Paul Bischof, BSc

MASTER'S THESIS

Mesh Optimisation for CFD Application in External Aerodynamic Simulation

to obtain the academic degree

Master of Science

Graz University of Technology
Technische Universität Graz

Mechanical Engineering

Institute of Automotive Engineering

Member of Frank Stronach Institute

Head of the Institute: Univ.-Prof. Dipl.-Ing. Dr.techn. Peter Fischer

Academic Tutor:
Univ.-Prof. Dipl.-Ing. Dr.techn. Peter Fischer

Industrial Tutors at Magna Steyr:
Nicola di Nardo, MSc
Dipl.-Ing. Severin Stadler

Graz, May 2017

locked for 2 years

Acknowledgement

At this point I would like to say thank you to everybody who supported me during the preparation for this master's thesis and contributed in some way to the outcome of this thesis.

First and foremost, a special thanks to Nicola Di Nardo, MSc, from Magna Steyr for his patience and expertise. He supported me, especially during the practical part of this thesis, and was always the first person for me to contact; be it at a beer at the weekend or directly at work.

Furthermore, I would like to say thank you to Dipl.-Ing. Severin Stadler from Magna Steyr and Univ.-Prof. Dipl.-Ing. Dr.techn. Peter Fischer from TU Graz, who both supported me mainly in terms of organisation and methodology.

It is hard for a student of mechanical engineering who is not familiar with this discipline to follow a topic of numerical simulation like this. However, it is even harder for a non-technical student to follow this topic. Thus, special thanks to Tanja Erlacher, Student of English, who has proofread this extensive thesis.

Last but not least, I would like to thank my family as well as the TSV Graz. My family who has facilitated these studies to me and the TSV-members for being my best friends during and also after finishing my studies.

Paul Bischof

Graz, 15th of May 2017

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources. The document uploaded in TUGRAZonline is identical with this master's thesis.

Graz, ______________________ ______________________

Date Signature

Eidesstattliche Erklärung[1]

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

Graz, am ______________________ ______________________

Datum Unterschrift

[1] Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008; Genehmigung des Senates am 1.12.2008

Abstract

The target of this thesis was the systematic investigation of different mesh types using steady-state CFD simulation. Four different cars were tested. Furthermore, two transient CFD simulations were executed, using mesh types based on the results of the steady-state simulations.

At first, a basic theoretical overview of fluid computational dynamics was presented. Current as well as future methods were examined. Based on the Navier-Stokes equations, the elementary equations of each method were described. Furthermore, the numerical features and requirements as well as advantages and disadvantages were presented. Closing the theory chapter, the current development as well as future potential of each method were highlighted. Based on this, the entire program structure of the turbulence modelling of ANSYS Fluent was sketched and described. For comparison, also the structure of software packages CD-adapco STAR-CCM+ and open source software OpenFOAM were sketched. Before starting the actual investigation, some numerical aspects of CFD were discussed. Additionally, three earlier investigations on different CFD models were presented.

At the beginning of the investigation, all different mesh types were discussed and described. The meshing strategy of each mesh was presented and the cell numbers of each mesh were compared for each car. Abnormalities or difficulties in the mesh generation process were documented and its reasons described, in case they are known. Following the geometry description, the solver settings, for steady-state as well as for transient simulation, were discussed in detail. Special features, especially of the transient case, were described accurately. Subsequently, the results were presented. The main focus was on the convergence behaviour of the different mesh types and their accuracy. One of the four vehicles was investigated more detailed. The qualitative difference of the calculated flow between the various mesh types was revealed. After a comparison of the results, two mesh types were chosen for the transient simulation. These results were discussed in detail and a comparison with the steady-state results was presented.

Concluding, the results of the thesis were summarised again, discussed and open questions were, if possible, answered. In a short perspective, the further proceedings and the general development of CFD were described.

Kurzfassung

Das Ziel dieser Arbeit war die systematische Untersuchung von verschiedenen Netztypen mittels stationärer CFD Simulation an vier verschiedenen Fahrzeugtypen. Weiters wurden transiente CFD Simulationen mit zwei ausgewählten Netztypen an einem dieser Fahrzeuge durchgeführt und bewertet.

Einleitend wurde die grundlegende Theorie der numerischen Strömungssimulation erläutert. Dabei wurden sowohl aktuelle als auch zukünftige Methoden behandelt. Hierbei wurden anfangs, aufbauend auf den Navier-Stokes Gleichungen, die elementaren Gleichungen der jeweiligen Methode beschrieben und erklärt. Des Weiteren wurden numerische Besonderheiten und Anforderungen sowie die Vor- und Nachteile der Methoden dargelegt. Abschließend wurde die aktuelle Entwicklung wie auch das Zukunftspotential ein jeder Methode aufgezeigt. Darauf aufbauend wurde die Programmstruktur der Turbulenzmodelle der Solversoftware ANSYS Fluent skizziert und beschrieben. Zu Vergleichszwecken wurden zusammenfassend auch die Strukturen der Softwarepakete CD-adapco STAR-CCM+ und des freien Pakets OpenFOAM skizziert. Vor der eigentlichen Untersuchung wurden noch einige numerische Aspekte der CFD Berechnung diskutiert und drei frühere Untersuchungen verschiedener Methoden präsentiert.

Am Anfang der Untersuchung wurden die verschieden Netztypen diskutiert und beschrieben. Ebenso wurde die Vernetzungsstrategie eines jeden Netzes präsentiert und die Zellenzahlen der Netze für die verschiedenen Fahrzeuge miteinander verglichen. Auffälligkeiten oder Schwierigkeiten bei der Netzerstellung wurden dokumentiert und die Gründe dafür, sofern bekannt, beschrieben. Darauf folgend wurden die Solvereinstellungen sowohl für die stationäre, als auch für die instationäre Simulation detailliert dargelegt und diskutiert. Auffälligkeiten, besonders im Falle der instationären Simulation, wurden eingehend beschrieben. Anschließend wurden die Ergebnisse der Simulationen präsentiert. Das Hauptaugenmerk lag dabei auf dem Konvergenzverhalten der verschiedenen Netztypen sowie auf deren Genauigkeit. Eines der vier Fahrzeuge wurde detaillierter untersucht und die qualitativen Unterschiede der berechneten Strömung offengelegt. Einem Vergleich der Ergebnisse folgte eine Auswahl zweier Netztypen für die transiente Simulation. Diese wurden dann ausgiebig präsentiert und die Ergebnisse diskutiert. Diese Ergebnisse wurden in weiterer Folge mit jenen der stationären Simulationen verglichen.

Abschließend wurden die Ergebnisse der Arbeit noch einmal zusammengefasst, diskutiert und offene Fragen, sofern möglich, beantwortet. In einem kurzen Ausblick wurden noch die weitere Vorgehensweise sowie die allgemeine zukünftige Entwicklung des Gebietes der numerischen Strömungssimulation beschrieben.

Contents

Abbreviations and Formula Symbols

Arabic Formula Symbols

A	[m^2]	Area (cross-section)
a	[ms^{-2}]	Acceleration
c	[ms^{-1}]	Speed of sound
c_D, c_x	[-]	Drag coefficient
c_F	[-]	Skin friction coefficient
c_L, c_z	[-]	Lift coefficient
c_P	[-]	Pressure coefficient
CFL	[-]	CFL number
E	[m^{2}s^{-2}]	Energy
f	[-]	Elliptic relaxation for wall influence
f	[-]	Density distribution function
f^b	[ms^{-2}]	External force
F	[N]	Force
k	[m^{2}s^{-2}]	Turbulence kinetic energy
Kn	[-]	Knudsen number
L_0	[m]	Representative physical length scale
L_t	[m]	Turbulent length scale
L_{vK}	[m]	von Karman length scale
m	[kg]	Mass
Ma	[-]	Mach number
p	[Nm^{-2}]	Pressure
P_k	[m^{2}s^{-3}]	Rate of production of k
Re	[-]	Reynolds number
S	[m^{2}s^{-2}]	Shear (strain) rate tensor
s	[Jkg^{-1}K^{-1}]	Specific entropy
t	[s]	Time
U	[ms^{-1}]	Velocity
U^+	[-]	Dimensionless velocity
u	[ms^{-1}]	Velocity vector
u_τ	[ms^{-1}]	Wall shear stress velocity
$\overline{v^2}$	[m^{2}s^{-2}]	Velocity variance scale
x	[m]	Coordinate vector
y	[m]	Wall distance
y^+	[-]	Dimensionless wall distance

Greek Formula Symbols

δ	[m]	Boundary layer thickness
ϵ	[m^{2}s^{-3}]	Rate of dissipation of turbulence kinetic energy k
κ	[-]	Isentropic exponent
λ	[m]	Mean free path
μ	[kgm^{-1}s^{-1}]	Dynamic viscosity
ν	[m^{2}s^{-1}]	Kinematic viscosity
ν_t	[m^{2}s^{-1}]	Turbulence (kinematic) eddy viscosity
ρ	[kgm^{-3}]	Density
τ	[N]	Shear stress tensor
τ'	[N]	Reynolds stress tensor
τ_w	[N]	Wall shear stress
ω	[s^{-1}]	Specific rate of dissipation of turbulence kinetic energy k
ζ	[-]	Normalised wall-normal velocity scale

Subscriptions and Superscriptions

0	State of rest
$1, 2, 3$	Orthogonal vector directions
∞	Infinite wall distance or outside body influence
D	Drag
L	Lift
P	Pressure
SGS	Sub-grid scale
t	Turbulence

Abbreviations

BSL	(Menter) Baseline
CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
CFL	Courant-Friedrichs-Lewy
CPU	Central Processing Unit
DB	Database
DDES	Delayed Detached Eddy Simulation
DES	Detached Eddy Simulation
DNS	Direct Numerical Simulation
DS	Direct Simulation
E-DES	Embedded Detached Eddy Simulation
EARSM	Explicit Algebraic Reynolds Stress Model
EB	Elliptic Blending
ELES	Embedded Large Eddy Simulation
EVM	Eddy Viscosity Model
FDM	Finite Difference Method
FEM	Finite Element Method
FFT	Fast Fourier Transformation
FLOPS	Floating Point Operations Per Second
FR	Flux Reconstruction
FVM	Finite Volume Method
GUI	Graphic User Interface
HPC	High Performance Computing
IDDES	Improved Delayed Detached Eddy Simulation
LBM	Lattice Boltzmann Method
LES	Large Eddy Simulation
LGA	Lattice Gas Automata
MFD	Mimetic Finite Difference
MRF	Moving Reference Frame
NASA	National Aeronautics and Space Administration
NDG	Nodal Discontinuous Galerkin
NS	Navier-Stokes
OEM	Original Equipment Manufacturer
PANS	Partially-Averaged Navier-Stokes
PDE	Partial Differential Equation
PID	Property Identification
PITM	Partially Integrated Transport Model
RAM	Random Access Memory
RANS	Reynolds-Averaged Navier-Stokes
RC	Rotation/Curvature Correction

RKE	Realizable K-Epsilon
RNG	Re-Normalisation Group
RSM	Reynolds Stress Model
SA	Spalart-Allmaras
SA-RC	Spalart-Allmaras Rotation Correction
SAE	Society of Automotive Engineers
SAS	Scale-Adaptive Simulation
SBES	Stress Blended Eddy Simulation
SD	Spectral Difference
SDES	Shielded Detached Eddy Simulation
SGS	Sub-Grid Scale
SIMPLE	Semi-Implicit Method for Pressure Linked Equations
SRS	Scale Resolving Simulation
SST	Shear Stress Transport
SUV	Sport Utility Vehicle
TKE	Turbulence Kinetic Energy
TUI	Text User Interface
URANS	Unsteady Reynolds-Averaged Navier-Stokes
VLES	Very Large Eddy Simulation
WALE	Wall Adapting Local Eddy
WF	Wall Function
WM	Wall Model
WMLES	Wall Modelled Large Eddy Simulation
WMLES S-O	Wall Modelled Large Eddy Simulation Stress-Omega
WT	Wind Tunnel
ZDES	Zonal Detached Eddy Simulation
ZFLES	Zonal Forced Large Eddy Simulation
ZLES	Zonal Large Eddy Simulation

1 Introduction

"An aeroplane cannot live without aerodynamics, a car in contrast can!" This sentence from Wolf-Heinrich Hucho is probably the best periphrasis to describe the status of aerodynamics in automotive industry till the mid 1990s. [133] Where fluid mechanics play a key role in aeroplane as well as ship design, the design of a car is based more on requirements like comfort, safety and design.

Even if the first aerodynamically designed automobile dates back to 1899 (record vehicle of Camille Jenatzy [133]), it would take another few decades till the automotive industry starts to get into aerodynamics. When the Wright brothers pioneered the first sustained flight of a heavier-than-air aircraft on the 17th of December 1903, aerodynamic development took giant steps forward within the following decades.

After the first world war, aircraft industry was prohibited in Germany. Many aircraft engineers were looking for new jobs in the automotive industry. The first attempts to implement aerodynamics in a car solely aimed at reducing drag. Maybe the most famous of these early experiments was the Rumpler Tropfenwagen (see figure 1.1). Designed and built by the former aircraft designer Edmund Rumpler. The vehicle was introduced in September 1921 at the Berlin Motor Show and had a drag coefficient of 0.28. Until the introduction of the Opel Omega in 1986, no production car had been able to reach this value. Due to its unconventional shapes, borrowed from the aircraft industry, the car was a commercial flop. [133]

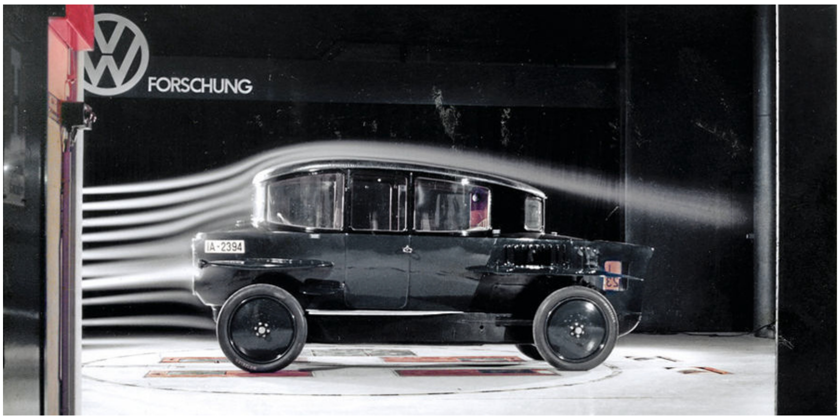


Figure 1.1: Rumpler Tropfenwagen from 1921 [133]

It took decades to separate the automotive from the aircraft aerodynamics. Till the late 1980s, the moving spirits behind the cars' architecture were fashion and new manufacturing technologies. Aerodynamically advanced cars were often designed by accident

and aerodynamics were at best stylistic devices for the designers. [133]
When the motorsports industry, especially Formula One, begun to discover the power of aerodynamics in the late 1960s, the automotive industry was still in a deep sleep concerning aerodynamic efficiency. Not before the first oil crisis in the mid 1970s, some attention was paid on reducing drag. Constantly rising demands on safety and comfort (and therefore rising weight), keeps the attention on reducing the aerodynamic drag to prohibit an increase of fuel consumption till today. [73]

1.1 Current Development in Formula One

Formula One is accepted to be the champions league in automotive aerodynamics. In no other industry, more money is spent on aerodynamic development. In 2014, a typical top class Formula One team employed about 300 design engineers. Half of them working in the aerodynamics department. A cleverly devised system including computational fluid dynamics (CFD), wind tunnel (WT) and track tests has been installed to push forward aerodynamic development. CFD is a pretty new technology introduced in the mid-1990s. WT and track tests have been well-known for decades - certainly with massive improvements of the used methods over the years. Back in 1976, Team Lotus took lodgings for WT tests at the Imperial College in London to explore race car aerodynamics. [118] Times have changed. In 2014, almost each Formula One team has its own WT (if not two). When BMW Sauber F1 Team introduced their supercomputer Albert2 in 2008, it was the most powerful industrial used machine in Europe. [49] Nowadays, each team has a massive CFD cluster to handle the ever-growing requirements of aerodynamics simulation. This development led to a memorable curiosity in Formula One.

The Virgin Experiment In the middle of 2008, the new Formula One technical regulations for the season 2009 were published. Especially the chassis rules were completely revised. In early 2009, a joint-venture of Formula Three team Manor Motorsport and the engineering company Wirth Research announced that they were preparing to join the Formula One world championship with the start of the 2010 season. They attracted worldwide motorsport attention with their unconventional concept to create the whole aerodynamic design exclusively via CFD and without any WT prove (see figure 1.2). Red Bull Racings chief technical officer Adrian Newey said: “It’s a new way. I think nowadays (2010) you need to combine both ways (simulation and WT). But maybe their car works well and I have to revise my point of view.” [114] He proved to be right.
Virgin Racing ended up at the bottom of the championship table in both 2010 and 2011. In the face of the disastrous results, the team split up the partnership with Wirth Research after the Grand Prix of Monaco 2011. The next year’s car was designed more conventional, with the complementary use of CFD and WT. Virgin was sold to Marussia Motors and the Virgin experiment came to an inglorious end with zero points in 38 races.
Nonetheless, a Formula One without CFD is unimaginable nowadays. Immediately after designing a new aero part, engineers start to create the CFD- as well as the WT model in a parallel process. If the results correlate within a specified range, the element is built

Figure 1.2: Virgin Racing VR-01, the first and only Formula One car designed exclusively via CFD [157]

and tested on the car. To verify the flow field directly on track, teams fit sensor arrays onto the car and compare this measured flow field on the real car with WT and CFD results. In addition to this, loads of pressure taps (up to some hundreds during trainings and about 60 during races) are spread all over the car and in same positions as well on the WT model to get a live tracking of the aerodynamic performance of the car.

1.2 Current Development in Automotive Industry

In motorsports, aerodynamics dictate the outer shape of the vehicle. And to go further, aerodynamics dictate the position of the entire mechanical system (in fine, the mechanics are aero slaves). In the automotive industry, it is the other way round. Aerodynamics are more a slave of safety (law) and comfort (customer) requirements.
At the International Motorshow Germany 2015, the IAA (Internationale Automobil-Ausstellung), Mercedes Benz introduced their new "Concept IAA" (see figure 1.3). IAA stands for Intelligent Aerodynamic Automobil and is a short description for the car's innovative concept. Manually or automatically at speeds of more than 80 kilometres per hour, the vehicle moves automatically from its so called "design-mode" into "aerodynamic-mode". Within this configuration, the car changes its shape. The rear extends by 390 mm to optimise the nearly drop-shaped rear section. Two flaps extend from the front bumper to improve the airflow around the front wheel arches and the rims get completely plane. The front splitter moves 60 mm rearwards to improve the airflow towards the flat underbody and an active radiator grill moves the gills according to the required cooling power. With this configurations, the car reaches an impressive drag coefficient of 0.19. Such a complex aerodynamic project would not be realisable without the extensive use of CFD simulation. It remains to be seen, if this is a trendsetting concept. [40]

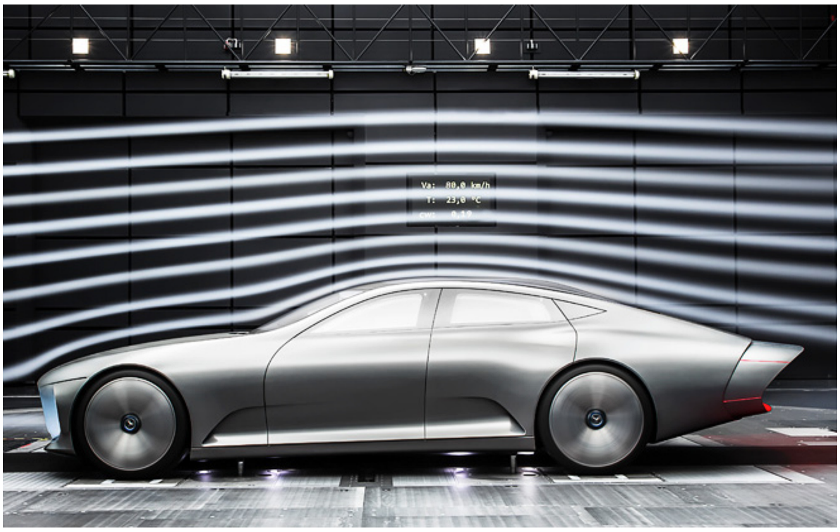


Figure 1.3: Mercedes-Benz "Concept IAA" [40]

1.3 Current Roll of CFD in the Industry

Since the mid 2000s, a big push towards hybrid methods has taken place (more on this in chapter 3). These methods are inherently unsteady. Large working groups have formed in order to investigate and refine these existing hybrid methods (NASA, Boeing, but also institutions in Europe). Some recent investigations are stated in chapter 6.
At higher Reynolds numbers, which certainly appear at cars at higher speeds, a laminar flow is likely to switch to a turbulent flow. As soon as a flow separates from a surface (for example tail of a car), the flow turns into a turbulent flow. Only unsteady methods are capable of catching a turbulent flow correctly. Turbulent flows are dissipative, meaning that kinetic turbulence energy (TKE) in small eddies is transformed into heat. [17] The best turbulence model is obviously the Navier-Stokes (NS) equations, which are almost universally accepted as an accurate description of turbulence. [21] Most current CFD methods are based on the NS equations.
For geometry without or with less flow separation, a steady-state simulation is capable of predicting drag forces very well. Once large separation areas appear, the drawback of the steady-state solution gets evident. That is the hour of unsteady simulation tools. Steady-state models typically under-predict the content of turbulence kinetic energy. This results in a too weak wake structure behind the body. Some steady-state turbulence models are also capable of predicting drag for properly separated flows (like a car model) relatively correct for specific geometry. But these models may fail completely for other geometry. [12] Only unsteady methods are able to simulate the complex turbulent flow field around bodies and produce acceptable results for each type of geometry. However, state-of-the-art steady-state simulations are able to predict drag of a ground vehicle at an accuracy of about three to eight per cent. [9]
Considering lift, steady-state is usually far off where unsteady solutions are mostly closer

within the experimental results. The reason for this can be found firstly in the fact that the area which is responsible for lift (base area) is a lot larger than the area which is responsible for drag (frontal area). The dynamic pressure change in x-direction (responsible for drag) is a lot higher than the dynamic pressure change in z-direction (responsible for lift). If it is assumed, that the pressure fluctuations underlie the same magnitude in x-direction as well as in z-direction, the relative changes of the resulting forces are a lot larger in z-direction due to the larger working area. [12]

Another major contribution to the deviation of the lift coefficient is the under-prediction of the TKE with steady-state methods. This comes due to the fact, that Reynolds-averaging damps out most of the turbulent fluctuations. With this comes a different volume and shape of the separated region and the wake. The wake of a car has a very high influence in drag, as well as lift coefficient. The fact that a steady-state simulation cannot predict flow separation correctly may also have a decent influence, especially at the rear axle lift coefficient. [12]

Also numerical errors play a role, but presumably a minor one. However, simultaneously to the push towards new hybrid CFD methods, new numerical discretisation methods with higher efficiency and lower numerical dissipation were put forward. [9] However, the unreliability of calculating the drag coefficient with CFD is a well known problem in the CFD-world and has not been completely understood and resolved yet.

1.4 Future Prospects of CFD

"Turbulence is the last unresolved problem of classical physics." [17, 47]

This quotation from Richard Feynman from 1964 is still valid today. Current CFD methods used in the industry are not able to reproduce the entire turbulent spectrum of a turbulent flow. And this will presumably not change within the next few decades (if not centuries). [146]

The following passages refer entirely to paper [139].

If compared to CAD or structural analysis, CFD still seems to be stuck in its infancy. However, big efforts are put into the development of more efficient methods (numerical as well as methodical) and more powerful hardware. Obviously, the significant bottlenecks of the CFD workflow are doubtlessly the mesh generation and adaptivity (pre-processing). Rapidly growing power of modern high performance computing (HPC) servers, which are capable of simulations with even higher resolutions, intensify these bottlenecks even more.

Another future problem is going to be the extensive amount of data which is produced by unsteady simulations. To handle this amount of data, new evaluation tools may be necessary.

A report, prepared for the NASA Langley Research Center, suggested a vision for a CFD workflow in 2030. This vision includes an automated management of errors and uncertainties and a much higher degree of automation in all steps of the analysis process. Furthermore, it should be able to effectively leverage the most capable state-of-the-art HPC hardware. A seamless integration of other complex engineering multidisciplinary

analyses and optimisations should also be included.
In addition, these vision plan to be able to simulate a full aircraft with wall resolved large eddy simulation (LES) or fully transient simulation for off-design turbofan engines. These goals should be achieved amongst others further developing HPC, physical modelling, numerical algorithms, geometry and grid generation and multi-disciplinary analysis and optimisation.
However, great potential is stuck not just within the computing methods, but even more within the pre-and post-processing steps of the CFD workflow.

2 Problem Statement

In current industrial CFD applications, the Reynolds-averaged Navier-Stokes (RANS) method is widely used and recognised as an established tool to predict the vehicles aerodynamic behaviour. *Magna Steyr* uses the RANS method in combination with tetrahedral mesh structures by default. The challenges of this thesis are to

- research and document the state-of-the-art of unsteady CFD simulation to generate a background for future decisions to use (or not to use) unsteady external aerodynamics simulation.
- pre-process four different real-life car models. This includes applying six different mesh types (tetrahedral, polyhedrals and hexahedral-hybrids) to each car model.
- run six RANS simulations with the different mesh types on the four different cars. That are 24 full car RANS simulations.
- compare the results of these 24 RANS simulations and determine the most efficient mesh type.
- execute unsteady simulation with the determined mesh type(s).
- post-process the results and compare the unsteady results with the steady-state RANS simulations.
- generate a recommendation for the use of unsteady CFD at *Magna Steyr* based on the research and simulation results.

Furthermore, a proposal for post-processing of unsteady aerodynamics simulation results should be created in order to have a basic template in case of the application of unsteady CFD methods.
The start impulse for this project was the fact, that some industries have started to introduce unsteady CFD methods. It is fashion so to say, to use these advanced methods. The first ones to use unsteady CFD methods industrially were the NASA and global players Boeing and Airbus. Meanwhile, the automotive instustry has also started to use these methods. So there is some pressure coming from the OEMs (Original Equipment Manufacturer) which wants their suppliers to use unsteady CFD. But to be able to execute simulations on this level, an extensive amount of computational resources is needed. Furthermore, a lot of experience is needed to run such a simulation reliably and correctly.
According to Moore's law, the number of components per integrated circuit is doubling every year. [112] This means the computational power is growing quite rapidly. But

even though the computational power is growing that rapidly, it is still not possible to compute pure unsteady simulations. Due to the limited computational power, hybrid methods (see chapters 3.4.5, 3.4.6, 3.4.7) had to be developed to compute unsteady simulations these days.
There must be a benefit when using these methods, if developers and industry put such a huge amount of work and money into this area to partially replace the relatively well established standard RANS methods. And of course there are benefits. The resulting flow field is much more detailed compared to a RANS simulation result and reproduces the real physics closer. Especially in massively separated flows the use of unsteady methods delivers much better results.
But unsteady simulations take a huge amount of computational power and therefore computing time. This leads to the questions this thesis should give an answer to:

- Why does *Magna Steyr* need to perform unsteady simulation?
- How can the process of unsteady simulations can be accelerated (In terms of which is the most efficient mesh)?
- What is the add-on value when using unsteady simulation?
- What are the relative costs of a transient simulation compared to a steady-state simulation?

The following thesis is an early step in this direction for *Magna Steyr* to evaluate unsteady methods for external aerodynamics simulation.

3 Theory of Computational Fluid Dynamics

3.1 What is Computational Fluid Dynamics?

A range of definitions on computational fluid dynamics (CFD) exists in literature.

> *"Computational Fluid Dynamics or CFD is the analysis of systems involving fluid flow, heat transfer and associated phenomena such as chemical reactions by means of computer-based simulation."* [153]

A more comprehensive definition can be found in the book "Computational Fluid Dynamics - An Introduction":

> *"The physical aspects of any fluid flow are governed by the following three fundamental principles: (1) mass is conserved; (2) F= ma (Newton's second law); and (3) energy is conserved. These fundamental principles can be expressed in terms of mathematical equations, which in their most general form are usually partial differential equations. Computational fluid dynamics is, in part, the art of replacing the governing partial differential equations of fluid flow with numbers, and advancing these numbers in space and/or time to obtain a final numerical description of the complete flow field of interest."* [3]

3.2 History

3.2.1 Early Developments

In 1755, Swiss physicist Leonhard Euler established the theory of classical hydrodynamics. In his thesis *"Principes généraux du mouvement des fluides"*, Euler described the mechanics of frictionless fluids with the so-called Euler equations. Essential in this work was his definition of pressure. With the addition of the equation of continuity, it was possible to solve this system of four unknowns coordinates-addicted velocities u_1, u_2, u_3 and pressure p. [141]

Almost 70 years later, in 1822, French engineer Claude Louis Marie Henri Navier published his papers in which he described the momentum equation for flow involving friction of Newtonian fluids. In 1845, Englishman George Gabriel Stokes managed to create a valid derivation of Naviers equations (actually Frenchman Adhémar Barré de Saint-Venant did this a few years earlier, it was his fault, not to publish his work). [141] This led to a system of nonlinear partial differential equations (PDE), including the equation of continuity and the NS equations for fluids with constant density $\rho = constant$ [1]: [149]

$$\frac{\partial u_1}{\partial x_1} + \frac{\partial u_2}{\partial x_2} + \frac{\partial u_3}{\partial x_3} = 0 \tag{3.1}$$

$$\frac{\partial u_1}{\partial t} + u_1 \frac{\partial u_1}{\partial x_1} + u_2 \frac{\partial u_1}{\partial x_2} + u_3 \frac{\partial u_1}{\partial x_3} = -\frac{1}{\rho}\frac{\partial p}{\partial x_1} + \nu \left(\frac{\partial^2 u_1}{\partial x_1^2} + \frac{\partial^2 u_1}{\partial x_2^2} + \frac{\partial^2 u_1}{\partial x_3^2} \right) + f_1^b \tag{3.2}$$

$$\frac{\partial u_2}{\partial t} + u_1 \frac{\partial u_2}{\partial x_1} + u_2 \frac{\partial u_2}{\partial x_2} + u_3 \frac{\partial u_2}{\partial x_3} = -\frac{1}{\rho}\frac{\partial p}{\partial x_2} + \nu \left(\frac{\partial^2 u_2}{\partial x_1^2} + \frac{\partial^2 u_2}{\partial x_2^2} + \frac{\partial^2 u_2}{\partial x_3^2} \right) + f_2^b \tag{3.3}$$

$$\frac{\partial u_3}{\partial t} + u_1 \frac{\partial u_3}{\partial x_1} + u_2 \frac{\partial u_3}{\partial x_2} + u_3 \frac{\partial u_3}{\partial x_3} = -\frac{1}{\rho}\frac{\partial p}{\partial x_3} + \nu \left(\frac{\partial^2 u_3}{\partial x_1^2} + \frac{\partial^2 u_3}{\partial x_2^2} + \frac{\partial^2 u_3}{\partial x_3^2} \right) + f_3^b \tag{3.4}$$

To keep a more general form, having a consistent scheme over the whole thesis, the continuous equation is given as

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{3.5}$$

and the momentum equation

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial x_j^2} + f_i^b \tag{3.6}$$

with

$$i = 1, 2, 3 \qquad \text{and} \qquad j = 1, 2, 3.$$

Analytically, this system of nonlinear PDEs is just solvable for some special cases. For general problems, only a numerical solution is possible. It would take more than hundred years, till solutions for those equations could be approximated by computers. The first methods were developed to resolve linearised potential equations in the 1930s. [109] One of the earliest attempts of calculations, in terms of CFD, were done by Lewis Fry

[1] To keep in mind: strictly speaking *incompressible* means only, that [149]

$$c^2 = \left(\frac{\partial p}{\partial \rho} \right)_s \gg u^2, \quad \text{that is} \quad Ma = \frac{u^2}{c^2} \approx 0$$

Richardson. Even if he failed dramatically, early CFD calculations during the 1940s used derived methods from Richardson's 1922 book. [125] In 2000, the Clay Mathematics Institute (CMI) in Peterborough, New Hampshire, United States, published its list of the seven mathematical problems that had not been solved by the year 2000, the so-called Millennium Prize Problems. One of these still unresolved problems is the proof of the existence and smoothness of the NS equations. [108]

3.2.2 1960s - The Fundamentals

Most of the current CFD codes are based on methods which were developed in the 1960s. From 1957 to the late 1960s, a group of scientists at the Los Alamos National Laboratory developed a variety of numerical methods to simulate transient two-dimensional fluid flows with the NS Equations. [64]
In 1963, Joseph Smagorinsky developed the theoretics for LES at the National Center for Atmospheric Research. [140] However, the first explorations on LES were done by Deardorff almost ten years later, in 1970. [42]

3.2.3 1970s - First Applications in Aerospace Industry

In the following years, companies, like Boeing (leading the way) [45], Lockheed [159], Douglas Aircraft [68], McDonnell Aircraft [26] and the NASA [8], developed three-dimensional panel codes. Herefrom it is clearly visible, that the aerospace industry was the initiator of the development of new CFD codes. The NASA's code PMARC (Panel Method Ames Research Center) is also commercially available. [116] Panel methods are based on the potential flow theory (see chapter 3.4.1) and are not in use anymore. [72] Developers turned then from the panel methods to full potential codes in the mid 1970s. This happened due to the drawback of panel codes, which are not able to calculate the nonlinear flow present at transonic speeds. The first full potential code was invented by Boeing in 1970. [115]

3.2.4 1980s - New Methods

In the 1980, the development started to move away from the potential flow theory towards the Euler equations (see chapter 3.4.2). This allows more accurate solution to transonic flows. All today's worldwide commercial and government codes are based on algorithms developed in the 1980s and 1990s. [78] The first three-dimensional Euler-code (FLO57) was developed by Anthony Jameson et al. in 1981. [79]
By the end of the decade, the first lattice methods appeared. The first one was the lattice gas automata (LGA, see chapter 3.4.10), which was the predecessor of the lattice Boltzmann method (LBM, see chapter 3.4.11). [63, 151]

3.2.5 1990s - Establishment of CFD in the Industry

In the 1990, the NS equations, which are the ultimate target in flow simulation, has been established in code development. With this came the intense development of turbulence

models. In 1992, Spalart and Allmaras from Boeing introduced their one equation turbulence model (see chapter 3.4.3) for aerospace applications. [147]
With his paper *"Advances in DNS/LES"*, Spalart started the development of hybrid RANS-LES methods (see chapter 3.4.6) The first detached eddy simulation (DES) method was introduced in 1997 as DES97. Today a wide range of these methods from different developers exists. [142, 146]

3.2.6 2000s - Development of Advanced Codes

In the first decade of the new millennium, a proper race for future technologies started. Loads of new methods, prior in the DES family, appeared. The new family of the partial integration models, namely partially-averaged Navier-Stokes (PANS) and partially integrated transport model (PITM), was also proposed in the mid 2000s. With the growth of the computational power, these methods got more and more feasible for industrial applications. And it was again the aircraft industry, this time together with Formula One, to be the main pusher for the development of these new methods. [56, 131, 146]

3.2.7 Since 2010 - Current Development

In the current decade more emphasis is put on validating and improving new methods. It is conceivable, that the strong family of hybrid RANS-LES methods will get competition from the new group of the partial integration models in the near future. LBM has also gained a strong postion in the commercial market and is already seen as an alternative to LES in aerospace industry. [117]
Today's discretisation codes base almost entirely on the finite volume method (FVM). The drawback of this method is, that it has a certain numerical dissipation. Therefore, a trend has appeared today to develop new dicretisation schemes with less numerical dissipation and a higher mathematical efficiency (refer also to chapter 3.4.4 LES). [78]

3.3 Some Definitions

To analyse and set up flow problems calculated by CFD, a range of numbers were introduced over the years. This chapter gives a very brief introduction and definitions of some of these numbers, which are essential to understand the context of this thesis. Please keep in mind, that this subsection is not intended to give an complete overview, but a very short introduction on the most important stuff.

3.3.1 Non-Dimensional Numbers

One of the most significant non-dimensional number in fluid dynamics is the Reynolds number. This number represents the ratio of inertial to viscous forces and is used to characterise a flow as laminar or turbulent. [72] The Reynolds number is defined as [149]

$$Re = \frac{\rho U L}{\mu} = \frac{UL}{\nu}. \tag{3.7}$$

Another group of dimensionless coefficients are the

- pressure coefficient [72]
$$c_P = \frac{p - p_\infty}{\frac{\rho}{2} U_\infty^2} \tag{3.8}$$
- drag coefficient [72] and
$$c_D = \frac{F_D}{\frac{\rho}{2} U_\infty^2 A} \tag{3.9}$$
- lift coefficient [72]
$$c_L = \frac{F_L}{\frac{\rho}{2} U_\infty^2 A}. \tag{3.10}$$

These three non-dimensional coefficients are used to quantify pressure, drag and lift. With a dimensionless quantity it is possible to compare different bodies without referring to their dimension. They are a very useful tool to determine the aerodynamic efficiency of a body (refer to chapter 9). [72]

3.3.2 Kolmogorovs Microscales and Dissipation

In turbulent flows, it is important to define the Kolmogorov length scales. The Kolmogorov microscales are the smallest scales in terms of length, time and velocity. In lower scales, the flow is dominated by the viscosity and the TKE is dissipated into heat (see figure 3.1). The Kolmogorov length scale is helpful in mesh generation as a reference for the minimum mesh size (refer also to chapter 3.4.4 LES). The Kolmogorov length scale is about reversed proportional to the Reynolds number. The scales are defined as the [17]

- Kolmogorov length scale [17]
$$\eta_k = \left(\frac{\nu^3}{\epsilon} \right)^{1/4}, \tag{3.11}$$
- Kolmogorov time scale [17] and
$$\tau_k = \left(\frac{\nu}{\epsilon} \right)^{1/2} \tag{3.12}$$
- Kolmogorov velocity scale [17]
$$u_k = (\nu \epsilon)^{1/4}. \tag{3.13}$$

Basically, the Kolmogorov scales define the range where TKE or dissipation is dominating the flow (see figure 3.1). This characteristic is essential to understand the intention of the unsteady methods described in chapter 3.4.

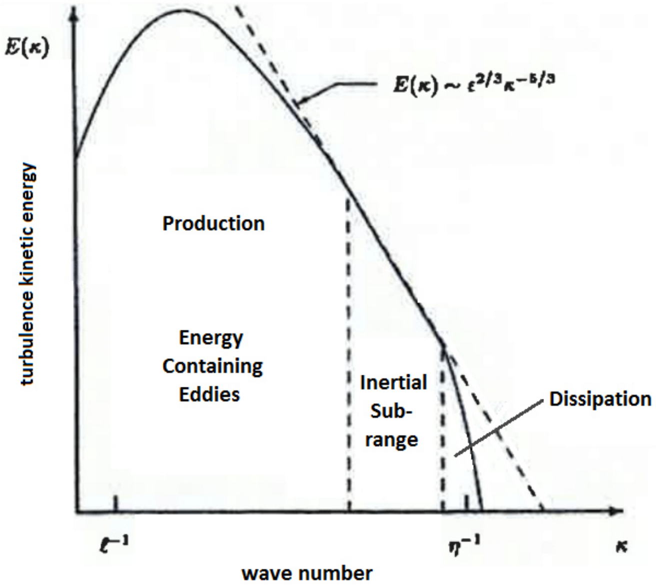


Figure 3.1: Energy or Kolmogorov spectrum for a turbulent flow (log-log scales)[2][17]

3.4 Methods

Over the past few decades, a variety of different CFD codes were developed. Overall, this codes can be divided into two different methodologies: [72]

- Navier-Stokes based methods
- Lattice based methods

Within the Navier-Stokes based group, a further division can be stated as: [72]

- Steady-state methods (time independent)
- Unsteady methods (time dependent)[3]

[2] $E(\kappa)d\kappa$ is the TKE, contained between the wave numbers κ and $d\kappa$.
[3] Lattice based methods are inherently unsteady.

The first suitable numerical codes solved the NS or Euler equations and were developed in the mid 1960s. The lattice methods are quite a recent development. The idea on the lattice methods (see chapter 3.4.10 and 3.4.11) dates back to the late 1980s. Where the Navier-Stokes based methods are well established in industry, the lattice methods are a very new CFD technology and just implemented in very few commercial codes. NS equations are 2nd order PDEs compared to LBM, which are based on 1st order equations. This results in extensive benefits during the computing process. In medium-term it is expected, that LBM will catch some market share from the classical NS based codes. [72, 117, 133]
The following chapters describe the most common established as well as new equation governing methods in CFD. Beside a brief theoretical description of the procedure, also the state-of-the-art of each method is discussed. In this regard, it is important to keep in mind, that state-of-the-art comprises only the top of the current development and not the practices used in the industry. All these methods are presented for the case of an incompressible fluid.

3.4.1 Potential Flow Theory

The potential flow theory was the very first approach, which allowed to apply numerical methods in terms of aerodynamics simulation. As already mentioned in chapter 3.2, the first methods were developed in the late 1960s and early 1970s and based on the potential flow theory. The potential flow theory set a row of limitations to simplify the flow field description: [22]

- Inviscid
- Incompressible $\nabla \cdot V = 0$
- Irrotational $\nabla \times V = 0$
- Steady $\frac{\partial}{\partial t} = 0$

These methods do not have any meaning in automotive engineering any more. They were replaced by the RANS method eventually in the 1990s. [72]

3.4.2 Euler Method

Basics of the Euler Method

The simplest form of the NS equations results from the negligence of compressibility and friction. These equations are known as the Euler equations. In contrast to the potential theory, the restriction of the irrotationality is not valid anymore. Furthermore, these equations were the moving spirit of the development of the finite volume method (most of the current CFD codes base on the FVM). Reason enough to give a little overview on this relatively simple method.

With the equation of continuity for the incompressible case [72]

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{3.14}$$

and the frictionless momentum equation (Euler equation) [72]

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho} \frac{\partial p}{\partial x_i} + f_i^b, \tag{3.15}$$

this results in a system of nonlinear PDEs of first order. Applications for these equations are laminar flows. As friction is not considered, no boundary layer is present and therefore no special wall treatment, numerical as well as methodical, is necessary. This means, a relatively coarse grid can be applied which decreases the computational requirements. However, the discretisation effort is about the same as with the RANS method (see chapter 3.4.3) which is one reason, why this method has not become a standard tool in engineering application. [72]

Current Development of the Euler Method

This method has lost attention in engineering application due to the rapid development of the RANS method. As the equations do not consider friction, no flow separation can be calculated. But as a passenger car is a (bluff) body, full of separations, this method is not applicable for automotive aerodynamics. [72] However, in aerospace this method was in use till the late 1990s when it was gradually replaced by the RANS method (which is in progress to get replaced again in some areas by hybrid RANS-LES methods today. See chapters 3.4.5 and 3.4.6.). [17]

3.4.3 (Unsteady) Reynolds-Averaged Navier-Stokes ((U)RANS)

Basics of RANS

An exact solution for the NS equations only exists for some special cases, like creeping flow according to Hele-Shaw and laminar flows according to Couette and Hagen-Poiseuille. [132] For technical applications, flow usually is turbulent. This requires a numerical solution and, to go further, some simplifications to the NS equations[4]. The most used and established method for this is the Reynolds-averaged NS (RANS) solution. The RANS method bases on splitting the flow values velocity and pressure into a time independent averaged and an overlying high frequent time dependent fluctuating value as can be seen in equation 3.16. [133]

$$u_i = \overline{u}_i + u_i' \qquad \text{and} \qquad p = \overline{p} + p' \tag{3.16}$$

[4] In fact it is indeed possible, even if not practical, to solve the NS equations without any simplifications. Refer to chapter 3.4.9 for this.

The values $\overline{u_i}$ and $\overline{p}$ denote the time averaged values where u_i' and p' denote the fluctuations (see figure 3.2). By implementing these into the NS equations, the complete equation of motion is [17]

$$\frac{\partial \overline{u}_i}{\partial t} + \overline{u}_j \frac{\partial \overline{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \nu \nabla^2 \overline{u}_i + \frac{1}{\rho}\frac{\partial \tau_{ij}'}{\partial x_j} \tag{3.17}$$

with the Reynolds stress tensor [17]

$$\tau_{ij}' = \rho \cdot \overline{u_i' u_j'} = \rho \cdot \begin{pmatrix} \overline{u_1'^2} & \overline{u_1' u_2'} & \overline{u_1' u_3'} \\ \overline{u_1' u_2'} & \overline{u_2'^2} & \overline{u_2' u_3'} \\ \overline{u_1' u_3'} & \overline{u_2' u_3'} & \overline{u_3'^2} \end{pmatrix}. \tag{3.18}$$

This equation is then called Reynolds equation.

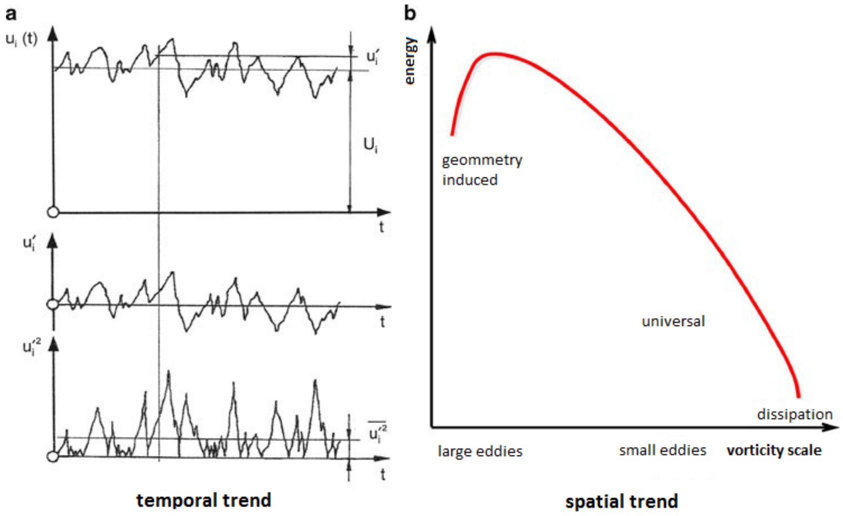


Figure 3.2: Time dependent behaviour of the velocity $u_i(t)$ in a turbulent boundary layer (**a**), energy spectrum of the turbulent motion (Kolmogorov-spectrum, **b**) [133]

If the time derivative $\frac{\partial \overline{u}_i}{\partial t}$ on the left hand side is neglected, it is spoken from RANS otherwise from unsteady RANS or URANS. The only differences to the complete NS equation are, that the time dependent values u_1, u_2, u_3 and p are replaced with their time averaged values $\overline{u}_1$, $\overline{u}_2$, $\overline{u}_3$ and $\overline{p}$ and the addition of the Reynolds stress tensor with its turbulent fluctuation values on the right hand side. This equations are then named Reynolds equations. With this definition, the turbulent mixing motion implies an exchange of momentum which generates additional stresses. These stresses are equivalent to an increase of the fluids viscosity (model error). [17] With the conversion of the NS to (U)RANS equations, an increase of unknowns comes along (namely the turbulent fluctuation terms). To resolve these unknowns, a so called turbulence model is needed.

Turbulence Models

A turbulence model connects the fluctuation motion $\overline{u_i'u_j'}$ with the time averaged values $\overline{u}_i$. Figure 3.3 shows the classification of the different available turbulence models.

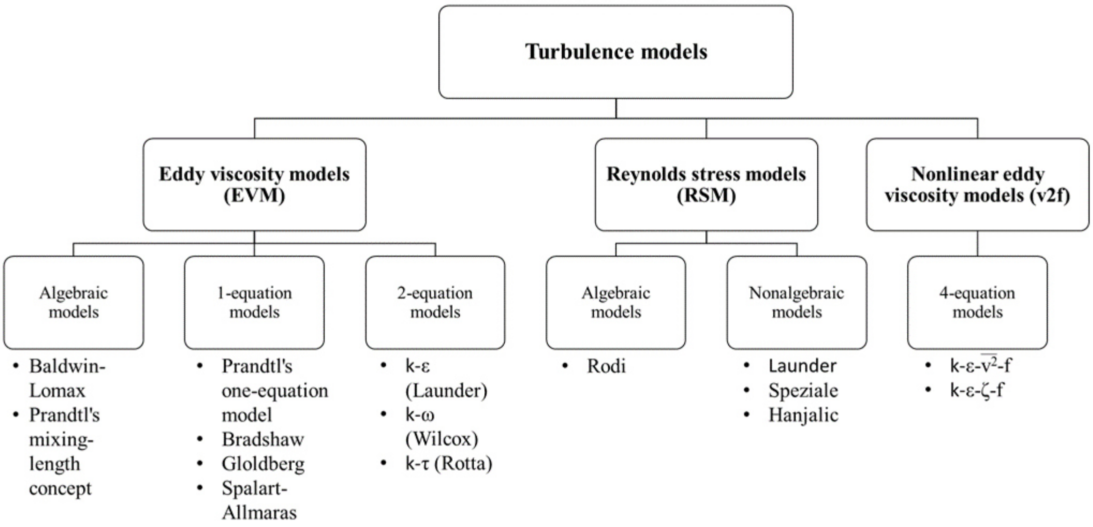


Figure 3.3: Classification of different turbulence models [17, 133]

About 90 per cent of industrial applications are using a $k-\epsilon$- or a $k-\omega$-model. These are so called eddy viscosity models (EVM) and are very robust and stable in almost every type and quality of mesh and flow. EVMs are based on Boussinesq's approximation[5] [20] and have, compared to the Reynolds stress models (RSM), the drawback, that they are pretty bad in predicting tumbling, swirling and mixing in flows. Their representation of shear stresses is better. This means, for internal flow problems (like combustion engines) it is more likely to use RSMs, in external flows EVMs. But there is no general guideline where to use which kind of model. It depends strongly on the application. [17] The following listing shows the currently most important turbulence models:

- **Spalart-Allmaras (Spalart and Allmaras 1992) (SA):** SA is a one-equation linear EVM which solves a modelled transportation equation for the turbulence kinematic eddy viscosity. [147] As it was developed by the NASA it is more suitable for aerospace than for automotive applications. With the appearance of the hybrid RANS-LES methods in the mid 2000s, the SA model experienced a revival due to its simplicity and low computational costs. [7, 17, 137]

[5] The relation between Boussinesq's approximation and the Reynolds stresses can be found in the appendix. Also the mathematical description of a related EVM is presented in the appendix.

- **$k-\epsilon$-model (Launder and Sharma 1972)** The different $k-\epsilon$-models are probably the most known turbulence models. They are two-equation linear EVMs, where k denotes the TKE and ϵ the rate of dissipation (of TKE k). The model is very stable in every type of mesh and computationally very efficient as the eddy viscosity is defined to be isotropic. [133] This model is suitable for calculating the free stream flow field outside the boundary layer. [88] It is the standard and most used turbulence model in CD-adapco STAR-CCM+. [17, 137]

- **$k-\omega$-model (Wilcox 1988):** The $k-\omega$-model is, like the $k-\epsilon$-model, also a linear eddy viscosity two-equation model, where k denotes again the TKE and ω the specific rate of dissipation (of TKE k). In contrast to the $k-\epsilon$-model, the $k-\omega$-model is more suitable for boundary layer flows. Especially the capability of prediction of separation is a big advantage over the $k-\epsilon$-model. [17, 158] A mathematical description of the hybrid $k-\omega$-SST model can be found in the appendix.[6]

- **Reynolds Stress Model (RSM) (Launder, Rodi and Reece 1975):** The RSM is a second order closuere turbulence model. It uses six plus one equations, which represent six independent Reynolds stresses and an additional closing equation. Thus, the RSM offers an exact representation of the convective stress transport. Its big advantage compared to EVMs is, that it resolves all components of the turbulent transport where EVMs uses isotropic eddy viscosity. Therefore RSM is computationally much more consuming than EVMs (about four times more computing time with the same mesh [99]). The big drawback is its limited robustness in meshes with poor quality. [17, 89]

- **$k-\epsilon-\overline{v^2}-f$-model (Durbin 1991, 1993, 1995):** The $k-\epsilon-\overline{v^2}-f$- or v2f-model is a hybrid model, based on the standard $k-\epsilon$-model with two additional equations, which include near-wall turbulence anisotropy and non-local pressure-strain effects. This makes the need of a wall model (see next chapter) unnecessary. The additional terms $\overline{v^2}$ and f describe the velocity variance scale and the elliptic relaxation for wall influence respectively. $\overline{v^2}$ can be seen as the velocity scale fluctuation normal to the streamlines. The v2f model is most comparable with the non-isotropic RSM, but is as robust as a two-equation EVM. Another variety within this hybrid group is the $k-\epsilon-\zeta-f$-model. Instead of the wall-normal velocity fluctuation $\overline{v^2}$, it uses the normalised wall-normal velocity scale $\zeta = \overline{v^2}/k$. [5, 17, 18, 44, 119]

In contrast to linear EVMs, RSMs can predict swirl. EVMs, linear, as well as nonlinear, are, due to their isotropic eddy approach, computationally much more efficient than RSMs. [17]

[6] This model is also used for the steady-state simulations in the practical part of this thesis.

Wall Models

"First, from a practical perspective, near-wall turbulence is of major interest, for it dictates the wall shear stress, which is responsible for friction drag, and the wall heat transfer, which is the key for cooling and heating." [90]
According to Ludwig Prandtl, a flow field around a body can be separated into two regions. A near-wall region around the body, where the fluid viscosity has a high influence on the velocity profile and the free stream, where the potential theory (no friction, no rotation, see chapter 3.4.1) is valid. This near-wall region is called "boundary layer". [133]
For flow simulations, it is essential to take account of turbulent motion caused by viscosity influence and wall proximity. At very low Reynolds numbers, the viscosity affects all turbulent interactions by causing departure of local isotropy. Another effect of a solid wall is its influence on normal to the wall fluctuations. A wall is primarily damping this normal to the wall turbulent fluctuations, forcing the turbulence to approach in a two dimensional state. With respect to wall treatment, two categories of turbulence models are common: low and high Reynolds number models. High Reynolds number models, which are mostly used in automotive aerodynamics simulation, are used in conjunction with wall functions. For the sake of completeness, it should be mentioned, that low Reynolds number models use a refined mesh close to the wall to resolve all important physics. However, they are not common in automotive aerodynamics. [17]
The use of wall models offers the following advantages: [17]

- Wall functions reduce computational requirements, as the wall-near turbulent regions do not have to be resolved.
- Wall functions improve numerical stability.
- Wall functions improve convergence speed, as the wall-near regions are calculated by wall functions. These wall functions are much simpler compared to the NS equations, which are applied outside the wall-near regions.

Figure 3.4 shows the difference between a boundary layer modelled by a wall function and a fully resolved boundary layer. If the boundary layer is fully resolved, the near-wall mesh has to be strongly refined. It is obvious that applying such a fine mesh around a complete car, the computational requirements increase a lot.
The dimensionless wall distance y^+ is defined as [72]

$$y^+ = \frac{u_\tau y}{\nu} \quad \text{with} \quad u_\tau = \sqrt{\frac{\tau_\omega}{\rho}} \tag{3.19}$$

and the dimensionless velocity u^+ as [72]

$$u^+ = f(y^+) = \frac{u}{u_\tau} \tag{3.20}$$

where u_τ is the friction or shear velocity, y the absolute wall distance, ν the kinematic viscosity, τ_ω the wall shear stress, ρ the fluid density and u the flow velocity. [133]

In CFD, y^+ is used to define the thickness of the first mesh layer around the surface. According to figure 3.4, the boundary layer is split into various layers: [17]

- Outer layer or free stream, where the potential flow theory is valid
- Inner layer, which is again split into:
  - Log-law region ($y^+ > 30$): The mean velocity is proportional to the logarithm of the wall distance.
  - Viscous sublayer ($y^+ > 5$): The Reynolds shear stress is negligibly compared to the viscous stress and therefore the mean velocity (almost) proportional to the wall distance.
  - Buffer layer ($5 < y^+ > 30$): The region between the viscous sublayer and the logarithmic-law region.

Mathematically, the non-dimensional velocity of the viscous sub layer and the logarithmic wall region is

$$u^+ = \begin{cases} y^+ & \text{if} \quad y^+ < 11.63 \\ \frac{1}{\kappa} ln\left(Ey^+\right) & \text{if} \quad y^+ \geq 11.63 \end{cases},$$

where E is an empirical integration constant, which is 9.793 for smooth walls and $y^+ = 11.63$ an empirical value, where, according to figure 3.4, the lines of the buffer layer and the log layer intersects. [17]

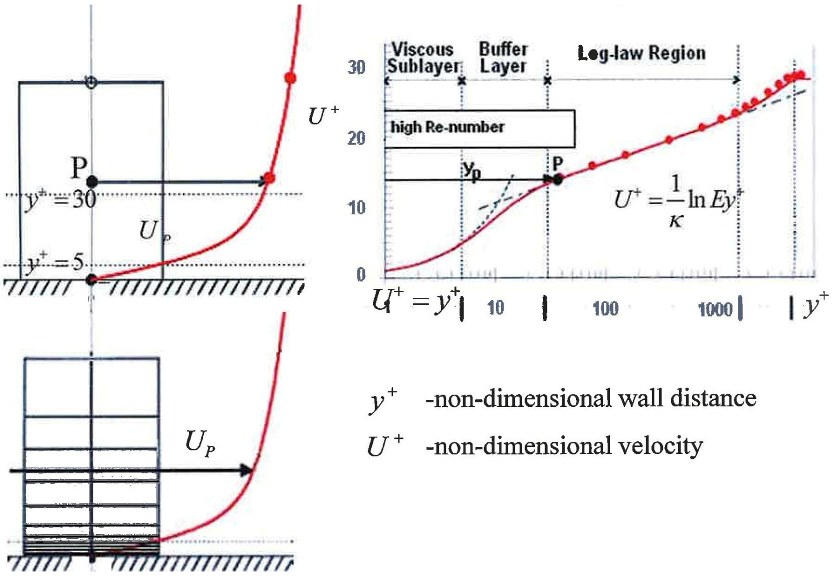


Figure 3.4: Comparison between a fully modelled (top left) and a fully resolved (bottom left) boundary layer [17]

Currently there is a trend towards defining hybrid wall treatments (commercial software developer have various names for this: universal wall treatment, compound wall treatment and others). One of the most popular wall models is Menter's SST model. [102] The big players in the industry like Airbus or Boeing almost entirely use this approach. [17] This (hybrid-) model blends a $k-\omega$-model near the wall with the $k-\epsilon$-model in wall-remote regions. [102] Refer to the appendix for a mathematical description of the $k-\omega$-SST model.

Apart from the standard wall function, it is tricky to implement more advanced wall models in unstructured meshes. [17]

Current Development of RANS

Some time ago, the automotive industry tried to establish LES (see chapter 3.4.4). When it was recognised, that with the current computational power it is not possible and economic to resolve a flow problem round a car with LES, RANS started to become the standard tool in (not just) automotive engineering. Toady's industrial standard is to perform loads of steady-state instead of few unsteady simulations. [17]

RANS has proven to be a relatively accurate and reliable simulation tool. However, the usability of the method is generally limited to relatively well behaved flows without significant flow separation or unsteadiness. A RANS simulation delivers a good impression of the drag and its distribution over the object. In contrast to this, the flow field, especially in challenging flow regions with large separation areas, is less satisfactory. Chapter 6.3 shows a comparison of the flow field around and behind a generic car model. This renderings (figure 6.8 and 6.9) show a noticeable difference in the shape of the wake. Due to the eddy viscosity approach, the predicted wake structure can be quite different compared to the actual wake structure. This difference is likely to be the major contribution to the difference observed in drag and especially lift coefficient. [12] With a good CFD code, a RANS simulation delivers results for drag which is within three to eight per cent of the WT results. [9, 50]

Figures 3.5 and 3.6 are showing a relatively accurate conformity between measured and simulated drag values and a decent difference between measured and simulated lift coefficients. These simulations were done by CD-adapco STAR-CD. Several different vehicle specifications (notchback, hatchback, fastback, estate tail) on a SAE reference-body were simulated and compared with WT measurements. [133] These simulations confirm the problems of current RANS CFD simulations. A good conformity in drag coefficient, but a relatively poor conformity in lift coefficient prediction.

The main development area of the RANS method is turbulence modelling. However, the large working groups developing turbulence models, like in the NASA Langley Research Center, are history since the mid 1990s. Due to the limits of RANS in more complex flows, there was hardly any development in the last ten years in this direction any more. The main research has been relocated to hybrid RANS-LES methods (see chapters 3.4.6 and 3.4.7) since the early 2000s. [17]

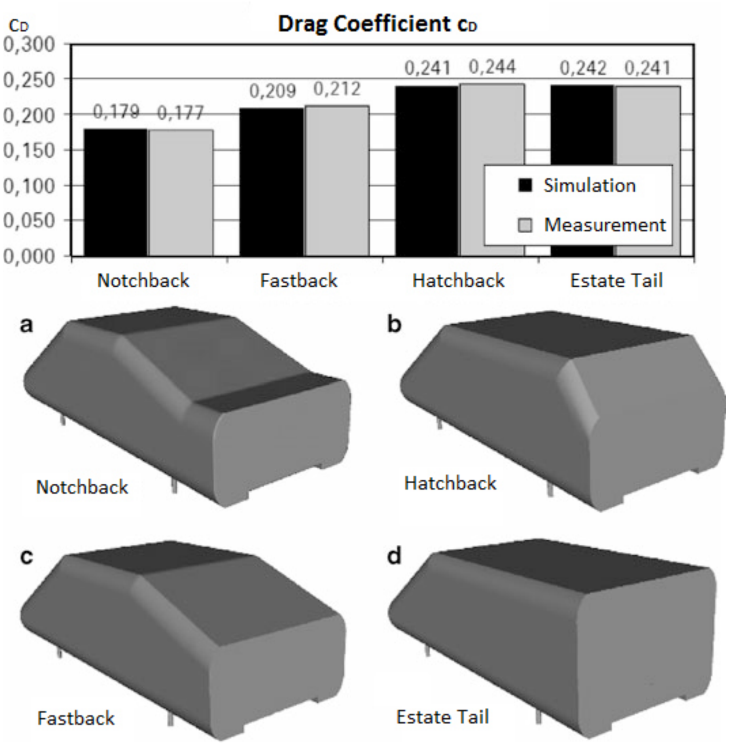


Figure 3.5: Comparison of measured and simulated drag coefficients on a SAE reference-body [156]

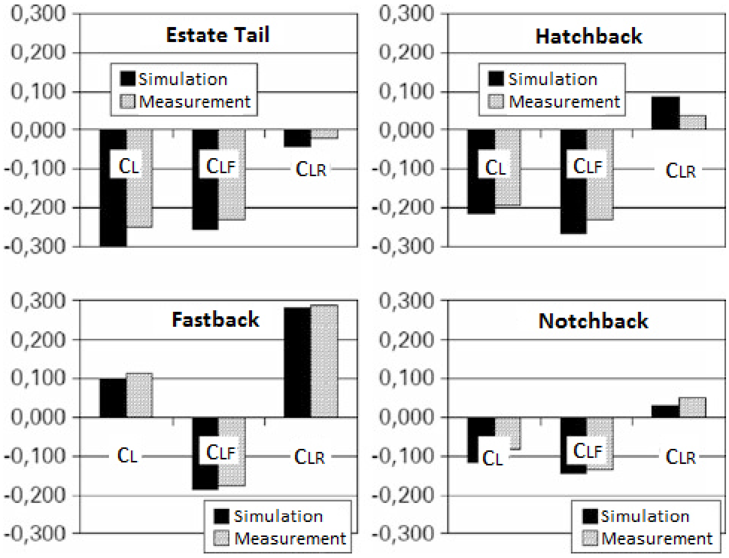
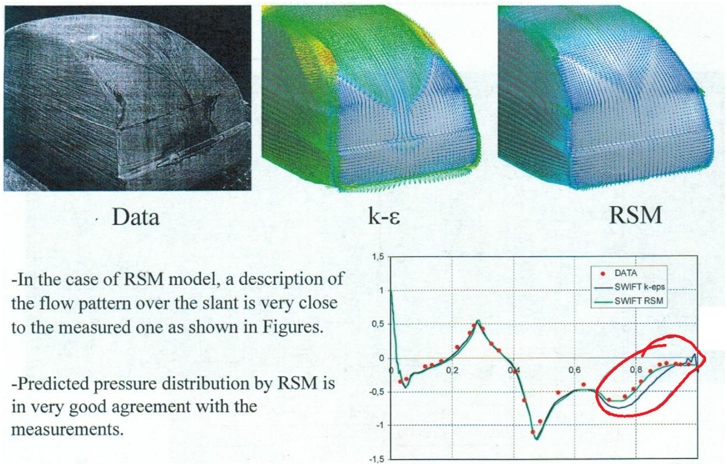


Figure 3.6: Comparison of measured and simulated lift coefficients on a SAE reference-body [156]

There are two commercial CFD codes based on the NS approach dominating the today's market: [17]

- ANSYS Fluent (using at *Magna Steyr*) which uses mainly the $k-\omega$-model
- CD-adapco STAR-CCM+ where most of the users apply the $k-\epsilon$-model

Even if there are much more advanced turbulence modelling approaches existing, 90 per cent of the users apply one of these two models. Figure 3.7 shows a comparison between WT data and simulations using $k-\epsilon$ as well as RSM turbulence models. At first glance, it can be seen, that the RSM solution is much closer to the experimental data. As long as the flow does not separate, both simulations deliver almost the same result, pretty close to the experimental data. As the separation appears at the rear of the car, there is a clear discrepancy between $k-\epsilon$-model and RSM. The predicted pressure distribution by the RSM match very well with the measurements. One disadvantage of RSM is the increased computing time of about four times more compared to a EVM. Anyhow, the main reasons why RSM has not become standard in the industry is its low robustness in combination with poor quality meshes. It is obvious, that a mesh created by hand has better quality than a auto generated mesh. As automatic mesh generators (which obviously cannot reach the mesh quality of a hand made mesh) have become standard over the past ten years, RSM has never been accepted as the standard turbulence model in automotive industry. RSMs have not even five per cent usage in the industry. Some critics say, that with the higher modelling amount (seven equations in RSM compared to two in most EVMs), more modelling errors appear. In any case, it is currently the best theoretical model available. [17]

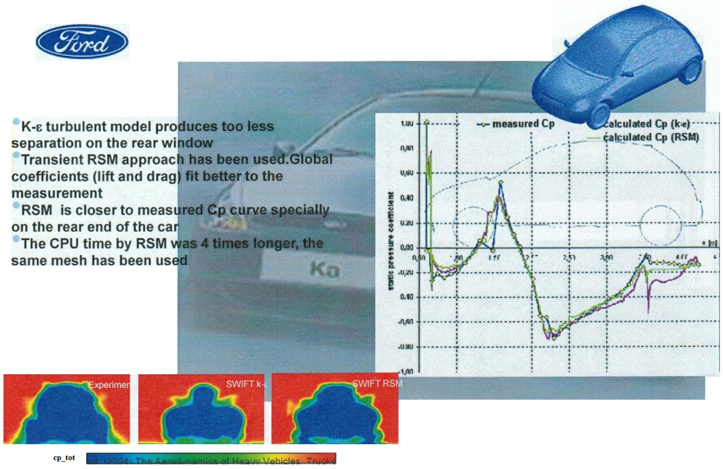


-In the case of RSM model, a description of the flow pattern over the slant is very close to the measured one as shown in Figures.

-Predicted pressure distribution by RSM is in very good agreement with the measurements.

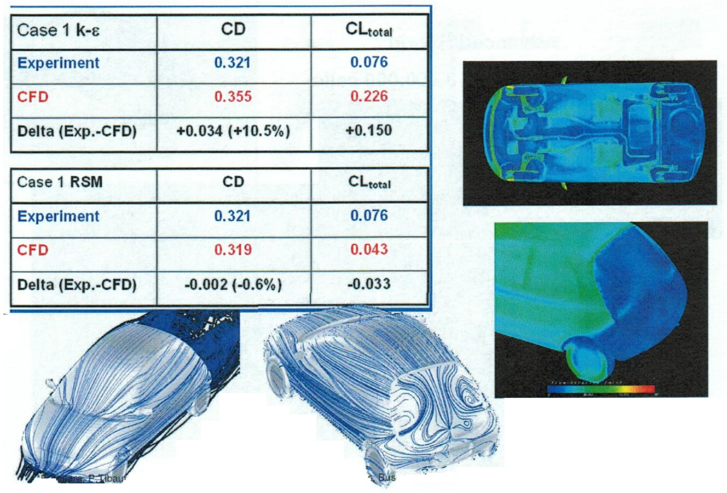
Figure 3.7: Comparison between k-epsilon and RSM on a VW model [16]

Another application example was published by the Ford company in 2004. Figures 3.8 and 3.9 show again different results of a RANS using a $k-\epsilon$-model on the one hand and a RSM on the other hand. At the back of the car, where large separation areas appear, the RSM predicts a flow field which correlates much more with the WT measurements than the flow field predicted by the $k-\epsilon$-model. In these simulations, a hybrid mesh with 3 400 000 cells was applied with both models. The $k-\epsilon$-model produced too less separation on the rear window. [99] All in all, the transient RSM approach delivered clearly more accurate global drag and lift coefficients. This can be seen on the c_P graph as well as in the c_D/c_L table. On the other hand, RSM's CPU time was about four times higher than the $k-\epsilon$. [17] With this results, it is obvious, that the RSM approach is a much more exact solution compared to the mainstream EVM approach.



- K-ε turbulent model produces too less separation on the rear window
- Transient RSM approach has been used.Global coefficients (lift and drag) fit better to the measurement
- RSM is closer to measured Cp curve specially on the rear end of the car
- The CPU time by RSM was 4 times longer, the same mesh has been used

Figure 3.8: Pressure coefficient on a Ford Ka with different turbulence models ($k-\epsilon$ and RSM) [99]



Case 1 k-ε	CD	CL$_{total}$
Experiment	0.321	0.076
CFD	0.355	0.226
Delta (Exp.-CFD)	+0.034 (+10.5%)	+0.150

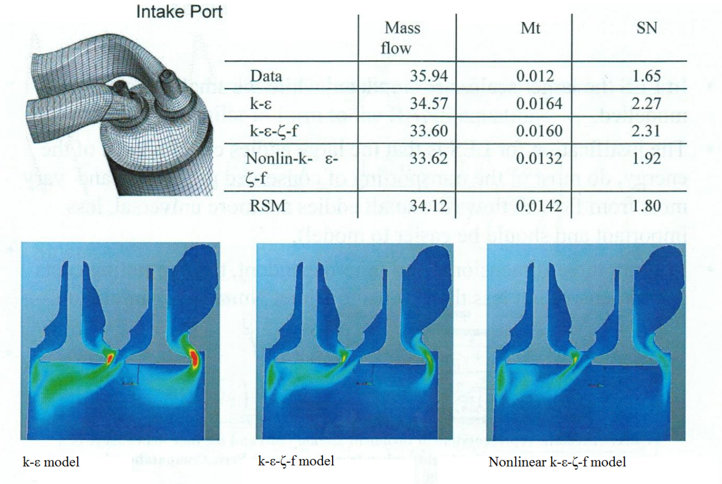
Case 1 RSM	CD	CL$_{total}$
Experiment	0.321	0.076
CFD	0.319	0.043
Delta (Exp.-CFD)	-0.002 (-0.6%)	-0.033

P. Tibaut
Bus

Figure 3.9: Comparison between drag and lift coefficient of a Ford Ka with different turbulence models ($k-\epsilon$ and RSM) [99]

Another turbulence model comparable in accuracy to RSM is the $k-\epsilon-\overline{v^2}-f$-model,

which is one of the most recent approaches. This nonlinear EVM has some advantages over the classical linear EVMs. According to Basara [17], it is the best model available today. Referring to figure 3.10, its results are pretty close to RSM but it is as robust as a standard two-equation model. In commercial codes it is the only EVM with more than two equations (four). Depending on application and turbulent intensity, the $k-\epsilon-\overline{v^2}-f$-model is within ± 15 per cent time demanding compared to a linear EVM. A comparison between the most important turbulence models is shown in figure 3.10. [17]
Another reason, why these advanced models are still not used a lot, is that the development of hybrid RANS-LES methods is much more promising for current as well as future applications. However, the $k-\epsilon-\zeta-f$-model is used within the PANS method which is described in chapter 3.4.7. [17]



	Mass flow	Mt	SN
Data	35.94	0.012	1.65
k-ε	34.57	0.0164	2.27
k-ε-ζ-f	33.60	0.0160	2.31
Nonlin-k- ε-ζ-f	33.62	0.0132	1.92
RSM	34.12	0.0142	1.80

Figure 3.10: Modelling of turbulence and wall treatment of an intake port with different turbulence models [17]

3.4.4 Large Eddy Simulation (LES)

Basics of LES

The time averaged RANS simulation does not differ between high- and low frequent fluctuations. Thereby all time dependent changes comprised by the turbulent spectrum are averaged. No difference is made between small and big scale eddy structures. Considering this, two facts should be kept in mind: [133]

- Big scale turbulent formations are predominantly responsible for momentum as well as energy transport.
- Small scale turbulent formations are predominantly responsible for dissipation of kinetic energy into thermal energy (heat). This small scale structures are almost isotropic.

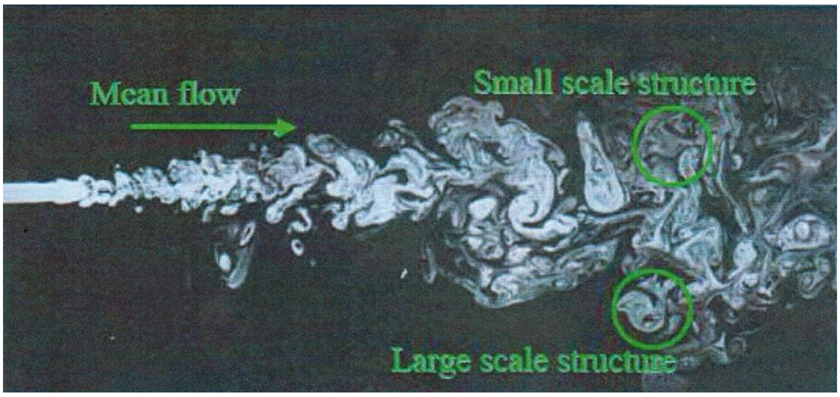


Figure 3.11: Small and large scale turbulent structures [17]

By considering this, it is obvious that averaging the whole spectrum cannot lead to a satisfying result. However, the structure of small scale eddies can be assumed as almost isotropic. In that area the RANS approach would deliver pretty satisfying results. In contrast to RANS, LES filters the NS equations (for example with a Gauß-filter) and splits the flow parameters (velocity, pressure) into low and high frequent fractions (see equation 3.21). The low frequent fractions are computed directly where the small structures (high frequent fractions) are modelled in the so called sub-grid-scale (SGS, see figure 3.12) [17, 133]

$$u_i = \widetilde{u}_i + u_{i,SGS} \qquad \text{and} \qquad p = \tilde{p} + p_{SGS}. \tag{3.21}$$

If the mesh size tends to zero, also the SGS fraction tends to zero[7]. That means, LES converges to the DNS (see chapter 3.4.9) if the mesh size converges to zero. A reference value for the number of mesh cells follows the function $N = Re^{3/2}$. [122] With equations 3.21, the LES momentum equation is [133]

$$\frac{\partial \widetilde{u}_i}{\partial t} + \widetilde{u_j}\frac{\partial \widetilde{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \tilde{p}}{\partial x_i} + \nu\nabla^2\widetilde{u}_i - \frac{\partial \tau_{ij,SGS}}{\partial x_j}. \tag{3.22}$$

Formal, the RANS and the LES equations are the same. Instead of the Reynolds stress tensor τ'_{ij}, the SGS stress tensor $\tau_{ij,SGS}$ appears at the right hand side of the equation. [133]

$$\tau_{ij,SGS} = \widetilde{\widetilde{u}_i\widetilde{u}_j} - \widetilde{u}_i\widetilde{u}_j + \widetilde{\widetilde{u}_i u_{j,SGS}} + \widetilde{u_{i,SGS}\widetilde{u}_j} + \widetilde{u_{i,SGS}u_{j,SGS}} \tag{3.23}$$

The most commonly used model approach for SGS stress tensor was developed by Smagorinsky in 1963 for stabilising bad resolved weather simulation. [70, 133]

[7] This follows from the definition of the filter. As this thesis does not intend to go deep into the methods theory, this is stated without any prove.

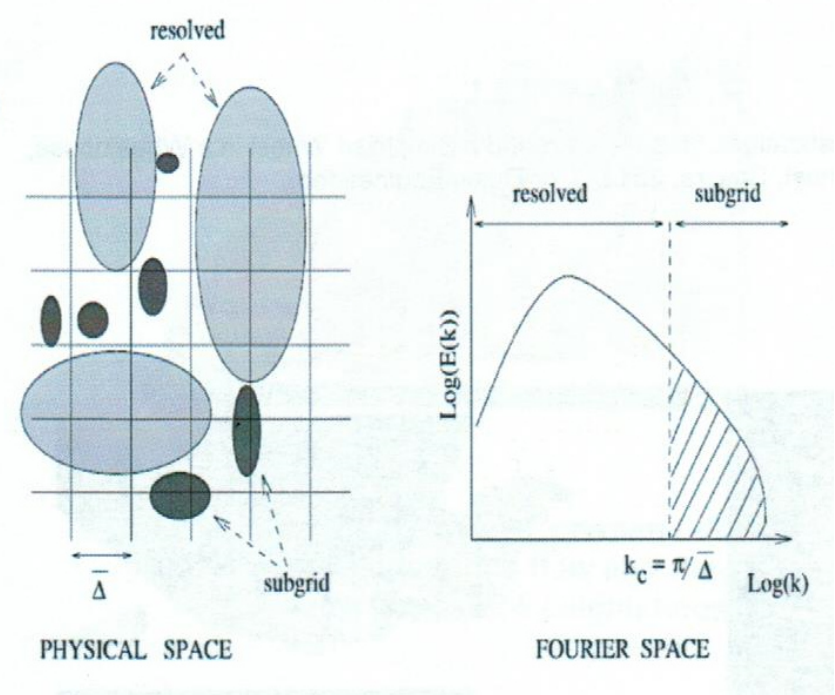


Figure 3.12: Sub-grid visualisation [17]

$$\tau_{ij,SGS} = -2 \cdot C_S \cdot \Delta^2 \cdot \left(2 \cdot \widetilde{S}_{ij} \cdot \widetilde{S}_{ij}\right)^{\frac{1}{2}} \cdot \widetilde{S}_{ij} \tag{3.24}$$

where [133]

$$\widetilde{S}_{ij} = \frac{1}{2} \cdot \left(\frac{\partial \widetilde{u}_i}{\partial x_j} + \frac{\partial \widetilde{u}_j}{\partial x_i}\right) \tag{3.25}$$

is the filtered shear rate tensor and [17]

$$\Delta = min[(V)^{1/3}, \delta] \tag{3.26}$$

the filter width. The filter width is usually taken as the minimum between the cubic root of the cell volume V and the distance to the nearest wall δ. The Smagorinsky constant C_S is given by 0,005...0,05 depending on flow application. [17, 133] As this SGS energy is smaller than the RANS turbulent energy, the model accuracy my be less crucial then in RANS. That is also the reason why the turbulence models in RANS (see chapter 3.18) are much more complicated than the models in LES (Smagorinsky is actually a zero-equation model). [17]

A reference point for the filter quality of a LES is the TKE ratio. The total TKE

$k_{Tot} = k_R + k_U$ is the sum of the resolved TKE $k_R = \frac{1}{2}\hat{u}_i$ and the unresolved TKE $k_U = f(u_{SGS}, ...)$. If [17]

$$\frac{k_U}{k_{Tot}} < 0.2, \tag{3.27}$$

the filter settings are okay (other sources say $\frac{k_U}{k_{Tot}} < 0.3$ [146]). If this is not the case, the (methodical as well as the numerical) error is too large and the result will not be satisfying. This means, that at least 80 per cent of the TKE has to be resolved, in order to achieve a acceptable result using LES. [17]

A massive difference between a LES and a RANS simulation represents the influence of the mesh size. A mesh refinement at the RANS method affects only the numerical error. On the contrary, in LES it affects the numerical as well as the methodical error. The reason for this is, that in RANS influential flow areas are modeled by a turbulence model on the one hand and a wall model on the other hand, where LES reproduce the large eddies and do not use any wall model. With smaller grid scales the percentage of the SGS and therefore model error is reduced. In 1999, Spalart mentioned that a 20% change of the Smagorinsky constant in a fine resoluted flow does not have any noticeable effect. [143]

Sometimes, in literature the expression VLES can be found. This simply means very large eddy simulation where just the largest eddies are filtered and resolved. In general it is better not to use this expression, as PowerFLOW uses a lattice Boltzmann model which is called LBM-VLES, but has in fact nothing to do with LES described in this chapter. [133]

Current Development of LES

In 2000 Philippe Spalart predicted, that a pure LES of an airborne or ground vehicle would use well over 10^{11} grid points and almost 10^7 time steps, which is estimated to be possible approximately in 2045. [146] Most of theses expenses come though the boundary layer which leads to the formulation of the DES method (see chapter 3.4.6).

One-and-a-half decades later this statement is still valid. In 2015, Noelting and Fares stated that it is not expected, that LES will get feasible for industrial applications within the next two decades. [117] The application of pure LES still fails on its computational demand and therefore costs. However, as a long term goal LES still seems to be the most promising of the current CFD technologies. Even if there are some very promising methods in development over the last years. Just to mention the PANS method, addressed in chapter 3.4.7 or LBM, addressed in chapter 3.4.11. The Japanese automotive industry takes big efforts since the 1980s to implement LES in their workflows and make it applicable for industrial use. In 2016, a range of hybrid RANS-LES methods exist, implemented in the various available software. Rapid advances in computer hardware make pure LES more and more feasible for research projects. To realise an economic industrial use, it is required to have [78]

- high order algorithms for complex geometry meshes,

- SGS models applicable to wall bounded flows and
- massively parallel computational implementation.

A problem of the well known traditional numerical schemes (FDM, FEM, FVM) for engineering problems is their high numerical dissipation of turbulent energy. They also do not provide sufficient accuracy for large eddy or direct numerical simulation (DNS). Recent researches dealt with the application of some new numerical methods, like spectral difference (SD) [155], nodal discontinuous Galerkin (NDG) [69] or flux reconstruction (FR) [75]. The requirement on the numerical schemes in the future can be defined as: [78]

- Solution has to be correct.
- Small numerical dissipation to capture unsteady flow features.
- Ability to solve complex geometry with unstructured and therefore computationally more demanding grids.
- Avoidance of wave propagation problems due to numerical fluxes.
- Capability for high resolution for transient and turbulent flows.
- Efficient implementation for code parallelism.

As this thesis is not about code generation, the development of new numerical schemes is not closer examined. Anyway, two of the main current issues in LES are: [78]

- The need of wall models in wall bounded flows. This destructs one of the big advantages of LES. But to be able simulating wall bounded flows at defensible computational costs, it is needed to apply wall models nowadays.
- There is still a lack of proper sub-grid filtering techniques for unstructured meshes. Complex geometry (like a car) is usually discretised with unstructured meshes.

Wall Modelled Large Eddy Simulation (WMLES)

As a pure LES is not applicable for industrial use nowadays, there are several different approaches. Chapters 3.4.5 and 3.4.6 describe hybrid RANS-LES methods, which is currently the most common approach. Somewhere between LES and the hybrid methods is the WMLES. All these methods have one purpose in common, namely to reduce the computational costs. In a pure LES, it has to be ensured that the distance from the wall to the first computational node is $y^+ < 2$. This leads to an extensive number of cells and therefore to massive computational requirements. This is neither feasible for production car development nor for fundamental research in the industry. [17, 133]
The mesh dependency of a pure LES is a function of the Reynolds number: $N \approx Re^{3/2}$. [122] WMLES is (together with ZLES and DES) an alternative to LES to reduce the

Reynolds number dependent grid resolution. When approaching the wall, the eddy structures get smaller and smaller and therefore more and more isotropic. Inside the viscous sublayer the viscosity takes over the dominant role and dampens out almost the entire turbulent eddy structures. At higher Reynolds numbers, the minimum size of the eddies decreases and the viscous sublayer gets smaller and smaller. In order to avoid the resolution of these very small wall-near eddies, the wall-near area is modelled with RANS. In contrast to DES, where it is intended to model the whole boundary layer, WMLES models just the viscous sublayer. For a closer insight of this model, refer to the original papers. [103, 136]
Closing this chapter, it should be noted that LES is seen, in contrast to DNS, as a medium or long term CFD method. Currently just used in research projects, it is possible that LES is taking over the roll from RANS as the standard CFD method in a few decades. For institutions or companies which have large computational resources, the DES method is already a very powerful tool. [78]

3.4.5 Zonal Large Eddy Simulation (ZLES)

Basics of ZLES

There is a range of various ZLES methods existing. Some just differ in name within the various software packages. But they all have the same purpose, namely to split the flow domain into RANS and LES portions ahead of the simulation. [103] The following list itemises some methods:

- **Embedded Large Eddy Simulation (ELES)**: A formulation in ANSYS Fluent, which allows the combination of most RANS models with all non-dynamic LES models, like for example the (linear) Smagorinsky model. [39, 103] The switch at the interface between the two turbulence models is achieved by the vortex method. [98] A synthetic turbulence is created at the interface to convert the turbulence from RANS to LES. [121]

- **Zonal Forced Large Eddy Simulation (ZFLES)**: This is a similar approach as ELES for ANSYS CFX. However, this method uses WMLES instead of LES. At the interface between the two models, synthetic turbulence is generated. [103]

- **Scale-Adaptive Simulation (SAS)**: SAS represents an improved URANS formulation. In unstable flow conditions, the turbulent spectrum is resolved. The method bases on the introduction of the von Karman length scale into the turbulence scale equation. In unstable flow regions this method offers a LES behaviour, in stable flow regions a RANS behaviour. [103, 121] SAS is implemented in ANSYS Fluent and OpenFOAM. [7, 67]

For ELES and ZFLES it is important to have some previous knowledge of the flow quality. In challenging flow areas where LES is active, the grid must be adapted which means a higher resolution. So the pre-processing is much more challenging than with a pure RANS or LES. ELES and ZFLES are just feasible in locally or marginally unstable flows. In globally unstable flows, SAS (or DES) should be applied. [107]

Scale-Adaptive Simulation (SAS)

As described above, SAS represents an improved URANS formulation where the flow field has a LES behaviour in unstable flow conditions and a RANS behaviour in stable flow conditions.[8] An URANS model does not allow the formation of a turbulent cascade. URANS can only predict single vortices shed from a body. [121] This behaviour follows due to the RANS formulation of time averaging, which eliminates all turbulent content from the flow field. [103]

SAS is based on the introduction of the von Karman length scale L_{vK} into the scale-defining equation (for example $\epsilon-$, $\omega-$ or L_t-equation). With $\Phi = \sqrt{k}L_t$ and constants σ_Φ, ζ_1, ζ_2, ζ_3, the scale-defining equation is [103]

$$\frac{\partial \Phi}{\partial t} + \frac{\partial(\overline{u}_j \Phi)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\frac{\nu_t}{\sigma_\Phi}\frac{\partial k}{\partial x_j}\right) + \frac{\Phi}{k\rho}P_k\left[\zeta_1 - \zeta_2\left(\frac{L_t}{L_{vK}}\right)^2\right] - \zeta_3 k \tag{3.28}$$

where the rate of production of k is [17]

$$P_k = -\overline{u_i' u_j'}\frac{\partial \overline{u}_j}{\partial x_j}. \tag{3.29}$$

This again is based on the theory of Rotta [126], using an exact definition of the turbulent length scale. [106] In order to determine the turbulent length scale (shear layer thickness), the diffusion term is needed. It will then return a length scale $l_0 \sim \sqrt{k}/\omega \sim \delta$ where δ is the turbulent length scale L_t. If the von Karman length scale [103]

$$L_{vK} = \kappa\left|\frac{\overline{u}'}{\overline{u}''}\right| \tag{3.30}$$

where [103]

$$\overline{u}' = S = \sqrt{2S_{ij}S_{ij}} \qquad \text{and} \qquad \overline{u}'' = \sqrt{\frac{\partial^2 \overline{u}_i}{\partial x_j \partial x_j}\frac{\partial^2 \overline{u}_i}{\partial x_k \partial x_k}}, \tag{3.31}$$

with the shear rate tensor [103]

$$S_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u}_i}{\partial x_j}\frac{\partial \overline{u}_j}{\partial x_i}\right), \tag{3.32}$$

is proportional to δ, a steady-state RANS solution is computed. Otherwise SAS is active. In the SAS mode, the exact transport equation for the integral length scale (for example modified ϵ-equation of SST-$k-\omega$) is resolved. [105, 121]

[8] This model is used for the unsteady simulations in the practical part of this thesis.

If no or little instability (separation) is present in the flow, SAS would switch to (U)RANS. Only flow fields with large separation and mixing zones would provide unsteady results. If the mesh is too coarse or Δt is too large, SAS would switch to steady-state RANS. According to Menter, SAS is currently the least problematic hybrid RANS-LES model available. This is due to its non explicit grid dependency in RANS portions. [121] Compared to DES models which are very sensitive to grid resolution (see chapter 3.4.6), SAS is a very robust hybrid RANS-LES model. According to ANSYS, SAS is currently the most reliable and robust hybrid RANS-LES method.

Current Development of ZLES

The flow field around an aircraft is extensively big. Challenging areas, especially around the turbine engines, are relatively small. So it does not make any sense to apply a global turbulence resolving model (LES, DES) to the domain. Refer to figure 3.13, the focus area around the turbine engines are only a little fraction of the entire domain. This is a perfect case to apply a model, which resolves the turbulence only in local regions (ZLES, SAS).

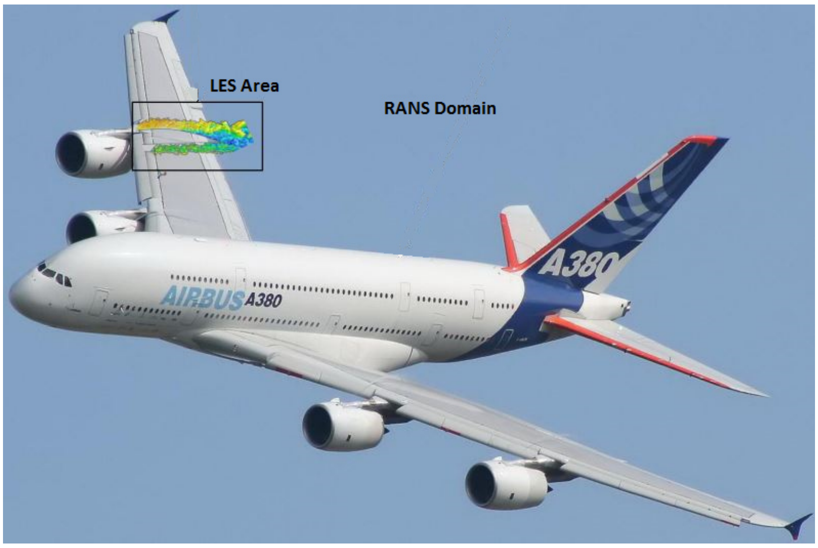


Figure 3.13: Example of an applied ZLES on an Airbus A380 [107]

Another application in aerospace is shown in figure 3.14. The flow areas within this gas turbine engine are separated. The flow in the compressor is calculated by a RANS model and the flow in the combustor by LES. When looking only at the compressor flow, it can be seen, that the flow is very regular and turbulent fluctuations are absent. In the combustor, in contrast, where the turbulent mixing of fuel and air is highly important, a LES model was applied. When the flow approaches the combustor (where the flow gets highly turbulent during the combustion process) and therefore the LES area, it gets disordered immediately. [100] This is another case, where the strengths of ZLES models can be applied.

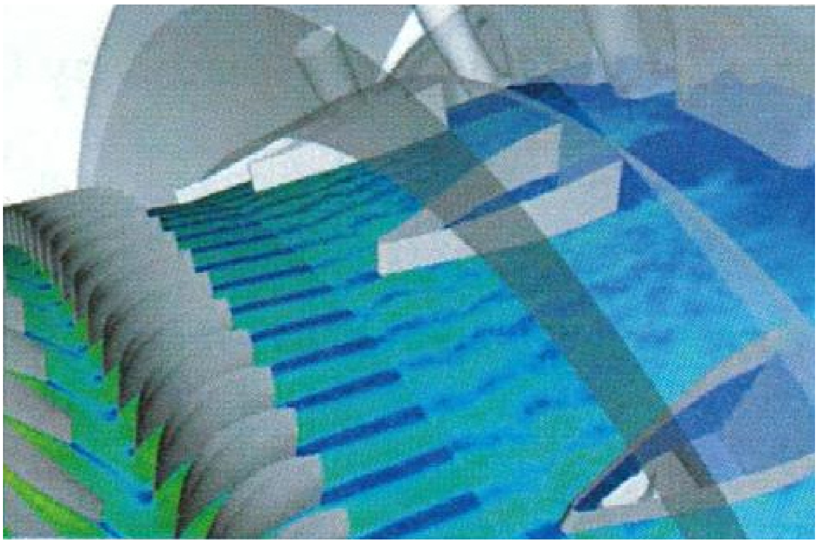


Figure 3.14: ZLES on a compressor and combustor of an aerospace gas turbine engine [17, 100]

3.4.6 Detached Eddy Simulation (DES)

Basics of DES

At high Reynolds numbers and massively separated flows, which is just how it is at a car, a pure LES is computationally way too expensive. A RANS simulation, however, is not capable of capturing the turbulence correctly. DES combines the LES and (U)RANS method to save (computational) costs. In wall-near regions the RANS approach is applied, where in the outer flow and the wake LES is applied (see figure 3.15). [142] As long as the turbulent length scales in the boundary layer are considerably lower (some magnitudes) than the scales in the outer flow, the DES approach is a good approximation. This is the case in automotive aerodynamics and therefore it is a good (future) option for this industry (as well as aerospace). In wall-near regions, the flow is pretty much universal in many applications. Mostly a one-equation turbulence model according to SA is used. The transition from LES to RANS happens at a specified wall distance d, which is defined as [133]

$$d = min\{y_{W,SA}; \Delta \cdot c_{DES}\} \tag{3.33}$$

with the empirical constant $c_{DES}(= 0.65)$, the length Δ which is the equivalent cell diagonal and the cell-wall-distance y_W. Is the cell-wall-distance large compared to the cell volume, LES is applied, otherwise RANS. [133] It should be kept in mind, that the RANS method is capable of predicting flow separation. But as it is an averaged method, in large separation areas its accuracy gets worse. Especially if engineering problems like vibrations or aeroacoustics are considered, it is clear, that an unsteady approach is needed. [146] The mathematics are exactly the same as in RANS and LES. By viewing figure 3.15 the two areas are visible very well. The green area denotes the wall-near RANS approach, whereas the red areas indicates the LES regions. [101]

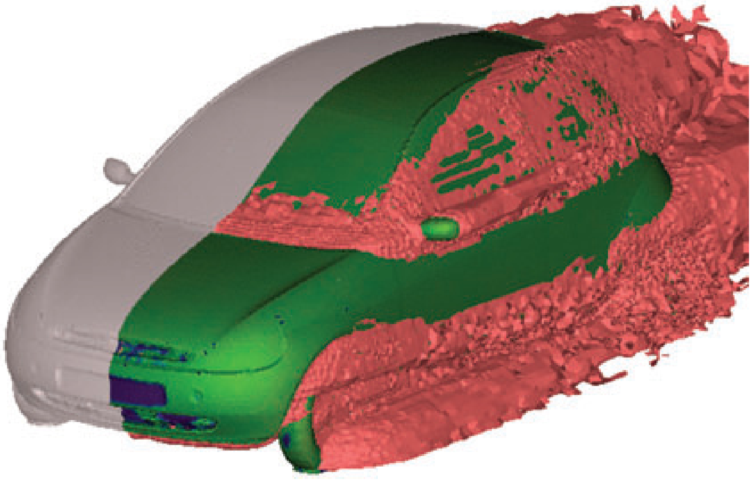


Figure 3.15: Acoustic-source isosurface around a Ford Ka automobile [101]

Figure 3.16 shows six different simulations of a flow past a circular cylinder. A massive difference between RANS, URANS and DES can be monitored. (1) shows a common steady RANS simulation with a unstable solution. Compared with (2) where a 2d URANS simulation is shown. Obviously the 2d simulation suppresses the three-dimensionality. (3) shows a 3d URANS. The three-dimensionality is much coarser than at the following DES simulations and will not get finer using a finer grid as it does in DES (compare chapter 3.4.4). (4) to (6) show three different DES', using different meshes and its response to finer grid sizes. [135, 146]

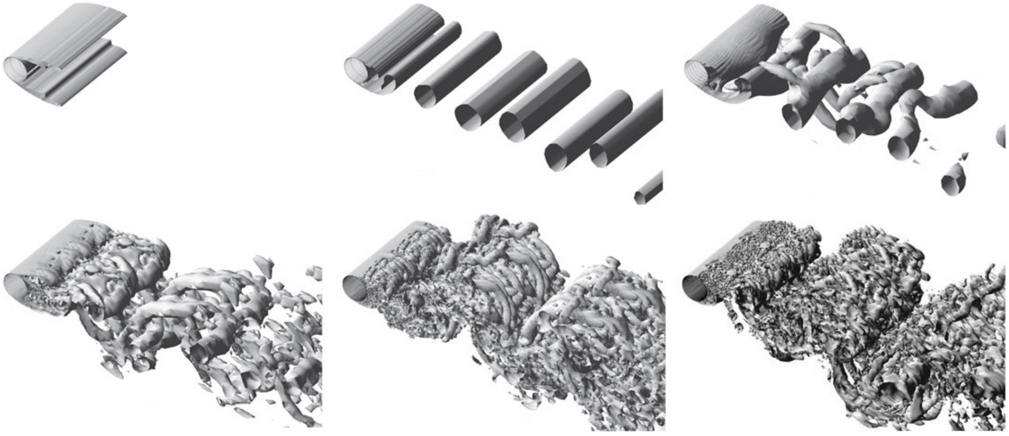


Figure 3.16: Top left to bottom right: RANS, 2d URANS, 3d URANS and DES with three different mesh resolutions [135, 146]

A problem of the various DES methods is a strong deterioration of the flow field prediction when the RANS-LES-switching border distance to the wall is larger than $y^+ = 30$. [17] Some further weaknesses of the DES methods are (for a closer insight to this refer to the original publication [146]):

- Conceptual issues (interface between RANS and LES)
- Modeled-stress depletion
- Grid-induced separation
- Logarithmic layer mismatch
- Slow LES development in mixing layers

It is a large development community which tries to resolve these issues of DES. The following chapter describes some approaches which are trying to resolve some of these problems.

Current Development of DES

DES was first proposed in 1997 and was based on an idea of Spalart in 1992. The first use of DES was in 1999. [146] Some recent proposals in the wide group of DES are:

- **Zonal Detached Eddy Simulation (ZDES):** The user can explicitly mark different regions where either RANS or DES is applied. This is probably the strongest type of the DES methods. The motivation for this was to get rid of some of the DES weaknesses described in the chapter above (modeled-stress depletion and grid-induced separation). [43] Each region can be treated separately concerning its importance and influence to the flow field.
- **Delayed Detached Eddy Simulation (DDES):** This model detects the boundary layer and sets the RANS-LES transition by itself. If the boundary layer is thicker than the standard DES limiter d (see formula 3.33), this model prolongs the full RANS mode. The detection device therefore is based on eddy viscosity and for this reason dependent on the solution. Like ZDES, DDES' purpose is to get rid of the DES' weaknesses. This method is likely to be the latest standard of the DES methods. [146]
- **Improved Delayed Detached Eddy Simulation (IDDES):** This approach is maybe the most ambitious one in this family. [136] Its aim is to resolve log-layer mismatch in addition to modelled stress depletion. One basis of the method is a new definition of Δ which includes the wall distance and not only the local characteristics of the grid. For more exact information, refer to the original papers. [146]

In 2008, Gretschner mentioned that: *"...DES is still in its infancy and undergoes continuing improvements."* [57]

In this regard it should be mentioned, that DES codes need some equalities that are absent in RANS codes and others that are absent in most LES codes. So there is still a lot of development in the DES area. Especially the interface between the RANS and LES region takes a lot of developers' attention. The feedback from RANS into the LES region as well as the influence of the LES unsteadiness into the RANS region is object of research and development for new methods. In ANSYS Fluent the latest model is SBES (see chapter 4.2.3). This model blends the interface between the RANS and LES region using the local shear stresses. [146]

An unfortunate trend is, that DES is moving away from its simple approach, assembled by Spalart in 1997 (DES97). This is in terms of the equations and the non uniqueness of the solution (DDES, IDDES), as well as in terms of the user decision load and the need to define regions (ZDES). [146]

DES' principal weakness is its response to ambiguous grids, in which the wall-parallel grid spacing is of the order of the boundary-layer thickness. In some situations, DES on a given grid is then less accurate than RANS on the same grid or DES on a coarser grid. [146] However, despite various problems, DES and its derivatives are by far the most used unsteady CFD simulation tools (aerospace industry to mention before all others). [17]

Closing this chapter, it should be kept in mind that not each mesh is suitable to DES. In some cases, RANS would even deliver a better result than DES. So when using DES, a lot of attention should be put in the process of mesh creation. [146]

3.4.7 Partially-Averaged Navier-Stokes (PANS)

Basics of PANS

PANS is a recent proposal and already implemented in AVL FIRE and OpenFOAM (from v3.0) code. [17, 120] It was originally proposed by Girimaji in 2003. [56] As the name suggests, this method groups the velocity spectrum in resolved (DNS) and modelled (RANS) portions. So the method lies somewhere between DNS (see chapter 3.4.9) and the URANS method (see chapter 3.4.3). PANS represents a new modelling approach, which differs clearly from the well known RANS, LES and their hybrid methods. [13, 55, 77]

The PANS method delivers a smooth transition between URANS and DNS. With the filter parameter f_k, the user can define the ratio between resolved and modelled turbulence. [13] In contrast to RANS, where the flow field is separated into time averaged and fluctuating terms, the PANS method decomposes the flow field into resolved (DNS) and unresolved or residual (RANS) terms. So the PANS method can be much more accurate than URANS as key fluctuations are considered in their fully resolved form. The difference to LES (see chapter 3.4.4) is, that PANS resolves a significantly lower number of scales (depends on the user specified parameter f_k). This comes due to the fact, that a PANS filter bases on kinetic energy content of the flow field rather than

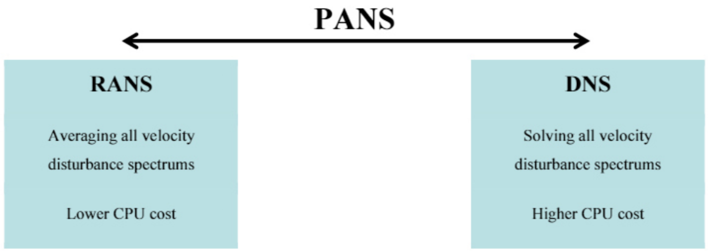

PANS
RANS
Averaging all velocity disturbance spectrums
Lower CPU cost
DNS
Solving all velocity disturbance spectrums
Higher CPU cost

Figure 3.17: Placement of the PANS method in the range between RANS and DNS [13]

cutting off wave numbers like an LES filter. The method is based, same as RANS, the equation of continuity [56, 77]

$$\frac{\partial \tilde{u}_i}{\partial x_i} = 0 \tag{3.34}$$

and the momentum equations which are written in the context of the PANS method as follows: [41]

$$\frac{\partial \tilde{u}_i}{\partial t} + \frac{\partial \tilde{u}_i \tilde{u}_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \tilde{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\nu \frac{\partial \tilde{u}_i}{\partial x_j} - \tilde{\tau}^t_{ij}\right) \tag{3.35}$$

The stress tensor $\tilde{\tau}^t_{ij}$ represents the fully modelled turbulent Reynolds stress tensor ($\overline{u_i u_j}$) (equation 3.36) on the one hand in terms of the RANS method (where $\tilde{u}_i \equiv \overline{u}_i$) and on the other hand, the stress tensor of the unresolved turbulence in terms of the PANS method (equation 3.37). [56, 77]

$$\tilde{\tau}^t_{ij} \equiv \overline{u_i u_j} = -\nu_t \left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i}\right) + \frac{2}{3} k \delta_{ij} \tag{3.36}$$

$$\tilde{\tau}^t_{ij} \equiv \left(\tilde{\tau}^t_{ij}\right)_u = -\nu_u \left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\right) + \frac{2}{3} k_u \delta_{ij} \tag{3.37}$$

The turbulence eddy viscosities ν_t and ν_u are defined via the $k - \epsilon - \zeta - f$-turbulence model (in AVL FIRE) or a similar approach. Thereby the the turbulence viscosity ν_t, the TKE k with its dissipation rate ϵ and the normalised wall-normal velocity $\zeta(\zeta = \frac{\overline{v^2}}{k})$ refer to the fully modelled turbulence in terms of the RANS method. In the entire flow field, just one turbulence model is active, which models the unresolved turbulent structures within the URANS method. This happens predominately in wall-near regions and as a small eddy model in the flow core. With this a seamless transition from URANS to DNS within the energy spectrum is achieved by partial integration of the spectrum. With the introduction of the filter parameters [77]

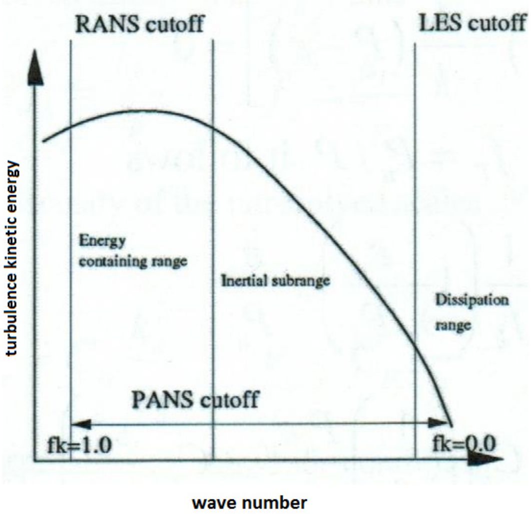


Figure 3.18: Energy spectrum showing the cut off wave numbers for PANS [17]

$$f_k = \frac{k_U}{k_{tot}} = \frac{1}{\sqrt{C_\mu^\zeta}} \left(\frac{\Delta}{\Lambda}\right)^{2/3} \tag{3.38}$$

and [77]

$$f_\epsilon = \frac{\epsilon_U}{\epsilon_{tot}}, \tag{3.39}$$

the control parameters of the PANS method are defined (refer to figure 3.18). Within these filters, the mesh size Δ is $\Delta = (\Delta_x \Delta_y \Delta_z)^{1/3}$ and the turbulent length scale Λ is $\Lambda = \frac{k^{3/2}}{\epsilon}$. Constant C_μ^ζ is defined via the $k - \epsilon - \zeta - f$-turbulence model (see chapter 3.18). f_k is the fraction of the unresolved TKE and f_ϵ is the fraction of the unresolved rate of dissipation of TKE k. The asymptotic behaviour of the PANS method secures a seamless transition from URANS to DNS with decreasing value of f_k. Within the numerical procedure, the lowest values of f_k are adapted to the current mesh size. Hence f_k is a dynamic parameter, changing at each mesh cell. This results in the beneficial property, that the PANS method delivers good unsteady results over the whole range of mesh resolution. LES, in contrast, is only applicable in very fine meshes. As shown in figure 3.18, if the filter parameter $f_k = 0$, the PANS method operates like a pure URANS simulation. If $f_k = 1$, the other extreme, PANS works as a pure DNS. [15, 77] For a closer insight into the theoretics of the PANS method, refer to [41] and [56].

Current Development of PANS

In previous investigations, the PANS method has shown promising results, but there are still some computational and physical issues that needs to be resolved. The PANS method was first proposed in 2003 by S. S. Girimaji et al.. [56] In 2016 it is implemented in commercial code AVL FIRE [17, 77] and in free software OpenFOAM (since 2013). [17, 29, 120] Usually the $k - \epsilon - \zeta - f$-turbulence model is applied (AVL FIRE). [15] However, PANS may become a bridging method together with the hybrid RANS-LES approaches to overcome the gap till pure LES is industrially applicable.

3.4.8 Partially Integrated Transport Model (PITM)

Basics of PITM

The PITM is the most recent proposal for a new CFD method. Introduced in 2005 by Schiestel and Chaouat, it has the same form like the PANS method but is less robust. It offers a seamless transition from URANS (fully modelled turbulence) to DNS (fully resolved turbulence). [17, 131] The main ingredient is a new dissipation rate equation for the sub-grid region which allows to convert almost any usual RANS transport model into a subfilter scale model. [30] The method allows to perform unsteady simulation on relatively coarse grids when the spectral cutoff is located within or before the inertial zone. [30, 32, 131] This model includes transport equations of all sub-grid stresses with additional dissipation rate equation. LES resolves about 80 per cent of the turbulent spectrum, PITM as well as PANS are able to resolve a considerable lesser amount of the turbulent spectrum. In complex flows, that is a remarkable advantage, as the user can define the rate of accuracy, and therefore the required computing power and time, by adapting the filter parameters. Another advantage, compared to LES, is the applicability on coarse grids, because the sub-grid turbulent scales are resolved via a turbulence model (mostly a $k - \epsilon$-model or derivative like $k - \epsilon - \overline{v^2} - f$). Due to the use of transport equations for all of the sub-grid stress components, it allows to take into account for more precise turbulent processes of production, transfer, pressure distribution and dissipation. The concept of turbulence viscosity therefore is no longer necessary. The sub-grid model used within the PITM, which accounts for history and non-local effects of the turbulence interactions, describes very accurately the anisotropy of the turbulence field. [32, 131] Like the PANS method, the PITM offers a continuous transition between URANS and DNS (see chapter 3.4.9) within the whole domain. In contrast, the hybrid RANS-LES methods are zonal methods, where the domain is split into regions where the different models are applied. [131] For closer insights into the theoretics, refer to [41].

Current Development of PITM

In contrast to the PANS model, the PITM has not been implemented yet in any commercial code (not even in OpenFOAM). There is some development going on at the Off Natl Etud Rech Aerosp, Computat Fuid Dynam Department, Chattillon, France. [1] For more information on this research, these four papers are available: [31, 32, 33, 34]

3.4.9 Direct Numerical Simulation (DNS)

Basics of DNS

The DNS approach is the most accurate method to resolve a fluid flow problem. The NS equations are solved without any simplification over the entire flow field. Therefore neither a turbulence nor a wall model is needed. The aim is to capture the whole turbulent spectrum down to the Kolmogorov scales (see chapter 3.3.2). For this application the mesh needs to be very fine, especially in near-wall regions. Therefore the number of cells follows the function $N = Re^{9/4}$. [122, 133] Also the time resolution has to be very high to represent the unsteady behaviour of the fluid flow representative. This requires a massive amount of computational resources, which is simply not available nowadays. Especially for turbulent flows with high Reynolds numbers, which are most common in engineering problems, DNS will not be practicable within the next few decades. [133]

Current Development of DNS

In 2000, Spalart mentioned, that DNS will not be applicable in industry before 2080. [143] Anyway, for the sake of completeness, a recent development will be presented here. Especially the Japanese automotive industry took some efforts since the end of the 1980s to make DNS feasible for automotive engineering. [133]
In 1988, a group of scientists under Tsuboi [152] used DNS to reproduce the measuring results done by Morel in 1978 [113] on a circular cylinder longitudinal in flow. The influence of the base angle was reproduced quite well by the calculation. Figure 3.19 shows the drag rises with increasing base angle ϕ till a critical value at about 44°, where drag decreases abrupt. The associated highly detailed flow field is shown in figure 3.20. No other method was able to reproduce such detailed and well agreeing results. However, the pressure gradient at the base shows considerable deviations compared to the measured results.

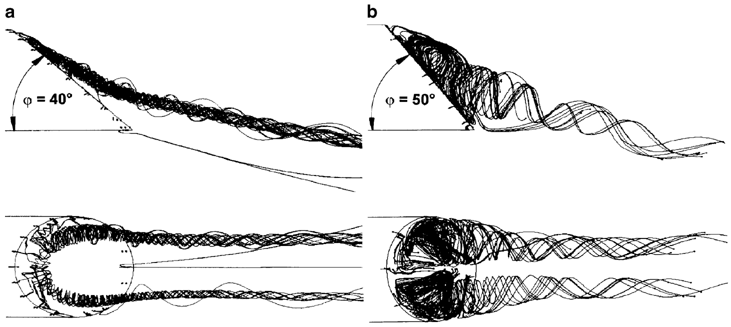


Figure 3.20: Flow field at the rear of the cylinder longitudinal in flow.[152] a) Horseshoe-eddy at subcritical base angle ϕ, b) widened horseshoe-eddy at overcritical base angle ϕ

There were some more developments on DNS going on, mainly in Japan. But in fact, we will not see DNS coming into the industrial applications within the next at least

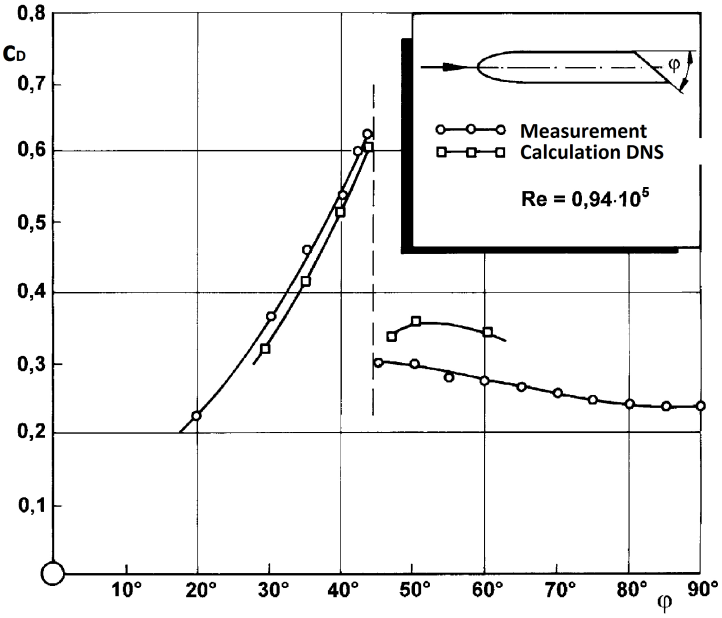


Figure 3.19: Drag coefficient of a cylinder longitudinal in flow, angled at the base over base angle [113, 152]

six or seven decades (in contrast to LES). For fundamental analysis at low Reynolds numbers, this method can serve well. This is for example in the development of new turbulence models. A further step to DNS would be DS (direct simulation) where the compressibility is also considered. Figure 3.21 shows the computational demand of the different methods.

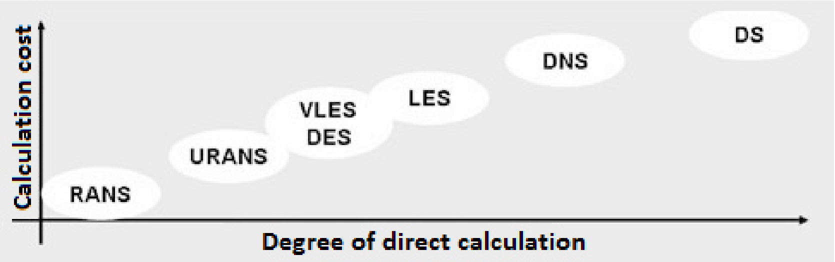


Figure 3.21: Level of direct calculation of the different scales with the different methods over computational effort [19]

In 2015, engineer and mathematician Anthony Jameson said: *"Eventually DNS may become feasible for high Reynolds number flows. Hopefully with a smaller power requirement than a wind tunnel."* [78]

3.4.10 Lattice Gas Automata (LGA)

The first lattice method, called the LGA, was proposed in 1973 (Hardy, Pomeau and de Pazzis). [63] With the upcoming of LBM (see chapter 3.4.11), the interest in this type of simulation tool leveled off. [151] Today there is no development on this method anymore. So as it has no importance nowadays, this method is not further discussed within this thesis.

3.4.11 Lattice Boltzmann Method (LBM)

Basics of LBM

In contrast to the various NS based methods, lattice Boltzmann bases on a different physical behaviour. The NS based methods resolve (or rather approximate) the NS equations which describe the macroscopic properties of a fluid. In contrast, LBM is based on a simpler and more general physic formulation. [117] The motivation is to simulate the physics in the microscopic (atomic, molecular) level, where the physics are much simpler. As it is computationally too expensive to simulate on the atomic level (one mm^3 of a gas contains approximately $5 \cdot 10^{16}$ atoms), [72] a distribution function $f(\vec{x}, \vec{c}, t)$ is applied and implemented into the Boltzmann equation. Therefore, LBM works on the mesoscopic level (statistics). The different physic representations of fluids can be seen in figure 3.22. [117]
So, instead of tracking each molecule, LBM uses the following particle density distribution function [117]

$$f(\vec{x}, \vec{c}, t), \tag{3.40}$$

which describes the number of particles at a given time t and position $\vec{x}$ with the velocity $\vec{c}$. Additionally, the density momentum and energy are given by: [117]

- Density

$$\rho(\vec{x}, t) = \int f(\vec{x}, \vec{c}, t)\, d\vec{c} \tag{3.41}$$

- Momentum

$$\rho(\vec{x}, t)u(\vec{x}, t) = \int f(\vec{x}, \vec{c}, t)\vec{c}\, d\vec{c} \tag{3.42}$$

- Energy

$$E(\vec{x}, t) = \int f(\vec{x}, \vec{c}, t)(\vec{c} - \vec{v})^2 d\vec{c} \tag{3.43}$$

By applying this into the following nonequilibrium (derivative of the Boltzmann equation) [117]

$$\frac{d}{dt} f(\vec{x}, \vec{c}, t) = \frac{\partial}{\partial t} f(\vec{x}, \vec{c}, t) + \vec{c}\nabla f(\vec{x}, \vec{c}, t) = C(\vec{x}, \vec{c}, t), \tag{3.44}$$

this leads to the Lattice-Boltzmann equation [117]

$$f_i(\vec{x} + \vec{c_i}\Delta t, t + \Delta t) = f_i(\vec{x}, t) + C_i(\vec{x}, t) \tag{3.45}$$

where

$$i = 1, 2, 3.$$

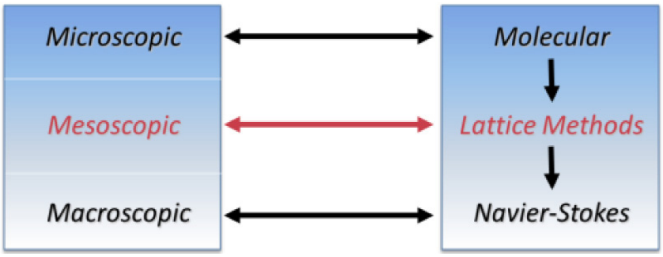
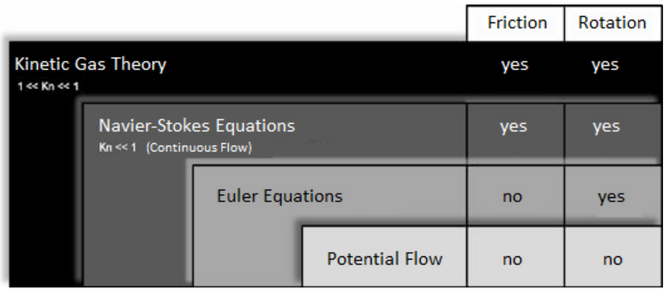


Figure 3.22: Physics representations of fluids [117]

Figure 3.23 describes the hierarchy of gas flows. The NS equations and therefore all methods based on them, are only valid for continuous flows. The kinetic gas theory, where LBM is based on, is a more general approach and therefore valid universally. The Knudsen number (see figure 3.23) is defined as [133]

$$Kn = \frac{\lambda}{L_0} \tag{3.46}$$

where λ describes the mean free path and L the representative physical length scale. If $Kn << 1$ it is a free molecular flow (kinetic gas theory), if $Kn >> 1$ it is a continuous flow (NS). [72]
Another essential difference between these two methods (NS and lattice based) is the type of discretisation. In NS based techniques, the discretisation is known under the collective term "mesh". A mesh can be unstructured and consisting from different element types.



	Friction	Rotation
Kinetic Gas Theory 1 << Kn << 1	yes	yes
Navier-Stokes Equations Kn << 1 (Continuous Flow)	yes	yes
Euler Equations	no	yes
Potential Flow	no	no

Figure 3.23: Classification of gas flows [133]

The lattice of LBM is always regular and mostly Cartesian and is not surface adapted. The models of LBM are identified as D_iQ_j, where D denotes the dimension (2d or 3d) and Q the number of the particle velocities. The fluid can only move in the direction of these arrows. The simulation does a probability calculation for each arrow, how big the probability is, that the fluid moves in this direction. Figure 3.24 shows a typical D_3Q_{19} LBM model. [72]

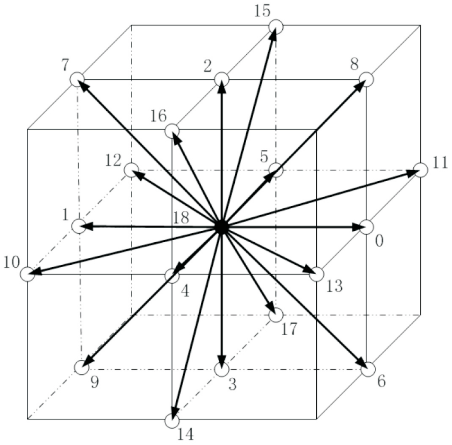


Figure 3.24: Illustration of the three dimensional D3Q19 model [86]

The discretisation process of a LBM is slightly different compared to a NS based method. At first the surface is wrapped with triangle elements ("facets") similar to the shell mesh of a NS based method. Then the domain is discretised with the (Cartesian) lattice. The facets on the volume surface are cut with the volume elements ("voxels") and so-called "surfels" are created (refer to figure 3.25). So the model surface is defined by the surfels rather than by the lattice. [133]
In the course of the simulation, fluid particles collide with the surfels and are reflected. Two extremes are possible within this process of momentum exchange. Figure 3.26 illustrates these two extremes. The specular reflection works like a mirror and defines the "slip condition". The bounce back reflection, in contrast, inverts the velocity direction and defines the "no slip condition" and therefore friction. [133]

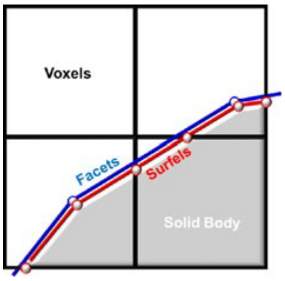


Figure 3.25: Principle of the surface discretisation [133]

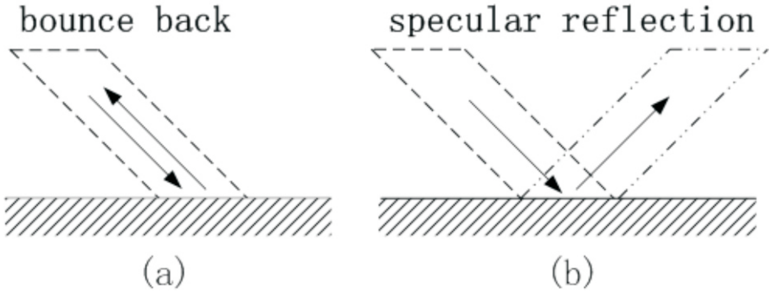


Figure 3.26: Particle reflection on walls [86]

The temporal discretisation is defined by LBM itself. The physical time step per calculation time step is given by [133]

$$\Delta t = \nu_{lattice} \frac{\Delta x}{v_{ref}} = c_{lattice} \cdot Ma \frac{\Delta x}{v_{\infty}} \left[\frac{voxel}{TS} \cdot \frac{m/voxel}{m/s} = \frac{s}{TS} \right] . \tag{3.47}$$

As the sonic speed $c_{lattice}$ is defined by the particle model, the voxel size Δx and the approach velocity v_{∞} fix the magnitude of the time step. This means, the user has only the lattice resolution as a parameter to adapt the time step and therefore the computing time. To overcome this drawback, PowerFLOW offers the possibility to simulate with a Mach number which differs from the approach velocity. This is exclusively to offer an additional parameter to adapt the computing time. This error is accepted, as the influence of the compressibility is very little in a certain range of speed (up to Mach 0.4). [133]

In contrast to a NS method, a LBM simulation does not have to solve PDEs iteratively to reach a residuum. So no convergence criterion is present. Sooner or later, a more or less strong unsteady flow field will result from the simulation. Therefore it is difficult to define a simulation time till the desired result is calculated. Typical simulation time vary from one to two seconds. [133]

LBM can be combined with the various turbulence models known from the NS based methods. In the current application of PowerFLOW, a modified $k - \epsilon$ two-equation model, based on the origin Re-Normalisation Group (RNG) formulation, is implemented to describe the sub-grid turbulence contribution. [117, 133]

The main advantages LBM methods offer are [96]

- a very high temporal resolution,
- an efficient unsteady simulation algorithm and
- a low dissipation and dispersion of numerical scheme.

Following from these advantages, the computational efficiency of LBM is one magnitude order or higher than the classical CFD solutions based on the NS equations of comparable

quality. [95] The main disadvantage of LBM is its limitation to incompressible flows and therefore relatively low speeds of less than Mach 0.4. [117]
As already mentioned in chapter 3.4.4, PowerFLOW offers a LBM, named LBM-VLES. This method can be compared to URANS. In this LBM-VLES method, wall models are applied like in the RANS method to limit calculation time. [133]
As LBM is less dissipative as for example a RANS simulation, it is a numerically less stable method. To overcome this drawback, PowerFLOW uses a finite difference method (FDM) to solve the PDEs. FDM secures the stability of the solution.

Current Development of LBM

Using LBM in numerical flow simulation is a relatively new development. Where the theory of this method was born in the late 1980s, [35, 36, 37] the evolution of LBM codes during the past ten years made it feasible to practical applications, like aerodynamics, aeroacoustics and thermal engineering. [46, 92]
The most cited commercial LBM code in publications is implemented in the PowerFLOW software. [133] It is often used in automotive industry. Others like market leaders ANSYS Fluent and CD-adapco STAR-CCM+ uses the NS approach. Where in the mid 2000s steady-state simulations were sufficient, ten years later more and more industries tried to establish transient and unsteady simulations. This is probably a big advantage for software based on LBM, because they are computationally much more effective in unsteady flows. [117, 133]
Another potential area for LBM is the design of high lift wings. Especially the assessment of $c_{L,max}$ typically still relies heavily on WT testing. [48] A potential issue in this term and in terms of subtle separation from smooth surfaces causing stall, is the reliance on wall models. Several investigations (for example [127]) have concluded, that smooth surface separations cannot be properly captured with wall functions when applied in the context of RANS methods. [117]
As mentioned above, the main disadvantage is the limitation of speeds up to about Mach 0.4. In the automotive sector this represents no restriction, where in aerospace industry the travel speeds are much higher. With recent extensions of LBM codes, also decent results on higher speeds, even supersonic flows can be achieved. This was possible by increasing the number of possible velocity directions (for example D3Q27 [154]) within the LBM algorithm. Another disadvantage is the non-existence of a consistent thermal-hydrodynamical scheme. [117] As this is not within the scope of work of this thesis, this topic is not discussed here.
To summarise LBM, the following listing shows its main characteristics with some comparisons to NS based methods: [117]

- LBM is based on physics, that are much simpler than the NS equations.
- Due to the simpler physics, also the algorithmic implementation is less complicated.
- LBM is, in contrast to NS based methods, most efficiently implemented on Cartesian grids for any type of geometry.

- Generation of Cartesian grids allow highly automated and robust grid generation of complex geometry.
- Same as unstructured grids, Cartesian grids also require wall-modelling. This is similar to NS grids with some accuracy implications on boundary layer prediction.
- LBM is, in contrast to NS based methods, inherently unsteady.
- For steady-state solutions it is computationally about one order more expensive compared to the steady-state RANS method.
- For unsteady solutions LBM computationally about one order less expensive compared to NS based methods.
- LBM offers low numerical dissipation (refer also to chapter 3.4.4 LES). Therefore it is highly suitable for wake and detached flow simulation as well as for aeroacustics.
- As LBM can easily be parallelised, it is ideally suited to run on massive modern HPC clusters with thousands of processor cores.

3.4.12 Further Methods

There is a number of further CFD methods existing. They are mainly used in university settings or research institutions and will not be introduced here. Most of these methods may never reach (automotive) industrial usability. Some of these methods are: [146]

- Limited Numerical Scales
- Flow Simulation Methodology
- Nonlinear Disturbance Equations
- Extra-Large Eddy Simulation
- Semideterministic Method
- Organized Eddy Simulation
- Self-Adapting Model

3.5 Some Thoughts

In general, linear EVMs do not resolve normal-stress anisotropy and do not account for transport of stresses by convection and diffusion. They also overestimate the stresses at higher strains. Furthermore, linear EVMs do not respond correctly to curvature strain, normal strain and rotation. This makes EVMs appropriate for flows with a single dominant shear stress. The absence of eddy viscosity reduces the robustness of most solution algorithms. Therefore nonlinear EVM (v2f) is more stable than RSM. However, two-equation models are poor in predicting separated flows. Especially curved surfaces cause problems for these models. The prediction of separation on sharp edges is less problematic. [17]

"Using eddy viscosity is like keeping one foot on the bottom when learning to swim - a restriction, but helpful if one does not venture into deep water." [21]

"Will RANS survive LES? I think yes, at least for the next few decades. True, we are going to witness an increased use of LES with the increase in computing power and wider accessibility of inexpensive high performance computers. Further improvements in LES-specific numerics and subgridscale models can be foreseen. Already, we are witnessing LES on meshes with $O(10^8)$ cells and it is realistic to expect that within a decade such computations would be much more frequent. LES in knowledgeable hands will take an increasingly important role as a research tool in parallel with DNS. But we will probably witness also more LES abuse and false claims: LES is relatively easy to perform provided one has sufficient computing power at disposal. And, temptations are great. Conventional LES on a too coarse grid of wall bounded flows, especially in attached flows regions, can be very erroneous and inferior to even simple conventional RANS." [62]

"Variable-resolution turbulence simulation schemes can be classified into two general categories: hybrid methods and bridging approaches. Hybrid computations entail Reynolds averaged Navier-Stokes (RANS) calculations in some flow regions and large eddy simulations (LES) in others. There exists no clear consensus on the criterion for switching from RANS to LES. The bridging methods, on the other hand, seamlessly transition from one flow resolution to another. The present approaches, e.g. PITM or PANS are based in a 'solid' mathematical background and highlight its advantages over hybrid methods." [54]

4 Methods Implemented in ANSYS Fluent

4.1 Reynolds-Averaged Navier-Stokes Simulation (RANS)

The steady-state RANS simulation is the most common used method for engineering problems and preferable for many applications. However, some flow problems limit the reachable accuracy with RANS. ANSYS Fluent offers the largest suite of turbulence models in commercial codes. [121] Following turbulence models are implemented in ANSYS Fluent (R17.2): [6, 121]

- Inviscid (0 equations)
- Laminar (0 equations)
- Spalart-Allmaras (1 equation)
  - Vorticity-Based, Strain/Vorticity-Based
- k-epsilon (2 equations)
  - Standard, RNG, Realizable
- k-omega (2 equations)
  - Standard, BSL, SST
- Transition k-kl-omega (3 equations)
- Transition SST (4 equations)
- Reynolds Stress (7 equations)
  - Linear Pressure-Strain, Quadratic Pressure-Strain, Stress-Omega, Stress-BSL

With an additional license, the v2f model is also available within ANSYS Fluent (since R6.1.18). The model can be activated via the following text user interface (TUI) command: [5]

```
allow-v2f-model
```

In addition to these steady-state turbulence models, three unsteady model groups are available in ANSYS Fluent (SAS, LES, DES).

4.2 Scale Resolving Simulation (SRS)

In contrast to the RANS methods, SRS delivers a unsteady instead of a steady-state solution. The advantage hereby is the potential for improved accuracy of the result as a decent part of the turbulent eddy structures is resolved. The unfortunate disadvantage is, that this methods require a lot of computational power. That comes firstly with the requirement for higher grid resolutions and secondly the need for small time steps to get a representative transient result. Additionally, this results in an extensive amount of produced data (which certainly has to be stored somewhere). ANSYS Fluent provides the following unsteady methods for SRS: [103, 121]

4.2.1 Scale-Adaptive Simulation (SAS)

SAS is an improved URANS formulation, which is capable of resolving the turbulent spectrum in unstable conditions. This results in a LES behaviour in unstable and a RANS behaviour in stable flow conditions. It is a in-house development from ANSYS, Inc. and is a variation of ZLES. [121] For further information, refer to chapter 3.4.5.
Compared to LES or DES, SAS has some decent advantages in certain flow conditions (unstable flow). Pure LES needs an extensive amount of computational power to resolve the whole flow domain including boundary layer. DES, in contrast, does not resolve the wall-near area, but needs an additional description for turbulence in that area. Both, LES and DES, are very sensitive to grid resolution and boundary conditions. DES eliminates some of this sensitivity.
SAS is suitable for highly and moderately unstable flows, but not for marginally unstable flows. The reason is already described in chapter 3.4.5. SAS covers the boundary layer automatically and switches into a LES behaviour. If unsteadiness is just very small, it stays in steady-state RANS mode, which delivers a good solution in that case anyway. In contrast to LES, SAS is also applicable in RANS meshes. [121]

4.2.2 Large Eddy Simulation (LES)

The theory of LES is described in chapter 3.4.4. ANSYS Fluent provides the following sub-grid scale models for LES: [7, 121]

- Smagorinsky-Lilly
- Wall adapting local eddy (WALE)
- Wall modelled large eddy simulation (WMLES)
- Wall modelled large eddy simulation stress-omega (WMLES S-O)
- Kinetic energy transport

As described in chapter 3.4.4, LES depends on time and space discretisation and resolves the large turbulent scales directly. The sub-grid turbulence is resolved by EVMs which are listed above (SGS models). The clear advantage of LES is, that the important large turbulent scales, which are responsible for energy transport, are resolved. Some of the drawbacks are already stated above in the SAS chapter. Another disadvantage is, that LES is always unsteady, which goes at the costs of computing time. This limits LES to Re=10^4 to 10^5 for wall bounded flows with the current computational power. To overcome these difficulties, special wall treatment is required, which leads to the formulation of DES. For this reason LES is very rarely used and if, for simulating turbulent structures behind bluff bodies or aeroacoustics. [4, 121]

Embedded Large Eddy Simulation (ELES) (from R13.0)

In many flow cases, some regions require a LES solution. These areas are surrounded by a RANS domain. In that case, a zonal model is advantageous. RANS and LES regions have to be defined by the user separately and different models have to be used. Via a generated synthetic turbulence, the interface between the two areas is guaranteed. EVMs are applied for the sub-grid turbulence. The benefit of this method is, that it is computationally less expensive than LES, but the large eddy structures are still resolved in the user defined areas. Certainly, in the LES regions, the method is very grid and boundary value sensitive which has to be kept in mind during the mesh generation. ELES is unstable in any case in the LES as well as in the RANS zones. [121]
ELES is not applicable in strongly turbulent flows solely by the reason, that it is difficult to find an appropriate position to place the interface between the LES and RANS regions. The lesser the turbulence gets, the more sense it makes to use ELES. The ELES model can be chosen within the LES option. [4, 121]

4.2.3 Detached Eddy Simulation (DES)

For a detailed description of DES refer to chapter 3.4.6. DES is a hybrid method between RANS and LES. In this approach RANS describes the wall-near regions. LES resolves the rest of the domain. Its clear advantage, compared to pure LES, is its reduced computational effort. But this does not come without any drawbacks. Due to its sensitivity to grid resolution a shielding between the RANS and the LES region is required to overcome numerical problems. An additional description for wall-near turbulence is required. ANSYS Fluent offers four different DES models: [7]

- Spalart-Allmaras
- k-epsilon
- k-omega
- Transition SST

ANSYS Fluent offers additional methods derived from DES (DDES, IDDES, SDES, SBES) which all reduce its grid sensitivity.

SST Blending Functions

The SST model offers two blending functions within the DES mode, F_1 and F_2. Within the k-omega SST model, these functions control the blend between the wall-near active k-omega and the free stream active k-epsilon model. F_1 ensures that within the region of $y^+ < 70$ the k-omega model is dominant and beyond, the k-epsilon model. F_2 restricts the limiter to the wall boundary layer. [17] The default is F_2, which is more conservative than F_1. [4] Refer to the appendix for closer information to the k-omega SST model.

Delayed Detached Eddy Simulation (DDES)

For a description of DDES refer to chapter 3.4.6. DDES is a derivative of the classic DES method described above. In ANSYS Fluent this method is applied in combination with the transition SST or the k-omega BSL and SST turbulence model. [7]
In the case of application, DDES is pretty similar to SAS, but it requires LES resolution for all free shear flows. It is feasible for strongly as well as moderately unstable flows but not for little unstable flows. [4]

Improved Delayed Detached Eddy Simulation (IDDES)

For a description of IDDES refer to chapter 3.4.6. IDDES is a further development of DDES and has similar properties. [4]
As DDES, IDDES is available for the the transition SST as well as for the k-omega BSL or SST model. [7]

Shielded Detached Eddy Simulation (SDES)

SDES builds up on DDES-SST (see chapter 3.4.6). A strong shielding is applied between the RANS and the LES region. This removes a lot of the numerical issues of the original DES formulation. But it creates kind of an unsteadiness between the two regions. The LES has no influence on the RANS region anymore and the RANS region does not give any feedback to the LES region. Also a faster switch from RANS to LES is achieved with this shielding. [103]
Same as DDES and IDDES, this method can be chosen as a hybrid model within the transition SST as well as in the k-omega BSL or SST option (refer to figure 4.1).

Stress Blended Eddy Simulation (SBES)

SBES is the latest method released in ANSYS Fluent R17.0. It builds up on the SDES model. Its improvements are the faster switch from RANS to LES mode and a blended transition between RANS and LES regions. The current version (R17.2) offers the usual SGS models known from LES with the exception of the kinetic energy transport model. The blended eddy viscosity is defined as

$$\nu_t^{SBES} = \nu_t^{RANS} \cdot f_{SBES} + \nu_t^{LES}(1 - f_{SBES}) \tag{4.1}$$

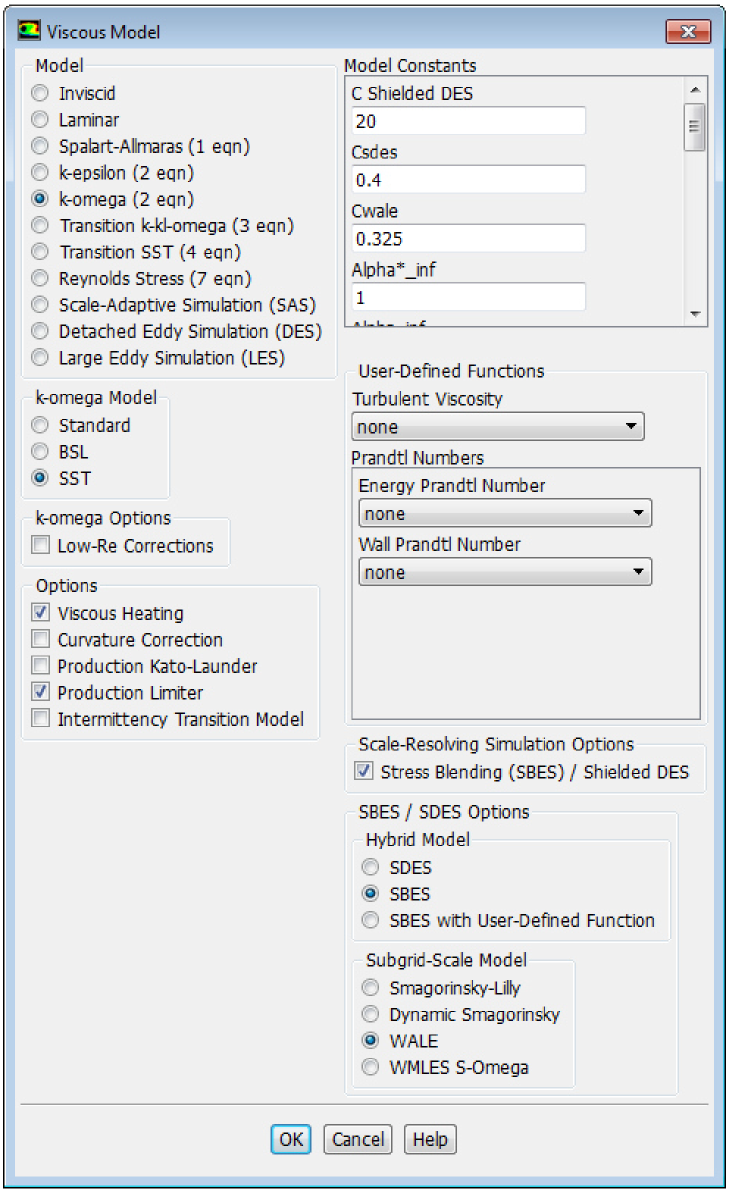
with ν_t^{SBES} as the eddy viscosity of SBES, ν_t^{RANS} as the eddy viscosity of RANS, ν_t^{LES} as the eddy viscosity of the LES region and f_{SBES} as the SBES blending function.
The blending of the RANS and LES model is performed on the turbulent stresses. The Reynolds stresses (τ'_{ij}) are calculated via the RANS method and the LES sub-grid scale stresses (τ^{LES}) via the LES model. The blending function f is determined at a specific point in the flow field and provides a smooth transition ($0 \leq f \leq 1$) between RANS ($f = 1$) and LES ($f = 0$) regions.
SBES is capable of offering results similar to existing models like WMLES, DES or DDES and offers additiolally the potential to mix and match the different RANS and LES models. As SDES, SBES also provides an unambiguous visualisation of the transition between the RANS and LES regions. For challenging flow cases there is also the potential, that SBES is able to deal more easily with it and delivers improved accuracy in the results than existing models. [104] SBES offers the following advantages: [103]

- An asymptotic shielding of the RANS boundary layer against the LES region.
- An explicit switch to user-specified LES model in the LES region.
- A rapid switch (or transition) from the RANS to LES region.
- A clear visualisation of the RANS and LES regions based on shielding function.
- A wall-modelled LES capability once in LES/WMLES mode.

Same as SDES, the SBES model can be chosen as a hybrid model within the transition SST as well as in the k-omega SST option (refer to figure 4.1).

4.3 Overview

Figure 4.2 shows a comprehensive overview over the whole program structure of all ANSYS Fluent models. This listing is based on the release 17.2 (August 2016). It comprises all models from the basic ANSYS Fluent package. Special models, like v2f, where a special license is required, are included. Two similar figures for CD-adapco STAR-CCM+ v11.04 as well as for freeware OpenFOAM v4.1 can be found in the appendix.



Viscous Model

Model

○ Inviscid
○ Laminar
○ Spalart-Allmaras (1 eqn)
○ k-epsilon (2 eqn)
● k-omega (2 eqn)
○ Transition k-kl-omega (3 eqn)
○ Transition SST (4 eqn)
○ Reynolds Stress (7 eqn)
○ Scale-Adaptive Simulation (SAS)
○ Detached Eddy Simulation (DES)
○ Large Eddy Simulation (LES)

k-omega Model

○ Standard
○ BSL
● SST

k-omega Options

□ Low-Re Corrections

Options

☑ Viscous Heating
□ Curvature Correction
□ Production Kato-Launder
☑ Production Limiter
□ Intermittency Transition Model

Model Constants

C Shielded DES: 20
Csdes: 0.4
Cwale: 0.325
Alpha*_inf: 1

User-Defined Functions

Turbulent Viscosity: none

Prandtl Numbers

Energy Prandtl Number: none
Wall Prandtl Number: none

Scale-Resolving Simulation Options

☑ Stress Blending (SBES) / Shielded DES

SBES / SDES Options

Hybrid Model

○ SDES
● SBES
○ SBES with User-Defined Function

Subgrid-Scale Model

○ Smagorinsky-Lilly
○ Dynamic Smagorinsky
● WALE
○ WMLES S-Omega

Figure 4.1: The ANSYS Fluent R17.2 viscous model dialog box with the SDES and SBES options within the k-omega SST model [7]

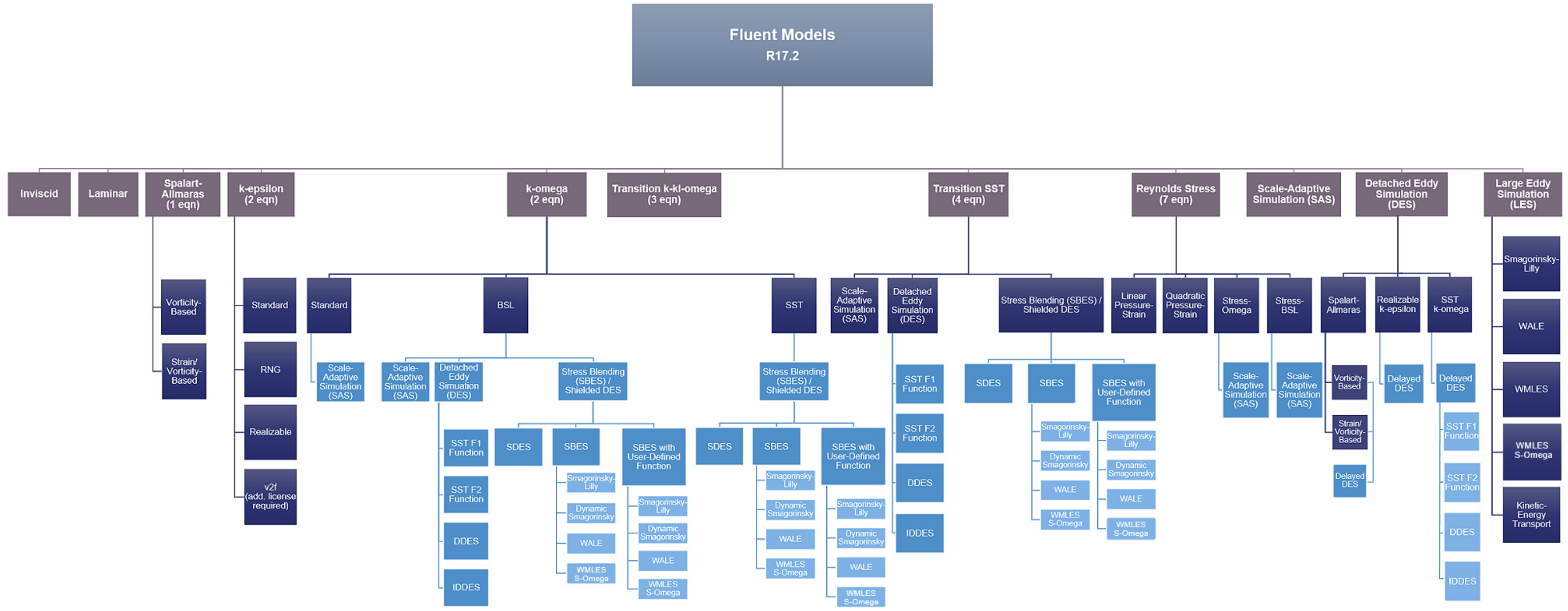


Figure 4.2: Full turbulence model structure of ANSYS Fluent R17.2 [7]

5 Numerical Aspects

5.1 Mesh Recommendations

5.1.1 Mesh Regions

Shortly after the introduction of the DES method (1997) [142], Spalart published a proposal for grid generation for the DES method. As described in the theory chapter above, it is obvious, that a LES or DES method has different mesh requirements than a RANS method. In his paper *"Young-Person's Guide to Detached-Eddy Simulation Grids"* a detailed description about the different mesh regions requirements in a DES domain is given. Depending on the topological position around the simulated body, the mesh regions can be classified as followed (refer to figure 5.1):

- Euler region (ER)
- RANS region (RR)
  - Viscous region (VR)
  - Outer region (OR)
- LES region
  - Viscous region (VR)
  - Focus region (FR)
  - Departure region (DR)

Each region has its specified mesh recommendations. For further information about mesh generation, refer to the original paper. [145]

5.1.2 Mesh Element Types

For decades, hexahedral as well as tetrahedral meshes were the standard mesh elements in engineering discretisation challenges. Especially hexahedral elements are highly suitable for creating structured meshes, which are useful in modelling the boundary layer. Using tetrahedral elements instead of hexahedral in the boundary layer increases the cell number massively. On the other hand, hexahedral elements are restricted in complex geometry. [24, 123]

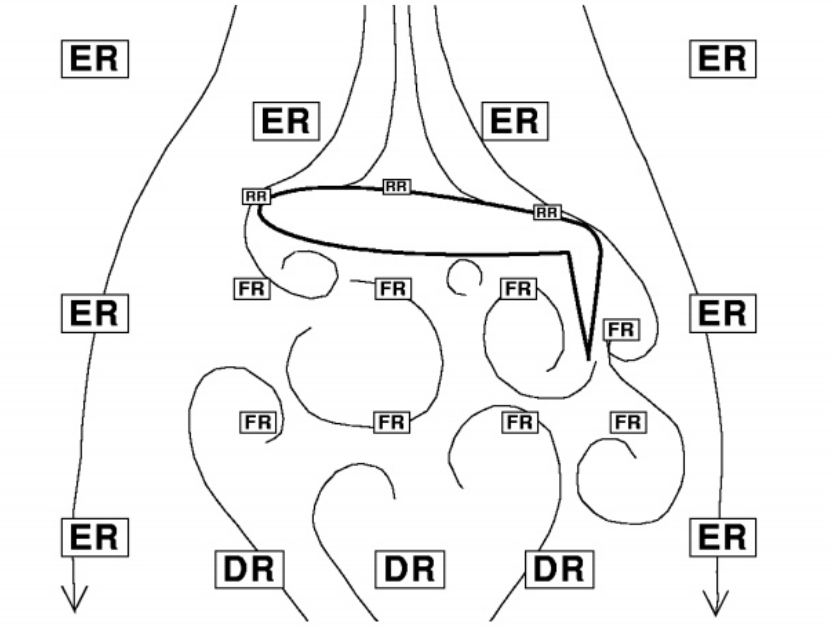


Figure 5.1: Sketch of flow regions around tilt-rotor airfoil in rotor downwash during hover [145]

Recommendation of CD-adapco

Tetrahedral elements are standard in all major CFD software tools. Their big advantage is, that both, their faces and volume centroid location, are very well defined. This makes them the perfect elements for automatic mesh generation. Complex geometry can be meshed with tetrahedral elements as well as simple geometry. The drawbacks therefore are, that they cannot be stretched so much, what restricts their flexibility. Furthermore, they only have four faces and therefore neighbours. This leads to numerical problems, which are described in. To achieve accurate solutions and good convergence properties, this results in the need of special discretisation schemes, which make the code more complicated and a large number of cells which increase memory and computing time. [123]
A polyhedral mesh, in contrast, offers the same automatic meshing benefits as tetrahedrons while overcoming these problems. A major advantage of polyhedral cells is the fact, that they have many neighbours (typically of order ten) which offers some benefits in numerics. Further details can be found in [123]. Some would say, more faces and therefore neighbours means more storage and computing effort per cell. But this is more than compensated by the higher accuracy (at lower cell numbers) for polyhedral meshes. Its lesser sensitivity to stretching and its flexibility in automatic mesh generation offers almost limitless possibilities in mesh structures and automatic meshing. Despite all these advantages compared to conventional cell types, many of the existing numerical codes

cannot be extended to polyhedral meshes due to numerical difficulties. In [23] and [25] a new discretisation method, called the mimetic finite difference (MFD) method, was developed to overcome these problems by a group of scientists at the Los Alamos National Laboratory. [24, 123] They have not been implemented in any commercial software yet, but might be in future.

In [123] a simulation (CD-adapco STAR-CCM+) of the flow through an engine water jacket was executed with six tetrahedral as well as six polyhedral meshes with varying grid resolution from very fine to coarse. The quantity of interest was the pressure drop between the inlet and the outlet. In all cases, the same discretisation and solution method were used. It is less interesting that the results with the polyhedral mesh is always more accurate than the result with a tetrahedral mesh with comparable cell number. But the following comparison shows the potential of polyhedral mesh structures (refer to figure 5.2): The result from a polyhedral mesh is slightly more accurate than the result from a tetrahedral mesh with a six times higher cell number. The computing time is about six times less with a polyhedral mesh compared to a tetrahedral mesh. Several more comparative applications have been performed so far, which are verifying this result. Also the computational convergence is better by using a polyhedral mesh than using a tetrahedral mesh. [123, 148]

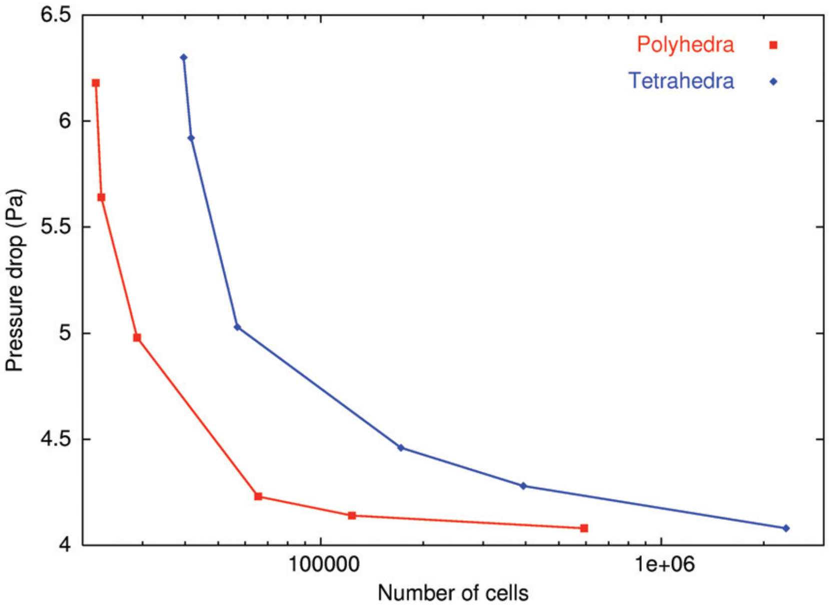


Figure 5.2: Comparison between tetrahedral and polyhedral meshes [123]

Recommendation of BETA CAE Systems

On the ANSYS 2013 Automotive Simulation World Congress [138] as well as on AMS Seminar Series 2015 [51] at the NASA Ames Research Center, BETA CAE Systems presented a comparison of various ANSA mesh types.

The test model was in both cases the DrivAer model.[1] OpenFOAM v2.3 was the used solver software. 15 different mesh combinations were tested. The k-omega SST was used as turbulence model at a velocity of 40 ms^{-1}. The simulations were executed without layers and with low Re layers as well as with high Re layers.[2] Four different meshes were tested. Namely Tetra, HexaInterior, HexaPoly and Polyhedral, each with coarse medium and fine resolution.

The medium Tetra mesh proved to be the most accurate solution (figure 5.3). It provides a spot-on drag coefficient and a deviation of 28 per cent for the lift coefficient. The polyhedral, in contrast, seemed to deviate a lot. Mesh refinement showed an acceptable mesh independence at medium size meshes. The presence of layers also proved to be necessary to capture flow details. Low Re layers offered more accuracy (figure 5.4). High Re layers require less computational resources. Another result of the study is, that force averaging (drag and lift) should be performed using several thousand iterations. Figure 5.5 shows a mesh comparison for the notchback model.

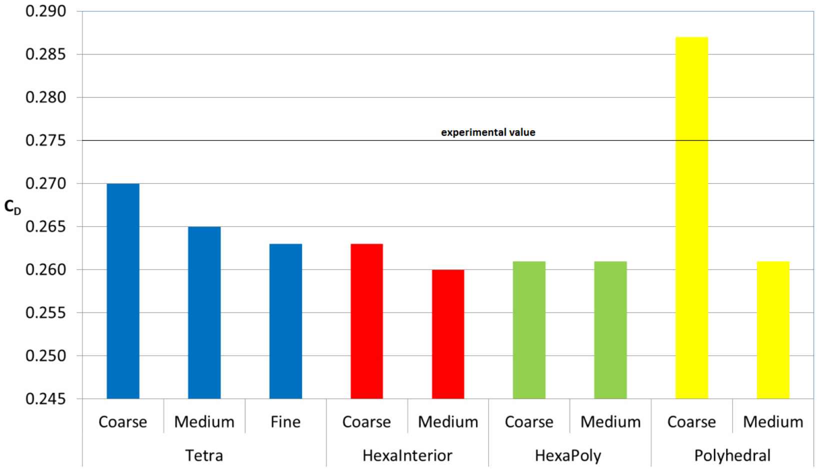


Figure 5.3: Effect of mesh types with high Re layers on force prediction for fastback DrivAer model with moving ground [138]

The results of this study suffer a bit from the inaccurate CFD modelling due to insufficient information of the physical model set-up of the DrivAer WT test. The study is still going on and will presumably be extended over the next few years.

[1] For a detailed description of the DrivAer model, refer to chapter 6.1.2

[2] A short description of the difference of low and high Re layers is given in the appendix.

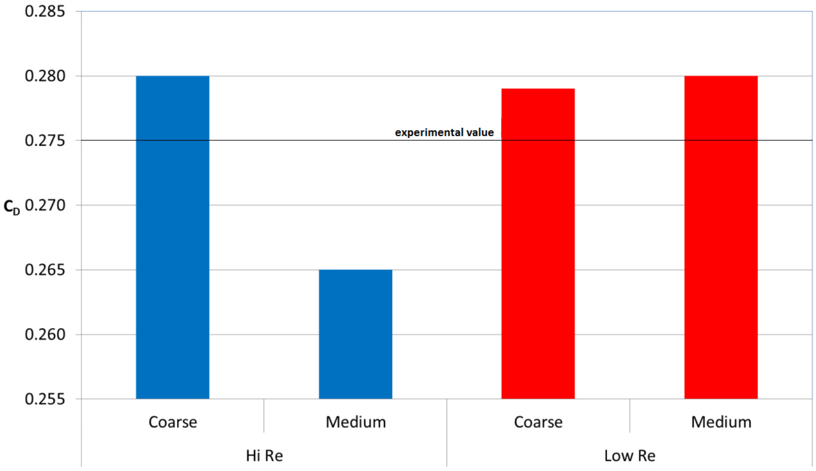
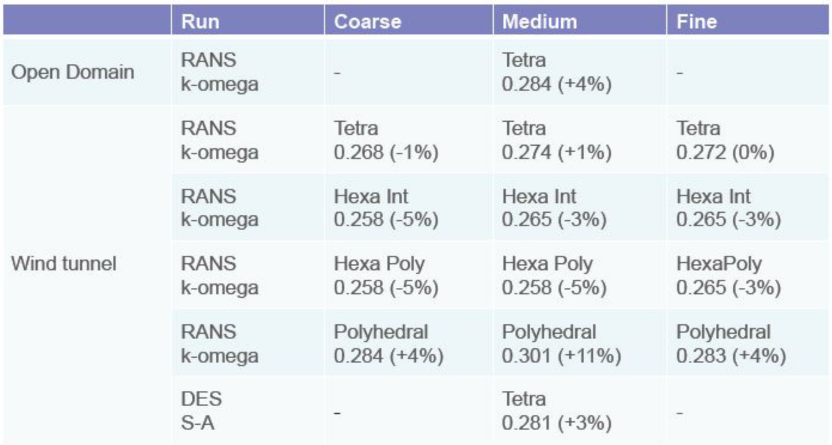


Figure 5.4: Effect of high and low Re layers on force prediction for fastback DrivAer model with moving ground [138]



	Run	Coarse	Medium	Fine
Open Domain	RANS k-omega	-	Tetra 0.284 (+4%)	-
Wind tunnel	RANS k-omega	Tetra 0.268 (-1%)	Tetra 0.274 (+1%)	Tetra 0.272 (0%)
	RANS k-omega	Hexa Int 0.258 (-5%)	Hexa Int 0.265 (-3%)	Hexa Int 0.265 (-3%)
	RANS k-omega	Hexa Poly 0.258 (-5%)	Hexa Poly 0.258 (-5%)	HexaPoly 0.265 (-3%)
	RANS k-omega	Polyhedral 0.284 (+4%)	Polyhedral 0.301 (+11%)	Polyhedral 0.283 (+4%)
	DES S-A	-	Tetra 0.281 (+3%)	-

Figure 5.5: Comparison with experimental drag coefficient value of 0.272 for notchback DrivAer model [138]

Other Recommendations

In [150], a hydrocyclone geometry was investigated using pure hexahedral and tetrahedral/prism meshes. For this simple geometry, the hexahedral mesh outperformed the tetrahedral/prism type mesh.
However, there is no clear thread, which mesh element types should be used in which application area. When talking to people, most favour polyhedral meshes. When reading papers, some recommend polyhedral meshes [123], some recommend tetrahedral meshes [138] and some recommend hexahedral meshes [150]. Anyway, the use of a specific mesh type is dependent strongly on geometry and area of application. The only statement,

that can be made on mesh types is that structured meshes should be applied if possible. [129]

5.2 Time Step

To determine the needed time step number, it must be referred to the focus region. In this region, the large eddies are present and the mesh size is designed to resolve these eddies in sufficient accuracy. Nevertheless, all regions run with the same time step.
The required time step [129]

$$\Delta t = \frac{\Delta x}{U_{max}} \tag{5.1}$$

depends on the mesh size of the focus region Δ_0 and the (estimated) highest velocity U_{max} in the focus region. This is usually somewhere between 1.5 and several times the free stream velocity. This formula delivers a guiding value for a time step based on accuracy, not on stability. [145] Based on equation 5.1, the CFL (Courant-Friedrichs-Lewy) number [129]

$$CFL = \frac{U_{max} \cdot \Delta t}{\Delta x} \tag{5.2}$$

was introduced. Δt describes the discrete time step and Δx the discrete length interval. The CFL number defines how many mesh cells per time step a flux value is allowed to move. The CFL condition predicates, that [129]

$$CFL < 1 \tag{5.3}$$

must be satisfied for the explicit Euler method to be stable. Similar conditions apply for other discretisation schemes. So the CFL condition is a good indicator to set the time step for transient simulations. However, it is not a matter stability, but of accuracy. If CFL number is too large, the flow information passes through too many mesh cells in one time step. Thus, fluctuations are smeared out and the simulation result suffers from this. [129] For complex geometry, the CFL number is not always reliable (for example, if the flow passes through a region of highly stretched cells). [103]

6 Former Investigations

After a theoretical introduction of the most important CFD methods, this chapter illustrates a few former investigations on different methods. Only methods with a potential in short- to mid-term application for industrial use are covered. That is firstly a comparison between RANS simulations with various turbulence models and some hybrid RANS-LES simulations, published by Ashton et al. in 2016. [12] This paper investigates the Ahmed body as well as the DrivAer car model. Subsequently an investigation of the DrivAer car model using the PANS method is presented, published by Jakirlić et al. in 2016. [77] Last but not least, an investigation of the DrivAer model with LBM is presented. This investigation was published by Schäufele in 2010. [130] A complete comparison between URANS, IDDES and LBM can be found in BMW paper [81].

6.1 Assessment of RANS and DES Methods for Ahmed Body and DrivAer Car Model

6.1.1 Ahmed Body

A row of investigations, simulating the Ahmed body with LES and DES have been done over the past few years. [10] [11] [58] [71] [83] [87] [94] [111] Recently, Ashton et al. [12] enlarged these investigations by additionally simulating the Ahmed body with different RANS turbulence models and comparing the results from RANS, LES and DES.
The Ahmed body, shown in figure 6.1, is a well defined and simplified generic car geometry which is well investigated experimentally as well as numerically in literature [2], [10], [94], [134], [97] and [28]. The model incorporates a bluff front and a slant at the rear base. While the Ahmed body represents a strongly simplified car model, it nevertheless provides many flow features that can be found in the flow field of a real-life car, such as complex vortex interactions and large separation areas in the wake as well as counter-rotating vortices produced by the rear slant. The slant angle has a major influence on the flow field around the back of the car. At 35°, the counter-rotating vortices are weaker and the flow is completely detached over the entire slant back of the body. At 25°, the counter-rotating vortices are strong enough to help to bring enough momentum into the flow reattaching half way down of the slant back. [12]
Simulations were done by CD-adapco STAR-CCM+ (v9.04). The Ahmed body was simulated with a base angle of $\Phi = 35°$ as well as with the more challenging $\Phi = 25°$. The flow is at a Reynolds number $Re = 768\,000$, based on the body height and a free stream velocity $U_\infty = 40\,\mathrm{ms}^{-1}$. For a closer insight of the applied boundary conditions, refer to the original paper by Ashton [12].

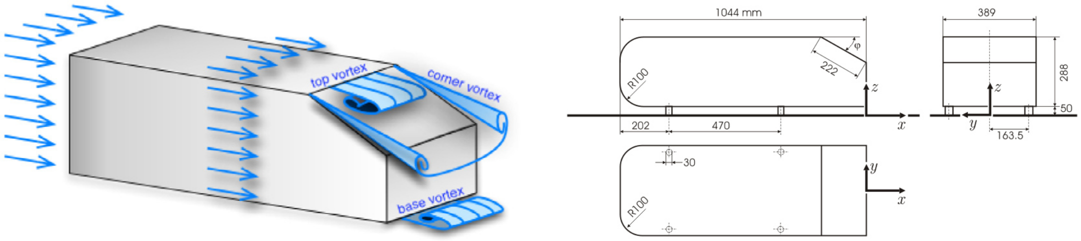


Figure 6.1: Ahmed car body flow physics and Ahmed car body dimensions. [2] [12]

For the 25° base angle, a structured 16 million cell mesh was used. The first near-wall cells had a $y^+ < 1$. The refinement was concentrated on the separation area at the back of the body.

For the 35° base angle, a unstructured polyhedral mesh with different refinements (coarse, medium, fine) with a first near-wall cell $y^+ < 1$ was used. All pre-processing was done by BETA-CAE Systems ANSA.

The mesh sizes were approximately four, ten and 19 million cells. The RANS simulations ran with a steady coupled incompressible FVM solver. For discretisation, a second order upwind scheme was used. The DDES simulations ran with an unsteady segregated incompressible FVM solver. For discretisation, a hybrid numerical scheme, which jumps between a bounded central differencing scheme in LES regions and a second order upwind scheme in RANS regions. For temporal discretisation, a second order Crank-Nicholson scheme was used. Throughout the entire LES region, the CFL < 1 condition is ensured. The most important results are shown here. For more insight, refer to the original paper by Ashton. [12]

Figure 6.2 shows mean streamwise velocity and TKE over as well as behind the slant rear of the Ahmed body using SST, EB-RSM and SST-IDDES. It can be seen, that the SST-IDDES provides by far the most accurate result. Especially behind the body, the experimental data agrees very well with the simulated results for the mean streamwise velocity as well as for TKE. However, there is still a clear under-prediction of TKE over the slant rear. This results in too little turbulence and mixing and therefore an over-prediction of separation. [12]

Figure 6.3 shows the region of separated flow for SST, EB-RSM and SST-IDDES. As mentioned above, the SST-IDDES predicts too much separation. But compared to that, the RANS models predict a much larger recirculation area. [12]

Figure 6.4 shows again the mean streamwise velocity as figure 6.4, but this time using SST-URANS, SST-DES and SST-E-DES (embedded-DES). Especially the embedded simulation shows very promising results for the Ahmed body. The domain was cut prior to the separation point and placing an inlet boundary condition on this plane, at which synthetic turbulence could be produced. However, this is not suitable for a realistic car configuration like the DrivAer model, as the modelling expenses for the synthetic turbulence would be too high. [12]

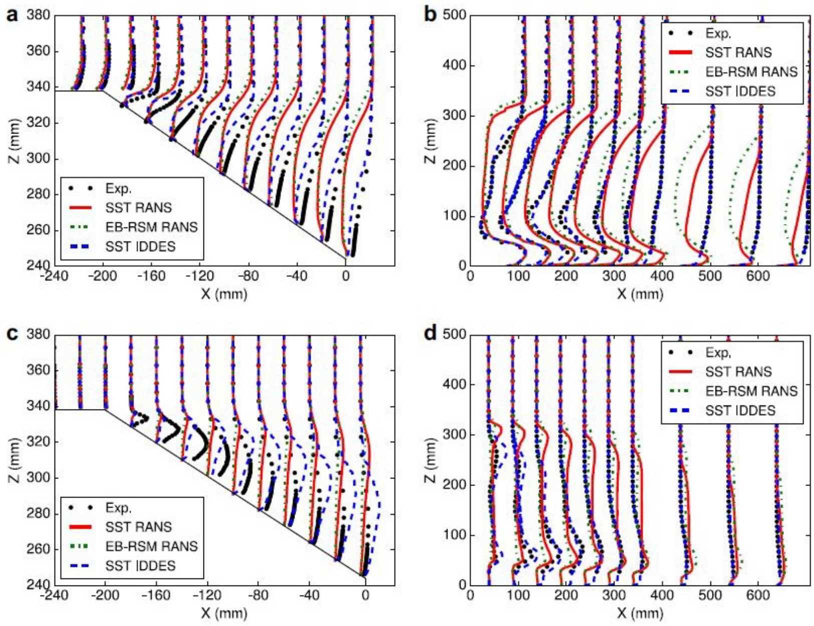


Figure 6.2: **(a)** Mean streamwise velocity along and **(b)** behind the Ahmed body and **(c)** mean TKE along and **(d)** behind the Ahmed body for the SST EBRSM RANS and SST-IDDES models. [12]

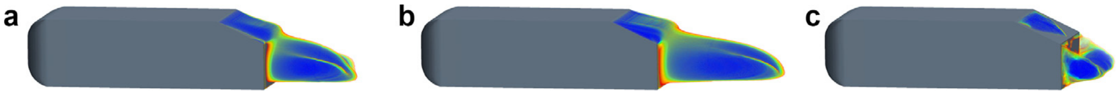


Figure 6.3: Visualisation of the separated flow for the Ahmed body, using the **(a)** SST RANS, **(b)** EBRSM RANS and **(c)** the SST-IDDES model. [12]

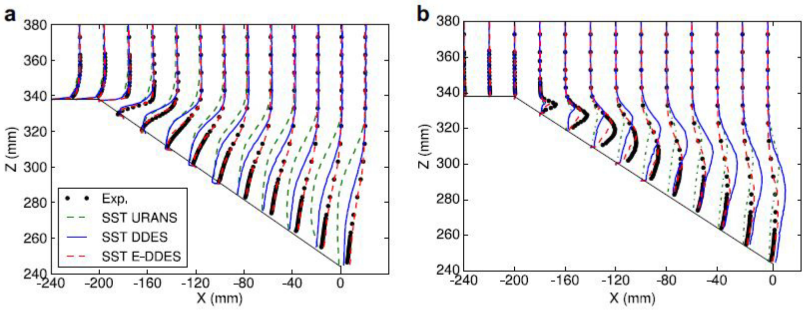


Figure 6.4: **(a)** Mean streamwise velocity along the Ahmed body for the SST-URANS, SST-DDES and SST-E-DDES models. **(b)** Mean TKE along the Ahmed body for the SST-URANS, SST-DDES and SST-E-DDES models. [12]

6.1.2 DrivAer Model

The DrivAer model, shown in figure 6.5, is a generic car model, created from original geometries of the Audi A4 and the BMW 3 series. In contrast to the Ahmed body, the DrivAer model represents no simplified car geometry, but a car model with all details of a commercial passenger car. The purpose of this model is to have a realistic car geometry with no confidentiality agreement preventing the publication of scientific investigations. [65] [66]

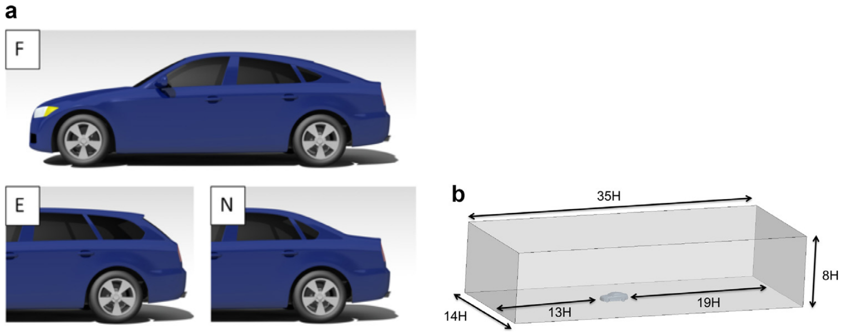


Figure 6.5: **(a)** DrivAer models, F-fastback, E-estate, N-notchback **(b)** DrivAer computational domain size. [12]

Most simulations were done by CD-adapco STAR-CCM+ (v9.04) and some more additionally also with freeware EDF Code_Saturn (V3.0.1) to evaluate differences between different codes and numerical approaches, respectively. CD-adapco STAR-CCM+ bases on FVM and EDF Code_Saturn bases on a finite element method (FEM). [12]

Two configurations of the DrivAer model were simulated. Namely estate and fastback, with the modification of a smooth underbody.[1] The same meshing approach was applied for both configurations. The flow is at a Reynolds number $Re = 1\,480\,000$, based on the car height and a free stream velocity $U_\infty = 40\,\mathrm{ms}^{-1}$. Turbulent viscosity ration was set to $\nu_t = 20$ and turbulent intensity to 1%. Simulation solver settings were the same as with the Ahmed body. [12]

For the RANS simulations, a 80 million cell grid composed from prism cells in the boundary layer and polyhedral cells in the rest of the domain is used. The same mesh is used for the coarse hybrid RANS-LES simulations as well as a another 100 million pure polyhedral cells mesh for the fine hybrid RANS-LES. [12]

Figures 6.6 and 6.7 show the comparison of the results for the drag and lift coefficient for the different turbulence models. It can be seen at first glance, that no model is able to predict the coefficients for both car configurations correctly. One model can be good for the first configuration, but fails dramatically for the second one. The SST model for example predicts the drag coefficient very well for the fastback, but fails at the estate

[1] Due to various configurations of the DrivAer car model and different mountings to the WT, experimental data varies from experiment to experiment. In chapter 6.2 another example of the DrivAer model investigated with the PANS method is presented, where the experimental data are quite different as presented in this example. This may be due to this circumstances.

configuration. Figure 6.7 shows the lift coefficient, where it is clearly visible, in a way that only the unsteady IDDES methods could come close. [12] In table 6.1, the results from various research groups are compared.

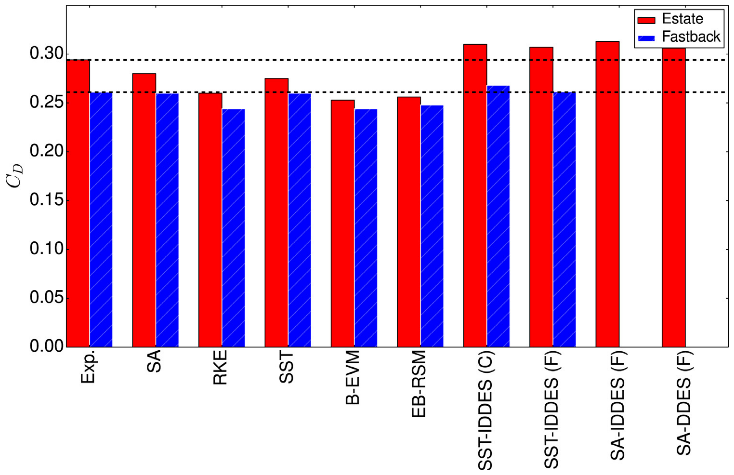


Figure 6.6: Drag coefficient for the estate and fastback configurations using RANS and DES models. [12]

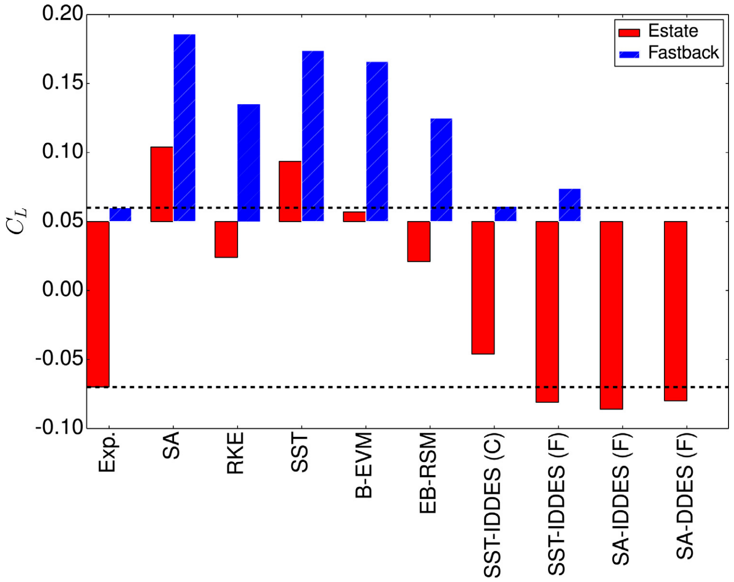


Figure 6.7: Lift coefficient for the estate and fastback configurations using RANS and DES models. [12]

Model	Estate		Fastback	
	c_D	c_L	c_D	c_L
Exp.	0.294	-0.12	0.261	0.01
$k-\omega$ SST from Heft et al. [65]	n/a	n/a	0.243	n/a
$k-\omega$ SST from Peters et al. [124]	n/a	n/a	0.256	0.069
EARSM from Guilmineau [59]	n/a	n/a	0.254	0.079
Spalart Allmaras	0.280	0.054	0.260	0.136
Realizable $k-\epsilon$	0.260	-0.026	0.244	0.085
$k-\omega$ SST	0.275	0.0436	0.260	0.124
$k-\epsilon$ B-EVM	0.253	0.007	0.2435	0.116
EB-RSM	0.256	-0.029	0.2482	0.075
SST-DES from Guilmineau [59]	n/a	n/a	0.266	0.024
SST-IDDES (Coarse)	0.310	-0.096	0.268	0.011
SST-IDDES (Fine)	0.307	-0.131	0.2615	0.024
SA-IDDES (Fine)	0.313	-0.136	n/a	n/a
SA-DDES (Fine)	0.307	-0.13	n/a	n/a

Table 6.1: Lift and drag coefficients for the computational and experimental results for the fastback and estate configurations of the DrivAer using different RANS models on 80 million fine RANS mesh and a range of DES variants on the coarse and fine meshes. [12]

By investigating table 6.1, the $k-\omega$-SST model can be identified at the most accurate RANS model (for detailed investigation, refer to original paper). In terms of the hybrid RANS-LES models, the SST-IDDES and the SA-DDES, each with fine mesh configuration, are the most accurate models. However, also the coarse mesh SST-IDDES and the fine mesh SA-IDDES show more accurate results than each RANS model (in terms of drag and lift coefficients). But in general, it can be seen, that no model, not even a hybrid-RANS-LES model, is able to predict all coefficients correctly. Tables 6.2 and 6.3 compare the most accurate steady and unsteady model. Especially in terms of the lift coefficient, the advantages of the hybrid RANS-LES method are clear.

Model	Estate			Fastback		
	c_D	$\Delta c_{D,abs}$	$\Delta c_{D,rel}$	c_D	$\Delta c_{D,abs}$	$\Delta c_{D,rel}$
Exp.	0.294			0.261		
$k-\omega$-SST	0.275	-0.019	-6.46 %	0.260	-0.001	-0.38 %
SST-IDDES (Fine)	0.307	0.013	+4.42 %	0.2615	0.0005	+0.19 %

Table 6.2: Comparison of the drag coefficients of the estate as well as the fastback for $k-\omega$-SST and SST-IDDES model

Model	Estate			Fastback		
	c_L	$\Delta c_{L,abs}$	$\Delta c_{L,rel}$	c_L	$\Delta c_{L,abs}$	$\Delta c_{L,rel}$
Exp.	-0.12			0.01		
$k-\omega$-SST	0.0436	0.1636	+136.33 %	0.124	0.123	+1 230.00 %
SST-IDDES (Fine)	-0.131	-0.011	-9.17 %	0.024	0.014	+140.00 %

Table 6.3: Comparison of the lift coefficients of the estate as well as the fastback for $k-\omega$-SST and SST-IDDES model

Figures 6.9 and 6.8 show a visualisation of the separated flow regions. There are noticeable differences in the volume and shape of the wake structures resulting from the various turbulence models. These differences are likely to be the major contribution to the differences in the drag and lift coefficients. [12]

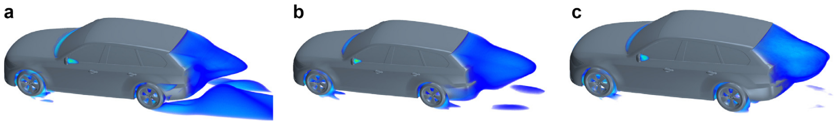


Figure 6.8: Volume render, showing separation regions for **(a)** SST RANS, (b) EBRSM RANS and **(c)** SST-IDDES model for estate vehicle. [12]

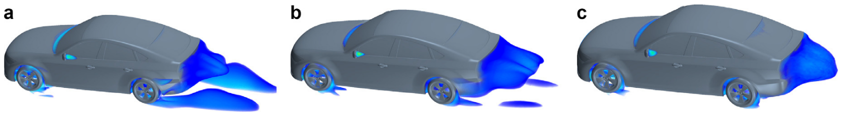


Figure 6.9: Volume render, showing separation regions for **(a)** SST RANS, (b) EBRSM RANS and **(c)** SST-IDDES model for fastback vehicle. [12]

By investigating figure 6.10, it can be seen, that each turbulence model within the RANS method predicts broadly similar pressure distribution, except for the region at bottom of the front window/bonnet intersection. Especially the SST and the RKE model predicts too strong suction in that area. However, all models under-predict the pressure coefficient for both, fastback and estate vehicle, over the roof of the car. This is due to the lack of the strut, which connects the car with the WT, in the CFD model. Furthermore, some variations between the models can be seen at the rear of the vehicle, mainly at the estate configuration. The SST model for example offers the highest drag in agreement with having the largest separation area. [12, 66]

Figure 6.11 shows the influence of the mesh cell size on the pressure coefficient. The results shown above are all from the fine grid configuration. However, it can be seen, that there is no big difference using a medium or fine mesh for most RANS models.

Figure 6.12 shows the pressure coefficient for both, fastback and estate configuration, with different models. Again it can be seen, that the hybrid RANS-LES method provides a more exact result than the SST-RANS method. This coincides with figures 6.6 and

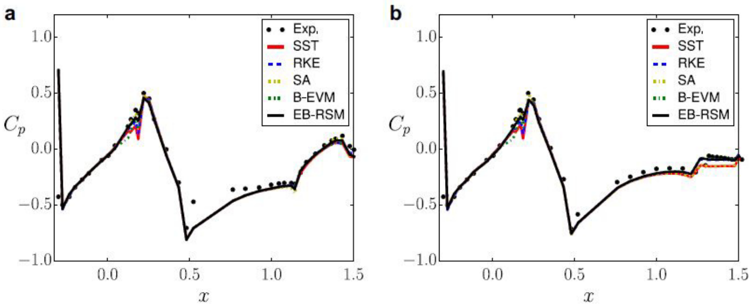


Figure 6.10: **(a)** Pressure coefficient over the top of the fastback configuration for each RANS model. **(b)** Pressure coefficient over the top of the estate configuration for each RANS model. [12]

6.7 where especially at the prediction of the lift coefficient the benefits of the hybrid RANS-LES methods opposite to the pure RANS methods is obvious.

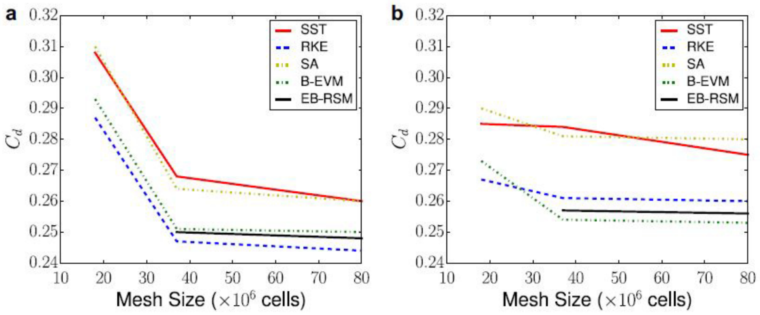


Figure 6.11: Mesh refinement for the fastback and estate configuration using RANS models. [12]

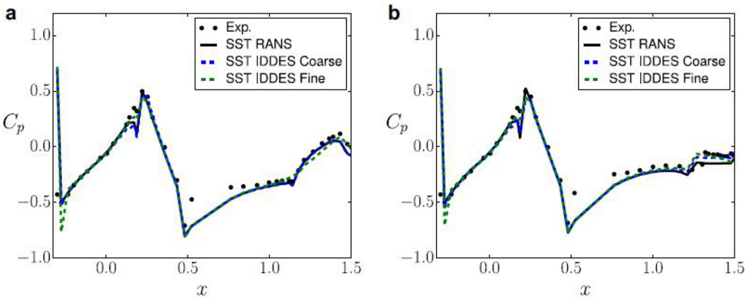


Figure 6.12: **(a)** Mean pressure coefficient over the top of the fastback configuration for each mesh using the SST-IDDES model. **(b)** Mean pressure coefficient over the top of the estate configuration for each mesh using the SST-IDDES model. [12]

A very interesting point, especially in terms of the economical aspect, is the comparison of the computational expenses between a RANS and a hybrid RANS-LES method. Table 6.4 illustrates the increased computational costs of the hybrid methods.

Method	Cells	Cores	Temporal scheme	Time per iteration	Compute time	Relative cost
SST-RANS	$80 \cdot 10^6$	512	Steady	6 s (2 500 it.)	4 h	1
SST-IDDES	$100 \cdot 10^6$	704	Transient $5 \cdot 10^{-5}$	7 s (26 000 t. steps)	50 h	17
SST-IDDES	$100 \cdot 10^6$	2 048	Transient $5 \cdot 10^{-5}$	2.5 s (26 000 t. steps)	18 h	17

Table 6.4: Computational expense of each RANS and DES simulations for the DrivAer vehicle. [12]

Due to the extensively increased computing time of hybrid RANS-LES methods compared to pure RANS simulation, Ashton suggests, that there is still a need to develop new RANS models to improve their accuracy. [12]

6.2 Example of PANS for DrivAer Car Model

Investigations with PANS on simplified ground vehicles, similar to the Ahmed body, were published in [14], [60] and [61].

In 2016, Jakirlić et al. published a paper, where the capability of the PANS method to catch a complex flow behind a realistic car model is proved. The simulations were done at Technical University of Darmstadt using AVL FIRE software. The model which is subject of this investigation is the generic DrivAer model (see chapter 6.3). In the course of this research study, three configurations of the fully detailed[2] DrivAer model were simulated (estate, fastback and notchback).

The model was simulated with a Reynolds number of $Re = 4\,870\,000$, which equals a free stream velocity of $U_\infty = 40\,\text{ms}^{-1}$. The rotation of the wheels was coupled with the moving floor (40ms^{-1}). A 24 million polyhedral cells mesh was used. The wall-near cells are hexahedral elements with a high orthogonality. The y^+ value of the wall-nearest node varies between 0.5 and 30. Along the entire walls (car surface and floor), the so-called universal wall function was used. The time step was set to $\Delta t = 0.001\,\text{s}$. [77]

Figure 6.13 shows the filter parameter f_k round the DrivAer model configuration. Values of $f_k \geq 1$ (red) indicate the regions of the flow field where the RANS method is active. Values below $f_k = 1$, which are predominantly appearing in the wake, identify the areas where LES is active. If $f_k = 0$ (dark blue), pure DNS is active. [77]

[2] This includes a fully detailed underbody with exhaust system, side mirrors and rotating wheels with brake discs and rims

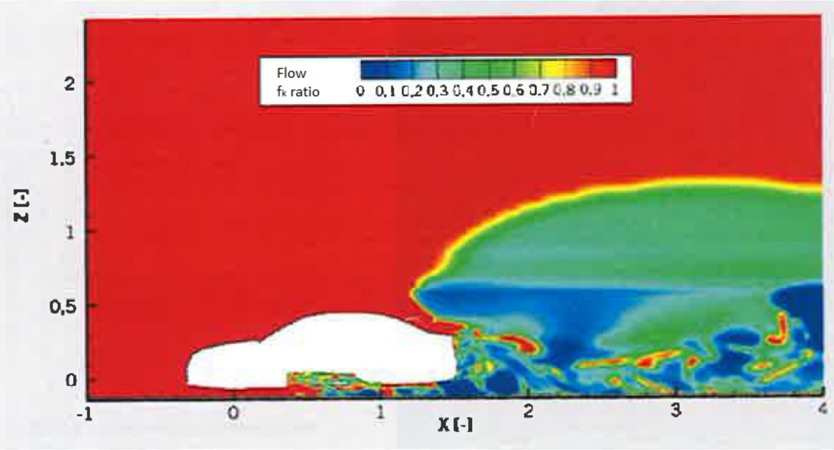


Figure 6.13: The field of the filter parameter fk round the notchback configuration of the DrivAer model. [77]

A direct comparison between the PANS-and the experimental results is visible in figure 6.14. The c_P-distribution shows an alternating positive and negative pressure gradients at the model lower and upper surface. The agreement with the experimental data is very good. The pressure coefficient is pretty similar at the underbody and the roof when approaching the rear at the estate and fastback configuration. At the rear, c_P rises aprupt at the estate, but it rises gradually at the fastback configuration. [77]

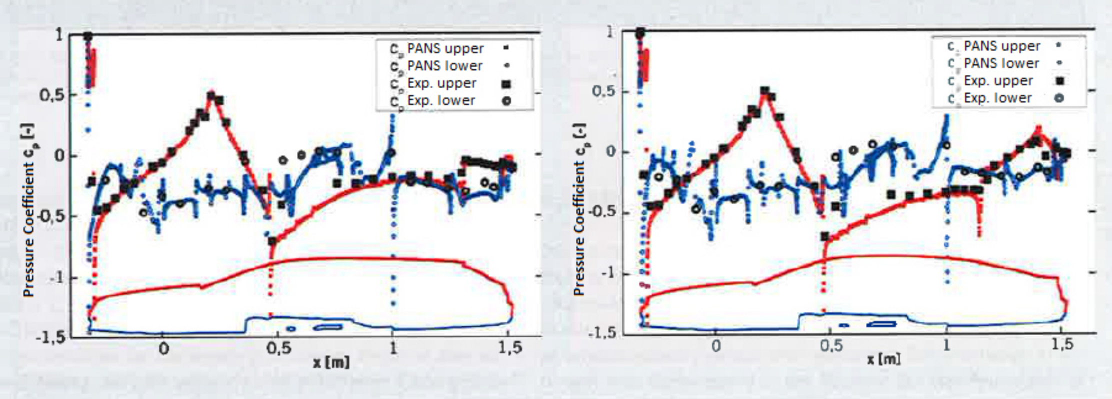


Figure 6.14: Distribution of the pressure coefficient (PANS method) at centre plane over vehicle surface at the estate (left) and fastback (right) configuration of the DrivAer model. [77]

Figure 6.15 shows the time sequence of c_D and c_L for the estate and notchback configuration. Both characteristics, without (blue line) and with wheels (red line) are shown. The initial progression represents the result of the steady-state RANS simulation. This is the initial solution for the subsequently performed URANS simulation. Following this, the PANS simulation is performed with consideration of temporal resolution. It is clearly visible, that with each refinement of the applied model, the calculated result converges up to the experimental result. Tables 6.5 and 6.6 compare the calculated result with the experimental one. [77]

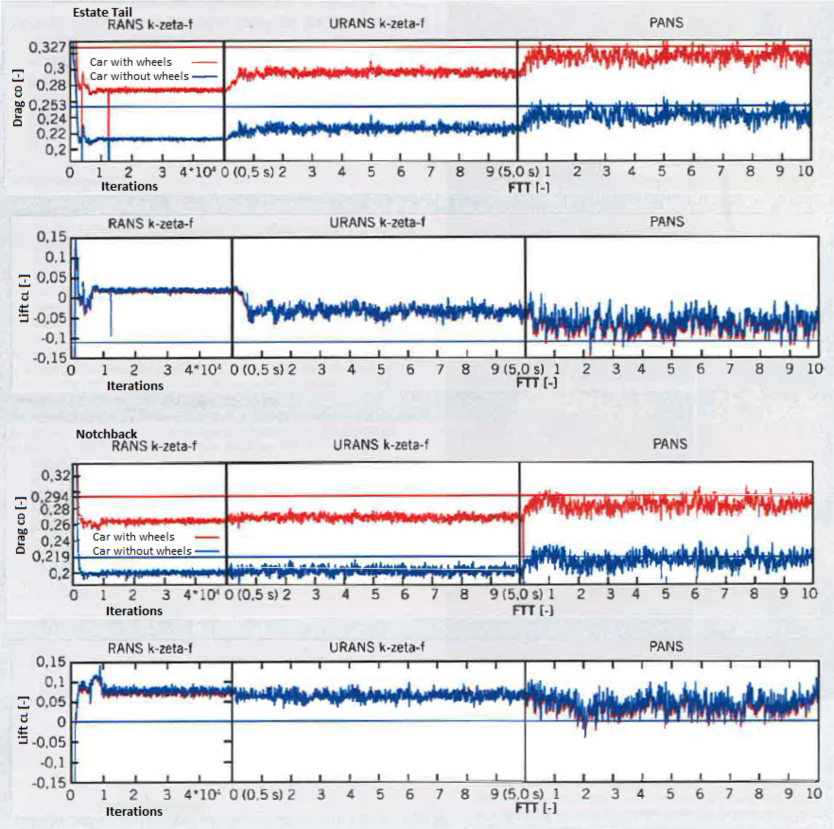


Figure 6.15: Comparison of coefficients between the three used methods on two different vehicle geometries (estate tail and notchback). [77]

Model	Estate			Notchback		
	c_D	$\Delta c_{D,abs}$	$\Delta c_{D,rel}$	c_D	$\Delta c_{D,abs}$	$\Delta c_{D,rel}$
Exp.	0.327			0.294		
RANS $k-\zeta-f$	0.273	-0.054	-16.51 %	0.262	-0.032	-10.88 %
URANS $k-\zeta-f$	0.294	-0.033	-10.09 %	0.269	-0.025	-8.50 %
PANS	0.316	-0.011	-3.36 %	0.283	-0.011	-3.74 %

Table 6.5: Comparison of the drag coefficients of the estate as well as the fastback for RANS, URANS and PANS.

Model	Estate			Notchback		
	c_L	$\Delta c_{L,abs}$	$\Delta c_{L,rel}$	c_L	$\Delta c_{L,abs}$	$\Delta c_{L,rel}$
Exp.	-0.11			0.0		
RANS $k-\zeta-f$	0.02	0.13	+218.82 %	0.07	0.07	-
URANS $k-\zeta-f$	-0.04	0.07	+63.64 %	0.06	0.06	-
PANS	-0.061	0.049	+44.55 %	0.043	0.043	-

Table 6.6: Comparison of the lift coefficients of the estate as well as the fastback for RANS, URANS and PANS.

Even though, the PANS method did not predict each value exactly, all the trends were predicted correctly. The drag coefficient increases between the notchback and the estate configuration, while the lift coefficient decreases equally.

6.3 Example of LBM for BMW Test Vehicle

A brief investigations with LBM on the Ahmed body can be found in [86].
In his paper from 2010, Kandasamy et al. published an investigation of a car model using the LBM code implemented in PowerFLOW 4.3. This research project was carried out in cooperation with the EXA Corporation and the BMW Group. As shown in figure 6.16, the investigated model was a BMW test model, similar to the DrivAer model (see chapter 6.3). This model was equipped with various front and rear (notchback, estate, fastback, hybrid notchback-fastback) geometry. [82, 130]

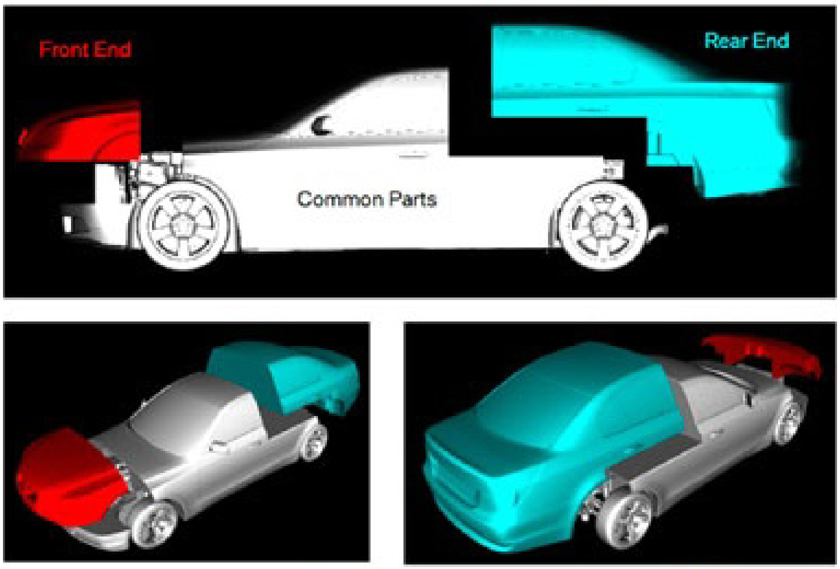


Figure 6.16: CAD surfaces of the full scale model. Front and rear end modules are removable to created different shape combinations. [82]

The model was simulated with a free stream velocity of $U_\infty = 40\,\mathrm{ms}^{-1}$. The rotating wheels were coupled to the moving floor. The VLES method, which resolves the large turbulent scales, was used. The small turbulent scales were resolved by a two-equation turbulence model. For closer details, refer to the original paper [82].
Referring to table 6.7, a traffic light system was introduced to evaluate the results from the LBM simulation compared to the experimental results.

Rating	Drag coefficient c_D	(Axis-) lift coefficient c_L
Green (good)	$\Delta c_D \leq\leq 0.005$	$\Delta c_L \leq\leq 0.010$
Yellow (acceptable)	$0.005 \leq \Delta c_D \leq\leq 0.010$	$0.010 \leq \Delta c_L \leq\leq 0.020$
Red (poor)	$\Delta c_D \geq 0.010$	$\Delta c_L \geq 0.020$

Table 6.7: Traffic light rating for difference between calculation and measurement [133]

Figures 6.17 to 6.19 show on the left hand side the absolute drag coefficient c_D as well as the lift coefficient for the front and rear axle c_{Lf}, c_{Lr} respectively. On the right hand side the difference between the calculated and measured values is shown with a colour chart, equal to table 6.7. [133]

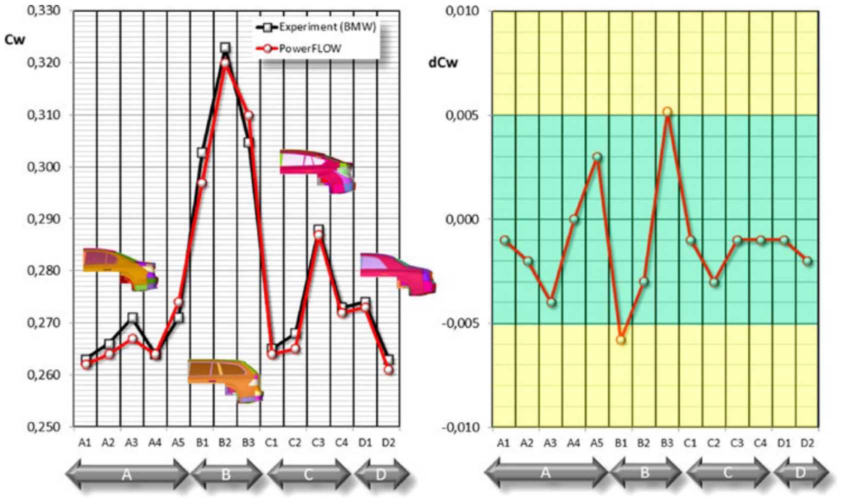


Figure 6.17: Comparison of simulations (PowerFLOW) and WT (BMW) results for drag coefficient [82]

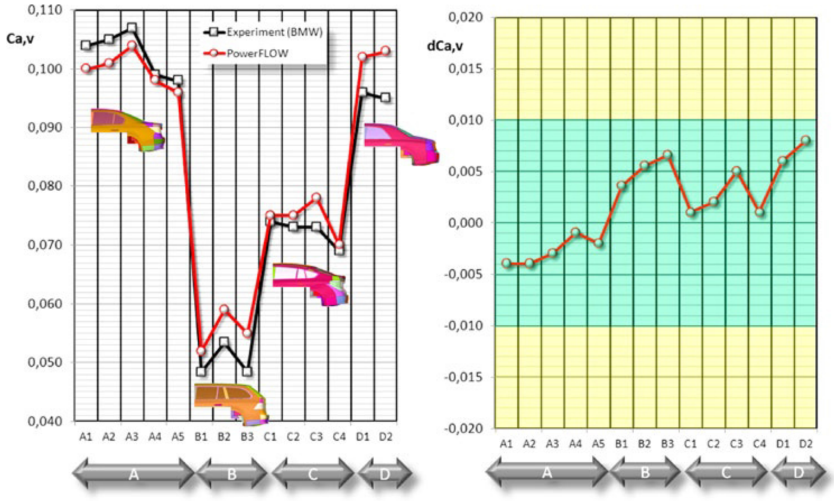


Figure 6.18: Comparison of simulations (PowerFLOW) and WT (BMW) results for lift coefficient front [82]

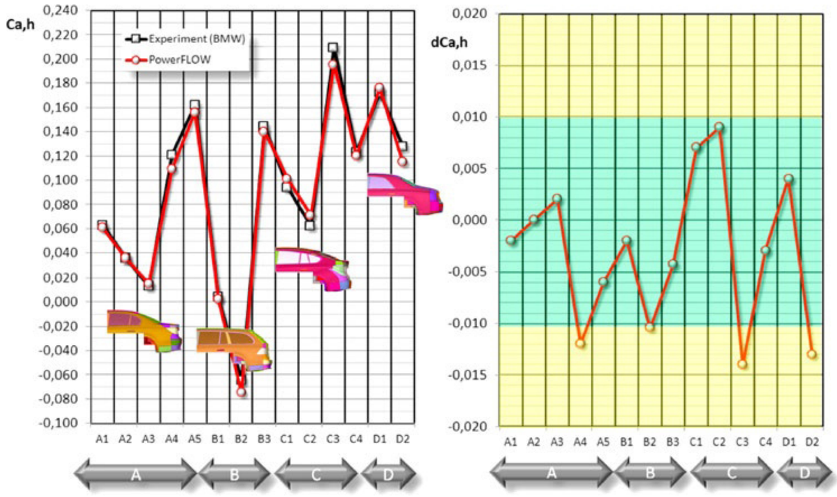
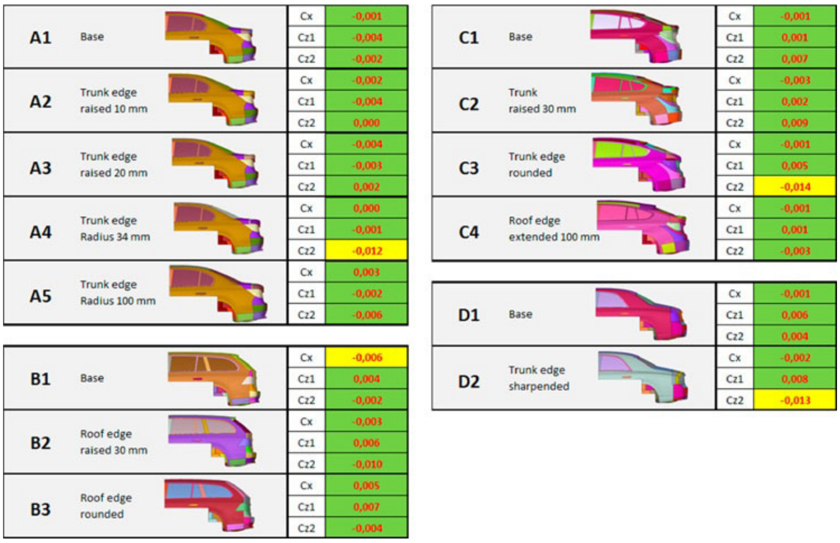


Figure 6.19: Comparison of simulations (PowerFLOW) and WT (BMW) results for lift coefficient rear [82]

The overview over all variations is shown in figure 6.20. It is remarkable, that only four of the 42 coefficients are marked with yellow (acceptable) and all others with green (good). This proofs that LBM is a very powerful tool for the aerodynamic development process in automotive engineering. [133] Tables 6.9 and 6.8 compare the results of the estate and the notchback configuration with the experimental results. Further insights into this investigation can be found in the original paper [82].



A1	Base	Cx	-0,001
		Cz1	-0,004
		Cz2	-0,002
A2	Trunk edge raised 10 mm	Cx	-0,002
		Cz1	-0,004
		Cz2	0,000
A3	Trunk edge raised 20 mm	Cx	-0,004
		Cz1	-0,003
		Cz2	0,002
A4	Trunk edge Radius 34 mm	Cx	0,000
		Cz1	-0,001
		Cz2	-0,012
A5	Trunk edge Radius 100 mm	Cx	0,003
		Cz1	-0,002
		Cz2	-0,006
B1	Base	Cx	-0,006
		Cz1	0,004
		Cz2	-0,002
B2	Roof edge raised 30 mm	Cx	-0,003
		Cz1	0,006
		Cz2	-0,010
B3	Roof edge rounded	Cx	0,005
		Cz1	0,007
		Cz2	-0,004
C1	Base	Cx	-0,001
		Cz1	0,001
		Cz2	0,007
C2	Trunk raised 30 mm	Cx	-0,003
		Cz1	0,002
		Cz2	0,009
C3	Trunk edge rounded	Cx	-0,001
		Cz1	0,005
		Cz2	-0,014
C4	Roof edge extended 100 mm	Cx	-0,001
		Cz1	0,001
		Cz2	-0,003
D1	Base	Cx	-0,001
		Cz1	0,006
		Cz2	0,004
D2	Trunk edge sharpended	Cx	-0,002
		Cz1	0,008
		Cz2	-0,013

Figure 6.20: Overview of the accuracy of drag and lift coefficients [156]

Model	Estate			Notchback		
	c_D	$\Delta c_{D,abs}$	$\Delta c_{D,rel}$	c_D	$\Delta c_{D,abs}$	$\Delta c_{D,rel}$
Exp.	0.303			0.263		
LBM VLES	0.297	-0.006	-1.98 %	0.262	-0.001	-0.38 %

Table 6.8: Comparison of the drag coefficient of the estate as well as the fastback for LBM.

Model	Estate			Notchback		
	c_L	$\Delta c_{L,abs}$	$\Delta c_{L,rel}$	c_L	$\Delta c_{L,abs}$	$\Delta c_{L,rel}$
Exp.	0.0535			0.168		
LBM VLES	0.055	0.0015	+2.80 %	0.162	-0.006	+3.57 %

Table 6.9: Comparison of the drag coefficient of the estate as well as the fastback for LBM.

7 Model Specification

7.1 Used Models

The first practical part of this thesis consists of the investigation of different mesh types. The meshes are created and converted using ANSA from BETA CAE Systems (version 16.2.1). The aim is to find the most feasible mesh element type for unsteady simulation. This means, good physical as well as numerical convergence and a high accuracy at low computational requirements. For this investigation, four different vehicle segments[1] were investigated, each with six different mesh types.

- **Limousine**: A mid-size limousine sedan car in notchback configuration.
- **City car**: A four seat mid-size city car in estate tail configuration.
- **SUV**: A large SUV in estate tail configuration.
- **Sports car - Peugeot RCZ**: A small 2+2 seated sports car with a fastback. This car is the main object in this thesis. Closer investigations of the results will be presented mainly with this car. All unsteady simulations are executed exclusively with this model. For this car, also a row of experimental WT data are available to compare.

Figure 7.1: Peugeot RCZ [93]

[1] All used vehicle geometries are real commercial passenger cars. Due to a non-disclosure contract, three of the four used cars are not free for publishing any information.

7.2 Geometry

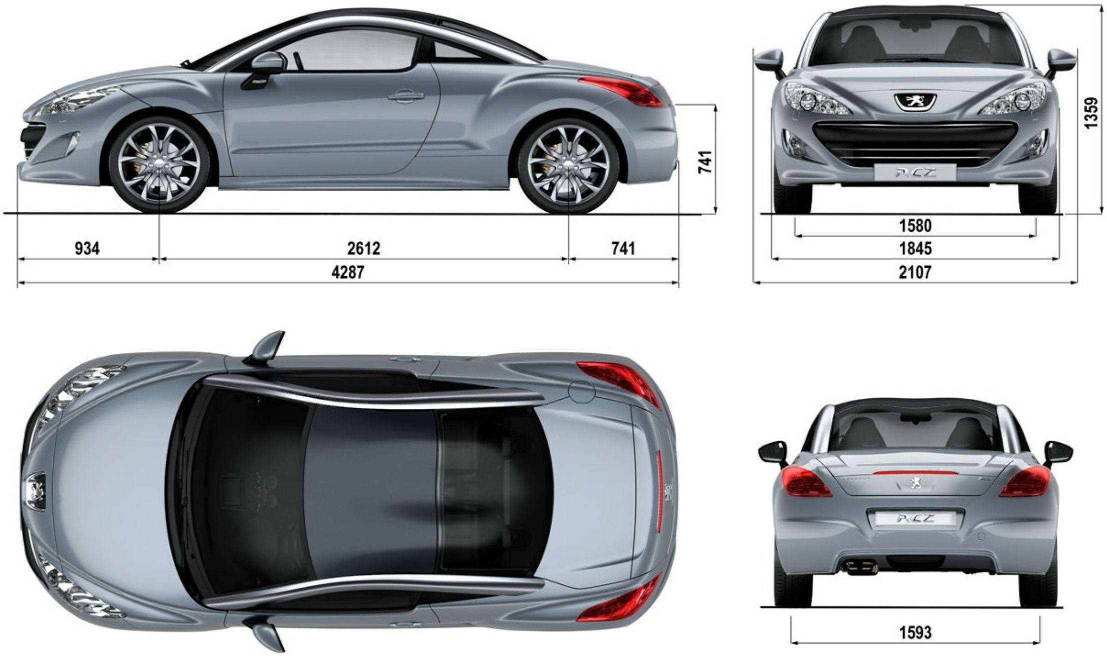


Figure 7.2: Sports car main external dimensions [27]

Apart from the sports car, all cars were provided fully meshed with a tetra volume mesh. They all included some Moving Reference Frame (MRF) zones in the fan of the cooling package as well as at the rims. Concerning this study, MRFs were not considered.
The sports car geometry had to be built up entirely from the CAD data (CATIA V5R19). All relevant CAD geometry (CATPart, CATProduct) was exported as CGR (Catia Graphical Representation) file type. Non required geometry such as pedal assembly or steering wheel was excluded from this data. After importing this data into ANSA, a quick visual check on each file was done to make sure that the export process has worked properly. In a few areas, especially front crash structure within the BIW (body in white) geometry, some errors were detected. These errors were mainly missing thickness on metal sheets. This was simply solved by applying a thickness corresponding to the neighbour geometry to these surfaces. After this, the vehicle was separated in two files, namely styling surface (upper shell) and base car (lower shell). Figure 7.3 shows the separated model of the sports car. Following this, the car was treated according to the *Magna Steyr* CFD standard.
All CFD models have the same domain size (virtual WT). Each domain includes eleven size boxes in order to refine specific area of interest (mirrors, engine, spoiler, wheels, wake, car). These refinement zones are always adapted to the car's shape. The sports car model has an additional refinement zone around the rear spoiler which is set into

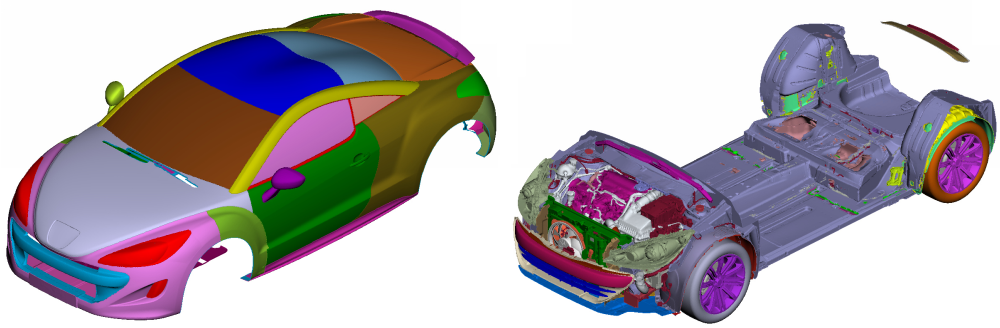


Figure 7.3: Sports car upper shell (styling surface) and lower shell (base car)

"position 1" (−19° from zero position). Figure 7.4 shows the domain of the sports car. Figures 7.5, 7.6 and 7.7 show the refinement zones for the mesh refinement in the domain of the sports car.

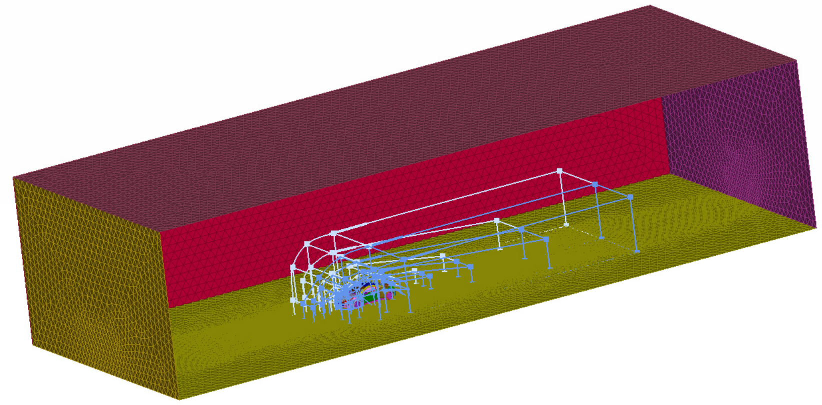


Figure 7.4: Sports car computational domain

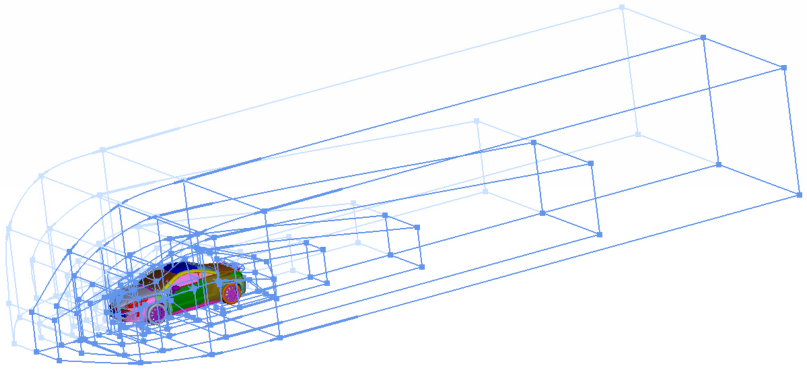


Figure 7.5: Sports car refinement zones in the computational domain

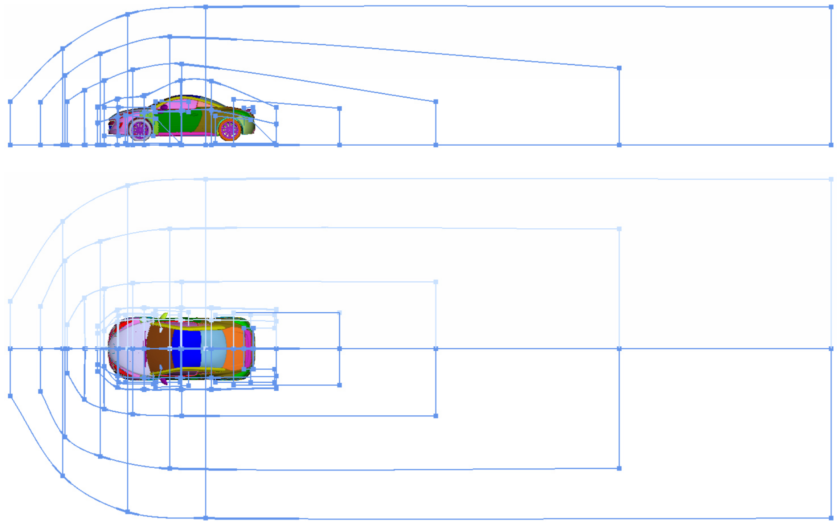


Figure 7.6: Sports car refinement zones in the computational domain side and top view

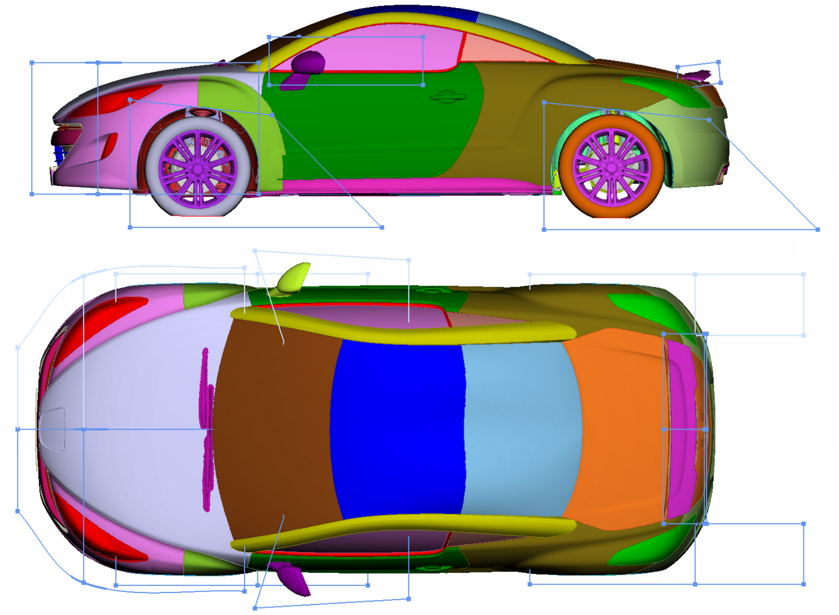


Figure 7.7: Refinement zones for the specific car areas in side and top view

7.3 Applied Volume Meshes

A Tetra mesh and two hybrid-hexa[2] meshes in HexaInterior and HexaPoly were created. Prism layers cover the viscous sublayer as well as the buffer layer. Two polyhedral meshes were converted from the Tetra mesh with ANSA Conv2Poly algorithm at each car. One poly mesh was converted including the prism layers (called PolyPoly in this thesis) and a second one was converted excluding the prism layers (PolyPrism). A third poly mesh was converted from the Tetra mesh in Fluent (PolyFluent).
Figure 7.8 and table 7.1 give an overview over all volume meshes with corresponding cell number. Following meshing algorithms were used in ANSA: Tetra Rapid, HexaInterior, HexaPoly and Conv2Poly. The PolyPoly volume mesh represents all three polyhedral conversions within figure 7.8 and table 7.1. A fully detailed listing of all four cars with element type numbers and solver run-time can be found in the appendix.

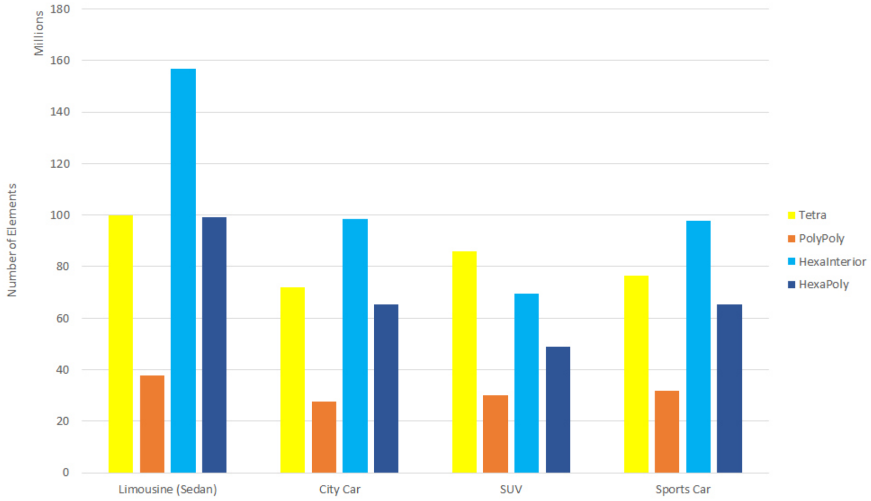


Figure 7.8: Cell count comparison of different volume mesh types

Mesh type Car type	Tetra	PolyPoly	HexaInterior	HexaPoly
Limousine sedan	99 775 374	37 679 741	156 753 500	99 259 852
City car	71 903 364	27 534 165	98 342 319	65 248 508
SUV	85 846 812	30 066 231	69 357 371	49 028 360
Sports car	76 664 114	31 669 855	97 707 188	65 477 795

Table 7.1: Comparison of different volume mesh types

[2] The ANSA hexa meshing algorithms base on a hexahedral mesh which is aligned to the global coordinate system. Tetrahedral and pyramid or polyhedral elements are applied in transition zones and domain boundaries. [51]

A comparison between all three different polyhedral volume meshes is shown in figure 7.9 and in table 7.2. The conversion from tetra to poly took between 1:30 hours and 2:45 hours. This depends on the volume mesh cell number.[3] Another conversion from tetra to poly was executed within Fluent.

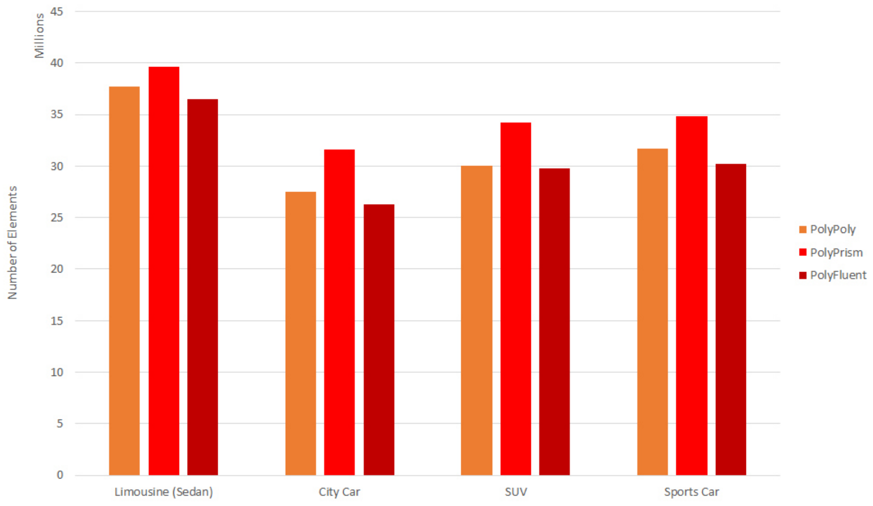


Figure 7.9: Cell count comparison of different poly volume mesh types

Mesh type	PolyPoly	PolyPrism	PolyFluent
Car type			
Limousine sedan	37 679 741	39 622 461	36 507 047
City car	27 534 165	31 578 727	26 245 306
SUV	30 066 231	34 246 637	29 762 237
Sports car	31 669 855	34 877 178	30 224 170

Table 7.2: Comparison of different poly volume mesh types

7.3.1 Tetrahedral Volume Mesh

The *Magna Steyr* standard volume meshing algorithm is the Tetra approach. All simulations were referred to the standard Tetra mesh. The intention was to create a mesh with a lower cell number than with a pure tetrahedral mesh. Figures 7.10 to 7.12 show the standard Tetra volume mesh of the sports car.

[3] The creation of a standard Tetra, HexaInterior or HexaPoly volume mesh requires between 1:00 hour and 2:00 hours computing time.

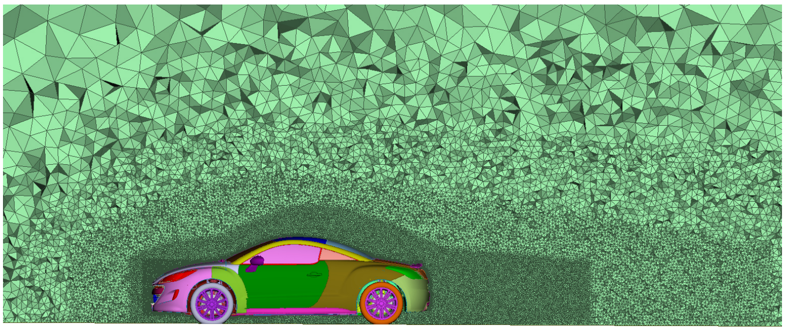


Figure 7.10: Tetra volume mesh of the sports car

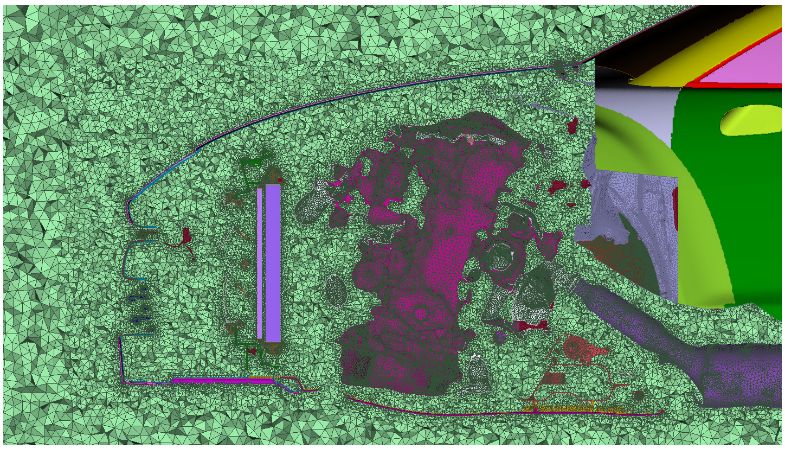


Figure 7.11: Tetra volume mesh in the engine compartment of the sports car; At the front, the cooling package including a radiator, a condenser and an intercooler (not visible in this figure) is can be seen (purple)

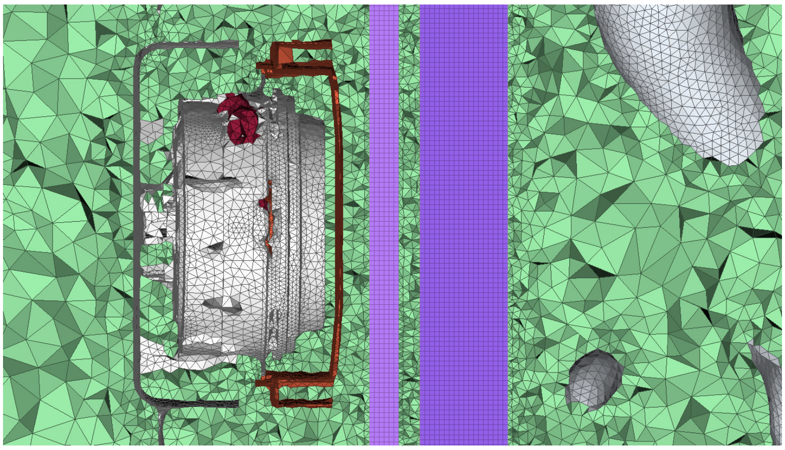


Figure 7.12: The sports car cooling package, with its oriented pentahedral volume mesh

7.3.2 Polyhedron Conversion

The base volume mesh for the polyhedron conversion was the Tetra volume mesh. Unsurprisingly, the poly meshing algorithm creates by far the lowest cell number. Poly volume meshes are created by the Conv2Poly algorithm of ANSA with each setting, including (PolyPoly) and excluding (PolyPrism) the prism layer (this option can be chosen within the Conv2Poly -> whole DB window or by selecting just the fluid main property). Another polyhedron conversion was done by the Fluent (PolyFluent). The PolyPoly and the PolyFluent conversion deliver pretty much the same cell number, even if the algorithm is different. The PolyFluent algorithm converts the whole domain (including cooling package and layers) into a polyhedral mesh. The PolyPrism conversion delivers a considerably higher cell number than the PolyPoly or the PolyFluent algorithm. Polyhedral elements are by definition much larger than the pentahedral volume elements in the original prism layer.

The polyhedron conversion process in Fluent starts with connecting face centroids with edge centroids of the corresponding face. This is exemplified in figure 7.13. Within each volume cell, new faces are created by connecting the corresponding cell centroid with the new edges on each face. Thus, each tetrahedral cell is divided into four subvolumes. Furthermore, these new faces establish the boundaries of the new polyhedral element. Hexahedral elements are not converted, except they join with a neighbouring non-hexahedral element. [4] The conversion algorithm within ANSA bases on a similar approach.

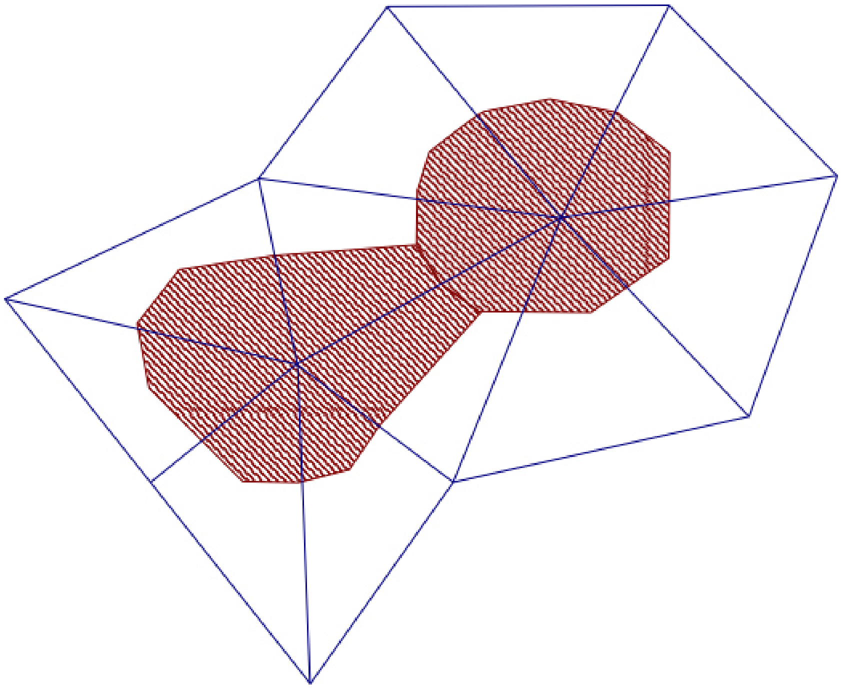


Figure 7.13: Polyhedral conversion - connection of edge centroids with face centroids [4]

A polyhedral volume mesh is illustrated in figure 7.14. The difference between the PolyPoly and the PolyPrism mesh is illustrated in figure 7.15. The PolyFluent mesh looks the same as the PolyPoly mesh. The prism layers are also converted into polyhedral elements. However, any polyhedral volume mesh has a considerable lower cell number than the standard Tetra mesh.
Interestingly, the Conv2Poly conversion algorithm causes fewer problems than for example a hybrid (HexaInterior, HexaPoly) mesh. Mesh quality is usually very good after converting to poly. In most cases, zero off elements appear in ANSA after converting to poly. Even though, in Fluent the mesh check reports mostly some problematic elements in the poly mesh. This requires a repair (Fluent command: mesh/repair-improve/repair).

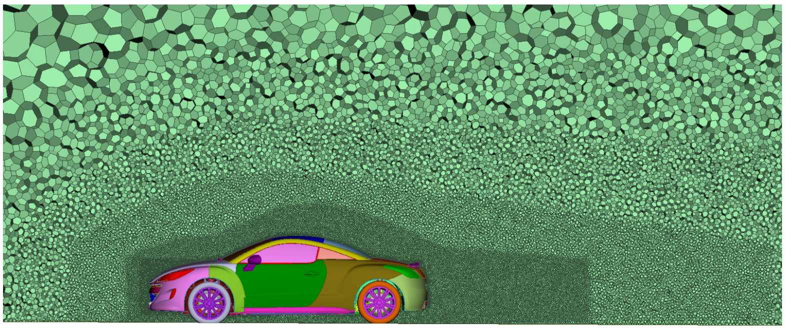


Figure 7.14: PolyPoly volume mesh of the sports car

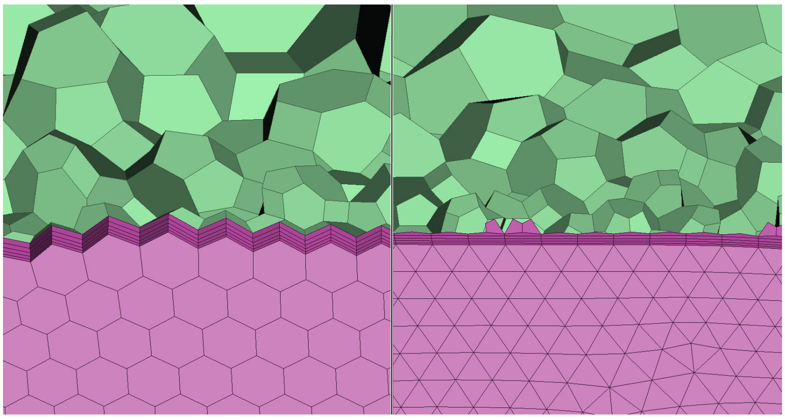


Figure 7.15: Layer mesh of the PolyPoly volume mesh (left) and of the PolyPrism volume mesh (right)

When converting to a polyhedral volume mesh, the selection of the domain has to be done very carefully. The Conv2Poly feature has three different options. "Whole DB" is the first option in the drop down menu. It is probably the worst option to choose, as this algorithm just converts the whole database. By using this option, it can be chosen to include or exclude the prism layer, but the porous media in the radiators, which is meshed with pentahedrons, will also be converted. The second option is the "select" option. This is a bit better, as the volume which should be converted can be chosen manually. The most flexible option is for sure the "visible" option. Within the property identification (PID) tab, all volumes which need to be converted can be made visible.

Difficulties in Generating Large Polyhedral Meshes

A polyhedral mesh, it does not matter which type (PolyPoly, PolyPrism, PolyFluent), has always about a third of the cell number of the base Tetra mesh from which it was converted. To increase the number of polyhedral cells, in the hope to get a more accurate result and better convergence, two attempts on the sports car were made to realise this. The PolyPrism mesh was chosen for this approach, as it showed the best results of all poly meshes. Three different meshes were created - a coarse one (77m cells), which is the standard approach, a medium (136m cells) and a fine one (218m cells). Table 7.3 shows the full cell numbers, for each mesh and each type of element. Also the file sizes are presented for the zipped ANSA files (.ansa.gz). The refinement was only applied to the prism layer shell mesh. The standard approach has a minimum and maximum shell element length of 2 mm and 4 mm, respectively. The medium one has 1.5 mm and 3 mm and the fine one has 1 mm and 2 mm, respectively. The test model was the sports car.

Mesh type	Tetra		Size	PolyPrism		Size
Element type	tetras	pentas	[GiB]	polys	pentas	[GiB]
Coarse (Std)	62 368 702	14 295 412	1.5	23 224 775	11 652 403	5.2
Medium	92 788 856	43 469 952	3.1	-	-	-
Fine	137 798 594	79 712 767	5.1	-	-	-

Table 7.3: Comparison of Tetra and PolyPrism meshes with different resolutions

The poly conversion from the standard Tetra mesh did not cause any problems. The creation of the medium and fine Tetra meshe was also less problematic, but very time consuming. The conversion of the medium (93m tetras to convert) and fine (138m tetras to convert) Tetra mesh, in contrast, was strongly problematic (ANSA crashed). After an email conversation with an engineer from BETA CAE Systems, it got clear, that it could be problematic to convert the fine Tetra mesh with the machines used at *Magna Steyr* (HP Z820). The limiting element would be the RAM. The machines used at *Magna Steyr* have a RAM of 189.1 GiB. According to BETA CAE Systems, the conversion of a 215m cells mesh (170m tetras to convert) builds up RAM and fills swap till eventually ANSA crashes due to a lack of RAM.[4] A conversion of a 160m cells mesh, with 120m tetras to convert, needed slightly over 120 GiB RAM according to BETA CAE Systems. However, the Z820 with its 189.1 GiB RAM should be able to convert the medium Tetra mesh, but in no case the fine Tetra mesh. [76] The fine Tetra mesh, in contrast, could be converted in the cluster with Fluent.[5] But this converts the whole domain, including cooling

[4] The test machine BETA CAE Systems used for this test had a RAM of 256 GiB, compared to the 189.1 Gib of the machines used at *Magna Steyr*.

[5] The *Magna Steyr* in-house CFD cluster has a RAM of 75 GiB per node, which is equivalent to 14.4 TiB overall.

package volumes and prism layer into polyhedral cells. Anyway, the pentahedrons in the prism layer are so small at the fine Tetra mesh (1/4 of the volume of the standard mesh), that the created polyhedrons are about the same size than the pentahedrons in the standard (coarse) mesh. At the time, this thesis was handed in, neither the fine, nor the medium Tetra mesh could be converted into PolyPrism.

7.3.3 HexaInterior Volume Mesh

Referring to figure 7.8 and table 7.1, it is clearly visible, that the HexaInterior algorithm creates by far the highest cell number. This is due to the fact, that this meshing algorithm creates tetrahedral or pyramidal elements at each boundary. This is at the prism layer, but also the boundaries between each refinement zone. Figures 7.16 and 7.17 illustrates this transition zones. Three buffer layers (means tetrahedral and pyramidal element layers) are applied by default setting (can be adjusted). So at each boundary are three layers full of tetrahedrons before changing to hexahedral elements. As at least eleven refinement boxes are used in all models, there are ten or more large areas where three layers of tetrahedral and pyramidal elements are created (refer to figure 7.5). This results in a cell number which is about 30 per cent higher than with the pure tetrahedral meshing algorithm.

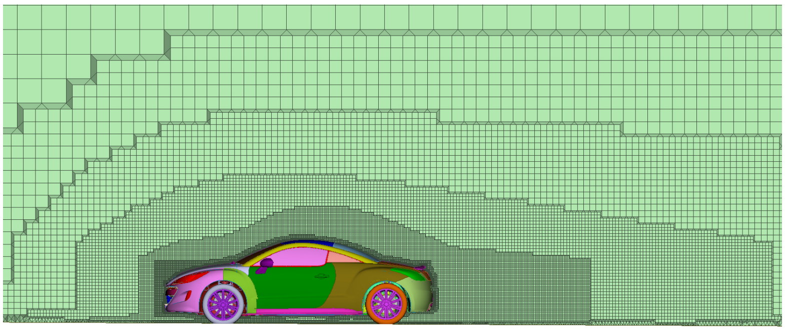


Figure 7.16: HexaInterior volume mesh of the sports car

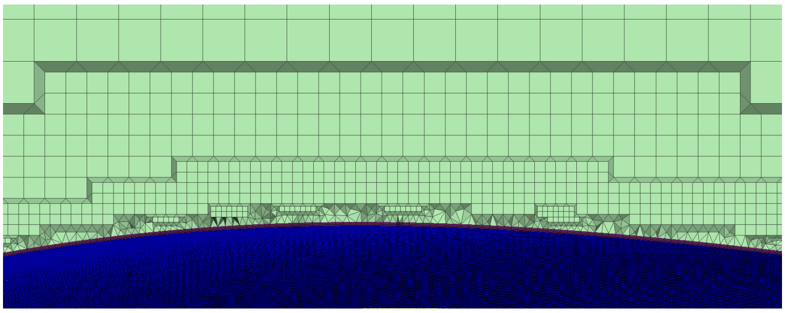
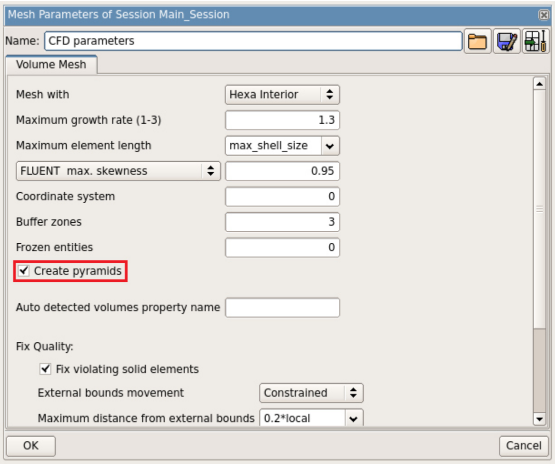


Figure 7.17: Transition between hexahedral and tetrahedral elements within the HexaInterior algorithm

"Create Pyramids" Option

The HexaInterior algorithm offers the option to disable "Create pyramids" (see figure 7.18). This option has just use in some finite element solvers that do not support pyramids. By disabling this, it generates problems in Fluent. The reason for this can be found in the geometric conditions of the hexahedrons. By disabling the "Create pyramids" option, the volume generator will not create any pyramids at all. But normally in HexaInterior, a quad face of a hexa element will be connected to a pyramids base. With the "Create pyramids" option disabled, the volume generator will create two tetrahedrons instead. This means, that this mesh is not conformal, as at a quad face of a hexahedral element, two triangular faces of the tetrahedral elements are generated. This is call a "hanging edge" in that region. The elements are actually not connected. During output process, these facets are in fact free and ANSA generates a shell mesh to cover these facets. This shell mesh is then placed in a PID called "default-exterior". In Fluent, this PID is a wall and it is not possible to change it to interior. So when using the HexaInterior meshing algorithm for Fluent, the "Create pyramids" option has to be enabled. A big advantage comes with the usage of pyramids. The hexahedral elements in the core of the domain are actually cubes. Pyramids which are attached to these cubes have a base aspect ratio of one and a height relative to the length of the base. Consequently these pyramids are of very high quality. [76]



Name: CFD parameters

Mesh with	Hexa Interior
Maximum growth rate (1-3)	1.3
Maximum element length	max_shell_size
FLUENT max. skewness	0.95
Coordinate system	0
Buffer zones	3
Frozen entities	0
☑ Create pyramids	
Auto detected volumes property name	

Fix Quality:

☑ Fix violating solid elements

External bounds movement	Constrained
Maximum distance from external bounds	0.2*local

Figure 7.18: "Create pyramids" option within the HexaInterior algorithm

7.3.4 HexaPoly Volume Mesh

The HexaPoly algorithm creates volume mesh cell number, which lies between the Tetra and the PolyPrism volume mesh. The HexaPoly meshing algorithm is very similar to the HexaInterior algorithm. At each boundary, polyhedral elements are applied with again three buffer layers. As polyhedral elements are by definition a lot larger than tetrahedral elements, the cell number is considerably lower than with the HexaInterior approach and also up to 15 per cent lower as with the standard tetrahedral meshing algorithm. Figures 7.19 and 7.20 shows the HexaPoly mesh of the sports car.

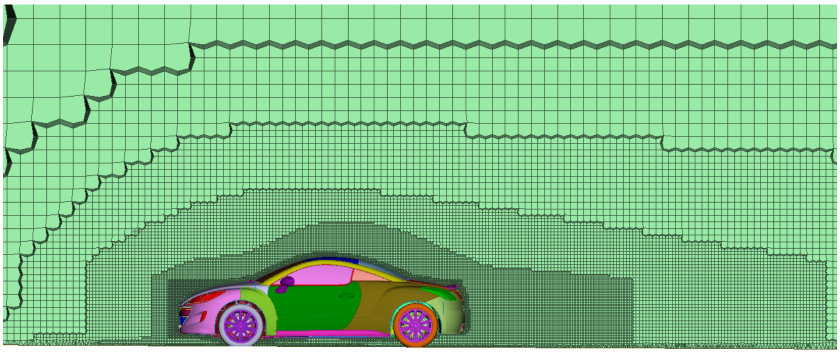


Figure 7.19: HexaPoly mesh of the sports car

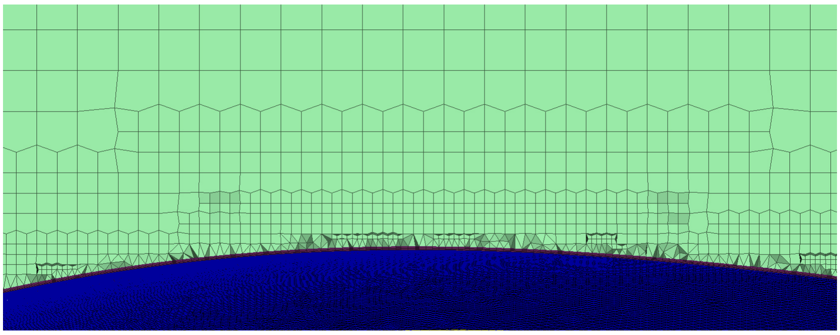


Figure 7.20: Transition between hexahedral and polyhedral elements within the HexaPoly algorithm

7.4 Used Hardware

It is worth mentioning, that most of the mesh generating work was executed using a HP Z820 workstation with 189.1 GiB RAM (random access memory) and 16 processors (Intel®Xeon®CPU E5-2639 v2), each at 3.50 GHz clocking. It is possible to create three meshes parallel (Tetra, HexaInterior, HexaPoly). However, it is not recommended to create more than two meshes at the same time as ANSA could crash if RAM and processor are in use for 100 per cent (which is almost the case when generating three meshes parallel) in use for a longer time period. A tetrahedral mesh for a full car takes about one to one and a half hours computing time. This depends on the cell number. The hybrid meshes (HexaInterior, HexaPoly) take about an hour more. The HexaPoly takes a bit less time than the HexaInterior. At the limousine sedan car, the HexaPoly mesh took almost two and a half hours to create (longest computing time for a mesh excluding Conv2Poly). The polyhedral meshes (PolyPoly, PolyPrism) take between one and a half and three hours. But to convert a polyhedral mesh, first a Tetra or a HexaInterior mesh needs to be created. So to create a polyhedral mesh, it takes at least two and a half to four hours for a car with low cell number.

A few tries were executed using a HP Z440 workstation with 62.8 GiB RAM and eight processors (Intel®Xeon®CPU E5-1620 v4), again each at 3.50 GHz clocking. Usual work with ANSA is no problem with a Z440. Creating a volume mesh is possible, but not two at the same time like with the Z820 (obviously, the Z820 has three times more RAM than the Z440). A poly conversion, does not matter how large the cell number is, is not possible with a Z440 for a full car. A Conv2Poly conversion was tried with a Z440 for a 100m (limousine) as well as a 77m (sports car) cell mesh, but both times ANSA crashed.

Both machines, Z820 as well as Z440, are equipped with a Linux operating system. So all pre-processing and simulation work was done using a Linux system. One try was executed, meshing the sports car (layers as well as Tetra volume mesh) using ANSA running on Windows 7 operating system. This was done on a Dell Precision 7510 laptop with 16.0 GiB RAM and a Intel®Corei7-6820HQ processor at 2.7 GHz clocking. However, the RAM was at the limit, but Tetra volume mesh creation run within just 32 minutes. With no other than ANSA resource sucking application in use, the laptop is good to use ANSA.

7.5 Summary

A few summarising words to conclude this chapter. It can be seen, that the PolyPoly mesh delivers farly the lowest cell count number. The PolyPoly, which converts the whole domain including the prism layers and excluding the cooling package, delivers a cell number which is considerably lower than with the basic Tetra algorithm (about minus 63 per cent). Also the PolyPrism, which keeps the prism layers, delivers a lower cell number than the Tetra algorithm (up to minus 60 per cent). The cell number of the PolyPrism is a bit higher than with the PolyPoly algorithm. PolyFluent is comparable to PolyPoly. However, its cell number is slightly lower. The HexaInterior algorithm increases the cell number intensively compared to the basic Tetra algorithm. The intention of testing different mesh types was to decrease the number of volume cells and therefore the computational requirements. The HexaPoly algorithm decreases the cell number compared to the Tetra algorithm. A HexaPoly mesh lies always between a Tetra or a PolyPrism mesh (a save compared to Tetra mesh of around 15 per cent). By summarising the results above, it seems pointless to simulate the HexaInterior mesh because of its extremely high cell number. However, simulations were done with all mesh types.

8 Solver Settings

8.1 Steady-State RANS Simulation

The simulation settings for the steady-state RANS simulations were all taken from the stand*Magna Steyr* script. For the sports car no script was available. The input file script for the solver was taken from the SUV and adapted to the sports car. The simulations were carried out on the *Magna Steyr* in-house CFD computer cluster with 192 cores. The software used for the steady-state simulations was Fluent R15.0.

All simulations were executed with the same settings. As turbulence model, the k-omega SST model was used. The fluid was set to air with a constant density $\rho = 1.20279\,\text{kgm}^{-3}$ and a constant viscosity $\mu = 1.83343 \cdot 10^{-5}\,\text{kgs}^{-1}\text{m}^{-1}$.

Velocity inlet was set to $38.89\,\text{ms}^{-1}$(=140 kph) at a turbulent intensity of 1% and a turbulent viscosity ratio of 5. For the sports car, this is equal to a Reynolds number of about 10.9m based on its overall length. The porosities of the cooling package were taken from former simulation files. In contrast to the wheels and rims, the fan of the cooling package was not equipped with moving walls.

Pressure-velocity couping was applied with a flow Courant number of 50 and explicit under-relaxation factors for momentum and pressure of 0.5.

A realizable k-epsilon model was applied for the first 50 iterations to stabilise the simulation residuals TKE k, specific rate of dissipation ω and turbulent viscosity ν_t. For near-wall-treatment, the enhanced-wall-treatment option was chosen. Table 8.1 shows the under-relaxation factors for the four simulation steps. In sum, 3 000 (city car, SUV, sports car) and 6 000 (limousine) iterations were applied on each simulation. In chapter 9, the number of iterations is detected, where the cx[1] value converges.

	iterations	model	k	ω	ν_t
step 1	50	ke-realizable	0.4	0.4	0.7
step 2	50	kw-SST	0.4	0.4	0.7
step 3	150	kw-SST	0.7	0.7	0.9
step 4	2750	kw-SST	0.8	0.8	1.0

Table 8.1: Under-relaxation factors for the four simulation steps

All the solver settings were applied by a script file. However, at one model it had been tried to change some mesh surfaces from "wall" to "interior" by using a TUI command in the script file. This forced the Fluent process to crash during writing the output .dat file. Thus a boundary condition type needs always be changed manually.

[1] cx is equal to c_D defined in the theory chapter. Furthermore, cz is equal to c_L.

8.2 Unsteady SAS

The SAS were started from the steady-state solution using the HexaPoly and Tetra mesh. The .cas and .dat files were read into Fluent R17.2 and the SAS settings were set. Within these files, the set-up was modified for the transient simulation.
At *Magna Steyr*, some experience in transient CFD simulation for aeroacoustics has been gathered since the mid of 2013. So there is a set-up for the SAS available for aeroacoutsics and these settings were also applied with a few modifications to the aerodynamics simulation executed within this thesis. The following paragraphs documents the full simulation settings for SAS section by section.

General A pressure-based segregated solver by using the SIMPLE (Semi-Implicit Method for Pressure-Linked Equations)[2] scheme was applied. Time was switched to transient.

Models The viscous model was switched from k-omega SST to SAS with keeping the standard model constant values. When choosing SAS, a information window appears which tells, that in conjunction with SAS turbulence model, only the Least Squares Cell Based gradient reconstruction method should be used (see table 8.2 below).

Solution Methods Following spatial discretisation settings were applied for SAS:

Spatial Discretisation	Setting
Gradient	Least Squares Cell Based[a]
Pressure	Standard[b]
Momentum	Bounded Central Differencing[c]
Turbulent Kinetic Energy	First Order Upwind
Specific Dissipation Rate	First Order Upwind

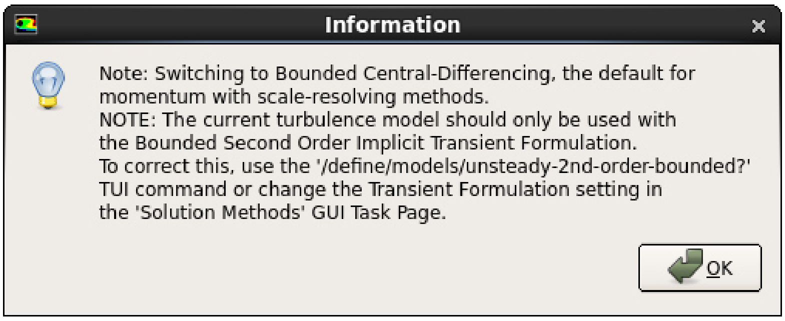
[a] Fluent signals, when using SAS turbulence model, least squares cell based gradient reconstruction method should be used.
[b] ANSYS disadvises to use a PRESTO! scheme in SRS.
[c] Fluent automatically switches to bounded central differencing discretisation, which is default for momentum with LES/DES/SAS.

Table 8.2: Spatial discretisation settings for SAS

The transient formulation was set to bounded second order implicit.[3] The checkmark for High Order Term Relaxation was set and in options window, flow variables only was checked on and relaxation factor was set to 0.75.
Most of the solution method settings are pre-defined by Fluent itself when using SAS turbulence model. Only the solver scheme (SIMPLE) and the spatial discretisations for TKE and omega (first order upwind) have to be set without a instruction from Fluent (figure 8.1).

[2] A basic explanation of solver types and the SIMPLE scheme can be found in the appendix.

[3] Fluent signals, the SAS turbulence model should only be used with the Bounded Second Order transient formulation



Note: Switching to Bounded Central-Differencing, the default for momentum with scale-resolving methods.
NOTE: The current turbulence model should only be used with the Bounded Second Order Implicit Transient Formulation.
To correct this, use the '/define/models/unsteady-2nd-order-bounded?' TUI command or change the Transient Formulation setting in the 'Solution Methods' GUI Task Page.

Figure 8.1: Fluent R17.2 information window when selecting SAS turbulence model

Solution Controls In contrast to the RANS simulations, in the solution controls section the default settings were not kept. The following under-relaxation factors were applied:

Under-Relaxation Factors	Value
Pressure	0.5
Density	1
Body Forces	1
Momentum	0.5
Turbulent Kinetic Energy	0.5
Omega	0.5
Turbulent Viscosity	0.5

Table 8.3: Under-relaxation factors for SAS

Under-relaxation factors α are included to stabilise the iterative process for segregated solvers. Equation 8.1 shows the application scheme for under-relaxation factors. ϕ represents the numerical fluxes.

$$\phi_{new} = \phi_{old} + \alpha \cdot (\phi_{calc} - \phi_{old}) \tag{8.1}$$

Decreasing the under-relaxation factor for momentum for example often helps the convergence, but elongates the computing time. The default settings by Fluent are aggressive but suitable for most problems. The *Magna Steyer* internal standard for SAS differs from the default settings.

Calculation Activities Two execute commands were set to stop the inner-iteration in case convergence is reached before the specified value of maximum iterations per time step is reached.

```
command-1:(stptmstp-resetvalues)
```

```
command-2:(stptmstp-chckcnvrg "report/surface-integrals vertex-avg
          point-1 () pressure no")
```

Run Calculation The time step size was set to $0.5 \cdot 10^{-3}$ s (in contrast to aeroacoustics simulation where the time step size is set to $0.2 \cdot 10^{-5}$ s) and the number of time steps to 2 000. This equals a real time simulation duration of one second. The maximum iterations per time step was set to 20.
By considering the CFL number using equations 5.2 and 5.3

$$1 > \frac{U_{max} \cdot \Delta t}{\Delta x},$$

where U_{max} describes the maximum velocity in the focus area, which is usually 1.5 to several times the free stream velocity, and [145] Δx describes the minimum length interval from one cell centroid to the next in the focus area. At a car, the focus area is usually the wake, where the most turbulent fluctuations appear. The SAS model uses RANS in wall-near regions, thus the important minimum length interval is in the wake, outside the prism layer.
Figure 8.2 shows the CFL plot of the transient result of the HexaPoly mesh of the sports car. In wall-near regions, CFL shows a value larger than one. As in wall-near regions RANS is active, this does not matter. However, in some minor regions in the wake, CFL is a bit larger than one. In [103], ANSYS recommends to use a CFL value of ~1. However, at the same time it is mentioned, that CFL can be larger when using SAS, but flow resolution will suffer. It is also mentioned, that CFL~5 has been used for experiments with getting satisfying results. [103] Summing up, the selected time step is satisfying.

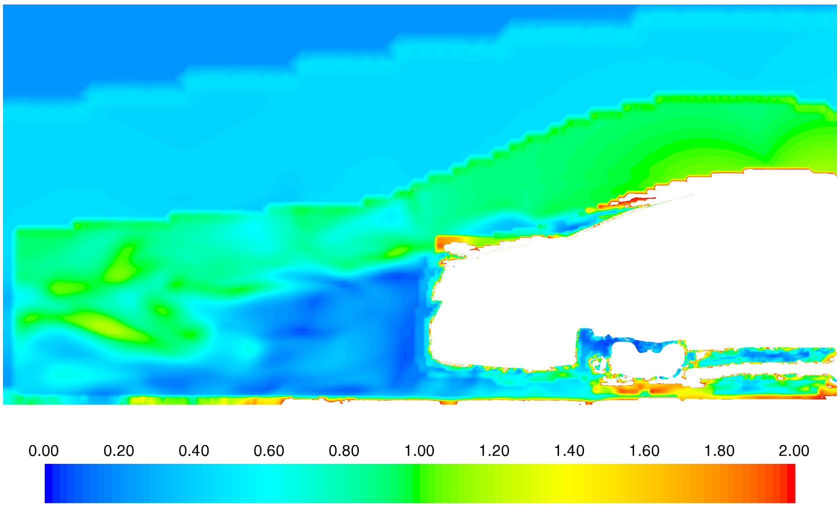


Figure 8.2: Courant number plot of the SAS result

9 Results

9.1 Steady-State RANS Simulation

All post-processing stuff for the steady-state simulations was executed using a HP Z820 workstation. The standard *Magna Steyr* script was used, which produced 210 pictures, including pressure plots (cpstat, cptot, contours, streamlines), skin friction plots and streamline velocity maps. Each in different views. The computing time for one case was between five and six hours. Three post-processing procedures for the SUV were executed using a HP Z440 workstation. But each time, the process stopped, presumably due to a RAM overload. It could be observed, that RAM was almost entirely full during the whole post-processing procedure. This post-processing was done using ANSYS CFD-Post.
Additionally, there is an automatic Excel file which evaluates all the data produced by the simulations, including cx-history (drag), cz-history (lift), cmy-history (pitch) and the data for the cooling package. Based on these, various plots, diagrams and tables were created to lead to a decision which mesh should be the most suitable[1] for a transient simulation.
Cx convergence as well as delta cx plots for for each mesh were created using Microsoft Excel. Convergence plots of the residuals (continuity, x-velocity, y-velocity, z-velocity, k, omega) was created in ANSYS CFD-Post. All cx values presented in the convergence plots are without the addition of the cooling package contribution.

9.1.1 Limousine

Mesh Type Comparison

Table 9.1 shows a comparison of all the meshes applied on the limousine car. All data is based on 6 000 iteration steps. Figure 9.2 shows the associated computing speed per mesh cell. It can be seen, that the Tetra mesh is the mesh type with the highest computational efficiency per cell element. Not too far off are the hybrid-hexa meshes. As expected, the poly elements require the highest computational power per cell. The PolyPoly mesh of the limousine was converted from Tetra including the cooling package. There might be an influence on the PolyPoly result.
For all cars, except the limousine, 3 000 iterations were applied. The limousine was simulated with additional 3 000 iterations (so 6 000 iterations overall), to get a more detailed image of convergence behaviour for all mesh types. This convergence behaviour was then compared with the 3 000 iteration convergence plots of all other cars. This is

[1] Suitable in terms of accuracy compared to the standard Tetra mesh, in terms of efficiency or computation speed and in terms of convergence speed.

intended to detect a similarity in convergence behaviour between all four cars for the same mesh type.

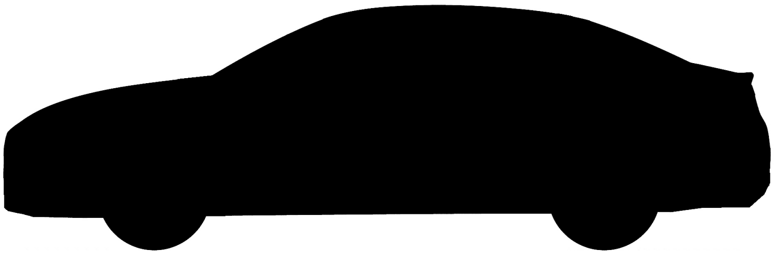


Figure 9.1: Limousine type car

Mesh type Limousine sedan	Cell number	Computing time	CPU hours
Tetra	99 775 374	07:01:43 h	1 344
PolyPoly	37 679 741	05:42:20 h	1 089
PolyPrism	39 622 461	06:04:35 h	1 165
PolyFluent	36 507 047	05:56:38 h	1 142
HexaInterior	156 753 500	12:45:39 h	2 448
HexaPoly	127 799 852	08:25:40 h	1 618

Table 9.1: Computational expenses for different mesh types on the limousine sedan car

However, poly meshes overcome this disadvantage by offering lower cell numbers at same level of accuracy, higher convergence speed and computational benefits compared to tetra as well as hexa elements. For more details refer to paper [123].

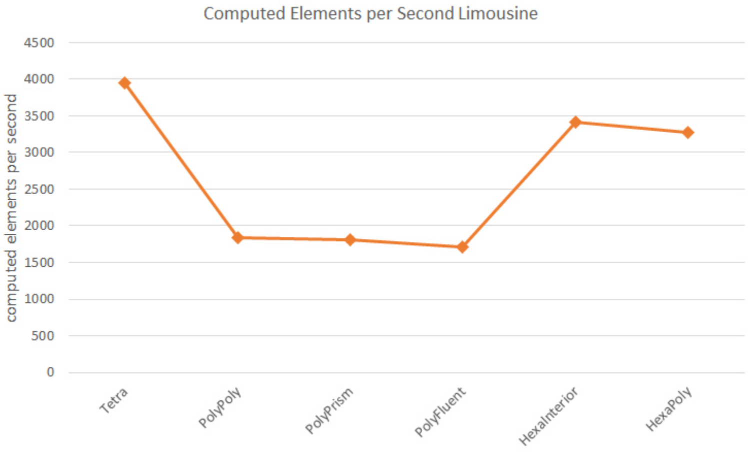


Figure 9.2: Computational efficiency of the different mesh types for the limousine

Coefficients and Convergence

Figure 9.3 shows a comparison of drag and lift coefficient between the different mesh types (average of the last 300 iterations). The Tetra mesh is the base and all other mesh types are referred to the Tetra result. The lift coefficient cz is just shown as an additional information. The focus of this thesis is on the drag coefficient. The HexaPoly mesh delivers a deviation of just minus 1.7 per cent for the limousine case. With plus 2.4 and plus 2.0 per cent, respectively, the PolyPoly and the PolyPrism are not far off. The PolyFluent and the HexaInterior are with plus 3.1 and minus 3.1 per cent, respectively, already a bit off the track. If the lift coefficient is taken into account, the HexaPoly is by far the best option in the case of the limousine.

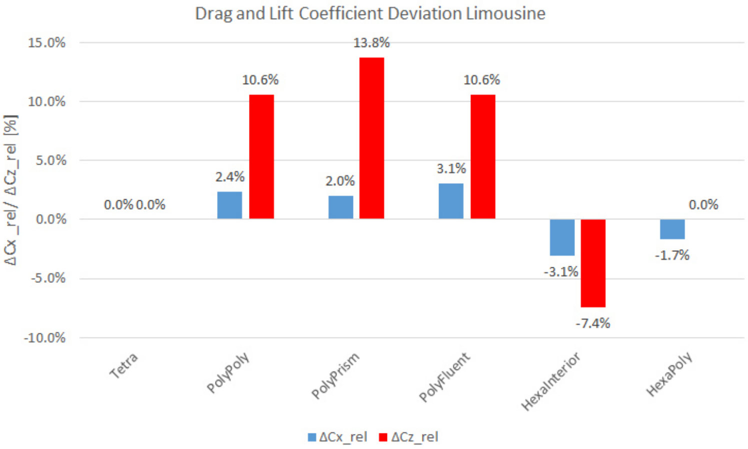


Figure 9.3: Drag and lift coefficient deviation of the limousine

Figures 9.4 to 9.15 show convergence plots of each mesh type for the limousine car. For each mesh type, a cx convergence plot and a convergence plot for the residuals are presented. On the cx plots, an averaging was done for steps of each 500 iterations, beginning at iteration 500. By showing these averaged cx values, a good impression of the convergence behaviour is given.

In the case of the limousine, the PolyFluent, HexaInterior and HexaPoly meshes show the best convergence behaviour. A summary and comparison of each mesh type for each car are given in chapter 9.1.5.

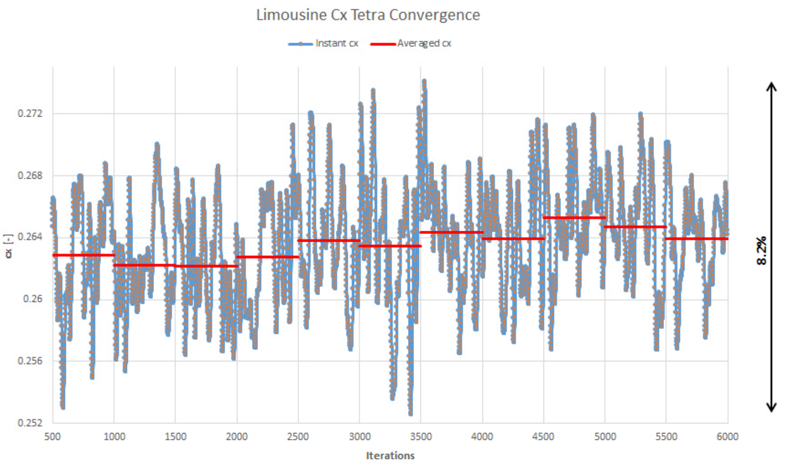


Figure 9.4: Convergence plot of cx of the limousine car Tetra mesh

Figure 9.4 shows the cx convergence plot of the Tetra mesh of the limousine. The period averaged drag coefficient stays within a range of 3.1 points (pts.) over 5 500 iterations (1.3 for the first 2 500 iterations). The maximum deviation between the lowest and the highest instant cx value is 8.2 per cent of the averaged cx value (7.2 for the first 2 500 iterations). The plot shows a clear tendency of the cx value to rise, including a slight sine wave. Figure 9.5 shows the corresponding convergence plot of the residuals. Convergence here is not reached before around 1 000 iterations. The velocity plots are very bumpy.

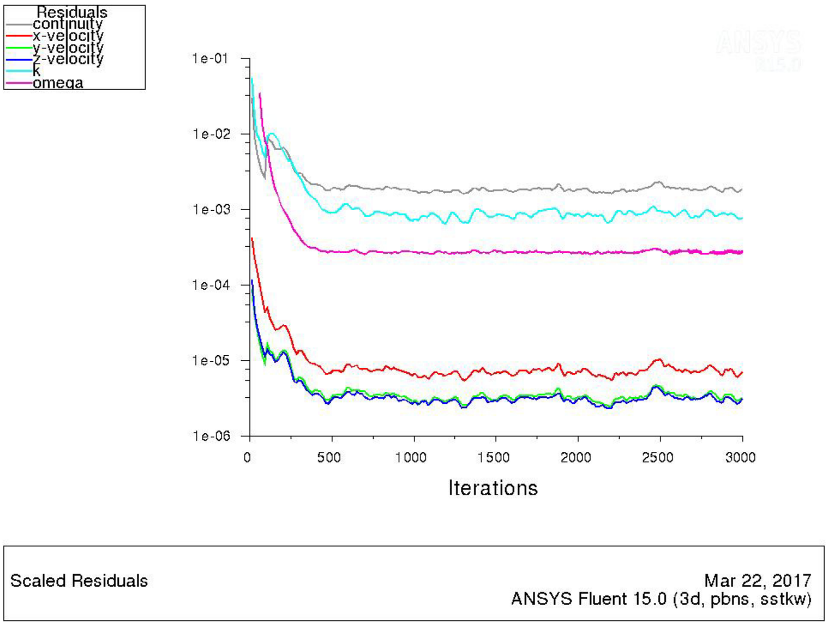


Figure 9.5: Convergence plot of the residuals of the limousine car Tetra mesh

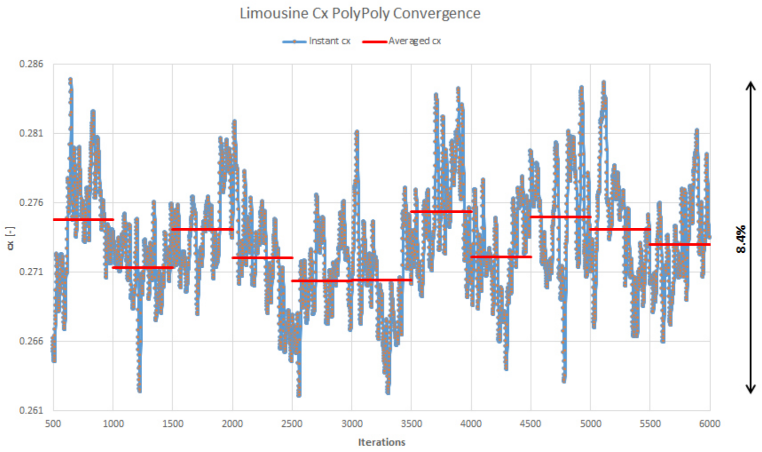


Figure 9.6: Convergence plot of cx of the limousine car PolyPoly mesh

Figure 9.6 shows the cx convergence plot of the PolyPoly mesh of the limousine. The period averaged drag coefficient stays within a range of 5.0 points over 5 500 iterations (4.4 for the first 2 500 iterations). The maximum deviation between the lowest and the highest instant cx value is 8.4 per cent of the averaged cx value. This plot shows a relatively strong oscillation, but no falling or rising tendency of the mean cx value. Figure 9.7 shows the corresponding convergence plot of the residuals. Convergence is reached much earlier compared to the Tetra mesh. After around 550 iterations, all residuals have converged. Again compared to the Tetra mesh, the velocity curves are a lot smoother.

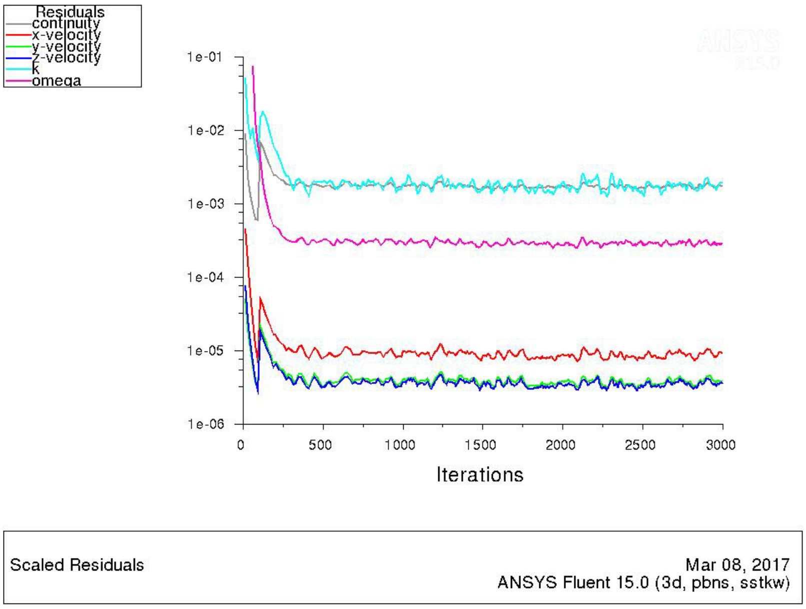


Figure 9.7: Convergence plot of the residuals of the limousine car PolyPoly mesh

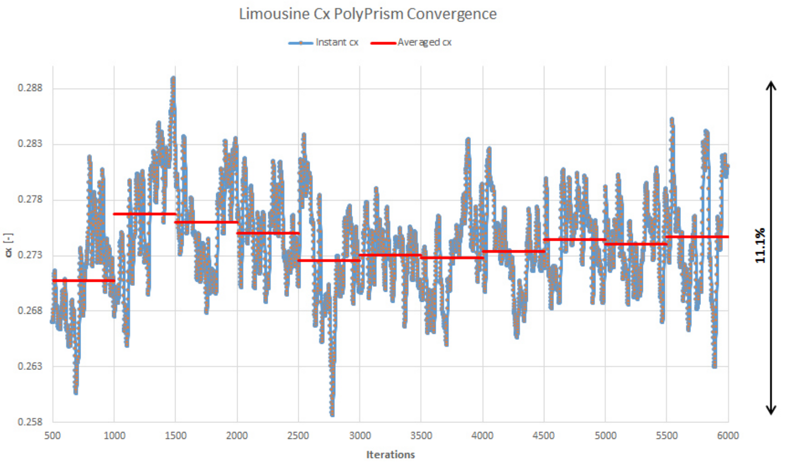


Figure 9.8: Convergence plot of cx of the limousine car PolyPrism mesh

Figure 9.8 shows the cx convergence plot of the PolyPrism mesh of the limousine. The period averaged drag coefficient stays within a range of 6.0 points over 2 500 as well as over 5 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 11.1 per cent of the averaged cx value. It looks as the cx value stabilises from about 3 000 iterations slowly but with a tendency to increase slightly. Figure 9.9 shows the corresponding convergence plot of the residuals. Convergence looks much worse compared to PolyPoly. Especially continuity, but also TKE curves, show a sine wave behaviour over the entire 3 000 iterations.

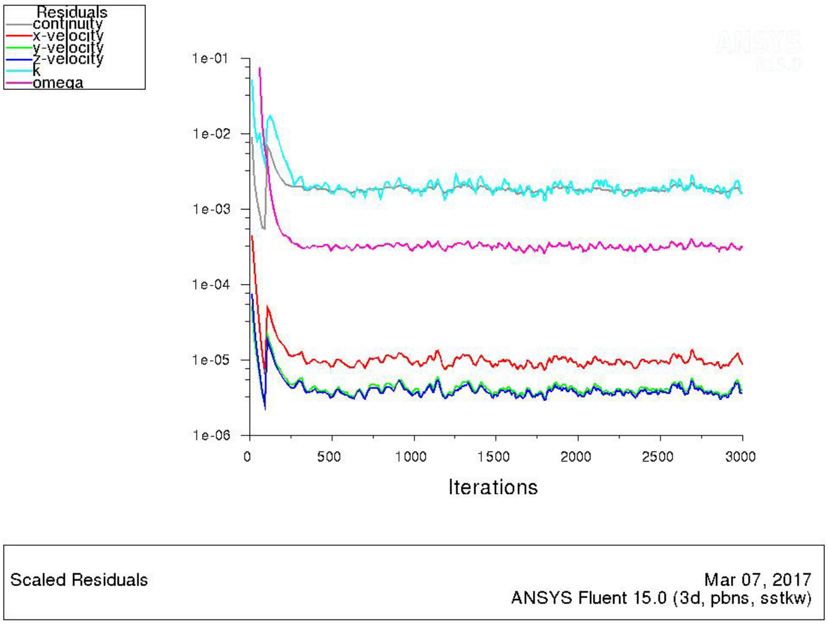


Figure 9.9: Convergence plot of the residuals of the limousine car PolyPrism mesh

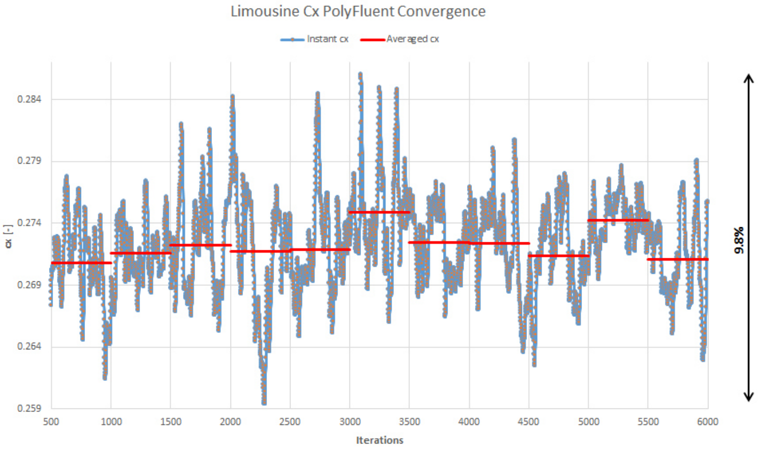


Figure 9.10: Convergence plot of cx of the limousine car PolyFluent mesh

Figure 9.10 shows the cx convergence plot of the PolyFluent mesh of the limousine. The period averaged drag coefficient stays within a range of 4.1 points over 5 500 iterations (1.4 points for the first 2 500 iterations). The maximum deviation between the lowest and the highest instant cx value is 9.8 per cent of the averaged cx value. Same as the PolyPoly mesh, the Poly Fluent mesh seems to oscillate around a mean cx value without a decreasing or an increasing tendency. Figure 9.11 shows the corresponding convergence plot of the residuals. Convergence is reached after about 900 to 1 000 iterations. The curves look a bit smoother than the PolyPrism curves, but worse than the PolyPoly ones.

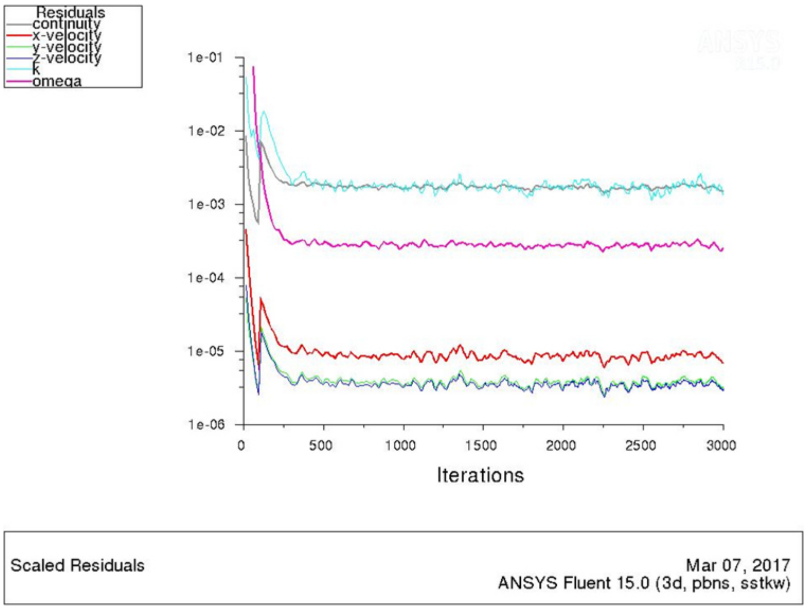


Figure 9.11: Convergence plot of the residuals of the limousine car PolyFluent mesh

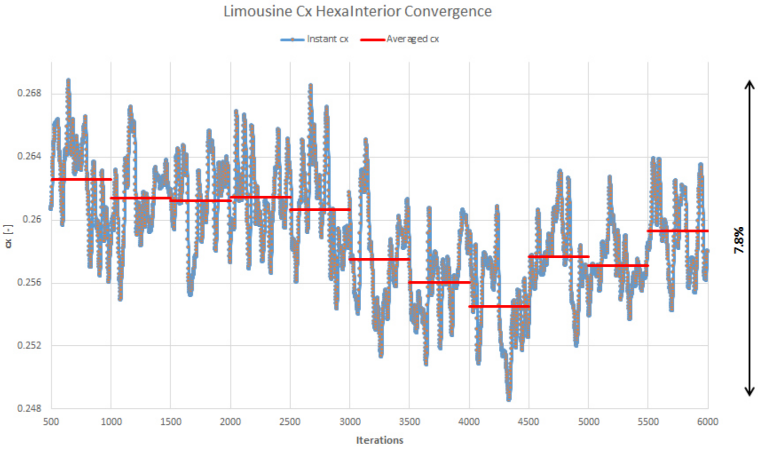


Figure 9.12: Convergence plot of cx of the limousine car HexaInterior mesh

Figure 9.12 shows the cx convergence plot of the HexaInterior mesh of the limousine. The period averaged drag coefficient stays within a range of 8.0 points over 5 500 iterations (1.9 for the first 2 500 iterations). The maximum deviation between the lowest and the highest instant cx value is 7.8 per cent of the averaged cx value (5.5 for the first 2 500 iterations). After the first 3 000 iterations a convergence was expected, but in the following 1 500 iterations there is a strong decrease of averaged cx with a subsequent increase of cx till about the level before. Figure 9.13 shows the corresponding convergence plot of the residuals. The residuals convergence of the HexaInterior looks similar as at the PolyPoly and PolyFluent, but better that the Tetra and PolyPrism convergence.

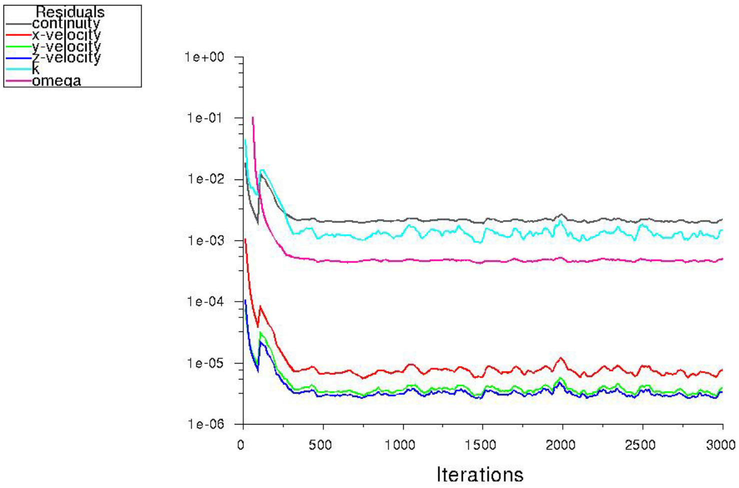


Figure 9.13: Convergence plot of the residuals of the limousine car HexaInterior mesh

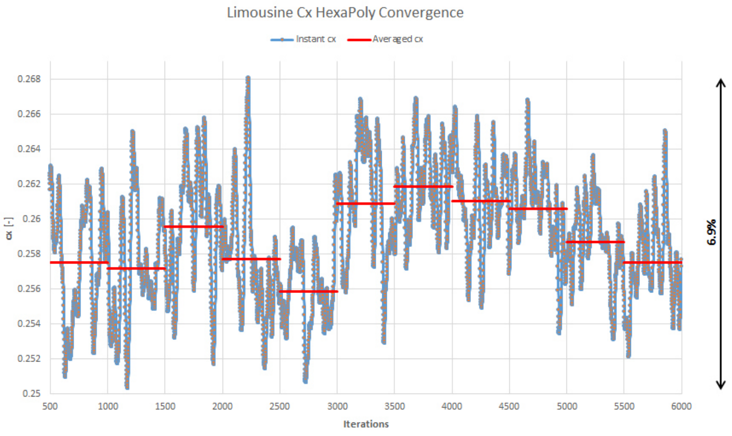


Figure 9.14: Convergence plot of cx of the limousine car HexaPoly mesh

Figure 9.14 shows the cx convergence plot of the HexaPoly mesh of the limousine. The period averaged drag coefficient stays within a range of 6.0 points over 5 500 iterations (3.7 after the first 2 500 iterations). The maximum deviation between the lowest and the highest instant cx value is 6.9 per cent of the averaged cx value (after 3 000 as well as 6 000 iterations). No clear convergence behaviour can be detected. With a FFT analysis, maybe a frequency could be detected for an oscillation around a mean cx value. Figure 9.15 shows the corresponding convergence plot of the residuals. By investigating this figure, the HexaPoly mesh shows by far the best convergence behaviour for the residuals. After about 600 iterations, the residuals have converged and especially the continuity and TKE curves are impressively smooth.

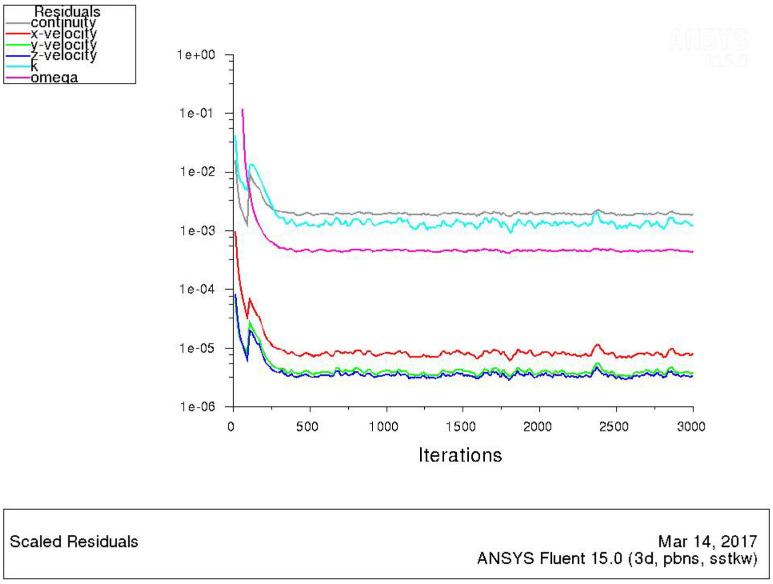


Figure 9.15: Convergence plot of the residuals of the limousine car HexaPoly mesh

9.1.2 City Car

Mesh Type Comparison

Table 9.2 shows a comparison of all the meshes applied on the city car. All data is based on 3 000 iteration steps. Figure 9.17 shows the associated computing speed per mesh cell. It can be seen, that the Tetra mesh is the mesh type with the highest computational efficiency per cell element. Not too far off are the hybrid-hexa meshes. But the difference between the HexaInterior and the HexaPoly mesh is larger than it is the case with the limousine. Again as expected, the poly elements require the highest computational power per cell. But the PolyPrism and the PolyFluent are on almost the same level, where in the limousine case, the PolyPoly and the Poly Prism were almost on the same level and the PolyFluent some counts below.

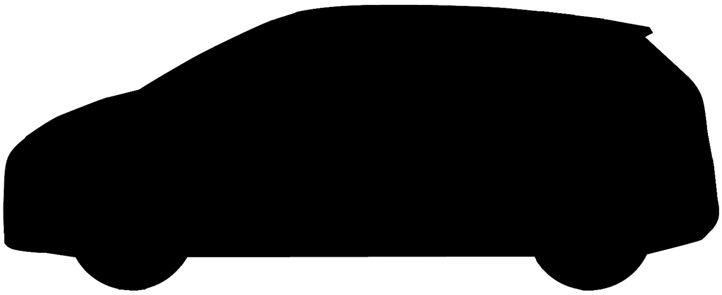


Figure 9.16: City car type car

Mesh type City car	Cell number	Computing time	CPU hours
Tetra	71 903 364	05:04:15 h	974
PolyPoly	27 534 165	04:04:51 h	784
PolyPrism	31 578 727	04:55:29 h	946
PolyFluent	26 245 306	04:04:51 h	784
HexaInterior	98 342 319	07:55:44 h	1 522
HexaPoly	65 248 508	05:34:08 h	1 069

Table 9.2: Computational expenses for different mesh types on the city car

It can be seen, that the poly meshes have the lowest absolute computational efficiency. But as stated in [123], the poly mesh types overcome this disadvantage by offering lower cell numbers at same level of accuracy, higher convergence speed and numerical benefits.

Coefficients and Convergence

Figure 9.18 shows a comparison of drag and lift coefficient between the different mesh types (average of the last 300 iterations). Again, all meshes are referred to the Tetra result. The lift coefficient cz is just shown as an additional information. The focus of

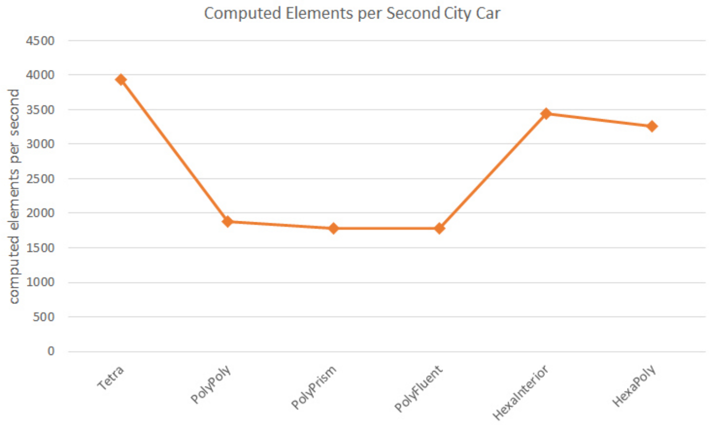


Figure 9.17: Computational efficiency of the different mesh types for the city car

this thesis is on drag coefficient. The HexaInterior and the HexaPoly mesh delivers a deviation of plus 1.3 and plus 2.0 per cent respectively for the city car. A bit off with minus 3.4, 3.7 and 3.0 per cent, all poly meshes deliver pretty much the same result. As it was the case at the limousine car, again the hybrid-hexa meshes deliver the best results.

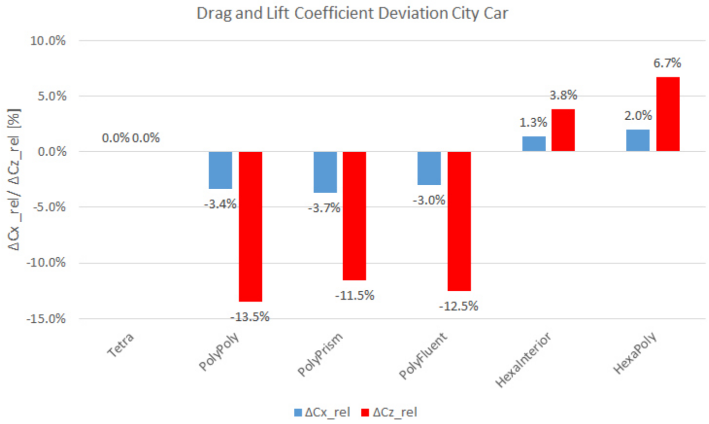


Figure 9.18: Drag and lift coefficient deviation of the city car

Figures 9.19 to 9.30 show convergence plots of each mesh type for the city car. For each mesh type, a cx convergence plot and a convergence plot for the residuals are shown. On the cx plots, an averaging was done for steps of each 500 iterations, beginning at iteration 500. By showing these averaged cx values, a good impression of the convergence behaviour is given.

In the case of the city car, again the HexaInterior and the HexaPoly meshes show the best convergence behaviour. A summary and comparison of each mesh type for each car are stated in chapter 9.1.5.

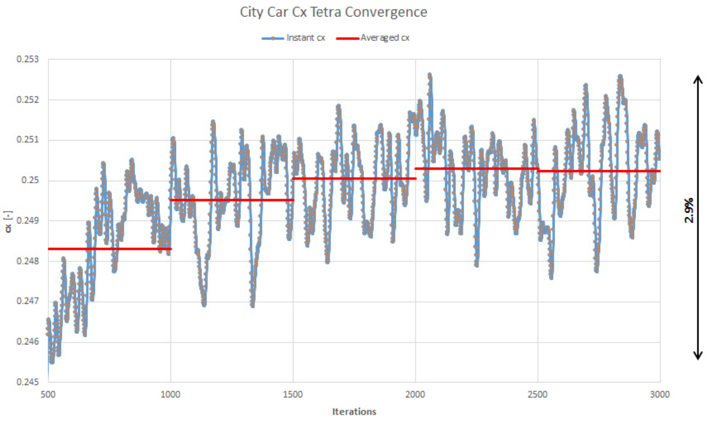


Figure 9.19: Convergence plot of cx of the city car Tetra mesh

Figure 9.19 shows the cx convergence plot of the Tetra mesh of the city car. The period averaged drag coefficient stays within a range of 2.0 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 2.9 per cent of the averaged cx value. It is clearly visible, that the Tetra mesh takes an awful lot amount of time to converge. A constant rise can be detected before cx decreases again a little between iteration 2 500 and 3 000. Figure 9.20 shows the corresponding convergence plot of the residuals. This plot shows the same as the cx convergence plot above. The Tetra mesh takes very long to converge. There can actually no point be detected, where the residuals are converged. Especially the velocity curves are pretty unstable.

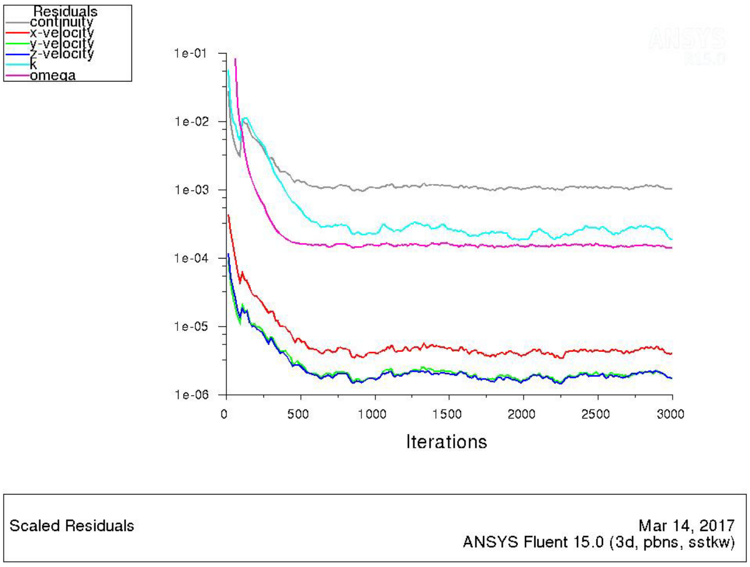


Figure 9.20: Convergence plot of the residuals of the city car Tetra mesh

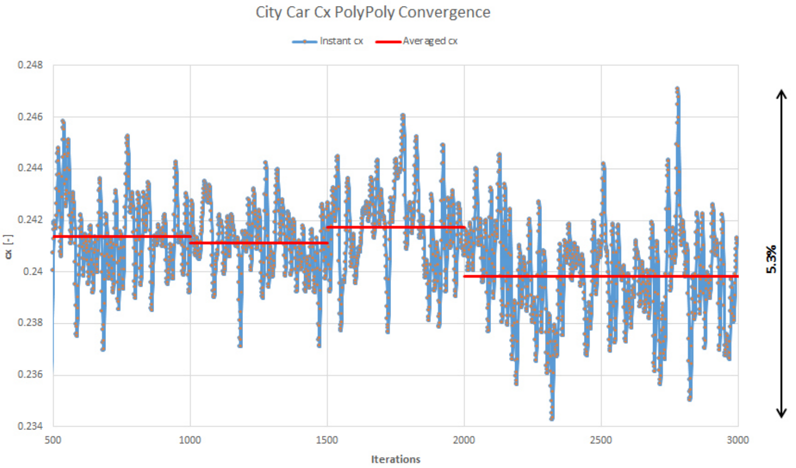


Figure 9.21: Convergence plot of cx of the city car PolyPoly mesh

Figure 9.21 shows the cx convergence plot of the PolyPoly mesh of the city car. The period averaged drag coefficient stays within a range of 1.9 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 5.3 per cent of the averaged cx value. No clear convergence behaviour is visible but a step of 1.8 points between iteration 1 500 and iteration 2 500 is shown. Figure 9.22 shows the corresponding convergence plot of the residuals. They converge after about 1 000 iterations.

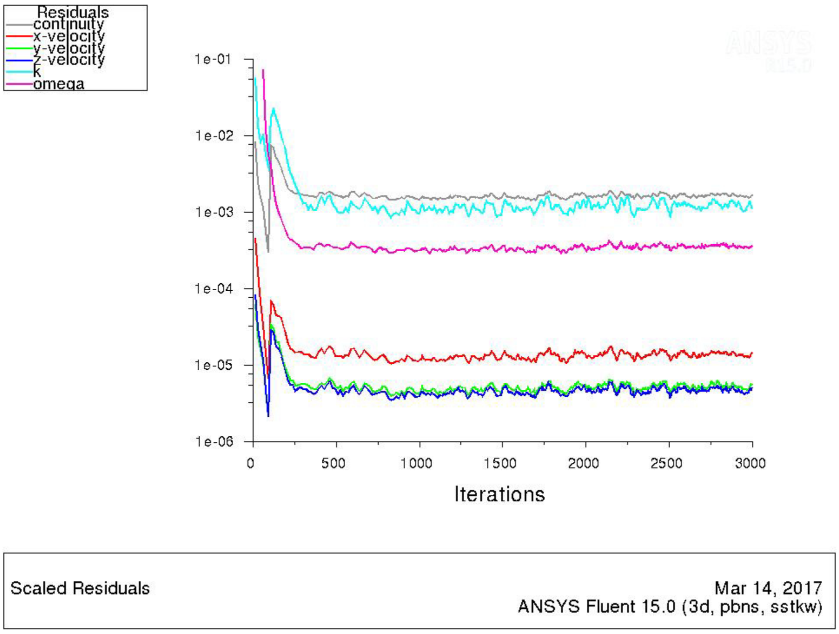


Figure 9.22: Convergence plot of the residuals of the city car PolyPoly mesh

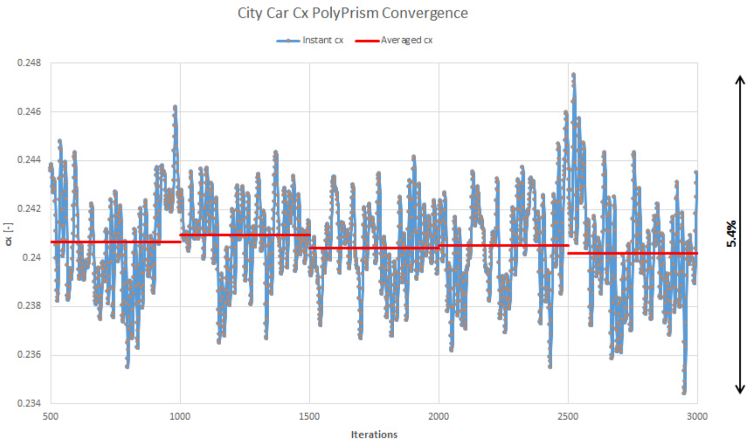


Figure 9.23: Convergence plot of cx of the city car PolyPrism mesh

Figure 9.23 shows the cx convergence plot of the PolyPrism mesh of the city car. The period averaged drag coefficient stays within a range of 0.7 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 5.4 per cent of the averaged cx value. The averaged cx shows a very slight sine wave behaviour with a tendency to decrease. When excluding the few outliers, the deviation of the instant cx value lies within just 3.3 per cent. Figure 9.24 shows the corresponding convergence plot of the residuals. The velocity curves stay pretty bumpy till the end of the simulation. But the PolyPoly mesh seems to have a better convergence for the residuals.

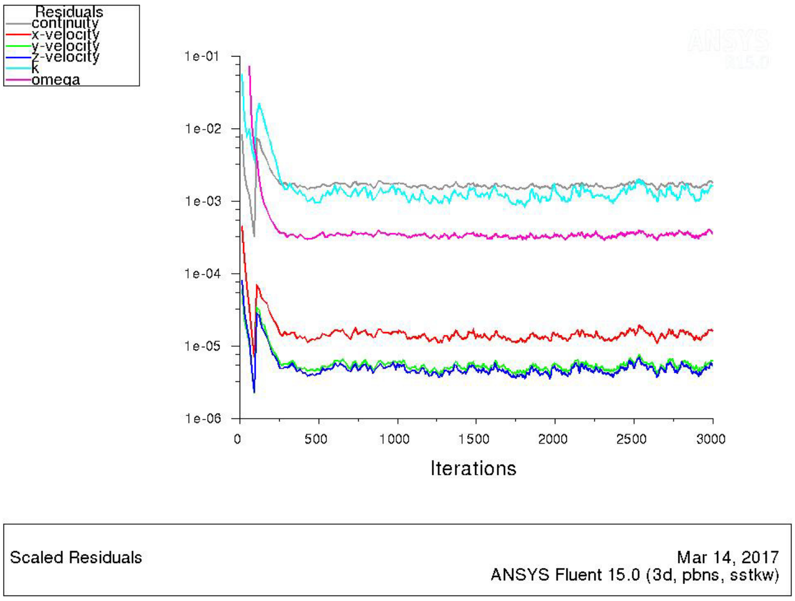


Figure 9.24: Convergence plot of the residuals of the city car PolyPrism mesh

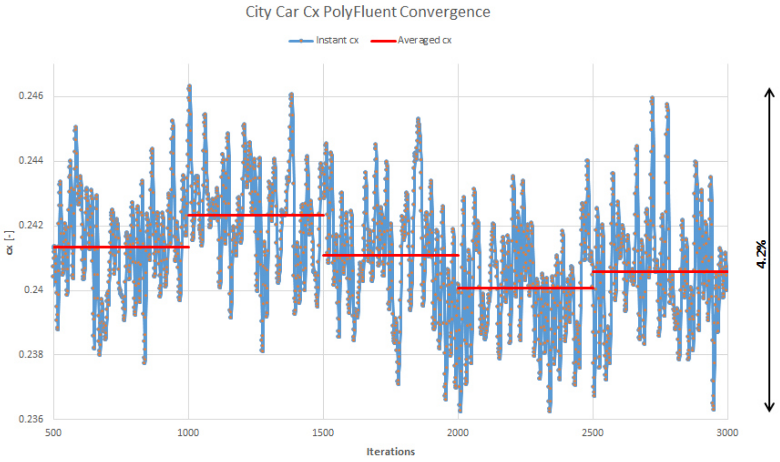


Figure 9.25: Convergence plot of cx of the city car PolyFluent mesh

Figure 9.25 shows the cx convergence plot of the PolyFluent mesh of the city car. The period averaged drag coefficient stays within a range of 2.3 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 4.2 per cent of the averaged cx value. The averaged cx shows again a sine wave behaviour. But a bit stronger than in the PolyPrism case. Figure 9.26 shows the corresponding convergence plot of the residuals. This shows the best convergence for all poly meshes for the city car. Convergence is reached after about 600 iterations and the amplitudes for all curves are relatively small.

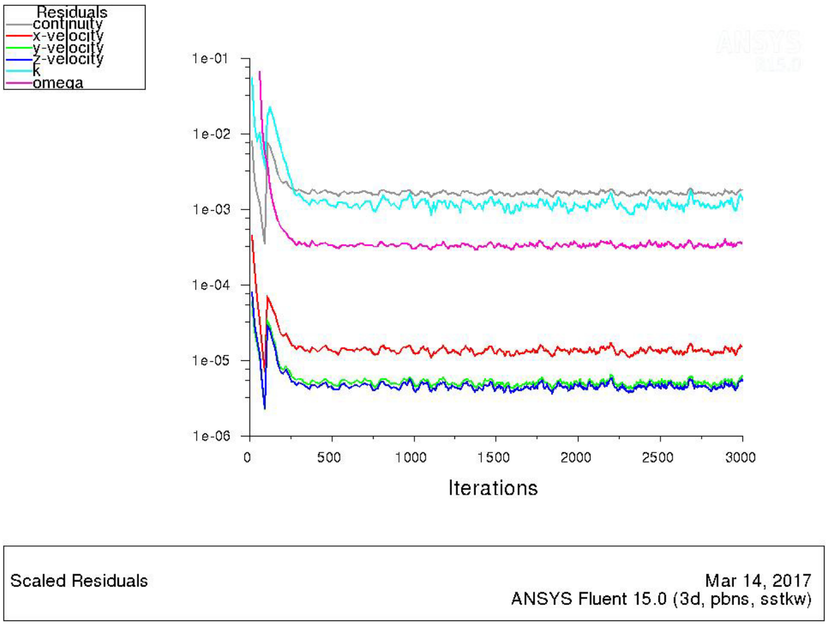


Figure 9.26: Convergence plot of the residuals of the city car PolyFluent mesh

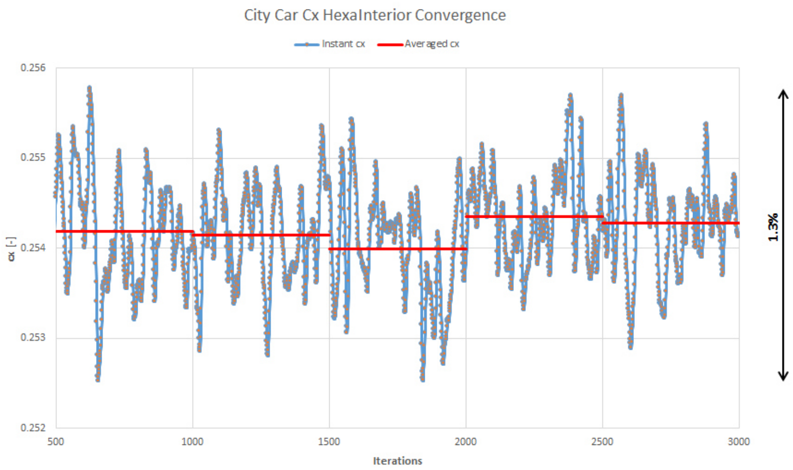


Figure 9.27: Convergence plot of cx of the city car HexaInterior mesh

Figure 9.27 shows the cx convergence plot of the HexaInterior mesh of the city car. The period averaged drag coefficient stays within a range of 0.4 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is marginal with just 1.3 per cent of the averaged cx value. The averaged cx value shows a slight sine wave behaviour. Figure 9.28 shows the corresponding convergence plot of the residuals. The convergence of the resiudals is reached a bit later than at the poly meshes. But the curves are much smoother.

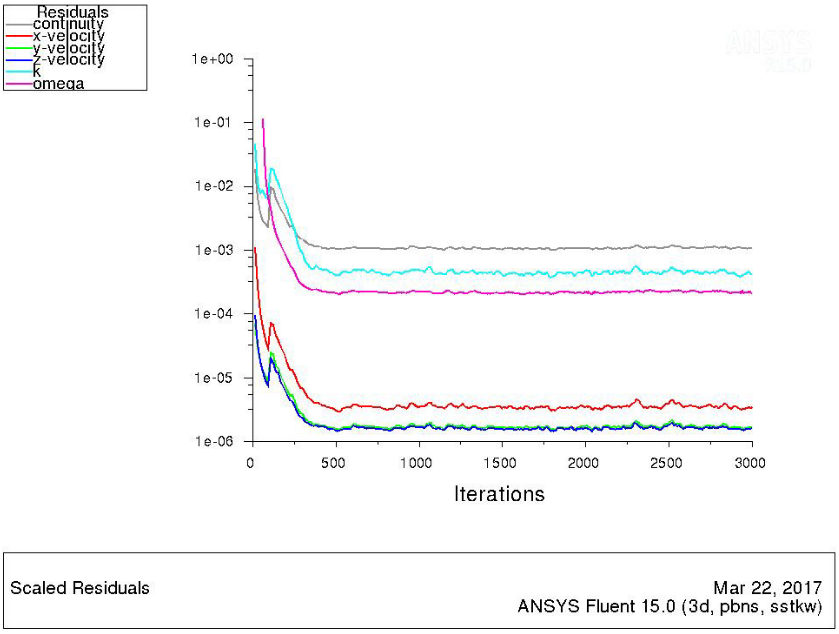


Figure 9.28: Convergence plot of the residuals of the city car HexaInterior mesh

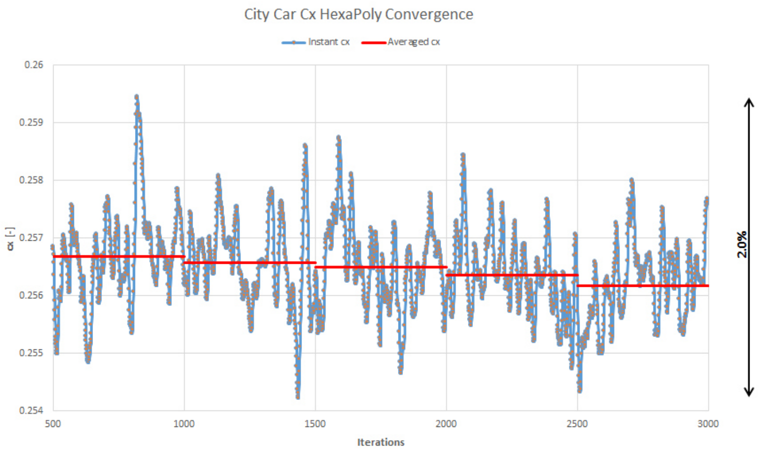


Figure 9.29: Convergence plot of cx of the city car HexaPoly mesh

Figure 9.29 shows the cx convergence plot of the HexaPoly mesh of the city car. The drag coefficient stays within a range of 0.5 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is just 2.0 per cent of the averaged cx value. The averaged cx shows a slight decrease behaviour in convergence. Figure 9.30 shows the corresponding convergence plot of the residuals. As it was the case with the limousine, the HexaPoly mesh again shows by far the best convergence for its residuals. Convergence is reached after just about 750 iterations and again all curves are incredibly smooth. At the continuity and the omega curve, almost no oscillation is visible.

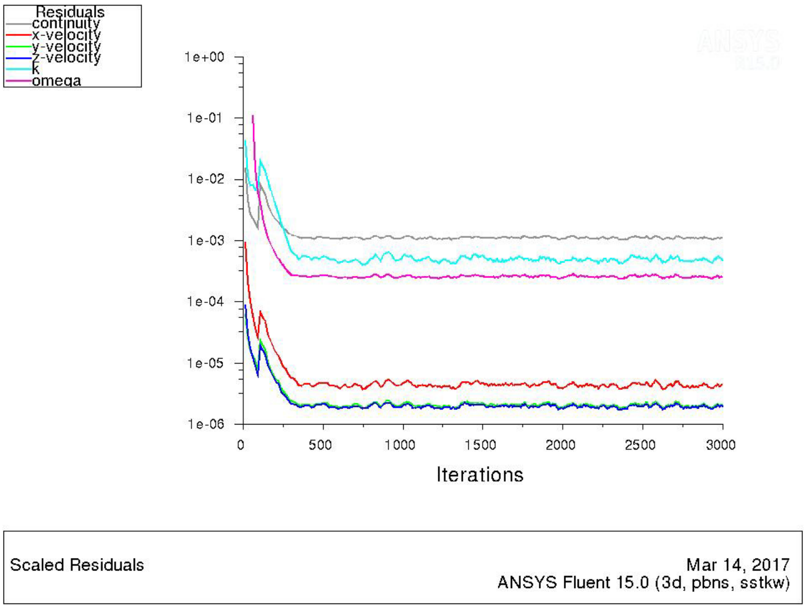


Figure 9.30: Convergence plot of the residuals of the city car HexaPoly mesh

9.1.3 SUV

Mesh Type Comparison

Table 9.3 shows a comparison of all the meshes applied on the SUV. All data is based on 3 000 iteration steps. Figure 9.32 shows the associated computing speed per mesh cell. Again, the Tetra mesh is the type with the highest computational efficiency per cell element. Both hexa-hybrid types are about on the same level and a bit below the Tetra mesh. Again decently below, all the poly mesh types, but again, the PolyFluent is the most inefficient one of the three poly types. But it should be kept in mind, that within the simulation time of the PolyFluent mesh, the converting time (from tetra to poly) is included.

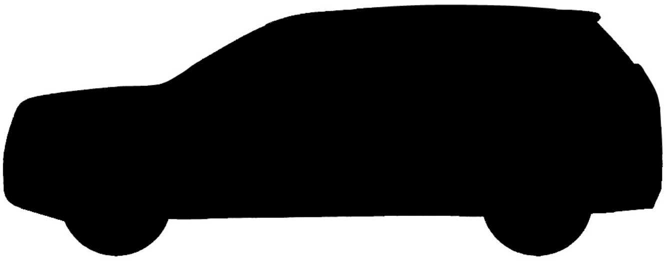


Figure 9.31: SUV type car

Mesh type SUV	Cell number	Computing time	CPU hours
Tetra	85 846 812	06:07:40 h	1 177
PolyPoly	30 066 231	04:45:36 h	914
PolyPrism	34 246 637	05:22:46 h	1 034
PolyFluent	29 762 237	06:38:45 h	1 276
HexaInterior	69 357 371	05:38:12 h	1 082
HexaPoly	49 028 360	04:06:15 h	788

Table 9.3: Computational expenses for different mesh types on the SUV

Again it can be stated, that the poly meshes have the lowest computational efficiency, but are offering a lot of other advantages compared to the other mesh types. Refer to paper [123] for more information.

Coefficients and Convergence

Figure 9.33 shows a comparison of drag and lift coefficient between the different mesh types (average of the last 300 iterations). The Tetra mesh is the base and all other mesh types are referred to the Tetra result. The lift coefficient cz is just shown as an additional information. The focus of this thesis is on drag coefficient. The PolyFluent mesh delivers a deviation of plus 1.6 per cent for the SUV. With this, the PolyFluent

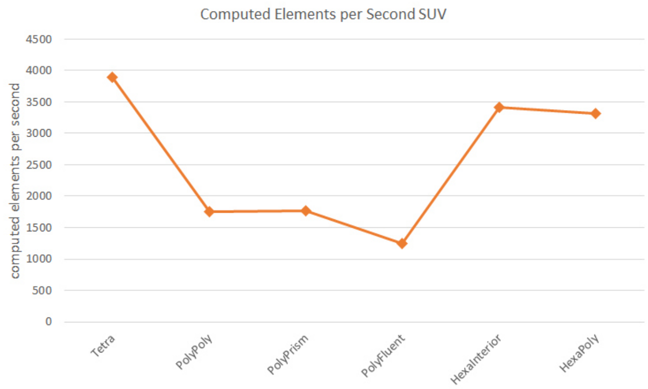


Figure 9.32: Computational efficiency of the different mesh types for the SUV

comes closest to the Tetra result. With minus and plus 2.2 per cent respectively, the PolyPoly and the PolyPrism mesh are not too far off. In contrast to the limousine and the city car, the HexaInterior and the HexaPoly deliver the worst results. But with plus 2.5 and plus 2.7 per cent, respectively, off, the hybrid-hexa meshes are not far off the poly meshes. It is conspicuously, that the PolyPrism and the PolyFluent predicts a higher cx and the PolyPoly a lower cx than the Tetra mesh. The reason for this lies presumably in the meshing approach, as at the PolyPoly mesh, also the porous media of the radiators were converted into poly. In the post processing files, some reasonable deviations were detected compared to all the other meshes on the SUV.

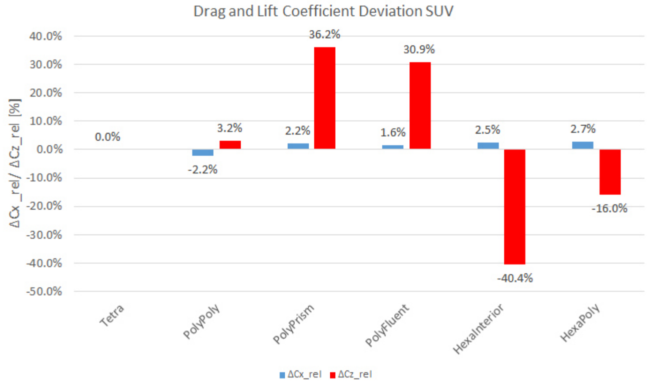


Figure 9.33: Drag and lift coefficient deviation of the SUV

Figures 9.34 to 9.45 show convergence plots of each mesh type for the SUV. For each mesh type, a cx convergence plot and a convergence plot for the residuals are shown. As with the cars above, an averaging of the cx plots was done for each 500 iterations, beginning at iteration 500. By showing these averaged cx values, a good impression of the convergence behaviour is given.

In the case of the SUV, the PolyFluent mesh shows the best convergence behaviour. A bit off are the PolyPoly and HexaPoly mesh. But all in all, the convergence behaviour on the SUV is by far the worst of all four cars for all meshes. A summary and comparison of each mesh type for each car are stated in chapter 9.1.5.

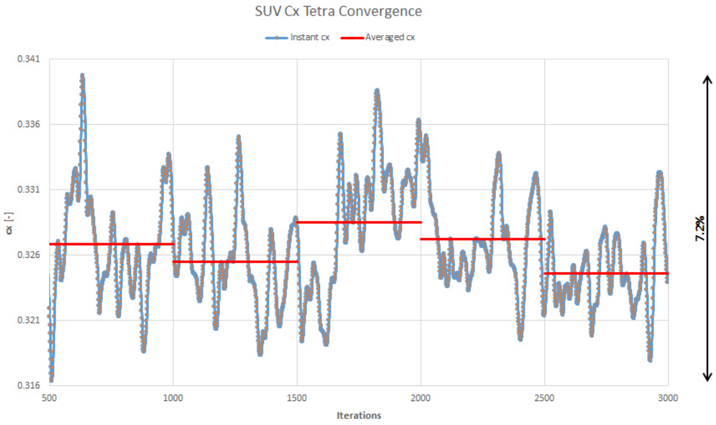


Figure 9.34: Convergence plot of cx of the SUV Tetra mesh

Figure 9.34 shows the cx convergence plot of the Tetra mesh of the SUV. The period averaged drag coefficient stays within a range of 3.9 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 7.2 per cent of the averaged cx value. There is no clear convergence behaviour detectable. Figure 9.35 shows the corresponding convergence plot of the residuals. It is pretty difficult to judge a convergence point here. But convergence is surely not reached before iteration 750.

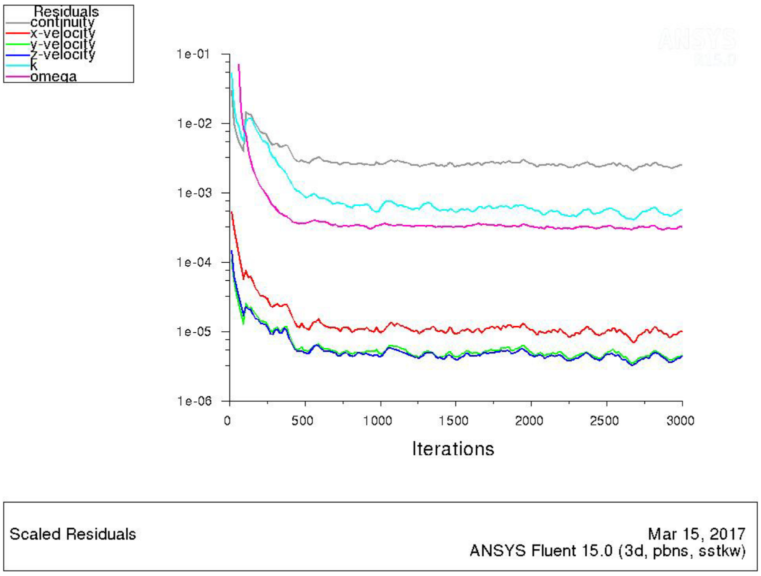


Figure 9.35: Convergence plot of the residuals of the SUV Tetra mesh

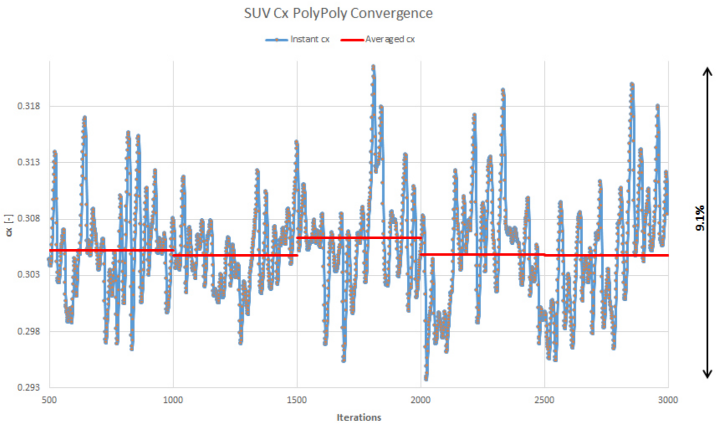


Figure 9.36: Convergence plot of cx of the SUV PolyPoly mesh

Figure 9.36 shows the cx convergence plot of the PolyPoly mesh of the SUV[2]. The period averaged drag coefficient stays within a range of 1.6 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 9.1 per cent of the averaged cx value. The averaged cx shows a good convergence behaviour towards iteration 3 000. Just very little steps between each period. However, it looks as if it behaves like a sine wave. But in general, the instant value looks relatively bumpy compared to the Tetra or the two hexa meshes. Figure 9.37 shows the corresponding convergence plot of the residuals. The residuals converge after about 1 000 iterations, but stay pretty unstable till the end of the simulation. Like in the Tetra case.

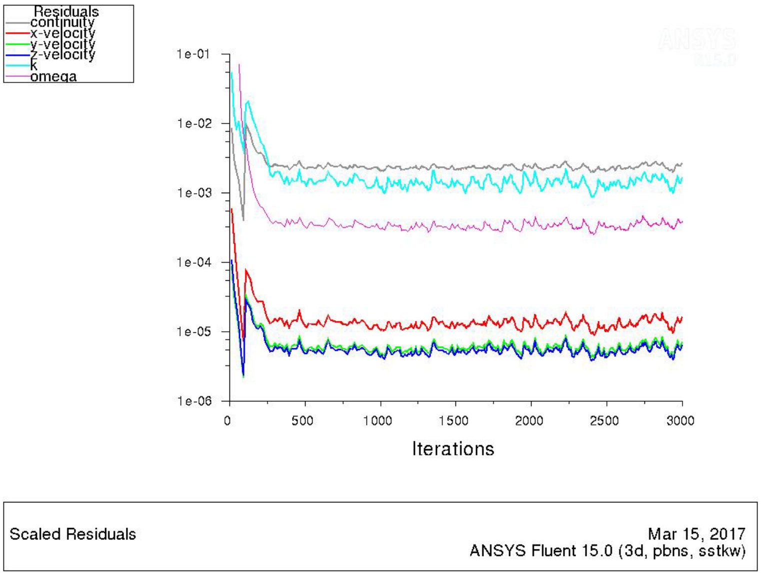


Figure 9.37: Convergence plot of the residuals of the SUV PolyPoly mesh

[2] Also porous radiator media converted into poly.

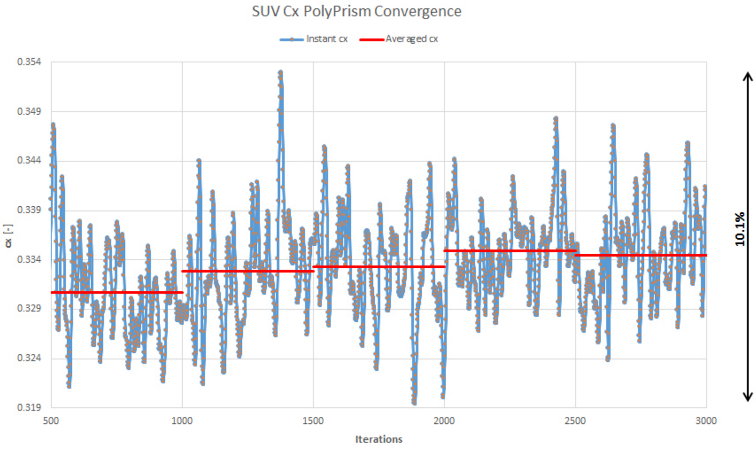


Figure 9.38: Convergence plot of cx of the SUV PolyPrism mesh

Figure 9.38 shows the cx convergence plot of the PolyPrism mesh of the SUV. The period averaged drag coefficient stays within a range of 4.2 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 10.1 per cent of the averaged cx value. The averaged cx rises constantly till the last period, where it slightly decreases. As the PolyPoly mesh, the PolyPrism shows a relatively bumpy instant cx distribution compared to the Tetra mesh or the two hybrid-hexa meshes. Figure 9.39 shows the corresponding convergence plot of the residuals. The PolyPrism residuals show about the same picture as in the PolyPoly case. Maybe with a bit earlier convergence. But again, all curves are very bumpy.

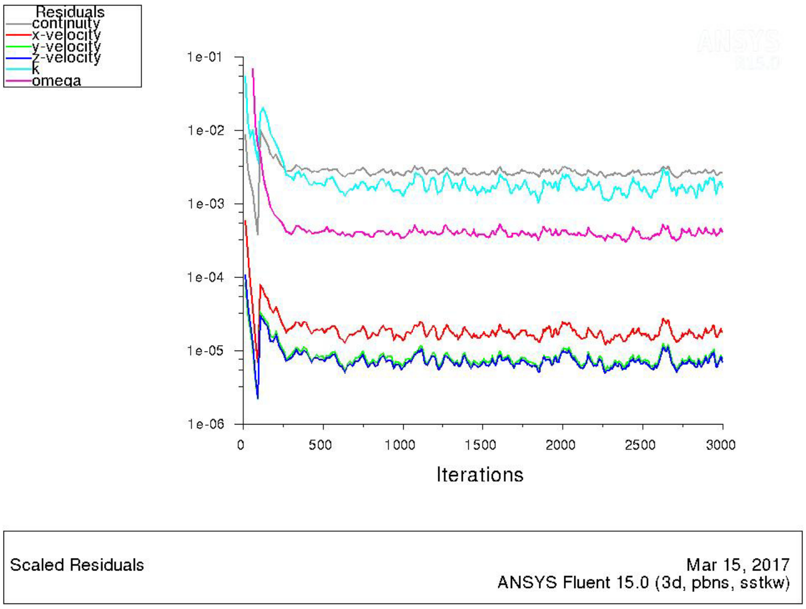


Figure 9.39: Convergence plot of the residuals of the SUV PolyPrism mesh

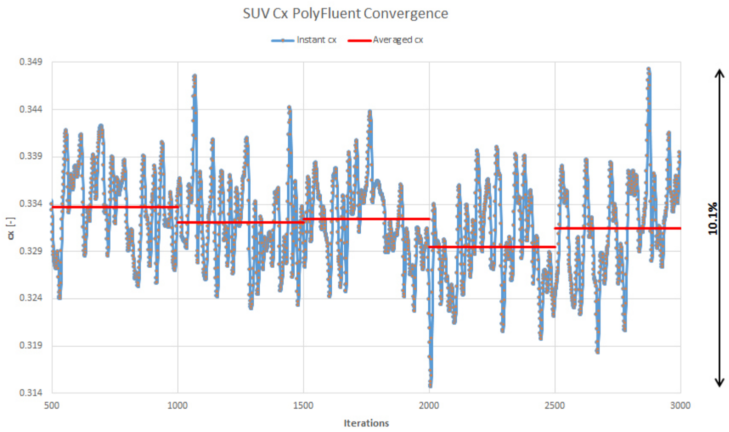


Figure 9.40: Convergence plot of cx of the SUV PolyFluent mesh

Figure 9.40 shows the cx convergence plot of the PolyFluent mesh of the SUV. The period averaged drag coefficient stays within a range of 4.2 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is at 4.2 per cent of the averaged cx value. This makes it the best poly mesh for the SUV. The averaged cx shows a sine wave behaviour with a tendency to decrease. As the PolyPoly and the PolyPrism mesh, the PolyFluent again shows a relatively bumpy instant cx distribution compared to the Tetra mesh or the two hexa meshes. Figure 9.41 shows the corresponding convergence plot of the residuals. The PolyFluent mesh shows probably best convergence of all poly meshes. The convergence takes approximately as long as at the other two poly cases, but the curves are maybe looking a bit more stable.

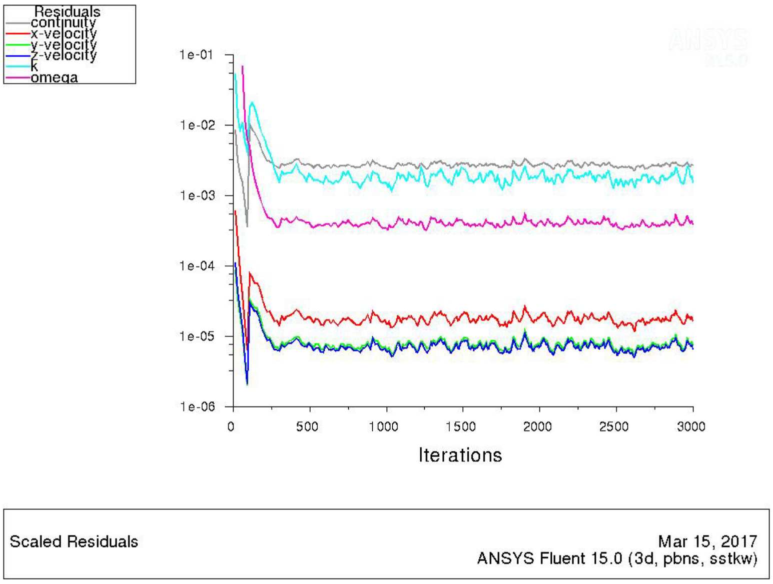


Figure 9.41: Convergence plot of the residuals of the SUV PolyFluent mesh

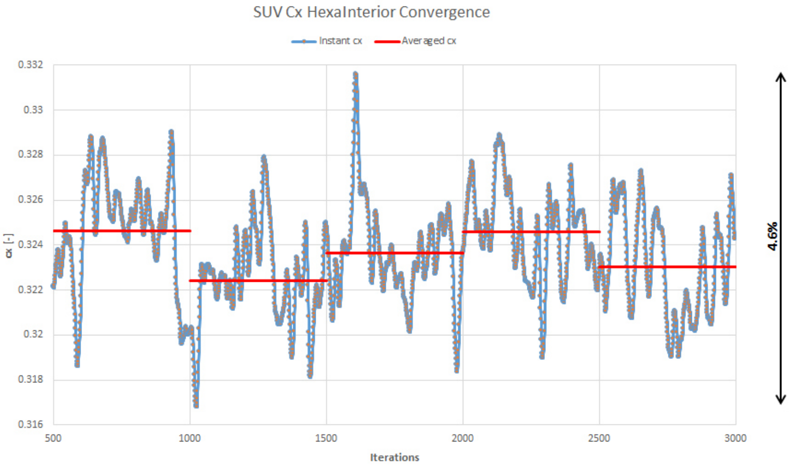


Figure 9.42: Convergence plot of cx of the SUV HexaInterior mesh

Figure 9.42 shows the cx convergence plot of the HexaInterior mesh of the SUV. The period averaged drag coefficient stays within a range of 2.2 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 4.6 per cent of the averaged cx value. The averaged cx shows a sine wave behaviour. Figure 9.43 shows the corresponding convergence plot of the residuals. The HexaInterior shows a pretty impressive convergence behaviour for the residuals when compared to the Tetra and various poly meshes for the SUV. Convergence is reached after about only 600 iterations and the curves look as stable as no other mesh at the SUV.

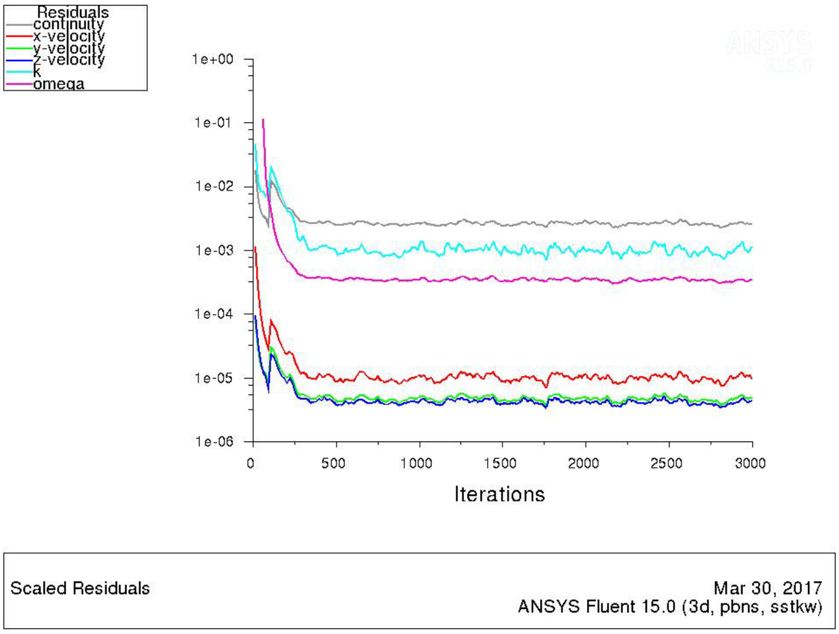


Figure 9.43: Convergence plot of the residuals of the SUV HexaInterior mesh

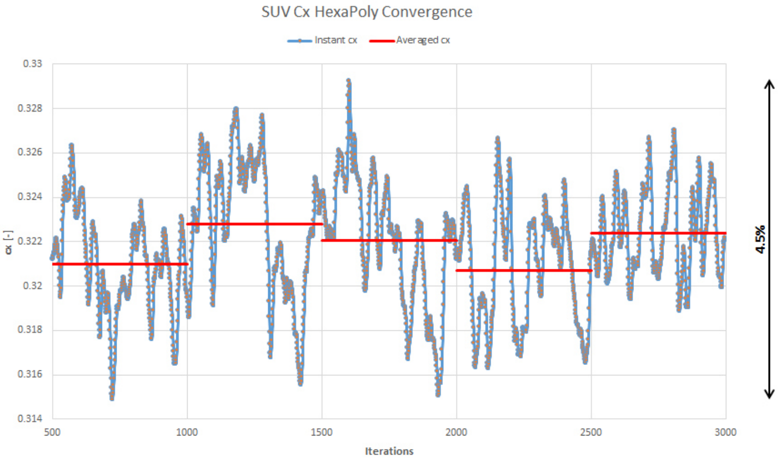


Figure 9.44: Convergence plot of cx of the SUV HexaPoly mesh

Figure 9.44 shows the cx convergence plot of the HexaPoly mesh of the SUV. The period averaged drag coefficient stays within a range of 2.1 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 4.5 per cent of the averaged cx value which is the lowest value for all SUV mesh types. The averaged cx shows a sine wave behaviour with the tendency to decrease. Figure 9.45 shows the corresponding convergence plot of the residuals. Similar to the HexaInterior mesh, the HexaPoly shows a much better convergence as the Tetra and various poly meshes above. Convergence seems to be reached after about 600 iterations.

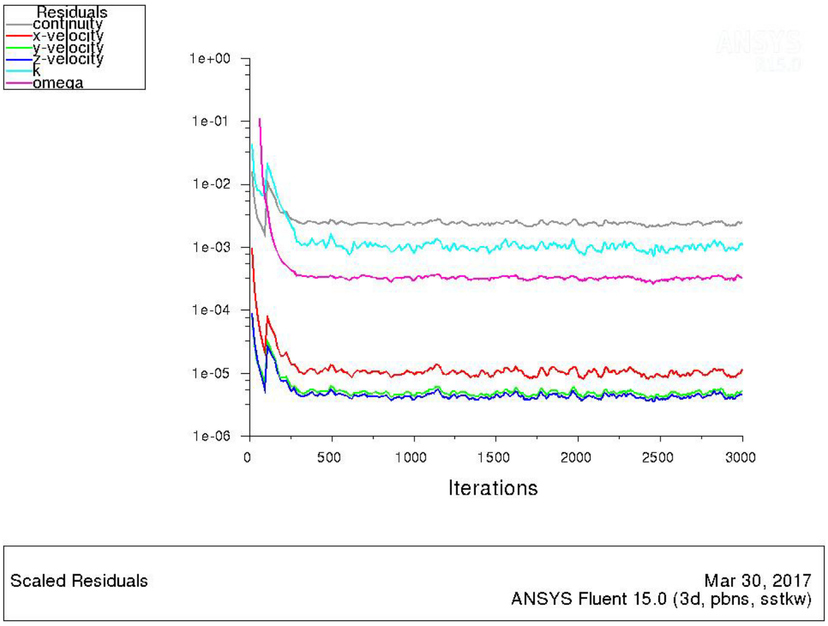


Figure 9.45: Convergence plot of the residuals of the SUV HexaPoly mesh

9.1.4 Sports Car

Mesh Type Comparison

Table 9.4 shows a comparison of all the meshes applied on the sports car. All data is based on 3 000 iterations. Figure 9.47 shows the associated computing speed per mesh cell. As it was the case with all the other cars, the Tetra mesh is the most efficient one. But the efficiency of the HexaInterior mesh is pretty close to the Tetra. The HexaPoly is a bit below and, as it was the case with the SUV, the PolyPoly and the PolyPrism mesh are quite a chunk below and, PolyFluent is another step lower in efficiency (but it still needs to be kept in mind, that the conversion time is included here).

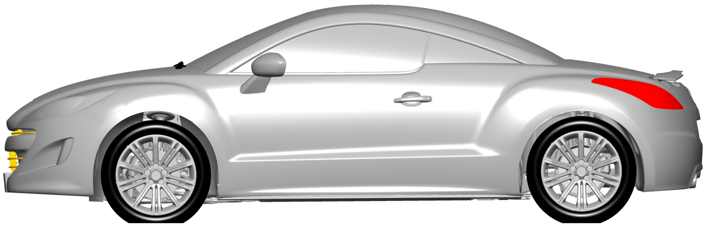


Figure 9.46: Sports car CAD model

Mesh type Sports car	Cell number	Computing time	CPU hours
Tetra	76 664 114	05:35:38 h	1 074
PolyPoly	31 669 855	05:09:48 h	991
PolyPrism	34 877 178	05:46:15 h	1 108
PolyFluent	30 224 170	06:36:54 h	1 270
HexaInterior	97 707 188	08:03:10 h	1 546
HexaPoly	65 477 795	05:47:45 h	1 113

Table 9.4: Computational expenses for different mesh types on the sports car

Same as at the limousine, the city car and the SUV, at the sports car, there is the same picture in mesh type element computational efficiency. Tetra are the strongest elements and poly have the lowest computational efficiency. But as stated before and referring to [123], a poly mesh offers a lot of other benefits which keeps them overall ahead of other mesh types.

Coefficients and Convergence

Figure 9.48 shows a comparison of drag and lift coefficient cx and cz between the different mesh types (average of the last 300 iterations) for the sports car. The Tetra mesh is the base and all other mesh types are referred on the Tetra result. The lift coefficient cz is just shown as an additional information. The focus of this thesis is on drag coefficient.

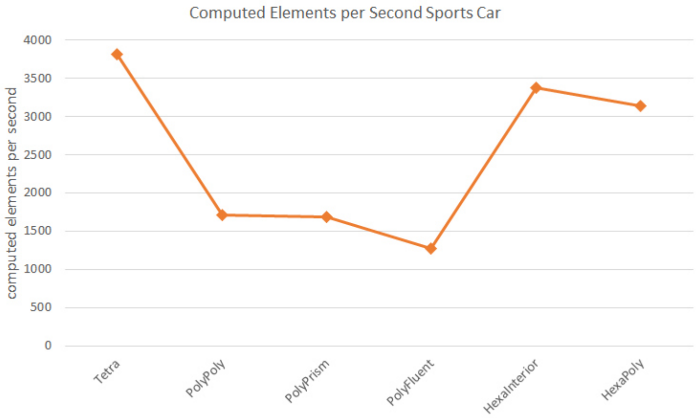


Figure 9.47: Computational efficiency of the different mesh types for the sports car

The HexaInterior shows a deviation of just plus 0.3 per cent. The HexaPoly mesh shows a deviation of plus 1.9 per cent. All three poly meshes are between plus 4.8 and plus 5.4 per cent off. So all three poly meshes deliver very similar results. However, the poly meshes are all quite far off the Tetra result. This time, the HexaInterior delivers the best result with just plus 0.3 per cent off. But also the HexaPoly mesh delivers again a very good result with plus 1.9 per cent off.

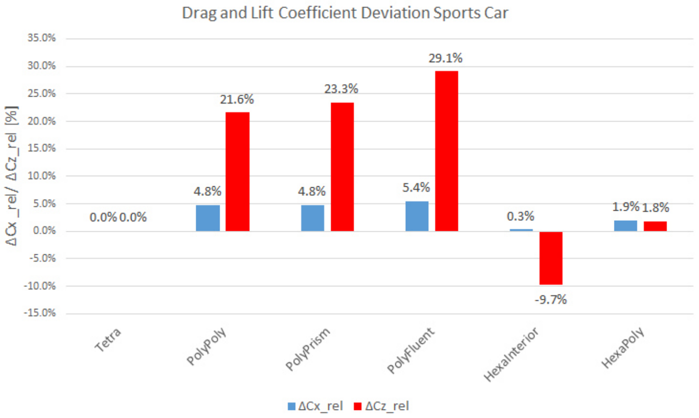


Figure 9.48: Drag and lift coefficient deviation of the sports car

Figures 9.49 to 9.60 show convergence plots of each mesh type for the sports car. For each mesh type, a cx convergence plot and a convergence plot for the residuals are presented. On the cx plots, an averaging was done for each 500 iterations, beginning at iteration 500. By showing these averaged cx values, a good impression of the convergence behaviour is given. As in the most cases before, both hybrid-hexa meshes show by far the best convergence behaviour. A summary and comparison of each mesh type for each car are stated in chapter 9.1.5.

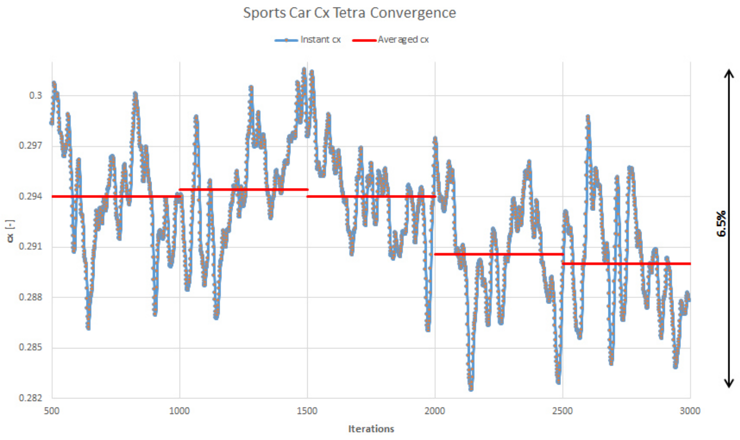


Figure 9.49: Convergence plot of cx of the sports car Tetra mesh

Figure 9.49 shows the cx convergence plot of the HexaPoly mesh of the sports. The period averaged drag coefficient stays within a range of 4.4 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 6.5 per cent of the averaged cx. There is no clear convergence behaviour detectable. First a little rise and then a strong decrease of cx. Figure 9.50 shows the corresponding convergence plot of the residuals. The convergence for the residuals is reached after about 1 500 iterations. But especially the velocity curves are moving a lot till the end of the simulation. The continuity and the omega curve look very smooth.

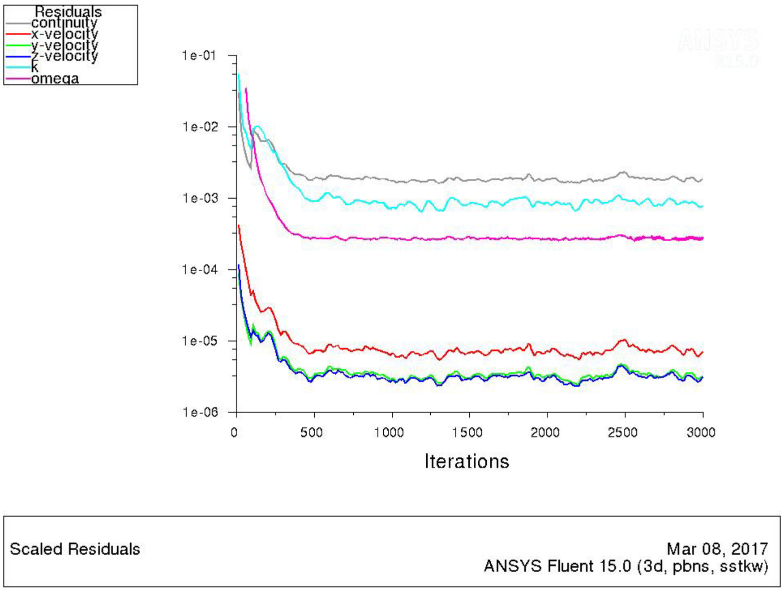


Figure 9.50: Convergence plot of the residuals of the sports car Tetra mesh

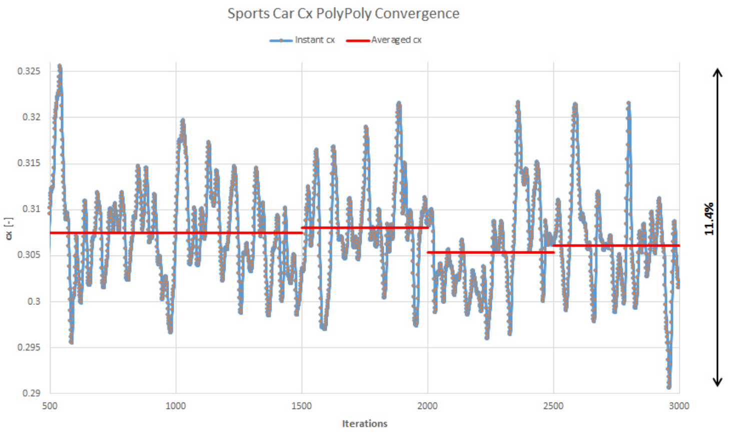


Figure 9.51: Convergence plot of cx of the sports car PolyPoly mesh

Figure 9.51 shows the cx convergence plot of the PolyPoly mesh of the sports. The period averaged drag coefficient stays within a range of 2.7 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 11.4 per cent of the averaged cx. The averaged cx shows kind of a sine wave as convergence behaviour. However, the averaged cx values are not far off each other, but the instant cx shows a lot strong outliers. Figure 9.52 shows the corresponding convergence plot of the residuals. The PolyPoly mesh shows a bit better and quicker convergence as the Tetra mesh. Convergence for the residuals seems to be reached after about 800 iterations. The oscillation of the velocity curves are all around the same value. The continuity and omega curves do not look as smooth as in the Tetra case.

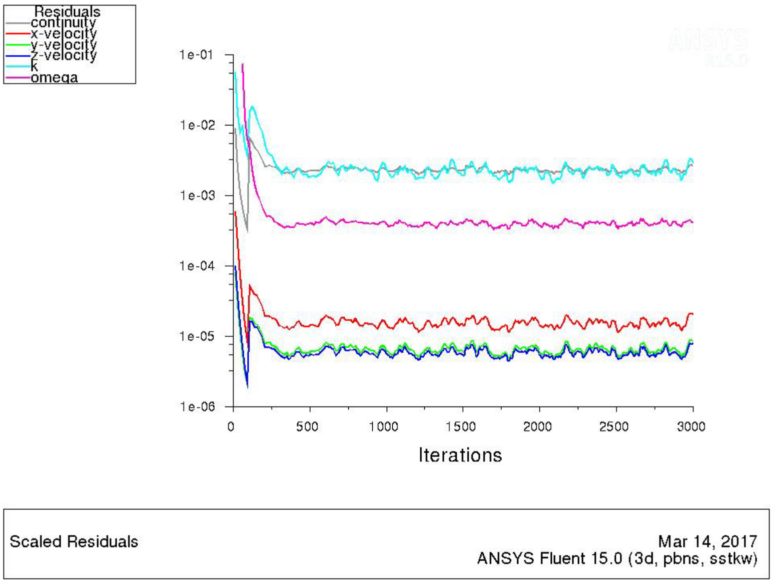


Figure 9.52: Convergence plot of the residuals of the sports car PolyPoly mesh

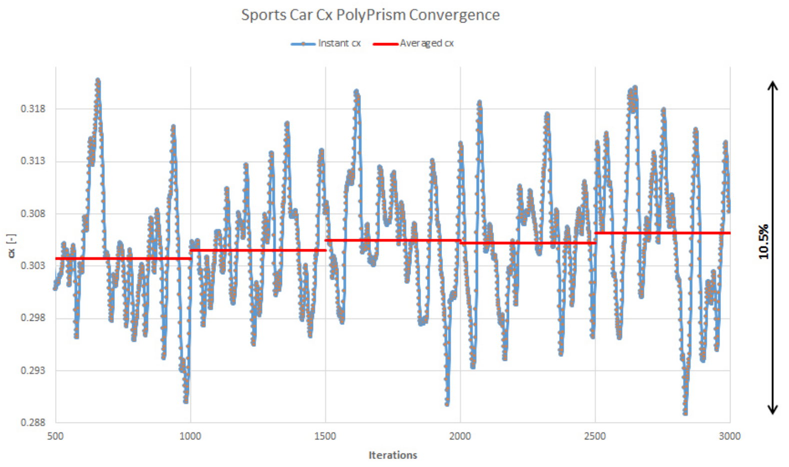


Figure 9.53: Convergence plot of cx of the sports car PolyPrism mesh

Figure 9.53 shows the cx convergence plot of the PolyPrism mesh of the sports. The drag coefficient stays within a range of 2.4 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 10.5 per cent of the averaged cx. The averaged cx shows a tendency to rise. But as it is the case with the PolyPoly mesh, the averaged cx values are pretty close to each other, but the instant cx is oscillating pretty intensely. Figure 9.54 shows the corresponding convergence plot of the residuals. The PolyPrism case looks very similar to the PolyPoly case, with maybe a bit smoother curves and slightly earlier convergence.

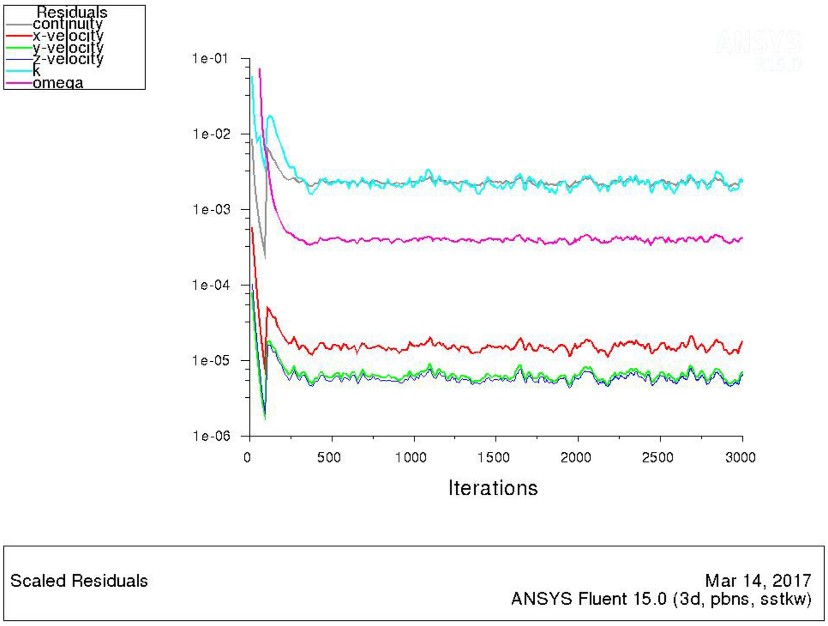


Figure 9.54: Convergence plot of the residuals of the sports car PolyPrism mesh

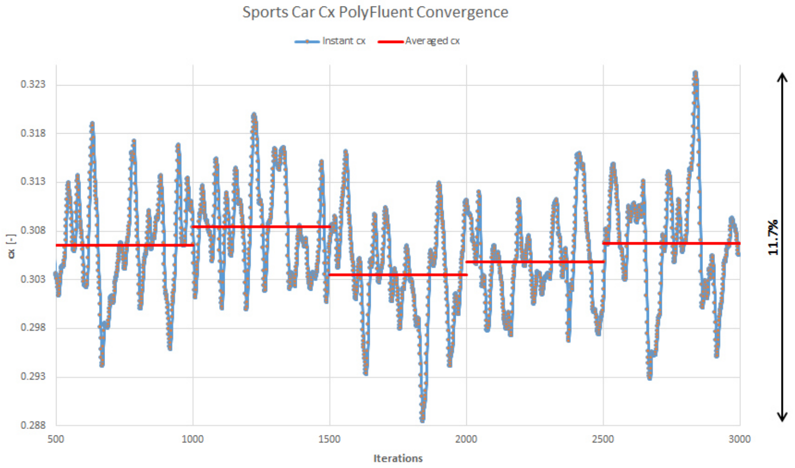


Figure 9.55: Convergence plot of cx of the sports car PolyFluent mesh

Figure 9.55 shows the cx convergence plot of the PolyFluent mesh of the sports. The period averaged drag coefficient stays within a range of 5.0 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 10.5 per cent of the averaged cx. As it is the case with both poly meshes above, the instant cx value has pretty high amplitudes compared to the Tetra mesh or to the two hexa meshes. Figure 9.56 shows the corresponding convergence plot of the residuals. The PolyFluent residuals show more or less exactly the same residual convergence as the PolyPoly case.

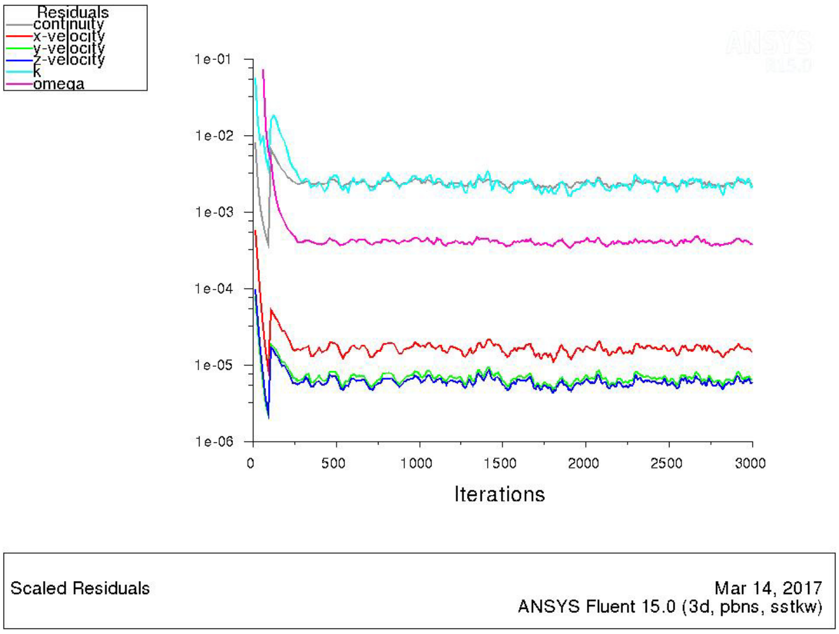


Figure 9.56: Convergence plot of the residuals of the sports car PolyFluent mesh

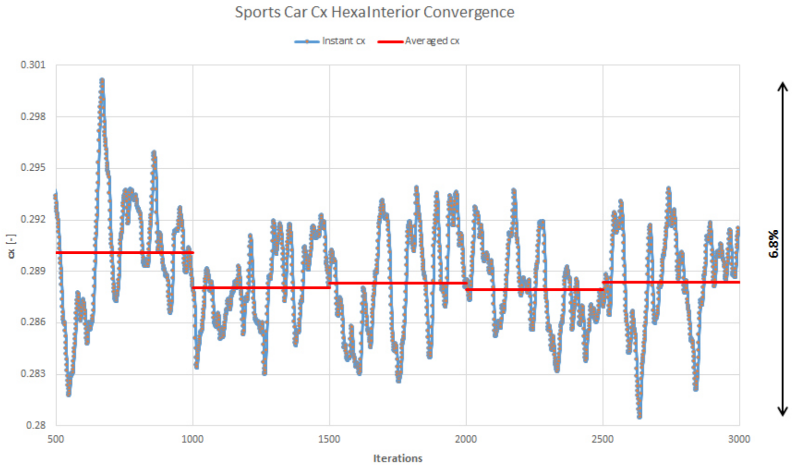


Figure 9.57: Convergence plot of cx of the sports car HexaInterior mesh

Figure 9.57 shows the cx convergence plot of the HexaInterior mesh of the sports. The period averaged drag coefficient stays within a range of 2.2 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 6.8 per cent of the averaged cx. The averaged cx shows a sine wave behaviour with, except for the first period (500 to 1 000 iterations), a very small amplitude. Figure 9.58 shows the corresponding convergence plot of the residuals. The HexaPoly mesh shows, together with the HexaPoly mesh, the quickest and best residual convergence.

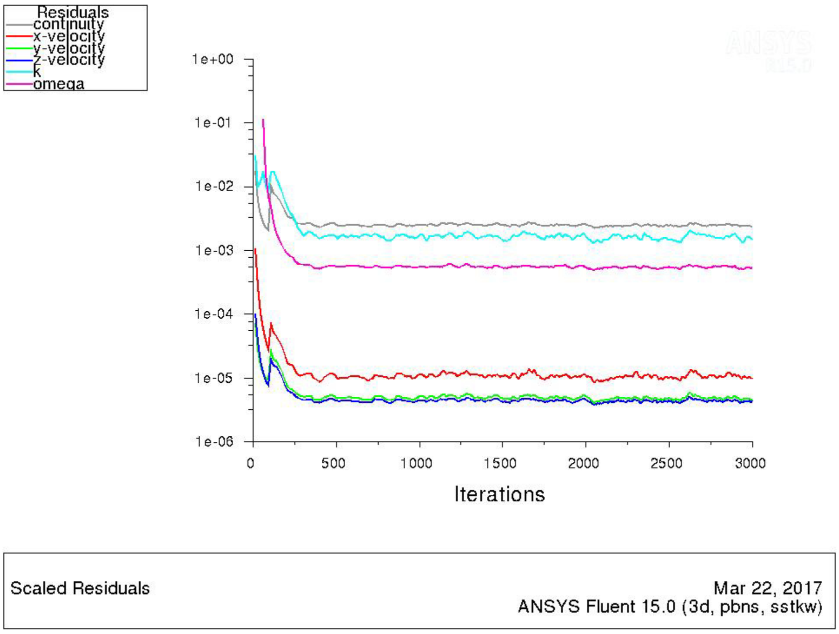


Figure 9.58: Convergence plot of the residuals of the sports car HexaInterior mesh

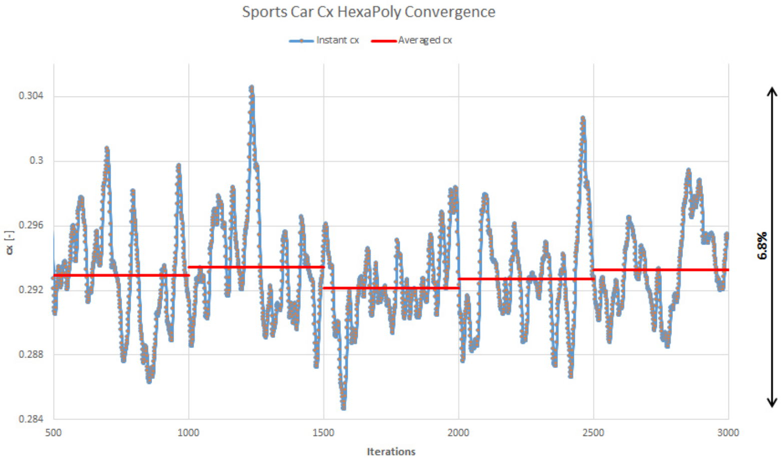


Figure 9.59: Convergence plot of cx of the sports car HexaPoly mesh

Figure 9.59 shows the cx convergence plot of the HexaPoly mesh of the sports. The period averaged drag coefficient stays within a range of 1.3 points over 2 500 iterations. The maximum deviation between the lowest and the highest instant cx value is 6.8 per cent of the averaged cx. Figure 9.60 shows the corresponding convergence plot of the residuals. As it is the case in two of the three cars above, again the HexaPoly shows by far the best residual convergence for the sports car. Convergence is reached at about 550 iterations and again most of the curves look impressively smooth when compared to the other mesh types.

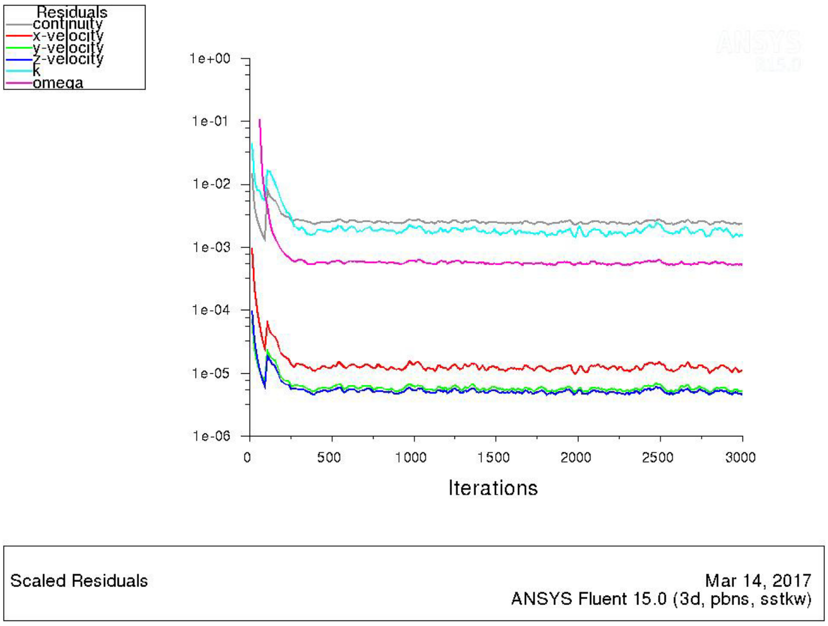


Figure 9.60: Convergence plot of the residuals of the sports car HexaPoly mesh

Force/Accumulated Forces at y=0-Section

The drag force Fx is closely related to the drag coefficient cx. Referring to equation 3.9, the relation between these two is

$$F_x = \frac{1}{2} c_x \rho u^2 A.$$

Figure 9.48 shows the deviation of the drag coefficient for each mesh type. The drag force is directly proportional to the drag coefficient. Figure 9.61 gives an impression how the drag force (and therefore drag coefficient) builds up over the length of the car. It is remarkable, that at the front of the car, all three poly meshes are below the Tetra reference mesh and both hexa meshes are above it. At the transition between the engine bonnet and the windscreen, all curves meet and the situation inverses. Behind this point, all poly meshes are over the reference thetra mesh and both hexa meshes are below. The last change happens at the tailgate, where the Tetra mesh falls below the hexa meshes and finally delivers the lowest drag force overall.

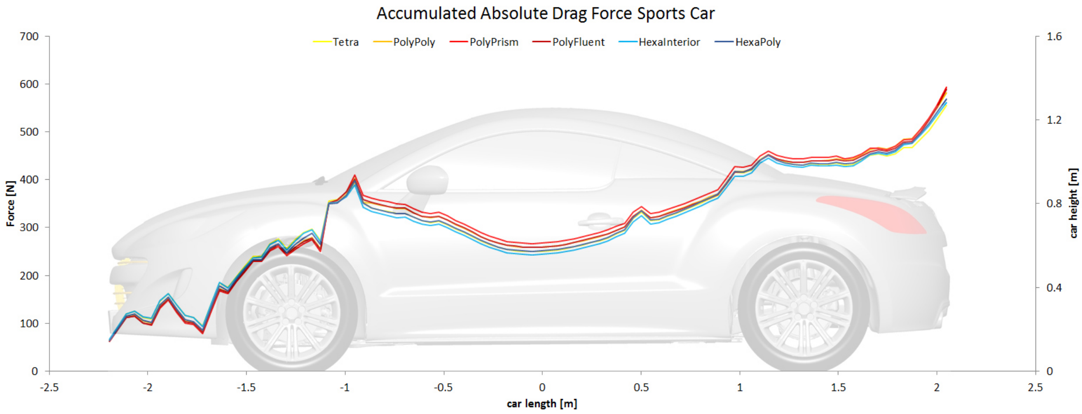


Figure 9.61: Accumulated drag force of the sports car for the six different meshes

Figure 9.62 delivers a bit a clearer view. This graphic shows delta force related to the Tetra mesh. All poly meshes deliver qualitative results. There are some minor differences in quantity, but especially the PolyPoly and the PolyFluent are almost identical. The PolyPrism is always on top. The qualitative tendency of the HexaPoly is more similar to the poly meshes than the HexaInterior. But it is likely to be the closest one to the Tetra result.

Figure 9.63 breaks down the force production over the car length. This graphic shows the local distribution of the drag force Fx. All meshes deliver the same curve with minor variations of quantity. Just at the rear of the car, a larger difference between the meshes can be detected. The poly meshes deliver the highest increase of drag force at the rear of the car. The hybrid-hexa meshes in contrast deliver the lowest increase of drag force at the rear of the car. The Tetra mesh lies in between.

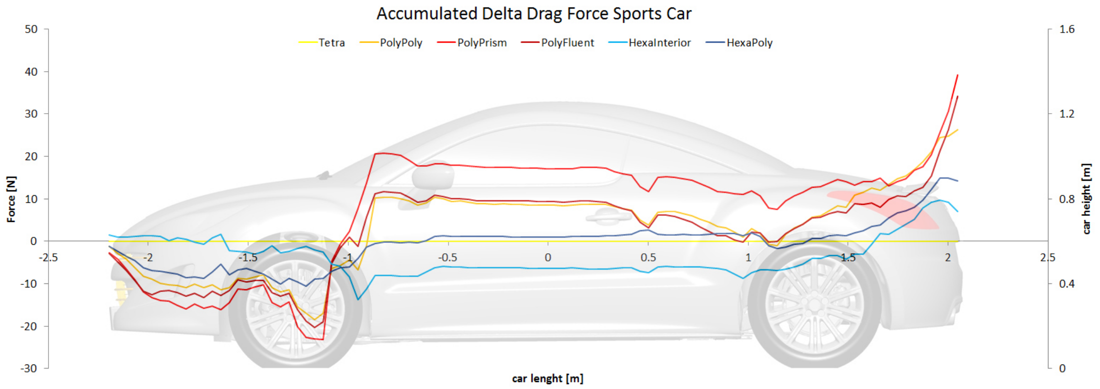


Figure 9.62: Accumulated drag force delta of the sports car for the six different meshes

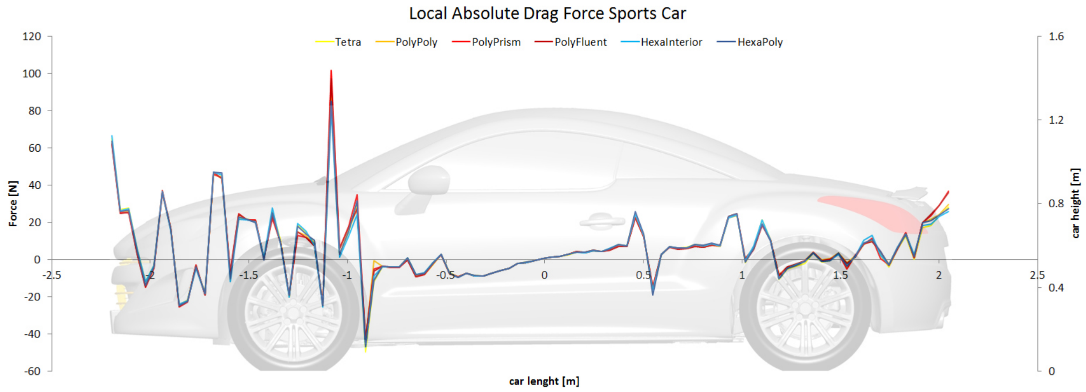


Figure 9.63: Local drag force of the sports car for the six different meshes

Figure 9.64 shows local delta force related to the Tetra mesh. This graphic just confirms that the poly meshes build up the highest drag force. Again it is visible, that almost everywhere, the PolyPrism mesh delivers the highest change of local force. Also the hybrid-hexa meshes stay relatively close to each other, but are pretty far off the poly meshes. Again it can be detected, that the HexaPoly mesh delivers the closest result related to the Tetra mesh. Just a bit more off is the HexaInterior mesh. The poly meshes are all pretty far off the Tetra mesh, when compared to the bybrid-hexa meshes. With the exception of the transition between the engine bonnet and the windscreen, the highest delta value is approximately 10 N. Considering the fact, that the entire drag force is approximately 600 N, this is not a big amount.

All in all, the different mesh types do not have a big influence on the local change of drag force. But accumulated, this results in a decent deviation of overall drag force and therefore drag coefficient (see figure 9.48). This behaviour can be transferred to all other cars tested.

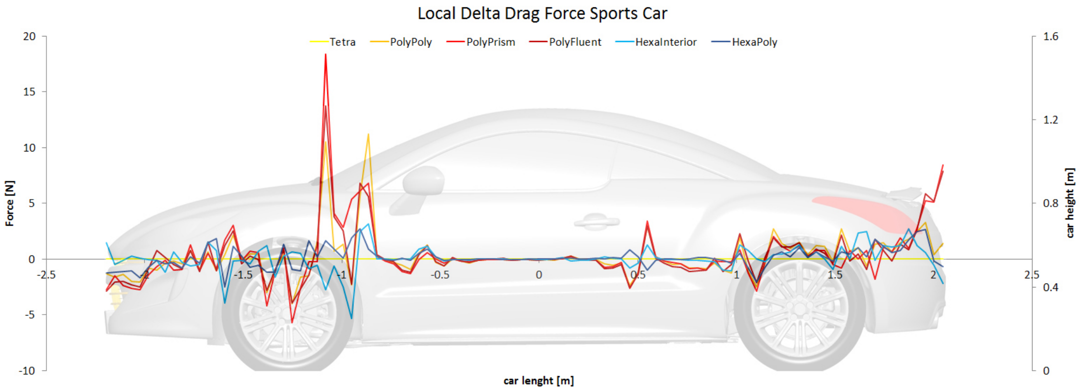


Figure 9.64: Local drag force delta of the sports car for the six different meshes

Velocity and Pressure Visualisation

The following subsections present figures which visualise the velocity as well as the pressure distribution in various sections, isosurfaces as well as on the car's surface. For each type of figure, a comparison between all six tested mesh types is given. A summary of these comparisons and differences of the results between the various meshes is discussed at the end of this series of figures.

It has to be kept in mind that these figures present the solution of the last iteration step. The results of the drag and lift coefficients as well as of the forces, in contrast, are averaged over the last 300 iteration steps.

Some of the figures show large differences between the various mesh types. Other figures, in contrast, look very similar for all six mesh types. An interesting fact is, that the Tetra mesh and both hybrid-hexa meshes show very similar flow qualities. The flow figure of the poly meshes, in contrast, sometimes differ quite a lot from the Tetra and hybrid-hexa types. This coincides with the numbers (cx, cz, Fx, Fz), presented in the previous sections. However, the Tetra mesh shows the most flow details of all mesh types. The PolyPrism mesh shows the best coincidence with the Tetra and hybrid-hexa meshes.

Cpstat Contours Car

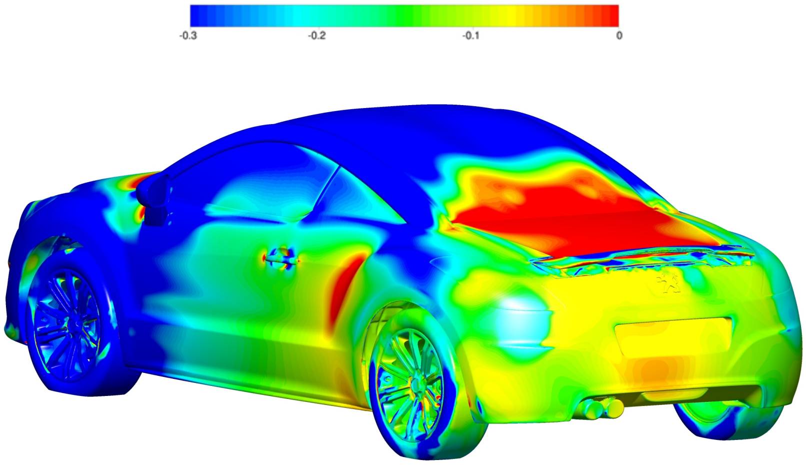


Figure 9.65: Static pressure coefficient contour of the sports car Tetra mesh

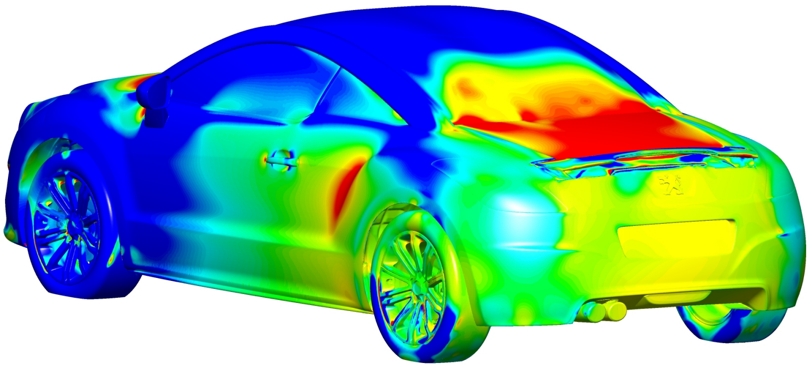


Figure 9.66: Static pressure coefficient contour of the sports car PolyPoly mesh

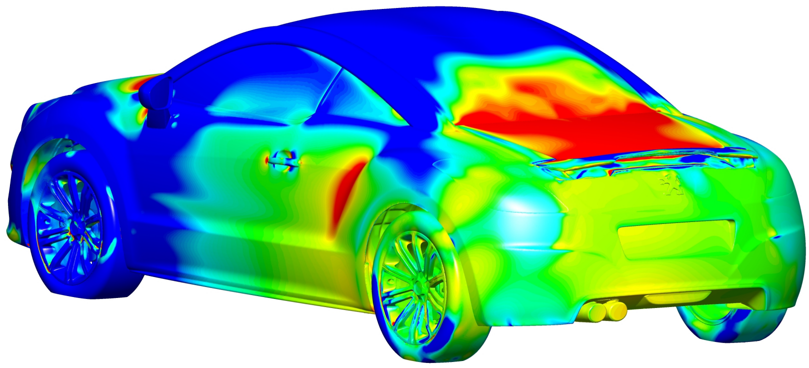


Figure 9.67: Static pressure coefficient contour of the sports car PolyPrism mesh

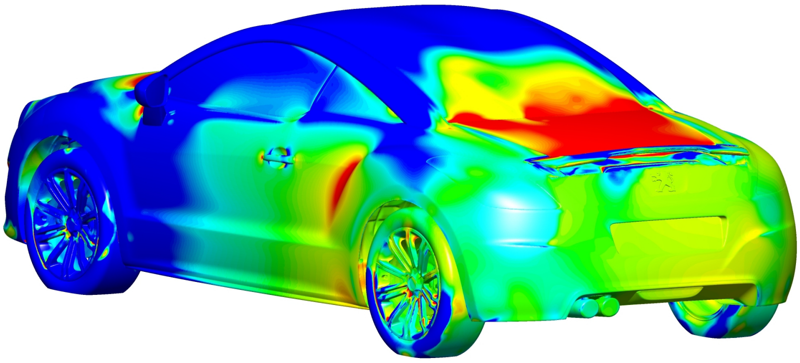


Figure 9.68: Static pressure coefficient contour of the sports car PolyFluent mesh

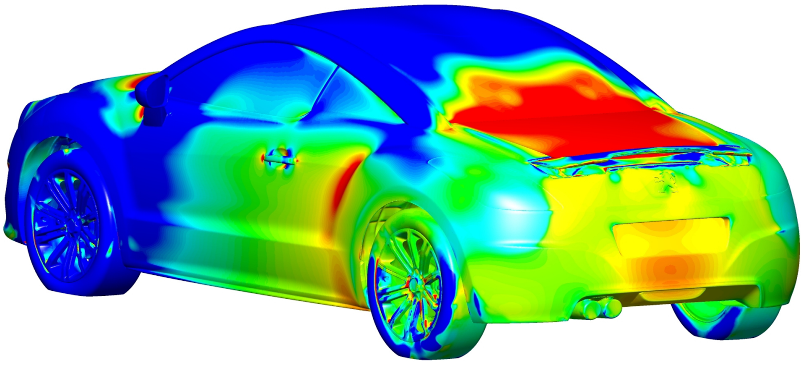


Figure 9.69: Static pressure coefficient contour of the sports car HexaInterior mesh

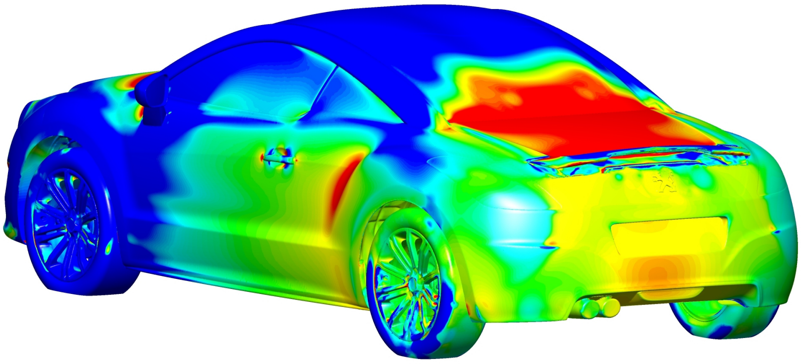
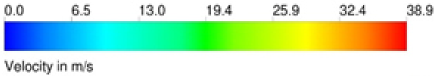


Figure 9.70: Static pressure coefficient contour of the sports car HexaPoly mesh

Isosurfaces cptot=0, Coloured by Velocity



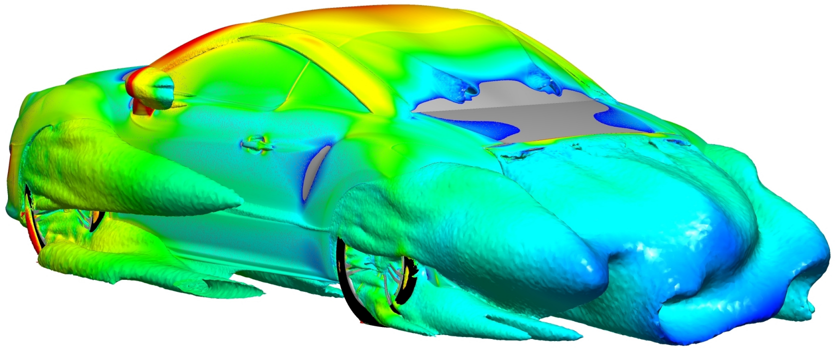


Figure 9.71: Isosurface of the total pressure coefficient of the sports car Tetra mesh

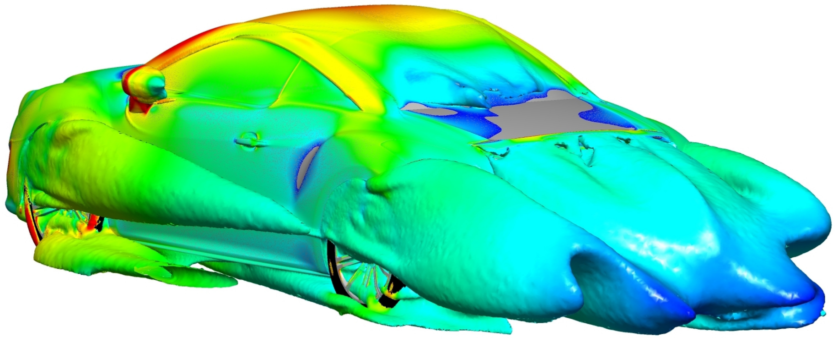


Figure 9.72: Isosurface of the total pressure coefficient of the sports car PolyPoly mesh

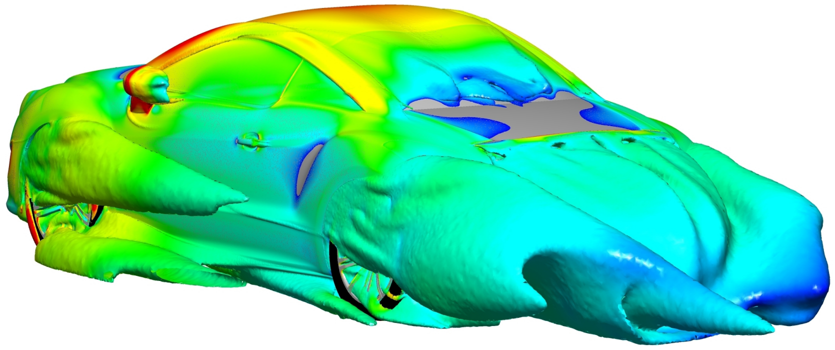


Figure 9.73: Iso surface of the total pressure coefficient of the sports car PolyPrism mesh

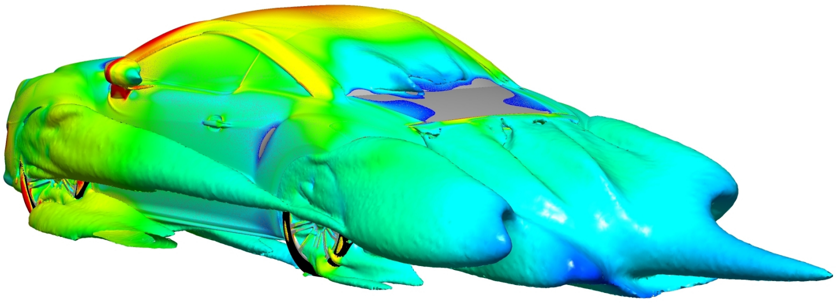


Figure 9.74: Isosurface of the total pressure coefficient of the sports car PolyFluent mesh

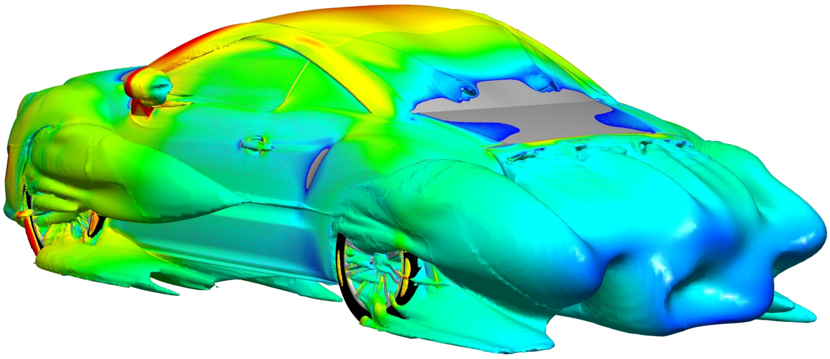


Figure 9.75: Isosurface of the total pressure coefficient of the sports car HexaInterior mesh

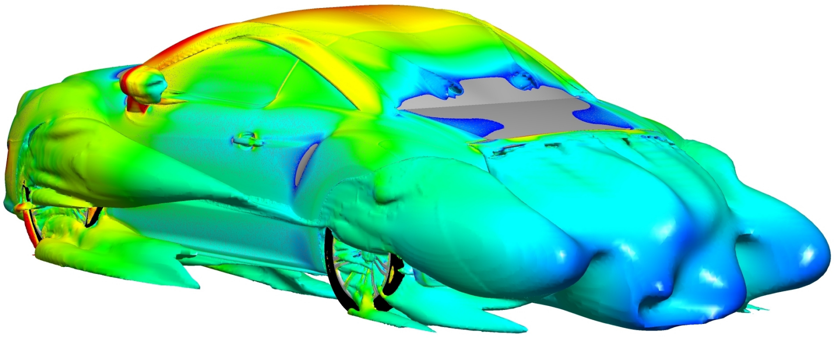


Figure 9.76: Isosurface of the total pressure coefficient of the sports car HexaPoly mesh

Skin Friction Coefficient

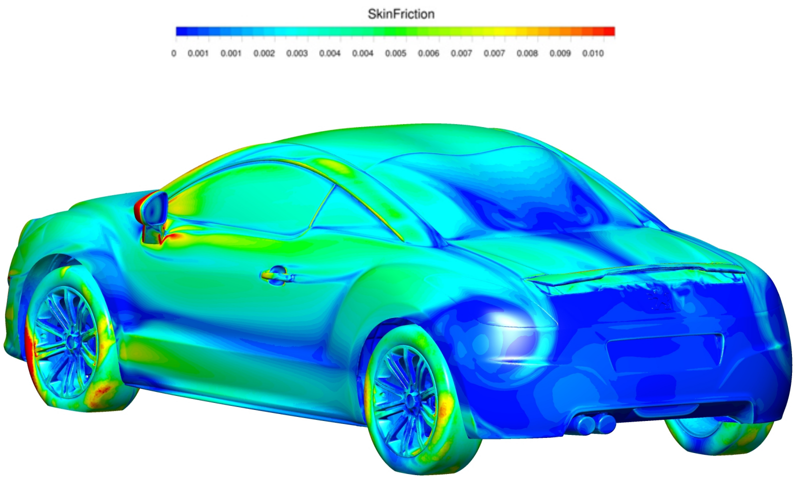


Figure 9.77: Skin friction coefficient contour of the sports car Tetra mesh

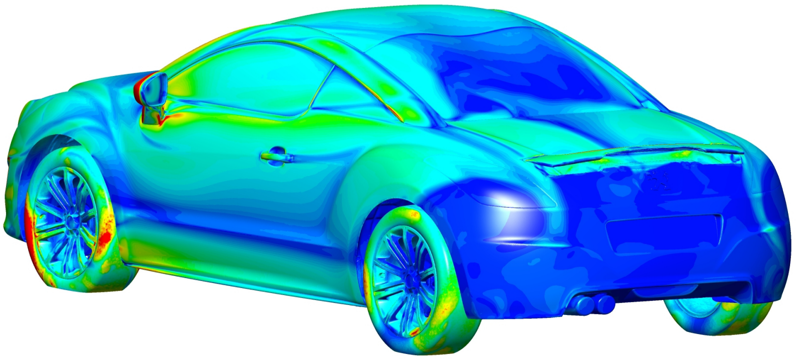


Figure 9.78: Skin friction coefficient contour of the sports car PolyPoly mesh

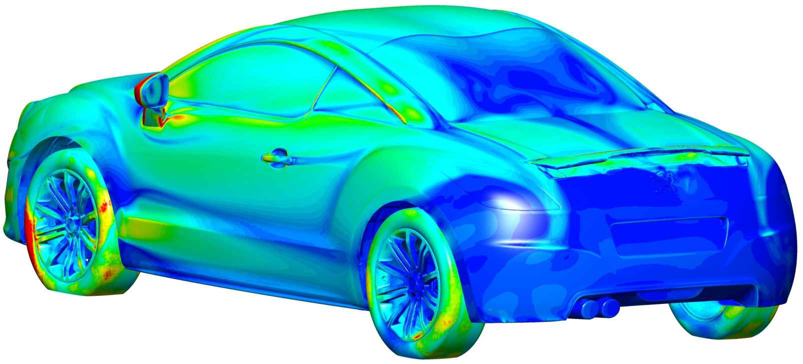


Figure 9.79: Skin friction coefficient contour of the sports car PolyPrism mesh

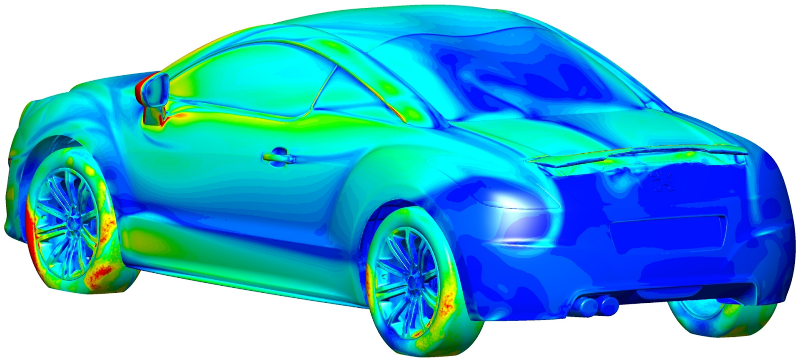


Figure 9.80: Skin friction coefficient contour of the sports car PolyFluent mesh

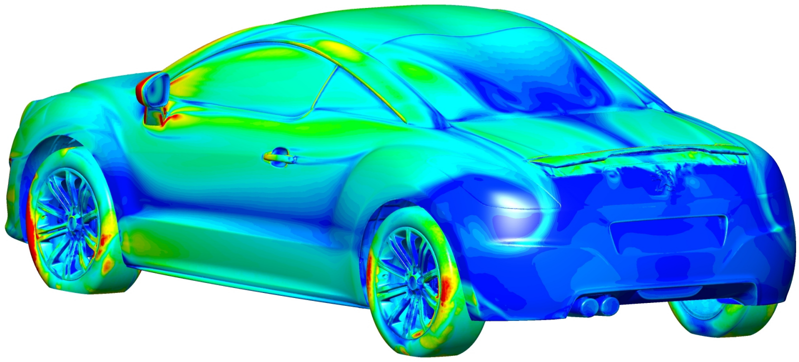


Figure 9.81: Skin friction coefficient contour of the sports car HexaInterior mesh

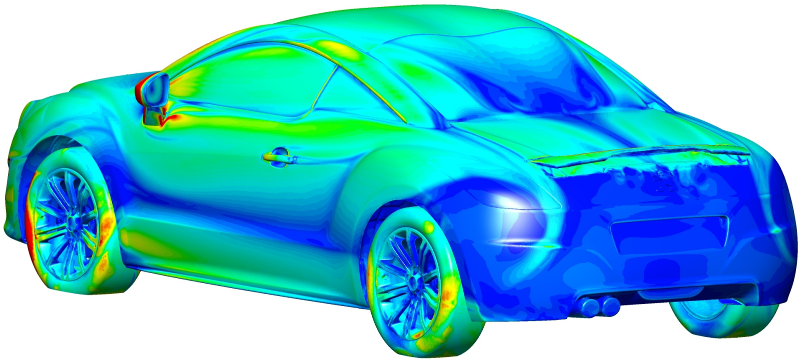
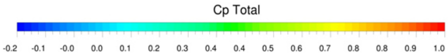


Figure 9.82: Skin friction coefficient contour of the sports car HexaPoly mesh

X-Section Cptot



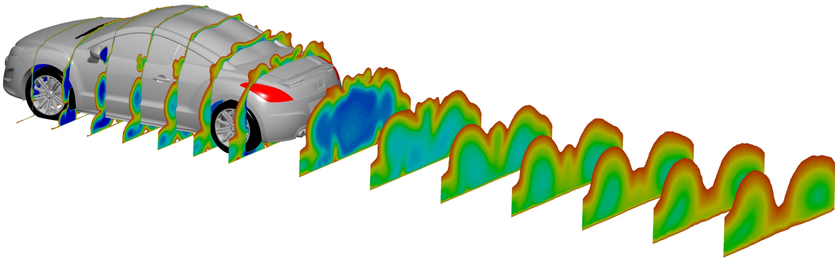


Figure 9.83: X-plane for total pressure coefficient for the sports car Tetra mesh

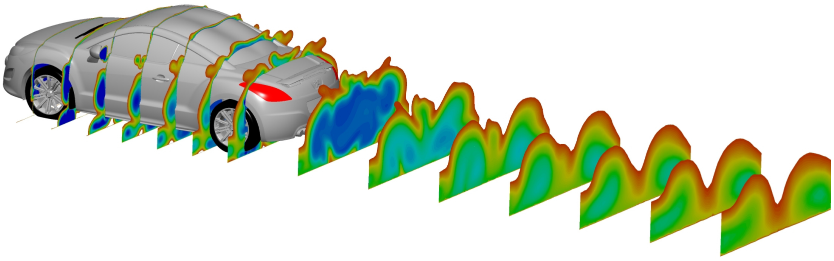


Figure 9.84: X-plane for total pressure coefficient for the sports car PolyPoly mesh

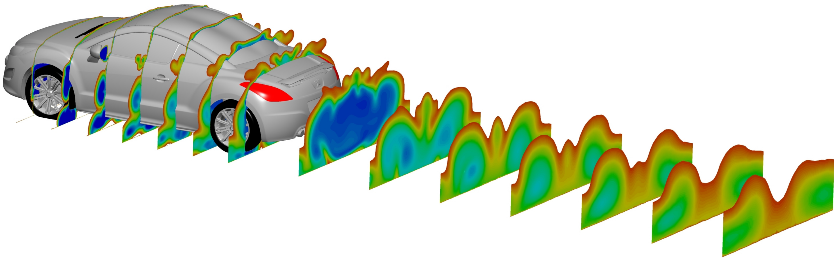


Figure 9.85: X-plane for total pressure coefficient for the sports car PolyPrism mesh

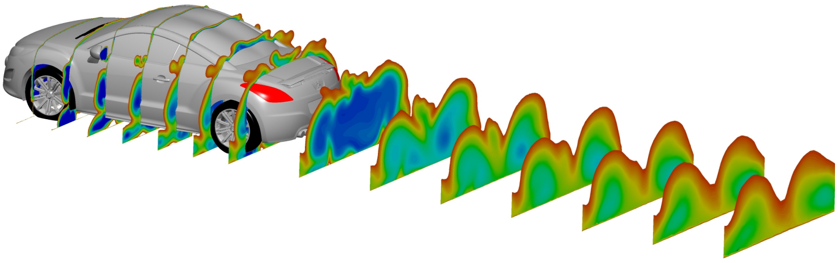


Figure 9.86: X-plane for total pressure coefficient for the sports car PolyFluent mesh

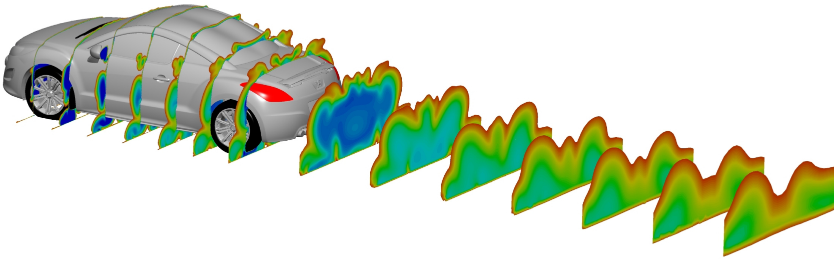


Figure 9.87: X-plane for total pressure coefficient for the sports car HexaInterior mesh

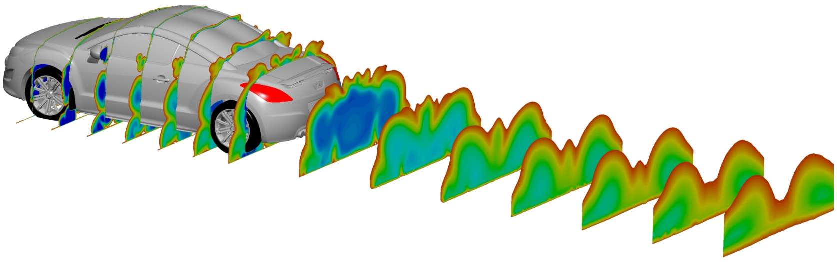
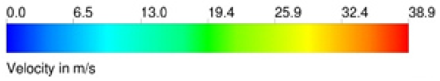


Figure 9.88: X-plane for total pressure coefficient for the sports car HexaPoly mesh

X-Section Velocity Streamlines



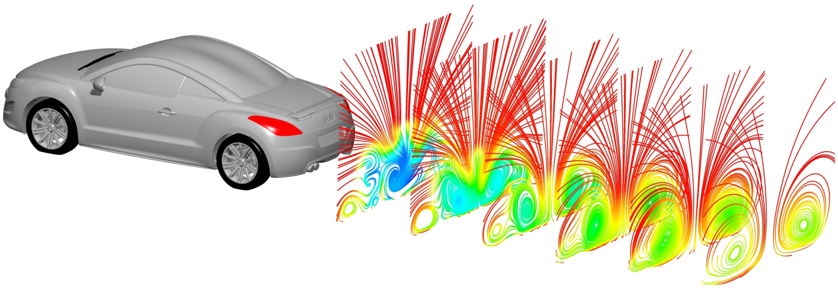


Figure 9.89: X-plane velocity streamlines in the y-plane of the sports car Tetra mesh

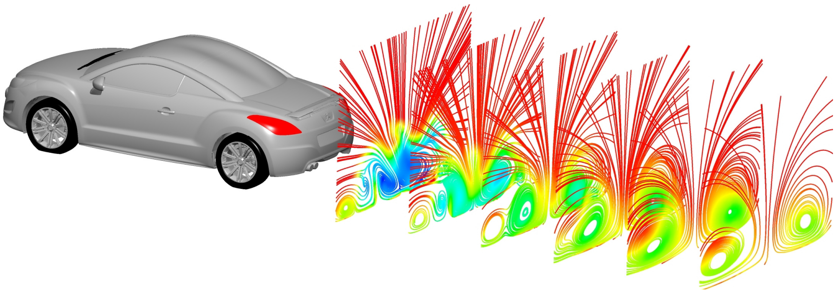


Figure 9.90: X-plane velocity streamlines in the y-plane of the sports car PolyPoly mesh

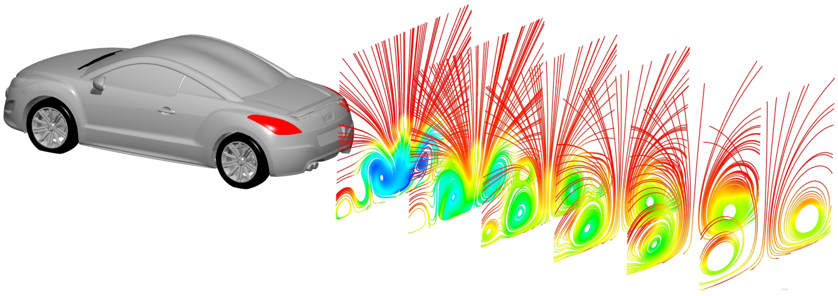


Figure 9.91: X-plane velocity streamlines in the y-plane of the sports car PolyPrism mesh

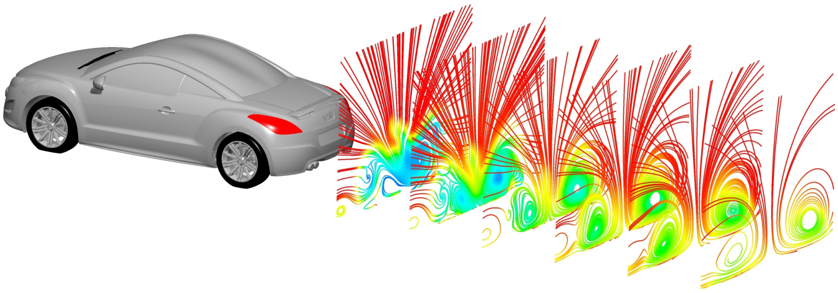


Figure 9.92: X-plane velocity streamlines in the y-plane of the sports car PolyFluent mesh

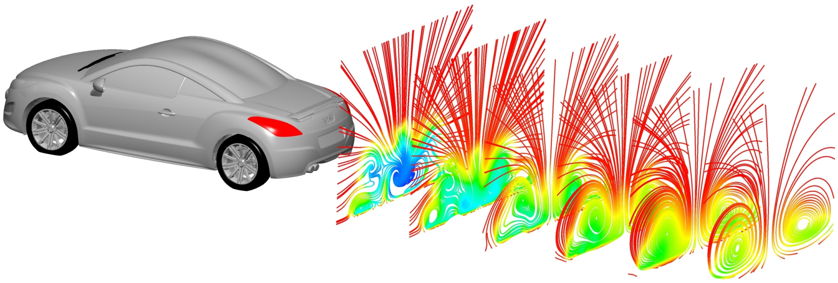


Figure 9.93: X-plane velocity streamlines in the y-plane of the sports car HexaInterior mesh

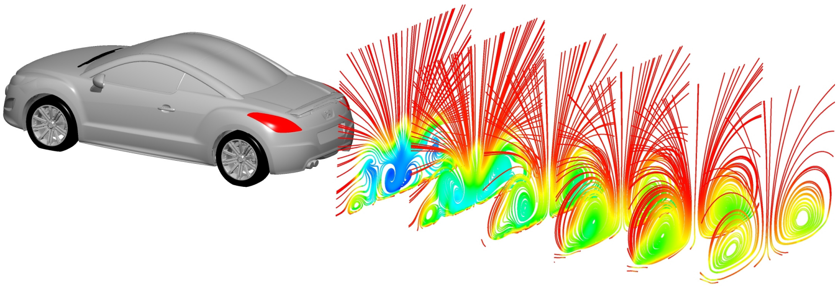
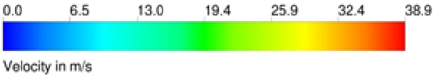


Figure 9.94: X-plane velocity streamlines in the y-plane of the sports car HexaPoly mesh

Velocity Contour and Streamlines in y=0-Section



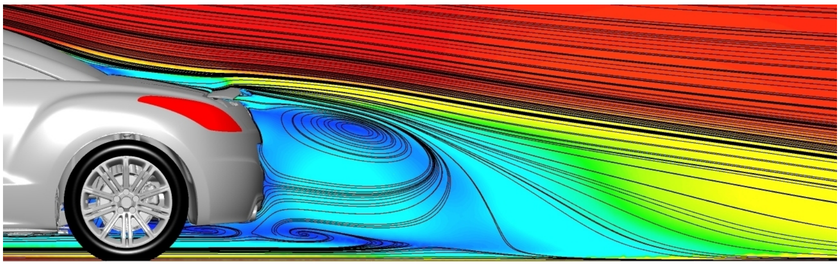


Figure 9.95: Velocity streamlines in the y-plane of the sports car Tetra mesh

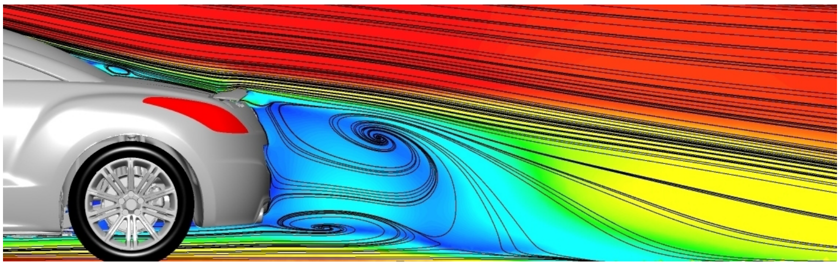


Figure 9.96: Velocity streamlines in the y-plane of the sports car PolyPoly mesh

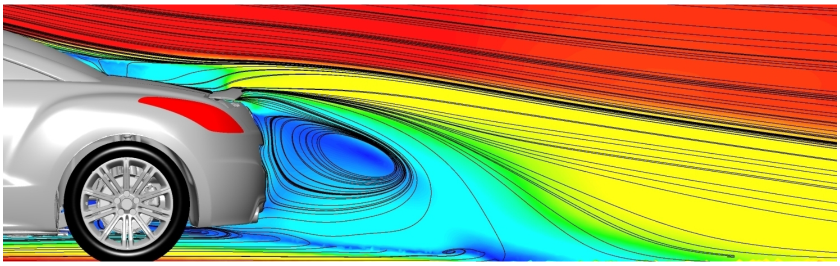


Figure 9.97: Velocity streamlines in the y-plane of the sports car PolyPrism mesh

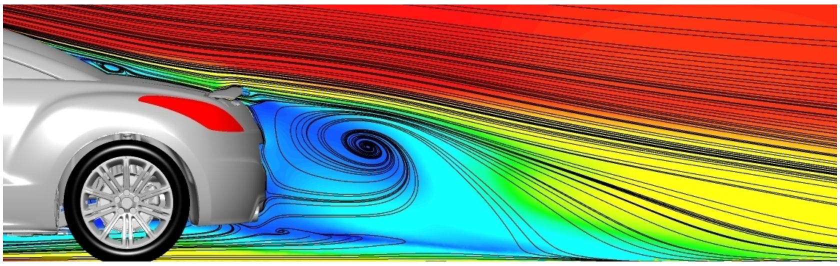


Figure 9.98: Velocity streamlines in the y-plane of the sports car PolyFluent mesh

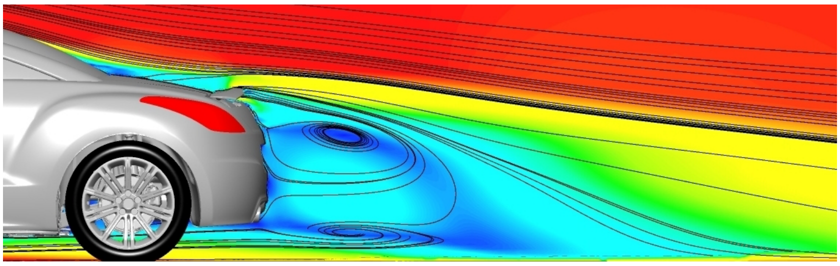


Figure 9.99: Velocity streamlines in the y-plane of the sports car HexaInterior mesh

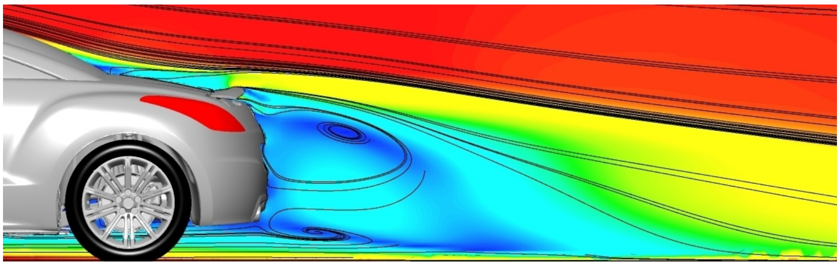
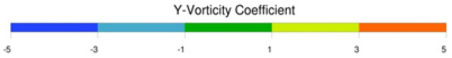


Figure 9.100: Velocity streamlines in the y-plane of the sports car HexaPoly mesh

Y-Component of Vorticity Coefficient



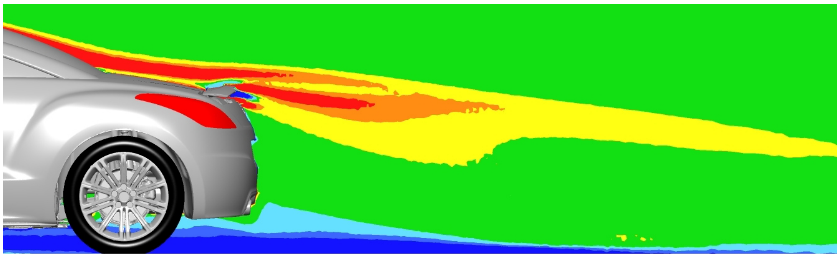


Figure 9.101: Y-component of the vorticity coefficient of the sports car Tetra mesh

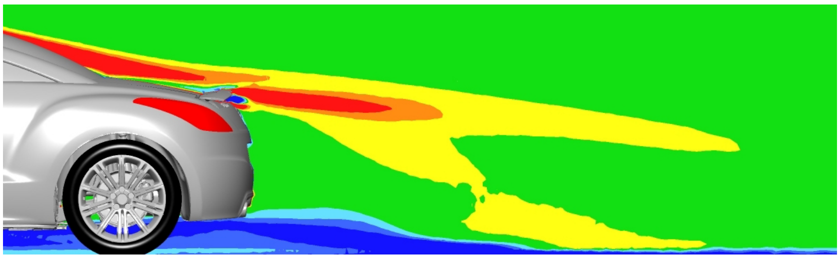


Figure 9.102: Y-component of the vorticity coefficient of the sports car PolyPoly mesh

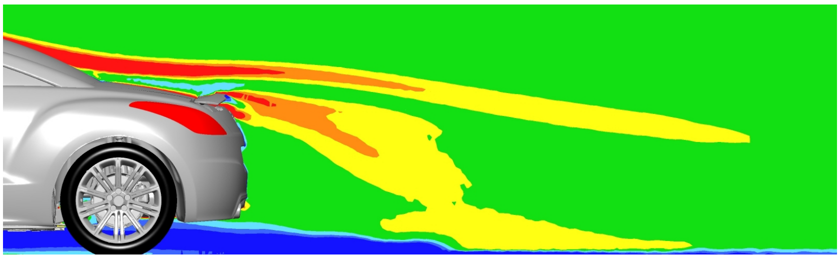


Figure 9.103: Y-component of the vorticity coefficient of the sports car PolyPrism mesh

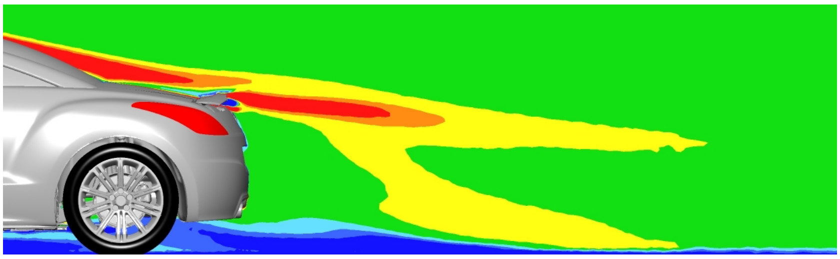


Figure 9.104: Y-component of the vorticity coefficient of the sports car PolyFluent mesh

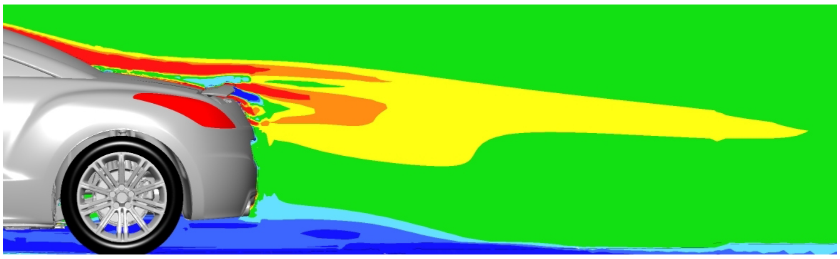


Figure 9.105: Y-component of the vorticity coefficient of the sports car HexaInterior mesh

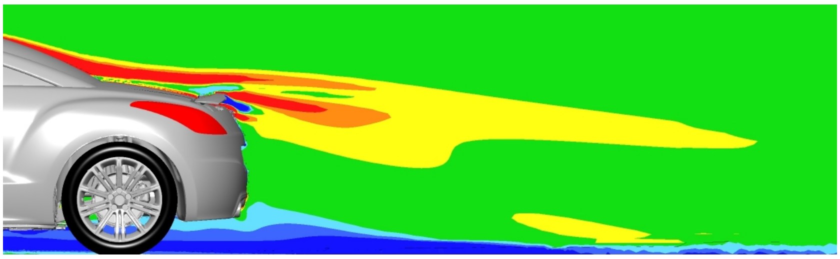


Figure 9.106: Y-component of the vorticity coefficient of the sports car HexaPoly mesh

Figures 9.65 to 9.70 show the static pressure coefficient contour for all different meshes in a range of +0.3 to 0.0. As there is hardly a difference visible at the front of the car between the different tested mesh types, only a view from the rear is presented. It can be seen, that the high pressure area at the rear window is pretty much the same at the Tetra and both hybrid-hexa meshes. The high pressure area seems also to be stronger at these two mesh types compared to the poly meshes. Additionally, below the license plate a pressure point can be detected for the Tetra and hybrid-hexa meshes which is missing on all poly meshes.

Figures 9.71 to 9.76 show the isosurfaces cptot=0, Coloured by Velocity for all tested meshes. These figures are ideally suited to display and compare the wake of the car in the steady-state mode. Anyway, not too much attention should be given to these figures, as the wake seems to commute from left to right, as it is very asymmetric in this figure. A few iterations later or earlier, the picture could be a pretty different one. Same as above, the Tetra and the hybrid-hexa meshes deliver pretty much the same results. The wake of all poly meshes look larger. This again coincidences with the pressure plot figures, as the Tetra and hybrid-hexa meshes have a stronger and larger high pressure area. Also the wake from the front wheels is considerably larger on all poly meshes.

Figures 9.77 to 9.82 show the skin friction coefficient contour plot for all tested mesh types. The skin friction coefficient (cf) is another indicator for separated flow. Where the skin friction is low, the flow acceleration is low. This is an indication for separated areas. It is relatively hard to judge a difference from these figures. The area on the rear screen displays some minor differences. The PolyPoly and the PolyFluent meshes (both without prism layer) seem to have a larger separation area than the PolyPrism. But the PolyPrism has still a larger separation area than the Tetra and both hybrid-hexas.

Figures 9.83 to 9.88 show x-plane sections for the cptot coefficient. These x-sections are helpful to visualise the pressure distribution in the wake area. The dark blue areas are equivalent to the isosurface figures above. This figures verify the statement above, that all poly meshes have a more extended low pressure area behind the car, what is maybe best visible in the second x-section behind the car. Everything else is pretty difficult to judge, as the wake presumably is changing from iteration to iteration.

Figures 9.89 to 9.94 show x-plane sections with velocity streamlines. These x-sections are ideally suited to show the flow structure in the wake of the car for steady-state solutions. As at the cptot coefficient x-sections above, it is hard to judge anything from these statical velocity plots. One effect can be seen, when concentrating on the very last section. In some figures, the right or left vortex is stronger than its opposite. This means, it contains more energy. The opposite vortex is weaker in the first case. In the second case (for example HexaPoly), both vortices are similarly strong. This is another indicator, that the wake is commuting from one side to the other from iteration to iteration.

Figures 9.95 to 9.100 show the velocity contour and streamlines in y=0 Section. These are maybe the most meaningful plots for a steady-state solution. They describe the flow path in the y-zero section. But as always, the figures should be observed with a lot of caution. The flow path can be very different in another y-section. The flow path in a different y-section of a specific mesh could be the same as the flow path in y-zero of the

Tetra mesh. But both y-zero flow paths could be considerably different. It is clearly visible, that the Tetra mesh patterns the most flow details. One large and two smaller vortices can be detected in the wake and at the underbody. The PolyPoly misses the vortex at the underbody, but shows a tiny one at the transition from the rear window to the tailgate. The PolyPrism shows the same vortices as the Tetra mesh, but the lower one in the wake is just rudimentary. The PolyFluent in contrast shows just the large one. The small vortex in the wake is also just rudimentary. An approach of the underbody vortex can also be detected as well as the vortex in the transition between the rear window and the tailgate. Both, HexaInterior and HexaPoly, show exclusively the large and the small vortex in the wake. Another phenomena which can be detected is, that on all models with a prism layer, the flow detaches at the end of the rear window. Just very small variations can be detected here. In the case of the PolyPoly and PolyFluent in contrast, the flow does just separate for a nuance. In that area, an additional vortex is created. The flow reattaches immediately and stays fully attached till the rear spoiler. Quantitatively the plots look pretty similar. Especially the velocity distribution is in fact the same. But in flow details, there are massive differences between the different mesh types. Even between the poly meshes, large differences can be seen.

Figures 9.101 to 9.106 show the y-component of the vorticity. These plots are a tool to detect vortices. This is maybe the most comparable steady-state tool to the Q-criterion[3] for unsteady flows. These figures show again a strong similarity between the Tetra and the hybrid-hexa meshes. One large positive rotating trail from the rear screen and rear spoiler and a negative rotating train on the ground. At all poly meshes, the top trail separates into two trails. The lower trail interferes with the ground sticking negative rotating trail. The ground sticking trail in addition seems to be smaller at the poly meshes.

9.1.5 Summary of Steady-State Simulation

Numbers

To get a good overview of the capabilities of all these different mesh types, it is helpful to summarise all data presented in the previous chapters. The objective of this thesis is to find a mesh, which offers good accuracy as well as convergence for all types of cars. One mesh could deliver good results for the city car, but could deliver the worst result on the SUV. In the next few pages, the most universal mesh is determined.

Figure 9.107 summarises the computational efficiency of the different mesh types. For all cars, a similar behaviour can be detected. The Tetra mesh has the best efficiency. Just a bit below are the hybrid-hexa meshes. The poly meshes have obviously the worst computational efficiency, but also by far the lowest mesh element number.

A very interesting summary gives figure 9.108. These are simple averaged values of the drag and lift coefficient deviation for all cars. Simply relying on this figure is a bit misleading. The PolyPoly mesh shows, together with the HexaInterior mesh, the lowest deviation in drag coefficient. But the PolyPoly mesh has a positive deviation for two

[3] Refer to chapter 9.2.

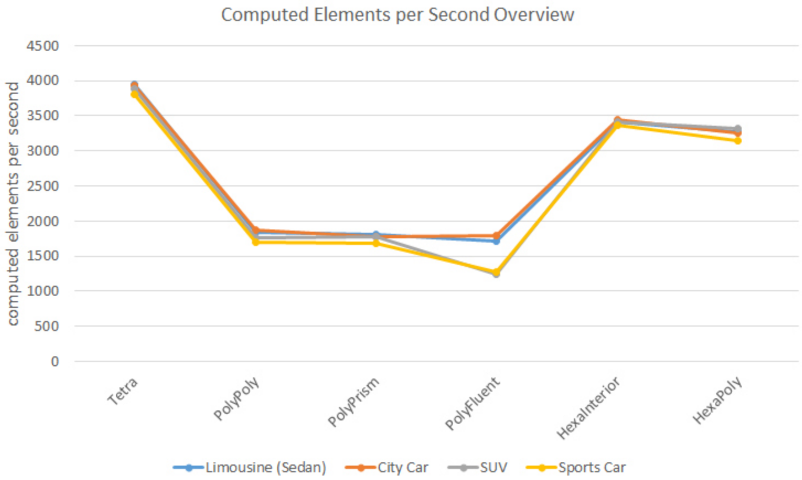


Figure 9.107: Computational efficiency of the different mesh types

cars and a negative deviation for the other two cars (+2.4, -3.4, -2.2, +4.8 per cent). The HexaInterior mesh has a positive deviation for three meshes and a negative deviation for just one mesh (-3.1, +1.3, +2.5, +0.3 per cent). This means, the PolyPoly mesh has relatively large deviations in both directions, positive as well as negative, and they more or less cancel each other out. The HexaInterior has mainly relatively small positive deviations and just one negative. So overall, the HexaInterior is much more accurate, even if figure 9.108 tells something different.

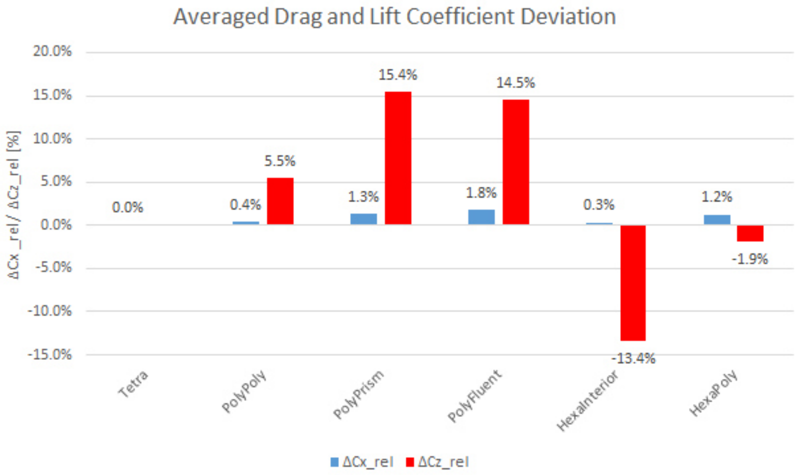


Figure 9.108: Averaged drag and lift coefficient deviation

By investigating the absolute averaged drag and lift coefficient deviation[4] according to

[4] Absolute averaged means, that all the absolute values of each deviation are summed and averaged.

figure 9.109, a clearer picture can be drawn. Both hybrid-hexa meshes deliver the most exact result related to the Tetra base mesh.

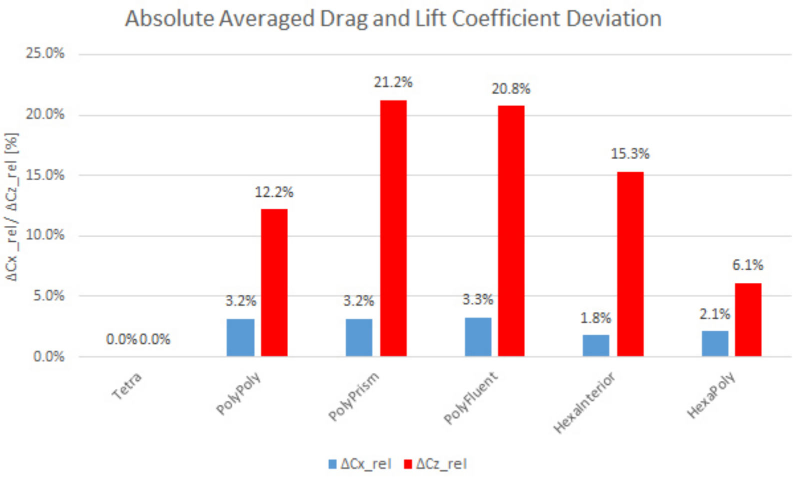
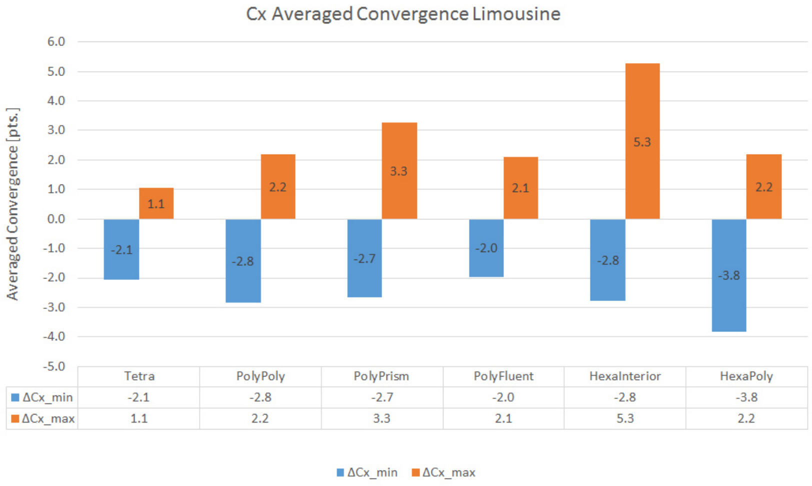


Figure 9.109: Absolute Averaged drag and lift coefficient deviation

Figure 9.110 shows the convergence range of the limousine period averaged cx. It should be kept in mind, that the limousine was simulated with 6 000 iterations and all other cars with just 3 000 iterations. The reference cx value was averaged over the last 3 300 iterations. Surprisingly, both hybrid-hexa meshes have the largest convergence range of all meshes tested. Both of them oscillating in a corridor of six or more points. With a maximum range of 3.2 points, the Tetra mesh offers the closest (but to keep in mind, not the quickest) convergence.



	Tetra	PolyPoly	PolyPrism	PolyFluent	HexaInterior	HexaPoly
ΔCx_min	-2.1	-2.8	-2.7	-2.0	-2.8	-3.8
ΔCx_max	1.1	2.2	3.3	2.1	5.3	2.2

Figure 9.110: Convergence behaviour of the limousine

Compared to the limousine, the city car delivers a complete different view (figure 9.111). Even by looking at the Tetra convergence, something looks weird. The minimum cx value as well as the maximum cx value are both negative. The reason for this can be found in the averaging method of the reference value. Just the last 300 iterations are averaged to determine the reference cx value. As the Tetra mesh shows a very slow convergence, most of the values are below this reference value. This is an indicator, that averaging should be applied over more iterations. This coincides with paper [51]. However, the convergence range is very tight in both hybrid-hexa cases. Just 0.3 and 0.6 points over the last 2 500 iterations.

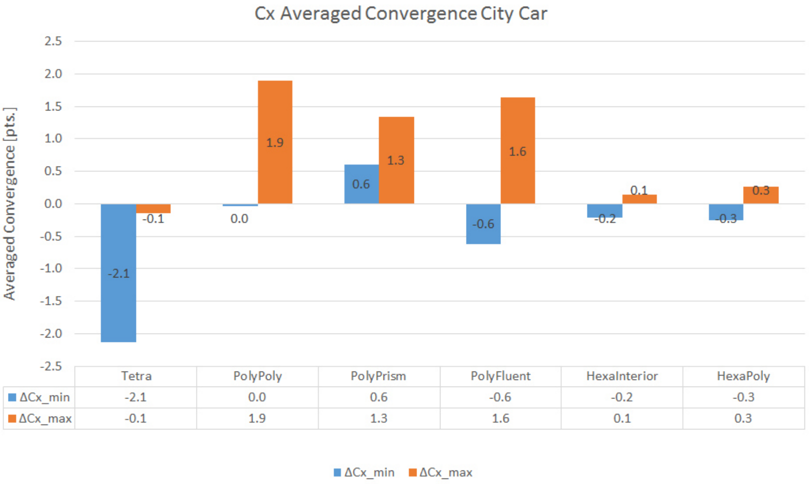


Figure 9.111: Convergence behaviour of the city car

A similar behaviour as in with the city car case can be detected for the SUV according to figure 9.112. In both, PolyPoly and PolyPrism, as well as in HexaInterior, the minimum and the maximum cx values have the same prefix. This means again, that the averaging of the reference value was applied over too less iterations (last 300). However, the PolyPoly mesh shows the tightest corridor with a range of just 1.6 points (-0.5 to -2.1 points). But this corridor lies completely below the reference value. A similar result is delivered by both hybrid-hexa meshes (range of 2.2 and 2.1 points), but with a corridor closer to the reference value.

Also at the sports car (figure 9.113), a similar behaviour as already seen in the city car and the SUV case can be detected. Three of six meshes have the same prefix for the minimum as well as for the maximum cx value. However, again both hybrid-hexa meshes deliver the tightest ranges.

Figure 9.114 shows the averaged convergence range. This is averaged over all cars. With an averaged range of less than 2.5 points over four different cars each with 2 500, respectively 5 500 iterations (limousine), the HexaPoly mesh delivers the best universal convergence behaviour. The residual plots of each car confirm this result.

Figures 9.115 and 9.116 present a general overview of all meshes for all cars. Figure

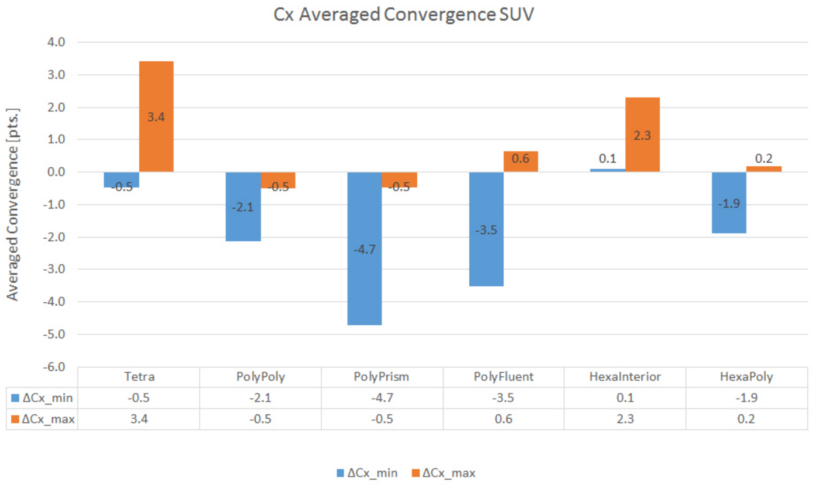


Figure 9.112: Convergence behaviour of the SUV

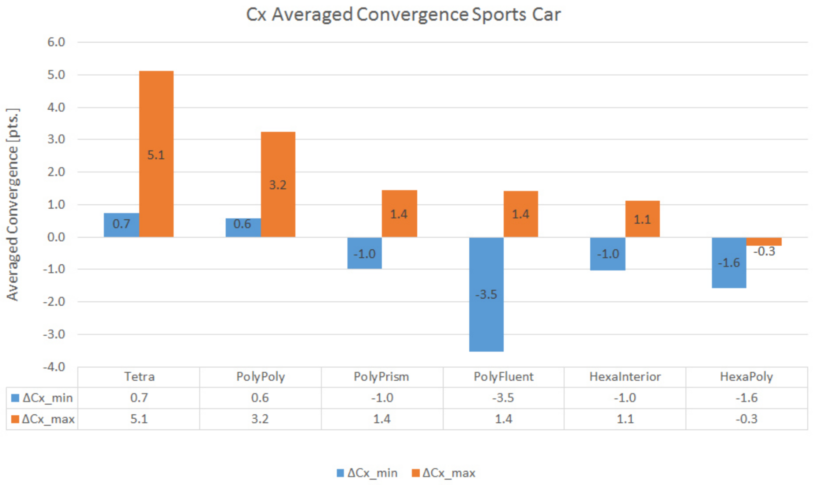


Figure 9.113: Convergence behaviour of the sports car

9.115 shows the convergence range of the averaged cx value car by car. Aside from the limousine case, the hybrid-hexa meshes offer a very tight range for all other cases, but this could change, if 6 000 iterations are applied to the other cars too. Figure 9.116 shows the same information as figure 9.115, but mesh by mesh.

Figure 9.117 shows the maximum relative deviation of the instant cx value, averaged over all four cars. Again both hybrid-hexa meshes deliver the best results. Not too far off is the Tetra mesh and all three poly cases are quite a bit off.
When looking at the numbers, both hybrid-hexa meshes offers the best results in terms of convergence as well as in terms of accuracy related to the Tetra mesh. Especially

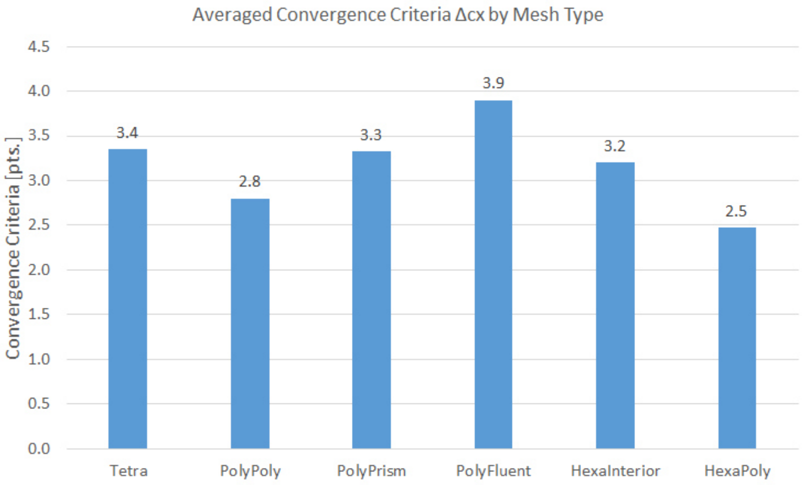
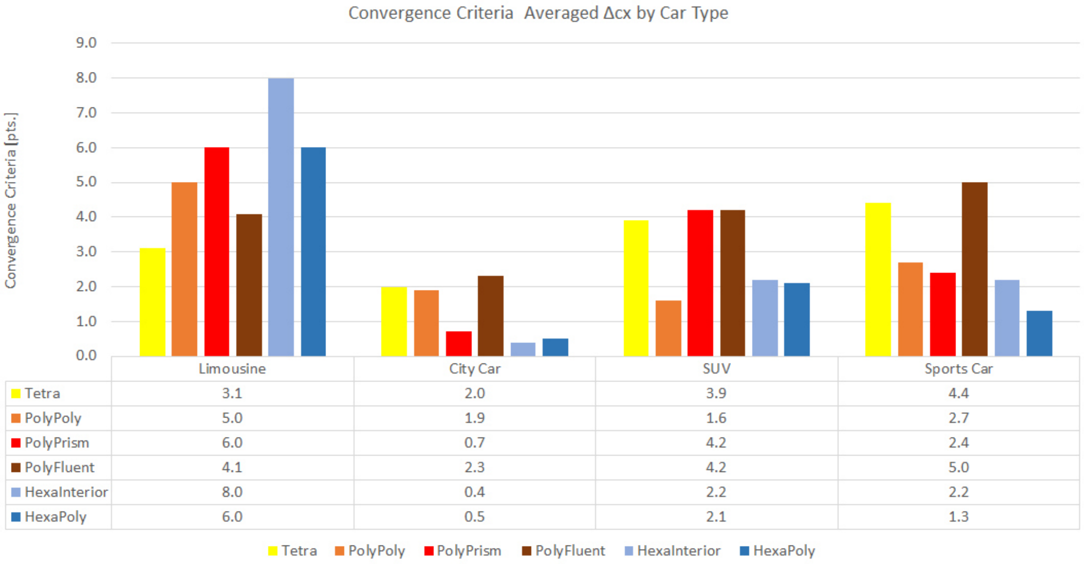


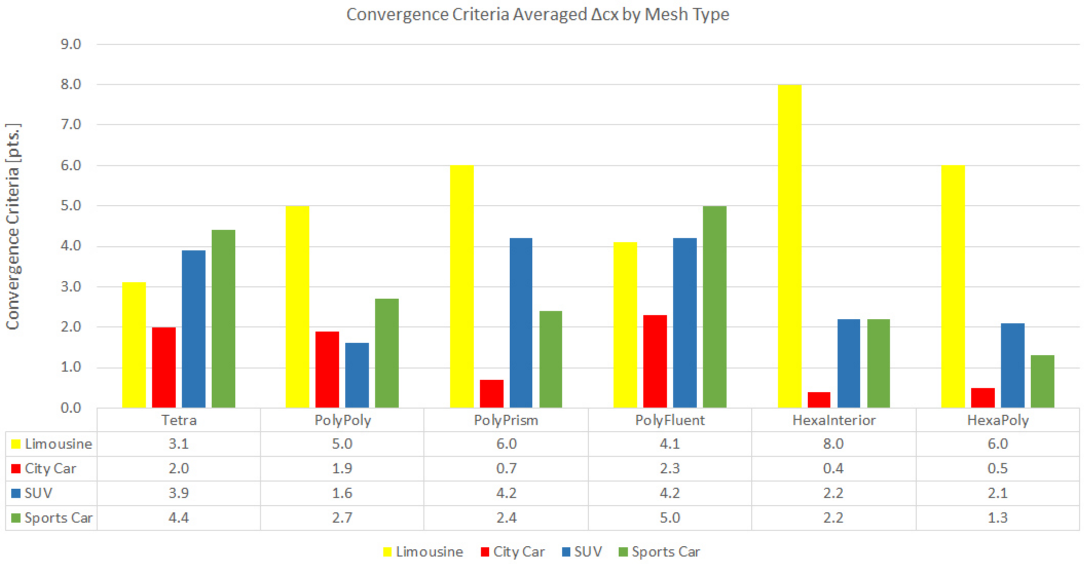
Figure 9.114: Averaged convergence by mesh type



	Limousine	City Car	SUV	Sports Car
Tetra	3.1	2.0	3.9	4.4
PolyPoly	5.0	1.9	1.6	2.7
PolyPrism	6.0	0.7	4.2	2.4
PolyFluent	4.1	2.3	4.2	5.0
HexaInterior	8.0	0.4	2.2	2.2
HexaPoly	6.0	0.5	2.1	1.3

Figure 9.115: Convergence of cx car by car type

the quick convergence of the residuals is impressive for the HexaInterior as well as for the HexaPoly mesh. The poly meshes usually also converge faster than the Tetra mesh, but their accuracy is often far away from the hybrid-hexa meshes. The bad accuracy could maybe be led back to their large resolution. Two refinements were applied to the PolyPrism mesh of the sports car, but simulation were not done at the time this thesis was handed in.



	Tetra	PolyPoly	PolyPrism	PolyFluent	HexaInterior	HexaPoly
🟨 Limousine	3.1	5.0	6.0	4.1	8.0	6.0
🟥 City Car	2.0	1.9	0.7	2.3	0.4	0.5
🟦 SUV	3.9	1.6	4.2	4.2	2.2	2.1
🟩 Sports Car	4.4	2.7	2.4	5.0	2.2	1.3

Figure 9.116: Convergence of cx mesh by mesh type

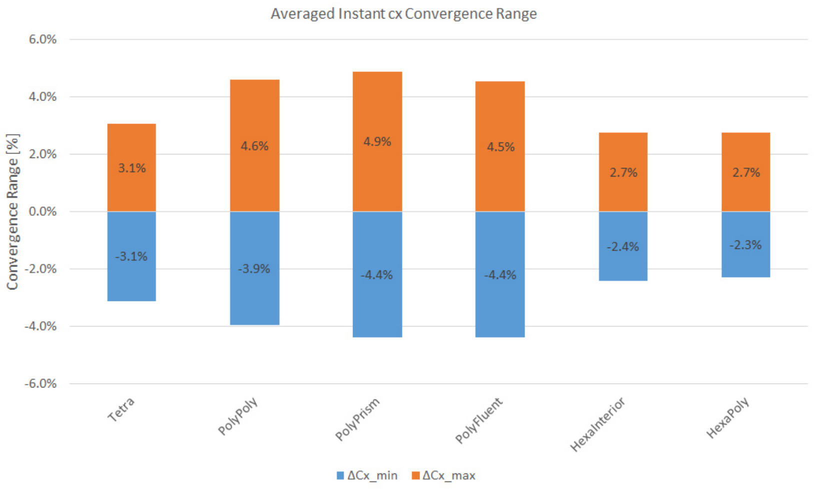


Figure 9.117: Averaged relative convergence by mesh type

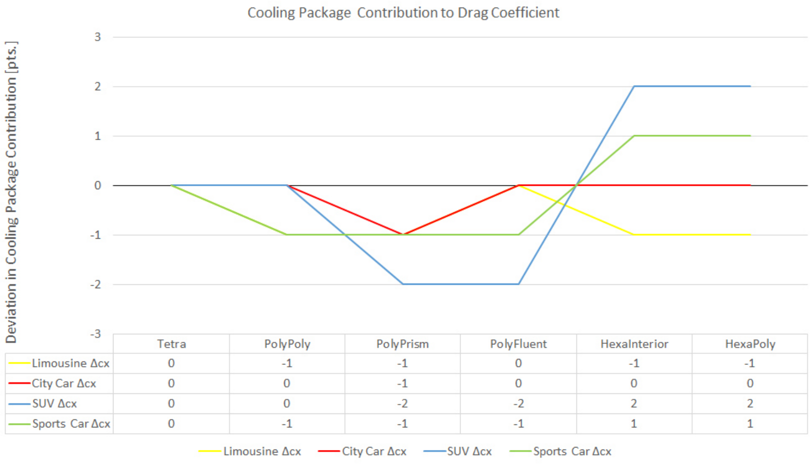
Graphics

It is difficult to judge from the flow figures. The Tetra mesh obviously shows the most flow details. The PolyPrism shows an approach to offer all the flow details which can be seen at the Tetra mesh. Presumably due to the coarse resolution of the PolyPrism mesh (in fact all poly meshes), the flow details are not as pronounced as in the Tetra case. The hybrid-hexa meshes show a similar flow field as the PolyPoly and the PolyFluent. But due to their fine prism layer, the flow separating area at the rear window and tailgate

is more like in the Tetra and the PolyPrism case. The isosurfaces of cptot=0 is quite the same in the Tetra as well as in the HexaInterior and the HexaPoly meshes. All poly meshes deliver a larger isosurface here.
However, due to its lower cell number compared to the HexaInterior mesh and some advantages in convergence behaviour, the HexaPoly mesh is chosen as mesh for the transient simulation.

Cooling Package Contribution

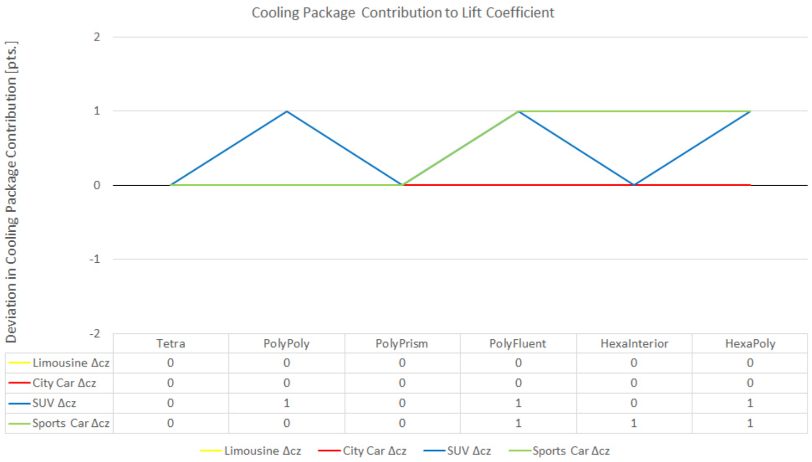
The cooling package is presented at the end on purpose. The changes in drag as well as lift coefficient of the cooling package contribution between the various meshes are only marginal.
At the cooling package for the Tetra, the PolyPrism and both hybrid-hexa meshes are exactly the same. All radiator, condenser and intercooler volumes consist of oriented pentahedral elements. The Fluent poly conversion converts the whole domain, including layer mesh and cooling package volumes. So in PolyFluent, the cooling package volumes consist of oriented polyhedral elements, which are in fact prisms with hexagonal base area.
It is a bit different with the PolyPoly case. On the limousine and the SUV, the cooling package was also converted to poly. On the city car and on the sports car, in contrast, the cooling package was not converted to poly and stayed in its initial volume of oriented pentahedrals. This was done to examine, if there is a difference in different cooling package meshing approaches.
In general, the changes in cx as well as cz contribution of the cooling package for the various mesh types are negligible. On the most cars, it is one or less points. Refer to figures 9.118 and 9.119 for actual values.



	Tetra	PolyPoly	PolyPrism	PolyFluent	HexaInterior	HexaPoly
Limousine Δcx	0	-1	-1	0	-1	-1
City Car Δcx	0	0	-1	0	0	0
SUV Δcx	0	0	-2	-2	2	2
Sports Car Δcx	0	-1	-1	-1	1	1

Figure 9.118: Contribution to drag coefficient for different mesh types

It can be seen, that the polyhedral meshes have a tendency to decrease the cooling package contribution to the overall drag coefficient when compared to the Tetra mesh. The hybrid-hexa meshes, in contrast, have a tendency to increase the cooling package contribution when compared to the Tetra mesh. However, the changes in the drag coefficient are plus and minus two points respectively, in maximum. If the SUV is neglected[5], the deviation is not more than one point.
Referring to figure 9.119, the changes in the contribution to the lift coefficient are, as expected, negligible. At the limousine and the city car, for all mesh types the cooling package has the same contribution to the lift coefficient. At the SUV and at the sports car, the contribution varies by just one point for some of the poly and hybrid-hexa meshes.



	Tetra	PolyPoly	PolyPrism	PolyFluent	HexaInterior	HexaPoly
Limousine Δcz	0	0	0	0	0	0
City Car Δcz	0	0	0	0	0	0
SUV Δcz	0	1	0	1	0	1
Sports Car Δcz	0	0	0	1	1	1

Figure 9.119: Contribution to lift coefficient for different mesh types

However, the meshing approach of the cooling package does not have a big impact on its contribution to the lift as well as to the drag coefficient. This is an important information from the position of which type of mesh is gonna be used. When using a polyhedral mesh, the PolyFluent conversion is the most efficient one, as it can be executed in the computer cluster, which saves time. The PolyFluent conversion converts the whole domain, including the cooling package volumes, into polyhedrons. As the figures above show, the type of mesh does not have a big impact on the cooling package contribution to drag as well as lift coefficient, this is not an important (or influential) mesh type selection criterion.

[5] It should still be kept in mind, that the SUV has a cooling package, consisting of oriented polyhedrons instead of oriented pentahedrons in the PolyPoly and PolyPrism mesh.

9.2 Unsteady SAS

Two transient simulations were performed on the sports car. At first, the HexaPoly mesh was chosen and, as a reference value, the Tetra mesh was also simulated with the transient SAS. The start for the transient simulations were the steady-state .cas[6] and .dat[7] files of the the Tetra and the HexaPoly mesh respectively. This was done to save computing time. If the transient simulation would have been started from zero, the whole transient process of accelerating the flow in the entire domain to the inlet velocity, would need to be calculated. Starting from the steady-state solution saves this process. Referring to table 9.5, the Tetra mesh has 76.7m cells and the HexaPoly mesh has 65.5m cells. The Tetra steady-state simulation ran for 05:36 hours, the HexaPoly steady-state simulation ran for 5:48 hours. The simulation settings for the transient simulation are described in chapter 8.2.

Mesh type Sports car	Cell number	Computing time	CPU hours	Data produced
Tetra	76 664 114	05:35:38 h	1 074	9.2 GiB
HexaPoly	65 477 795	05:47:45 h	1 113	8.6 GiB

Table 9.5: Computational expenses of the steady-state simulation for the mesh types used for the transient SAS

The HexaPoly mesh was chosen, because of its computational efficiency, convergence behaviour and accuracy. The Tetra mesh was also simulated with the transient SAS to have a reference value. In contrast to the steady-state RANS simulations, the transient SAS were executed using Fluent R17.2 instead of Fluent R15.0. All post-processing stuff was executed using a HP Z820 workstation. Videos were generated for cptot=0, Q-criterion and vorticity, each in front and rear isometric view. They can be found on the hpc server at *Magna Steyr*. The post-processing was done using ANSYS CFD-Post. Table 9.6 shows the computing time for the transient SAS.

Mesh type Sports car	Cell number	Computing time[a]	CPU hours	Data produced
Tetra	76 664 114	49:21:33 h	9 477	36.4 GiB
HexaPoly	65 477 795	36:31:06 h	7 012	30.7 GiB

[a] Both meshes were actually simulated in two segments. The denoted time includes twice the initialisation of the solution and twice the writing of the .dat file. This means, the actual computing time is a few minutes less.

Table 9.6: Computational expenses of the transient SAS simulation for both mesh types

[6] The .cas file is the simulated case and contains the whole geometry and simulation settings. The typical size for a zipped .cas file is between 1.7 GiB to 2.0 GiB.

[7] The .dat file contains the simulation results of the corresponding .cas file. The typical size for a zipped .dat file of a steady-state solution is between 7.4 GiB and 9.0 GiB. The size of a .dat file of a transient solution can easily increase up to 30 GiB.

Coefficients and Convergence Behaviour

Figure 9.120 shows a comparison of both, the Tetra and the HexaPoly mesh, for the steady-state as well as for the transient solution. The base is again the steady-state Tetra mesh result. It can easily be seen, that the steady-state HexaPoly solution is closest to the steady-state Tetra solution. Considering the drag coefficient, both transient solutions are over ten points off the reference steady-state Tetra solution. However, this is equal to a deviation of less than five per cent to the absolute value. It is noticeable, that the lift coefficient of the transient solution is a lot below the lift coefficient of both steady-state solutions. Anyway, this behaviour coincides exactly with the behaviour of transient simulations detected in former investigations like for example presented in paper [12] and [77]. A more detailed comparison of these four cases and a WT experiment is presented in chapter 9.3.

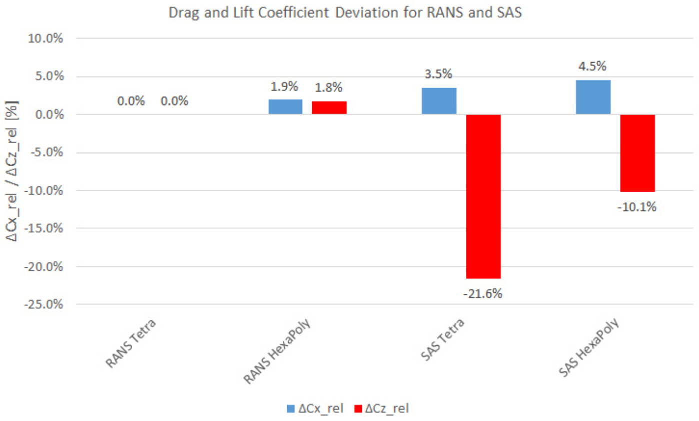


Figure 9.120: Drag and lift coefficient deviation of the sports car for steady-state as well as for transient solution

Figures 9.121 to 9.124 show the convergence behaviour for both, Tetra and HexaPoly, meshes of the sports car transient SAS simulation. It should be kept in mind, that the term "convergence" in connection with transient simulations does not have the same meaning as it has for steady-state simulations. In transient simulations, the fluctuations of the drag or lift coefficient present actually the actual fluctuations over a time period. In steady-state simulations, each data point in the diagram presents the solution of one iteration. However, in steady-state as well as in transient problems, the values are averaged over a range of iterations (steady-state) and time steps (transient), respectively. In figures 9.121 and 9.123 the convergence of cx is shown. It is clearly visible, that there is an increase of cx between the steady-state solution and the transient solution. This is due to the fact, that steady-state simulations under-predict the TKE. The transient method, in contrast, resolves a specific fraction of the turbulent spectra. Thus, the TKE is higher in a transient simulation which results in an increase of drag.

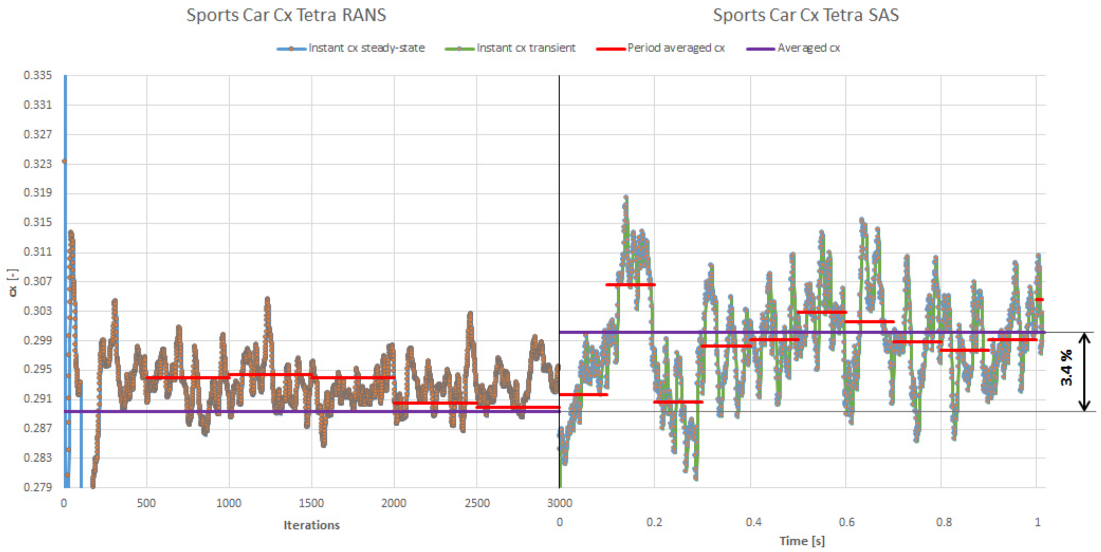


Figure 9.121: Convergence plot of cx of the sports car Tetra mesh for steady-state and transient simulation

Figure 9.121 shows the cx convergence plot of the Tetra mesh of the sports car. Both, the steady-state and the transient case, are shown. The difference of the averaged steady-state cx and the transient cx is 3.4 per cent (10.8 points). The steady-state cx is averaged over the last 300 iterations. The transient cx is averaged over the last thousand time steps. This equals an averaging over half a second real-time. It is also obvious, that the fluctuations in the transient case are a lot larger than in the steady-state case. Considering the maximum of 20 iterations per time step and the 3 000 iterations of the steady-state case, about 43 000 iterations were executed in this case. Figure 9.122 shows the corresponding convergence plot of the residuals for the transient case. The saw tooth thread profile of the residuals shows that the simulation process is stable.

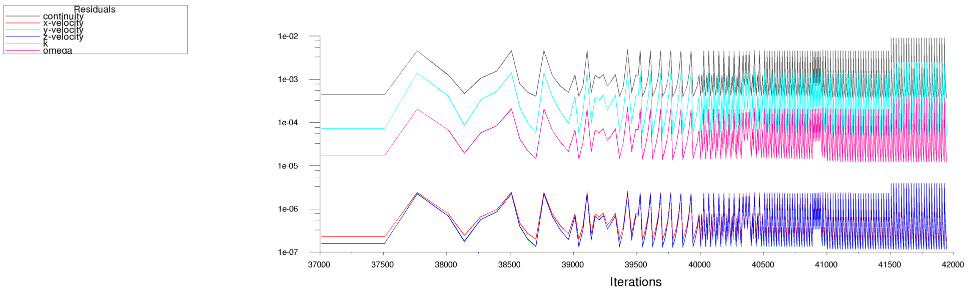


Figure 9.122: Convergence plot of the residuals of the sports car Tetra mesh in transient mode

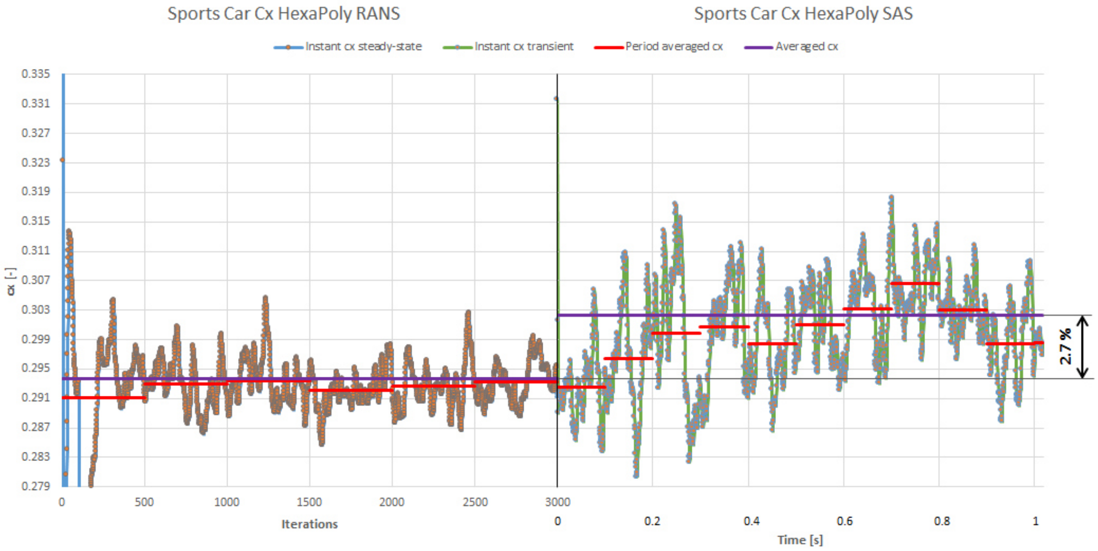


Figure 9.123: Convergence plot of cx of the sports car HexaPoly mesh for steady-state and transient simulation

Figure 9.123 shows the cx convergence plot of the HexaPoly mesh of the sports car. Both, the steady-state and the transient case, are shown. Both, the steady-state and the transient case, are shown. The difference of the averaged steady-state cx and the transient cx is 2.7 per cent (8.6 points). The steady-state cx is averaged over the last 300 iterations. The transient cx is averaged over the last thousand time steps, which equals an averaging over half of a second real-time. The first time step of the transient simulation is a runaway value, the rest of the simulation looks similar to the transient Tetra case. Same as in the Tetra case, all in all about 43 000 iterations are executed in this case. Figure 9.124 shows the corresponding convergence plot of the residuals for the transient case. The saw tooth thread profile of the residuals shows that the simulation process is stable.

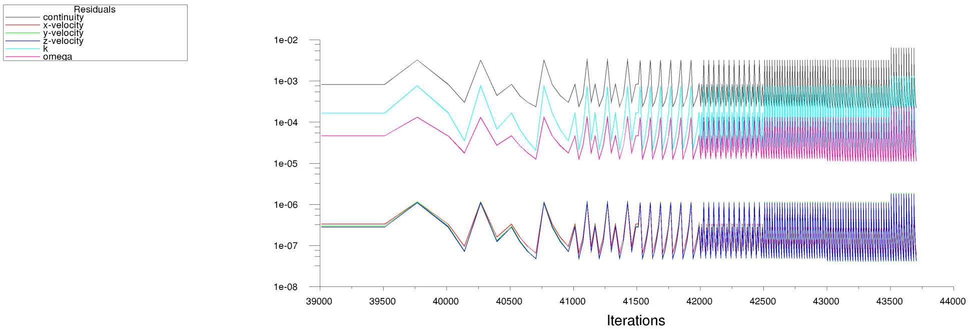


Figure 9.124: Convergence plot of the residuals of the sports car HexaPoly mesh in transient mode

Cpstat Contours Car

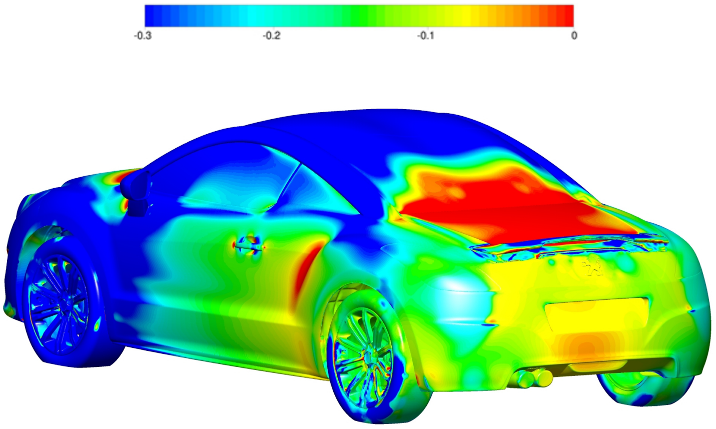


Figure 9.125: Static pressure coefficient contour of the sports car in steady-state mode

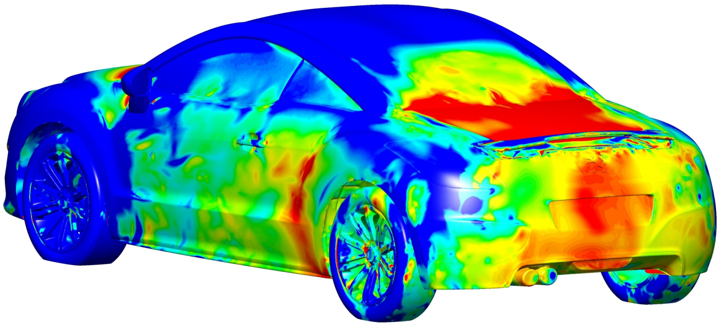


Figure 9.126: Instant static pressure coefficient contour of the sports car in transient mode

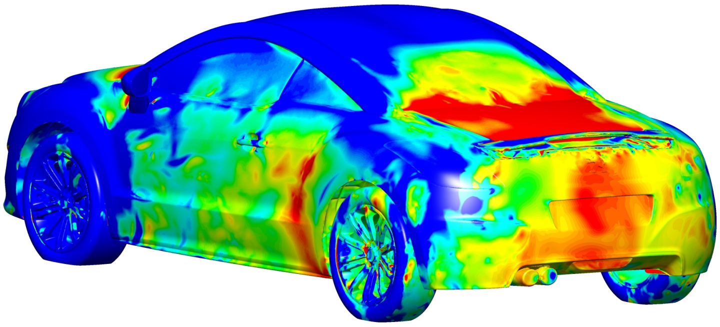
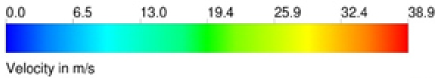


Figure 9.127: Time-averaged static pressure coefficient contour of the sports car in transient mode

Isosurfaces cptot=0, Coloured by Velocity



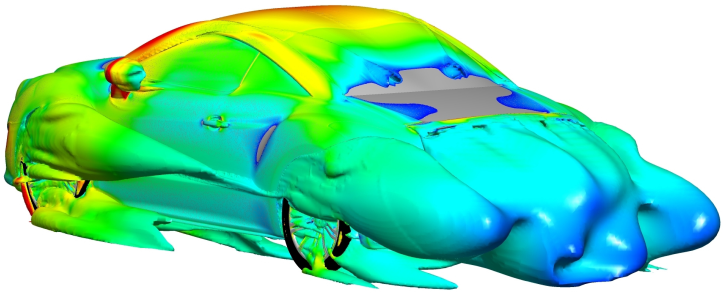


Figure 9.128: Cptot=0 isosurface plot in steady-state mode

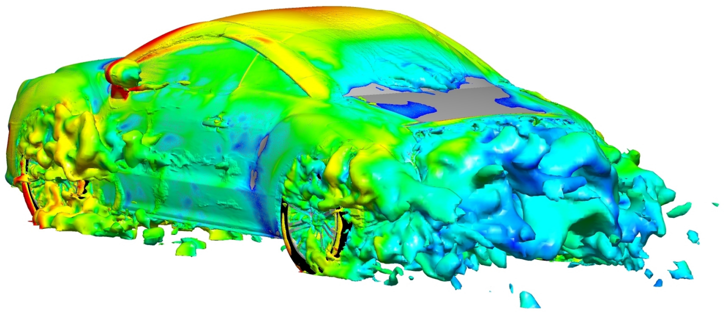


Figure 9.129: Instant cptot=0 isosurface plot in transient mode

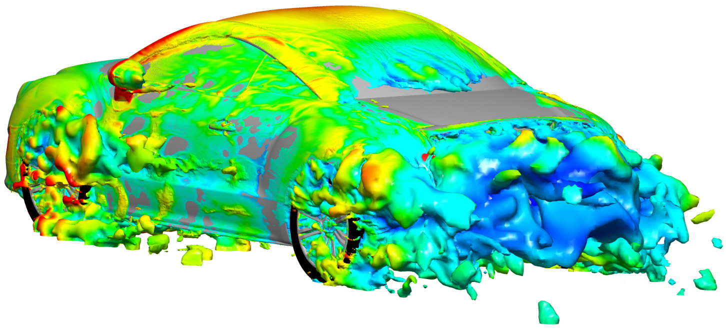


Figure 9.130: Time-averaged cptot=0 isosurface plot rear in transient mode

Skin Friction Coefficient

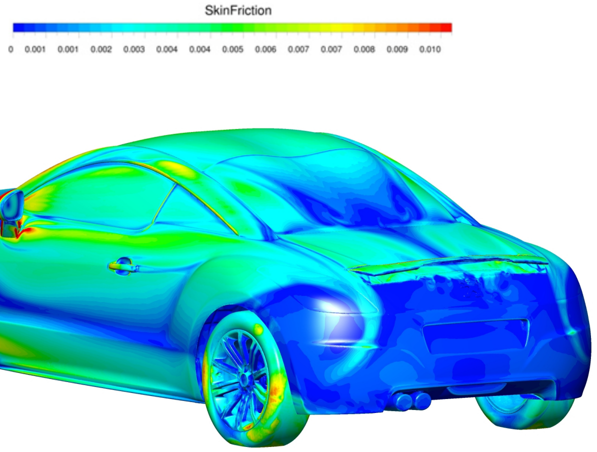


Figure 9.131: Skin friction coefficient of the sports car in steady-state mode

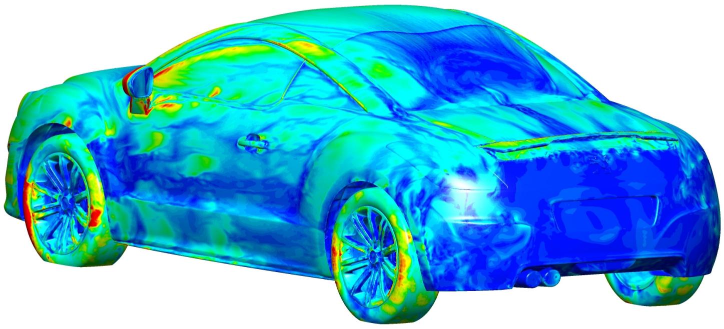


Figure 9.132: Instant skin friction coefficient of the sports car in transient mode

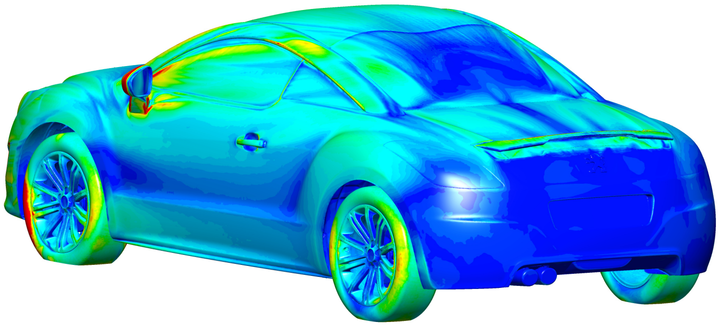
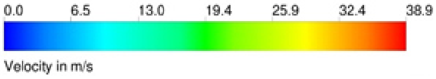


Figure 9.133: Time-averaged skin friction coefficient of the sports car in transient mode

Velocity Contour and Streamlines in y=0-Section



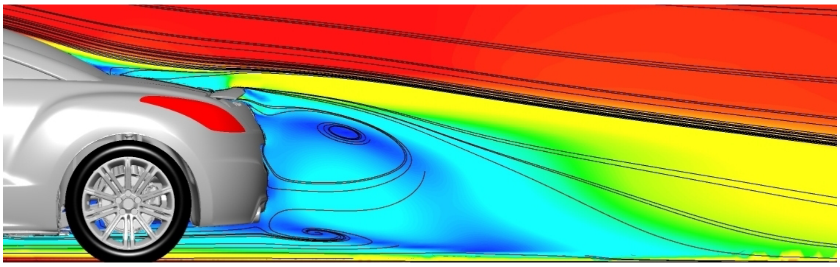


Figure 9.134: Velocity streamlines in the y-plane of the sports car in steady-state mode

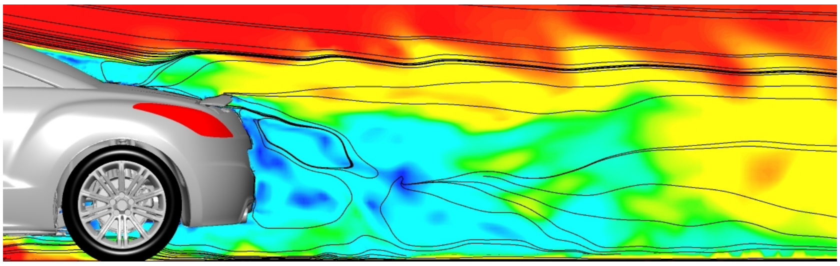


Figure 9.135: Instant velocity streamlines in the y-plane of the sports car in transient mode

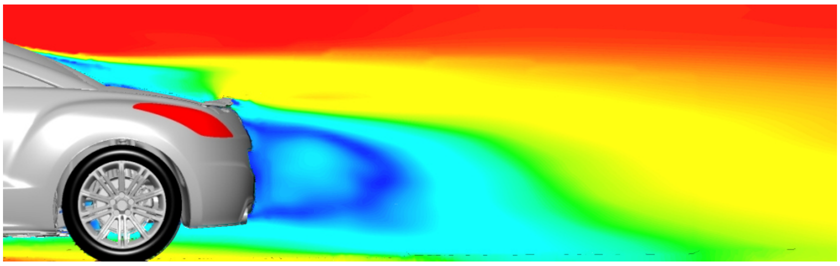
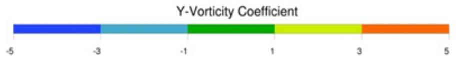


Figure 9.136: Time-averaged velocity in the y-plane of the sports car in transient mode

Y-Component of Vorticity Coefficient



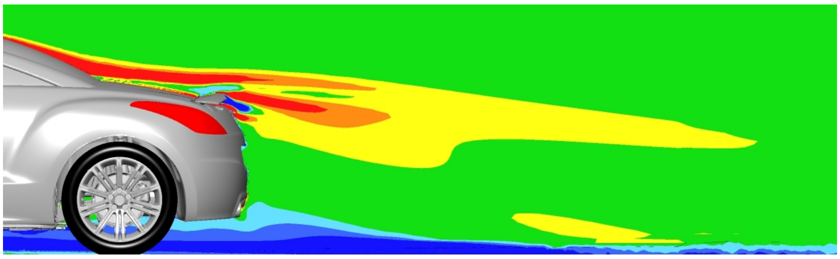


Figure 9.137: Y-component of the vorticity coefficient of the sports car in steady-state mode

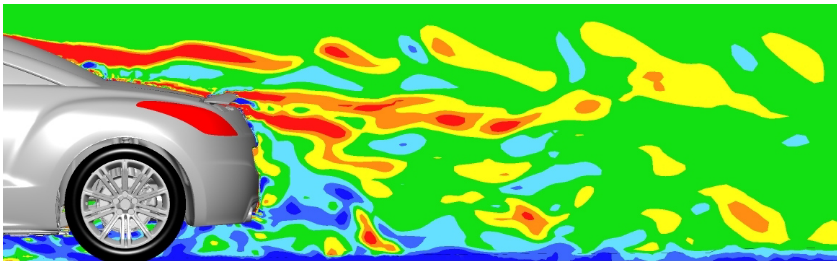


Figure 9.138: Instant y-component of the vorticity coefficient of the sports car in transient mode

All figures presented here, result from the solution of the steady-state and the transient solution of the HexaPoly mesh. Figures were generated as well from the transient solution of the Tetra mesh, but not presented here.
Figures 9.125 to 9.127 show a comparison of the cpstat contour for the steady-state[8], the instant transient[9] and the time-averaged transient[10] solution. The difference between a steady-state and a transient simulation can be seen best when comparing the steady-state (figure 9.125) and the instant transient (figure 9.126) results. Due to the Reynolds-averaging of the steady-state solution, all turbulent fluctuations are filtered out. The transient simulation, in contrast, resolves a specific fraction of the turbulent spectrum. The solution of the steady-state simulation delivers a very smooth picture of the static pressure coefficient contour at each iteration step. The transient solution delivers a very disturbed picture for each time step. If the transient solution is time-averaged, the resulting figure looks similar to the instant transient solution.

[8] This is the solution of the last iteration process of the steady-state simulation.

[9] This is the solution of the last time step of the transient simulation.

[10] This is the time-averaged solution over the entire 2 000 time steps of the transient solution.

Figures 9.128 to 9.130 show a comparison of the isosurface cpstot=0, coloured by the local velocity for the steady-state, the instant transient and the time-averaged transient solution. These are maybe the figures, where the difference between the steady-state and the transient mode is most obvious. The wake looks completely different, when comparing figure 9.128 with figures 9.129 and 9.130. It is evident, that the wake in the transient mode is a lot shorter and has less volume than in the steady-state mode. This should cause a decrease of drag. The wake is also much more distorted than in the steady-state mode. Even in the time-averaged solution, the wake is very distorted.

Figures 9.131 to 9.133 show a comparison of the skin friction coefficient contour for the steady-state, the instant transient and the time-averaged transient solution. These figures present a similar picture as the static pressure coefficient - a very smooth distribution of the coefficient for the steady-state solution and a very disturbed distribution in the instant transient case. The time-averaged transient solution is again much closer to the steady-state solution, but it features again more flow details. Some obvious differences can be detected at the a-pillar and at the rear window. However, qualitative, bot solutions look pretty similar, but in detail, the transient solution delivers much more information.

Figures 9.134 to 9.136 show a comparison of the velocity contour coloured streamlines in y=0 section for the steady-state, the instant transient and the time-averaged transient solution. The instant transient figure looks a bit disturbing. This figure shows all the turbulent fluctuations of one time step in the y=0 section. The time-averaged transient solution, in contrast, delivers a pretty enlightening picture. Same as in the cptot=0 isosurface, a separation can be seen in the middle of the rear window. The wake is, in contrast to the steady-state result, not approaching the ground and looking a bit weaker. Unfortunately, no time averaged steam lines are available neither in Fluent nor in CFD-Post.

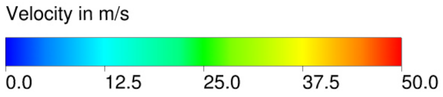
Figures 9.137 and 9.138 show a comparison of the y-component of the vorticity coefficient for the steady-state and the instant transient solution. The vorticity is probably the best or the easiest way to describe vortical structures in the steady-state mode. However, if the two figures, which show both the instant solution of the last iteration and time step, are compared, it gets obvious again, that the steady-state simulation suppresses actually the entire turbulent fluctuations and prohibits the formation of vortical structures. Anyway, for transient observations, more sophisticated tools are available to visualise vortical structures. The best known tools for this are:

- Q-criterion
- Δ-criterion
- λ_2-criterion

In this thesis, exclusively the Q-criterion is used. A mathematical description of the Q-criterion can be found in the appendix. Furthermore, continuative literature for the Δ-criterion as well as for the λ_2-criterion is given.

Isosurfaces Q-Criterion, Coloured by Velocity

Figures 9.139 to 9.142 show the Q-criterion coloured by the velocity magnitude for the steady-state as well as for the transient mode. Each, an isometric front and rear view for both cases are presented. The Q-criterion was created with an iso-value of 0.16 per cent of the maximum of constant turbulence ($3\,900\,\mathrm{s}^{-2}$ for the transient mode). Figures 9.139 and 9.141 show the instant Q-criterion for the steady-state case after 3 000 iteration steps. Figures 9.140 and 9.142 show the instant Q-criterion for the transient case after about 1.0 seconds of simulation time.



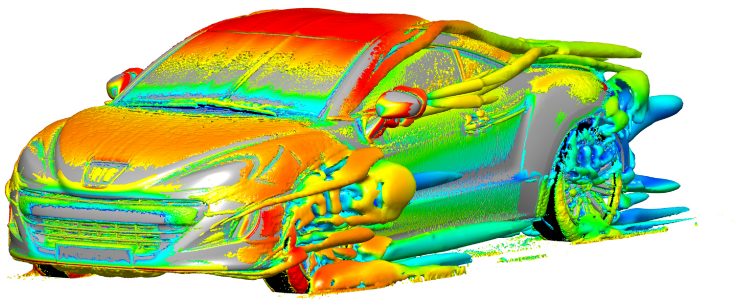


Figure 9.139: Q-criterion plot front with iso-value of 0.16 per cent of maximum turbulence in steady-state mode, coloured by velocity

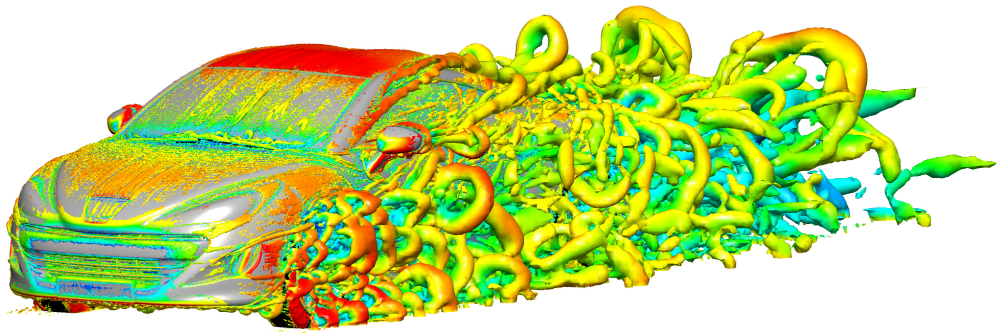


Figure 9.140: Q-criterion plot front with iso-value of 0.16 per cent of maximum turbulence in transient mode, coloured by velocity

The Q-criterion shows impressively the advantage of a transient simulation over a steady-state simulation. The steady-state simulation captures just a very little fraction of the turbulent spectrum. The transient simulation, in contrast, covers a high amount of the turbulent spectrum.[11] Figure 9.141 and 9.142 show impressively the stability of the a-pillar vortex. The a-pillar vortices stay in almost the same position when changing the steady-state model to the transient SAS mode. The wake in the transient mode is much more distinctive when compared to the steady-state mode.[12] Only the strong a-pillar vortical structures stay the same in the transient case. A video was created from a series of these pictures which shows the transition from the steady-state to the transient mode beautifully.

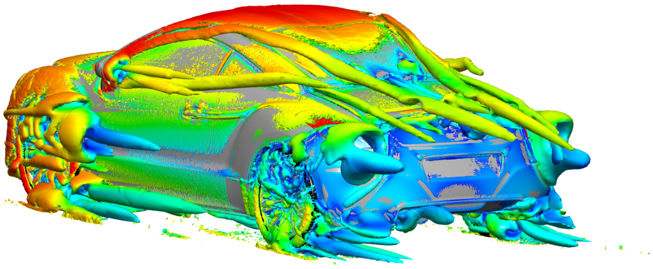


Figure 9.141: Q-criterion plot rear with iso-value of 0.16 per cent of maximum turbulence in steady-state mode, coloured by velocity

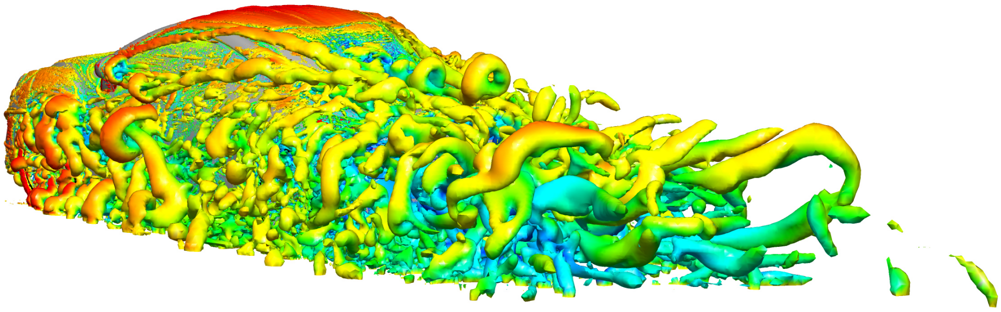


Figure 9.142: Q-criterion plot rear with iso-value of 0.16 per cent of maximum turbulence in transient mode, coloured by velocity

[11] The resolution of the turbulent spectrum of a transient simulation can be adjusted to a certain degree by modifying the mesh resolution. At constant boundary conditions, a mesh refinement lowers the CFL number and increases the degree of resolution of the turbulent fluctuations.

[12] This confirms the fact, that a steady-state simulation under-predict TKE due to averaging the pressure and velocity over the whole turbulent spectrum.

Summary of Transient Simulation

To summarise the results of the transient simulation, it can be resumed, that the time demand of a transient simulation is much less than expected. With a run-time of only 36.5 hours for one second of transient simulation, the SAS took only about six times more computing time than the steady-state simulation. However, when creating a video for the transient simulation, the computing time increases rapidly. This is described in the following chapter. The amount of produced data is much larger and more comprehensive. Despite this, the result needs to be time-averaged to get useful information. The instant values just show the solution of one time step.

As described above, the transient simulation also captures much more flow details. Especially in the wake, the difference is impressive. Due to the consideration of a decent amount of the turbulent spectrum, a high amount of the turbulent fluctuations are captured and can be visualised for example with the Q-criterion.

Furthermore, the HexaPoly mesh is, compared to the Tetra mesh, computational much more efficient. In the steady-state case, the Tetra mesh was slightly more efficient than the HexaPoly mesh. In the transient mode, the HexaPoly has an efficiency, which is almost twice as good as the efficiency of the Tetra mesh.

A comprehensive comparison of both transient simulations and their steady-state counterparts can be found in chapter 9.3.

Time Demand for Transient Video Creation

The possibility of the creation of a real-time video for a transient simulation is a big advantage over a steady-state solution. But this comes with an intensive increase of time demand for the computing process. A frame rate has to be defined, for which range of time steps, a picture should be produced by the solver. The six videos created at the HexaPoly mesh for this thesis were recorded with a frequency of 400 Hz, which equals one picture for each 2.5 ms. This is necessary to get a smooth video. One time step (0.5 ms) requires a computing time of 61 seconds for 20 iterations. Each five time steps, six pictures were generated for the video creation. To generate one picture, 135 seconds are required by average. Five time steps require 305 seconds of computing time. After each fife time steps, six pictures were generated, which requires approximately 810 seconds. This means, to produce these six videos, it takes about 265 per cent more computing time than simulating without generating a video. Therefore, when simulating transient, the video generation should be handled with caution, not to increase the computing time extensively. Refer also to table 9.8 for exact numbers.

9.3 Steady-State RANS Versus Transient SAS

Table 9.7 shows a comparison of the drag as well as the lift coefficient for the steady-state and the transient simulation for both tested mesh types. An additional comparison to the WT experiment is given as a reference. Due to non-disclosure agreements, the exact values are not given. Thus only a relative comparison is presented. It can be seen, that both steady-state and both transient solutions come very close to the experimental value in terms of drag coefficient. The steady-state HexaPoly is even a direct hit. Interestingly, in terms of lift coefficient, the steady-state solutions are closer than the transient solutions. However, also the WT is only a (physical) experiment and presents simply another case. Furthermore, the WT as well as the simulations are highly dependent on boundary conditions. Thus, even a deviation of 28 per cent in the transient Tetra solution of the lift coefficient (refer to figure 9.143) seems not to be an extraordinary error.

Model	Sports Car			
	c_x	$\Delta c_{x,rel}$	c_z	$\Delta c_{z,rel}$
Exp.[1]	n/a	0.0 %	n/a	0.0 %
$k-\omega$-SST Tetra	0.312	–1.9 %	0.227	–8.1 %
$k-\omega$-SST HexaPoly	0.318	0.0 %	0.231	–6.5 %
SAS Tetra	0.323	+1.6 %	0.178	–27.9 %
SAS HexaPoly	0.326	+2.5 %	0.204	–17.4 %

[1] The experimental data were accessible to the author of this thesis. However, due to non-disclosure agreements the exact values are not allowed of being published.

Table 9.7: Comparison of the drag and lift coefficients of the sports car for k-omega and SAS model in comparison with the WT experiment

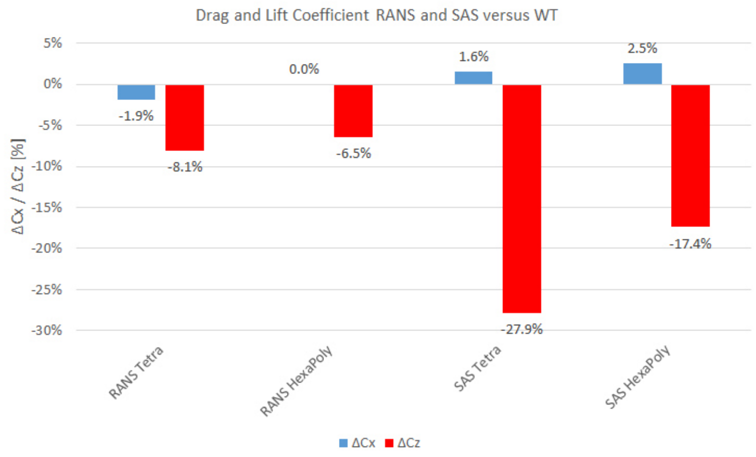


Figure 9.143: Relative deviation of drag and lift coefficient compared to the WT experiment for the sports car

Table 9.8 shows a comparison of the computational expenses for the steady-state and the transient simulation for both tested mesh types.[13] Both steady-state simulations require approximately the same amount of computing time. This has already been investigated in chapter 9.1. An interesting fact is, that the Tetra mesh is much less efficient in transient simulation compared to the HexaPoly mesh. The HexaPoly mesh had a runtime of just 36.5 hours, which is only six times more than the steady-state simulation. The Tetra mesh, in contrast, required about 40 per cent more computing time than the HexaPoly mesh for the transient simulation.
An additional case is presented in 9.8, considering video creation in transient simulations. In this case, six videos were created (isosurfaces cptot=0, vorticity and Q-criterion each in front and rear isometric view). At the end of chapter 9.2, a detailed description of the video creation is given. However, for not even half a second of simulation time (exactly 0.4875 seconds) with creating six videos, the simulation took almost double the time of a one second transient simulation (63 hours versus 36.5 hours). The relative costs of computing time increase about four times compared to a simulation, when not creating videos. So, video creation should always be treated with care, not to increase the computing time extremely.

Method	Cells	Cores	Temporal scheme	Time per iteration	Compute time	Relative cost
SST-RANS Tetra	$76.7 \cdot 10^6$	192	Steady	6 s (3 000 it.)	05:35:38 h	1
SST-RANS HexaPoly	$65.5 \cdot 10^6$	192	Steady	6.5 s (3 000 it.)	05:47:45 h	1.04
SAS Tetra	$76.6 \cdot 10^6$	192	Transient $5 \cdot 10^{-4}$	84 s (2 028 t. steps)	49:21:33 h	8.70
SAS HexaPoly	$65.5 \cdot 10^6$	192	Transient $5 \cdot 10^{-4}$	61 s (2 035 t. steps)	36:31:06 h	6.24
SAS + 6 Videos HexaPoly	$65.5 \cdot 10^6$	192	Transient $5 \cdot 10^{-4}$	61 s (975 t. steps)	62:47:42 h	23.03

Table 9.8: Computational expense of RANS and SAS simulations for the sports car

To conclude this comparison, it got clear that the time demand of a transient simulation using SAS is much less than expected. At least at this car type (fastback), the results of the transient simulations are not closer to the WT data than the steady-state solutions. However, the main add-on of the transient simulation are not coercively the numbers, but the quality and and degree of detail of the flow field. With this information, the flow field around the car can be investigated on stability. Also the influence of the wake on following cars could be examined. Furthermore, a transient simulation could deliver data regarding the influence of side wind on the cars longitudinal stability.

[13] To have a base of comparison, the 3 000 iterations of the RANS simulation were equalised to 2 000 time steps (one second real-time) of the transient SAS.

10 Conclusion

One goal in the frame of this thesis was firstly to examine the various unsteady methods in CFD. In the course of the comprehensive theory chapter, the most important methods (URANS, LES, ZLES, DES, PANS, PITM, DNS, LBM) were explained and the current development was presented, occasionally with the help of some examples. Comparisons and assessments between the methods were presented. Following this research work, the PANS model seemed to be the most capable hybrid RANS-LES method for the near future. Methods like ZLES and DES divide the computational domain in pure RANS and pure LES areas without any consideration of the state of the flow field. The areas are divided randomly, without any solid mathematical background. So the turbulence is just resolved locally. The PANS and the PITM model in contrast, offer a seamless transition between RANS and DNS behaviour in the entire domain. The degree of resolving turbulence is user-defined, seamless and is valid globally. Some history and the most important theory basics were also explained in this chapter.

Following this, a complete overview was given on the models available in ANSYS Fluent. All methods implemented in ANSYS Fluent R17.2 were described. At the end of this chapter, an organisation chart of the whole model structure of ANSYS Fluent is shown. Two similar organisation charts for CD-adapco STAR-CCM+ v11.04 and for OpenFOAM v4.1 can be found in the appendix.

Some aspects regarding the numerics were listed afterwards including general information about mesh generation for unsteady simulations and some references on efficient mesh element types. Additionally, a few words on choosing the correct time step for transient simulations were written.

Before joining the practical part, a few examples of former investigations in unsteady simulations were presented. The first example was a comparison of RANS and DES methods, using the Ahmed body and the DrivAer car model. Additionally, some mesh refinements were tested. The Ahmed body was simulated with a structured mesh as well as with a polyhedral mesh. The DrivAer model was simulated, using polyhedral meshes with different resolutions. The result was, that some RANS models could calculate the drag coefficient very well. Some RANS models were also capable of predicting the lift coefficient fairly accurately. But none of these RANS models could reliably and accurately predict drag coefficient as well as lift coefficient for both car specifications (estate and notchback). The DES methods, in contrast, were able to predict both coefficients for both car specifications pretty accurately. However, the computing time of the DES methods were 17 times higher than with the RANS models. This study used CD-adapco STAR-CCM+ as well as freeware EFD Code_Saturn software packages.

Another paper presenting a comparison of the PANS and a RANS model was cited. Within there, the unbeatable advantage of this bridging hybrid RANS-LES methods was shown. Unfortunately, there was no information about the required time amount of the PANS model compared tho the RANS model. AVL FIRE[1] software was used in this study.

The last example presented in this thesis was an investigation on LBM, using a BMW test vehicle, similar to the DrivAer model. EXA PowerFLOW software was used in this study. No comparison to a RANS model was implemented in this study. But the results were all pretty close to the experimental results.

The main target of this thesis was to determine the most efficient mesh for an unsteady simulation. After presenting these studies, the model specification was explained. Four real car models were built up for this thesis - a limousine, a city car, a SUV and a sports car. For each car six different mesh types were applied using the ANSA pre-processing software. One tetrahedral mesh (Tetra),[2] three different polyhedral meshes (PolyPoly, PolyPrism, PolyFluent)[3] and two hybrid-hexa meshes (HexaInterior, HexaPoly)[4] were applied for each car.

A special feature is included in the HexaInterior algorithm. When creating this mesh, the option "create pyramids" can be chosen. When this option is disabled, the CFD mesh does not work because of geometrical inconsistence between tetrahedrons and hexahedrons. This option has only use in some finite element solvers that do not support pyramids.

Basically, the HexaInterior is the mesh type with the most elements. The Tetra mesh has more elements than the HexaPoly (a range of one to 40 per cent more elements). This varies strongly with the car's size. The polyhedral meshes all have about 40 per cent less cells than the Tetra mesh.

Some hardware problems were detected when generating large polyhedral meshes. To get a polyhedral mesh with a higher resolution, a fine Tetra mesh needs to be generated. For the workstations, used at *Magna Steyr* (HP Z820), the conversion from a Tetra mesh to a polyhedral mesh is only possible for meshes with less than 100m volume elements.

The steady-state solver settings for the RANS simulations were taken from the *Magna Steyr* standard. 24 RANS simulations were set up. Three cars were simulated with 3 000 iterations. The limousine was simulated with 6 000 iterations to get a better impression of the convergence behaviour of the different mesh types. The settings for the unsteady SAS were taken from the experience *Magna Steyr* gained on this model over the last few years, using it for aeroacoustics.

The basis of comparison for all simulations was the Tetra mesh, which is the standard at

[1] AVL FIRE is one of the few commercial software packages which offers the PANS model.

[2] The Tetra mesh contains tetrahedral elements in the entire domain and features a prism layer on the car's surface, consisting of pentahedral elements.

[3] The PolyPoly mesh is a pure polyhedral mesh where also the prism layer is converted into polyhedrons. The PolyPrism mesh keeps the pentahedrons in the prism layer, but converts the rest of the domain to polyhedrons. The PolyFluent mesh is equivalent to the PolyPoly mesh, but converted in Fluent instead of ANSA.

[4] The HexaInterior mesh features a prism layer consisting of pentahedrons and a domain from hexahedrons. All transition zones between various sized hexahedrons or between the prism layer and the hexahedrons are realised with tetrahedrons and pyramids. The HexaPoly mesh is similar to the HexaInterior mesh, but the transition zones consist of polyhedrons.

Magna Steyr. On the first sight, the hybrid-hexa meshes showed a clear advantage over the various polyhedral meshes as well as over the Tetra mesh. Especially their convergence behaviour was impressive, considering 3 000 iterations in steady-state simulation. Above all, the behaviour of the residuals (continuity, x-velocity, y-velocity, z-velocity, TKE, omega) was amazing, compared to the tetrahedral and polyhedral meshes. In terms of accuracy, the hybrid-hexahedral meshes delivered mostly a considerably better result than the polyhedral meshes. Due to this, an attempt was made to create polyhedral meshes with a higher number of cells. But this failed due to hardware limitations as described above.

Subsequently, the simulations of the limousine car were extended to 6 000 iterations. It was detected, that the convergence behaviour of cx did not get better. The fluctuations of the instant cx value in the case of all meshes got even bigger. However, convergence of the residuals remained to be the best for the hybrid-hexahedral meshes. This behaviour may change from car to car. However, the averaging of the drag coefficient was executed on the last 300 iterations. By viewing the results, it got evident, that this was not enough. As recommended in paper [51], the averaging of the drag as well as the lift coefficient should be executed over several thousand iterations. For this fact, a very quick convergence is useful. The simulations with the tetrahedral mesh often converged not before 1 000 to 1 500 iterations. The hybrid-hexahedral as well as the polyhedral meshes converged much earlier.

Furthermore, it could be detected, that the wake of the polyhedral mesh simulations was larger than at the other mesh types. The wake of the Tetra and the hybrid-hexa meshes looked pretty much the same. The Tetra mesh was able to capture the most flow details. The PolyPoly and the PolyFluent showed a different picture in the flow separation at the rear window. The only polyhedral mesh, which was able to deliver similar flow details to the Tetra mesh, was the PolyPrism mesh. Also the hybrid-hexa meshes were not able to capture as much flow details as the Tetra mesh.

By summarising and averaging all the steady-state results, it got evident, that both, the HexaInterior and the HexaPoly mesh, deliver the best results compared to the Tetra mesh. It became obvious, that the HexaPoly mesh type delivers the best results overall. It offers the best convergence behaviour. Also the drag as well as lift coefficient are often the closest ones compared to the tetrahedral mesh. The relatively poor results of the initially favoured polyhedral meshes could possibly lead back to the relatively coarse resolution. The polyhedral meshes had a cell number which was on average lower than 40 per cent of the cell number of the Tetra mesh. As already mentioned above, an attempt was made to create polyhedral meshes (PolyPrism) with finer resolution. But that failed due to limitations in the hardware. 189 GiB RAM was not enough to convert a 136m cells Tetra mesh to a PolyPrism mesh. A fine Tetra mesh (219m cells) was generated to convert in Fluent using the CFD cluster. The result had not been post-processed when this thesis was handed in. However, the mesh influence seems to decrease in transient simulations.

Also the contribution to the drag and the lift coefficient due to the cooling packaged was investigated for all six tested mesh types. In terms of the drag coefficient, the cooling packaged contribution deviated between plus and minus two points, for the SUV and plus and minus one point for all the other cars. The deviation of the contribution of the lift coefficient is even smaller. So the influence of the mesh to the cooling package contribution to the drag and lift coefficient is negligible. Also a consistent visible difference between pentahedral cooling package volumes (Tetra, PolyPrism, HexaInterior, HexaPoly) and polyhedral cooling package volumes (PolyPoly, PolyFluent) could not be detected.

Based on the results of the steady-state simulations, two transient SAS were run. The HexaPoly mesh was chosen for the first transient SAS. The HexaPoly showed the best and quickest convergence behaviour in the steady-state case and also showed the highest overall accuracy and conformity with the results of the base Tetra mesh. As a reference, the Tetra mesh was also simulated with transient SAS. Interestingly, in the steady-state mode, the Tetra mesh was slightly faster than the HexaPoly mesh (about four per cent less time demand for 3 000 iterations at the sports car). In the transient case, in contrast, the HexaPoly mesh was almost twice as fast as the Tetra mesh. This means, that in a transient simulation, hexahedral elements are much more efficient, than tetrahedral elements. The transient HexaPoly SAS took only about six times longer than the steady-state HexaPoly RANS simulation (5:47:45 hours versus 36:31:06 hours). The transient Tetra simulation took almost nine times longer than the steady-state Tetra RANS simulation (5:35:38 hours versus 49:21:33 hours). The data output increased from about 9 GiB in the steady-state case (3 000 iterations) to over 30 GiB in the transient case (one second, 2 000 time steps).

If the drag coefficient is considered, all four simulation results, Tetra mesh and HexaPoly mesh, each steady-state as well as transient, lie within just 14 points. This is a range of less than five per cent of the drag coefficient. If the lift coefficient is considered, the range is 53 points, which is equal to a deviation of about 16 per cent. Hereby the drag coefficient of the transient solution is in both cases above the drag coefficient of the steady-state solution. The lift coefficient of both transient solutions, in contrast, is below the lift coefficient of the steady-state solutions. This coincides with the results of former investigations like in [12], [77] or [81]. Furthermore, this behaviour also coincides with the theory, explained in chapter 1.3 and chapter 3. This behaviour is achieved by the fact, that in a steady-state RANS simulation, most turbulent fluctuations (TKE) are damped out by the Reynolds-averaging (pressure as well as velocity vectors are averaged). Moreover, the absence of the unsteady temporal term prohibits the capturing of the temporal fluctuations. The used SAS model is an improved URANS formulation (refer to 3.4.5). An unsteady method resolves a fraction of the turbulent Kolmogorov spectrum. This allows the formation of a turbulent cascade (except classical URANS, as only the temporal fluctuations are covered by the unsteady temporal term, but not the spatial fluctuations). Where the RANS method under-predicts TKE, unsteady methods are able to capture a fraction of the turbulent spectrum and therefore TKE. More TKE means more energy to dissipate and more energy to dissipate is equal to an increase of drag.

A bit more complicated is the explanation of the decrease of the lift coefficient in transient simulations compared to steady-state simulations. However, this behaviour is again documented in former investigations, like [12], [77] or [81]. One reason for this can be found again in the under-prediction of the TKE in steady-state simulations. The bumpy underbody is predestined to be an object of producing strong turbulent fluctuations. On the other hand, turbulent fluctuations are energy containing structures. This (turbulent) kinetic energy reduces the pressure between the ground and the vehicle which results in an decrease of lift.

Furthermore, the deviation of lift compared to the experimental results is usually a lot larger than the deviation of drag compared to the experimental values. This can easily be explained with the turbulent fluctuations of the pressure coefficient. It is considered, that the fluctuations have the same magnitude for the drag in longitudinal (x) direction as in the vertical (z) direction. But the area, which is responsible for drag (y-z cross section) is a lot smaller than the area which is responsible for the lift (x-y cross section). Thus, a fluctuation of the same magnitude has more influence on the lift than on the drag. The x-y cross section is usually about four magnitudes larger than the y-z cross section.

Moreover, with the transient solver settings used in the course of this thesis, the simulations ran very stable. As specified from ANSYS, SAS can be applied with a CFL number larger than one. The transient simulations presented in this thesis ran with a CFL number of up to 1.5 in the focus region. Anyway, the CFL number is not a matter of stability, but a matter of accuracy. In SAS, an increase of the CFL number will result in a decrease of calculated flow details.

There are some meaningful reasons to use unsteady CFD simulation. The output of a transient simulation is much more comprehensive than the output of a steady-state simulation. The main profit of transient simulation is not predominantly the possibility to get a more accurate result of the drag or lift coefficient. Numbers are just one possible benefit. The main add-on is the much more detailed visualisation of the flow field.

With a transient CFD simulation, flow instability can be detected very easily. This is a big advantage, especially at rear slant angles between 25° and 30° where the flow is highly unstable. With the use of the Q-criterion for example, vortical structures (stable or not stable) can be visualised very efficiently. With this information, the car's design could possibly be adapted in a way, to affect vortex shedding in a positive direction in terms of drag reduction. So with the additional information of a transient CFD simulation, the aerodynamic design iterations could be reduced.

In addition to the WT experiment, this could be a great method to visualise the airflow around the car. Possibly even some (expensive) WT runs could be stinted. Furthermore, OEMs use transient CFD simulation to develop their cars. So, if an OEM places an order at a supplier, the OEM also wants the supplier to use transient CFD simulation.

These should be enough reasons for using unsteady CFD simulation. But the use of unsteady simulation should be well-founded. In many cases, a steady-state RANS simulation is sufficient.

Surprisingly, the transient simulation did not require as much time as expected. When referring to paper [12], a transient IDDES (1.3 seconds) required 17 times more computing time than a steady-state k-omega SST (2 500 iterations) simulation. In the case of this thesis, the transient SAS (1.0 seconds) required 6.2 times more computing time than the steady-state k-omega SST (3 000 iterations) simulation. If brought on same time and

iteration level, the SAS executed in this thesis requires 9.6 times more computing time than the steady-state k-omega SST model. However, within less than 40 hours, a full car (76m cells) transient aerodynamic simulation can easily be executed over a weekend. An important fact to remember is the time demand of creating a transient video. The generation of just one video (one second) requires double the time of one steady-state simulation (3 000 iterations). Just to get an imagination of how time consuming and expensive it is to create transient videos. One transient simulation was executed in the course of this thesis with creating six videos. The time demand was 23 times higher than a standard steady-state simulation and almost four times higher than a transient simulation without creating any videos. The transient video creation should be handled with great care not to expand the simulation time unnecessarily.
To summarise these thoughts, the use of transient simulation in external aerodynamics makes sense especially in the early stages of the car design. Furthermore, the results of unsteady CFD simulation can be used for other disciplines too. That is: [121]

- Acoustics: In aeroacoustics, the use of transient simulation is absolutely necessary. The turbulent spectrum offered by RANS simulation is not reliable.
- Combustion: The mixing of fuel and oxygen in the combustion chamber should be described in detail.
- Vortex cavitation: To determine cavitation, it is required to resolve vortices.
- Maneuverability of vehicles: To simulate the influence of turbulence or side wind, it is unavoidable to use transient simulation. This is valid for cars as well as for aeroplanes and ships.
- Safety and comfort: In terms of automotive engineering, the wake of a vehicle driving in front of another vehicle, could affect the following vehicles aerodynamics in a way, that safety issues could appear. Only with transient simulation it is possible to investigate such interferences between two vehicles.
- Fluid-structure interaction: Unsteady forces determine the frequency response of solid structures.

Collectively seen, the use of transient simulations is quite meaningful in some situations. But as it could be seen, the results in terms of numbers can be pretty good in steady-state simulations. However, a transient simulation delivers much more data to post-process than a steady-state simulation. Moreover, big differences appear by the use of different mesh types. Further investigations in terms of the use of alternative mesh types or alternative transient CFD methods will follow, based on the results of this thesis.

11 Perspective

Predicting the future is generally ill advised. However, as described above, in mid term LES seems to be the long term future in CFD. The Japanese car industry is still trying to establish LES in their workflows, [133] while many others already moved to hybrid RANS-LES methods. As Spalart reported in 1998, *"a full LES around a three-dimensional wing will not be tractable until the year 2045, even assuming that wall modelling has been achieved."* [128, 144] This has not changed until today. Noelting and Swen reported in 2015, that it will take at least two more decades, to make LES feasible for industrial use. [117] New hybrid methods, all in front DES with its derivative methods (DDES, IDDES, SDES, SBES), seem to deliver good results. Companies which have the computational power are in progress to establish DES methods for aerodynamics. [17] *Magna Steyr* already uses unsteady SAS since mid 2013 for aeroacoustics. This thesis is a first approach, to apply unsteady CFD for (external) aerodynamics.

A next step, based on the results of this thesis is to build up an additional ANSA model. This model will investigate the influence of the element size for a PolyPrism mesh. This is already under progress when this thesis is handed in. The element size of the shell mesh for the prism layer is reduced in two steps from between 2 mm and 4 mm to between 1 mm and 2 mm. This results in a prism layer with four times more volume elements and a much finer polyhedral mesh in the wall-near region. The intention of this is to find out, if the polyhedral meshes used in this thesis were to coarse to deliver accurate results. If the result of this PolyPrism mesh delivers a high accuracy, comparable with the hybrid-hexa meshes, an attempt with SAS may follow.

Another further investigation would be to compare SAS and SBES methods. As described in chapter 4.2.3, SBES is the latest method implemented in ANSYS Fluent. It is a derivative of the classic DES method. Furthermore, an investigation on the influence of different time steps for transient simulations is planned.

Sooner or later, *Magna Steyr* will acquire a new computer cluster. An increased computational power can be used for various purposes: hybrid RANS-LES instead of RANS, a reduction of a numerical error, the usage of a larger domain to get more appropriate boundary conditions or the inclusion of more sophisticated modules. [17] Figure 11.1 gives an early estimation of the possible future utilisation of the increase of computational power. This estimation was created in 2004. According to Basara, the utilisation has been moved more towards hybrid RANS-LES methods.

One of the leading persons in CFD, Sharath Girimaji, professor at the Texas A&M University, on present and future research focuses: *"Variable-resolution turbulence simulation schemes can be classified into two general categories: hybrid methods and bridging approaches. Hybrid computations entail Reynolds averaged Navier-Stokes (RANS) calculations in some flow regions and large eddy simulations (LES) in others. There exists*

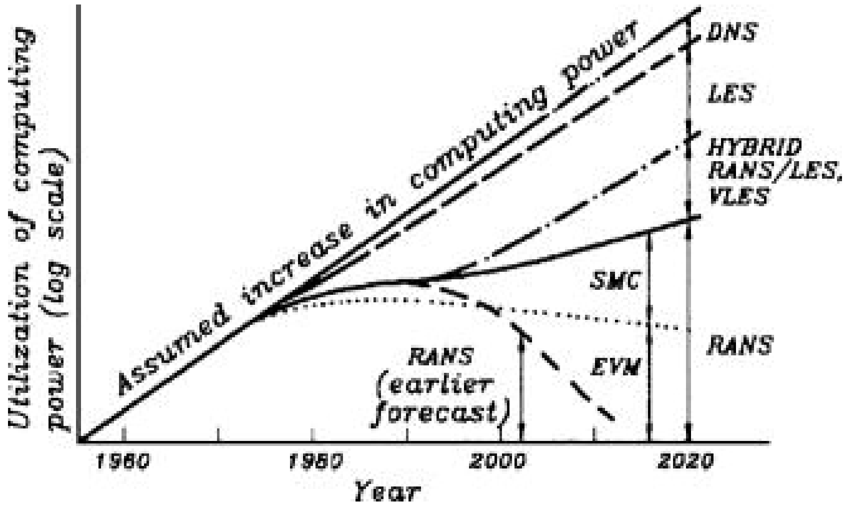


Figure 11.1: Possible future utilisation of the increase of computing power [62]

no clear consensus on the criterion for switching from RANS to LES. The bridging methods, on the other hand, seamlessly transition from one flow resolution to another. The present approaches, e.g. PITM or PANS are based in a 'solid' mathematical background and highlight its advantages over hybrid methods." [54]

For the next years, the focus for *Magna Steyr* will surely lay on SAS and maybe SBES. Unsteady simulation could do a good job in investigating design surfaces. Vortical structures could be made visible. Stability checks and vortex break up could be investigated. Natural turbulence or side wind could be implemented in the models. With these information, there would not only be an input into the cars longitudinal dynamics, but also in lateral dynamics. Also for vibration and noise[1] analysis unsteady simulation would deliver the needed information.

[1] *Magna Steyr* in fact already uses unsteady simulation for aeroacoustics.

Bibliography

[1] Acton, Q. A. National Office for Aerospace Studies and Research, Chatillon: Simulations of turbulent rotating flows using a subfilter scale stress model derived from the partially integrated transport modeling method. In *Issues in General Physics Research: 2013 Edition*, Q. A. Acton, Ed., 1 ed. ScholarlyEditions, 2013, ch. Fluid Rese, pp. 37–38.

[2] Ahmed, S. R., Ramm, G., and Faitin, G. Some Salient Features Of The Time-Averaged Ground Vehicle Wake. 1984.

[3] Anderson, J. D. J., Degroote, J., Degrez, G., Dick, E., Grundmann, R., and Vierendeels, J. *Computational Fluid Dynamics: An Introduction*, 3 ed. Springer-Verlag, Berlin-Heidelberg, 2009.

[4] ANSYS Inc. ANSYS Fluent User's Guide, 2013.

[5] ANSYS Inc. ANSYS Fluent V2F Turbulence Model Manual, 2013.

[6] ANSYS Inc. www.sharcnet.ca/Software/Ansys/17.0, 2016.

[7] ANSYS Inc. Fluent R17.2, 2017.

[8] Ashby, D. L., Dudley, M., and Iguchi, S. K. Development and Validation of an Advanced Low-Order Panel Method, 1988.

[9] Ashton, N. Computaional Challenges of the Aerodynamic Design Process in Formula 1. In *NASA Ames Applied Modelling & Simulation Seminar (AMS)* (Moffett Field, California, 2015), U. if Manchester, Ed., University of Manchester, pp. 1–41.

[10] Ashton, N., and Revell, A. Key factors in the use of DDES for the flow around a simplified car. *International Journal of Heat and Fluid Flow 54*, August (2015), 236–249.

[11] Ashton, N., Revell, A., and Poletto, R. Grey-Area Mitigation for the Ahmed Car Body Using Embedded DDES. In *Progress in Hybrid RANS-LES Modelling*, S. Girimaji, W. Haase, S.-H. Peng, and D. Schwamborn, Eds. Springer International Publishing, 2015, ch. Grey-Area, pp. 119–129.

[12] Ashton, N., West, A., and Revell, A. Assessment of RANS and DES methods for realistic automotive models. *Computers and Fluids 128*, January (2016), 1–15.

[13] Asnaghi, A. CFD with OpenSource Software: interPhasChangeFoam Tutorial and PANS Turbulence Model, 2013.

[14] Basara, B., Krajnović, S., and Girimaji, S. PANS vs. LES for Computations of the Flow Around a 3D Bluff Body. In *7th International ERCOFTAC Symposium on "Engineering Turbulence Modelling and Measurements", June 4-6, 2008, Limassol, Cyprus.* (Limassol, Cyprus, 2008).

[15] Basara, B., Krajnović, S., Pavlovic, Z., and Ringqvist, P. Performance analysis of Partially-Averaged Navier-Stokes method for complex turbulent flows. 6th AIAA Theoretical Fluid Mechanics Conference, 2011, pp. 1–10.

[16] Basara, B., Przulj, V., and Tibaut, P. On the Calculation of External Aerodynamics: Industrial Benchmarks. In *SAE 2001 World Congress* (Detroit, Michigan, 2001), B. Basara, V. Przulj, and P. Tibaut, Eds., AVL List, SAE International.

[17] Basara, B., and Steiner, H. Vorlesung an der TU Graz - Numerische Methoden in der Strömungslehre und Wärmeübertragung, 2016.

[18] Behnia, M., Parneix, S., Shabany, Y., and Durbin, P. A. Numerical study of turbulent heat transfer in confined and unconfined impinging jets. *International Journal of Heat and Fluid Flow 20*, 1 (1999), 1–9.

[19] Blumrich, R. Berechnungsmethoden für die Aeroakustik von Fahrzeugen. In *ATZ/MTZ-KONFERENZ AKUSTIK - Akustik zukünftiger Fahrzeuge, Stuttgart, 17-18. Mai 2006* (Stuttgart, Germany, 2006), R. Blumrich, Ed., Konferenz Akustik and Vieweg Technology Forum.

[20] Boussinesq, J. M. Théorie de l'écoulement tourbillonnant et tumultueux des liquides dans les lits rectilignes a grande section, 1897.

[21] Bradshaw, P. The Best Turbulence Models for Engineers. In *Modeling Complex Turbulent Flows*, M. D. Salas, J. N. Hefner, and L. Sakell, Eds., 7 ed. Springer Netherlands, 1999, pp. 9–28.

[22] Brenn, G. Vorlesung an der TU Graz - Strömungslehre und Wärmeübertragung I, 2012.

[23] Brezzi, F., Lipnikov, K., and Shashkov, M. Convergence of the mimetic Finite Difference Method for Diffusion Problems on Polyhedral Meshes. *SIAM J. Numer. ANAL. 43*, 5 (2005), 1872–1896.

[24] Brezzi, F., Lipnikov, K., and Shashkov, M. New Discretization Methodology for Diffusion Problems on Polyhedral Meshes, 2005.

[25] Brezzi, F., Lipnikov, K., and Simoncini, V. A family of mimetic finite difference methods on polygonal and polyhedral meshes. *Math. Models Methods Appl. Sci. 15* (2005), 1533–1551.

[26] Bristow, D. R. Development of Panel for Subsonic Analysis a Development for Subsonic of Panel Methods Analysis and Design, 1980.

[27] Car Body Design. http://www.carbodydesign.com/archive/2009/10/peugeot-rcz/, 2009.

[28] Caridi, D., Cokljat, D., Schuetze, J., and Lechner, R. Embedded Large Eddy Simulation of Flow around the Ahmed Body. *SAE International Journal of Commercial Vehicles - V121-2 121*, 2 (2012), 1–7.

[29] CFD Online. www.cfd-online.com, 2017.

[30] Chaouat, B. The partially integrated transport modeling method for continuous hybrid non-zonal RANS/LES simulations of turbulent flows. *Notes on Numerical Fluid Mechanics and Multidisciplinary Design 111* (2010).

[31] Chaouat, B. Simulations of turbulent rotating flows using a subfilter scale stress model derived from the partially integrated transport modeling method. *Physics of Fluids 24* (2012).

[32] Chaouat, B., and Schiestel, R. A new partially integrated transport model for subgrid-scale stresses and dissipation rate for turbulent developing flows. *Physics of Fluids 17*, 6 (2005).

[33] Chaouat, B., and Schiestel, R. Analytical insights into the partially integrated transport modeling method for hybrid Reynolds averaged Navier-Stokes equations-large eddy simulations of turbulent flows. *Physics of Fluids 24* (2012).

[34] Chaouat, B., and Schiestel, R. Partially integrated transport modeling method for turbulence simulation with variable filters. *Physics of Fluids 25* (2013).

[35] Chen, H. Realization of Fluid Boundary Conditions via Discrete Boltzmann Dynamics. *International Journal of Modern Physics 09*, 08 (1998), 1281–1292.

[36] Chen, H. Volumetric formulation of the lattice Boltzmann method for fluid dynamics: Basic concept. *Phys. Rev. E 58*, 3 (1998), 3955–3963.

[37] Chen, H., Kandasamy, S., Orszag, S., Shock, R., Succi, S., and Yakhot, V. Extended Boltzmann Kinetic Equation for Turbulent Flows. *Science 301*, August 2003 (2003), 633–636.

[38] Chong, M., Perry, A., and Cantwell, B. A General Classification of Three-Dimensional Flow Fields. *Physics of Fluids 2*, 5 (1990), 765–777.

[39] Cokljat, D., Caridi, D., Link, G., Lechner, R., and Menter, F. Embedded LES Methodology for General-Purpose CFD Solvers. In *Proceedings of 6th International Symposium on Turbulence and Shear Flow Phenomena* (2009), ANSYS Inc., pp. 1191–1196.

[40] Daimler AG. www.mercedes-benz.com/de/mercedes-benz/design/mercedes-benz-design/konzeptfahrzeuge/concept-iaa-intelligent-aerodynamic-automobile/, 2015.

[41] Davidson, L. Fluid mechanics, turbulent flow and turbulence modeling, 2016.

[42] Deardorff, J. W. A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers. *Journal of Fluid Mechanics 41*, 2 (1970), 453–480.

[43] Deck, S. Zonal detached-eddy simulation of the flow around a high-lift configuration with deployed slat and flap. *AIAA Journal 43*, 11 (2005), 2372–2384.

[44] Durbin, P. A. Application of a near-wall turbulence model to boundary layers and heat transfer. *International Journal of Heat and Fluid Flow 14*, 4 (1993), 316–323.

[45] Epton, M. A., and Magnus, A. E. PAN AIR: A computer program for predicting subsonic or supersonic linear potential flows about arbitrary configurations using a higher order panel method. Volume 1: Theory document (version 3.0), 1981.

[46] Fares, E. Unsteady flow simulation of the Ahmed reference body using a lattice Boltzmann approach. *Computers & Fluids 35*, 8 (2006), 940–950.

[47] Feynman, R. P. Spoken Quotation, 1964.

[48] Flaig, A. Airbus A380 – Solutions to the Aerodynamic Challenges of Designing the World's Largest Passenger Aircraft. In *Lecture to DGLR/VDI & RAeS* (Hamburg, Germany, 2008), Royal Aeronautical Society Hamburg Branch, pp. 1–80.

[49] Formula 1 Dictionary. www.formula1-dictionary.net, 2017.

[50] Forsythe, J. R., Spalart, P. R., Squires, K. D., and Wurtzler, K. E. Detached-Eddy Simulation of the F-15E at High Alpha. *Journal of Aircraft 41*, 2 (2004), 193–200.

[51] Fotiadis, G., Skaperdas, V., Iordanidis, A., and Peddiraju, P. The influence of mesh characteristics on external airflow CFD simulations of the DrivAer model. In *AMS Seminar Series* (2015), BETA CAE Systems USA, Inc., pp. 1–49.

[52] Gerasimov, A. Vortex Visualisation in ANSYS Fluent Part 1 : Scale-Resolving Simulations. Tech. rep., ANSYS Sweden AB, 2016.

[53] Gerasimov, A. Vortex Visualisation in ANSYS Fluent Part 2 : RANS. Tech. rep., ANSYS Sweden AB, 2016.

[54] Girimaji, S., and Rubinstein, R. Bridging Approach to variable-resolution turbulence simulations – Unification of PANS and PITM closures. *Bulletin of the American Physical Society 55*, 16 (2010).

[55] Girimaji, S. S. Partially-Averaged Navier-Stokes Model for Turbulence: A Reynolds-Averaged Navier-Stokes to Direct Numerical Simulation Bridging Method. *Journal of Applied Mechanics 73*, 3 (2006), 413–421.

[56] Girimaji, S. S., Srinivasan, R., and Jeong, E. PANS Turbulence Model for Seamless Transition Between RANS and LES: Fixed-Point Analysis and Preliminary Results. In *ASME/JSME 2003 4th Joint Fluids Summer Engineering Conference* (Honolulu, Hawaii, USA, 2003), S. S. Girimaji, R. Srinivasan, and E. Jeong, Eds., Texas A&M University, College Station, TX, pp. 1901–1909.

[57] Greschner, B., Thiele, F., Jacob, Marc, C., and Casalino, D. Prediction of sound generated by a rod–airfoil configuration using EASM DES and the generalised Lighthill/FW-H analogy. *Turbulent Flow and Noise Generation 37*, 4 (2008), 402–413.

[58] Guilmineau, E. Numerical simulation with a DES approach for automotive flows. *Journal of Fluids and Structures 27*, September 2011 (2011), 807–816.

[59] Guilmineau, E. Numerical Simulations of Flow around a Realistic Generic Car Model. *SAE International Journal of Passenger Cars - Mechanical Systems 123*, 6 (2014), 1–8.

[60] Han, X., Krajnović, S., and Basara, B. Study of active flow control for a simplified vehicle model using the PANS method. *International Journal of Heat and Fluid Flow 42* (2013), 139–150.

[61] Han, X., Krajnović, S., and Bruneau, C.-H. Comparison of URANS, PANS, LES and DNS of flows around simplified ground vehicles with passive flow manipulation. In *ERCOFTAC Series* (Dresden, Germany, 2013), vol. 9.

[62] Hanjalic, K. Will RANS Survive LES? A View of Perspectives. *Journal of Fluids Engineering 127*, 5 (2005), 831–839.

[63] Hardy, J., Pomeau, Y., and De Pazzis, O. Time evolution of a two dimensional model system. I. Invariant states and time correlation functions. *Journal of Mathematic Physics 14* (1973), 1746–1759.

[64] Harlow, F. H. Fluid dynamics in Group T-3 Los Alamos National Laboratory. *Journal of Computational Physics 195*, 2 (2004), 414–433.

[65] Heft, A. I., Indinger, T., and Adams, N. Experimental and Numerical Investigation of the DrivAer Model. In *ASME 2012 Fluids Engineering Division Summer Meeting* (Rio Grande, Puerto Rico, USA, 2012), pp. 41–51.

[66] Heft, A. I., Indinger, T., and Adams, N. Introduction of a New Realistic Generic Car Model for Aerodynamic Investigations, 2012.

[67] Heft, A. I., Indinger, T., and Adams, N. A. Investigation of Unsteady Flow Structures in the Wake of a Realistic Generic Car Model. In *29th AIAA Applied Aerodynamics Conference* (Honolulu, Hawaii, USA, 2011), American Institute of Aeronautics and Astronautics.

[68] Hess, J. L., and Friedman, D. M. Analysis of complex inlet configurations using a higher-order panel method, 1983.

[69] Hesthaven, J. S., and Warburton, T. *Nodal Discontinuous Galerkin Methods - Algorithms, Analysis, and Applications*, 1 ed. Springer Verlag, 2008.

[70] Hickel, S. Vorlesung an der Technischen Universität München - Angewandte Strömungssimulation 5. Vorlesung.

[71] Hinterberger, C., García-Villalba, M., and Rodi, W. Large eddy simulation of flow around the Ahmed body. In *The Aerodynamics of Heavy Vehicles: Trucks, Buses, and Trains*, R. McCallen, F. Browand, and J. Ross, Eds. Springer International Publishing, 2004, ch. Large eddy, pp. 77–87.

[72] Hucho, W.-H. *Aerodynamik der stumpfen Körper - Physikalische Grundlagen und Anwendungen in der Praxis*, 2 ed. Springer Verlag, 2012.

[73] Hucho, W.-H. Numerische Verfahren. In *Aerodynamik der stumpfen Körper - Physikalische Grundlagen und Anwendungen in der Praxis*, W.-H. Hucho, Ed., 2 ed. Springer Verlag, 2012, ch. Numerische, pp. 498–572.

[74] Hunt, J., Wray, A., and Moin, P. Eddies, Streams, and Convergence Zones in Turbulent Flows. In *Proceedings of the Summer Programm 1988* (1988), Center for Turbulence Research, pp. 193–208.

[75] Huynh, T. H. A Flux Reconstruction Approach to High-Order Schemes Including Discontinuous Galerkin Methods. In *18th AIAA Computational Fluid Dynamics Conference 25 - 28 June 2007* (2007), p. 42.

[76] Iordanidis, A. Email conversation with Beta CAE Systems, 2017.

[77] Jakirlic, S., Kutej, L., Basara, B., and Tropea, C. Numerische Fahrzeugaerodynamik am Beispiel der DrivAer-Modellkonfigurationen. *Automobiltechnische Zeitschrift 118*, 05 (2016), 72–79.

[78] Jameson, A. Computational Fluid Dynamics Past, Present and Future. 2015.

[79] Jameson, A., Schmidt, W., and Turkel, E. Numerical solutions of the Euler equations by finite volume methods using Runge-Kutta time-stepping schemes. *AIAA paper* (1981), 1–19.

[80] Jeong, J., and Hussain, F. On the identification of a vortex. *Journal of Fluid Mechanics 285* (1995), 69–94.

[81] Jungmann, J., Schütz, T., Jakirlic, S., and Tropea, C. Flow past a DrivAer body in a scale model wind tunnel: computational study by reference to a complementary experiment. 2016.

[82] Kandasamy, S., Duncan, B., Gau, H., Maroy, F., Belanger, A., Gruen, N., and Schäufele, S. Aerodynamic Performance Assessment of BMW Validation Models using Computational Fluid Dynamics. In *SAE 2012 World Congress & Exhibition* (2012), SAE International, pp. 1–21.

[83] Kapadia, S., Roy, S., and Wurtzler, K. Detached Eddy Simulation Over a Reference Ahmed Car Model. In *41st Aerospace Sciences Meeting and Exhibit* (Reno, Nevada, USA, 2003), K. Wurtzler, Ed., no. January, American Institute of Aeronautics and Astronautics, pp. 1–10.

[84] Kolář, V. Vortex identification: New requirements and limitations. *International Journal of Heat and Fluid Flow 28*, 4 (2007), 638–652.

[85] Kolář, V. Brief notes on vortex identification. *Proc of the 8th WSEAS Intl Conf on Fluid Mechanics* (2011), 23–28.

[86] Kotapati, R., Keating, A., Kandasamy, S., Duncan, B., Shock, R., and Chen, H. The Lattice-Boltzmann-VLES Method for Automotive Fluid Dynamics Simulation, a Review. In *SIAT 2009* (2009), SAE International, pp. 1–13.

[87] Krajnovic, S., and Davidson, L. Large-Eddy Simulation of the Flow Around Simplified Car Model. In *2004 SAE World Congress* (Detroit, USA, 2004), SAE International, pp. 1–10.

[88] Launder, B. E., Jones, W. P., and Launder, B. E. The prediction of laminarization with a two-equation model of turbulence. *International Journal of Heat and Mass Transfer 15*, 2 (1972), 301–314.

[89] Launder, B. E., Reece, G. J., and Rodi, W. Progress in the development of a Reynolds-stress turbulence closure. *Journal of Fluid Mechanics 68*, 03 (1975), 537–566.

[90] Leschziner, M. *Statistical Turbulence Modelling for Fluid Dynamics - Demystified: An Introductory Text for Graduate Engineering Students*, 1 ed. Imperial College Press, London, UK, 2015.

[91] Lesieur, M. *Turbulence in Fluids*, 4 ed. Springer International Publishing, 2008.

[92] Li, Y., Richard, S., Zhang, R., and Chen, H. Numerical study of flow past an impulsively started cylinder by the lattice-Boltzmann method. *Journal of Fluid Mechanics 519* (2004), 273–300.

[93] Location longue durée - Véhicules particuliers pour entreprises. www.locationlonguedureevoiture.com/devis-location-longue-duree-LLD-peugeot-rcz-305, 2010.

[94] Manceau, R. Report on the 10th joint ERCOFTAC (SIG-15)/ IAHR. In *QNET-CFD Workshop on Refined Turbulence Modelling Poitiers , October 10-11 , 2002* (2002), Université de Poitiers, Ed., Laboratoire d'études aérodynamiques, pp. 3–5.

[95] Manoha, E., and Caruelle, B. Summary of the LAGOON Solutions from the Benchmark problems for Airframe Noise Computations-III Workshop. In *21st AIAA/CEAS Aeroacoustics Conference* (Dallas, Texas, USA, 2015), E. Manoha and B. Caruelle, Eds., 21st AIAA/CEAS Aeroacoustics Conference 2015.

[96] Marié, S., Ricot, D., and Sagaut, P. Comparison between lattice Boltzmann method and Navier-Stokes high order schemes for computational aeroacoustics. *Journal of Computational Physics 228*, 4 (2009), 1056–1070.

[97] Mathey, F., and Cokljat, D. Zonal Multi-Domain Rans/Les Simulation of Airflow Over the Ahmed Body. In *Conference: 6th International Symposium on Engineering Turbulence Modelling and Measurements* (2005).

[98] Mathey, F., Cokljat, D., Bertoglio, J.-P., and Sergent, E. Specification of LES Inlet Boundary Conditions Using Vortex Method. *Turbulence, heat and mass transfer 4* (2003), 475–482.

[99] McCallen, R., Browand, F., and Ross, J. *The Aerodynamics of Heavy Vehicles: Trucks, Buses, and Trains*, 1 ed. Springer Verlag, 2004.

[100] Medic, G., Kalitzin, G., and You, M. Annual Research Briefs. Tech. rep., Stanford University, Stanford, 2006.

[101] Mendonça, F., Allan, R., de Charentenay, J., and Lewis, M. Towards understanding LES and DES for industrial aeroacoustic prediction. In *Int. Workshop LES Acoustic, Göttingen 2002* (Göttingen, Germany, 2002), Int. Workshop LES Acoustic, Göttingen.

[102] Menter, F. R. Improved two-equation k-omega turbulence models for aerodynamic flows. Tech. rep., 1992.

[103] Menter, F. R. *Best Practice: Scale-Resolving Simulations in ANSYS CFD.* No. 2.00. ANSYS Germany GmbH, 2015.

[104] Menter, F. R. Stress-Blended Eddy Simulation (SBES) - A new Paradigm in hybrid RANS-LES Modeling. In *Sixth HRLM Symposium, 26-28 September 2016* (Strasbourg, France, 2016), F. R. Menter, Ed., ANSYS Germany GmbH, pp. 1–5.

[105] Menter, F. R., and Egorov, Y. A Scale Adaptive Simulation Model using Two-Equation Models. In *43rd AIAA Aerospace Sciences Meeting and Exhibit 2005* (Reno, Nevada, USA, 2005), American Institute of Aeronautics and Astronautics.

[106] Menter, F. R., and Egorov, Y. Scale-Adaptive Simulation Method for Unsteady Flow Predictions. Part 1: Theory and Model Description. *Journal Flow Turbulence and Combustion 85*, 1 (2010), 113–138.

[107] Menter, F. R., Gritskevich, M. A., Egorov, Y., and Schütze, J. Scale-Resolving Simulations in Industrial CFD -Models and Best Practice Motivation for Scale-Resolving Simulation (SRS). 2013.

[108] Millennium Problems. www.claymath.org/millennium-problems, 2000.

[109] Milne-Thomson, L. M. *Theoretical Aerodynamics*, 1 ed. Dover Publications, 1973.

[110] Minelli, G. CFD with OpenSource Software: PANS turbulence model implementation, 2015.

[111] Minguez, M., Antipolis, N. S., and Serre, E. High-order large-eddy simulation of flow over the “ Ahmed body ” car model. *Physics of Fluids 20*, September (2008).

[112] Moore, G. E. Cramming More Components Onto Integrated Circuits, Electronics, April 19, 1965. *Electronics 38*, 8 (1965), 82–85.

[113] Morel, T. The Effect of Base Slant on the Flow Pattern and Drag of Three-dimensional Bodies with Blunt Ends. In *Aerodynamic Drag Mechanisms of Bluff Bodies and Road Vehicles*, T. Morel, Ed., 1 ed. Springer Verlag, 1978, ch. 7, pp. 191–226.

[114] Motorsport Total. www.motorsport-total.com/f1/news/2010/02/Newey_haelt_nicht_viel_vom_Virgin-Weg_10021015.html, 2010.

[115] Murman, E. M., and Cole, J. D. Calculation of plane steady transonic flows. *AIAA Journal 9*, 1 (1970), 114–121.

[116] NASA Ames Research Center. www.software.nasa.gov/software/ARC-14407-1, 2016.

[117] Noelting Swen, F. E. The Lattice-Boltzmann Method: An Alternative to LES for Complex Aerodynamic and Aeroacoustic Simulations in the Aerospace Industry. *SAE International* (2015), 1–12.

[118] Nye, D. *Das große Buch der Formel- I- Rennwagen. Die Dreiliterformel ab 1966.* Autocourse, 1992.

[119] Ooi, A., Iaccarino, G., Durbin, P. A., and Behnia, M. Reynolds averaged simulation of flow and heat transfer in ribbed ducts. *International Journal of Heat and Fluid Flow 23*, 6 (2002), 750–757.

[120] OpenCFD Ltd. OpenFOAM - The Open Source CFD Toolbox: User Guide. Tech. rep., OpenCFD Ltd., 2017.

[121] Oswald, M. Scale Resolving Turbulence Modeling. Tech. rep., ANSYS Germany GmbH, 2010.

[122] Oswald, M. Aerodynamics & Aeroacoustics with ANSYS CFD. Tech. rep., ANSYS Germany GmbH, 2015.

[123] Peric, M., and Fergouson, S. The advantage of polyhedral meshes, 2004.

[124] Peters, B. C., Uddin, M., Bain, J., Curley, A., and Henry, M. Simulating DrivAer with Structured Finite Difference Overset Grids. In *SAE 2015 World Congress & Exhibition* (2015), SAE International, pp. 1–9.

[125] Richardson, L. F. *Weather Prediction by Numerical Process*, 1 ed. Cambridge Mathematical Library, 1922.

[126] Rotta, J. C. *Turbulente Strömungen - Eine Einführung in die Theorie und ihre Anwendung*, 15 ed. Vieweg+Teubner Verlag, Göttingen, Germany, 1972.

[127] Rudnik, R. Stall Behaviour of the EUROLIFT High Lift Configurations, 2008.

[128] Sagaut, P., and Deck, S. B. Large eddy simulation for aerodynamics: status and perspectives. *Philosophical Transactions of the Royal Society* (2009), 2849 – 2860.

[129] Sanz, W. Vorlesung an der TU GRaz - Computational Fluid Dynamics, 2014.

[130] Schäufele, S. *Validierung der neuen Windkanäle im Aerodynamischen Versuchszentrum der BMW Group und Analyse der Übertragbarkeit der Ergebnisse.* Dissertation, KIT Karlsruhe, 2010.

[131] Schiestel, R., and Dejoan, A. Towards a new partially integrated transport model for coarse grid and unsteady turbulent flow simulations. *Theoretical and Computational Fluid Dynamics 18*, 6 (2005), 443–468.

[132] Schlichting, H., and Gersten, K. *Grenzschicht-Theorie*, 10 ed. Springer Verlag, 2006.

[133] Schütz, T. *Hucho – Aerodynamik des Automobils*, 6 ed. Springer Viehweg, München, Deutschland, 2013.

[134] Serre, E., Minguez, M., Pasquetti, R., and Rodi, W. On Simulating the Turbulent Flow around the Ahmed Body : A French-German Collaborative Evaluation of LES and DES. *Computers & Fluids*, June (2011).

[135] Shur, M., Spalart, P. R., Squires, K. D., Strelets, M., and Travin, A. Three-Dimensionality in Reynolds-Averaged Navier-Stokes Solutions Around Two-Dimensional Geometries. *AIAA Journal 43*, 6 (2005), 1230–1242.

[136] Shur, M. L., Spalart, P. R., Strelets, M. K., Travin, A. K., Streletsa, M. K., and Travin, A. K. A hybrid RANS-LES approach with delayed-DES and wall-modelled LES capabilities. *International Journal of Heat and Fluid Flow 29*, 6 (2008), 1638–1649.

[137] Siemens Product Lifecycle Management Software Inc. CD-adapco STAR-CCM+, 2017.

[138] Skaperdas, V., and Iordanidis, A. The influence of mesh characteristics on CFD simulations for automotive applications. In *ANSYS 2013 Automotive Simulation World Congress* (Frankfurt am Main. Germany, 2013), BETA CAE Systems S.A., Ed., Beta CAE Systems, pp. 1–37.

[139] Slotnick, J., Khodadoust, A., Alonso, J., D., D., Gropp, W., E., L., and Mavriplis, D. CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences. Tech. rep., NASA Langley Research Center, Hampton, Virginia, 2013.

[140] Smagorinsky, J. General Circulation Experiments With the Primitive Equations. *Monthly Weather Review 91*, 3 (1963), 99–164.

[141] Sonar, T. Turbulenzen um die Fluidmechanik. *Spektrum der Wissenschaft*, April (2009), 78–87.

[142] Spalart, P., Jou, W.-H., Strelets, M., and Allamaras S.R. Advances in LES-DNS. C. L. Liu and Zhining, Eds., Greyden Press, Columbus, pp. 136–147.

[143] Spalart, P. R. Strategies for Turbulence Modelling and Simulations. *International Journal of Heat and Fluid Flow 21*, 3 (2000), 252–263.

[144] Spalart, P. R. Trends in Turbulence Treatments. In *AIAA Paper A-33784* (Seattle, Washington, 2000), Boeing Commercial Airplanes, American Institute of Aeronautics and Astronautics (AIAA), pp. 1–13.

[145] Spalart, P. R. Young-Person's Guide to Detached-Eddy Simulation Grids. In *NASA/CR-2001-211032* (2001), NASA, Ed., NASA Center for AeroSpace Information (CASI), pp. 2–23.

[146] Spalart, P. R. Detached-Eddy Simulation. *Annu. Rev. Fluid Mech 41* (2009), 181–202.

[147] Spalart, P. R., and Allamaras, S. A One-Equation Turbulence Model for Aerodynamic Flows. In *AIAA Paper 9226285* (Reno, Nevada, 1992), 30th Aerospace Sciences Meeting & Exhibit, p. 23.

[148] Spiegel, M., Redel, T., Zhang, Y. J., Struffert, T., Hornegger, J., Grossman, R. G., Doerfler, A., and Karmonik, C. Tetrahedral and Polyhedral Mesh Evaluation for Cerebral Hemodynamic Simulation - a Comparison.

M. Spiegel, T. Redel, Y. J. Zhang, T. Struffert, J. Hornegger, R. G. Grossman, A. Doerfler, and C. Karmonik, Eds., 31st Annual International Conference of the IEEE EMBS, pp. 1–4.

[149] Steiner, H. Vorlesung an der TU Graz - Höhere Strömungslehre und Wärmeübertragung, 2015.

[150] Stephens, D. W., and Mohanarangam, K. Turbulence Model Analysis of Flow Inside a Hydrocyclone. *7th International Conference on CFD in the Minerals and Process Industries*, December (2009), 1–8.

[151] Succi, S. The Lattice Boltzmann Equation for Fluid Dynamics and Beyond. *European Journal of Mechanics - B/Fluids 22*, 1 (2001), 281.

[152] Tsuboi, K., Himeno, R., Shirayama, S., and Kuwahara, K. Computational study of the effect of base slant. In *4th Joint Fluid Mechanics, Plasma Dynamics and Lasers Conference* (Atlanta, Georgia, USA, 1986), K. Tsuboi, R. Himeno, S. Shirayama, and K. Kuwahara, Eds., American Institute of Aeronautics and Astronautics (AIAA).

[153] Versteeg, H. K., and Malalasekera, W. *An introduction to Computational Fluid Dynamics - The Finite Volume Method*, 1 ed. Longman Scientific & Technical, Essex, UK, 1995.

[154] Wagner, A. J. A Practical Introduction to the Lattice Boltzmann Method, 2008.

[155] Wang, Z. J., Liu, Y., May, G., and Jameson, A. Spectral difference method for unstructured grids II: Extension to the Euler equations. *Journal of Scientific Computing 32*, 1 (2007), 45–71.

[156] Wäschle, A. Numerische und experimentelle Untersuchung des Einflusses von drehenden Rädern auf die Fahrzeugaerodynamik, 2006.

[157] Wikipedia. www.wikipedia.org, 2016.

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

[159] Youngren, H. H., Bouchard, E. E., Coopersmith, R. M., and Miranda, L. R. Comparison of panel method formulations and its influence on the development of QUADPAN, an advanced low-order method, 1983.

List of Figures

List of Tables

List of Equations

Appendix

Definitions

Turbulence: A laminar flow flows in parallel layers with no mixing between these layers. A turbulent flow is a highly unstable flow with highly disordered velocity vectors. In turbulent flow, diffusion increases. [17]
"A turbulent flow is a flow, which is disordered in time and space." [91]

Instability: Flow instability is closely related to the term turbulence. Turbulence cannot appear without flow instability, but a flow can be unstable without getting turbulent. A flow field is unstable if there are deviations to the mean velocity. If these deviations get large, the flow gets turbulent. Probably the best known unstable flow phenomena are the Kelvin-Helmholtz instability or the Kármán vortex street. These are non turbulent flows with periodical vortex structures.

High and low Re-layers: A high Re-layers consists of much thicker and less prism layers than low Re layers. In sum, both should cover about the same wall distance. In [138], the first layer of the high Re-layer is about 2mm where the first layer of the low Re model is 0.05mm thick. Both with a growth rate of 1.2 cover a wall distance of about 30mm using seven and 25 single layers respectively.

Segregated solver: A segregated solver solves all equations separately. For each variable (p, u_i, k, ω) a flow field considering all cells is solved. The segregated solver is very memory efficient, but its convergence is usually very slow.

Coupled solver: A coupled solver solves the momentum equation and the equation of continuity in a coupled manner. The variables of the turbulence model (k, ω) are solved in a segregated manner like in the classical segregated algorithm.

SIMPLE algorithm: The SIMPLE (Semi-Implicit Method for Pressure Linked Equations) algorithm is applied, when the pressure field is unknown. The idea behind this is to approximate the pressure field, then solve the velocity field and close the system with the equation of continuity. From these results a pressure correction, followed by a velocity correction. This loop is repeated till the pressure and velocity approaches at a given level of accuracy. A further development is the SIMPLER (SIMPLE-Revised) algorithm. [157]

Points (pts.): Aerodynamic lift or drag is specified by dimensionless coefficients cx (c_D) and cz (c_L). Drag coefficient usually range from about 0.220 to 0.380 for commercial passenger cars. In order to have an entity to describe the change in drag (or lift), the point-system was introduced. A change in drag or lift of 0.001 equals to one point.

Q-criterion: The Q-criterion is a method to identify and display coherent structures within a flow field. It was introduced by Hunt et al. in 1988. The identification of the connected fluid regions uses a positive second invariant of ∇u [74, 84]

$$Q = C_Q \left(\Omega_{ij}^2 - S_{ij}^2 \right) > 0$$

with

$$\Omega_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial xj} - \frac{\partial u_j}{\partial x_i} \right) \quad \text{and} \quad S_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial xj} + \frac{\partial u_j}{\partial x_i} \right).$$

This means, the vorticity magnitude prevails over the shear strain rate magnitude. Ω denotes the vorticity and S denotes the strain rate. Different definitions of C_Q can be found in the literature. ANSYS Fluent uses $C_Q = 0.5$ and ANSYS CFD-Post uses $C_Q = 0.25$. This constant is in fact not important, as only the visual impression of the vorticity is interesting and not its size. [103] Furthermore, it is required, that the pressure in the vortex core needs to be lower than the ambient pressure. This means, a vortex is a connected region, where Q>0 and the pressure is lower than the ambient value. The Q-criterion is only suitable for incompressible flows. [84]
For best visualisation, ANSYS recommends an iso-value of five per cent to 25 per cent of the maximum value of constant turbulence. For special applications, ANSYS Fluent offers the normalised Q-criterion. For general cases, ANSYS recommends to refrain from this option. Its main use is for scripting and quicker post-processing. [52]
A series of further methods to identify vortical structures exists. The most popular of these are: [85]

- Δ-criterion [38]
- λ_2-criterion [80]

With an additional term, the λ_2-criterion is also suitable for compressible flows. All these criteria are mainly applicable for transient simulations. For steady-state simulations, other tools like velocity component magnitude, pathlines, vorticity magnitude, helicity but also the normalised Q-criterion for some cases. [53]

Formulas

Relation between the Boussinesq approximation (buoyancy) and the Reynolds stress tensor: Following formulas and their relation are quoted from [20], [29] and [157]. In terms of turbulence modelling, the Boussinesq approximation describes the basic theory of the EVMs.[1] The Reynolds stress tensor used in the Reynolds equations needs to be resolved by a turbulence model. The Reynolds stress tensor is defined as

$$\tau'_{ij} = \rho \cdot \overline{u'_i u'_j}.$$

The Boussinesq approximation is defined as

$$-\overline{u'_i u'_j} = \nu_t \left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i} - \frac{2}{3} \frac{\partial \overline{u}_k}{\partial x_k} \delta_{ij} \right) - \frac{2}{3} k \delta_{ij}$$

where

$$\delta_{ij} = \begin{cases} 1 & \text{if} \quad i = j \\ 0 & \text{if} \quad i \neq j \end{cases}$$

is the Kronecker delta and

$$k = \frac{1}{2} \left(\overline{(u'_1)^2} + \overline{(u'_2)^2} + \overline{(u'_3)^2} \right) = \frac{1}{2} \overline{u'_i u'_i}$$

the TKE. The turbulence eddy viscosity ν_t is solved by the turbulence model. Within the k-epsilon turbulence models, the turbulence eddy viscosity is defined as

$$\nu_t = C_\mu \frac{k^2}{\epsilon},$$

where C_μ is the model constant. Within Wilcox's k-omega turbulence models, the turbulence eddy viscosity is modelled as

$$\nu_t = \frac{k}{\omega}.$$

Within one equation models like the Spalart-Allmaras turbulence model, the turbulence eddy viscosity is resolved directly. A transport equation for the turbulence eddy viscosity is assembled.

All EVMs, does not matter if linear or nonlinear, base on the Boussinesq approximation. RSMs, in contrast, base on the theory of resolving the Reynolds stresses directly. They rely on the exact exact Reynolds stress transport equation. An equation for each Reynolds stress is resolved. RSMs are able to capture complex interactions in turbulent flows, such like tumbling, swirling and mixing.

The following paragraph shows exemplarily the mathematical setup of the k-omega SST turbulence model with its relation to the Boussinesq approximation.

[1] In his paper of 1897, Boussinesq published a row of approximations within the field of the hydrodynamics. But only the theory of turbulence is presented within this thesis.

The SST k-omega model: Following formulas and their relation are quoted from [17], [29] and [102]. The k-omega SST model is a two layer hybrid EVM. It combines the best of two worlds. In near-wall regions, the k-omega model is active, in the free shear flow, the k-epsilon model is applied. Between the two models, a blend is applied (F_1 and F_2 blending functions). The kinematic turbulence eddy viscosity is defined as

$$\nu_t = \frac{\alpha_1 k}{\max\left(\alpha_1 \omega, SF_2\right)},$$

where α_1 is the model constant for the k-omega mode and S is the shear rate tensor. By default, the F_2 blending function is active. In the DES mode in ANSYS Fluent, it can be switched manually to F_1 (refer to figure 4.2 and chapter 4.2.3). The blending functions are defined as

$$F_1 = \tanh\left\{\left\{\min\left[\max\left(\frac{\sqrt{k}}{\beta^* \omega y}, \frac{500\nu}{y^2\omega}\right), \frac{4\sigma_{\omega 2} k}{CD_{k\omega} y^2}\right]\right\}^4\right\}$$

and

$$F_2 = \tanh\left[\left[\max\left(\frac{2\sqrt{k}}{\beta^* \omega y}, \frac{500\nu}{y^2\omega}\right)\right]^2\right]$$

with

$$CD_{k\omega} = \max\left(2\rho\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}, 10^{-10}\right).$$

The TKE equation in the SST k-omega model is defined as

$$\frac{\partial k}{\partial t} + \overline{u}_j \frac{\partial k}{\partial x_j} = \frac{1}{\rho}\tilde{P}_k - \beta^* \omega k + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_k \nu_t)\frac{\partial k}{\partial x_j}\right]$$

and the omega equation as

$$\frac{\partial \omega}{\partial t} + \overline{u}_j \frac{\partial \omega}{\partial x_j} = \frac{\gamma}{\nu_t \rho} P_k - \beta\omega^2 + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_\omega \nu_t)\frac{\partial \omega}{\partial x_j}\right] + 2(1 - F1)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}.$$

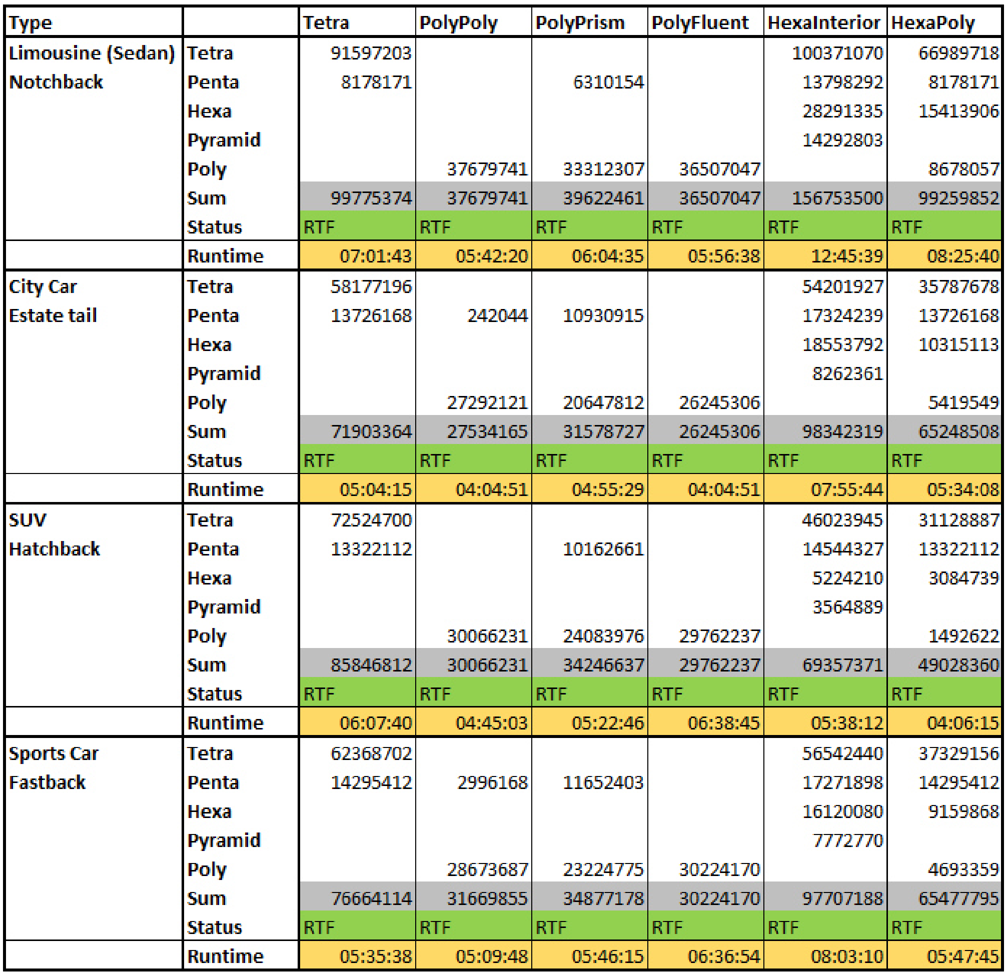
The rate of production for both equations is defined as followed:

$$P_k = -\overline{u_i' u_j'}\frac{\partial \overline{u}_j}{\partial x_j}$$

$$\tilde{P}_k = \min\left(P_k, C_l \epsilon\right)$$

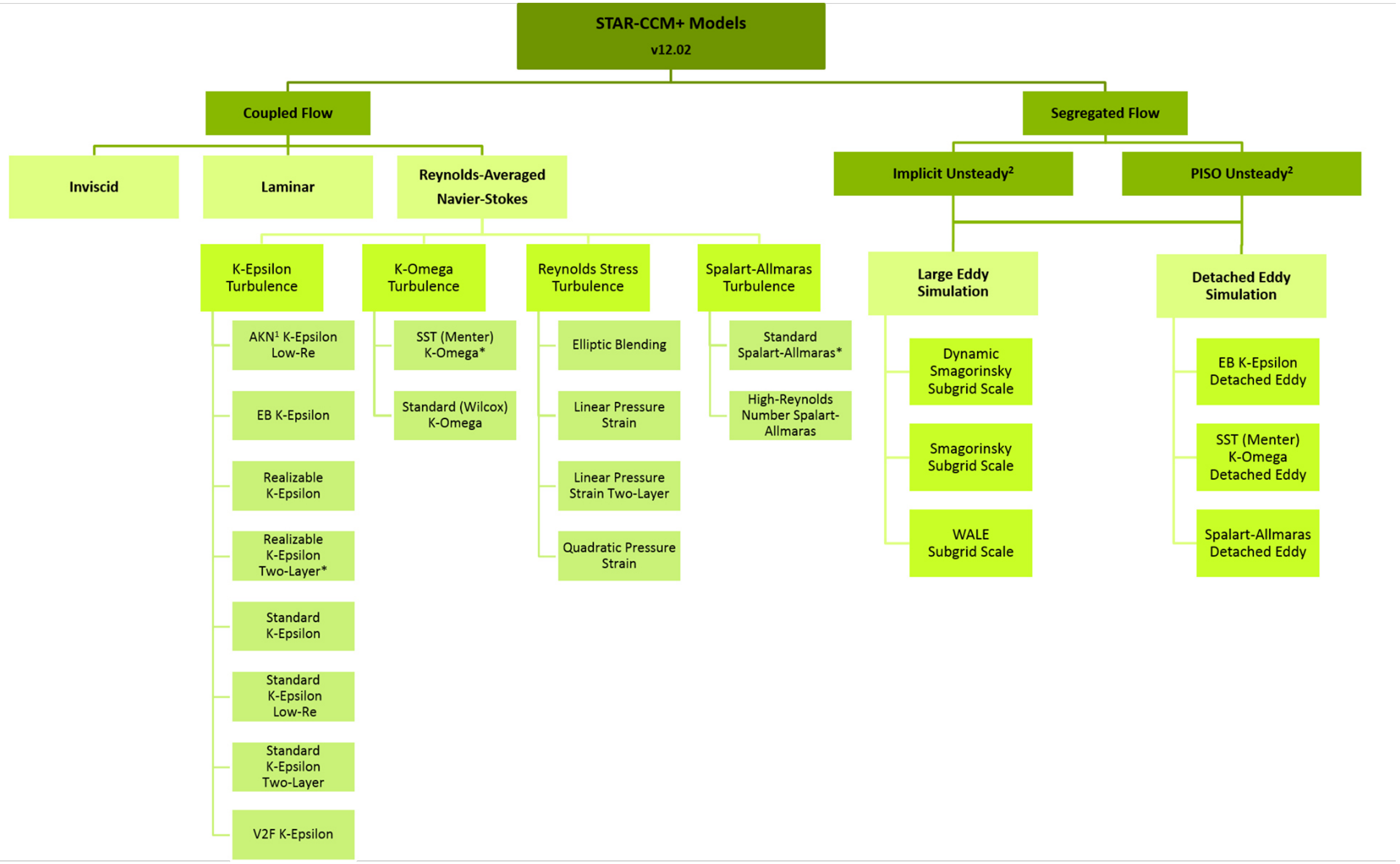
Values for model constants α_1, β, β^*, γ, σ_k, σ_ω, $\sigma_{\omega 2}$ and C_l can be found in the various literature. [17, 29, 102]

Figures and Tables



Type		Tetra	PolyPoly	PolyPrism	PolyFluent	HexaInterior	HexaPoly
Limousine (Sedan)	Tetra	91597203				100371070	66989718
Notchback	Penta	8178171		6310154		13798292	8178171
	Hexa					28291335	15413906
	Pyramid					14292803	
	Poly		37679741	33312307	36507047		8678057
	Sum	99775374	37679741	39622461	36507047	156753500	99259852
	Status	RTF	RTF	RTF	RTF	RTF	RTF
	Runtime	07:01:43	05:42:20	06:04:35	05:56:38	12:45:39	08:25:40
City Car	Tetra	58177196				54201927	35787678
Estate tail	Penta	13726168	242044	10930915		17324239	13726168
	Hexa					18553792	10315113
	Pyramid					8262361	
	Poly		27292121	20647812	26245306		5419549
	Sum	71903364	27534165	31578727	26245306	98342319	65248508
	Status	RTF	RTF	RTF	RTF	RTF	RTF
	Runtime	05:04:15	04:04:51	04:55:29	04:04:51	07:55:44	05:34:08
SUV	Tetra	72524700				46023945	31128887
Hatchback	Penta	13322112		10162661		14544327	13322112
	Hexa					5224210	3084739
	Pyramid					3564889	
	Poly		30066231	24083976	29762237		1492622
	Sum	85846812	30066231	34246637	29762237	69357371	49028360
	Status	RTF	RTF	RTF	RTF	RTF	RTF
	Runtime	06:07:40	04:45:03	05:22:46	06:38:45	05:38:12	04:06:15
Sports Car	Tetra	62368702				56542440	37329156
Fastback	Penta	14295412	2996168	11652403		17271898	14295412
	Hexa					16120080	9159868
	Pyramid					7772770	
	Poly		28673687	23224775	30224170		4693359
	Sum	76664114	31669855	34877178	30224170	97707188	65477795
	Status	RTF	RTF	RTF	RTF	RTF	RTF
	Runtime	05:35:38	05:09:48	05:46:15	06:36:54	08:03:10	05:47:45

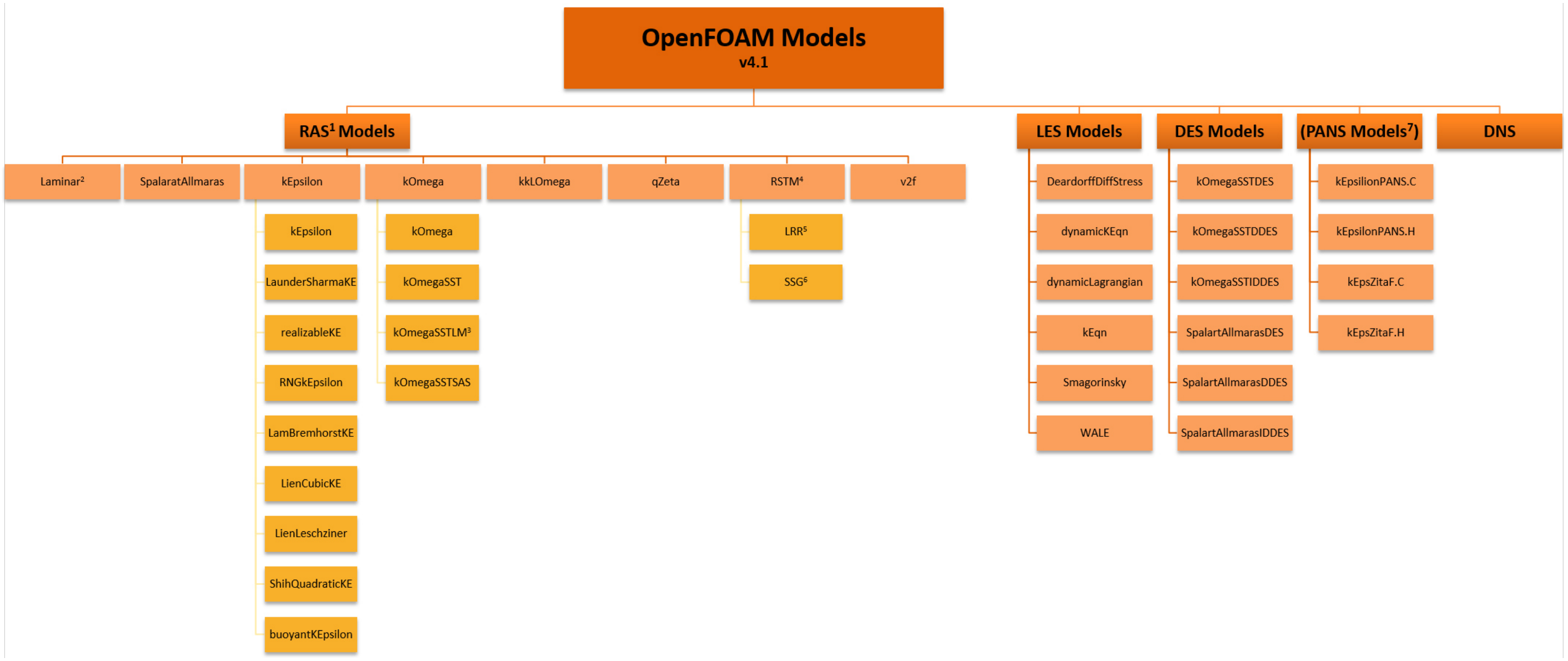
Full mesh number table (RTF = Ready To Fluent)


STAR-CCM+ Models
v12.02
Coupled Flow
Segregated Flow
Inviscid
Laminar
Reynolds-Averaged Navier-Stokes
Implicit Unsteady[2]
PISO Unsteady[2]
K-Epsilon Turbulence
K-Omega Turbulence
Reynolds Stress Turbulence
Spalart-Allmaras Turbulence
Large Eddy Simulation
Detached Eddy Simulation
AKN[1] K-Epsilon Low-Re
EB K-Epsilon
Realizable K-Epsilon
Realizable K-Epsilon Two-Layer*
Standard K-Epsilon
Standard K-Epsilon Low-Re
Standard K-Epsilon Two-Layer
V2F K-Epsilon
SST (Menter) K-Omega*
Standard (Wilcox) K-Omega
Elliptic Blending
Linear Pressure Strain
Linear Pressure Strain Two-Layer
Quadratic Pressure Strain
Standard Spalart-Allmaras*
High-Reynolds Number Spalart-Allmaras
Dynamic Smagorinsky Subgrid Scale
Smagorinsky Subgrid Scale
WALE Subgrid Scale
EB K-Epsilon Detached Eddy
SST (Menter) K-Omega Detached Eddy
Spalart-Allmaras Detached Eddy

[1] AKN = Abe-Kondoh-Nagano

[2] LES and DES can only be chosen, when implicit unsteady or PISO (Pressure Implicit with Splitting of Operator) unsteady is active.

Full turbulence model structure of CD-adapco STAR-CCM+ v11.04 [137]



[1] RAS = Reynolds-Averaged Simulation
[2] Dummy turbulence model for laminar flow
[3] LM = Langtry-Menter
[4] RSTM = Reynolds Stress Model
[5] LRR = Launder, Reece and Rodi
[6] SSG = Speziale, Sarkar and Gatski
[7] The official OpenFOAM manual does not include any PANS model. However, some papers on implementing the model into OpenFOAM are available. [110]

Full turbulence model structure of OpenFOAM v4.1 [120]