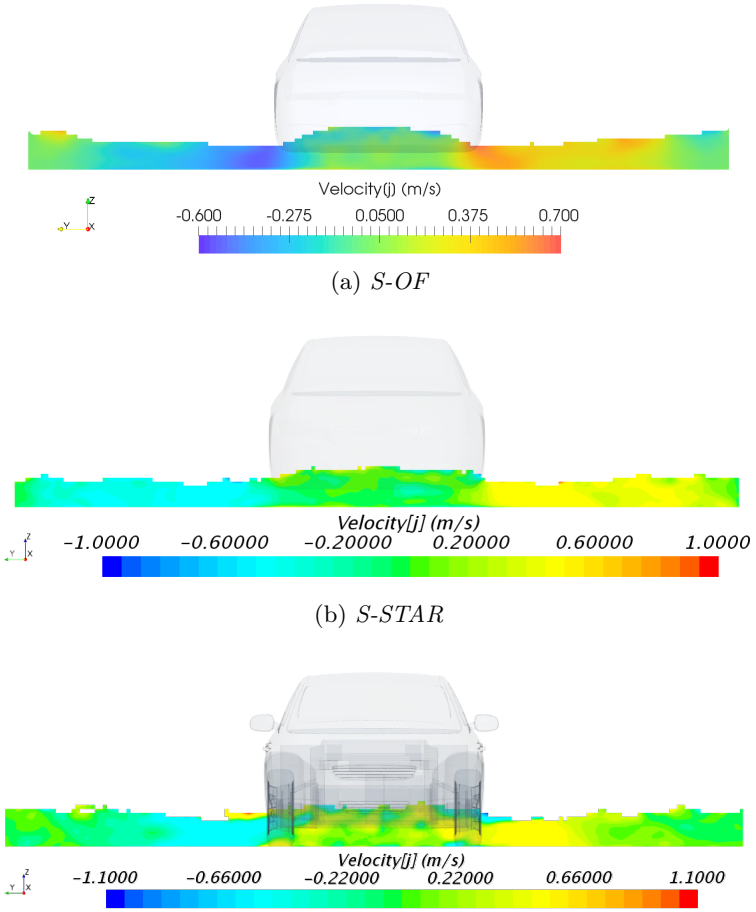
(d) x=3.0 m

(e) x=4.3 m

(f) x=5.5 m

Figure 7.5.5: *Velocity magnitude of the water phase at cross sections along the x-direction for simulation D-STAR, seen from the inlet. The cross section locations can be seen in Figure 7.5.2.*

In the wake behind the car, the regions of different velocity are mixed. This observation is further supported by Figure 7.5.6. This figure shows the y-component of the water velocity at x=4.3 m and it can be seen that the water from the outer region moves towards the middle of the domain.



(a) *S-OF*

(b) *S-STAR*

(c) *D-STAR*

Figure 7.5.6: *Velocity y-component of the water phase at a cross section at x=4.3 m for all simulations.*

7.6 Engine bay water ingress

For simulation D-STAR results for the water surface inside the engine bay area and around the battery pack were obtained. In the following two sections, all presented results are from this simulation.

With the more detailed geometry it was possible to study the water ingress in the engine bay area. Figure 7.6.1 shows the water phase fraction at a cross section plane zoomed in on this region. Here, the gradual change of the phase fraction at the surface obtained with the VOF method is visible. The phase fraction field is similar to but less turbulent than comparable results shown by Makana et al. [10]. However, the velocity and water depth used for producing these results are not known and therefore a complete comparison is

not possible. A significant amount of water enters the engine bay through the lower grille and submerges half of the gearbox and the electric motor. Therefore, it is necessary that these parts and related electrical connections are made watertight. However, long-term exposure of such connections to water, especially water contaminated with salt and mud, can lead to wear and damage [8]. The constant flow of water past the gearbox and the electric motor could give rise to such problems. One option to decrease the water ingress is to modify the design, by adding for example protection panels. In case liquid cooling is used, air intakes can be omitted, preventing all water ingress through the front grille. No water reaches the power converter.

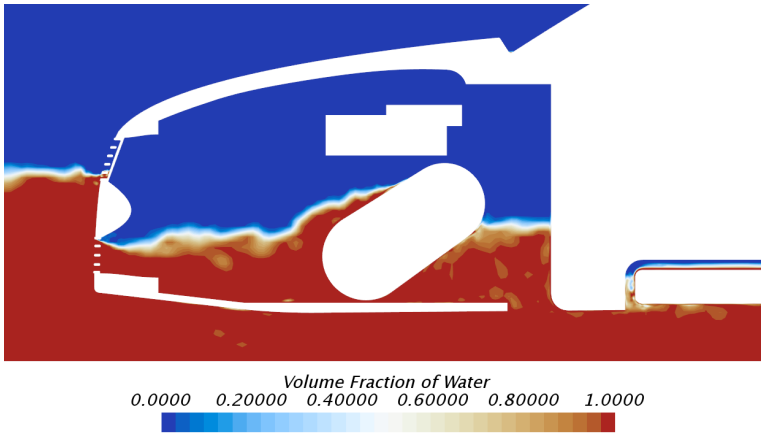


Figure 7.6.1: *Volumetric phase fraction of water in a middle cross section in the y-direction, zoomed in on the engine bay area.*

After reaching the final water level the water flow in the engine bay is relatively steady with an even water surface, and no splashing taking place. Figure 7.6.2 shows the water phase fraction on the firewall, where the water level is rather even. This is probably due to the low velocity (2 m/s) of the car. Zheng et al. [11] showed in their study that low velocity corresponded with less splashing on the firewall but similar amount of water ingress.

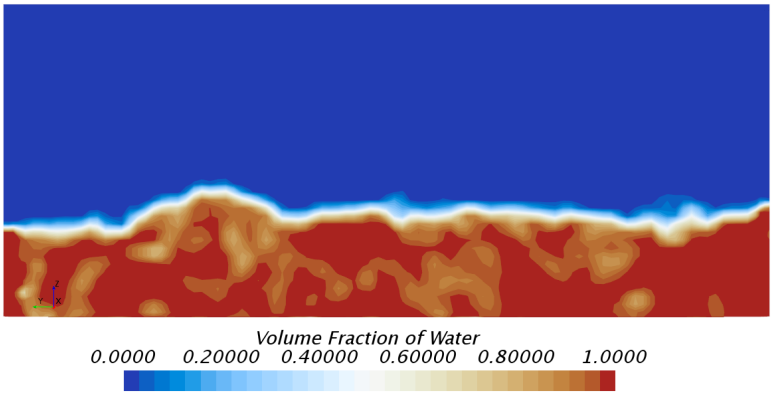


Figure 7.6.2: *Volumetric phase fraction of water on the firewall surface.*

7.7 Pressure forces

The pressure forces exerted by the water can cause damage on sensitive components of the car, such as the battery housing. Figure 7.7.1 shows the pressure on the housing of the battery pack. The pressure is higher on the bottom part. This pressure difference is probably due to that the initial water level of 0.3 m is forced under the battery, which is placed approximately 0.24 m above the road. During the simulation the maximum pressure on the battery pack was monitored. The pressure on the battery surface increased as the water approached and stabilized at approximately 1900 Pa above atmospheric pressure. As this corresponds to a pressure only ca 2 % higher than atmospheric pressure, the pressure difference is practically negligible. Therefore, there is no risk of material failure due to water pressure in a wading situation like this, with low speed and low splashing.

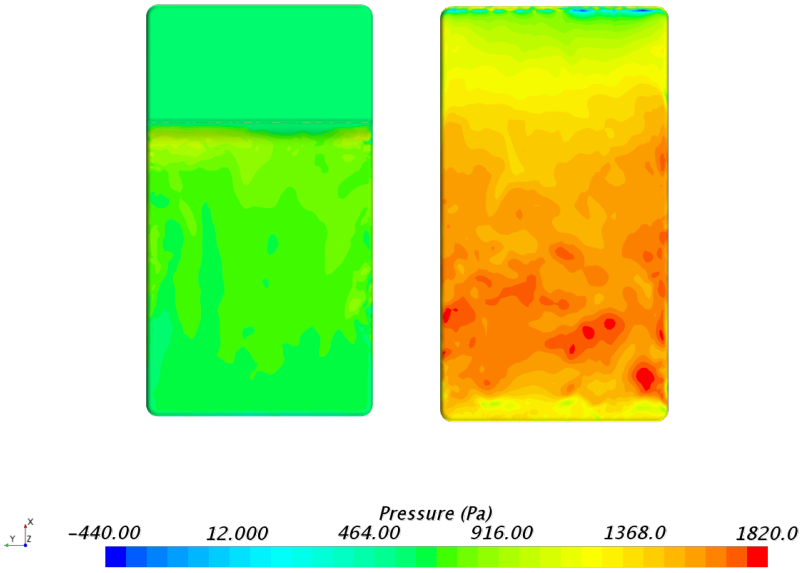


Figure 7.7.1: *Pressure on the battery pack surface, for battery top (left) and bottom (right). The pressure value is presented relative to atmospheric pressure.*

Figure 7.7.2 shows the water surface in the area around the battery pack, with all other parts of the car excluded. Initially, it was thought that the air in the thin gap between the battery pack and the underbody would be trapped when water started to fill up under the car. This could have caused a high pressure as well as prevent water from reaching the top of the battery pack. However, Figure 7.7.2 shows that water indeed manages to flow on top of the battery pack, however not completely covering it. As mentioned in Chapter 2 the battery housing is watertight, but the fact that a large part of the battery pack is submerged under water still poses a risk. The interior components of the battery pack are sensitive to water and long exposure of battery pack connections and valves could lead to wear and damage for other reasons than high pressure.

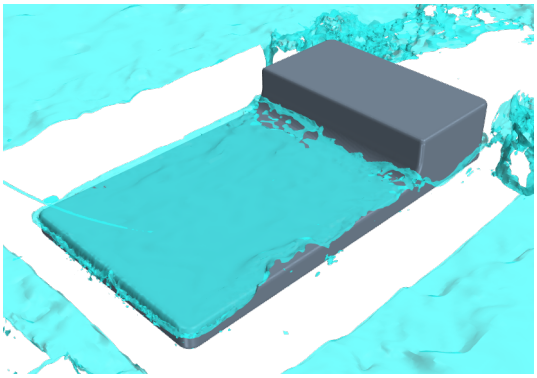


Figure 7.7.2: *Water isosurface zoomed in on the area around the battery pack. Other car parts are hidden.*

Figure 7.7.3 shows the pressure on the gearbox and electric motor. The gearbox experiences a slightly higher pressure that the battery housing, but it is still too low to be considered a risk. This pressure is caused by the water ingress through the lower grille that hits the engine bay components, and is not due to splashing. The absence of splashing reduces the risk of engine bay components being exposed to high local pressures from fast-moving water. Therefore, these components are unlikely to undergo material failure. Using rotating wheels in the simulation could cause splashing with high force, leading to an increased risk for local high pressure on the battery housing, underbody and maybe even engine bay components, depending on the direction of the splashing.

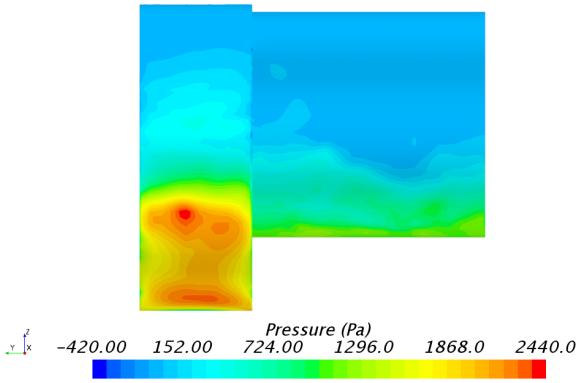


Figure 7.7.3: *Pressure on the gearbox and electric motor surfaces. The pressure value is presented relative to atmospheric pressure.*

7.8 Software evaluation

One of the main objectives of this study was to evaluate the ability of OpenFOAM to solve wading problems. Here, some comments on working with OpenFOAM as well as other softwares involved in the developed method in the current project will be given. OpenFOAM will also be compared to the commercial software STAR-CCM+ based on experiences from the current project.

The main differences between OpenFOAM and STAR-CCM+ are the lack of a GUI and user-support in the former. External open source and commercial GUIs exist for OpenFOAM, but were not used in the current project. Also numerous user guides exist, and tutorials and advice can be found through the vast user community related to OpenFOAM. This can however not compare to the user guide included in STAR-CCM+ which provides tutorials and best practices for a large variety of flows in one place.

To setup a new simulation case in OpenFOAM from the beginning was very time consuming, as there were no sufficient tutorial cases available simulating the same flow situation as in the current project. Therefore, the process was to a large extent based on trial and error until appropriate boundary conditions and solver settings were found. These were used for the simplified geometry mesh and the solver showed convergence. However, when applying the same solver settings and boundary conditions to the detailed geometry mesh the solver diverged. It was not possible to find optimal solver settings for this case within the project time and with the available computational resources.

The meshing procedure in ANSA preceding the simulation setup in OpenFOAM was also time consuming, as OpenFOAM was found to be sensitive to the mesh. Even low numbers of low-quality cells led to solver divergence. Much work was therefore required in order to obtain a mesh of good quality. Numerous geometry modifications were required in conjunction with this.

The entire workflow in STAR-CCM+ including meshing and simulation setup took only a fraction of the time required for the same with the OpenFOAM method. However, the simulation setup in STAR-CCM+ was not investigated as thoroughly as the OpenFOAM simulation setup. It aimed to be equivalent to the OpenFOAM setup but was based on information from the user guide. As settings resulting in a stable solution were found rapidly these were not further reviewed. This includes settings that showed convergence also for the detailed geometry mesh, with a larger time step than was used with OpenFOAM. A further benefit of STAR-CCM+ is the native automated mesher, which produced an error-free mesh within hours. However, geometry alterations removing problematic areas had then already been done during the meshing procedure in ANSA.

8

Summary and Conclusions

In this project, a method for studying vehicle wading with the open source CFD software OpenFOAM has been developed. The method encompasses geometry preparation, surface and volume meshing, selection of flow models and solver setup, and choice of boundary and initial conditions. The numerical method is based on the VOF method. The method has been applied to a simplified geometry of a car and results have been obtained for the water surface, velocity field and forces acting on the car. A comparable simulation has been conducted with STAR-CCM+ in order to evaluate the performance of the developed methodology.

Furthermore, wading with EVs has been investigated by performing simulations in STAR-CCM+, with a more detailed car geometry resembling a BEV. The simulation setup is similar to the developed OpenFOAM method. Results have been obtained for the water surface, velocity field, forces acting on the car, water ingress in engine bay and pressure forces on sensitive components.

8.1 Effect of wading on EVs

It was found that in a low speed wading situation like the one studied in the current project, the water flow is relatively steady and therefore predictable. There is no splashing taking place, and therefore the risk of material failure due to high local pressure on components is low. The pressure forces caused by the slow-moving water are too low to be considered hazardous. The only risk identified is that long-term exposure of sensitive connections to heavy water flow can lead to wear and damage. As several engine bay components are submerged in water, watertightness is a must. Alternatively, protection panels or an entirely watertight engine bay can be used.

High lift forces were observed for both the BEV geometry and the simplified geometry. A high lift force makes the car more sensitive to side forces and constitutes a risk during deep water wading. This is naturally true also for other car types than EVs.

8.2 Evaluation of OpenFOAM method

The developed OpenFOAM method was applicable to simpler geometries, but showed divergence when applied to more complex geometries. Evaluation of the y^+ value showed that mesh refinement is required in some areas, while some areas had a too fine mesh.

Evaluation of the Courant number showed that a smaller time step is required for increased stability and accuracy. Even for simpler geometries the solver was sensitive to the choice of divergence schemes. This implies that the chosen solver settings are suboptimal.

The results obtained with the developed OpenFOAM method were similar to the results obtained with STAR-CCM+ but not entirely equivalent. For the water surface, similar topology but differing water levels were observed. The velocity fields were more similar but still differed in magnitude in some regions. Larger deviations were found for both drag and lift force, however the cause of the deviations could not be detected. The comparison alone is not sufficient to evaluate the accuracy of the OpenFOAM method, as the accuracy of the STAR-CCM+ method is unknown. To evaluate the performance of the OpenFOAM method and investigate the solver differences, benchmark testing with multiple cases and comparison with experimental results is required.

Setting up an entirely new simulation case in OpenFOAM was time consuming compared to using a commercial software, and therefore the use of the software is only profitable once a method for a specific type of problem is established. However, for the automotive industry where the development process is long and a limited type of flow cases are studied it can be profitable to spend time developing such a method. Thereby OpenFOAM has the potential to reduce the cost of numerical simulations in the automotive industry.

8.3 Future work

Initially, it was desired to investigate wading with EVs using the OpenFOAM method. Due to the instability issues the method has not been applied to the more detailed BEV geometry. Future work should focus on developing the OpenFOAM method, to find optimal solver settings and eliminate the convergence problems when studying complex geometries. This includes optimizing the discretization schemes and solvers used.

Further improvement of the method could be obtained by using a different solver than interFoam. The recently developed interIsoFoam gives less surface smearing than interFoam and could give a better surface representation. Investigating contact angle boundary conditions for the walls is also of interest.

Adding more details to the geometry, such as rotating wheels and more components inside the engine bay, would give a more realistic flow situation. Finally, conducting a more extensive wading study using several water depths and velocities would give a better understanding of the wading phenomena.

References

[1] *Summary of the Paris agreement*. UNFCCC. 2017. URL: http://bigpicture.unfccc.int/#content-the-paris-agreemen Accessed: Sept. 8, 2017.

[2] *CO2 emissions from fuel combustion, Highlights 2016*. International Energy Agency. 2016. URL: http://www.iea.org/publications/freepublications/publication/CO2EmissionsfromFuelCombustion_Highlights_2016.pdf Accessed: Sept. 8, 2017.

[3] *Global EV outlook 2017*. International Energy Agency. 2017. URL: https://www.iea.org/publications/freepublications/publication/GlobalEVOutlook2017.pdf Accessed: Sept. 8, 2017.

[4] A. Ajanovic. The future of electric vehicles: prospects and impediments. *Wiley Interdisciplinary Reviews: Energy and Environment* 2015, 4(6), p. 521–36.

[5] C. Linse and R. Kuhn. Design of high-voltage battery packs for electric vehicles. In: B. Scrosati, J. Garche, and W. Tillmetz, (eds.). *Advances in battery technologies for electric vehicles*. Amsterdam, Netherlands: Woodhead Publishing, 2015, p. 245–63.

[6] E. A. Grunditz and T. Thiringer. Performance analysis of current BEVs based on a comprehensive review of specifications. *IEEE Transactions on Transportation Electrification* 2016, 2(3), p. 270–89.

[7] *Official Flickr profile for Land Rover in the Middle East and North Africa*. Land Rover MENA. 2015. URL: https://www.flickr.com/photos/landrovermena/19856150571 Accessed: Feb. 23, 2018.

[8] P. Khapane, V. Chavan, and U. Ganeshwade. Water ingress analysis and splash protection evaluation for vehicle wading using non-classical CFD simulation. *SAE International Journal of Passenger Cars - Mechanical Systems* 2017, 10(1), p. 183–94.

[9] P. Khapane and U. Ganeshwade. Wading simulation - challenges and solutions. *SAE Technical Paper 2014-01-0936* 2014.

[10] M. Makana, G. Kumar, and F. Regin. Passenger car water wading evaluation using CFD simulation. *SAE Technical Paper 2016-28-0072* 2016.

[11] X. Zheng, X. Qiao, and F. Kong. Vehicle wading simulation with STAR-CCM+. In: SAE-China and FISITA, (eds.). *Proceedings of the FISITA 2012 World Automotive Congress*. New York, USA: Springer, 2013. Chap. 3, p. 157–66.

[12] C. C. Chan. The state of the art of electric and hybrid vehicles. *Proceedings of the IEEE* 2002, 90(2), p. 247–75.

[13] E. A. Grunditz. *Design and assessment of battery electric vehicle powertrain, with respect to performance, energy consumption and electric motor thermal capability.* PhD thesis. Chalmers University of Technology, 2016.

[14] J. H. Ferziger and M. Perić. *Computational methods for fluid dynamics.* 5th ed. New York, USA: Springer, 2002.

[15] C. Strangfeld, D. Wieser, H.-J. Schmidt, R. Woszidlo, C. Nayeri, and C. Paschereit. Experimental study of baseline flow characteristics for the realistic car model DrivAer. *SAE Technical Paper 2013-01-1251* 2013.

[16] M. Ishii and T. Hibiki. *Thermo-fluid dynamics of two-phase flow.* 9th ed. New York, USA: Springer Science+Business Media, Inc., 2006.

[17] C. W. Hirt and B. D. Nichols. Volume of fluid (VOF) method for the dynamics of free boundaries. *Journal of Computational Physics* 1981, 39, p. 201–25.

[18] V. R. Gopala and B. G. van Wachem. Volume of fluid methods for immiscible-fluid and free-surface flows. *Chemical Engineering Journal* 2008, 141(1), p. 204–21.

[19] E. D. Christensen. Large eddy simulation of spilling and plunging breakers. *Coastal Engineering* 2006, 53(5), p. 463–85.

[20] R. J. McSherry, K. V. Chua, and T. Stoesser. Large eddy simulation of free-surface flows. *Journal of Gydrodynamics* 2017, 29(1), p. 1–12.

[21] S. B. Pope. *Turbulent flows.* 1st ed. Cambridge, United Kingdom: Cambridge University Press, 2000. ISBN: 0-521-59125-2.

[22] P. Sagaut. *Large eddy simulation for incompressible flows.* 3rd ed. Berlin, Germany: Springer, 1998.

[23] J. Smagorinsky. General circulation experiments with the primitive equations. I: The basic experiment. *Monthly Weather Review* 1963, 91(3), p. 99–165.

[24] E. Lévêque, F. Toschi, L. Shao, and J.-P. Bertoglio. Shear-improved smagorinsky model for large-eddy simulation of wall-bounded turbulent flows. *Journal of Fluid Mechanics* 2007, (570), p. 491–502.

[25] A. I. Heft, T. Indinger, and N. A. Adams. Introduction of a new realistic generic car model for aerodynamic investigations. *SAE Technical Paper 2012-01-0168* 2012.

[26] E. Guilmineau. Numerical simulations of flow around a realistic generic car model. *SAE International Journal of Passenger Cars - Mechanical Systems* 2014, 7(2), p. 646–53.

[27] G. Shinde, A. Joshi, and K. Nikam. Numerical investigations of the DrivAer car model using opensource CFD solver OpenFOAM. *Conference: OSCIC 13, At Hamburg, Germany, 2013* 2013.

[28] J. Nordin. *CFD study of optimal under-hood flow for thermal management of electric vehicles.* MA thesis. Chalmers University of Technology, 2017.

[29] G. C. J. Morgan. *Application of the interFoam VOF code to coastal wave/structure interaction.* PhD thesis. University of Bath, Sept. 2013.

[30] T. Holzmann. *Mathematics, numerics, derivations and OpenFOAM(R).* Loeben, Germany: Holzmann CFD, URL: `https://holzmann-cfd.de` (visited on Nov. 29, 2017).

[31] S. M. Damián and N. M. Nigro. An extended mixture model for the simultaneous treatment of small-scale and large-scale interfaces. *International Journal for Numerical Methods in Fluids* 2014, 75(8), p. 547–74.

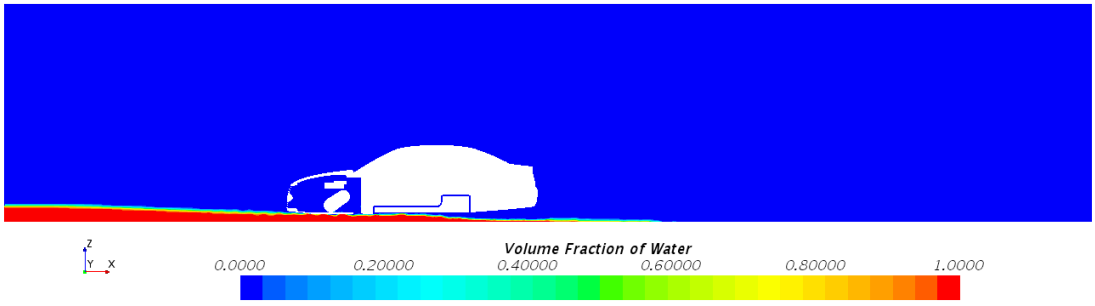
[32] *OpenFOAM User guide*. Version 5. CFD Direct. 2017. URL: `https://cfd.direct/openfoam/user-guide/` Accessed: Sept. 18, 2017.

[33] *OpenFOAM User guide*. OpenCFD. 2016. URL: `https://www.openfoam.com/documentation/user-guide/` Accessed: Feb. 3, 2018.

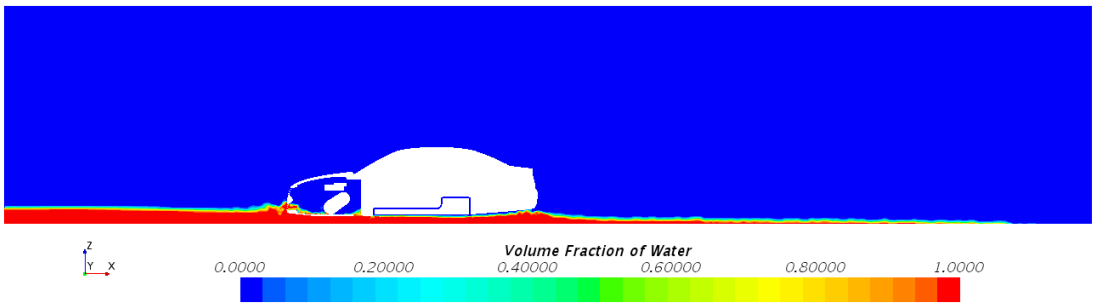
[34] *STAR-CCM+ Documentation*. Version 12.06. Siemens. 2017.

A

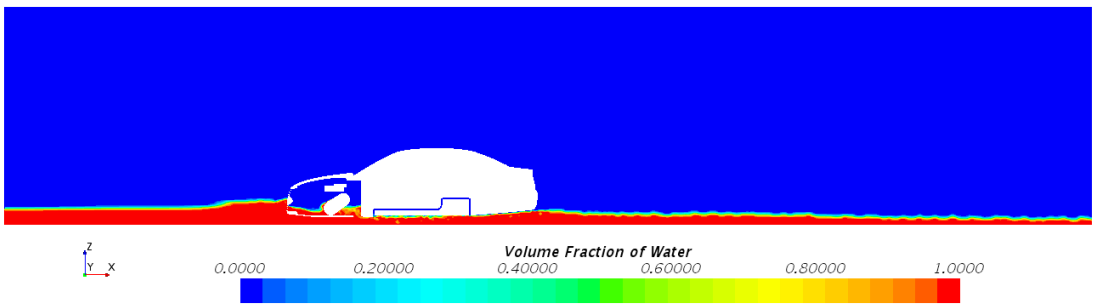
Water Surface Evolution



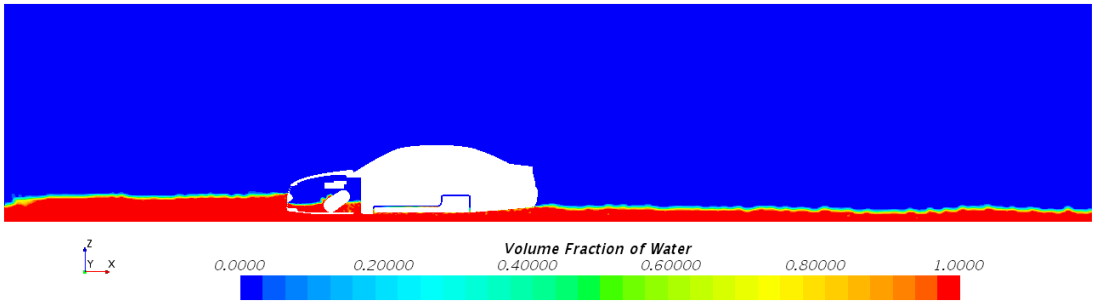
(a) *3 s*



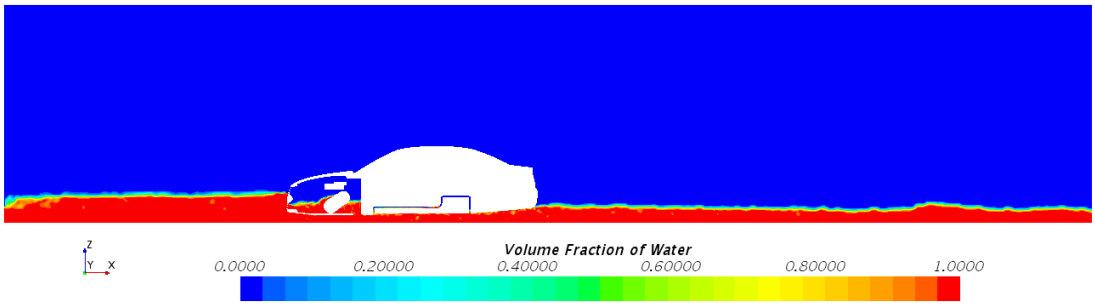
(b) *5 s*



(c) *10 s*



(d) *20 s*



(e) *40 s*

Figure A.0.0: *Volumetric phase fraction of water at a middle cross section in the y-direction at 5 different time steps, for simulation D-STAR.*

B

OpenFOAM files

Here the most important files used for the OpenFOAM simulations are presented. A simulation case in OpenFOAM consists of a case folder with three main directories, `0`, `constant` and `system`. The content of these directories is described and presented below.

B.1 0 directory

The `0` directory contains the boundary and initial conditions for the field variables used by the solvers, in this case the phase fraction of water, here `alpha.water`, the velocity, here `U`, the dynamic pressure, here `p_rgh`, and the turbulent viscosity, here `nut`. The boundary conditions are given for each boundary and the initial conditions are set with `internalField`. Since the initial internal field for the water phase fraction is non-uniform it consists of an array with each cell value, making the `alpha.water` file very long. Therefore, the file `alpha.water.orig` with a uniform field of no water in the domain is displayed here instead.

Listing B.1: alpha.water.orig

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  plus                                  |
|   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
        version 2.0;
        format ascii;
        class volScalarField;
        location "";
        object alpha1;
}
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/

dimensions [0 0 0 0 0 0 0];

internalField uniform 0.;

boundaryField
{
```

```
25          car
26          {
27                  type zeroGradient;
28          }
29
30          inlet_air
31          {
32                  type fixedValue;
33                  value uniform 0;
34          }
35
36          inlet_water
37          {
38                  type fixedValue;
39                  value uniform 1.;
40          }
41
42          outlet
43          {
44                  type inletOutlet;
45                  inletValue uniform 0;
46                  value uniform 0;
47          }
48
49          road
50          {
51                  type zeroGradient;
52          }
53
54          top
55          {
56                  type inletOutlet;
57                  inletValue uniform 0;
58                  value uniform 0;
59          }
60
61          sides
62          {
63                  type symmetry;
64          }
65
66  }
67
68  // ************************************************************************* //
```

Listing B.2: U

```
1  /*--------------------------------*- C++ -*----------------------------------*\
2  | =========                 |                                                 |
3  | \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
4  |  \\    /   O peration     | Version:  plus                                  |
5  |   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
6  |    \\/     M anipulation  |                                                 |
7  \*---------------------------------------------------------------------------*/
8  FoamFile
9  {
10         version 2.0;
11         format ascii;
```

```
        class volVectorField;
        location "";
        object U;
}
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/

dimensions [0 1 -1 0 0 0 0];

internalField uniform ( 2.0 0.0 0.0 );

boundaryField
{
        car
        {
                type noSlip;
        }

        inlet_air
        {
                type fixedValue;
                value uniform (2 0 0);
        }

        inlet_water
        {
                type fixedValue;
                value uniform (2 0 0);
        }

        outlet
        {
                type pressureInletOutletVelocity;
                value uniform (0. 0. 0.);
        }

        road
        {
                type noSlip;
        }

        top
        {
                type pressureInletOutletVelocity;
                value uniform (0 0 0);
        }

        sides
        {
                type symmetry;
        }

}

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

Listing B.3: p_rgh

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


dimensions [1 -1 -2 0 0 0 0];

internalField uniform 0.;


boundaryField
{
        car
        {
                type fixedFluxPressure;
                value uniform 0;
        }

        inlet_air
        {
                type fixedFluxPressure;
                value uniform 0;
        }

        inlet_water
        {
                type fixedFluxPressure;
                value uniform 0;
        }

        outlet
        {
                type totalPressure;
                p0 uniform 0;
                U U;
                phi phi;
                rho rho;
                psi none;
                gamma 1;
                value uniform 0;
        }

```

```
        road
        {
                type fixedFluxPressure;
                value uniform 0;
    }

        top
        {
                type totalPressure;
                p0 uniform 0;
                U U;
                phi phi;
                rho rho;
                psi none;
                gamma 1;
                value uniform 0;
        }

        sides
        {
                type symmetry;
        }

}

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

Listing B.4: nut

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


dimensions      [0 2 -1 0 0 0 0];

internalField   uniform 0;

boundaryField
{

    inlet_air
    {
            type calculated;
            value $internalField;
    }
```

```
30
31      inlet_water
32      {
33              type calculated;
34              value $internalField;
35      }
36
37      outlet
38      {
39              type calculated;
40              value $internalField;
41      }
42
43      road
44      {
45              type nutkWallFunction;
46              value $internalField;
47      }
48
49      top
50      {
51              type calculated;
52              value $internalField;
53      }
54
55      sides
56      {
57              type symmetry;
58      }
59
60      car
61      {
62              type nutkWallFunction;
63              value $internalField;
64      }
65
66  }
67
68  // ************************************************************************* //
```

B.2 constant directory

The `constant` directory contains a folder called `polyMesh` with all files for defining the mesh. These are however too long to include here. The most important file in the `polyMesh` directory is the `boundary` file where the boundaries of the geometry are defined. The `constant` directory also contains the files `transportProperties` and `turbulenceProperties`, which contain fluid properties and the turbulence model respectively. These files are included below. The directory also contains a file `g` with the definition of gravity.

Listing B.5: transportProperties

```
1  /*--------------------------------*- C++ -*----------------------------------*\
2  | =========                 |                                                 |
```

```
/ \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
/  \\    /   O peration     | Version:  plus                                  |
/   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
/    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
        version 2.0;
        format ascii;
        class dictionary;
        location "";
        object transportProperties;
}
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/


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

phases (water air);

water
{
    transportModel  Newtonian;
    nu              [0 2 -1 0 0 0 0] 1e-06;
    rho             [1 -3 0 0 0 0 0] 1000;
}

air
{
    transportModel  Newtonian;
    nu              [0 2 -1 0 0 0 0] 1.48e-05;
    rho             [1 -3 0 0 0 0 0] 1;
}

sigma           [1 0 -2 0 0 0 0] 0.07;


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

Listing B.6: turbulenceProperties

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  plus                                  |
|   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/


FoamFile
{
        version 2.0;
        format ascii;
        class dictionary;
        location "";
        object turbulenceProperties;
```

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

simulationType LES;

LES
{
        LESModel Smagorinsky;

        turbulence on;

        printCoeffs on;

        delta vanDriest;

        cubeRootVolCoeffs
        {
        deltaCoeff 1;
        }

        vanDriestCoeffs
        {
        delta cubeRootVol;

                cubeRootVolCoeffs
                {
                deltaCoeff 1;
                }

                smoothCoeffs
                {
                delta cubeRootVol;

                cubeRootVolCoeffs
                        {
                        deltaCoeff 1;
                        }
                maxDeltaRatio 1.1;
                }

        Aplus 26;
        Cdelta 0.158;
        }

}

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

B.3 system directory

The `system` directory contains all files with the solver settings. The file `controlDict` contains settings for the transient solver, settings for writing the result to file and some additional field variables and functions of interest. The file `fvSchemes` contains the discretization schemes used by the solvers for the different terms in the transport equations.

In the file `fvSolution` the equation solvers, algorithms and convergence tolerances are specified. These three files are included below. Apart from these files, the `system` directory also contains files for defining initial conditions in different regions of the domain and for decomposing cases for running in parallel.

Listing B.7: controlDict

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  plus                                  |
|   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/

FoamFile
{
        version 2.0;
        format ascii;
        class dictionary;
        location "";
        object controlDict;
}
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/

application interFoam;

startFrom startTime;

startTime 0;

stopAt endTime;

endTime 60;

deltaT 0.0002;

writeControl runTime;

writeInterval 0.1;

purgeWrite 0;

writeFormat     ascii;

writePrecision 6;

writeCompression uncompressed;

timeFormat general;

timePrecision 6;

graphFormat     raw;

runTimeModifiable yes;

```

```
adjustTimeStep off;

maxCo 20;

maxAlphaCo 1;

maxDeltaT 1.;

functions {

Co1
{
    type CourantNo;
    libs ("libfieldFunctionObjects.so");
    writeControl runTime;
    writeInterval 0.1;
}

forceCoeffs1
{
    type forceCoeffs;
    libs ( "libforces.so" );
    writeControl runTime;
    writeInterval 0.1;
    log yes;
    patches (car);
    rho rhoInf;
    rhoInf 1;
    liftDir (0 0 1);
    dragDir (1 0 0);
    CofR (1.399 0 0);
    pitchAxis (0 1 0);
    magUInf 2;
    lRef 2.787;
    Aref 2.162423;

}

yPlus1
{
    type yPlus;
    libs ("libfieldFunctionObjects.so");
    writeControl runTime;
    writeInterval 0.1;

}

}

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

Listing B.8: fvSchemes

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  plus                                  |
|   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
```

```
/    \\/     M anipulation  |                                                    |
\*---------------------------------------------------------------------------*/
FoamFile
{
        version 2.0;
        format ascii;
        class dictionary;
        location "";
        object fvSchemes;
}
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/

ddtSchemes
{
    default Euler;
}

gradSchemes
{
    default Gauss linear;
    grad(U) cellLimited Gauss linear 1;
}

divSchemes
{
    div(rhoPhi,U)  Gauss linearUpwindV grad(U);
    div(phi,alpha)  Gauss vanLeer;
    div(phirb,alpha) Gauss interfaceCompression;
    div(((rho*nuEff)*dev2(T(grad(U))))) Gauss linear;

    div(phi,U) bounded Gauss linearUpwindV grad(U);
    div(phi,k) bounded Gauss upwind;
}

laplacianSchemes
{
    default Gauss linear corrected;
}

interpolationSchemes
{
    default linear;
}

snGradSchemes
{
    default corrected;
}
fluxRequired
{
        default no;
        p_rgh;
        pcorr;
        alpha;
}

wallDist
{
```

```
    method meshWave;
}

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

Listing B.9: FvSolution

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  plus                                  |
|   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/


FoamFile
{
        version 2.0;
        format ascii;
        class dictionary;
        location "";
        object fvSolution;
}
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/


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

solvers
{
    "alpha.water.*"
    {
        nAlphaCorr 1;
        nAlphaSubCycles 3;
        cAlpha 1;

        MULESCorr yes;
        nLimiterIter 3;

        solver smoothSolver;
        smoother symGaussSeidel;
        tolerance 1e-8;
        relTol 0;
    }

    pcorr
    {
        solver PCG;
        preconditioner DIC;
        tolerance 1e-5;
        relTol 0;
    }

    p_rgh
    {
        solver PCG;
```

```
        preconditioner DIC;
        tolerance 1e-5;
        relTol 0.05;
    }

    p_rghFinal
    {
        $p_rgh;
        tolerance 1e-5;
        relTol 0;
    }

    U
    {
        solver smoothSolver;
        smoother symGaussSeidel;
        preConditioner DILU;
        tolerance 1e-05;
        relTol 0.1;
    }

    UFinal
    {
        $U;
        relTol 0;
    }
}

PIMPLE
{
    momentumPredictor yes;
    nOuterCcorrectors 1;
    nCorrectors 3;
    nNonOrthogonalCorrectors 1;
    pRefCell 18155803;
    pRefValue 0;

}


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