An Explicit Discontinuous Galerkin Method for Parallel Compressible Two-Phase Flow Simulations

A thesis accepted by the Faculty
of Aerospace Engineering and Geodesy of the University of Stuttgart
in partial fulfillment of the requirements for the degree of
Doctor of Engineering Sciences (Dr.-Ing.)

by

Malte Hoffmann

born in Goslar

Main referee:	Prof. Dr. rer. nat. Claus-Dieter Munz
Co referee:	Prof. Dr.-Ing. Michael Dumbser
Date of defence:	August 10, 2017

Institute of Aerodynamics and Gas Dynamics
University of Stuttgart

2017

Preface

This thesis was written during my work as academic employee at the Institute of Aerodynamics and Gas Dynamics (IAG) of the University of Stuttgart.

I have first to express my gratitude to my doctoral supervisor Prof. Dr. Claus-Dieter Munz for the exeptional working conditions in his research group and for his scientific advice. Furthermore, I thank Prof. Dr. Michael Dumbser for being my co-referee.

For the pleasant working atmosphere and fruitful scientific discussions I want to thank all my colleags at the IAG, especially to all past and present members of the Numerics Research Group.

This thesis was developed within a project financed by the Federal Ministry of Education and Research of Germany.

Aalen, October 2017

Malte Hoffmann

Contents

Symbols and Abbreviations

Symbols

Latin symbols

a	input value for the quadtree (x-axis)
$\vec{a}$	covariant basis functions
b	input value for the quadtree (y-axis)
B	identification binary number
c	speed of sound
$\mathcal{C}$	sub-cell element
d	spatial dimension
$\underline{D}$	differentiation matrix
e	specific internal energy
E	specific total energy
$\mathcal{E}$	reference element
$\vec{\mathbf{F}}$	flux vector in physical space
$\vec{\boldsymbol{\mathcal{F}}}$	flux vector in reference space
g	Gibbs free energy
$\vec{\boldsymbol{\mathcal{G}}}$	inviscid flux vector in reference space
$\vec{\boldsymbol{\mathcal{H}}}$	viscous flux vector in reference space
$\underline{I}$	unit matrix
J	determinant of the Jacobian matrix
$\underline{J}$	Jacobian matrix
l	quadtree level
ℓ^N	1D Lagrange polynomial of degree N
L	left Riemann state

L_∞	error-criterion
L_2	error-criterion
L_q	maximum level of the quadtree
$\mathbb{L}_2$	space of square-integrable functions
$\vec{m}$	polynomial mapping function
$\underline{M}$	mass matrix
N	polynomial degree
$\vec{n}$	normal vector
p	pressure
$\mathbf{P}$	source term
Pr	Prandtl number
q	quadtree variable
$\vec{q}$	heat flux vector
Q	arbitrary area
$\mathcal{Q}$	quadtree domain
R_s	specific gas constant
R	right Riemann state
$\underline{R}$	right eigenvector matrix
$\hat{s}$	surface element
S	wave speed
$\vec{\mathbf{S}}$	approximation to the solution gradients
t	time
$\vec{t}$	tangential vector
T	temperature
$\underline{\mathcal{T}}$	transformation matrix
$\mathbf{U}$	vector of conservative variables
$\hat{\mathbf{U}}$	solution at the solution points
$\vec{\mathcal{U}}$	vector of conservative variables in reference space
$\bar{\mathbf{U}}$	mean value in finite volume sub-cell
$\mathbf{U}_p$	vector of primitive variables
$\mathfrak{U}$	numercial flux
$\vec{v}$	velocity vector

$\underline{V}$	Vandermonde matrix
W	characteristic solution
$\vec{x}$	physical space coordinates

Greek symbols

α	vapor volume fraction
χ	vapor quality
δ	Kronecker delta function
Δ^l	quadtree element length for a specific level
Δt	time step
ϵ	error limit
κ	adiabatic coefficient
λ	heat conductivity
μ	viscosity
$\vec{\nabla}_x$	gradient operator in physical space
$\vec{\nabla}_\xi$	gradient operator in reference space
ω	integration weights
Ω	cut-cell area
$\boldsymbol{\phi}$	test functions
Φ	Helmholtz free energy formulation
ϖ	sub-cell length
ψ	interpolation basis function
ρ	density
σ	cavitation number
$\underline{\tau}$	viscous stress tensor
$\vec{\xi}$	reference space coordinates

Abbreviations

1D,2D,3D	one-, two-, three-dimensional
ADER	arbitrary high order using derivatives
BR1	first method of Bassi and Rebay
CFD	computational fluid dynamics
CP	critical point
DG	discontinuous Galerkin
DGSEM	discontinuous Galerkin spectral element method
DOF	degrees of freedom
FE	finite element
FV	finite volume
HLLC	Harten-Lax-van Leer-contact
HLRS	high performance computing center Stuttgart
HPC	high performance computing
IAPWS-95	international association for the properties of water and steam formulation 1995
ID	identification
LES	large eddy simulations
LLF	local-Lax-Friedrich
LSERK4	low storage explicit Runge-Kutta 4^{th} order
MPI	message passing interface
NACA	national advisory committee for aeronautics
NSE	Navier-Stokes equations
PID	performance index
RP-W	Riemann problem for water
SFC	space-filling curve
w/o	without

Kurzfassung

In dieser Arbeit wird ein effizientes numerisches Verfahren zur Simulation von kavitierenden Strömungen reiner Fluide vorgestellt. Die Simulation von kavitierender Strömung bringt verschiedene Anforderungen mit sich. Zum einen muss der Phasenübergang zwischen der Dampf- und Flüssigkeitsphase betrachtet werden und zum anderen ist eine hoch auflösende Numerik erforderlich, die die auftretenden räumlichen und zeitlichen Skalen abbilden kann. Zusätzlich können bei Kavitation die thermodynamischen Größen (z. B. Dichte, Druck) um mehrere Größenordnungen über einen sehr kleinen räumlichen Bereich variieren. Die starken Gradienten müssen von dem numerischen Verfahren dargestellt werden können.

Um den Phasenübergang zu berücksichtigen, ist in dieser Arbeit eine realistische Zustandsgleichung erforderlich, die in der Lage ist, die Dampf-, Flüssigkeits- und Zweiphasenbereiche eines solchen Fluids aufzulösen. CoolProp, eine thermodynamische Parameterdatenbank für über 100 Flüssigkeiten, ist für diese Aufgabe gut geeignet, da sie die sehr genaue Helmholtz-Energie Formulierung als Zustandsgleichung verwendet. Unter der Annahme des thermodynamischen Gleichgewichts und der Anwendung der Maxwell-Konstruktion im Zweiphasenbereich können die kompressiblen Navier-Stokes Gleichungen durch die Zustandsgleichung aus der CoolProp-Bibliothek geschlossen werden. Sowohl im Zweiphasenbereich als auch im Dampfzustand sind Kompressibilitätseffekte zu berücksichtigen. Reibung und Wärmefluss werden durch die Navier-Stokes Gleichungen ebenfalls dargestellt.

Aus der Klasse der numerischen Verfahren ist die diskontinuierliche Galerkin Methode ein guter Kandidat, um die auftretenden räumlichen und zeitlichen Skalen abzubilden. Das hier verwendete diskontinuierliche Galerkin Spektralelementverfahren hoher Ordnung löst die Navier-Stokes Gleichungen mit einer expliziten Zeitintegration. Dieses Verfahren ist für seine niedrige numerische Dissipation und gute Skalierungsfähigkeiten auf Hochleistungscomputern der neuesten Generation bekannt. Ein Nachteil ist aber, dass es weder

Stöße noch starke Gradienten behandeln kann. Diese auftretenden physikalischen Phänomene müssen durch eine Methode abgebildet werden, die mit diesen umgehen kann, aber die gute Skalierungsfähigkeit nicht beeinträchtigt. Für diesen Fall wird ein Finite-Volumen Subzellansatz zweiter Ordnung vorgestellt. Die Finite-Volume Methode ist nur in den Elementen aktiv, in denen die diskontinuierliche Galerkin Methode nicht in der Lage ist, diese starken Gradienten bzw. die Stöße stabil aufzulösen.

Da die Auswertung der Helmholtz-Energie Formulierung mit hoher Rechenzeit verbunden ist, wird die Zustandsgleichung in einem der Simulation vorausgehenden Arbeitsschritt in eine Polynomdarstellung auf einem hierarchisch adaptierbaren Gitter (sog. Quadtree) gebracht. Dieser Schritt wird parallelisiert und kann auf einer beliebig großen Anzahl von Prozessoren durchgeführt werden. Die Daten des Quadtree werden gespeichert und die polynomiale Näherung bildet die ursprüngliche Lösung bis zu einem frei gewählten Fehler ab. Während der Simulation werden die gespeicherten Quadtrees in den Speicher jedes an der Simulation beteiligten Prozessors geladen. Dann werden die Daten der Quadtrees ausgewertet, um die notwendigen Beziehungen zwischen den thermodynamischen Größen bereitzustellen. Dieser Ansatz reduziert die Rechenzeit um drei Größenordnungen gegenüber der direkten Auswertung und eignet sich somit sehr gut für Berechnungen auf Hochleistungsrechnern.

Der hier vorgestellte Ansatz wird validiert und mit Ergebnissen aus der Literatur für eindimensionale Simulationen verglichen. Die gewünschte Konvergenzrate wird erreicht und die erzielten Ergebnisse stimmen sehr gut mit den Referenzdaten aus der Literatur überein. Eine zweidimensionale Simulation, bei der Wasser um eine Tragfläche strömt, zeigt die Bildung von Kavitation. Die beim Zerfall der Kavitationsgebiete entstehenden starken Druckwellen werden von der Simulation erfasst und numerisch stabil aufgelöst. Abschließend wird das hier entwickelte Verfahren für eine komplexe, dreidimensionale Anwendung aus der Industrie benutzt, um die Güte der hier vorgestelleten Methode zu demonstrieren und zu zeigen, dass komplexe Multiskalprobleme auf mehreren tausend Prozessoren in angemessener Zeit berechnet werden können. Für diese industrielle Anwendung wird auch die gute Skalierung auf einem Hochleistungsrechner gezeigt.

Abstract

In this thesis an efficient numerical method is presented to enable simulations with cavitating flow for a pure fluid. The simulation of cavitating flow poses various challenges. On the one hand, the phase transition between the vapor and liquid phase must be considered, and on the other hand a high-resolving numerical method is required, which can resolve the occurring spatial and temporal scales. In addition, in the presence of cavitation, the thermodynamic quantities (e.g. density, pressure) can vary by several orders of magnitude over a very short distance.

To consider phase change, in this work an accurate equation of state is needed which is able to resolve the vapor, liquid and two-phase regions of such a fluid. CoolProp, a thermodynamic property database for over 100 fluids, is well suited for this task since it uses the most-accurate Helmholtz free energy formulation as equation of state. Assuming thermodynamic equilibrium and using the Maxwell construction in the two-phase region, the compressible Navier-Stokes equations can be closed by the equation of state from the CoolProp library. Compressibility effects need to be considered in the two-phase region as well as in the vapor state. Also friction and the heat flux are represented by the Navier-Stokes equations.

From the class of the numerical methods, the discontinuous Galerkin method is a good candidate to resolve the occurring spatial and temporal scales. The here used discontinuous Galerkin spectral element method solves the Navier-Stokes equations with an explicit time integration. This method is known for its low numerical dissipation and good scaling capabilities on state of the art high performance computers. A disadvantage of this method is that it can handle neither shocks nor high gradients. These occurring physical phenomena must be resolved by a method which can deal with these phenomena, but does not affect the good scaling ability. For this case a second order finite volume sub-cell approach is presented. The finite volume method is only ac-

tive in those elements where the discontinuous Galerkin method is not able to resolve the high gradients or shocks.

Since the evaluation of the Helmholtz free energy formulation is linked with high computational effort, the equation of state is stored in a preprocessing step as a polynomial representation on a hierarchically adaptable grid (so-called quadtree). This step is parallelized and performs on an arbitrary number of processors. The data of the equation of state are stored in a polynomial representation with a user-defined error compared to the original solution. During the simulation the stored quadtrees are loaded into the memory of each processor. Then the data of the quadtrees are used to provide the necessary relations between the thermodynamic variables. This approach reduces the computational effort by three orders of magnitude compared to the direct evaluation and is well suited for calculations on state of the art high performance computers.

The presented method is validated and it is compared to results in literature for one dimensional simulations. The desired convergence rate is reached and the obtained results are in very good agreement with the reference data from the literature. A two dimensional calculation shows water streaming around a hydrofoil producing cavitation. The strong pressure waves arising during the collapse of the cavitation regions are captured by the simulation and resolved in a numerically stable manner. Finally, the framework developed here is applied to a complex, three-dimensional application from the industry to demonstrate the quality of the method and to show that complex multiscale problems can be calculated on several thousand processors in a reasonable time. For this industrial application also the good scaling on a high performance computer is shown.

1 Introduction

In the last decades, numerical simulation has become an indispensable method for research and development in many areas such as meteorology and finance, but especially in the field of computational fluid dynamics (CFD). While only simple steady-state calculations were possible in the early days of engineering applications with regard to CFD, the use of high-performance computing (HPC) clusters and efficient methods made it possible to consider unsteady complex multiscale problems. Such problems can be solved more and more satisfactorily for single phase flows. The focus of academic and industrial research moves currently also to multiphase flows, which increase the complexity of numerical simulations even more. Multiphase flows occur when the state of matter in a fluid flow changes and more than one state of matter coexist. Particularly two-phase flows can be observed in hydraulic system like injection equipment of modern combustion engines as well as in high-pressure pumps or at the propeller of ships. In all three of these examples, a significant pressure drop can occur in the flow, due to acceleration of the fluid, which leads to evaporation and so liquid and vapor phase coexist. These developed evaporation areas can condensate abruptly, due to, e.g., a higher surrounding pressure. This formation and the collapse of the evaporated areas is called cavitation and is associated with strong gradients in the flow properties and strong non-linear unsteady phenomema like pressure waves. The collapse of these cavitation bubbles can lead to high pressure waves which permanently damage parts of the injection equipment, pumps and propellers. The need for numerical simulations is especially evident for injection components. Due to the small size of these devices, experimental data is hard to obtain. To resolve the change of the state of matter like evaporation, condensation and therefore cavitation, a highly resolving numerical method is needed, which can handle the resulting temporal and spatial scales as well as represent the behavior of the thermodynamics in cavitating flows correctly. Due to these high requirements it must be assured that the numerical approach can be used efficiently

on state of the art high performance computing clusters. The aim of this work is to develop such a numerical approach.

1.1 Discontinuous Galerkin Methods

The class of high order discontinuous Galerkin (DG) methods has gained significant importance during the last years. It combines the advantageous features of a high-order approach with great flexibility and it is also an optimal method for state of the art multiscale computations with very good parallelization properties. These properties result from the element local structure of the DG method. Similar to finite element (FE) methods the solution of the DG method is approximated by a polynomial. The difference to the classical FE methods is that the approximation is allowed to be discontinuous at element boundaries, which is also the case for finite volume (FV) methods. A particular modification of DG methods is the discontinuous Galerkin spectral element method (DGSEM) [30, 35]. This method is limited to hexahedral elements but therefore it can be implemented very efficiently because a tensor product basis can be used. Due to its mostly element local structure it is also very efficient on high-performance computers [2]. The disadvantage of the restriction to hexahedral elements can be limited, since the DGSEM allows to use curved surfaces and elements as well as unstructured meshes, which makes the calculation of complex geometries possible [29]. The great advantage of good scalability on HPC-clusters is one of the main reasons why the DGSEM can be considered for the calculation of two-phase flows, because those calculations require a lot of computing effort, due to the complexity of these flows. Another reason to use this method is the ability to solve multiscale problems very well [7]. Other applications for the DGSEM are direct numerical simulation of laminar and turbulent single-phase flows as well as large eddy simulations (LES) of turbulent flows [8] based on implicit LES models. This method can also be used for the direct aeroacoustic simulations of turbulent flow noise [21, 22]. Further it has been shown that the DGSEM is well suited to solve the Maxwell equations with particle in cell methods [41, 42, 43]. The DGSEM was also used to calculate single droplets in two-phase simulations at extreme conditions [19].

1.2 Two-Phase Flow

A two-phase flow occurs when two of the three classical states of matter (liquid, solid, vapor) occur side by side. In this thesis only the simultaneous occurrence of liquid and vapor phase as well as the behavior of the two-phase region is considered. If several fluids are present in the two-phase flow, it is called multi-component two-phase flow. This kind of flow is not covered in this work. The changes of state which can occur in the liquid-vapor two-phase flows are condensation (transition from vapor to liquid phase) and evaporation (transition from liquid to vapor phase). Of particular interest in this work is the occurrence of cavitation.

1.2.1 Cavitation

Cavitation is derived from the Latin term *cavus* = hollow and describes the appearance of vapor due to a pressure decrease and the abrupt collapse of vapor or two-phase regions in fluids. The phenomenon of cavitation has been studied experimentally for several hundred years and numerically for the last decades. For the first time cavitation was observed by Sir Isaac Newton in 1704 and was later described theoretically by Leonhard Euler in 1754 [73]. By a pressure drop within the flow (for example by accelerating the fluid at ship propellers or in throttles), the static pressure drops to the vapor pressure and evaporation leads to the formation of cavitation regions. These areas can have the form of single bubbles, clouds or whirls, and can collapse by condensation due to a pressure increase in the surrounding fluid. Cavitation is always a highly unsteady process and the formation and collapse of the cavitation areas take place in a period of nano- and microseconds. In the decay process, shock waves can occur with very large pressure amplitudes, as well as high gradients in density and pressure. Snapping shrimps make use of these pressure waves by generating cavitation with a specialized claw, thereby stunning or killing their prey [38, 68]. In hydraulic systems, the occurrence of cavitation has advantages and disadvantages. On the one hand the mass flow can be controlled by targeted generation of cavitation, but on the other hand the collapse and the associated pressure waves produce noise and damage the components. Also on ship propellers, cavitation leads to damage and CFD

simulations are becoming increasingly important to get a better understanding of how to prevent cavitation [11].

1.3 Two-Phase Flow in Computational Fluid Dynamics

Two-phase flows can be simulated in different ways. The main methods to solve two-phase flows are the following: one class of methods takes account of the phase boundary between the two phases and another class is based on an average phase state.

If both phases are considered separately from each other, the phases do not necessarily have to be in thermodynamic equilibrium since they may have different thermodynamic states. This separation can be performed with the diffuse interface approach or with the sharp interface approach. The diffuse interface approach, which is characterized by a continuous transition between the two phases, leads to a discrete thickness of the interface. A typical representative of this method is the Baer-Nunziato model, which contains three conservation equations for each phase and additionally a transport equation for the volume fraction of a phase from which the velocity of the phase boundary surface is solved [54, 55]. The difficulty of this model lies in the thermodynamically consistent smearing of the phase boundary, since this is in reality a discontinuity. For the sharp interface approach proposed by Fedkiw et al. [20], the smearing is compensated by a discontinuous approximation of the transition. These discontinuities must be resolved by the numerical method. Appropriate jump conditions as well as the exact position of the discontinuity must be known in the sharp interface approach. In order to determine the exact position of the phase boundary, there are again several approaches. Mostly used are the volume of fluid approach introduced by Hirt and Nichols [31], and the level-set method introduced by Osher and Sethian [44]. Due to the high computational effort, both approaches for resolving the phase interface are usually used only for droplet examinations. With both approaches the influence of surface tension can be taken into account. But because of the high computational effort, it is not used in this work.

If an average phase state is considered, the simplest model for the description of two-phase flows, the so-called barotropic model can be used, or alterna-

tively the more advanced homogeneous equilibrium two-phase model. In both methods thermodynamic equilibrium and the same velocities for both phases are assumed. The equation of state (EOS) in the barotropic model can be expressed by $p = p(\rho)$. This requirement leads to the fact that usually only the mass and impulse equations are considered. In this case the influence of the temperature is neglected, which in reality has an effect on the vapor pressure and also on the density. For the homogeneous equilibrium model the EOS can be expressed by $p = p(\rho, T)$ and thus the influence of the temperature can be taken into account. In this case, the Euler or Navier-Stokes equations (NSE) are mostly solved. The underlying EOS must be able to represent the liquid as well as vapor phase, but also the two-phase region. Surface tension cannot be taken into account without resolving the interface, but since it is not essential for the investigation of cavitating flows this is not considered in this work.

1.3.1 Equation of State

In this work the approximation of the thermodynamic relationships between the liquid and vapor phase is an important part. This approximation depends essentially on the chosen equation of state. While no EOS is needed in the incompressible description, for a compressible description an EOS is necessary since then the conservative equations are coupled. The task is to correctly describe the behavior of two-phase phenomena by the EOS, whereby the resolution of the (weakly) compressible liquid region as well the resolution of the compressible vapor or gas region have to be considered. Advantages and disadvantages of different EOS are described in the literature and only a short extract is given here. To resolve the different effects of phase transition in the two-phase domain at least a cubic EOS must be used which can be written as a cubic function of the density.

The class of the cubic EOS includes, e.g, the van der Waals EOS [69] and Peng-Robinson EOS [45]. A complete overview about other variations of this kind of EOS is given by Poling et al. [47]. The van der Waals EOS can be solved analytically, while the Peng-Robinson EOS has to be solved iteratively since in this EOS the temperature appears in the equation of pressure nonlinearly. The problem of all cubic EOS is, that they have negative pressure gradients in the two-phase region, which leads to non-physical results in the speed of sound. For all methods which resolve the phase boundary these neg-

ative gradients are not relevant, but for the homogeneous equilibrium model, the Maxwell construction must be performed in the two-phase domain to obtain physically consistent results. The advantage of this type of EOS is its relatively simple form; in some cases, even analytical descriptions can be used to reproduce the behavior of the fluid well. Limitations are around the critical point and under extreme ambient conditions with high temperatures and pressures, where this equation of state, due to its simplicity, cannot reflect the physics correctly.

Setzmann und Wagner [59, 60] have presented a more complex method which represents the reality more precisely, even at the critical point as well as in extreme environmental conditions, bringing the approximation error to below 1% for all phases. Around the critical point normally the largest approximation error is found, the other areas can usually be approximated with an error around 1 ‰. The so-called Helmholtz free energy formulation contains 65 terms for water (with the IAPWS-95 standard [70]) to represent the realistic behavior of the fluid in all phases. For many fluids parameters are available and also calculation programs such as CoolProp [9] and Refprop [36]. These programs can be used to evaluate the equation of state. The major deficit of these programs is the high computation time needed to evaluate the EOS.

In this thesis, the Helmholtz free energy formulation is used to ensure a precise description of the fluid in all areas. The high computing time is drastically reduced by a special preprocessing step explained in this work.

1.3.2 Cavitation in Computational Fluid Dynamics

Many studies have dealt with the calculation of two-phase flows with the appearance of cavitation in order to obtain a better understanding of this phenomenon [15, 17, 18, 24, 32, 53, 63, 71]. Most of these approaches are based on the barotropic two-phase model. To take into account the influence of the temperature in the case of cavitating flows, the homogeneous two-phase model with a high-precision EOS is applied in this work. The NSE are solved with the DGSEM to be able to resolve friction effects and the influence of the heat flux.

1.4 Shock Capturing

Caused by cavitation or other flow conditions high pressure or density gradients and shocks can occur in two-phase flows. The DG method cannot resolve these effects, because a polynomial representation of a discontinuity does not converge and leads to oscillations. If no special care is taken against the oscillations, it is not possible to assure that the positivity of density and pressure can be maintained, which is necessary to represent the physics correctly. Without any care it leads to the termination of the simulation. It is necessary to resolve these high gradients and shocks using a shock capturing technique.

One idea to put this into practice originally comes from the FV method and is called artificial viscosity, see, e.g., Jameson et al. [34]. This theory has been extended in recent years to the context of the DG procedures by Persson and Perraire [46] and they provide a consistent treatment of the artificial viscosity terms in the volume and surface integrals. The idea is to add the viscosity to the discontinuities in order to be able to solve them directly with a high-order method. This approach smears the discontinuity in order to be able to represent it with a polynomial and in addition the time step becomes smaller than without artifical viscosity.

Another idea to solve those problems is the ENO / WENO reconstruction. Low order information is used to reconstruct high order data and by choosing the least oscillating polynomial it is guaranteed that the reconstruction process takes care of the oscillation behavior. In case of oscillation, all the high order information which causes the oscillations is removed. In this case a way of reconstruction could be, e.g., instead to take into account all information covering the discontinuity, only one-sided data can be used. Essentially non-oscillatory (ENO) [25] and weighted ENO (WENO) [37] include the measurement of oscillations for the shock capturing.

The h/p-adaption is a promising approach to establish shock capturing in a DG method by reducing the order of the polynomial and simultaneously refining the grid [6, 10]. Huerta et al. [33] described an idea for DG methods, in which the space for the polynomial approach was extended by piecewise constant functions in sub-cells of the original DG element. Another procedure in which the DG element is replaced by finite volume sub-cells is presented by Sonntag et al. [62]. In their approach the number of degrees of freedom is kept constant. In general the reduction of the polynomial degree leads to a decrease

of oscillation but also to a decline in the solution quality. This decline can be prevented by increasing the resolution with h-refinement.

In this thesis the method of Sonntag et al. [62] is used in a slightly modified form. It can be implemented very efficiently and does not greatly reduce the performance on high-performance computers. And more importantly, it does not affect the time step negatively. The FV method is used with a second order reconstruction where the reconstructed gradients are limited. The class of total variation diminishing (TVD) limiters has proved to be very robust. Among others the minmod and the superbee limiter of Roe [50] are well-known TVD limiters in the FV context. In this work the minmod limiter is used. As all TVD limiter, it falls back to a first order FV method if new extrema would be generated. Since this can lead to a loss of accuracy, care must be taken in which areas the FV sub-cell process is active. This is implemented in this work with an indicator of Persson and Peraire [46], which detects DG elements with oscillations and activates the FV sub-cell procedure if necessary.

1.5 High Performance Computing

HPC is widely used in the CFD community, because the calculation of unsteady multiscale problems are computation time consuming due to the necessity to resolve all scales or at least to model them. Over the last years the resolution of CFD calculations raised from thousands of solution points to millions and even over several hundred million points. The ability to split the problem into subproblems and solve these subproblems on many cores in a parallel fashion makes such big calculations possible in a reasonable amount of time. Many industrial companies installed HPC clusters with several thousand processors. In order to solve the simulations, which are presented in this work, in an acceptable time, a good scalability on several thousand processors is required. The scalability of a software describes the behavior of its parallel performance. An ideal scaling is realized if by doubling the number of cores, the simulation time is halved. The DGSEM is known to be efficient on high-performance computers [2] and also the shock capturing used here is very suitable in the HPC context [62]. For the two-phase flow simulations the use of the highly-accurate EOS may not lead to a significant performance loss on HPC clusters. In this work a main focus is to solve industrial two-phase

flows almost as efficiently as the single-phase flow on several thousands of processors. All simulations shown in this work are calculated on the Cray XC40 'HAZELHEN' of the High Performance Computing Center Stuttgart (HLRS) of the University of Stuttgart.

1.6 The HONK Project

The Federal Ministry of Education and Research (BMBF) supported in the third BMBF HPC-Call the project HONK "Industrialization of high-resolution numerical analysis of complex flow phenomena in hydraulic systems". The aim of the HONK project was to simulate and analyze complex flow phenomena, like cavitation, in an industrial application context. Four project partners have worked three years to reach this goal. Partners of the project were the Robert Bosch GmbH, Dept. CR/ARF, the Visualisation Research Centre, the High Performance Computing Center Stuttgart and the Institute of Aerodynamics and Gas Dynamics (IAG). The last three are all institutes of the University of Stuttgart. This thesis was written during my work for the HONK project at the IAG.

1.7 Objectives

The aim of this work is the development of a high order DGSEM for industrial two-phase flow applications with cavitation. In this context the following aspects are covered:

- Extending the DGSEM for the compressible Navier-Stokes equations to two-phase flows with the thermodynamic equilibrium assumption.
- Efficient implementation and usage of the Helmholtz free energy formulation as EOS
- Implementation and verification of the FV sub-cell shock capturing method
- Demonstrate the performance for an industrial application on a HPC cluster

1.8 Outline

The structure of this thesis is as follows: in Chapter 2 the underlying equations and their numerical discretization are explained. This includes the NSE as well as the Helmholtz free energy formulation. Also the DGSEM is described with the FV sub-cell shock capturing. In Chapter 3 an efficient storage and evaluation technique for the Helmholtz free energy formulation is presented which reduces the evaluation time for the accurate EOS by several orders of magnitude. The validation of the CFD solver and simulations in one, two and three dimensional space are shown in Chapter 4 as well as the scaling on a HPC machine. The thesis ends with a conclusion and the outlook on future work in Chapter 5.

2 Numerics

In this chapter the fluid flow equations used for the CFD simulations and also the numerical methods are described. The compressible Navier-Stokes equations are closed with a realistic equation of state, namely the Helmholtz free energy formulation. This equation of state gives a very good approximation to the gas and liquid phase. Especially the two-phase region, where gas and liquid coexist, is approximated with a thermodynamic equilibrium assumption and Maxwell construction. The main method to solve this equation system implemented in this work is a Discontinuous Galerkin Spectral Element Method, but since during phase change high gradients can occur, a shock capturing technique is needed. In the elements containing the gradients a 2^{nd} order finite volume method is activated, if necessary, to handle these gradients. For both methods Riemann solvers are needed and two of them are explained in more detail. Also the used boundary conditions are explained as well as the time integration scheme. At the end of this chapter a short overview over the parallelization technique is given.

2.1 Navier-Stokes Equations

The three dimensional compressible Navier-Stokes equations can be written as

$$\frac{\partial \mathbf{U}}{\partial t} + \vec{\nabla}_x \cdot \vec{\mathbf{F}}_{\mathrm{a}}(\mathbf{U}) - \vec{\nabla}_x \cdot \vec{\mathbf{F}}_{\mathrm{v}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U}) = \mathbf{P}, \tag{2.1}$$

where $\mathbf{U}$ is the vector of conservative variables $\mathbf{U} = (\rho, \rho\vec{v}, \rho E)^T$, where ρ is the density, $\vec{v} = (v_1, v_2, v_3)^T$ is the velocity vector in three space dimensions and E is the specific total energy. $\mathbf{P} \in \mathbb{R}^5$ is the source term vector, $\vec{\mathbf{F}}_{\mathrm{a}} = (\mathbf{F}_{\mathrm{a}_1}, \mathbf{F}_{\mathrm{a}_2}, \mathbf{F}_{a_3})^T \in \mathbb{R}^{3\times 5}$ are the inviscid or advection fluxes and

$\vec{\mathbf{F}}_{\mathrm{v}} = (\mathbf{F}_{\mathrm{v}_1}, \mathbf{F}_{\mathrm{v}_2}, \mathbf{F}_{\mathrm{v}_3})^T \in \mathbb{R}^{3\times 5}$ are the viscous or diffusion fluxes in three-dimensional space. The gradient operator $\vec{\nabla}_x$ is defined as

$$\vec{\nabla}_x = \left(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \frac{\partial}{\partial x_3}\right)^T . \tag{2.2}$$

The notation is as follows: bold characters (like $\mathbf{U}$) represent a vector of conservative or primitive variables needed for the Navier-Stokes equations. All vectors in space directions are symbolized with an arrow (like $\vec{v}$). This means that the flux (like $\vec{\mathbf{F}}$) is a vector with components in each space direction where each direction contains a vector of needed Navier-Stokes variables. The vector of the advection fluxes is given by

$$\mathbf{F}_{\mathrm{a}_d}(\mathbf{U}) = \begin{pmatrix} \rho\, v_d \\ \rho\, v_1 v_d + \delta_{1d}\,\mathrm{p} \\ \rho\, v_2 v_d + \delta_{2d}\,\mathrm{p} \\ \rho\, v_3 v_d + \delta_{3d}\,\mathrm{p} \\ \rho\, E v_d + \mathrm{p}\, v_d \end{pmatrix}, \; d = 1, 2, 3, \tag{2.3}$$

with δ_{ij} being the Kronecker delta function. The viscous fluxes vector can be written as

$$\mathbf{F}_{\mathrm{v}_d}(\mathbf{U}, \vec{\nabla}_x \mathbf{U}) = \begin{pmatrix} 0 \\ \tau_{1d} \\ \tau_{2d} \\ \tau_{3d} \\ \sum_{j=1}^{3} \tau_{dj} v_j - q_d \end{pmatrix}, \; d = 1, 2, 3. \tag{2.4}$$

The viscous stress tensor $\underline{\tau}$ is given as (with the viscosity μ and the unit matrix $\underline{I}$)

$$\underline{\tau} = \mu(\vec{\nabla}_x \vec{v} + (\vec{\nabla}_x \vec{v})^T - \frac{2}{3}(\vec{\nabla}_x \cdot \vec{v})\underline{I}), \tag{2.5}$$

and $\vec{q} = (q_1, q_2, q_3)^T$ is the heat flux where λ is the heat conductivity and the flux is proportional to the gradient of the temperature (T) ,

$$\vec{q} = -\lambda \vec{\nabla}_x T \, . \tag{2.6}$$

When the viscosity μ and the heat conductivity λ are zero, the system reduces to the Euler equations. In this work the NSE are used and the source vector $\mathbf{P}$ is set to zero if not mentioned otherwise. This means that gravitation for example is not considered, which is allowed because the effect of the gravitation is negligible in the performed calculations in this work.

2.2 Equation of State

The most well known equation of state is the ideal gas EOS. This is mostly used in CFD calculations for a variety of applications and can be written as follows:

$$p(\rho, T) = \rho R_s T, \tag{2.7}$$

$$e(T) = \frac{R_s T}{(\kappa - 1)}, \tag{2.8}$$

$$c(T) = \sqrt{\kappa R_s T}, \tag{2.9}$$

where p is the pressure, e the specific internal energy and c is the sound speed. The adiabatic coefficient κ and the specific gas constant R_s are parameters varying for each gas. The ideal gas EOS is only used for validation calculations in this work but the focus lies on a realistic EOS, namely the Helmholtz free energy formulation. With this the liquid, vapor and two-phase region of a fluid can be approximated very accurately. This realistic EOS is evaluated by the CoolProp library [9] which is used in this work. CoolProp is a fluid library with over 100 fluids in its database, like dodecane or water as example. In this work however only water applications are investigated, but other fluids are also possible as seen in [26, 28], where the fluid is methane. CoolProp (version 4.2.6) has the 'International Association for the Properties of Water and Steam formulation 1995' (IAPWS-95) [70] implemented. This standard describes the Helmholtz free energy formulation and the needed parameters for water. This formulation is described shortly in this section. The Helmholtz free energy formulation can be written in the reduced form and consists of two parts:

$$\Phi(\tau, \delta) = \Phi^o(\tau, \delta) + \Phi^r(\tau, \delta), \tag{2.10}$$

Φ^o represents the ideal gas part and Φ^r the residual part. τ is the inverse reduced temperature T_r/T and the reduced density ρ/ρ_r is denoted by δ. These parameters are usually taken as the critical states ($T_r = T_c$, $\rho_r = \rho_c$) [9, 64]. As seen in Equation (2.10) this EOS is written as a function depending on the density and temperature of a fluid. The ideal gas part can be written as [65]:

$$\Phi^o = \ln(\delta) + c^{II} + c^I \tau + c_0 \ln(\tau) + \sum_{i=1}^{I_{\text{Pol}}} c_i \tau^{t_i} + \sum_{i=I_{\text{Pol}}+1}^{I_{\text{Pol}}+I_{\text{Exp}}} m_i \ln(1 - \exp(-\gamma_i^o \tau)). \tag{2.11}$$

Here, c^{II}, c^I, c_0, c_i, t_i, m_i and γ_i^o are parameters fitted for each fluid. I_{Pol} is the number of polynomial terms and I_{Exp} is the number of the exponential ones, depending on the thermodynamic state (e.g. liquid) and the fluid. For water in the IAPWS-95 standard the ideal gas part does not contain the polynomial term and is defined as follows:

$$\Phi^o_{\text{water}} = \ln(\delta) + n_1^o + n_2^o \tau + n_3^o \ln(\tau) + \sum_{i=4}^{8} n_i^o \ln(1 - \exp(-\gamma_i^o \tau)), \tag{2.12}$$

where the n_i^o and γ_i^o are shown in Table 2.1.

The residual part contains again a polynomial and exponential part,

$$\Phi^r(\tau, \delta) = \sum_{i=1}^{I_{\text{Pol}}} n_i \tau^{t_i} \delta^{d_i} + \sum_{i=I_{\text{Pol}}+1}^{I_{\text{Pol}}+I_{\text{Exp}}} n_i \tau^{t_i} \delta^{d_i} \exp(-\delta^{p_i}). \tag{2.13}$$

Here, n_i, t_i, d_i and p_i are coefficients which are determined by nonlinear fits of analytically or experimentally gained thermodynamic properties. For water

Table 2.1: Coefficients for Φ^o_{water}

i	n_i^o	γ_i^o	i	n_i^o	γ_i^o
1	−8.32044648201	-	5	0.97315	3.53734222
2	6.6832105268	-	6	1.27950	7.74073708
3	3.00632	-	7	0.96956	9.24437796
4	0.012436	1.28728967	8	0.24873	27.5075105

described by the IAPWS-95 standard, I_{Pol} and I_{Exp} are independent of the thermodynamic state and the residual part can be written as:

$$\Phi^r_{\text{water}} = \sum_{i=1}^{7} n_i \delta^{d_i} \tau^{t_i} + \sum_{i=8}^{51} n_i \delta^{d_i} \tau^{t_i} \exp\left(-\delta^{c_i}\right) \tag{2.14}$$

$$+ \sum_{i=52}^{54} n_i \delta^{d_i} \tau^{t_i} \exp\left(-\alpha_i(\delta-\epsilon_i)^2 - \beta_i(\tau-\gamma_i)^2\right) + \sum_{i=55}^{56} n_i \Delta^{b_i} \delta\psi, \tag{2.15}$$

with

$$\Delta = \theta^2 + B_i \left[(\delta-1)^2\right]^{\alpha_i}, \tag{2.16}$$

$$\theta = (1-\tau) + A_i \left[(\delta-1)^2\right]^{1/(2\beta_i)}, \tag{2.17}$$

$$\psi = \exp\left(-C_i(\delta-1)^2 - D_i(\tau-1)^2\right. . \tag{2.18}$$

The needed coefficients $(\cdot)_i$ are listed in [70]. Using Equation (2.10) all other state variables, like pressure, specific internal energy, speed of sound, etc., can be evaluated by analytic differentiation with respects to density and temperature:

$$\frac{p(\delta,\tau)}{\rho RT} = 1 + \delta\Phi^r_\delta, \tag{2.19}$$

$$\frac{e(\delta,\tau)}{RT} = \tau(\Phi^o_\tau + \Phi^r_\tau), \tag{2.20}$$

$$\frac{c^2(\delta,\tau)}{RT} = 1 + 2\delta\Phi^r_\delta + \delta^2\Phi^r_{\delta\delta} - \frac{(1+\delta\Phi^r_\delta - \delta\tau\Phi^r_{\delta\tau})^2}{\tau^2(\Phi^o_{\tau\tau} + \Phi^r_{\tau\tau})}. \tag{2.21}$$

The needed derivatives can also be found in [70]. The Helmholtz free energy formulation for water is built out of 65 terms. For other fluids this number of terms is different. Methane as example needs 49 terms [58].

In the two-phase region, where vapor and liquid exist at the same time, it is necessary to evaluate the phase equilibrium. For a given temperature the pressure and Gibbs free energy (g) are constant in the two-phase region for a pure fluid. These leads to a system of equations for a given saturation temperature T_s

$$p(T_s,\rho') = p(T_s,\rho''), \tag{2.22}$$

$$g(T_s, \rho') = g(T_s, \rho''). \tag{2.23}$$

To solve this equation system a method proposed in [1] is used by the CoolProp library [9]. The saturated vapor and liquid density, ρ'' and ρ', as well as the saturation pressure p_s can be calculated for a given temperature T_s by this method. CoolProp also provides the quantities (e, p, c, μ, λ) in the two-phase region needed by the NSE.

For the Navier-Stokes equations, the specific internal energy e can be calculated from the conservative variables: $e = E - 0.5\,\vec{v}^2$. Since the Helmholtz free energy formulation uses the density and temperature as input, the correlation between these quantities must be evaluated. An easy way to do this is to iterate the temperature to find the corresponding specific internal energy. The starting point is determined by the given specific internal energy e_g and density ρ_g:

$$f(T) = e_g - e(\rho_g, T) \stackrel{!}{=} 0, \tag{2.24}$$

where $e(\rho_g, T)$ can be evaluated with Equation (2.20). The bisection method [72] is used in this work to find the correct temperature since Newton's method does not converge at the transition between different phases. After the bisection algorithm has finished, the corresponding temperature is known and with ρ_g and T all other parameters can be calculated easily. The same technique is needed to use the density and pressure as given parameters and calculate the corresponding temperature.

This procedure is very robust but numerically very time consuming. In Chapter 3 an idea is explained, which uses this method in a preprocessing step to build a highly accurate quadtree for the EOS, which can later be used during the calculation and postprocessing.

2.3 Discontinuous Galerkin Spectral Element Method

In context of this CFD method the computational domain is split into non-overlapping hexahedral elements, called mesh, where the system of governing equations are solved. For the DGSEM unstructured meshes can be used with curved hexahedral elements. An example of such a domain is seen in Figure

2.1. This picture shows a slice of a 3D physical domain, which represents a throttle. DGSEM is derived for the NSE, a system of hyperbolic-parabolic

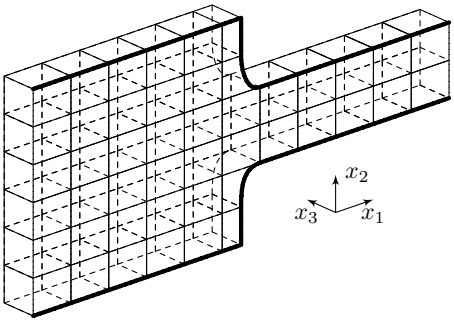

Figure 2.1: Schematic mesh of a channel flow

conservation equations, in one 3D element following [30, 35],

$$\frac{\partial \mathbf{U}}{\partial t} + \underbrace{\vec{\nabla}_x \cdot \vec{\mathbf{F}}_a(\mathbf{U}) - \vec{\nabla}_x \cdot \vec{\mathbf{F}}_d(\mathbf{U}, \vec{\nabla}_x \mathbf{U})}_{\vec{\nabla}_x \cdot \vec{\mathbf{F}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U})} = 0. \tag{2.25}$$

Equation (2.25) is written in physical space but for efficiency every element from the mesh, which discretizes the calculation domain, is mapped to a reference element $\mathcal{E} \in [-1,1]^3$ using the reference coordinates $\vec{\xi} = (\xi_1, \xi_2, \xi_3)^T$. An illustration of this is shown in Figure 2.2. It shows a curved element of the physical domain on the left and the reference element on the right side. The black and gray dots are symbolizing the solution points or degrees of freedom (DOF). A polynomial mapping function $\vec{m}(\vec{\xi})$ is used to map from the reference element to the one in physical space. But to express Equation (2.25) in reference space, also the gradients have to be considered. Following [35] this is achieved with

$$\vec{\nabla}_x \cdot \vec{\mathbf{F}} = \frac{1}{J} \vec{\nabla}_\xi \cdot \vec{\boldsymbol{\mathcal{F}}}. \tag{2.26}$$

The above equation introduces the transformed fluxes $\vec{\boldsymbol{\mathcal{F}}}$:

$$\boldsymbol{\mathcal{F}}_d = (J \vec{a}_d) \cdot \vec{\mathbf{F}}, \quad d = 1, 2, 3, \tag{2.27}$$

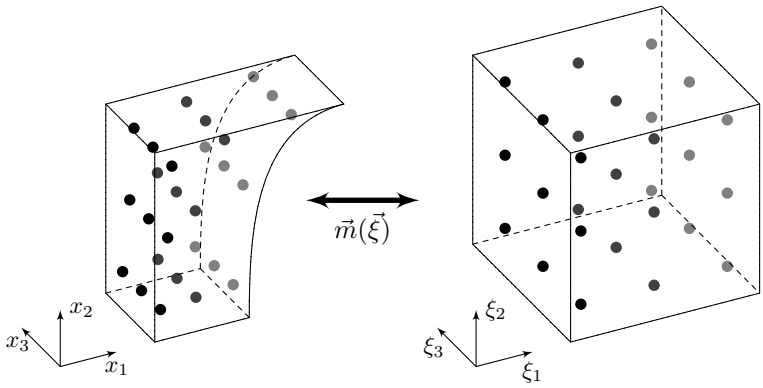

Figure 2.2: Transformation from physical to reference space

with the covariant basis functions

$$\vec{a}_d = \frac{\partial \vec{m}}{\partial \xi_d}, \quad d = 1, 2, 3. \tag{2.28}$$

These basis functions are defining the Jacobian matrix $\underline{J} = (\vec{a}_1, \vec{a}_2, \vec{a}_3)^T$ and the determinant $J = \det(\underline{J})$ of that matrix which is called *Jacobian*. Also needed by Equation (2.26) is the gradient operator in reference space:

$$\vec{\nabla}_\xi = \left(\frac{\partial}{\partial \xi_1}, \frac{\partial}{\partial \xi_2}, \frac{\partial}{\partial \xi_3} \right)^T. \tag{2.29}$$

Now transforming Equation (2.25) into reference space yields

$$\mathbf{U}_t + \frac{1}{J} \vec{\nabla}_\xi \cdot \vec{\boldsymbol{\mathcal{F}}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U}) = \mathbf{U}_t + \frac{1}{J} \vec{\nabla}_\xi \cdot (\vec{\boldsymbol{\mathcal{G}}}(\mathbf{U}) - \vec{\boldsymbol{\mathcal{H}}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U})) = \mathbf{0}. \tag{2.30}$$

The contributions of inviscid and viscous terms to the flux are the terms $\vec{\boldsymbol{\mathcal{G}}}(\mathbf{U})$ and $\vec{\boldsymbol{\mathcal{H}}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U})$, respectively. In each element the solution vector is approximated by a tensor product of 1D Lagrange polynomials ℓ^N of degree N,

$$\mathbf{U}(\vec{\xi}, t) \approx \sum_{i,j,k=0}^{N} \hat{\mathbf{U}}_{ijk}(t) \psi_{ijk}^N(\vec{\xi}), \qquad \psi_{ijk}^N(\vec{\xi}) = \ell_i^N(\xi_1)\ell_j^N(\xi_2)\ell_k^N(\xi_3). \tag{2.31}$$

This is a nodal interpolation ansatz and $\hat{\mathbf{U}}_{ijk}(t)$ are time-dependent nodal DOF representing the solution at the solution points . In Figure 2.2 the locations of these are visualized by dots. As basis functions the Lagrange polynomials are used and defined by a set of arbitrarily chosen nodal points $\{\xi_i\}_{i=0}^N \in [-1, 1]$ in each spatial direction. The Lagrange polynomials are given by:

$$\ell_j^N(\xi) = \prod_{i=0, j\neq i}^{N} \frac{\xi - \xi_i}{\xi_j - \xi_i}, \quad j = 0, \ldots, N, \tag{2.32}$$

with the Lagrange property

$$\ell_j^N(\xi_i) = \delta_{ij}, \quad i, j = 0, \ldots, N. \tag{2.33}$$

δ_{ij} is the Kronecker delta. In this work, following [30] and [35], the $N + 1$ Gauss-Legendre points are used as the interpolation node set $\{\xi_i\}_{i=0}^N$ for a polynomial degree of N. The flux $\vec{\boldsymbol{\mathcal{F}}}$ is also approximated by a polynomial

$$\boldsymbol{\mathcal{F}}_d(\vec{\xi}) \approx \sum_{i,j,k=0}^{N} \hat{\boldsymbol{\mathcal{F}}}_{d_{ijk}} \psi_{ijk}^N(\vec{\xi}), \quad d = 1, 2, 3 \tag{2.34}$$

$$\hat{\boldsymbol{\mathcal{F}}}_{d_{ijk}} = \boldsymbol{\mathcal{G}}_d(\mathbf{U}) \mid_{\vec{\xi}_{ijk}} - \boldsymbol{\mathcal{H}}_d(\mathbf{U}, \vec{\nabla}_x \mathbf{U}) \mid_{\vec{\xi}_{ijk}}. \tag{2.35}$$

Multiplying Equation (2.30) by the *Jacobian* J it leads to

$$J\mathbf{U}_t + \vec{\nabla}_\xi \cdot \vec{\boldsymbol{\mathcal{F}}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U}) = \mathbf{0}. \tag{2.36}$$

To achieve the variational formulation the projection of the approximated solution (Equation (2.36)) to the exact solution (which equals $\mathbf{0}$) is build

$$\langle J\mathbf{U}_t + \vec{\nabla}_\xi \cdot \vec{\boldsymbol{\mathcal{F}}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U}), \boldsymbol{\phi} \rangle_{\mathbb{L}_2(\mathcal{E})} = \langle \mathbf{0}, \boldsymbol{\phi} \rangle_{\mathbb{L}_2(\mathcal{E})}, \tag{2.37}$$

where the $\mathbb{L}_2$ inner product $\langle f, g \rangle_{\mathbb{L}_2(Q)}$ in an area Q is defined as

$$\langle f, g \rangle_{\mathbb{L}_2(Q)} = \int_Q f(x) g(x) dx. \tag{2.38}$$

The test functions $\boldsymbol{\phi}(\vec{\xi})$ are from the same space as the polynomial interpolation basis function ψ. Rewriting Equation (2.37) the variation formulation can also be written as follows

$$\int_{\mathcal{E}} \left(J \mathbf{U}_t + \vec{\nabla}_\xi \cdot \vec{\boldsymbol{\mathcal{F}}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U}) \right) \boldsymbol{\phi}(\vec{\xi}) \, d\vec{\xi} = \mathbf{0}. \tag{2.39}$$

By using integration by parts for the flux term, the weak formulation of the variational formulation is achieved

$$\int_{\mathcal{E}} J \mathbf{U}_t \boldsymbol{\phi} \, d\vec{\xi} + \oint_{\partial \mathcal{E}} \underbrace{(\boldsymbol{\mathcal{G}}_n^* - \boldsymbol{\mathcal{H}}_n^*)}_{\boldsymbol{\mathcal{F}}_n^*} \boldsymbol{\phi} \, d\vec{\xi}_s - \int_{\mathcal{E}} \vec{\boldsymbol{\mathcal{F}}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U}) \cdot \vec{\nabla}_\xi \boldsymbol{\phi} \, d\vec{\xi} = \mathbf{0}. \tag{2.40}$$

The solution $\mathbf{U}$ is allowed to be discontinuous over the element interface $\partial \mathcal{E}$. Therefore a Riemann solver (see Section 2.5 for more details) is used to approximate the inviscid numerical flux function normal to the surfaces $\boldsymbol{\mathcal{G}}_n^* = \boldsymbol{\mathcal{G}}_n^*(\mathbf{U}^+, \mathbf{U}^-)$. The superscripts $\pm$ stand for the solution at the element surface of the neighbor element and the local one, respectively. To derive the viscous flux terms $\boldsymbol{\mathcal{H}}_n^*$, the governing equations are rewritten with an additional variable $\vec{\mathbf{S}}$ as an approximation to the solution gradients as a corresponding system of first order equations:

$$\begin{aligned} \mathbf{U}_t + \vec{\nabla}_x \cdot \vec{\mathbf{F}}(\mathbf{U}, \vec{\mathbf{S}}) &= \mathbf{0}, \\ \vec{\mathbf{S}} - \vec{\nabla}_x \mathbf{U} &= \mathbf{0}. \end{aligned} \tag{2.41}$$

Following the steps from above, the equation system leads to

$$\int_{\mathcal{E}} J \mathbf{U}_t \boldsymbol{\phi} \, d\vec{\xi} + \oint_{\partial \mathcal{E}} (\boldsymbol{\mathcal{G}}_n^* - \boldsymbol{\mathcal{H}}_n^*) \, \boldsymbol{\phi} \, \vec{\xi}_s - \int_{\mathcal{E}} \vec{\boldsymbol{\mathcal{F}}}(\mathbf{U}, \vec{\mathbf{S}}) \cdot \vec{\nabla}_\xi \boldsymbol{\phi} \, d\vec{\xi} = \mathbf{0}, \tag{2.42}$$

$$\int_{\mathcal{E}} J \mathbf{S}_d \boldsymbol{\phi} \, d\vec{\xi} - \oint_{\partial \mathcal{E}} \boldsymbol{\mathfrak{U}}_n^{d^*} \boldsymbol{\phi} \, d\vec{\xi}_s + \int_{\mathcal{E}} \vec{\boldsymbol{\mathcal{U}}}^d \cdot \vec{\nabla}_\xi \boldsymbol{\phi} \, d\vec{\xi} = \mathbf{0}. \tag{2.43}$$

The Equation (2.43) is given for each spatial direction d, with $\vec{\boldsymbol{\mathcal{U}}}^d = (J\vec{a}_d)\mathbf{U}$. The numercial flux for the additional equation is $\mathfrak{U}_n^{d^*}$ and

$$\boldsymbol{\mathcal{H}}_n^* = \boldsymbol{\mathcal{H}}_n^*(\mathbf{U}^+, \mathbf{U}^-, \vec{\mathbf{S}}^+, \vec{\mathbf{S}}^-)$$

stands for the numerical flux function for the viscous terms in reference space in normal direction to the surface. Since at the surface the solution is double valued an approximation of these fluxes is needed. As introduced in [5], these approximations are given by

$$\boldsymbol{\mathcal{H}}_n^* = \left(\alpha_{\text{visc}}\, \boldsymbol{\mathcal{H}}_n(\mathbf{U}^+, \vec{\mathbf{S}}^+) + (1-\alpha_{\text{visc}})\, \boldsymbol{\mathcal{H}}_n(\mathbf{U}^-, \vec{\mathbf{S}}^-)\right), \tag{2.44}$$

$$\mathfrak{U}_n^{d^*} = \left(\alpha_{\text{visc}}\, \mathbf{U}^+ + (1-\alpha_{\text{visc}})\, \mathbf{U}^-\right) n_d, \tag{2.45}$$

where $\vec{n}$ is the surface normal vector pointing outwards. α_{visc} is chosen to be $\frac{1}{2}$, this method is named BR1 (first method of Bassi and Rebay [5]). With this method the gradients $\vec{\nabla}_x \mathbf{U} = (\vec{\nabla}_x \rho, \vec{\nabla}_x \rho\vec{v}, \vec{\nabla}_x \rho E)$ can be approximated. But for Equation (2.4) the velocity gradients are needed as well as the temperature gradient $\vec{\nabla}_x T$. For the ideal gas EOS (see Equation (2.7)) $\vec{\nabla}_x T$ can be calculated using the gradients of the conservative variables as follows:

$$T = e\,\frac{\kappa-1}{R_s} \quad \rightarrow \quad \vec{\nabla}_x T = (\vec{\nabla}_x e)\,\frac{\kappa-1}{R_s}, \tag{2.46}$$

$$\rho E = \rho e + \frac{1}{2}\rho|\vec{v}|^2 \quad \rightarrow \quad \vec{\nabla}_x \rho E = \vec{\nabla}_x \rho e + \frac{1}{2}\sum_{i=1}^{3} \vec{\nabla}_x \rho v_i^2. \tag{2.47}$$

Applying the product rule to Equation (2.47) and rearranging it leads to the desired derivatives $\vec{\nabla}_x e$:

$$\vec{\nabla}_x e = \frac{1}{\rho}\left((\vec{\nabla}_x \rho)e + \frac{1}{2}\sum_{i=1}^{3}\left((\vec{\nabla}_x \rho v_i)v_i + (\vec{\nabla}_x v_i)\rho v_i\right) - (\vec{\nabla}_x \rho E)\right) \tag{2.48}$$

with

$$\vec{\nabla}_x v_i = \frac{1}{\rho}(\vec{\nabla}_x \rho v_i - (\vec{\nabla}_x \rho)v_i). \tag{2.49}$$

Inserting Equation (2.48) and (2.49) into the right part of Equation (2.46) the temperature gradient can be expressed by the gradients of the conservative

variables. Since in this work the Helmholtz free energy formulation is used as EOS the approach with the gradients of the conservative variables is not applicable, because $T(\rho, e)$ is not given as an analytical equation (see Section 2.2). In a first step the temperature is calculated with the use of the conservative variables in the volume and at the interface of an element and then the temperature gradient is approximated with Equation (2.43). But Equation (2.49) is necessary also for the approach with the Helmholtz free energy formulation since for the diffusion fluxes only the velocity gradient $\vec{\nabla}_x v_i$ for all spatial directions and the temperature gradient $\vec{\nabla}_x T$ are needed (see Section 2.1).

2.3.1 Semi-Discrete Formulation

To derive the semi-discrete formulation of Equations (2.42) and (2.43) an approximation of the integrals is introduced in this subsection by using Gauss-Legendre quadrature. The efficiency of the DGSEM is partly achieved by collocating the integration nodes and the interpolation nodes. This means the Gauss-Legendre nodes are used for integration nodes as well as interpolation nodes in this work. For a function $g(\xi)$ the Gauss-Legendre quadrature reads:

$$\int_{-1}^{1} g(\xi) d\xi = \sum_{k=0}^{N} g(\xi_k)\omega_k, \tag{2.50}$$

where ω_k are Gauss-Legendre weights which are defined analytically by

$$\int_{-1}^{1} \ell_k^N(\xi) d\xi = \omega_k, \tag{2.51}$$

where ℓ_k^N is given by the corresponding Lagrange polynomial. $g(\xi)$ is represented by a Lagrange polynomial according to Equation (2.31) and the approximation of $g(\xi)$ is done by projection which leads to the following equation:

$$\int_{-1}^{1} g(\xi) d\xi \approx \sum_{k=0}^{N} g(\xi_k)\omega_k = \sum_{k=0}^{N}\sum_{j=0}^{N} \hat{g}_j \underbrace{\ell_j^N(\xi_k)}_{\delta_{jk}} \omega_k. \tag{2.52}$$

Since the same node sets are used for interpolation and integration the Lagrange property is used to reduce Equation (2.52) to:

$$\int_{-1}^{1} g(\xi)d\xi \approx \sum_{k=0}^{N} \hat{g}_k \omega_k. \tag{2.53}$$

This integration is exact for all polynomials of degree $2N+1$. Applying the above described process to the time derivative integral part of Equation (2.42) and setting the test function equal to one basis function $\phi = \psi_{ijk}^N$, this leads for one DOF at the i,j,k-position to

$$\begin{aligned}
&\int_{\mathcal{E}} J\mathbf{U}_t \phi \, d\vec{\xi} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} J(\vec{\xi}) \left(\frac{\partial}{\partial t} \sum_{d,e,f=0}^{N} \hat{\mathbf{U}}_{def}(t) \psi_{def}^N(\vec{\xi}) \right) \psi_{ijk}^N(\vec{\xi}) d\xi_1 d\xi_2 d\xi_3 \\
&\approx \sum_{\mathbb{i},\mathbb{j},\mathbb{k}=0}^{N} J(\vec{\xi}_{\mathbb{i}\mathbb{j}\mathbb{k}}) \left(\frac{\partial}{\partial t} \sum_{d,e,f=0}^{N} \hat{\mathbf{U}}_{def}(t) \underbrace{\ell_d^N(\xi_{1_\mathbb{i}})}_{=\delta_{d\mathbb{i}}} \underbrace{\ell_e^N(\xi_{2_\mathbb{j}})}_{=\delta_{e\mathbb{j}}} \underbrace{\ell_f^N(\xi_{3_\mathbb{k}})}_{=\delta_{f\mathbb{k}}} \right) \psi_{ijk}^N(\vec{\xi}_{\mathbb{i}\mathbb{j}\mathbb{k}}) \omega_\mathbb{i} \omega_\mathbb{j} \omega_\mathbb{k} \\
&= \sum_{\mathbb{i},\mathbb{j},\mathbb{k}=0}^{N} J(\vec{\xi}_{\mathbb{i}\mathbb{j}\mathbb{k}}) \frac{\partial}{\partial t} \hat{\mathbf{U}}_{\mathbb{i}\mathbb{j}\mathbb{k}}(t) \underbrace{\ell_i^N(\xi_{1_\mathbb{i}})}_{=\delta_{i\mathbb{i}}} \underbrace{\ell_j^N(\xi_{2_\mathbb{j}})}_{=\delta_{j\mathbb{j}}} \underbrace{\ell_k^N(\xi_{3_\mathbb{k}})}_{=\delta_{k\mathbb{k}}} \omega_\mathbb{i} \omega_\mathbb{j} \omega_\mathbb{k} \\
&= J(\vec{\xi}_{ijk}) \omega_i \omega_j \omega_k \frac{\partial}{\partial t} \hat{\mathbf{U}}_{ijk} \, .
\end{aligned} \tag{2.54}$$

For the surface integral, applying the same steps as for the time derivative integral, the outcome for the i,j,k-DOF is

$$\begin{aligned}
\oint_{\partial\mathcal{E}} \boldsymbol{\mathcal{F}}_n^* \phi \, d\vec{\xi}_s \approx &\left([\boldsymbol{\mathcal{F}}^*_{\xi_1^+} \, \hat{s}_{\xi_1^+}]_{jk} \hat{\ell}_i(+1) + [\boldsymbol{\mathcal{F}}^*_{\xi_1^-} \, \hat{s}_{\xi_1^-}]_{jk} \hat{\ell}_i(-1) \right) \omega_j \omega_k \\
&+ \left([\boldsymbol{\mathcal{F}}^*_{\xi_2^+} \, \hat{s}_{\xi_2^+}]_{ik} \hat{\ell}_j(+1) + [\boldsymbol{\mathcal{F}}^*_{\xi_2^-} \, \hat{s}_{\xi_2^-}]_{ik} \hat{\ell}_j(-1) \right) \omega_i \omega_k \\
&+ \left([\boldsymbol{\mathcal{F}}^*_{\xi_3^+} \, \hat{s}_{\xi_3^+}]_{ij} \hat{\ell}_k(+1) + [\boldsymbol{\mathcal{F}}^*_{\xi_3^-} \, \hat{s}_{\xi_3^-}]_{ij} \hat{\ell}_k(-1) \right) \omega_i \omega_j \, .
\end{aligned} \tag{2.55}$$

$\vec{\xi}^{\pm}$ stand for the normal directions, $+$ pointing inwards and $-$ outwards. $\hat{s}$ denotes the surface element, which is the correlation between the physical and the reference surface. In Figure 2.3 the DOF distribution for a polynomial

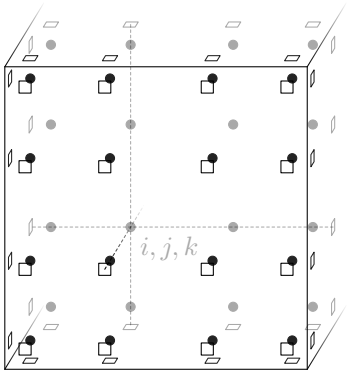

Figure 2.3: DOF distribution for a DG element for $N = 3$

degree of $N = 3$ is shown. The black and gray dots represent the DOF. The ξ_3 is only drawn for half of the element for a better understanding. The position of one i, j, k-position is illustrated as well as the connection to the surface. The squares denote the position of the interface fluxes $\boldsymbol{\mathcal{F}}_n^*$. Using the same steps as above for the discretization of the volume integral, the result is (again for the i, j, k-DOF):

$$\begin{aligned}\int_{\mathcal{E}} \vec{\boldsymbol{\mathcal{F}}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U}) \cdot \vec{\nabla}_\xi \, \phi \, d\vec{\xi} \approx & \omega_j \omega_k \sum_{\mathbb{i}=0}^{N} D_{i\mathbb{i}} \hat{\boldsymbol{\mathcal{F}}}_{1_{ijk}} \omega_{\mathbb{i}} + \omega_i \omega_k \sum_{\mathbb{j}=0}^{N} D_{j\mathbb{j}} \hat{\boldsymbol{\mathcal{F}}}_{2_{ijk}} \omega_{\mathbb{j}} \\ & + \omega_i \omega_j \sum_{\mathbb{k}=0}^{N} D_{k\mathbb{k}} \hat{\boldsymbol{\mathcal{F}}}_{3_{ijk}} \omega_{\mathbb{k}} ,\end{aligned} \tag{2.56}$$

with the differentiation matrix

$$D_{ij} = \left. \frac{d\ell_j(\xi)}{d\xi} \right|_{\xi=\xi_i}, \quad i, j = 0, ..., N. \tag{2.57}$$

The complete derivation of the surface and volume integral can be found in [29]. Inserting Equations (2.54), (2.55) and (2.56) into Equation (2.42) the semi-discrete formulation for one DOF is achieved

$$\begin{aligned}
-J_{ijk}\left(\hat{\mathbf{U}}_{ijk}\right)_t =& \sum_{\mathbb{i}=0}^{N} \hat{D}_{i\mathbb{i}} \hat{\boldsymbol{\mathcal{F}}}_{1_{\mathbb{i}jk}} + [\boldsymbol{\mathcal{F}}^*_{\xi_1^+}\ \hat{s}_{\xi_1^+}]_{jk}\hat{\ell}_i(+1) + [\boldsymbol{\mathcal{F}}^*_{\xi_1^-}\ \hat{s}_{\xi_1^-}]_{jk}\hat{\ell}_i(-1) \\
&+ \sum_{\mathbb{j}=0}^{N} \hat{D}_{j\mathbb{j}} \hat{\boldsymbol{\mathcal{F}}}_{2_{i\mathbb{j}k}} + [\boldsymbol{\mathcal{F}}^*_{\xi_2^+}\ \hat{s}_{\xi_2^+}]_{ik}\hat{\ell}_j(+1) + [\boldsymbol{\mathcal{F}}^*_{\xi_2^-}\ \hat{s}_{\xi_2^-}]_{ik}\hat{\ell}_j(-1) \\
&+ \sum_{\mathbb{k}=0}^{N} \hat{D}_{k\mathbb{k}} \hat{\boldsymbol{\mathcal{F}}}_{3_{ij\mathbb{k}}} + [\boldsymbol{\mathcal{F}}^*_{\xi_3^+}\ \hat{s}_{\xi_3^+}]_{ij}\hat{\ell}_k(+1) + [\boldsymbol{\mathcal{F}}^*_{\xi_3^-}\ \hat{s}_{\xi_3^-}]_{ij}\hat{\ell}_k(-1)\ ,
\end{aligned} \tag{2.58}$$

with the weighted differentiation matrix and the weighted basis functions

$$\begin{aligned}
\hat{D}_{ij} =& -D_{ji}\frac{\omega_j}{\omega_i}, \qquad i,j = 0,...,N, \\
\hat{\ell}_i =& \frac{\ell_i}{\omega_i}, \qquad\qquad\quad i = 0,...,N.
\end{aligned} \tag{2.59}$$

For Equation (2.43) also a semi-discrete formulation can be derived, following the same steps which are needed to derive Equation (2.58), yielding for each spatial direction d

$$\begin{aligned}
J_{ijk}\hat{\mathbf{S}}_{d_{ijk}} =& \left(\sum_{\mathbb{i}=0}^{N} \hat{D}_{i\mathbb{i}}\boldsymbol{\mathcal{U}}^{d,1}_{\mathbb{i}jk} + \hat{D}_{j\mathbb{i}}\boldsymbol{\mathcal{U}}^{d,2}_{i\mathbb{i}k} + \hat{D}_{k\mathbb{i}}\boldsymbol{\mathcal{U}}^{d,3}_{ij\mathbb{i}}\right) \\
&+[\mathfrak{U}^{d^*}_{\xi_1^+}\ \hat{s}_{\xi_1^+}]_{jk}\hat{\ell}_i(+1) + [\mathfrak{U}^{d^*}_{\xi_1^-}\ \hat{s}_{\xi_1^-}]_{jk}\hat{\ell}_i(-1) \\
&+[\mathfrak{U}^{d^*}_{\xi_2^+}\ \hat{s}_{\xi_2^+}]_{ik}\hat{\ell}_j(+1) + [\mathfrak{U}^{d^*}_{\xi_2^-}\ \hat{s}_{\xi_2^-}]_{ik}\hat{\ell}_j(-1) \\
&+[\mathfrak{U}^{d^*}_{\xi_3^+}\ \hat{s}_{\xi_3^+}]_{ij}\hat{\ell}_k(+1) + [\mathfrak{U}^{d^*}_{\xi_3^-}\ \hat{s}_{\xi_3^-}]_{ij}\hat{\ell}_k(-1)\ .
\end{aligned} \tag{2.60}$$

2.4 Finite Volume Subcell Approach

A shock capturing technique is needed because strong gradients during phase change can occur and also ordinary shocks have to be resolved without oscillation. A high-order method like the above described DGSEM cannot resolve

those. The strong gradients and shocks would lead to oscillations in the polynomial. A shock capturing technique can be split into two parts: at first the elements containing oscillating polynomials have to be identified and in a second step the elements have to be treated in a way to reduce the oscillations. In this work the first part is done by an indicator proposed by Persson and Peraire [46]. This indicator calculates for each element a scalar value depending on the oscillation of the polynomial representation of the solution. With this indicator-value and a chosen indicator-limit, elements can be found which need treatment to reduce the oscillations. In this work the treatment is done by reducing the polynomial order of the solution but at the same time increasing the spatial discretization. The DG element is split into $(N+1)^3$ equidistant sub-cells. In these cells a second-order finite volume method is used to solve the NSE. The FV method is well-known for handling strong gradients and shocks. In every Runge-Kutta time step (see Section 2.7) the indicator-value is calculated. Then for each element it is known whether it can use the DGSEM or the FV sub-cell approach to solve the NSE for this time step. This can vary for each time-step for an element. The building blocks needed for this 'switching' between these two methods are explained in this section as well as the FV method.

The number of DOF is kept the same for both methods as seen in Figure 2.4. The figure shows the DG distribution on the left and the FV on the right side. The black dots symbolize the solution points for both methods. With the polynomial degree $N = 3$ as an example there are 4 points in each direction making it $4^3 = 64$ in total for both methods. In the figure only half of the ξ_3 direction is drawn to avoid visual clutter. The DG solution points are Gauss-Legendre distributed, as stated earlier, and the FV ones are placed equidistantly. The dashed lines in the right picture outline the FV cells. The open squares represent the locations where the flux $\boldsymbol{\mathcal{F}}_n^*$ needs to be evaluated via a Riemann solver (see Section 2.5). This flux is calculated over two conjunct elements. The black filled squares denote the inner element flux needed by the FV method.

For switching between the DG and FV sub-cell method the solution has to be converted between the different point distributions. This conversion can be reduced to a matrix-vector multiplication. In 3D this multiplication can be applied dimension by dimension in a tensor product fashion. In this case it is adequate to derive the conversion in one dimensional space. Since the FV sub-

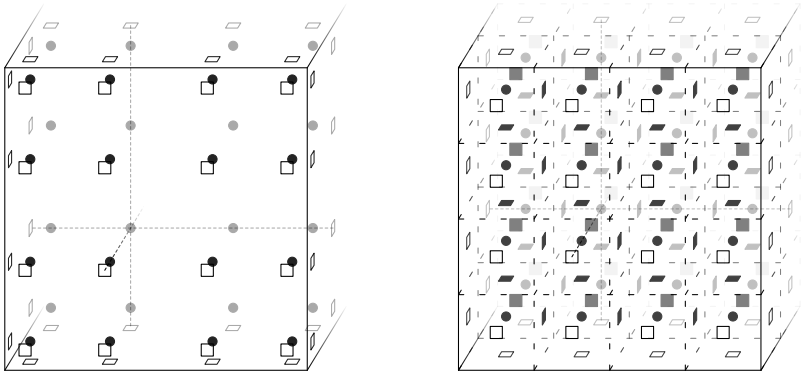

Figure 2.4: DOF distribution for a DG element (left) and FV cells (right) for $N = 3$

cells are equidistantly spread in the DG element, the FV sub-cells are defined by:

$$\text{sub-cell}_l = [-1 + l\varpi, -1 + (l+1)\varpi] \quad l = 0, \ldots, N. \tag{2.61}$$

The length of the sub-cell in one spatial direction is $\varpi = 2/(N+1)$. The important part for changing the solution distribution is to keep the integral mean value constant. To achieve this the polynomial DG solution has to be integrated for each FV sub-cell

$$\int_{\mathcal{E}} \mathbf{U} d\xi = \int_{-1}^{1} \mathbf{U} d\xi = \sum_{l=0}^{N} \int_{-1+l\varpi}^{(-1+(l+1)\varpi)} \mathbf{U} d\xi. \tag{2.62}$$

Replacing the solution by interpolation with Lagrange basis functions (see Section 2.3), the last term of Equation (2.62) leads to

$$\sum_{l=0}^{N} \int_{-1+l\varpi}^{(-1+(l+1)\varpi)} \sum_{i=0}^{N} \hat{\mathbf{U}}_i \psi_i(\xi) d\xi. \tag{2.63}$$

To integrate over the sub-cells, integration points in each sub-cell are introduced, where ξ_r are again the Gauss-Legendre nodes

$$\left\{\xi_r^l\right\}_{r=0}^N = \left\{-1 + l\varpi + (\xi_r + 1)\frac{\varpi}{2}\right\}_{r=0}^N, \quad l = 0, \ldots, N, \tag{2.64}$$

with $\varpi/2$ as scaling factor between the length of the reference interval and the subintervals. This leads to a total of $(N+1)^2$ integration nodes for the DG element in 1D. Using these nodes for numerical integration of Equation (2.63) leads to

$$\int_{\mathcal{E}} \mathbf{U} d\xi \approx \frac{\varpi}{2} \sum_{l=0}^{N} \sum_{\alpha=0}^{N} \sum_{i=0}^{N} \hat{\mathbf{U}}_i \ell_i^N(\xi_\alpha^l)\omega_\alpha. \tag{2.65}$$

The Lagrange property cannot be achieved here, because the interpolation and integration points are not the same. For the FV sub-cell distributed solution the integral mean value can be calculated as follows (using the midpoint integration rule and the mean value for each FV sub-cell $\bar{\mathbf{U}}_l$):

$$\int_{\mathcal{E}} \mathbf{U} d\xi \approx \sum_{l=0}^{N} \bar{\mathbf{U}}_l \varpi_l. \tag{2.66}$$

To keep the integral mean value constant, Equation (2.65) and (2.66) have to be equal

$$\sum_{l=0}^{N} \bar{\mathbf{U}}_l \varpi \stackrel{!}{=} \frac{\varpi}{2} \sum_{l=0}^{N} \sum_{\alpha=0}^{N} \sum_{i=0}^{N} \hat{\mathbf{U}}_i \ell_i^N(\xi_\alpha^l)\omega_\alpha. \tag{2.67}$$

Each FV sub-cell mean value can now be calculated from the DG solution with

$$\bar{\mathbf{U}}_l = \frac{1}{2} \sum_{\alpha=0}^{N} \sum_{i=0}^{N} \hat{\mathbf{U}}_i \ell_i^N(\xi_\alpha^l)\omega_\alpha. \tag{2.68}$$

To transform the FV sub-cell solution back to the DG solution the transformation matrix in 1D

$$\underline{V}_{\text{FV}} = \left[\frac{1}{2} \sum_{\alpha=0}^{N} \ell_i^N(\xi_\alpha^l)\omega_\alpha\right]_{i,l=0}^{N}, \tag{2.69}$$

is inverted $\underline{V}_{\mathrm{FV}}^{-1}$ and multiplied with Equation (2.69)

$$\underline{V}_{\mathrm{FV}}^{-1}\{\bar{\mathbf{U}}\}_{l=0}^{N} = \{\hat{\mathbf{U}}\}_{i=0}^{N}. \tag{2.70}$$

As mentioned earlier for the transformation in three-dimensional space this procedure can be applied dimension by dimension in a tensor product way. This holds for both conversion directions DG $\rightarrow$ FV and FV $\rightarrow$ DG. Not only the solution needs to be converted but also the fluxes, because if an element solved via DG method and an element solved via the FV sub-cell method are conjunct to each other the flux point representation is different for both methods (see Figure 2.4). This flux conversion is done by a transformation matrix not for the volume but for the surface of an element which can be derived using the same steps described above.

The derivation of the FV approach starts with the Navier-Stokes equations in reference space and integration over the sub-cell element $\mathcal{C}$ at the l, m, n position

$$\int_{\mathcal{C}_{lmn}} J\mathbf{U}_t \,\mathrm{d}\xi + \int_{\mathcal{C}_{lmn}} \nabla_\xi \cdot \vec{\boldsymbol{\mathcal{F}}}(\mathbf{U}, \vec{\nabla}_x \mathbf{U}_p) \,\mathrm{d}\xi = \mathbf{0} \qquad l, m, n = 0, \ldots, N. \tag{2.71}$$

All gradients used in the FV method are constructed and limited in primitive variables $\mathbf{U}_p = (\rho, \vec{v}, T)^T$ where T is the temperature. Applying the Gauss-theorem on the flux term the FV method is written as

$$\int_{\mathcal{C}_{lmn}} J\mathbf{U}_t \,\mathrm{d}\xi + \oint_{\partial\mathcal{C}_{lmn}} \underbrace{\left(\boldsymbol{\mathcal{G}}_n^*(\mathbf{U}) - \boldsymbol{\mathcal{H}}_n^*(\mathbf{U}, \vec{\nabla}_x \mathbf{U}_p)\right)}_{\boldsymbol{\mathcal{F}}_n^*} \mathrm{d}s = \mathbf{0}. \tag{2.72}$$

$\boldsymbol{\mathcal{F}}_n^*$ is the numerical flux for each FV cell interface multiplied with the normal direction, which can be split into advection $\boldsymbol{\mathcal{G}}_n^*$ and diffusion $\boldsymbol{\mathcal{H}}_n^*$ part. The FV approach can be written in a discrete way as

$$\begin{aligned} -J_{lmn}(\bar{\mathbf{U}}_{lmn})_t = s_{FV} &\left[\boldsymbol{\mathcal{F}}^{*,\xi_1}_{l-\frac{1}{2},m,n} + \boldsymbol{\mathcal{F}}^{*,\xi_1}_{l+\frac{1}{2},m,n}\right] \\ + s_{FV} &\left[\boldsymbol{\mathcal{F}}^{*,\xi_2}_{l,m-\frac{1}{2},n} + \boldsymbol{\mathcal{F}}^{*,\xi_2}_{l,m+\frac{1}{2},n}\right] \\ + s_{FV} &\left[\boldsymbol{\mathcal{F}}^{*,\xi_3}_{l,m,n-\frac{1}{2}} + \boldsymbol{\mathcal{F}}^{*,\xi_3}_{l,m,n+\frac{1}{2}}\right], \end{aligned} \tag{2.73}$$

where $s_{FV} = \left(\frac{2}{N+1}\right)^2$ stands for the surface of one FV-subcell. The subscript n for the $\boldsymbol{\mathcal{F}}_n^*$ was omitted here for the sake of simplicity. If Equation (2.73) and Eq. (2.58) are compared it shows that these two can be exchanged for calculating the time derivative of the solution vector $\mathbf{U}_t$.

As mentioned earlier, a second-order reconstruction is implemented in this work, which is illustrated in Figure 2.5. The figure shows the nodal points of

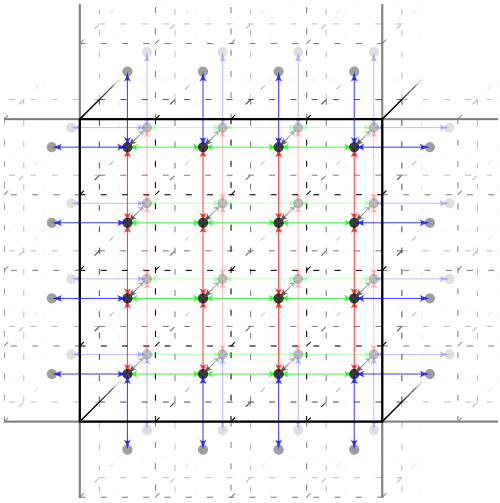

Figure 2.5: Inner-cell and over-interface reconstruction

the FV sub-cells and all their connections for the reconstruction. The arrows are representing the necessary reconstructions. The green and red arrows show the inner cell reconstruction. The blue arrows symbolize the across element reconstruction. The values needed from the neighboring elements have to be distributed equidistantly, this means that an DG element, which is conjunct to a FV sub-cell element, has to convert its solution to the FV distribution. The gradients used for the calculation of the values $\bar{\mathbf{U}}$ on the FV sub-cell interface for the advection flux $\boldsymbol{\mathcal{G}}_n^*(\mathbf{U})$ are limited with the MinMod-Limiter [50], which gives this approach the total variation diminishing property. For

the viscous fluxes $\boldsymbol{\mathcal{H}}_n^*(\mathbf{U}, \overline{\vec{\nabla}_x \mathbf{U}_p})$, in the FV method the mean value of two neighboring gradients are calculated, $\overline{\vec{\nabla}_x \mathbf{U}_p} = \frac{\vec{\nabla}_x \mathbf{U}_p^+ + \vec{\nabla}_x \mathbf{U}_p^-}{2}$, in each direction. The superscripts $+$ and $-$ denote the left and right side of a surface. A more detailed description of this shock capturing method can be found in [62].

2.5 Riemann Solvers

As seen in Equation (2.40) and Eq. (2.72), $\boldsymbol{\mathcal{G}}_n^*$ needs to be evaluated. This is not trivial since the solution on the interface is double valued and therefore also the normal flux. $\mathbf{U}^+$ and $\mathbf{U}^-$ are the two contributions from the two conjunct surfaces, which imposes that the normal component of the flux on the surface needs to be approximated because it is not uniquely defined. This approximation is commonly done by defining the unique flux as a solution of a Riemann problem. A Riemann problem is an initial value problem with piecewise constant states, see Equation (2.78). This approach originates in the FV method community [67]. In this work only the approximation to the Riemann problem is considered, since the evaluation of the exact solution (Godunov flux functions) needs too much calculation time.

Since the Euler equations are rotationally invariant a flux normal to a surface can be calculated,

$$\vec{\boldsymbol{\mathcal{G}}} \cdot \vec{n} = \boldsymbol{\mathcal{G}}_n^* = \underline{\mathcal{T}}^{-1} \boldsymbol{\mathcal{G}}_{\mathcal{T}}(\underline{\mathcal{T}}\mathbf{U}). \tag{2.74}$$

The transformation matrix $\underline{\mathcal{T}}$ from the reference system to the coordinate system normal to the surface is given by

$$\underline{\mathcal{T}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & n_1 & n_2 & n_3 & 0 \\ 0 & t_{1_1} & t_{1_2} & t_{1_3} & 0 \\ 0 & t_{2_1} & t_{2_2} & t_{2_3} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.75}$$

where $\vec{n}$ is the normal vector to the surface and $\vec{t}_1$ together with $\vec{t}_2$ define the surface. These three vectors are orthogonal to each other. By rotating the solution $\mathbf{U}_{\mathcal{T}} = \underline{\mathcal{T}}\mathbf{U}$ to the normal direction, only one flux vector has

to be evaluated instead of three. To determine the unique flux a non-linear hyperbolic system has to be solved

$$(\mathbf{U}_\mathcal{T})_t + \boldsymbol{\mathcal{G}}_\mathcal{T}(\mathbf{U}_\mathcal{T})_n = \mathbf{0} \quad \text{or} \quad (\mathbf{U}_\mathcal{T}) + \underline{A}(\mathbf{U}_\mathcal{T})\mathbf{U}_{\mathcal{T}_n} = \mathbf{0}. \tag{2.76}$$

A linearization is introduced $\mathbf{U}_\mathcal{T}|_{\partial E} = \widetilde{\mathbf{U}} = \widetilde{\mathbf{U}}(\mathbf{U}^-_\mathcal{T}, \mathbf{U}^+_\mathcal{T})$ and a Riemann problem for the linearized system is solved

$$(\mathbf{U}_\mathcal{T})_t + \underline{A}(\widetilde{\mathbf{U}})\mathbf{U}_{\mathcal{T}_n} = \mathbf{0}. \tag{2.77}$$

With the set of right eigenvectors $\underline{R}$ of $\underline{A}(\widetilde{\mathbf{U}})$ the above equation can be diagonalized by transforming it to characteristic variables $W = \underline{R}\mathbf{U}_\mathcal{T}$, leading to five decoupled scalar linear advection equations (in case of the Euler-Equations) of the form

$$w_t + \lambda w_x = 0 \tag{2.78}$$

$$(x, t = 0) = \begin{cases} w^- \text{ for } x < 0, \\ w^+ \text{ for } x > 0, \end{cases} \tag{2.79}$$

The eigenvalues of $\underline{A}$ are the advection speeds $\boldsymbol{\lambda}$. Each decoupled equation represents a Riemann problem. The solution of each problem is given by

$$F_a(u)\Big|_{\partial E} = \frac{1}{2}(F_a(u^-) + F_a(u^+)) - a\frac{1}{2}(u^+ - u^-). \tag{2.80}$$

These solutions can be rotated back from the characteristic variables, which gives the exact solution for the linearized Riemann problem (Equation (2.77)). This exact solution approximates the solution for the desired Riemann problem of Equation (2.76)

$$\boldsymbol{\mathcal{G}}_\mathcal{T}(\mathbf{U}_\mathcal{T})\Big|_{\partial E} \approx \underline{A}(\widetilde{\mathbf{U}})\frac{1}{2}(\mathbf{U}^+_\mathcal{T} + \mathbf{U}^-_\mathcal{T}) - \frac{1}{2}\underline{R}^{-1}|\Lambda|\underline{R}(\mathbf{U}^+_\mathcal{T} - \mathbf{U}^-_\mathcal{T}). \tag{2.81}$$

The diagonal matrix with the eigenvalues of the matrix $\underline{A}(\widetilde{\mathbf{U}})$ is denoted by Λ with the eigenvalues $\lambda_1 = \tilde{v}_1 - \tilde{c}$, $\lambda_{2,3,4} = \tilde{v}_1$ and $\lambda_5 = \tilde{v}_1 + \tilde{c}$. Different choices for the linearization state $\widetilde{\mathbf{U}}$ can be found in [67]. The characteristic wave pattern, which illustrates the solution of the Riemann problem can be

seen in Figure 2.6. The eigenvalue λ_1 is the advection speed for the rarefaction wave and λ_5 is the advection speed for the shock wave. The eigenvalues λ_2,λ_3 and λ_4 have the same value and are the advection speed for the contact discontinuity. Over the rarefaction wave and the shock wave all state variables change, but over the contact discontinuity the pressure and the normal velocity stay constant. The region between the rarefaction and shock wave is the so-called star region. All possible solutions for the Riemann problem depending on the states $\mathbf{U}_{\mathcal{T}}^{-}$ and $\mathbf{U}_{\mathcal{T}}^{+}$ are illustrated in [67]. Two different

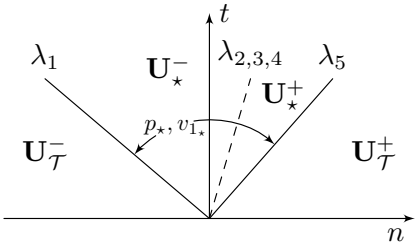

Figure 2.6: Characteristic waves depending on the $-$ and $+$ state

approximations of Equation (2.81) are used in this work, namely the Local-Lax-Friedrich (LLF or Rusanov flux) and the Harten-Lax-van Leer-Contact (HLLC) Riemann solver. Both are widely used for DG implementations. The choice was driven by the fact that the LLF is the simplest Riemann solver and the HLLC is the best choice when efficiency and accuracy are compared [48, 49]. All these shown Riemann solvers can be used with the equation of state introduced in Section 2.2 but they are not specially adapted for the highly accurate EOS. A more accurate Riemann solver for the used EOS is proposed in [16].

2.5.1 Local-Lax-Friedrich

For this simple Riemann solver only the fastest wave speed of both states is used

$$S = \max(|v_{\mathcal{T}_1}^{-}|, |v_{\mathcal{T}_1}^{+}|) + \max(c^{-}, c^{+}). \tag{2.82}$$

This leads to the highest amount of numerical viscosity compared to other Riemann solvers [67]. With this maximum wave speed the numerical flux for the local Lax-Friedrich Riemann solver can be computed with the mean value of the fluxes and a penalty term

$$\boldsymbol{\mathcal{G}}_{\mathcal{T}} = \frac{1}{2}(\boldsymbol{\mathcal{G}}_{\mathcal{T}}^{-} + \boldsymbol{\mathcal{G}}_{\mathcal{T}}^{+}) - \frac{1}{2} S\,(\mathbf{U}_{\mathcal{T}}^{+} - \mathbf{U}_{\mathcal{T}}^{-}). \tag{2.83}$$

2.5.2 Harten-Lax-van Leer-Contact

This kind of Riemann solver approximates the wave speeds for the rarefaction and shock wave and also estimates the wave speed for the contact discontinuity. For the rarefaction and shock wave the Davis wave speed estimates [14] are used

$$S^{-} = \min(v_{\mathcal{T}_1}^{-} - c^{-}, v_{\mathcal{T}_1}^{+} - c^{+}), \tag{2.84}$$

$$S^{+} = \max(v_{\mathcal{T}_1}^{+} + c^{+}, v_{\mathcal{T}_1}^{-} + c^{-}). \tag{2.85}$$

The advection speed for the contact discontinuity is estimated as follows:

$$S_{\star} = \frac{p^{+} - p^{-} + \rho^{-} v_{\mathcal{T}_1}^{-}(S^{-} - v_{\mathcal{T}_1}^{-}) - \rho^{+} v_{\mathcal{T}_1}^{+}(S^{+} - v_{\mathcal{T}_1}^{+})}{\rho^{-}(S^{-} - v_{\mathcal{T}_1}^{-}) - \rho^{+}(S^{+} - v_{\mathcal{T}_1}^{+})}. \tag{2.86}$$

With this the corresponding numerical flux can be calculated

$$\boldsymbol{\mathcal{G}}_{\mathcal{T}} = \begin{cases} \boldsymbol{\mathcal{G}}_{\mathcal{T}}^{-} & \text{if} \quad 0 \leq S^{-}, \\ \boldsymbol{\mathcal{G}}_{\mathcal{T}_\star}^{-} & \text{if} \quad S^{-} \leq 0 \leq S_{\star}, \\ \boldsymbol{\mathcal{G}}_{\mathcal{T}_\star}^{+} & \text{if} \quad S_{\star} \leq 0 \leq S^{+}, \\ \boldsymbol{\mathcal{G}}_{\mathcal{T}}^{+} & \text{if} \quad 0 \geq S^{+}, \end{cases} \tag{2.87}$$

where the fluxes in the star regions are given as

$$\boldsymbol{\mathcal{G}}_{\mathcal{T}_\star}^{\pm} = \boldsymbol{\mathcal{G}}_{\mathcal{T}}^{\pm} + S^{\pm}(\mathbf{U}_{\star}^{\pm} - \mathbf{U}_{\mathcal{T}}^{\pm}). \tag{2.88}$$

Furthermore the solutions in the star regions can be calculated with

$$\mathbf{U}_\star^\pm = \rho^\pm \left(\frac{S^\pm - v_1^\pm}{S^\pm - S_\star} \right) \begin{pmatrix} 1 \\ S_\star \\ v_{\mathcal{T}_2}^\pm \\ v_{\mathcal{T}_3}^\pm \\ E^\pm + (S_\star - v_{\mathcal{T}_1}^\pm) \left[S_\star + \frac{p^\pm}{\rho^\pm (S^\pm - v_{\mathcal{T}_1}^\pm)} \right] \end{pmatrix}. \tag{2.89}$$

The interested reader finds a detailed explanation in [66].

2.6 Boundary Conditions

As explained earlier the elements of the calculation domain are only coupled by the normal surface flux $\boldsymbol{\mathcal{F}}_n^*$ (see Eq. (2.40) and Eq. (2.72)). This is a weak coupling and is also used for the boundary conditions. For a Dirichlet boundary type this means that the boundary state $\mathbf{U}_b$ is necessary to calculate the corresponding flux along with the inner state. With these two states the normal surface flux can be calculated via a Riemann solver. This type of boundary condition is used for inflow and outflow in this work. For the wall boundary a full-slip-wall is implemented, which by default works with the equation of state described in Section 2.2. It is reasonable to neglect the friction at the wall, because the influence of the boundary layer is insignificant for the investigated phenomena. This means that the boundary layer is not resolved in the calculations performed in this work..

2.7 Time Integration

For time integration, a low-storage explicit Runge-Kutta 4^{th} order (LSERK4) algorithm is used [12]. In algorithm 1 the vector $\mathbf{K}$ has the same dimension as the solution vector $\mathbf{U}$. The constants a_i, b_i and c_i can be taken from Table 2.2. t_s is the stage time and can be used for time dependent boundary conditions

$\mathbf{K} = 0$;
$t_s = t$;
for $i \leftarrow 1$ **to** 5 **do**
    $\mathbf{K} = a_i \mathbf{K} + \mathbf{U}_t(\mathbf{U}, t_s)$;
    $\mathbf{U} = \mathbf{U} + b_i \Delta t\, \mathbf{K}$;
    $t_s = t + \Delta t\, c_i$;
end

Algorithm 1: Low storage Runge-Kutta 4th order algorithm

and t is the current simulation time. The time step Δt is calculated with the advection time step for a DGSEM approach

$$\Delta t^a_{\min} = \frac{1}{\lambda^a_{\max}} \frac{\Delta x}{2N+1}, \tag{2.90}$$

where Δx is the element length and $\lambda^a_{\max}$ the maximum eigenvalue of the flux Jacobian matrix for advection. Also the diffusion time step is needed

$$\Delta t^d_{\min} = \frac{1}{\lambda^d_{\max}} \left(\frac{\Delta x}{2N+1} \right)^2, \tag{2.91}$$

where $\lambda^d_{\max}$ is the maximum eigenvalue of the diffusion matrix [13]. Δt is the minimum out of $\Delta t^a_{\min}$ and $\Delta t^d_{\min}$. Since the DG time step is more restrictive than the FV time step [62] it can be taken for both methods to integrate over time.

Table 2.2: Coefficients for the LSERK4 algorithm [12]

i	a_i	b_i	c_i
1	0	$\frac{1432997174477}{9575080441755}$	0
2	$-\frac{567301805773}{1357537059087}$	$\frac{5161836677717}{13612068292357}$	$\frac{1432997174477}{9575080441755}$
3	$-\frac{2404267990393}{2016746695238}$	$\frac{1720146321549}{2090206949498}$	$\frac{2526269341429}{6820363962896}$
4	$-\frac{3550918686646}{2091501179385}$	$\frac{3134564353537}{4481467310338}$	$\frac{2006345519317}{3224310063776}$
5	$-\frac{1275806237668}{842570457699}$	$\frac{2277821191437}{14882151754819}$	$\frac{2802321613138}{2924317926251}$

2.8 Parallelization

The parallelization of the described CFD solver is done with message passing interface (MPI) routines. It is designed to handle one DG element per core at its limit. As it is presented from Atak et al. [4] super-scaling can be achieved on the HLRS Cray XC40 for polynomial degrees higher then six for the one element per core limit. This can be achieved with a communication-hiding technique.

At the beginning of the parallelization the mesh needs to be distributed on the processors used for the simulation. As described in [29] a space-filling curve (SFC) is generated in a preprocessing step which gives each element a unique position on a 1D curve. A SFC has the property that neighboring elements in 2D and 3D spaces are mapped as close as possible on the 1D curve. For each mesh the SFC needs to be build only once and than can be used for any calculation with an arbitrary amount of processors up to the limit of one element per processor. In a first step of the simulation the SFC is cut into equidistant parts where the number of parts equals the number of processors which can vary for each simulation. If the SFC can not be separated

equidistantly, because the number of elements cannot be divided by the number of processors in whole numbers, some parts of the SFC contain a few more elements than others. Each part represents a domain of the mesh and this information is distributed to the processors, therefore each processor calculates a specific number of elements. For the DGSEM only communication with von Neumann neighbors are necessary for one element, because each face needs only the solution from the conjunct face to calculate the fluxes (see Section 2.5 and Equations (2.44) and (2.45)). The rest of the DGSEM operator, namely the volume integral, is an element local operation and does not depend on informations from other elements. But communication is only needed for conjunct faces which are located on different processors, so called MPI-faces. The evaluation of the volume terms as well as the calculation of the fluxes for conjunct faces on one processor can be used to hide the communication for MPI-faces. This can be achieved with non-blocking communications [39]. A more detailed explanation for structured meshes is given in [2] and the extansion for the used unstructured meshes in this work is given in [29].

For the used kind of FV shock capturing the memory layout does not change and the parallelization technique can be easily adapted [62]. As well as for the DGSEM implementation the sending of MPI-face date can be hidden by evaluating the inner element sub-cells in between. Since the reconstruction for the 2^{nd} order FV scheme is also done over MPI interfaces this leads to an additional communication of the needed neighboring mean value to construct the gradient. This communication is again hidden by local processor work and this hiding technique contributes to an efficient method on state of the arts HPC clusters.

3 Efficient Implementation of the Equation of State

To resolve effects like cavitation and shock condensation as well as evaporation in a CFD-simulation, an EOS, which can handle fluid, gas and two-phase states, has to close the Navier-Stokes equations. In this work the CoolProp library [9] in version 4.2.6 is used which evaluates the Helmholtz free energy formulation described in Section 2.2. For water, as an example, the IAPWS-95 [70] standard with thermodynamic equilibrium and Maxwell construction in the two-phase region is implemented in the library. In CoolProp more then 100 fluids are included and could be used for numerical calculations, but the focus in this work lies on water as a fluid. The easiest way to implement the EOS would be to couple CoolProp with the DG solver directly during calculation. That involves also the temperature iteration explained in Subsection 2.2, which is not part of CoolProp. This iteration is needed for density and specific internal energy as input values. The library can handle these values, but in this work control over the iteration process is needed (see Subsection 3.3.3) and this process is also used in a different context explained in Section 3.4.

If one compares the evaluation time between a perfect gas and realistic EOS it is easy to see why a different approach is needed. Table 3.1 shows the evaluation time for density and specific internal energy as input parameters. Both types of EOS use these two inputs and calculate the following output parameters: temperature (T), pressure (p), speed of sound (c), viscosity (μ) and heat conductivity (λ). All these parameters are needed by the CFD solver explained in Chapter 2. The evaluation of the perfect gas EOS is several orders of magnitude faster then the direct CoolProp evaluation with temperature iteration. Also the evaluation time for the realistic EOS varies for the thermodynamic state of the fluid. This means, by using the realistic EOS directly, the performance of the code will drop significantly. This is not usable in a CFD context.

In the next subsections an approach is explained, which reduces this factor extremely, and is labeled 'EOS-quadtree' in Table 3.1.

Table 3.1: Comparison of evaluation time (µs) of perfect gas and realistic EOS

Perfect Gas	0.01712		
	Vapor	Two-Phase	Liquid
direct EOS	232.015	2392.149	260.016
EOS-quadtree	0.32402	0.36802	0.30002

3.1 Quadtree Approach

In this work a quadtree approach is used to reduce the evaluation time overhead of the realistic EOS. The idea originates from [17] and is adapted for the efficient use with the numerical method presented in Chapter 2. The concept is to store the data of the EOS in a quadtree as a preprocessing step and use this quadtree later for several calculations to evaluate the EOS. To build such a quadtree, which stores the information of the realistic EOS, the minimum and maximum values for the input parameters define a two-dimensional space. This area is divided into $2^l \times 2^l$ quadratic identical elements. l is an integer and stands for the level of the quadtree and normally the build process starts at level 1. A typical resolution required to obtain sufficient accuracy is $l > 15$. Without further procedures over 10^9 ($2^{15} \times 2^{15}$) elements for level 15 would be needed. To overcome this problem an adaptive quadtree refinement is implemented. The boundaries for this area are defining the computational domain $\mathcal{Q} = [a_{\text{min}}, a_{\text{max}}] \times [b_{\text{min}}, b_{\text{max}}]$. a and b are the input parameters for the EOS and as explained in Section 2.2 ($a = \rho, b = e$), ($a = T, b = \rho$) or ($a = \rho, b = p$) can be used. The reason why in the second case $a = T$ and $b = \rho$ is explained in Section 3.4. Each element has the size $\left[\Delta_a^l, \Delta_b^l\right] = \left[\frac{a_{\text{max}} - a_{\text{min}}}{2^l}, \frac{b_{\text{max}} - b_{\text{min}}}{2^l}\right]$ depending on the level. Very similar to the DG method explained in Section 2.3, every element is mapped to a unit element $\mathcal{E} \in [0, 1]^2$ with the coordinates ξ_1 and ξ_2. For each element a poly-

nomial representation of the thermodynamic variables in the unit element is built by

$$q_h^N(\xi_1, \xi_2) = \sum_{i,j=0}^{N} \hat{q}_{ij} \psi_{ij}(\xi_1, \xi_2), \quad \psi_{ij} = \ell_i^N(\xi_1)\ell_j^N(\xi_2). \tag{3.1}$$

This is again a nodal interpolation ansatz (see Section 2.3) with a tensor product of 1-D Lagrange polynomials ℓ^N of degree N as basis functions. The nodal value $\hat{q}_{ij}$ is evaluated with the help of the EOS library and two input values

$$\hat{q}_{ij} = q(a_{\text{root}}^{\text{elem}} + \xi_{1_i}\Delta_a^l, b_{\text{root}}^{\text{elem}} + \xi_{2_j}\Delta_b^l). \tag{3.2}$$

N is again the polynomial degree. The root of each element is the lower left corner which is denoted by the minimum values for a and b in this element. The used nodal Lagrange basis functions $\psi(\xi_1, \xi_2)$ are build with Chebyshev-Gauß nodes. These are defined in the interval $[-1, 1]$ and a mapping to the desired unit space is given by

$$\xi_1 = \frac{\xi_{1_{\text{CG}}} + 1}{2} \quad \text{and} \quad \xi_2 = \frac{\xi_{2_{\text{CG}}} + 1}{2}. \tag{3.3}$$

The polynomial degree N can vary for each element. The maximum polynomial degree of an element can be choosen from 1 to 10 at the beginning of the building process. This chosen degree is only the upper limit. For each polynomial degree from 1 to the chosen upper limit the L_∞-error is calculated for every element

$$L_\infty = \max\left(\frac{q_h(\xi_1, \xi_2) - q(\xi_1, \xi_2)}{q(\xi_1, \xi_2)}\right) < \epsilon_t, \quad \xi_1, \xi_2 \in [0, 1]. \tag{3.4}$$

The lowest polynomial degree, which satisfies the L_∞-error, is used in the corresponding element. By adapting the polynomial degree for each element the quadtree storage size is reduced. The infinity norm error-criterion (L_∞) is checked on 20×20 equidistant points in each element, by using the interpolation (q_h) (see Equation (3.1)) and evaluating the EOS directly (q) at these points. The equidistant points also include the element boundaries, which ensures that the jump of the solution over element interfaces is

lower than ϵ_t. In case Equation (3.4) is not satisfied by any polynomial degree for an element, this element is going to be refined by dividing it in 2×2 new quadratic elements. All four new elements are in the next level $(l + 1)$ compared to the refined element, which is in level l. This procedure is repeated for each element until the error is lower than ϵ_t or the maximum level is reached. With this technique not every level contains $2^l \times 2^l$ elements, instead the amount depends on ϵ_t and the chosen upper limit for the polynomial degree. In this implementation $L_q = 32$ is the maximum level because the element localization, e.g., finding the root of an element, is done with a 64-bit binary number. Figure 3.1 shows the 1st and 2nd level of a

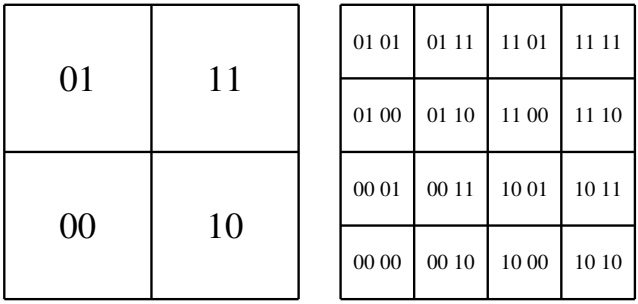

Figure 3.1: Level 1 and 2 with identification numbers

table with the corresponding identification (ID) binary number B. This figure illustrates an example where all elements in level 1 needed refinement. The number B is read from left to right. The first two digits correspond to level one. The second pair of numbers belongs to the second level and so on. This approach is similar to the Morton numbering [40]. Algorithm 2 shows a

way to calculate the root values for each element by using the ID number B.

$a_{\text{root}}^{\text{elem}} = a_{min}$;
$b_{\text{root}}^{\text{elem}} = b_{min}$;
for $l \leftarrow 1$ **to** l_e **do**
    $\Delta_a^l = \frac{a_{max} - a_{min}}{2^l}$;
    $\Delta_b^l = \frac{b_{max} - b_{min}}{2^l}$;
    $a_{\text{root}}^{\text{elem}} = a_{\text{root}} + B[2\,l - 1]\Delta_a^l$;
    $b_{\text{root}}^{\text{elem}} = b_{\text{root}} + B[2\,l]\Delta_b^l$;
end

Algorithm 2: Finding the root for an identification number N

In this algorithm l_e is the level of the element. $B[i]$ denotes the bit at the i^{th} position and the value of $B[i]$ is either 0 or 1. A schematic example of a quadtree is shown in Figure 3.2 with the connections to the previous levels. The top plane represents a level l with four equidistant elements. The middle

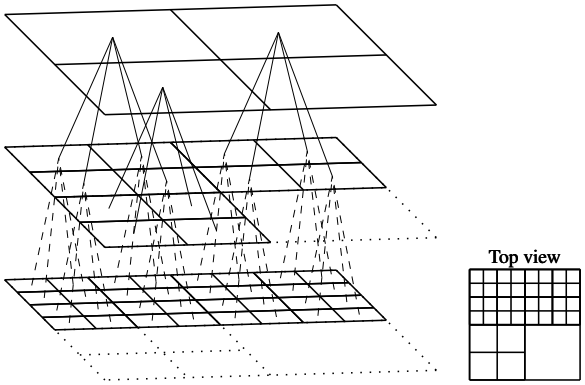

Figure 3.2: Schematic quadtree with level connections

plane shows level $l + 1$. For this schematic example only three elements need to be refined from level l to level $l + 1$ and one satisfies the error criterion stated by Eq. (3.4). The bottom plane shows level $l + 2$ and displays the refined elements from the previous level. The top view is visible in the lower

right of the picture. From level $l+1$ to level $l+2$ four elements do need refinement in this example.

Figure 3.3 shows a quadtree for water build with the above described method for $(a,b) \rightarrow q = (\rho, e) \rightarrow T$, with $\rho \in [\rho_{\text{min}} = 1, \rho_{\text{max}} = 1330]\,\text{kg/m}^3$ and $e \in [e_{\text{min}} = -9.9, e_{\text{max}} = 4.056 \cdot 10^3]\,\text{kJ/kg}$.The building error was set to $\epsilon_t = 1 \cdot 10^{-7}$ and $L_q = 17$ for this quadtree. The black dot in Figure 3.3 is situated in the two-phase region, which is enveloped by the saturation lines and shows one root of an element given by following ID number B

$$B = \underbrace{00}_{l=1}\underbrace{11}_{2}\underbrace{10}_{3}\underbrace{11}_{4}\underbrace{01}_{5} \rightarrow \rho_{\text{root}}^{\text{elem}} = 582.44\,\frac{\text{kg}}{\text{m}^3}\,,\; e_{\text{root}}^{\text{elem}} = 1387.76\,\frac{\text{kJ}}{\text{kg}}. \tag{3.5}$$

Because of very high gradients at the liquid and vapor saturation line the algorithm has to refine along these lines to reach the desired L_∞-error. To reduce the amount of elements due to refinement at these lines, a cut-cell approach is implemented and explained in Section 3.4, which was also used to build Figure 3.3. Every element of such a quadtree is saved during the build process even if it does not satisfy Equation (3.4). This is done to be able to load the quadtree with a higher error than for the build process to save memory during runtime. The memory usage is lower for a higher error since a lower amount of elements has to be loaded.

To evaluate the quadtree during computation the first step is to load the saved quadtree into memory. For the loading process the error can be set higher than the build error $\epsilon_l \geq \epsilon_t$. Starting at level one each element in the first level is checked if the desired error is reached by the polynomial approximation inside the element. For each element where the desired error is not reached the next level is examined and the connection to the upper level is built. This is repeated until the maximum level of the table is reached or all elements satisfy the error ϵ_l. After the table is loaded into memory a fast quadtree bisection method shown in Algorithm 3 is used to find the corresponding element for a given input set (a_g, b_g). In this algorithm INT() symbolizes a integer devision, $(a_{\text{mid}}^{\text{quadtree}}, b_{\text{mid}}^{\text{quadtree}})$ is the middle point of the quadtree. In the first level the element is set to the corresponding i,j element. The row 'elem $\rightarrow$ elem%connection$_{ij}$' changes the current element to the element one level below the current element by using the i,j connection. The point $(a_{\text{mid}}^{\text{upper elem}}, b_{\text{mid}}^{\text{upper elem}})$ is the middle point of the element one level

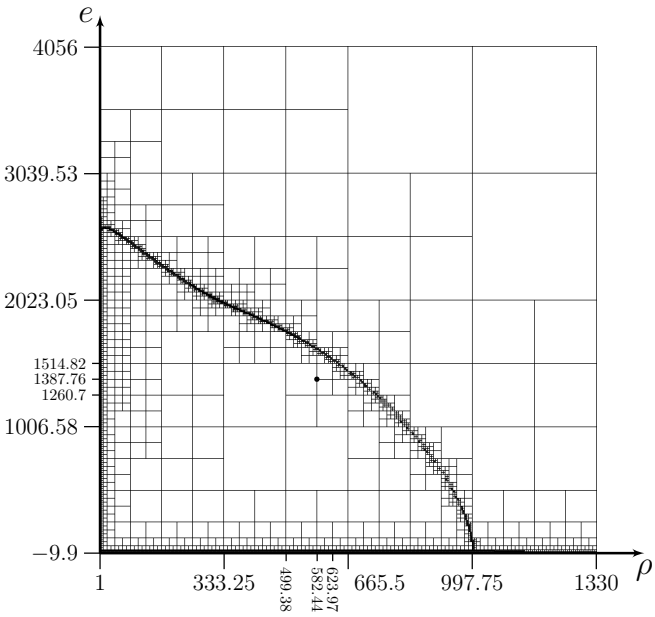

Figure 3.3: A visualization of the quadtree for water (L_q=17). $\rho \in [1, 1330]\,\mathrm{kg/m^3}$ and $e \in [-9.9, 4.056 \times 10^3]\,\mathrm{kJ/kg}$

up, which is connected to the current element. An example for Algorithm 3 is given in Table 3.2 with the given values $a_g = \rho_g = 582.5\,\mathrm{kg/m^3}$ and $b_g = e_g = 1338\,\mathrm{kJ/kg}$. The element ID corresponding to the given values is $B =$ 00 11 10 11 01. This element can have two states, the first is that it contains a approximated solution to the desired variable q and the second state is that this element does not contain a approximated solution. If the first state is true than the approximated solution is evaluated at (ξ_{1_g}, ξ_{2_g}) with Equation (3.1), where ξ_{1_g} and ξ_{2_g} are given by the transformation from physical space to reference space

$$\xi_{1_g} = \frac{a_g - a_{\text{root}}^{\text{elem}}}{\Delta_a^l}, \quad \xi_{2_g} = \frac{b_g - b_{\text{root}}^{\text{elem}}}{\Delta_b^l}. \tag{3.6}$$

$i = \text{INT}\left(\frac{a_g}{a_{\text{mid}}^{\text{quadtree}}}\right)$;
$j = \text{INT}\left(\frac{b_g}{b_{\text{mid}}^{\text{quadtree}}}\right)$;
elem $\rightarrow$ elem$_{ij}$;
while *elem has connections* **do**
  $i = \text{INT}\left(\frac{a_g}{a_{\text{mid}}^{\text{upper elem}}}\right)$;
  $j = \text{INT}\left(\frac{b_g}{b_{\text{mid}}^{\text{upper elem}}}\right)$;
  elem $\rightarrow$ elem%connection$_{ij}$;
end

Algorithm 3: Finding the element for a given input set (a_g, b_g)

To accelerate the evaluation of Equation (3.1) the following property is used

$$\boldsymbol{\psi}(\xi) = \underline{V}^{-T}\boldsymbol{\phi}(\xi) \quad \text{or} \quad \psi_i = \sum_{n=0}^{N} V_{in}^{-T}\phi_n(\xi) \tag{3.7}$$

where $\boldsymbol{\phi}(x) = (1, x, x^2, \ldots, x^N)^T$ is the monomial basis and $\underline{V}$ the so called Vandermonde matrix

$$\underline{V}_{mn} = \phi_n(\xi_m). \tag{3.8}$$

Rewriting Equation (3.1) the result is:

$$q_h^N\left(\xi_{1_g}, \xi_{2_g}\right) = \sum_{i,j=0}^{N} \hat{q}_{ij} \sum_{n=0}^{N} V_{in}^{-T}\phi_n(\xi_{1_g}) \sum_{m=0}^{N} V_{mj}^{-T}\phi_m(\xi_{2_g}). \tag{3.9}$$

The terms $\phi_n(\xi_{1_g})$ and $\phi_m(\xi_{2_g})$ can be precomputed. The evaluation of Equation (3.9) is by a factor of 1.3 faster compared to the evaluation of Equation (3.1) with the Lagrange basis function. This is due the higher computational cost of evaluating the Lagrange basis (see Equation (2.32)). The time to calculate the needed variables temperature, pressure, speed of sound, viscosity and heat conductivity from density and specific internal energy with Equation (3.9) for the quadtree approach can be seen in Table 3.1 compared to the perfect gas equation and the EOS library. If the second state of the element is

Table 3.2: Example for finding the corresponding element for $a_g = \rho_g = 582.5\,\mathrm{kg/m^3}$ and $b_g = e_g = 1338\,\mathrm{kJ/kg}$

Level	Middle value	integer division	Element ID
1	$\rho_{\mathrm{mid}}^{\mathrm{quadtree}} = 665.5\,\mathrm{kg/m^3}$ $e_{\mathrm{mid}}^{\mathrm{quadtree}} = 2023.05\,\mathrm{kJ/kg}$	$i = 0$ $j = 0$	00
2	$\rho_{\mathrm{mid}}^{00} = 333.25\,\mathrm{kg/m^3}$ $e_{\mathrm{mid}}^{00} = 1006.58\,\mathrm{kJ/kg}$	$i = 1$ $j = 1$	11
3	$\rho_{\mathrm{mid}}^{00\,11} = 499.38\,\mathrm{kg/m^3}$ $e_{\mathrm{mid}}^{00\,11} = 1514.82\,\mathrm{kJ/kg}$	$i = 1$ $j = 0$	10
4	$\rho_{\mathrm{mid}}^{00\,11\,10} = 582.44\,\mathrm{kg/m^3}$ $e_{\mathrm{mid}}^{00\,11\,10} = 1260.7\,\mathrm{kJ/kg}$	$i = 1$ $j = 1$	11
5	$\rho_{\mathrm{mid}}^{00\,11\,10\,11} = 623.97\,\mathrm{kg/m^3}$ $e_{\mathrm{mid}}^{00\,11\,10\,11} = 1387.76\,\mathrm{kJ/kg}$	$i = 0$ $j = 1$	01

true, which means that it does not contain a approximate solution, then the calculation stops and a better resolved table has to be used. This second state occurs if the maximum level during loading the quadtree is reached and the approximated solution of the element does not satisfy the loading-error ϵ_l.

3.2 Parallelization

The building process of the quadtree is fully parallelized. It can be build with any number of processors. The parallelization is done level by level. The root core is setting up each level by creating the unique ID numbers B for all elements, which need to build the approximated solution in one level. The amount of the elements, and therefore the amount of identification numbers B, is divided by the number of cores. Each core knows the minimum and maximum of the input variables (e.g. density and specific internal energy). The ID numbers for each core and the current level are broadcasted by the root core to every core. This is illustrated in Figure 3.4 with 4 CPUs. The upper right part of this exemplary quadtree does not need to be evaluated,

because the upper level element satisfies the desired error. Also in Figure 3.4 the black line inside the right picture shows the Morton or Z-curve [40]. This space filling curve aligns all elements on a 1D line and is used to distribute the elements over the available cores. With these ID-numbers each core can

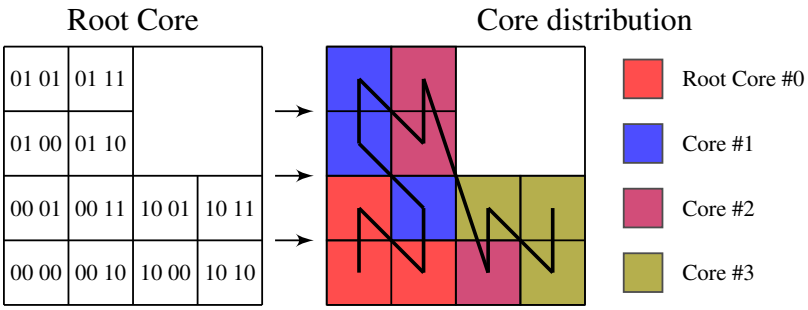

Figure 3.4: Exemplary element distribution

evaluate the polynomial functions to approximate the EOS for each element it received from the root core. When all elements are done the cores send the data back to the root. They send back the following information: maximum error norm for each element, polynomial representation for quadratic element and the needed data for the cut-cell approach if necessary (see Section 3.4). The root collects all this information and builds the quadtree for each level and saves it to disk. If a new refinement level is needed it builds the identification numbers for the new level and sends it evenly distributed to the used cores. If the number of cores is higher than the number of elements some of the cores idle.

The scaling is shown in Figure 3.5. The figure shows with the solid line the scaling of the $(\rho, e) \rightarrow T$-table with the ranges mentioned in Section 3.1. The ideal scaling is shown by the dashed line. For the scaling 262 144 elements were evaluated on different number of cores. This is the amount of elements added to the above mentioned quadtree from level 16 to level 17. The core number ranges from 24 to 6144 on the CRAY XC40 of the HLRS and for calculating the speed up, the run on one 24 core node is the reference value. The numbers of cores are doubled for each run, starting at 24 cores.

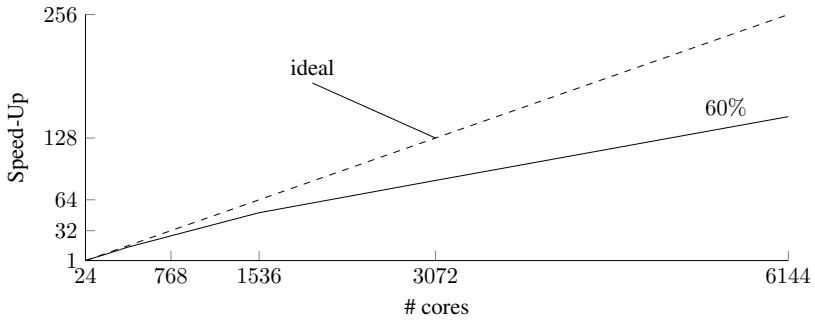

Figure 3.5: Scaling for the table approach

The calculation time for this level needs $3535.21\,\mathrm{s}$ on 24 cores and $23.54\,\mathrm{s}$ on 6144 cores. Keep in mind that this is for one level and the amount of elements is increasing for higher levels. As seen the scaling is not optimal. There are reasons for the scaling drop to 60% on 6144 cores. One reason is the cut-cell approach explained in Section 3.4. This produces load imbalances between the cores, which is more relevant on higher number of cores, because it can happen that one core only evaluates cut-cells and another has only to evaluate normal elements at the same level. A further reason is the communication to the root core. For higher number of cores the CPUs are more separated from each other, which produces higher communication time between them. Also for the 24-core calculation this time is reduced to a minimum, because the CPUs are on a single computation node.

3.3 Quadtree Types

During calculation different quadtree types are needed. As seen in Equation (2.1) the density ρ and the specific total energy E are conservative variables during calculation. As explained in Section 2.2 the specific internal energy $e = E - 0.5\,\vec{v}^2$ is needed to calculate the primitive variables. For the Navier-Stokes equations these are the temperature T, pressure p, speed of sound c, viscosity μ and heat conductivity λ. The conversion $(\rho, e) \rightarrow T, p, c, \mu, \lambda$

is required and since the FV approach uses primitive variables for the reconstruction (see Section 2.4) $(T, \rho) \rightarrow e$ is also needed. This means that at least two quadtrees have to be built, neglecting boundary conditions. With a density range of $\rho = [1 \times 10^{-4}, 1330]\,\mathrm{kg/m^3}$ and a specific internal energy range $e = [-9.9, 4.056 \times 10^3]\,\mathrm{kJ/kg}$ as well as the temperature range $T = [277, 1273]\,\mathrm{K}$ some benchmarks were made. The triple point for water is at $(T, \rho) = (273.16\,\mathrm{K}, 4.8 \times 10^{-3}\,\mathrm{kg/m^3})$. For a temperature lower then the triple point water only exist in solid state which can not be handled in this work. To reduce unnecessary refinement, the lowest temperature is set to 277 K. A density of $1 \times 10^{-4}\,\mathrm{kg/m^3}$ can be reached at higher temperatures but lower then that is not needed in this work.

The main task for the performed benchmarks was to find a optimal ratio between build time and memory usage. The best ratio on build time and used memory during calculation is achieved if the needed quadtree $(\rho, e) \rightarrow T, p, c, \mu, \lambda$ is split into several parts. At first the output variables are reduced to the minimum, which is the temperature. This leads to a $(\rho, e) \rightarrow T$ quadtree and reduces the amount of elements per level since only the temperature is used as a variable for Equation (3.4). Further benchmarks revealed that the refinement needed between the density from $1 \times 10^{-4}\,\mathrm{kg/m^3}$ to $1\,\mathrm{kg/m^3}$ was very high. At levels above 15 half of the elements were located in this density range. The reason for this is, that even level 20 ($\Delta_\rho^{20} \approx 1 \times 10^{-3}\,\mathrm{kg/m^3}$) cannot resolve the lower density regime. To calculate higher levels than level 20 for this kind of quadtree takes several hours on 4800 cores. To overcome that problem the $(\rho, e) \rightarrow T$ is split again into two parts. One $(\rho, e) \rightarrow T$ quadtree from $\rho = [1, 1330]\,\mathrm{kg/m^3}$ and one $(\mathcal{V}, e) \rightarrow T$ quadtree, where $\mathcal{V} = 1/\rho$ is the specific volume with the range $\mathcal{V} = [1, 10000]\,\mathrm{m^3/kg}$. This reduces the needed elements. In addition to the last two presented quadtrees a $(T, \rho) \rightarrow e, p, c, \mu, \lambda$ quadtree is needed to evaluated the other variables. For this quadtree the range for the density is again $\rho = [1 \times 10^{-4}, 1330]\,\mathrm{kg/m^3}$ since levels higher than 20 can be built in less than 10 minutes on 4800 cores. The higher levels can be achieved because the temperature iteration is not needed (see Section 2.2) for this quadtree. As seen in Table 3.3 the benefit of splitting the (ρ, e)-quadtree into several parts causes a memory profit during runtime. All the quadtrees are build until level 17 to be comparable. The amount of used memory is measured per core. The uncovered quadtree area is the sum of all element areas, which do not satisfy the L_∞-norm, divided by

Table 3.3: Benefit of splitting the table into parts

Level 17			
Input	(ρ, e)	(ρ, e) and $(\mathcal{V}, e)$	(T, ρ)
Output	p, T, c, μ, λ	T	e, p, c, μ, λ
L_∞ limit	10^{-6}	10^{-7}	10^{-6}
uncovered quadtree area [%]	2.58×10^{-3}	2.56×10^{-3}	1.87×10^{-4}
building time (4800 Cores) $[s]$	752.6	711.22	48.62
# Cells × # Variables	3 375 755	884 761	742 775
Runtime Memory usage [MB]	583	512	

the whole quadtree area. By adding the building times of the three quadtrees they are almost as fast as building the single quadtree (see Table 3.3). But the main advantage is seen in the memory usage per core during runtime, this can be reduced by 14 % just using three quadtrees instead of one for the above mentioned quadtree ranges. This memory reduction is due to the lower total amount of variables shown in Table 3.3. By building the (T, ρ)-quadtree it includes the $(T, \rho) \rightarrow e$ which is needed because of the primitive reconstruction in the FV sub-cell approach. This was not considered in Table 3.3 which increases the benefit even more by using three quadtrees. A L_∞-error lower than 10^{-6} seems unnecessary but various calculations showed that this is a good error margin to keep the calculation thermodynamically consistent. Higher building errors are not essentially making the calculation unstable but the thermodynamic error adds up with the numerical one. The evaluation time during a simulation is the same since the temperature evaluation for every DOF is mandatory for calculating the temperature gradient. Later in the implementation this temperature can be used to evaluate the other needed variables.

It has to be mentioned that the error for building the $(\rho, e), (\mathcal{V}, e) \rightarrow T$ has to be lower than for the single quadtree, because the evaluated temperature is reused in the (T, ρ)-quadtree. To get the same accuracy in all variables the accumulated error has to be considered for the quadtree approach where the temperature is reused in another table. By knowing the error for the temperature, the error for evaluating the other variables correlates with their normal-

ized gradients. In the (T, ρ)-quadtree the highest normalized gradient for all output parameters is

$$\bar{p}_T|_\rho = \left.\frac{p_T}{p}\right|_\rho \approx 10\,\frac{1}{\mathrm{K}}, \quad \text{with } p_T = \left.\frac{\partial p}{\partial T}\right|_\rho . \tag{3.10}$$

The gradient was built with a second-order finite difference approach

$$p_T|_\rho \approx \frac{p(T + \frac{1}{2}\epsilon_g, \rho) - p(T - \frac{1}{2}\epsilon_T, \rho)}{\epsilon_g}, \tag{3.11}$$

where ϵ_g is set close to the machine precision which is around 10^{-14}. This leads with a build error for the (ρ, e) quadtree of $\epsilon_t = 10^{-7}$ to an accumulated error

$$\epsilon = \left.\frac{p_T}{p}\right|_\rho \times \epsilon_t \approx 10^{-6} \tag{3.12}$$

which is in the same range than the error of the single quadtree.

The quadtrees used during calculation in this work were built with $N = 4$ and are listed in Table 3.4. The table shows the input and output variables. The used quadtree dimensions are stated in x and y direction. The used target error for the building process and the used numbers of cores are also listed in the table. With these used parameters the corresponding building time and the area size which could not be resolved can be found in the table information. Even with the highest level some elements still do not satisfy the error criterion. Since the elements are getting smaller in each level the uncovered area becomes very tiny but is still important because the calculation stops when it tries to evaluate an uncovered area. Since the density range is over seven orders of magnitude, the (T, ρ)-quadtree is built until level 22 since then $\Delta_\rho^{22} = 3.17 \times 10^{-4}\,\mathrm{kg/m^3}$ is achieved, which is needed to cover most of the area lower than $\rho < 1\,\mathrm{kg/m^3}$. In the 'purpose'-row C stands for the conservative variables and P for the primitive ones. The arrow shows the conversion directions for which the quadtree is used. One quadtree is used for both directions as mentioned earlier. For the $C \rightarrow P$ calculation first the $(\rho, e) \rightarrow T$ quadtree is evaluated and with the received temperature the $(T, \rho) \rightarrow (e, p, a, \mu, \lambda)$ quadtree calculates the primitive variables. In the other direction the density and temperature are the primitive inputs and the specific

Table 3.4: Quadtrees for water used in this work

Input	(ρ, e)	$(\mathcal{V}, e)$	(T, ρ)
Output	T	T	e, p, a, μ, λ
x-value range	$[1, 1330]\ \frac{\text{kg}}{\text{m}^3}$	$[1, 10000]\ \frac{\text{m}^3}{\text{kg}}$	$[276, 1273]\ \text{K}$
y-value range	$[-9.9, 4056]\ \frac{\text{kJ}}{\text{kg}}$	$[-9.9, 4056]\ \frac{\text{kJ}}{\text{kg}}$	$[0.0001, 1330]\ \frac{\text{kg}}{\text{m}^3}$
error ϵ_t	1×10^{-7}	1×10^{-7}	1×10^{-7}
build time [min]	7.2	13.4	5
# cores	4800	4800	4800
uncovered area	$1.36 \times 10^{-3}\%$	$2.98 \times 10^{-4}\%$	$9.7 \times 10^{-6}\%$
Purpose	$C \to P$	$C \to P$	$C \leftrightarrow P$
storage size [MB]	175	908	3933
max. quadtree level	17	19	22
total elements	723912	3517520	4838668
Valid elements	43%	52%	39%
Memory usage [MB]		2321	

internal energy is an output. The storage size specifies the needed space on a hard disk. Keep in mind, that all elements are saved to disk regardless the error. The percentage of valid elements shows the amount of elements which satisfy the desired L_∞-error criteria. This means that the needed amount of used memory per core during calculation is lower then the storage size on disk. The percentage lower than 50% needs explanation, which is given with the (T, ρ) quadtree. For a maximum level of 22 this table contains 4 838 668 elements from which 1 907 526 satisfy the L_∞-error criteria. These almost 2 million elements are covering 99.9999903% of the desired quadtree area but also 1 721 476 elements are in the $9.7 \times 10^{-6}\%$ uncovered area. If the next level is built, $1\ 721\ 476 \times 4 = 6\ 885\ 904$ elements need to be evaluated. This is more than the quadtree contains in level 22 and this is only for a very small area. With a maximum level of 23 the quadtree would contain 11 724 572 elements and only 3 098 736 are valid elements. This means a bit more than 1 million elements are satisfying the desired error out of these almost 7 million elements, which have been evaluated for level 23. The area covered by the

valid elements is increasing slightly to 99.9999904%. This explains why the percentage for the valid elements is below 50% for higher levels.

3.3.1 Quadtree for Post-Processing

For post-processing a quadtree can be used with a lower error and more output variables which are not needed during calculation. The lower error reduces the needed disk space drastically. This leads to a very high uncovered area, but for evaluation of the quadtree during post-processing an interpolation is done for these areas. This interpolation is not thermodynamically consistent but sufficient for e.g. visualization. In Table 3.5 more information about such a quadtree is given.

Table 3.5: Post-processing quadtree for water

Input	(T, ρ)
Output	$e, p, c, \mu, \lambda, s, q$
x-value range	$[0.0001, 1330]\ \frac{\text{kg}}{\text{m}^3}$
y-value range	$[274, 1273]\ \text{K}$
error	1×10^{-3}
build time [min]	10
# cores	48
area not covered [%]	0.19
Purpose	Post-processing
size [GB]	0.2
max. quadtree level	14
Valid elements [%]	51

3.3.2 Quadtree for Slip-Wall Boundary Condition

For the slip-wall boundary condition an extra quadtree is needed. Since the pressure at the wall and the density are known, the temperature and specific internal energy have to be evaluated. This leads to the $(\rho, p) \rightarrow (T, e)$-quadtree.

The information for the quadtree are illustrated in Table 3.6. The reason why the uncovered area is very big and why the valid elements in the quadtree are only at 17% is explained in Subection 3.3.3.

Table 3.6: Quadtree for water at slip-wall

Input	(ρ, p)
Output	T, e
x-value range	$[10^{-4}, 1330]\,\mathrm{kg/m^3}$
y-value range	$[0.01, 10000]$ MPa
error ϵ_t	10^{-6}
build time [min]	10
# cores	4800
area not covered [%]	0.1
Purpose	BC
size [MB]	519
max. quadtree level	15
Valid elements [%]	17

3.3.3 Limitation of the Quadtree Approach

As described in Section 2.2 the temperature and the density are the primary input parameters for the Helmholtz free energy formulation. The CoolProp-library in the used version states the minimum usable temperature as $T_{\text{min}} = 273.16\text{K}$ and the maximum $T_{\text{max}} = 1273.0\,\text{K}$ for water. Also for the density the maximum value is given as $\rho_{\text{max}} = 1332.409\,\text{kg/m}^3$. With these values a quadratic shape is represented by the quadtree which is visualized in the middle picture of Figure 3.6. This is the base for the left and the right picture.

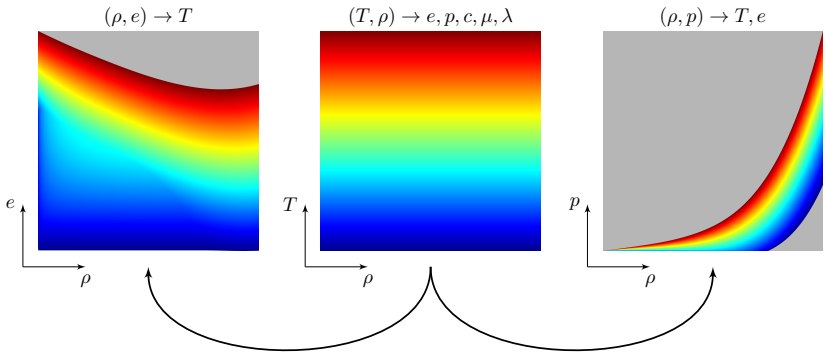

Figure 3.6: Shapes of the different quadtree types based on the (T, ρ)-quadtree colored by the temperature from 277 K (blue) to 1273 K (red)

For the left picture the y-axis is the specific internal energy and for the right picture the y-axis is the pressure. For all three pictures in Figure 3.6 the x-axis is the density. The coloring is the temperature range from blue (277 K) to red (1273 K). In the middle picture this is of course constant across the x-axis because the y-axis is the temperature. In the left and right pictures of Figure 3.6 the colored area is the region which is generated by evaluating the quadtree shown in the middle picture of this figure. This means that only the colored area has to be enveloped by the (ρ, e)- or (ρ, p)-quadtree to be consistent. But this curvy shaped area cannot be handled by the presented quadtree approach

so in addition to the colored area the gray one has to be built on top. In the gray area the CoolProp-library cannot evaluate the needed variables in every point. To solve this problem and avoid unnecessary refinement the temperature is forced to be the maximum or the minimum value in the gray area depending on its location.

This procedure is only done to avoid needless refinement in the quadtree building process. It does not influence the calculation since the colored area is assembled with the correct values and the gray area is not reached because the (T, ρ)-quadtree does not cover that region anyway. For the (ρ, e)-quadtree this approach works well, but for the (ρ, p)-quadtree the algorithm still refines slightly at the boarder between gray and colored area which causes higher uncovered area in this quadtree as seen in Table 3.6. Since this quadtree is only suitable for one type of boundary condition the uncovered area between the important and unimportant regions does not affect the calculation.

3.4 Cut-Cell Approach

To reduce the number of elements and the building time of each quadtree a cut-cell approach is also implemented in the quadtree algorithm. The mechanism was proposed in [17]. To find a cut-cell the saturation lines and the current element are checked for intersection points. These points can be found for the (ρ, e)-quadtree with a given density and the vapor quality (or vapor mass fraction) χ for the liquid or vapor saturation line, where $\chi = 0$ and $\chi = 1$, respectively. To find the line cutting an element, the saturation temperature has to be iterated to find the corresponding temperature $T(\rho, \chi)$. This iteration has to be done, because the used CoolProp version can not handle this two input parameters. The iteration function used is similar to Equation (2.24) and is solved with the same algorithm

$$f(T) = \rho(T, \chi_g) - \rho_g \stackrel{!}{=} 0. \tag{3.13}$$

For each element two saturation temperatures for each χ have to be evaluated with the density ρ_g given by the minimum and maximum values for one element, ($\rho_1 = \rho_{\text{min}}$ and $\rho_2 = \rho_{\text{min}} + \Delta_\rho^l$). Two points are found in each element for each saturation line. These two point are connected by a linear line. This line is checked if it is cutting the element. Figure 3.7 illustrates the procedure

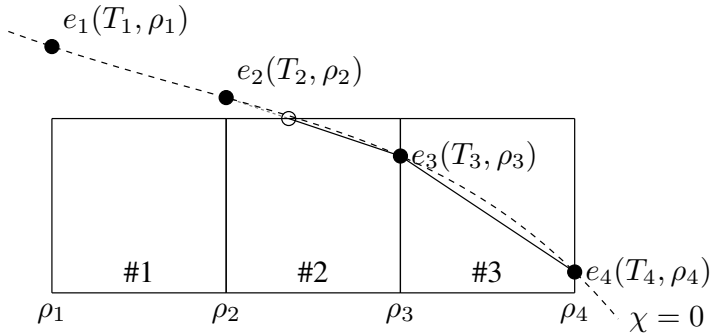

Figure 3.7: Finding cutcells

of finding the cutting line. In this example element number 2 and 3 are cut by the liquid saturation line. For the (ρ, e)-quadtree the found temperature and the given density denote the specific internal energy which is needed for the cutting line. Of course this is a linear approximation to a curved line, but at higher levels the saturation line gets better approximated because the elements are becoming very small. For the (T, ρ)-quadtree the iteration is not needed, because the corresponding saturation density on the saturation lines can be evaluated directly by the CoolProp library with the temperature and the vapor quality as input parameters. This is the reason why the (T, ρ)-quadtree is built and not the (ρ, T)-quadtree. As mentioned earlier CoolProp can not handle (ρ, χ) as input parameters and an additional iteration would be needed if a (ρ, T)-quadtree were used. In Figure 3.8 the four possible cut-cell types are illustrated neglecting the rotation. The saturation line cuts the element into two parts Ω_I and Ω_{II} also seen in Figure 3.8. For each part an approximated polynomial solution is build with the $\mathbb{L}_2$ projection (see Equation (2.38))

$$\langle q_h, \boldsymbol{\phi} \rangle_{\mathbb{L}_2(\Omega)} = \langle q, \boldsymbol{\phi} \rangle_{\mathbb{L}_2(\Omega)}, \tag{3.14}$$

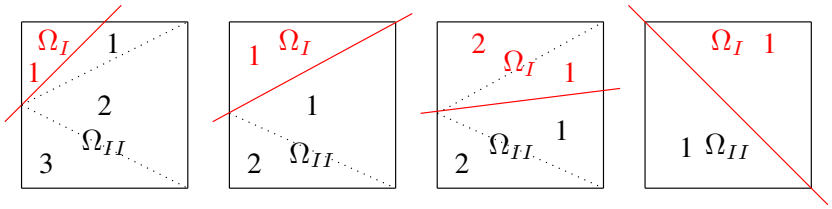

Figure 3.8: 4 possible cut-cells types

with q_h the approximation to the exact solution q and ϕ is a basis function. In the cut-cell case each Ω part is divided by S sub-triangles, where S ranges from one to three as seen in Figure 3.8. At first the exact solution is derived

$$\langle q, \phi \rangle_{\mathbb{L}_2(\Omega)} = \int_\Omega q(\xi_1, \xi_2)\phi(\xi_1, \xi_2)d\xi_1 d\xi_2 = \sum_{l=1}^{S} \int_{\Delta_l} q^l(\xi_1^l, \xi_2^l)\phi(\xi_1^l, \xi_2^l)d\xi_1^l d\xi_2^l. \tag{3.15}$$

In Equation (3.15) the integration domain is divided into S triangles. For the basis function ϕ a monomial representation of degree $N = 2$ is used, which in the case of a triangle leads to $(N+1)(N+2)/2$ functions

$$\phi(\xi_1, \xi_2) = (1, \xi_1, \xi_1^2, \xi_2, \xi_1\xi_2, \xi_2^2)^T. \tag{3.16}$$

An integration on a triangle can be exchanged with

$$\int_{\Delta_l} q^l(\xi_1^{\Delta_l}, \xi_2^{\Delta_l})\phi(\xi_1^{\Delta_l}, \xi_2^{\Delta_l})d\xi_1^{\Delta_l} d\xi_2^{\Delta_l} = \int_{\square_l} Jq^l(\xi_1^{\square_l}, \xi_2^{\square_l})\phi(\xi_1^{\square_l}, \xi_2^{\square_l})d\xi_1^{\square_l} d\xi_2^{\square_l}, \tag{3.17}$$

where $J = J(\xi_1^{\square_l}, \xi_2^{\square_l})$ is the *Jacobian* of the mapping. In this case again the unite space $[0, 1]$ is used. The mapping is then defined as

$$\vec{\xi}^{\Delta_l}(\vec{\xi}^{\square_l}) = \begin{pmatrix} \xi_1^{\square_l} \\ \xi_2^{\square_l}(1 - \xi_1^{\square_l}) \end{pmatrix}, \tag{3.18}$$

which leads to the *Jacobian*:

$$J(\xi_1^{\square_l}, \xi_2^{\square_l}) = \det\left(\frac{\partial\vec{\xi}^{\Delta_l}}{\vec{\xi}^{\square_l}}\right) = 1 - \xi_1^{\square_l}. \tag{3.19}$$

Inserting this into Equation (3.17) leads to

$$\int_{\Delta_l} q^l \phi d\xi_1^{\Delta_l} d\xi_2^{\Delta_l} = \int_{\square_l} (1-\xi_1^{\square_l}) q^l \phi d\xi_1^{\square_l} d\xi_2^{\square_l}. \tag{3.20}$$

To integrate the above equation, the Jacobi-Gauss quadrature is well suited

$$\int_0^1 \int_0^1 f(\xi_1,\xi_2)(1-\xi_1)^{\alpha_{\xi_1}}(1+\xi_1)^{\beta_{\xi_1}}(1-\xi_2)^{\alpha_{\xi_2}}(1+\xi_2)^{\beta_{\xi_2}} d\xi_1 d\xi_2, \tag{3.21}$$

with the coefficients $\alpha_{\xi_1} = 1$, $\beta_{\xi_1} = 0$, $\alpha_{\xi_2} = 0$ and $\beta_{\xi_2} = 0$. Normally this is defined in $[-1,1]$ and a mapping is needed:

$$\xi_1 = \frac{\xi_{1_{\text{JG}}} + 1}{2} \quad \text{and} \quad \xi_2 = \frac{\xi_{2_{\text{JG}}} + 1}{2}. \tag{3.22}$$

With these coefficients the integration is done with

$$\int_0^1 \int_0^1 f(\xi_1,\xi_2)(1-\xi_1) d\xi_1 d\xi_2 = \sum_{m=1}^{N+1} \sum_{n=1}^{N+1} w_m^{1,0} w_n^{0,0} f(\xi_{1_m}, \xi_{2_n}), \tag{3.23}$$

where $w_m^{1,0}$ and $w_n^{0,0}$ are the integration weights for the chosen α and β also mapped to the desired space:

$$w^{1,0} = \frac{1}{4} w_{\text{JG}}^{1,0} \quad \text{and} \quad w^{0,0} = \frac{1}{2} w_{\text{JG}}^{0,0}. \tag{3.24}$$

Now the Equation (3.20) can be integrated using the explained Jacobi-Gauss quadrature with the corresponding Jacobi-Gauss nodes, assuming $\square_l \in [0,1]^2$,

$$\int_{\square_l} (1-\xi_1^{\square_l}) q^l \phi d\xi_1^{\square_l} d\xi_2^{\square_l} = \sum_{m=1}^{N+1} \sum_{n=1}^{N+1} w_m^{1,0} w_n^{0,0} q^l(\xi_{1_m}^{\square_l}, \xi_{2_n}^{\square_l}) \phi(\xi_{1_m}^{\square_l}, \xi_{2_n}^{\square_l}). \tag{3.25}$$

Inserting the above steps into Equation (3.15) leads to

$$\langle q, \phi \rangle_{\mathbb{L}_2(\Omega)} = \sum_{l=1}^{S} \sum_{m=1}^{N+1} \sum_{n=1}^{N+1} A^l w_m^{1,0} w_n^{0,0} q^l(\xi_{1_m}^{\square_l}, \xi_{2_n}^{\square_l}) \phi(\xi_{1_m}^{\square_l}, \xi_{2_n}^{\square_l}), \tag{3.26}$$

with A^l being the percentage area covered by each triangle in the cut-cell. The exact solution q^l can be calculated with the EOS presented in Section 2.2.

The approximated solution is calculated by

$$q_h = \sum_{i=j}^{\frac{(N+1)(N+2)}{2}} a_j \phi_j = \mathbf{a} \cdot \boldsymbol{\phi}, \tag{3.27}$$

using the same monomial basis

$$q_h = a_0 + a_1\xi_1 + a_2\xi_1^2 + a_3\xi_2 + a_4\xi_1\xi_2 + a_5\xi_2^2 = \mathbf{a} \cdot \boldsymbol{\phi}(\xi_1, \xi_2). \tag{3.28}$$

Applying the same steps as for the exact solution this leads to

$$\langle q_h, \boldsymbol{\phi} \rangle_{\mathbb{L}_2(\Omega)} = \sum_{l=1}^{S} \sum_{m=1}^{N+1} \sum_{n=1}^{N+1} A^l w_m^{1,0} w_n^{0,0} \mathbf{a} \cdot \boldsymbol{\phi}(\xi_{1_m}^{\square_l}, \xi_{2_n}^{\square_l}) \boldsymbol{\phi}(\xi_{1_m}^{\square_l}, \xi_{2_n}^{\square_l}). \tag{3.29}$$

To find the unknown $\mathbf{a}$, an equation system needs to be solved

$$\underline{M}\mathbf{a} = \mathbf{b} \rightarrow \mathbf{a} = \underline{M}^{-1}\mathbf{b}, \tag{3.30}$$

where $b_i = \langle q, \phi_i \rangle_{\mathbb{L}_2(\Omega)}$ and $\underline{M}$ is the so called mass matrix defined by

$$M_{ij} = \sum_{l=1}^{S} \sum_{m=1}^{N+1} \sum_{n=1}^{N+1} A^l w_m^{1,0} w_n^{0,0} \phi_j(\xi_{1_m}^{\square_l}, \xi_{2_n}^{\square_l}) \phi_i(\xi_{1_m}^{\square_l}, \xi_{2_n}^{\square_l}). \tag{3.31}$$

As mentioned earlier two polynomial representations for the solution in each cut-cell are built. For the part Ω_I the $\mathbf{a}_I$ are calculated and for Ω_{II} the $\mathbf{a}_{II}$. The inversion of the mass matrix (Equation (3.30)) for each part is done with the use of LAPACK [3].

To evaluate the approximate solution in the cut cells the scalar product seen in Equation (3.28) is calculated. Depending on the location of the desired evaluation point, $\mathbf{a}_I$ or $\mathbf{a}_{II}$ is used. Also for the cut-cell the maximum error is evaluated on 20×20 equidistant points containing the edges of the element. This evaluation is done on the quadratic uncut element using $\mathbf{a}_I$ and $\mathbf{a}_{II}$ depending on the location of the equidistant point. If the element fulfills the condition of Equation (3.4), the element does not need more refinement and is saved to the quadtree. The cut-cell approach is only used if the normal polynomial representation (see Section 3.1) does not satisfy the desired error.

3.4.1 Performance Gain with Cut-Cells

To show the importance of the cut-cell approach, the building process of EOS-quadtrees is compared with and without (w/o) the use of cut-cells. To build each quadtree 4800 cores were used. The maximal polynomial degree is set to $N = 4$ for the calculations. The desired error was set to $\epsilon_t = 10^{-7}$ for both types. For higher levels the gain of the cut-cells gets bigger because cut-cells found in early levels reduce the amount of normal elements significantly. The quadtree dimensions are

$$(\rho, e) = \left(1 \text{ to } 1330 \, \frac{\text{kg}}{\text{m}^3}, -9.9 \text{ to } 4056 \, \frac{\text{kJ}}{\text{kg}}\right) \tag{3.32}$$

$$(T, \rho) = \left(277 \text{ to } 1273 \, \text{K}, 1 \times 10^{-4} \text{ to } 1330 \frac{\text{kg}}{\text{m}^3}\right). \tag{3.33}$$

In Table 3.7 the result for the above mentioned quadtrees is shown. The

Table 3.7: Compare EOS-quadtrees without and with cut-cells

Level 17	$(\rho, e) \to T$		
	w/o cut-cells	with cut-cells	reduction
# elements	658 012	309 971	2.12
Uncovered Area	$2.33 \times 10^{-3}\%$	$1.37 \times 10^{-3}\%$	1.7
# cut-cells	0	2 499	-
build time	757.47s	464.49s	1.63
Level 22	$(T, \rho) \to e, p, c, \mu, \lambda$		
	w/o cut-cells	with cut-cells	reduction
# elements	13 702 089	1 903 821	7.18
Uncovered Area	$4.18 \times 10^{-5}\%$	$6.73 \times 10^{-6}\%$	6.211
# cut-cells	0	154 284	-
build time	989.69s	250.51s	3.95

(ρ, e) quadtree is built until level 17. The algorithm finds 2499 cut-cells which satisfy the build error. By using the cut-cell approach the build time of this

table is reduced by a factor of 1.63, the uncovered area is reduced by a factor of 1.7 and in total less than half of the elements, compared to the quadtree without cut-cell approach, are needed. For the (T, ρ) quadtree, which is built until level 22, these reductions are even higher as seen in Table 3.7.

3.5 Efficient Implementation of the Equation of State in Brief

The implemented quadtree approach reduces the evaluation time of the accurate EOS by a factor up to 6500. In a preprocessing step the quadtree is built on an arbitrary number of cores with a good scaling capability. The data of the EOS is stored with a nodal polynomial representation. Over 100 fluids can be used to build a quadtree provided by the CoolProp library. During calculation the quadtree is loaded into the memory of each processor and evaluated with a bisection quadtree algorithm as well as a fast interpolation of the polynomial solution. For boundary conditions and post processing of the CFD data also special quadtrees can be built. This approach is not limited to a special thermodynamic property library and any EOS could be stored in a quadtree.

4 Results

The Chapters 2 and 3 explained the numerical method and the essential equation of state. These are now used in this chapter to solve numerical calculations. To show that these components work well together at first the convergence is demonstrated in one- and three-dimensional (3D) space. In 1D validation calculations are performed which are taken from two publications showing two-phase flows. In the work of Dumbser et al. [17] the focus lies on two-phase flows and cavitation and also in the work of Saurel et al. [56] shock condensation is considered. With some of these calculations the Riemann solvers are compared, described in Section 2.5. After these validations a two-dimensional (2D) space calculation is demonstrated. A 2D hydrofoil, where cavitation occurs, is used as showcase. Since the EOS-quadtree implementation increases the calculation time a very good parallel computing performance is needed to reduce the overall computation time. This is shown with a 3D industrial application. A water throttle, where cavitation occurs, is used for this purpose. Also for this industrial application the benefit of the coupled DG/FV approach is shown, compared to a pure FV method.

4.1 Convergence

To show the correct implementation of the methods convergence studies are made. For this purpose a manufactured solution [52] is built. As explained in

[23] an arbitrary smooth solution has to be chosen which is inserted into the Navier-Stokes equations to calculate the source term $\mathbf{P}$. The solution is set to:

$$\mathbf{U} = \begin{pmatrix} \sin(\beta)\gamma + 2 \\ \sin(\beta)\gamma + 2 \\ \sin(\beta)\gamma + 2 \\ \sin(\beta)\gamma + 2 \\ (\sin(\beta)\gamma + 2)^2 \end{pmatrix}, \tag{4.1}$$

where $\beta = k \sum_{j=1}^{d} x_j - \omega t$. The constants γ, k and ω are set to $\gamma = 0.1$ and $k = \omega = 2\pi$ in this work. The variable d represents the space dimension. Inserting this solution into Equation (2.1) the source term is derived as:

$$\mathbf{P} = \gamma \begin{pmatrix} \cos(\beta)\,(d\,k - \omega) \\ \cos(\beta)\,A + \sin(2\beta)\gamma\,k\,(\kappa - 1) \\ \cos(\beta)\,A + \sin(2\beta)\gamma\,k\,(\kappa - 1) \\ \cos(\beta)\,A + \sin(2\beta)\gamma\,k\,(\kappa - 1) \\ \cos(\beta)\,B + \sin(2\beta)\gamma\,(d\,k\kappa - \omega) + \sin(\beta)\left(\frac{d\,k^2\mu\kappa}{\mathrm{Pr}}\right) \end{pmatrix}, \tag{4.2}$$

with

$$\begin{aligned} A &= -\omega + \frac{k}{d-1}\left((-1)^{d-1} + \kappa\,(2\,d - 1)\right), \\ B &= \frac{1}{2}\left((d^2 + \kappa(6 + 3\,d))\,k - 8\,\omega\right). \end{aligned} \tag{4.3}$$

To perform a convergence test the initial solution of the test is Equation (4.1) with $t = 0\,\mathrm{s}$. The CFD method described in Chapter 2 solves the NSE with the source term introduced above. Since the exact solution is know for every time t an error between the calculated and exact solution can be evaluated. This test is performed in 1D and also in 3D setups to show correct implementation in both spaces. By increasing the number of elements the spatial convergence can be evaluated. All convergence tests are done with periodic boundary conditions. As equation of state the quadtree back-end is used with the perfect gas EOS. For this EOS the fluid properties $\kappa = 1.4$ (heat capacity

ratio), Pr $= 0.72$ (Prandtl number) and μ (viscosity) are constant which is necessary to derive Equation (4.2). If the desired convergence is reached with the perfect gas EOS it can be assumed that the implementation for all equations of state is correct since only the quadtree has to be exchanged and this has been tested in Chapter 3. For each test two convergence-rates are shown. The first is achieved by setting the viscosity $\mu = 0$ so that the Navier-Stokes equations are reduced to the Euler equations and only the advection problem is solved. The second test solves the full Navier-Stokes equation by setting μ in a way that the viscous terms dominate ($\mu = 20$). This is stated as a diffusion problem in the next subsections. This test cannot be achieved with water as a test fluid since the viscosity cannot be chosen arbitrarily and κ as well as Pr are not constant with the explained EOS in Section 2.2. By doing this the implementations of $\vec{\mathbf{F}}_a(\mathbf{U})$ and $\vec{\mathbf{F}}_d(\mathbf{U}, \nabla_x \mathbf{U})$ (see Equation (2.1)) are checked separately. The DG and FV methods are tested each on its own and in combination. The experimental order of convergence (EOC)

$$\text{EOC} = \frac{\log\left(\frac{L_2^{\text{coarse}}}{L_2^{\text{fine}}}\right)}{\log\left(\frac{\Delta x^{\text{coarse}}}{\Delta x^{\text{fine}}}\right)} \tag{4.4}$$

should converge to EOC $\approx N + 1$ for the DG method and EOC ≈ 2 for the FV and mixed ansatz. In the equation above L_2 stands for the calculated error norm and Δx for the length of one DG element on a coarse and a fine resolution. Keep in mind that for the pure FV method the DG elements are split into equidistant sub-cells. Therefore Δx is also the length of one DG element even if the NSE is solved only by the FV sub-cell method. The mixed approach is tested in a way that no DG element is connected to an other DG element, there is always a FV sub-cell element in between. This is also true for all directions in 3D. For all tests the LLF Riemann solver was used. In this work only convergence rates for $N = 3$ and $N = 4$ are shown. Rates for higher polynomial degrees with the pure DGSEM implementation are shown in [30] and for the FV method in [62] but those rates were built without the EOS-quadtree approach.

4.1.1 1D Convergence

For the 1D convergence setup a domain is used with the edge length of 2. With the above described manufactured solution the pure DG method, the pure FV approach and the mixed ansatz are calculated. The first is a pure DG mesh. The result is shown in Table 4.1. For both polynomial degrees the EOC reaches the correct value. It has to be mentioned that for high order simulations the order of the time discretization (see Section 2.7) can limit the overall convergence. This happens if the spatial discretization error is lower than the temporal one, which is the case in the pure DG calculation with $N = 4$ if no special care is taken: by multiplying the minimum time step with 0.1 the accuracy of the time integration is increased and the spatial error dominates the calculation which leads to the correct convergence rate for $N = 4$. To check the convergence for the FV scheme all elements are forced to be represented as FV sub-cells. As seen in Table 4.2 an order of up to 1.69 is reached for both the advection and diffusion dominated convergence. As mentioned in Section 2.4 for the second order reconstruction and the advection flux the MinMod-limiter is used. This decreases the theoretical order and is well known for a reconstruction with TVD limiter [51]. The important part of this code is the combination of the DG and the FV method. Since both methods reach convergence on their own, it has to be shown that it also holds for the combination. For the third test in 1D a mixed mesh is used. This means that every other element uses the DG method and the rest of the elements uses the FV sub-cell approach. By comparing the results for the advection EOC and diffusion EOC in Table 4.3 we see the influence of the gradients for the viscous flux calculation (see Subsection 2.4). The second order can be reached because of the combination of the DG method and central evaluated gradients in the FV approach. Here only half of the elements are FV sub-cells and the error gets smaller because of this. Also the advection part still reaches around 1.7 and the EOC is slightly better than for the pure FV sub-cell test-case, see Table 4.3.

Table 4.1: L_2 errors and convergence rates of the density for Eq. (4.1) for the pure DG method.

		advection		diffusion	
	# elements	L_2	EOC	L_2	EOC
N=3	24	8.17×10^{-8}		7.77×10^{-8}	
	48	4.80×10^{-9}	4.09	5.23×10^{-9}	3.89
	96	2.99×10^{-10}	4.01	2.89×10^{-10}	4.18
	192	1.87×10^{-11}	4.00	1.81×10^{-11}	4.00
N=4	24	1.15×10^{-9}		1.90×10^{-9}	
	48	4.02×10^{-11}	4.84	6.45×10^{-11}	4.88
	96	1.27×10^{-12}	4.98	2.00×10^{-12}	5.01

Table 4.2: L_2 errors and convergence rates of the density for Eq. (4.1) for pure FV sub-cells method.

		advection		diffusion	
	# elements	L_2	EOC	L_2	EOC
N=3	24	1.65×10^{-3}		3.61×10^{-5}	
	48	5.14×10^{-4}	1.68	1.20×10^{-5}	1.58
	96	1.60×10^{-4}	1.69	3.99×10^{-6}	1.59
	192	5.00×10^{-5}	1.67	1.31×10^{-6}	1.60
N=4	24	1.13×10^{-3}		2.54×10^{-5}	
	48	3.52×10^{-4}	1.69	8.45×10^{-6}	1.59
	96	1.10×10^{-4}	1.68	2.79×10^{-6}	1.60
	192	3.44×10^{-5}	1.67	9.18×10^{-7}	1.61

Table 4.3: L_2 errors and convergence rates of the density for Eq. (4.1) for mixed DG and FV sub-cells method.

		advection		diffusion	
	# elements	L_2	EOC	L_2	EOC
N=3	24	1.26×10^{-3}		3.46×10^{-5}	
	48	4.45×10^{-4}	1.50	1.04×10^{-5}	1.74
	96	1.27×10^{-4}	1.81	2.88×10^{-6}	1.85
	192	3.61×10^{-5}	1.81	7.33×10^{-7}	1.97
N=4	24	9.78×10^{-4}		2.37×10^{-5}	
	48	3.06×10^{-4}	1.67	6.92×10^{-6}	1.78
	96	8.32×10^{-5}	1.88	1.87×10^{-6}	1.89
	192	2.47×10^{-5}	1.75	4.72×10^{-7}	1.98

4.1.2 3D Convergence

Table 4.4: L^2 errors and convergence rates of the density for Eq. (4.1) for DG and FV sub-cells method with a polynomial degree $N = 3$ of the DG elements.

		advection		diffusion	
	# elem.	L_2	EOC	L_2	EOC
DG	12^3	2.28×10^{-6}		2.18×10^{-6}	
	24^3	1.41×10^{-7}	4.02	1.43×10^{-7}	3.93
	48^3	8.88×10^{-9}	3.99	9.90×10^{-9}	3.86
FV	12^3	1.46×10^{-3}		2.50×10^{-4}	
	24^3	5.20×10^{-4}	1.49	7.62×10^{-5}	1.71
	48^3	1.59×10^{-4}	1.70	2.20×10^{-5}	1.79
DG/FV	12^3	1.85×10^{-3}		2.06×10^{-4}	
	24^3	6.99×10^{-4}	1.40	6.46×10^{-5}	1.67
	48^3	2.38×10^{-4}	1.55	1.94×10^{-5}	1.74
	96^3	7.15×10^{-5}	1.73	5.53×10^{-6}	1.81

It is very important to check the convergence behavior also in 3D, since in 1D only the code implementation in one space direction is investigated. But for the 3D convergence test the spatial resolution is lower than for the 1D cases because of the higher computational costs. A diagonally moving sine wave with constant velocity passes through a cubic mesh with an edge length of 2. The pure DG and FV test cases generate good results (see Table 4.4). For the mixed mesh in 3D again no DG element is adjacent to another DG element which results in a 3D checker board pattern. If these rates are compared with the 1D results, they show the same behavior and the L_2 error is in the same range. This proves the correct behavior of the implemented code.

4.2 1D Validations

For validation of the used EOS approach a couple of 1D calculations are discussed in this section. Five Riemann-Problems for water are described by Dumbser et al. [17]. Each is defined by a left and a right initial state, U^{left} and U^{right}. All of the states are either in liquid or two-phase region. In the work by Dumbser et al. the Euler equations are used but in this work the Navier-Stokes equations are solved instead. Three additional calculations are performed. Two which produce evaporation by rarefaction or strong heating are also taken from Dumbser et al. [17], the third one is a shock condensation test-case taken from [56]. For all 1D validations the FV sub-cell approach is active and the chosen indicator-limit is set in a way that some FV sub-cells occur to stabilize the calculations. The adjustment of the indicator-limit is not perfect and more FV sub-cells occur than needed for the presented simulations. In Subsection 4.2.1.8 the influence of the chosen indicator-limit is shown and for one calculation it is adjusted to reduce the amount of FV sub-cells. This validates the complete setup between CFD solver and EOS-quadtree approach.

4.2.1 Riemann Problems

All Riemann problem calculations are performed on a 50 element DG mesh with a polynomial degree of $N = 3$. This leads to 200 DOF. The Riemann problems for water are named RP-W1 to RP-W5. To solve those the Local-Lax-Friedrich Riemann solver is used. The exact solution for comparison is taken from [17]. Also a comparison of the different Riemann solvers introduced in Section 2.5 is made.

In the top plot of Figure 4.1 a phase diagram for water is shown. The dashed line is the vapor saturation line and the dotted line is the liquid saturation line. The horizontal dash-dotted line denotes the critical temperature. Above this temperature water is in the supercritical phase. The vapor, two-phase and liquid phase are separated by the saturation lines. The critical point is labeled 'CP' in this plot. In the same plot all locations of RP-W1 to RP-W5 are shown by rectangular shapes. These rectangles give an overview which phases (liquid, vapor and two-phase) are possible during each RP-W. RP-W1 is nested in the region of RP-W3. In Figure 4.1 also all zooms into the RP-W regions are illustrated. Each plot for a single RP-W in Figure 4.1 shows

the thermodynamic path by evaluating each DOF at a certain time and plot the temperature and density for each DOF. Also in each plot the locations of the left shock wave (between L and $L_\star$), the contact discontinuity (between $L_\star$ and $R_\star$) and the right shock wave (between R and $R_\star$) are plotted. For the different waves and states of a Riemann Problem see Section 2.5. For the RP-Ws no evaporation wave occurs.

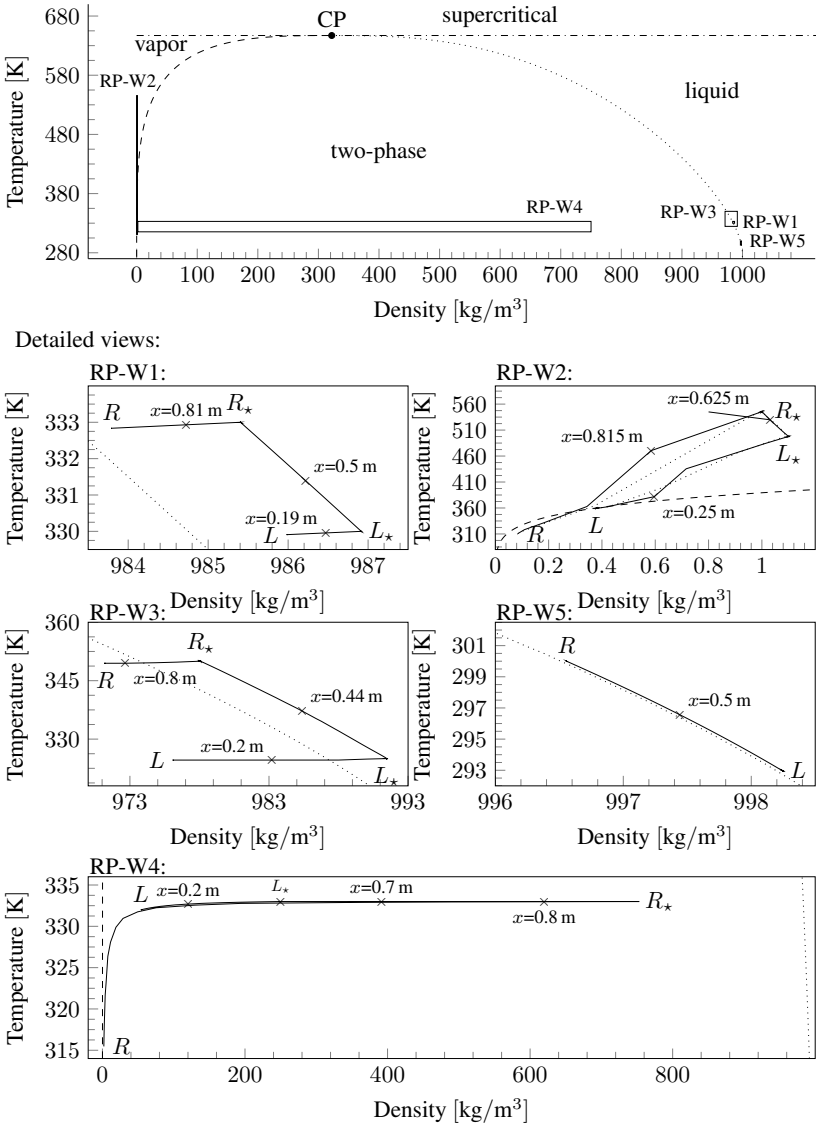

Figure 4.1: Phase diagram of water and locations of the 5 Riemann problems

4.2.1.1 RP-W1

For RP-W1 the given states are

$$\mathbf{U}_p^{\text{left}} = \begin{pmatrix} 985.9853\,\text{kg/m}^3 \\ 6.4656\,\text{m/s} \\ 329.9096\,\text{K} \end{pmatrix}, \qquad \mathbf{U}_p^{\text{right}} = \begin{pmatrix} 983.7899\,\text{kg/m}^3 \\ 2.44908\,\text{m/s} \\ 332.8354\,\text{K} \end{pmatrix}, \tag{4.5}$$

in primitive variables $\mathbf{U}_p = (\rho, v_1, T)$. The initial location of the discontinuity is $x = 0.5\,\text{m}$ in a tube with a length of $1\,\text{m}$. The calculation is performed to an end time of $t = 2 \times 10^{-4}\,\text{s}$. Both initial states are in the liquid region and also the intermediate states. This is also shown in the zoom of RP-W1 in Figure 4.1. The left (L) and right (R) state are very close located to the liquid saturation line. The intermediate states $L_\star$ and $R_\star$ are further away from the saturation line. In this Riemann problem no phase change occurs because all the four states are in the liquid phase. The solid black line in the zoom of RP-W1 in Figure 4.1 represents the thermodynamic path of this Riemann problem for the specified end time. The path is drawn over the complete tube length. On the path the location of the left shock wave ($x = 0.19\,\text{m}$) and contact discontinuity ($x = 0.5\,\text{m}$) as well as the right shock wave ($x = 0.81\,\text{m}$) are marked. These waves and the discontinuity are also plotted in Figure 4.2 where the numerical (black solid line) and exact solution (dashed line) for the primitive variables and the pressure are compared. The lines match very well, but of course the intermediate contact is smeared by this kind of Riemann solver. In the density plot in top right corner of Figure 4.2 also the elements are shown where the FV sub-cell approach is active. The activation is visualized by thick horizontal line segments at the top region of the plot. The indicator-limit for switching from DG representation to the FV sub-cell approach is adjusted very sensitively and all three discontinuities are resolved with the FV sub-cell scheme.

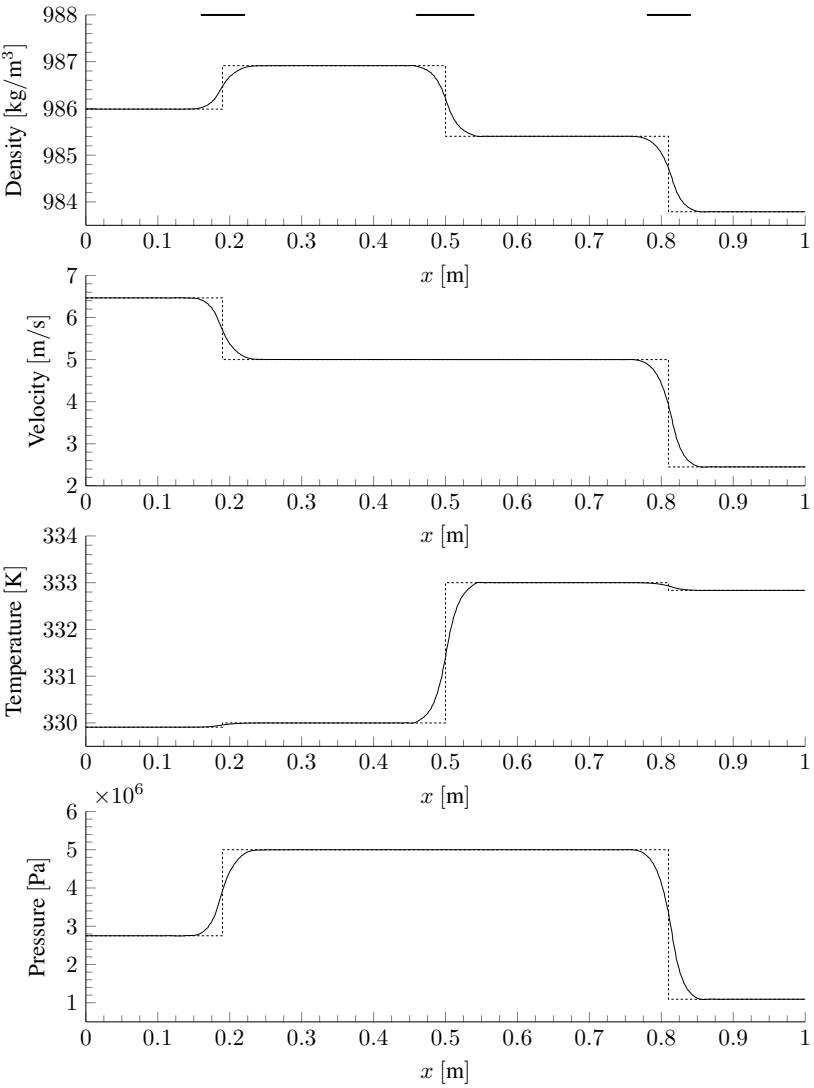

Figure 4.2: Riemann Problem RP-W1 at $t = 2 \times 10^{-4}$ s: comparison of the exact solution (dashed) with the numerical solution (solid). FV sub-cells are shown in density-plot

4.2.1.2 RP-W2

For the RP-W2 the initial discontinuity is again at $x = 0.5\,\mathrm{m}$ and the given states are located in the two-phase region,

$$\mathbf{U}_p^{\text{left}} = \begin{pmatrix} 0.376413\,\mathrm{kg/m^3} \\ 676.6966\,\mathrm{m/s} \\ 358.9432\,\mathrm{K} \end{pmatrix}, \qquad \mathbf{U}_p^{\text{right}} = \begin{pmatrix} 0.08458825\,\mathrm{kg/m^3} \\ -1523.296\,\mathrm{m/s} \\ 310.7925\,\mathrm{K} \end{pmatrix}. \quad (4.6)$$

This can be seen in the zoom for RP-W2 in Figure 4.1. The left state is located right under the vapor saturation line. The intermediate states are located in the vapor phase. This means the states in the Riemann problem are crossing the vapor saturation line during calculation. This involves a phase change. From the left state L to the intermediate state $L_\star$ (over the left shock wave) evaporation occurs as well as over the right shock wave (from R to $R_\star$). The black solid line denotes the calculation with 50 elements and $N = 3$. As seen in the zoom for RP-W2 in Figure 4.1 the thermodynamic path has kinks at the locations of the shock waves ($x = 0.25\,\mathrm{m}$ and $x = 0.815\,\mathrm{m}$) which are not on the vapor saturation lines. By increasing the resolution by a factor of 5 the kinks are only located on the saturation line because the thermodynamic behavior changes since only phase change occur in the two-phase region. The dotted line shows the results with 200 elements and $N = 4$ in the zoom for RP-W2 in Figure 4.1. In Figure 4.3 the agreement of the numerical solution with 50 elements and $N = 3$ (black solid line) and the exact solution (dashed line) is compared. The contact discontinuity is smeared out because of the used Riemann solver. For the shock waves the match is very good. This shows that for computations with a high accurate equation of state, like it is used in this work, also the thermodynamic path needs to be checked for a better validation. The FV sub-cell method resolves all three discontinuities as seen in the density plot where the horizontal lines at the top represents the activation of the FV sub-cell method.

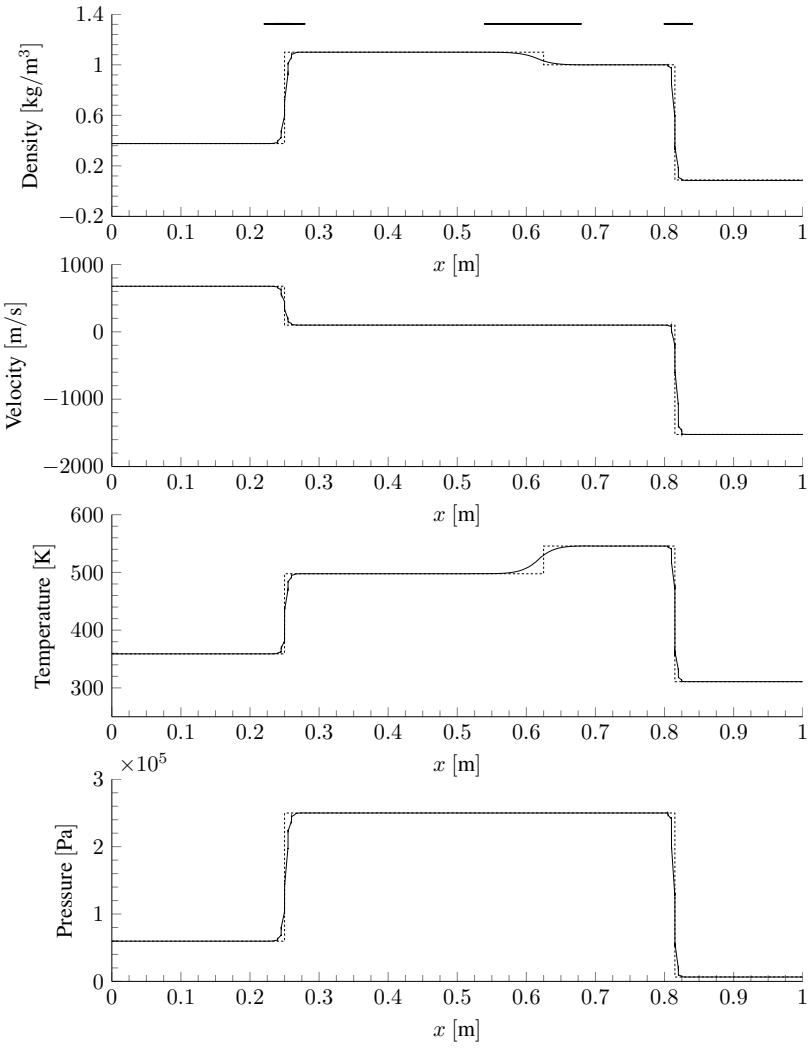

Figure 4.3: Riemann Problem RP-W2 at $t = 1.25 \times 10^{-3}$ s: comparison of the exact solution (dashed) with the numerical (solid). FV sub-cells are shown in density-plot

4.2.1.3 RP-W3

RP-W3 is again calculated in a tube with the length of $1\,\mathrm{m}$ and the initial discontinuity located at $x = 0.5\,\mathrm{m}$. The initial states are given by

$$\mathbf{U}_p^{\text{left}} = \begin{pmatrix} 976.0968\,\mathrm{kg/m^3} \\ -187.4091\,\mathrm{m/s} \\ 324.6175\,\mathrm{K} \end{pmatrix}, \qquad \mathbf{U}_p^{\text{right}} = \begin{pmatrix} 971.2215\,\mathrm{kg/m^3} \\ -208.485\,\mathrm{m/s} \\ 349.4697\,\mathrm{K} \end{pmatrix}, \tag{4.7}$$

again in primitive variables. The initial states are located in the two-phase region but the intermediate states are in the liquid area as seen in Figure 4.1 in the plot for RP-W3. In this test-case over the left shock wave (L to $L_\star$) condensation occurs as well as over the right shock wave (R to $R_\star$). For the phase change the pressure jumps from the vapor pressure to the pressure in the liquid phase. This is a jump over three orders of magnitude because the initial pressure is $p^{\text{left}} = 1.328 \times 10^4\,\mathrm{Pa}$ and $p^{\text{right}} = 4.078 \times 10^4\,\mathrm{Pa}$ whereas for the intermediate pressure $p_\star = 1 \times 10^7\,\mathrm{Pa}$ is reached. In Figure 4.4 the numerical and exact solutions are compared. The agreement is again reasonable. Some oscillations occur for the pressure. These are also seen in [17] but can be prevented there by using a different Riemann solver. The FV sub-cell method is active as seen in the density plot visualized by thicker horizontal lines.

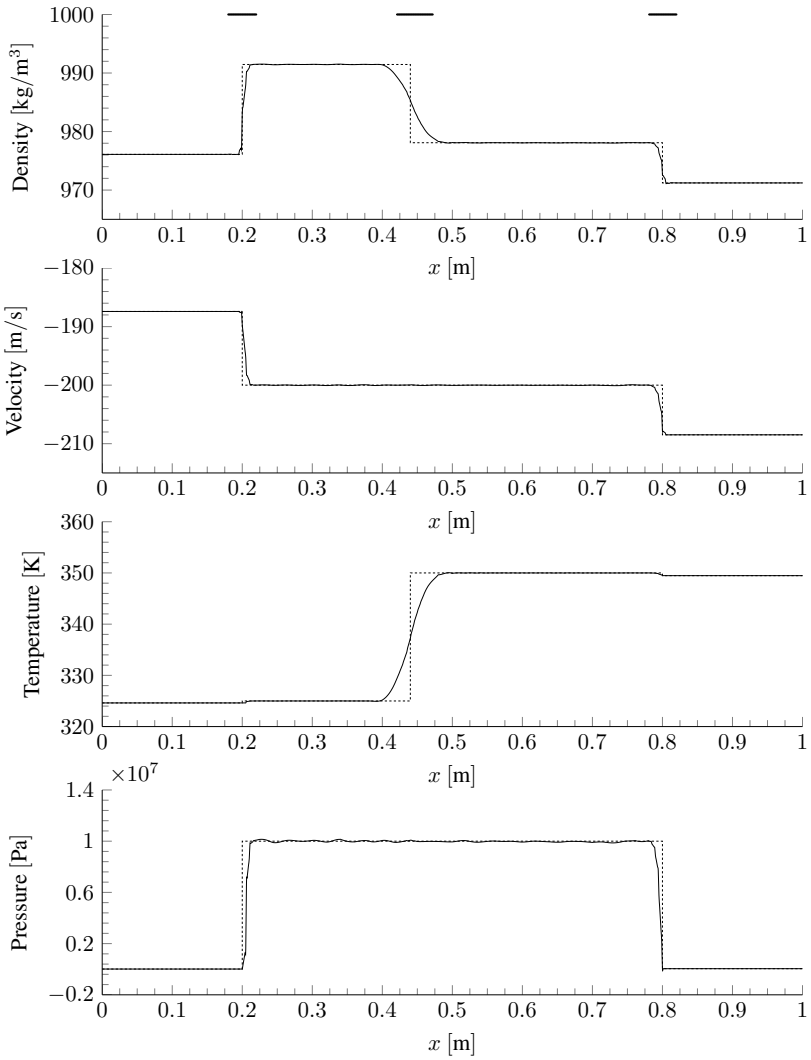

Figure 4.4: Riemann Problem RP-W3 at $t = 3 \times 10^{-4}$ s: comparison of the exact solution (dashed) with the numerical (solid). FV sub-cells are shown in density-plot

4.2.1.4 RP-W4

The initial states in a tube of length $1\,\mathrm{m}$ with the initial location of the discontinuity at $x = 0.7\,\mathrm{m}$ are given by

$$\mathbf{U}_p^{\text{left}} = \begin{pmatrix} 53.97074\,\mathrm{kg/m^3} \\ 3.63214\,\mathrm{m/s} \\ 331.9906\,\mathrm{K} \end{pmatrix} \qquad \mathbf{U}_p^{\text{right}} = \begin{pmatrix} 1.96013\,\mathrm{kg/m^3} \\ -76.32555\,\mathrm{m/s} \\ 315.4950\,\mathrm{K} \end{pmatrix}. \tag{4.8}$$

In this test-case all states are located in the two-phase region as seen in the bottom plot of Figure 4.1. In this Riemann problem the saturation lines are not crossed by the thermodynamic path but nevertheless phase change occurs in this example, because the vapor quality changes. The lowest vapor quality (χ) is reached at $R_\star$ with $\chi = 4.098 \times 10^{-5}$ and the highest at R with $\chi = 0.029$. This means condensation occurs from R to $R_\star$ and also from L to $L_\star$. In this test-case phase change occurs even over the contact discontinuity ($R_\star$ to $L_\star$). In Figure 4.5 two numerical solutions are compared with the exact one (dashed line). The blue solution represents a resolution of 200 DOF and the red one was calculated on 500 DOF with 100 elements and $N = 4$. This increase of resolution shows that the calculation converges to the exact one. To show also the effect of the FV sub-cell method on the solution, the area where it is active, seen as thick horizontal lines in the density plot, is reduced for the higher resolved calculation. This lower percentage of used FV sub-cells leads to higher oscillations in the density plot. In this case the higher resolution does not change the thermodynamic path significant and is not shown in Figure 4.1 because of that. In [17] also a calculation with the Rusanov Riemann solver is shown. Comparing the solutions it can be seen in the density plot of Figure 4.5 that they match well. The green line represents the calculation presented in [17].

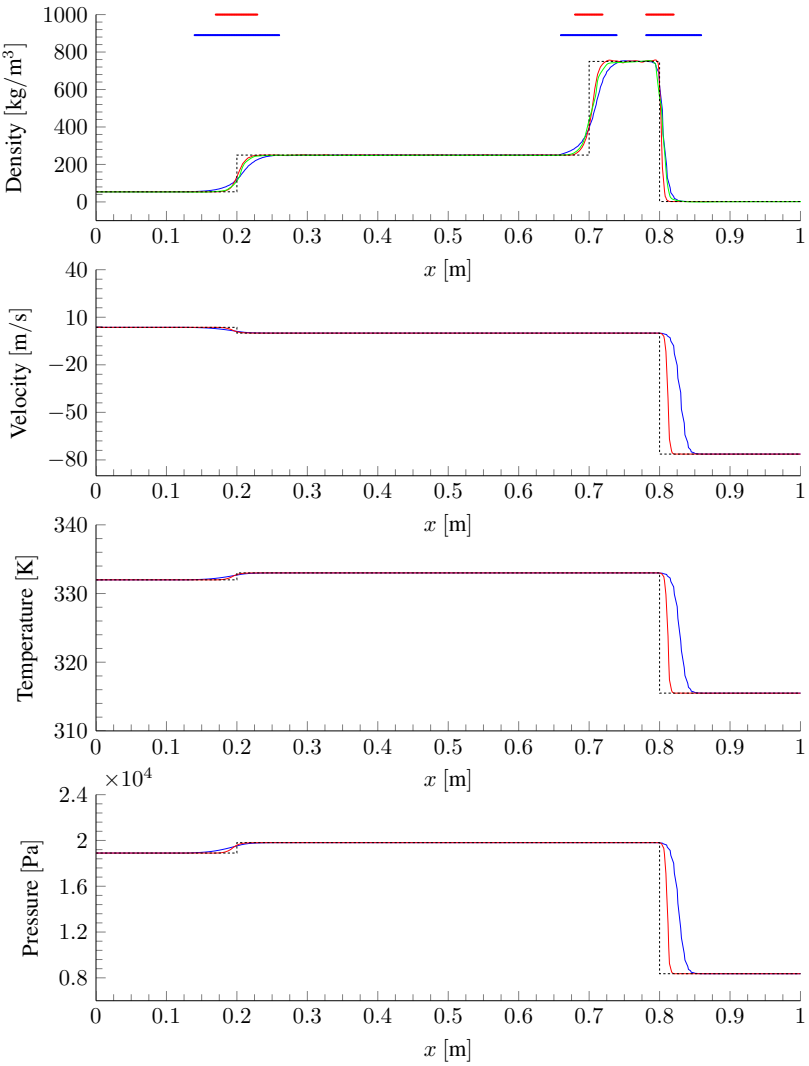

Figure 4.5: Riemann Problem RP-W4 at $t = 0.5\,\mathrm{s}$: comparison of the exact solution (dashed) with the numerical solution (blue = 200 DOF, red = 500 DOF). FV sub-cells and ADER-FV P_0P_2 Rusanov flux [17] are shown in density plot (green).

4.2.1.5 RP-W5

For RP-W5 the tube length is 2 m from -1 m to 1 m. The given states are

$$\mathbf{U}_p^{\text{left}} = \begin{pmatrix} 998.23739\,\text{kg/m}^3 \\ 100.0\,\text{m/s} \\ 293.0\,\text{K} \end{pmatrix}, \qquad \mathbf{U}_p^{\text{right}} = \begin{pmatrix} 996.55634\,\text{kg/m}^3 \\ 100.0\,\text{m/s} \\ 300.0\,\text{K} \end{pmatrix}. \quad (4.9)$$

The initial location of the discontinuity is $x = -0.5$ m. It demonstrates a isolated moving contact discontinuity and as seen in Figure 4.1, the initial states and the thermodynamic path are very close to the liquid saturation line but the values always remain in the liquid region which indicates that no phase change occurs. As mentioned in [57] the use of highly non-linear EOS, which is done in this work, can produce oscillation in pressure and velocity for moving contact discontinuities. As seen in Figure 4.6 this is true for this calculation. The oscillations are more noticeable at the location of the discontinuity for pressure and velocity but almost not visible for the velocity. For the pressure they are around 1 ‰ of the initialized data. Compared to the results in [17] the oscillations are around the initial state but the present work shows no jump in the velocity or pressure.

4.2.1.6 Computational Cost of the Quatree Evaluation

Table 4.5: Comparison of the PID for perfect gas and EOS-quadtree

	RP-W1	RP-W2	RP-W3	RP-W4	RP-W5
Perfect Gas [µs]	11.305	14.765	11.228	15.575	11.052
EOS-quadtree [µs]	39.089	50.049	36.713	49.222	36.764
Factor	3.46	3.39	3.27	3.16	3.33

Table 4.5 compares the average performance index (PID) for the perfect gas and EOS-quadtree approach over five runs. The PID is defined as

$$\text{PID} = \frac{\text{computation-time} \times \#\text{cores}}{\#\text{DOF} \times \#\text{time-steps}}. \quad (4.10)$$

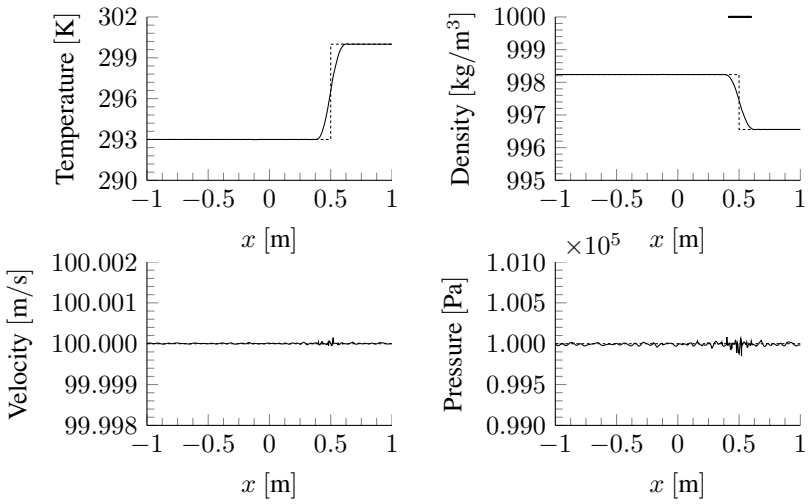

Figure 4.6: Riemann Problem RP-W5 at $t = 1 \times 10^{-2}$ s: comparison of the exact solution (dashed) with the numerical solution (solid). FV sub-cells are shown in density-plot

The PID shows the time needed to update one DOF to the next physical time level. In this comparison the quadtree evaluation is exchanged with the direct analytically evaluation of the perfect gas EOS. Of course the perfect gas EOS cannot solve the presented Riemann problems for water but in this case only the computational cost is of interest. If the used DG method with FV sub-cell approach is optimized for the perfect gas EOS the lowest PID ranges from $5\,\mu s$ to $6\,\mu s$ [62] measured also on the Cray XC40 of the HLRS. This means that compared to an optimized code the EOS-quadtree approach is around a factor 6 to 10 slower. But this comparison is inappropriate since the optimized code is only able to solve the perfect gas equation whereas the code presented in this work is able to change the EOS easily. This capability produces a factor around two overhead which can be seen if the PIDs of both implementations of the perfect gas EOS are compared.

4.2.1.7 Comparison of Different Riemann Solvers

In Section 2.5 two Riemann solvers are described which can work with a highly accurate EOS. Both are compared with the RP-W1 and RP-W2 in this work. The amount of FV sub-cells are the same for both Riemann solver simulations. In Figure 4.7 the solution for RP-W1 and the density are shown. The LLF produce a good result but the HLLC resolves the contact discontinuity much sharper. In Figure 4.8 the solution for RP-W2 is shown and again the

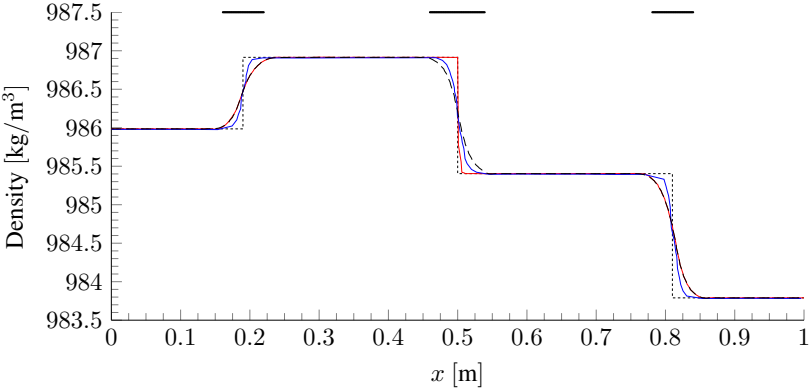

Figure 4.7: RP-W1 ($t = 2 \times 10^{-4}$ s) with LLF (long dashed,black), ADER-FV P_0P_2 FORCE flux [17] (blue) and HLLC (red) Riemann solvers. Exact solution is dashed. 50 elements were used with $N = 3$. FV sub-cells are shown in density-plot.

density is plotted. Here again the LLF produces a good result but the shock waves as well as the contact discontinuity are better approximated to the exact solution by using the HLLC Riemann solver. The HLLC can handle all five Riemann problems described in the last subsections but for the RP-W4 the time-step needs to be reduced by a factor of 0.4 compared to the LLF calculation otherwise the calculation is unstable. But despite this reduced time step the solution for RP-W4 and the HLLC Riemann solver is still oscillating and differs from the exact solution. In total the solution quality gets better by using

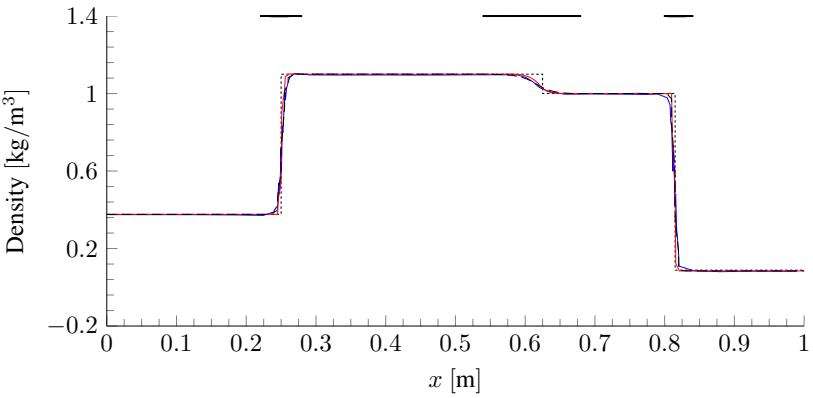

Figure 4.8: RP-W2 ($t = 1.25 \times 10^{-3}$ s) with LLF (long dashed,black), ADER-FV P_0P_2 FORCE flux [17] (blue) and HLLC (red) Riemann solvers. Exact solution is dashed. 50 elements were used with $N = 3$. FV sub-cells are shown in density-plot.

the HLLC Riemann solver but it tends to more oscillations and is not as robust as the LLF Riemann solver. For a robust calculation the LLF should be used but for higher solution quality the HLLC is the superior choice. To compare the computational cost between the LLF and HLLC Riemann solver all five Riemann problems are computed. The average PID over five runs is compared in Table 4.6. The increase of computational cost is due to the HLLC needing to evaluate the pressure for the left and right state as described in Section 2.5.

Table 4.6: Comparison of the PID for the LLF and HLLC Riemann solver

	RP-W1	RP-W2	RP-W3	RP-W4	RP-W5
LLF [µs]	39.089	50.049	36.713	49.222	36.764
HLLC [µs]	44.502	62.224	44.694	64.245	48.456
Factor (HLLC/LLF)	1.14	1.24	1.22	1.31	1.32

4.2.1.8 Influence of the Indicator-Limit

As described in Section 2.4 the switching between the DGSEM and the FV sub-cells approach depends on the chosen indicator-limit. This means that the FV sub-cells used during a simulation can be adjusted. As an example RP-W1 with the LLF Riemann solver is chosen. In Figure 4.9 three simulations are plotted with different indicator-limits. The black line is the same solution as seen in Figure 4.2. The red line is the result of RP-W1 with the indicator value adjusted so that just no oscillations occur and the blue line in Figure 4.9 represents the calculation for the pure DGSEM without shock capturing. The bars at the top of the plot represent the elements where the FV sub-cell approach is active at this specific time $t = 2 \times 10^{-4}\,\mathrm{s}$. By adjusting the indicator value the solution quality is increased until a certain value when the oscillations starts. If the results are compare with the exact solution (dashed line) in Figure 4.9 the result with the perfect adjusted indicator-limit resolves all three discontinuities much sharper than the one with less adjustment. The pure DGSEM solution resolves the discontinuities even sharper but the solution is not oscillation free. This means that not only the Riemann solver can decrease the solution quality but also a poorly adjusted indicator value. But the perfect value is hard to find and for more complex simulations also varies over time.

4.2.2 Strong Rarefaction

A different validation calculation is presented in this subsection. The initial states are given as

$$\mathbf{U}_p^{\text{left}} = \begin{pmatrix} 997.0854\,\mathrm{kg/m^3} \\ -10\,\mathrm{m/s} \\ 298.0\,\mathrm{K} \end{pmatrix}, \qquad \mathbf{U}_p^{\text{right}} = \begin{pmatrix} 997.0854\,\mathrm{kg/m^3} \\ 10\,\mathrm{m/s} \\ 298.0\,\mathrm{K} \end{pmatrix}, \quad (4.11)$$

and the initial position of this discontinuity is $x = 0\,\mathrm{m}$ in a tube from $-1\,\mathrm{m}$ to $1\,\mathrm{m}$. The initial states have the same thermodynamic state but the velocities are pointing away from each other in the outward direction of the tube which leads to a strong rarefaction. This causes evaporation since the pressure drops to the vapor pressure. Due to a significant reduction in the speed of sound the evaporation wave moves very slowly and far behind the rarefaction wave.

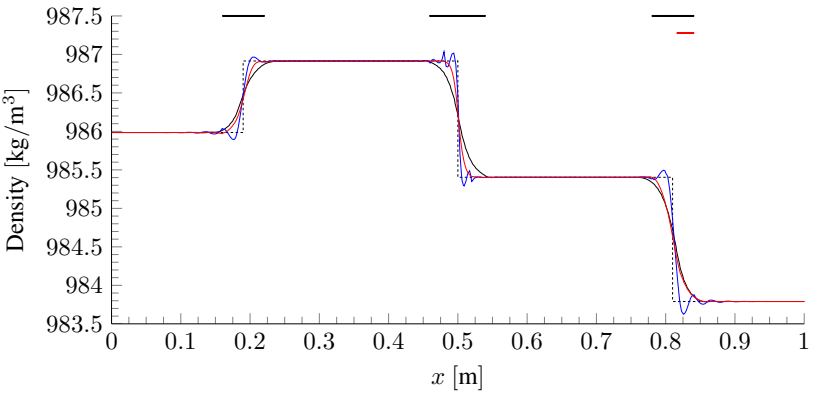

Figure 4.9: RP-W1 with different indicator values. More FV sub-cells are active for the black line whereas for the red line these are reduced to the necessary minimum. The blue line shows the solution without shock-capturing.The dashed line represents the exact solution

Here 2500 elements and a polynomial degree of 3 is used with the LLF Riemann solver. As seen in Figure 4.10 due to the opposite velocity directions the water reaches the two-phase area. The sound speed drops by several orders of magnitude which is normal for multiphase problems and this drop must be captured with the FV sub-cell approach which is again plotted in the density graph of Figure 4.10. Also a zoom is shown in the density graph which visualizes the rarefaction wave. The bottom plot shows the solution in the two-phase region. The initial states ($x = -1\,\text{m}$ and $x = 1\,\text{m}$) are in the liquid region and the density drops almost to the vapor saturation line ($x = 0\,\text{m}$). The results are in excellent agreement with those presented in [17] where the Rusanov Riemann solver is used.

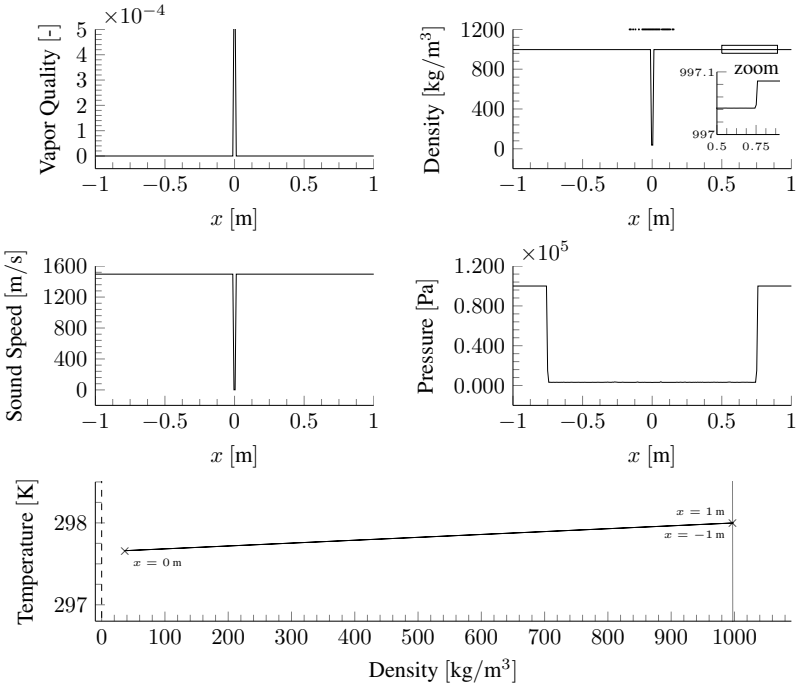

Figure 4.10: 1D evaporation by strong rarefaction at $t = 5 \times 10^{-4}$ s

4.2.3 Strong Heating

In this test-case the tube is filled completely with the initial condition

$$\mathbf{U}_p = \begin{pmatrix} 998.2374\,\mathrm{kg/m^3} \\ 0.0\,\mathrm{m/s} \\ 293.0\,\mathrm{K} \end{pmatrix}. \tag{4.12}$$

For the calculation the source term $\mathbf{P}$ is not zero (see Section 2.1). A heat source is considered which is active between $x_L = 0.45\,\mathrm{m}$ and $x_R = 0.55\,\mathrm{m}$ in a tube of length 1 m. The heat energy is $q = 5 \times 10^{12}\,\mathrm{J}$ and is applied to the

area from the beginning of the calculation for $\Delta t^{\text{heat}} = 6 \times 10^{-5}$ s. Due to the heating the temperature and the pressure rise. When the heating stops the high pressure region collapses and the pressure falls below the vapor pressure and evaporation occurs. The duration of the total simulation is $t_{\text{end}} = 2 \times 10^{-4}$ s. The results are shown in Figure 4.11. The temperature plot shows the rise of the temperature due to the heating. In the pressure plot the dotted line indicates the vapor pressure for the temperature at the corresponding x location in the tube. When the vapor pressure is reached the water starts to evaporate and then the two-phase region is reached as seen in the phase diagram. Since it is a symmetric problem the thermodynamic state for $x = 0\,\text{m}$ and $x = 1\,\text{m}$ are the same as well as the state for $x = 0.23\,\text{m}$ and $x = 0.77\,\text{m}$. In the density plot the location where the FV sub-cell method is active is also shown. In this calculation 250 elements and a polynomial degree of $N = 3$ with the LLF Riemann solver is used. The match with the calculation in [17] is very good.

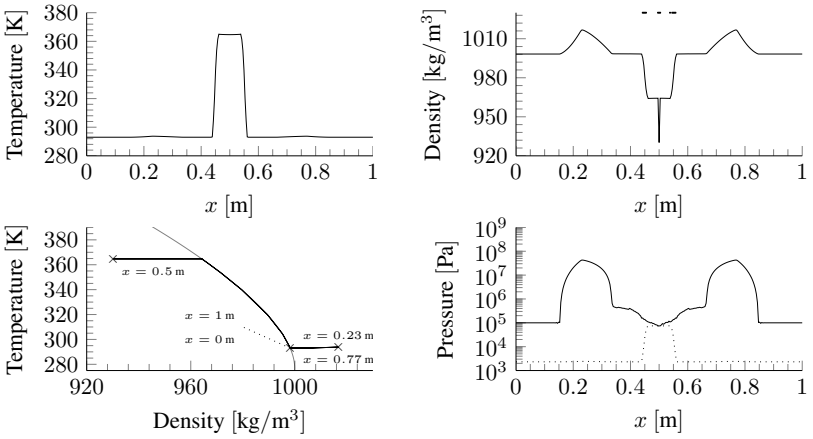

Figure 4.11: Strong heating at $t = 2 \times 10^{-4}$ s

4.2.4 Shock Condensation

This test-case is taken from [56] and it calculates a shock condensation. The given states are

$$\mathbf{U}_p^{\text{left}} = \begin{pmatrix} 1200.0\,\text{kg/m}^3 \\ 0.0\,\text{m/s} \\ 300.0\,\text{K} \end{pmatrix}, \qquad \mathbf{U}_p^{\text{right}} = \begin{pmatrix} 500.0\,\text{kg/m}^3 \\ 0.0\,\text{m/s} \\ 300.0\,\text{K} \end{pmatrix}. \tag{4.13}$$

The left state is filled with liquid and the right with liquid-vapor mixture with a vapor volume fraction of $\alpha = 0.5$. The vapor volume fraction can be calculated with the vapor quality

$$\alpha = \left. \frac{\rho_{\text{liq}} - \rho_\chi}{\rho_{\text{liq}} - \rho_{\text{vap}}} \right|_T . \tag{4.14}$$

Here ρ_{liq} and ρ_{vap} are the liquid and vapor saturation density for a given temperature and ρ_χ is the density for a specific vapor quality at the same temperature. A slight adjustment to the initial discontinuity location has to be made because of the different EOS used in both methods. In [56] a barotropic EOS is used which approximates the sound speed differently as the one used in this work. Due to different sound speed assumptions in this case the initial discontinuity is located at $x = 0.64\,\text{m}$. With this a very good result is reached compared to the original results in [56]. Figure 4.12 shows the obtained result with a resolution of 50 elements and $N = 3$. The thermodynamic path is only plotted at $t = 150 \times 10^{-6}\,\text{s}$. Condensation occurs over the shock wave which can be seen with the help of the thermodynamic path plot. In this case the LLF Riemann solver is used again.

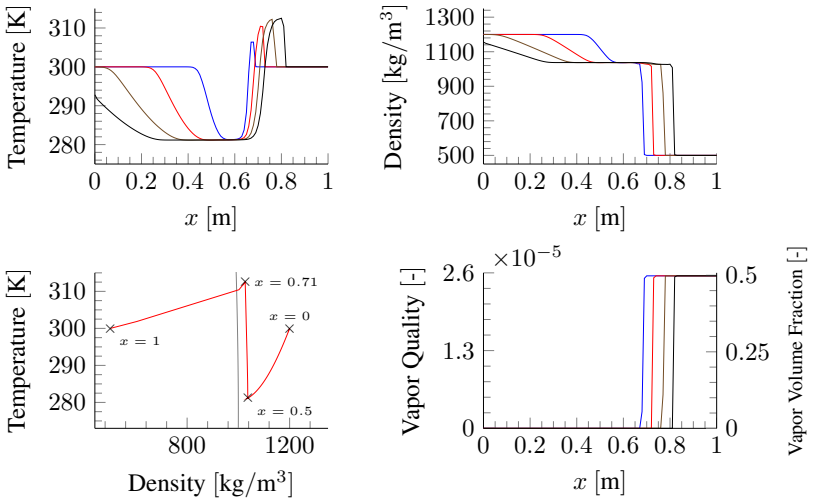

Figure 4.12: Condensation over time. 75 µs (blue), 150 µs (red), 225 µs (brown) and 300 µs (black)

4.3 2D Calculations: Hydro-Foil

For a 2D test case a NACA-0012 airfoil is used as a hydrofoil. It is surrounded by water at a temperature of 330 K and a pressure of $p_{\text{in}} = 5 \times 10^5$ Pa. This leads to a density of $\rho = 984.9608\,\text{kg/m}^3$. The angle of attack is $8°$ and the cavitation number

$$\sigma = \frac{p_{\text{in}} - p_{\text{sat}}}{2\rho\,|\vec{v}|^2} = 1. \tag{4.15}$$

The saturation pressure at the given temperature is $p_{\text{sat}}(330\,\text{K}) = 17213.15\,\text{Pa}$ and the norm of the free-stream velocity vector is $|\vec{v}| = 15.655\,\text{m/s}$. A grid with 12 194 elements is used with a polynomial degree of $N = 4$. This leads to ca. 0.3 million DOF. The simulation runs on 240 cores and the LLF Riemann solver is used. The chord length is 0.1334 m and the domain has a radius of one meter. This simulation is a show-case to demonstrate the capability of the code to run such simulations with DG and FV coupling. Figure 4.13

shows the cavitation region which develops from the leading edge of the foil. The white area shows the elements which are using the FV method. The rest of the computational domain uses the DGSEM at this particular time frame. Since a indicator is used to switch between these methods the distribution of the FV sub-cell elements changes for every time step. In Figure 4.13 several

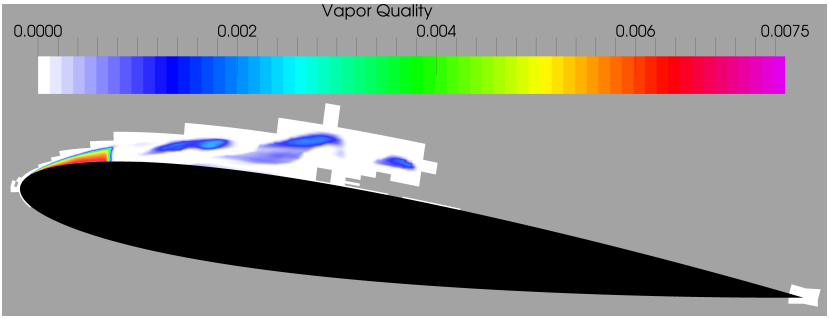

Figure 4.13: Hydro-Foil Naca-0012 at $t = 0.03423\,\mathrm{s}$. The white area highlights where the FV sub-cell method is active. The coloring visualizes the vapor quality

areas with cavitation are visible. The color represents the vapor quality. The highest amount of vapor is at the leading edge of the hydrofoil. To demonstrate the collapse behavior two additional pictures at a later time are shown in Figure 4.14. The right most cavitation area in Figure 4.13 has collapsed in the right picture of Figure 4.14. The pressure distribution is colored in Figure 4.14 and it can be seen that a very low pressure region is located at the leading edge of the hydrofoil. If a cavitation area leaves this region and is transported into a higher pressure region, this area collapses. The left picture in Figure 4.14 shows the pressure distribution before the collapse. The collapse takes place very close to $t = 0.0342379\,\mathrm{s}$ with a pressure peak of $p_{\mathrm{peak}} = 4.247 \times 10^6\,\mathrm{Pa}$ and a temperature of $T_{\mathrm{peak}} = 330.26\,\mathrm{K}$ which leads to a density of $\rho_{\mathrm{peak}} = 986.46\,\mathrm{kg/m^3}$. The right picture in that figure shows the time shortly after the collapse. The pressure wave generated from the collapse moves through the water. In this calculation the water is liquid or in the two-phase state but never reaches the vapor state. This calculation proves that

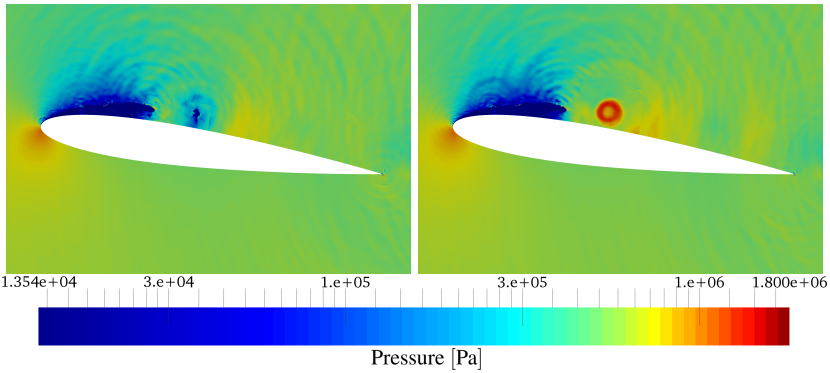

Figure 4.14: Cavitation: One cavitation bubble collapses. The pressure distribution is plotted. Left picture is at $t = 0.03473\,\mathrm{s}$ and the right picture at $t = 0.03474\,\mathrm{s}$

the setup is very well suited for resolving cavitational flows. The average time step is $\Delta t = 1.65 \times 10^{-8}\,\mathrm{s}$ and the average PID is $42.888\,\mu\mathrm{s}$. For a calculation duration of $t = 0.03474\,\mathrm{s}$ around 2.1 million time-steps are needed which leads to a overall calculation cost of 7 645 CPUh for this specific duration.

4.4 3D Calculation

With the described method in Chapter 2 and Chapter 3 an industrial application is calculated. This application is a 3D calculation for a micro-channel flow with water. This micro-channel is a test geometry for throttle valves which can be found in fuel injection systems. The geometry can be seen in Figure 4.15 and is used to investigate cavitation inside the throttle. The length of the throttle is $l = 10^{-3}\,\mathrm{m}$ and the height and the width are both $h = w = 3 \times 10^{-4}\,\mathrm{m}$. The beginning of the throttle has a radius of $r = 4 \times 10^{-5}\,\mathrm{m}$. The aim of this calculation is to predict cavitation inside engine injections and to help in the design of engines with reduced cavitation. In [18] this geometry was used with a barotropic EOS and the energy function was neglected since for the

barotropic EOS the pressure is only a function of the density. In the present work the full NSE are solved with the explained quadtree EOS approach. The throttle is filled with liquid water at $p = 10^7$ Pa and $T = 330$ K also the inlet pressure is higher than the outlet pressure. Due to this pressure gradient the water streams through the throttle and inside the throttle it starts to cavitate. The inlet is pressurized with $p_{\text{in}} = 3 \times 10^7$ Pa and a temperature of $T_{\text{in}} = 330$ K. The outlet has a pressure of $p_{\text{out}} = 10^7$ Pa and the same temperature as the inlet. Due to the acceleration of the fluid the pressure is reduced. Inside the throttle cavitation occurs because of this pressure drop. Further upstream the two-phase area collapses due to the higher pressure around it. In this work the scaling of a high-performance computer is shown and the benefit of the DG and FV coupling. The analysis of this calculation and comparison with other results is done in the PhD thesis of Fabian Hempert [27] who also kindly provided the mesh and the restart file needed for the calculations in this section.

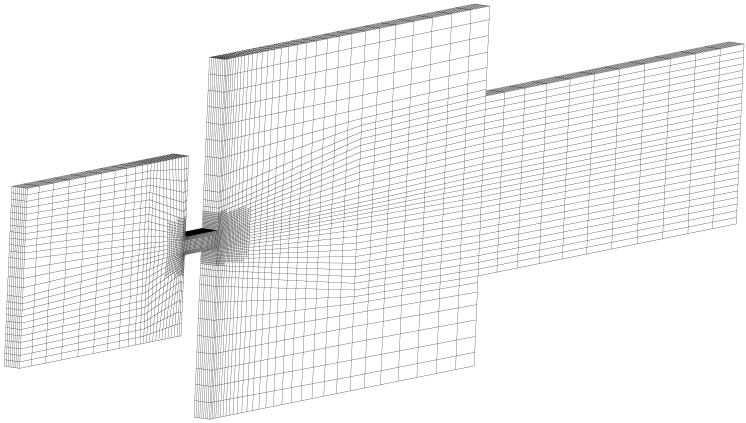

Figure 4.15: Mesh for the throttle flow with 46 592 elements

4.4.1 Scaling

To show the scaling the above described industrial application is used. The mesh for this calculation can be seen in Figure 4.15. As seen in this figure non-conforming elements are allowed. The implementation of these is explained in [61]. The mesh has a total of 46 592 elements which leads to almost 3 million DOF for $N = 3$ and 10 million DOF for $N = 5$. These two polynomial degrees are both used for the scaling test. For this test a operating point is needed where cavitation occurs in the throttle. This can be seen in Figure 4.16. Keep in mind that in this picture only one slice in the middle plane of the throttle is shown but the water cavitates over the complete depth of the throttle. An operation point near the saturation lines is chosen on purpose, because there the EOS-quadtree has the highest refinement level and the computational cost for its evaluation is maximal. Also in this test the coupling of the DG and FV scheme is active. All boundary conditions are full-slip walls, except for the inlet and outlet boundary conditions which are of the Dirichlet type. The LLF Riemann solver is used in this calculation.

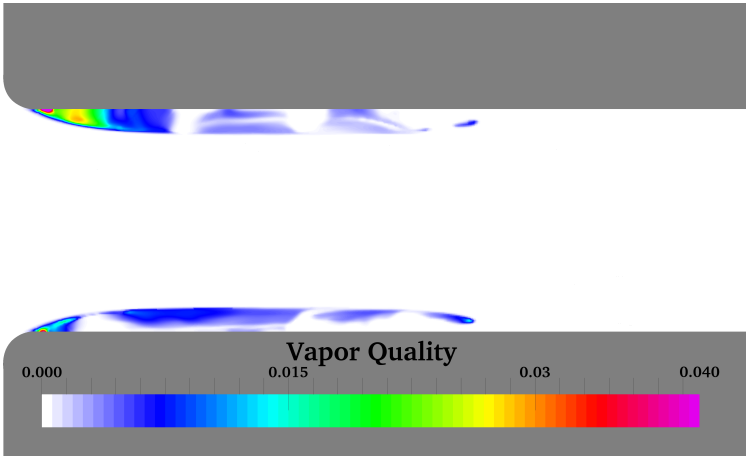

Figure 4.16: Cavitation inside the throttle at t=7.5 $\times$ 10^{-5} s

For the scaling the amount of cores ranges from 24 up to 6 144. With each run the number of used cores are doubled. The speed up of all calculations is calculated relative to the 24 core run. 100 time-steps are evaluated at each run. The average PID ranges from 39.1 µs to 58.8 µs as seen in Table 4.7. Also the average calculation time is shown in that table. By increasing the number of cores this time can be reduced drastically. The scaling can be seen in Figure

Table 4.7: Comparison of the PID and calculation time for different number of cores

	$N = 3$		$N = 5$	
# cores	24	6 144	24	6 144
PID $[\mu s]$	41.0	58.8	39.1	58.7
Calc. time $[s]$	509.4	2.85	1 639.2	9.615

4.17. The minimum load per core are 7 to 8 elements which leads to 448 DOF ($N = 3$) and 1 512 DOF ($N = 5$) per core at a core number of 6 144. The lowest scaling compared to the 24 core run is still at 70 % which is normal with the DG and FV coupling, see [62]. The decrease of the ideal speed-up is

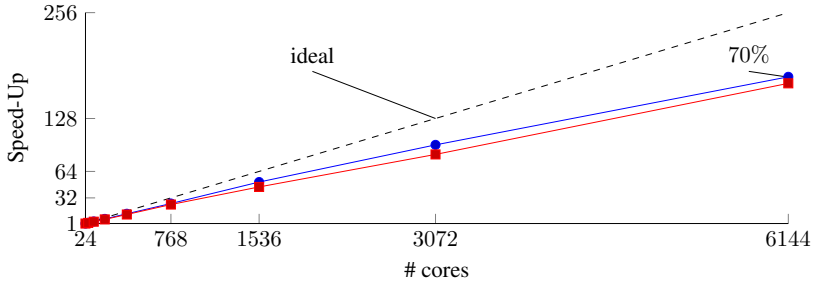

Figure 4.17: Scaling with $N = 3$ (blue) and $N = 5$ (red) from 24 to 6 144 cores

due to a load-imbalance between the cores. There are several reasons for this:

- the DG elements cannot be distributed evenly on the cores.
- the FV sub-cells are more computation-time consuming than the DG elements by a factor of around 1.5 [62].
- the table approach also adds an imbalance because the time to evaluate the quantities depends on the level and polynomial degree of the evaluated table element.
- an DG element which has a boundary condition face needs slightly more time than an element without it.

If we compare this scaling to a pure DGSEM solver [4] with ideal gas EOS or the DG method with FV sub-cell approach with the ideal gas EOS [62] it has to be mentioned that in this work no scaling above 100 % (super-scaling) is achieved. In both of those works super-scaling is reached due to cache effects but this is not possible in the current implementation with the quadtree EOS because the amount of data needed with the quadtree EOS does not fit in the cache of a core.

4.4.2 High Resolved Simulation

For this calculation 630 448 elements with $N = 4$ are used with leads to 78 806 000 DOF. For this calculation the HLLC Riemann solver is used. Figure 4.18 shows results at $t = 5 \times 10^{-5}$ s in the center plane of the throttle. The picture in the middle of this figure illustrates the DGSEM with the FV sub-cell approach. Fine structures and waves due to the collapse of cavitation areas are visible. The bottom picture shows the elements where the FV sub-cell approach is active. It can be seen that the used indicator captures the important cavitation areas. In the bottom picture of Figure 4.18 it is also visible that the FV sub-cell approach is also active in elements where no cavitation occurs. This can be reduced by adjusting the indicator-limit but not avoided completely. Since this is the first high resolved simulation with this implementation, the indicator-limit is not perfectly adjusted. The top picture of Figure 4.18 shows a result where all elements where forced to use the FV sub-cell approach. For ca. 7 000 time steps the pure FV method was solved from $t = 4.9 \times 10^{-5}$ s to $t = 5 \times 10^{-5}$ s. As seen in the top picture the fine

structures and waves cannot be resolved by the pure FV method. Even for this short time period the solution quality decreases significantly and the benefit of coupling the DGSEM with the FV sub-cell approach can be seen. The time step for the DGSEM calculation is around $\Delta t = 9.9 \times 10^{-11}$ s and the PID is at 69 µs for a calculation on 8 400 cores. This leads to a computational cost around 765 000 CPUh for the mentioned physical time of $\Delta t = 5 \times 10^{-5}$ s.

4.5 Results summary

The coupling of the DGSEM and the FV sub-cell approach as well as the implemented EOS-quadtree is validated with convergence tests and a variety of 1D calculations. The results are in good agreement with the literature. The two investigated Riemann solvers can be used for the shown simulations but the choice between robustness and solution quality has to be made. By using the indicator by Persson and Peraire for the density as indicator variable the calculations can be stabilized. The indicator-limit has a significant influence on the solution quality. Hence, it must be very finely tuned. The 2D and 3D calculations are showing the ability of the approach to solve complex multiphase simulations with cavitation. The scaling on HPC clusters is good and due to this, highly resolved 3D calculations can be performed in a reasonable amount of computational time.

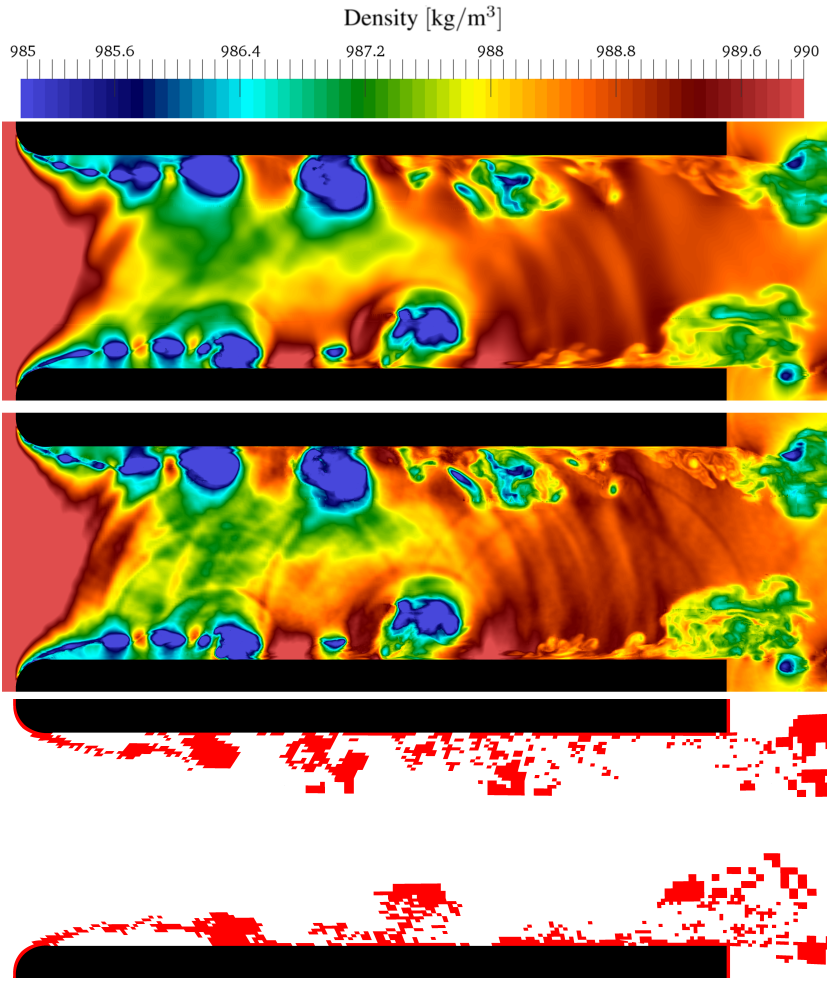

Figure 4.18: Cavitation inside the throttle. Top picture: pure FV methode. Middle picture: DGSEM with FV sub-cell approach. Bottom picture: elements where the FV sub-cell approach is active. All three pictures at $t = 5 \times 10^{-5}$ s

5 Conclusion and Prospects

5.1 Conclusion

The aim of this work was to present an efficient and accurate numerical method for two-phase flow simulations in which cavitation, shock condensation and evaporation occur. This method is based on a high-order DGSEM discretization of the compressible NSE with a complex EOS. We have chosen the DGSEM because of its low numerical dissipation and its very good scalability on high performance computers. To take effects like cavitation into account, the evaluation of the most-accurate Helmholtz free energy formulation as equation of state is provided by the CoolProp library. Thermodynamic equilibrium is assumed between the phases. We have chosen CoolProp because it provides very accurate data for more than 100 fluids which can be used in CFD simulations and it is an open-source library. We have investigated that the computational cost for directly evaluating the CoolProp library is not feasible in the context of a CFD simulation. To reduce this computational cost, we have stored the data of the library to hard-disk in a polynomial representation on a quadtree domain. The building of a quadtree is fully parallelized for an arbitrary number of cores. We have implemented a fast quadtree evaluation algorithm and with this the stored data can be used during simulations. By this the evaluation time has been reduced by a factor up to 6500 compared to the direct evaluation. Single phase calculations can be performed without any special treatment to the DGSEM but when phase change is involved high gradients occur, especially in the pressure and speed of sound variables. We have captured these gradients and we also have been able to resolve shocks by using a 2^{nd} order finite volume scheme, which we have implemented in a way that the spatial operator can be calculated either by the DGSEM or FV scheme, depending on the value of an indicator. For both methods the memory layout and the parallelization technique have been kept unchanged which leads

to an efficient shock capturing approach on HPC clusters, because the communication hiding technique implemented in the DGSEM could be used in a straightforward manner. We have found out that the shock capturing shows a strong sensitivity to the indicator. To find a good choice, where the simulation is still stable and the solution quality decent, some a priori parameter studies are needed. Since cavitating flows need a very fine time resolution to resolve the collapses of a cavitation area accurately, we have implemented an explicit time stepping, which shows good results in resolving the time scales of cavitating flows.

We have validated the described method with convergence tests and 1D test-cases. The results are in very good agreement compared to other publications and these results show that shock condensation and evaporation can be well resolved with this method. The two described Riemann solvers can both be used to solve cavitating flows. We have found out, that the HLLC Riemann solver is less dissipative but the LF Riemann solver is more robust for two-phase flow calculations. As an example for cavitating flows we have simulated a 2D hydrofoil in water and the results show that cavitation occurs and collapses as postulated. With an industrial 3D application we have verified the good performance on state of the art high performance computers and also we have demonstrated the benefit of the high-order approach. With these results the method shows the capability to simulate highly resolved two-phase applications correctly in a reasonable amount of time.

In conclusion we have introduced a very scalable high order CFD solver in this work which can handle two-phase phenomena like e.g. cavitation for over 100 fluids due to the easily exchangeable equation of state. The performance is promising and the achieved results are in good comparison with the literature.

5.2 Prospects

In the long term, the aim is to use this method for more complex industrial applications like fuel injection systems and hydraulic pumps where cavitation occurs. The method has been shown to be suitable for two-phase calculations, however, some further improvements are necessary, e.g. a better approximation of the physics and performance optimization if more complex calculations are simulated. In this context the following aspects could be considered:

- The presented Riemann solvers are not especially adapted for two-phase flows. This could be done following [16] by implementing a more accurate HLLEM Riemann solver.
- In this work only walls without friction are considered. For a better comparison with experiments isothermal and adiabatic wall boundary conditions can be implemented. This is a challenging task since besides the vapor and liquid phase also the two-phase region needs to be considered for these boundary conditions.
- To compare the simulations with experiments the dissolved gas in liquids needs to be considered. The outgassing can not be prevented in experiments since a small amount of gas is always dissolved in the liquid.
- Since the coupling between DGSEM and FV method as well as the quadtree approach are introducing load imbalances between the CPUs, a load balancing algorithm is needed to improve the scalability of the approach. During the simulation the load for each CPU core could be calculated and the elements could be distributed in a way that every core has the same load. Since cavitation for example is very unsteady, an efficient load balancing algorithm is needed which can be applied several times during calculation.
- The quadtree approach could be improved by building thermodynamic consistent quadtrees which are not rectangular by default. This could save building time and memory during calculation.
- A Large eddy simulations model could improve the calculation. Following [8] the application of an implicit model is straightforward.

Bibliography

[1] R. Akasaka. “A Reliable and Useful Method to Determine the Saturation State from Helmholtz Energy Equations of State.” In: *Journal of Thermal Science and Technology*, 3.3, pp. 442–451, 2008.

[2] C. Altmann, A. D. Beck, F. Hindenlang, M. Staudenmaier, G. J. Gassner, and C.-D. Munz. “An Efficient High Performance Parallelization of a Discontinuous Galerkin Spectral Element Method.” In: *Facing the Multicore-Challenge III: Aspects of New Paradigms and Technologies in Parallel Computing*, pp. 37–47, 2013.

[3] E. Anderson, Z. Bai, J. Dongarra, A. Greenbaum, A. McKenney, J. Du Croz, S. Hammerling, J. Demmel, C. Bischof, and D. Sorensen. “LAPACK: A Portable Linear Algebra Library for High-performance Computers.” In: *Proceedings of the 1990 ACM/IEEE Conference on Supercomputing*, pp. 2–11, 1990.

[4] M. Atak, A. Beck, T. Bolemann, D. Flad, H. Frank, and C.-D. Munz. “High Fidelity Scale-Resolving Computational Fluid Dynamics Using the High Order Discontinuous Galerkin Spectral Element Method.” In: *High Performance Computing in Science and Engineering´15*, pp. 511–530, 2016.

[5] F. Bassi and S. Rebay. “A High-Order Accurate Discontinuous Finite Element Method for the Numerical Solution of the Compressible Navier-Stokes Equations.” In: *Journal of Computational Physics*, 131.2, pp. 267–279, 1997.

[6] C. E. Baumann and J. T. Oden. “A discontinuous hp finite element method for the Euler and Navier-Stokes equations.” In: *International Journal for Numerical Methods in Fluids*, 31, pp. 79–95, 1999.

[7] A. Beck. “High order discontinuous Galerkin methods for the simulation of multiscale problems.” PhD thesis. University of Stuttgart, Germany, 2015.

[8] A. D. Beck, T. Bolemann, D. Flad, H. Frank, G. J. Gassner, F. Hindenlang, and C.-D. Munz. “High-order discontinuous Galerkin spectral element methods for transitional and turbulent flow simulations.” In: *International Journal for Numerical Methods in Fluids*, 76.8, pp. 522–548, 2014.

[9] I. H. Bell, J. Wronski, S. Quoilin, and V. Lemort. “Pure and Pseudo-pure Fluid Thermophysical Property Evaluation and the Open-Source Thermophysical Property Library CoolProp.” In: *Industrial & Engineering Chemistry Research*, 53.6, pp. 2498–2508, 2014.

[10] A. Burbeau, P. Sagaut, and C.-H. Bruneau. “A Problem-Independent Limiter for High-Order Runge-Kutta Discontinuous Galerkin Methods.” In: *Journal of Computational Physics*, 169.1, pp. 111–150, 2001.

[11] J. Carlton. *Marine propellers and propulsion*. 2011.

[12] M. Carpenter and C. Kennedy. “Fourth-order 2N-storage Runge–Kutta schemes, NASA TM-109112.” In: *National Aeronautics and Space Administration, Langley Research Center, Hampton, VA*, 1994.

[13] B. Cockburn and C.-W. Shu. “Runge-Kutta Discontinuous Galerkin Methods for Convection-Dominated Problems.” In: *Journal of Scientific Computing*, 16.3, pp. 173–261, 2001.

[14] S. Davis. “Simplified second-order Godunov-type methods.” In: *SIAM Journal on Scientific and Statistical Computing*, 9.3, pp. 445–473, 1988.

[15] N. Dittakavi, A. Chunekar, and S. Frankel. “Large eddy simulation of turbulent-cavitation interactions in a Venturi nozzle.” In: *Journal of Fluids Engineering*, 132.12, 2010.

[16] M. Dumbser and D. S. Balsara. “A new efficient formulation of the HLLEM Riemann solver for general conservative and non-conservative hyperbolic systems.” In: *Journal of Computational Physics*, 304, pp. 275–319, 2016.

[17] M. Dumbser, U. Iben, and C.-D. Munz. “Efficient implementation of high order unstructured WENO schemes for cavitating flows.” In: *Computers & Fluids*, 86, pp. 141–168, 2013.

[18] C. P. Egerer, S. Hickel, S. J. Schmidt, and N. A. Adams. “Large-eddy simulation of turbulent cavitating flow in a micro channel.” In: *Physics of Fluids (1994-present)*, 26.8, 2014.

[19] S. D. Fechter. “Compressible multi-phase simulations at extreme conditions using a discontinuous Galerkin scheme.” PhD thesis. University of Stuttgart, Germany, 2015.

[20] R. P. Fedkiw, T. Aslam, B. Merriman, and S. Osher. “A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method).” In: *J. Comput. Phys.* 152.2, pp. 457–492, 1999.

[21] D. Flad, A. D. Beck, G. Gassner, and C.-d. Munz. “A Discontinuous Galerkin Spectral Element Method for the direct numerical simulation of aeroacoustics.” In: *AIAA Aviation*, 2014.

[22] H. M. Frank and C.-D. Munz. “Direct aeroacoustic simulation of acoustic feedback phenomena on a side-view mirror.” In: *Journal of Sound and Vibration*, 371, pp. 132–149, 2016.

[23] G. J. Gassner. “Discontinuous Galerkin Methods for the Unsteady Compressible Navier-Stokes Equations.” PhD thesis. University of Stuttgart, Germany, 2009.

[24] A. Gnanaskandan and K. Mahesh. “A numerical method to simulate turbulent cavitating flows.” In: *International Journal of Multiphase Flow*, 70, pp. 22–34, 2015.

[25] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy. “Uniformly high order essentially non-oscillatory schemes, III.” In: *J. Comput. Phys.* 71, pp. 231–303, 1987.

[26] F. Hempert, S. Boblest, T. Ertl, F. Sadlo, P. Offenhäuser, C. Glass, M. Hoffmann, A. Beck, C.-D. Munz, and U. Iben. “Simulation of real gas effects in supersonic methane jets using a tabulated equation of state with a discontinuous Galerkin spectral element method.” In: *Computers & Fluids*, 145, pp. 167–179, 2017.

[27] F. Hempert. “Simulation of Multiphase Flows with Multiparameter Equation of States and the Discontinuous Galerkin Method.” PhD thesis. University of Stuttgart, Germany, in preparation.

[28] F. Hempert, S. Boblest, M. Hoffmann, P. Offenhäuser, F. Sadlo, C. W. Glass, C.-D. Munz, T. Ertl, and U. Iben. “High-Pressure Real-Gas Jet and Throttle Flow as a Simplified Gas Injector Model Using a Discontinuous Galerkin Method.” In: *High Performance Computing in Science and Engineering ’16*, at press.

[29] F. Hindenlang. “Mesh Curving Techniques for High Order Parallel Simulations on Unstructured Meshes.” PhD thesis. University of Stuttgart, Germany, 2014.

[30] F. Hindenlang, G. J. Gassner, C. Altmann, A. Beck, M. Staudenmaier, and C.-D. Munz. “Explicit discontinuous Galerkin methods for unsteady problems.” In: *Computers and Fluids*, 61, pp. 86–93, 2012.

[31] C. W. Hirt and B. D. Nichols. “Volume of fluid (VOF) method for the dynamics of free boundaries.” In: *J. Comput. Phys.* 39.1, pp. 201–225, 1981.

[32] B. Huang, Y. Zhao, and G. Wang. “Large eddy simulation of turbulent vortex-cavitation interactions in transient sheet/cloud cavitating flows.” In: *Computers & Fluids*, 92, pp. 113–124, 2014.

[33] A. Huerta, E. Casoni, and J. Peraire. “A simple shock-capturing technique for high-order discontinuous Galerkin methods.” In: *International Journal for Numerical Methods in Fluids*, 69.10, pp. 1614–1632, 2012.

[34] A. Jameson, W. Schmidt, and E. Turkel. “Numerical Solution of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes.” In: *Proc. of the AIAA 14th Fluid and Plasma Dynamic Conference*. AIAA-1981-1259. Paolo Alto, California, USA, 1981.

[35] D. A. Kopriva. *Implementing spectral methods for partial differential equations: Algorithms for scientists and engineers*. 2009.

[36] E. W. Lemmon, M. McLinden, and D. Friend. “Thermophysical Properties of Fluid Systems.” In: *NIST Standard Reference Database*, 69, 2015.

[37] X.-D. Liu, S. Osher, and T. Chan. “Weighted essentially nonoscillatory schemes.” In: *J. Comput. Phys.* 115, 1994.

[38] D. Lohse, B. Schmitz, and M. Versluis. “Snapping shrimp make flashing bubbles.” In: *Nature*, 413.6855, pp. 477–478, 2001.

[39] Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard, Version 3.0*. Tech. rep. 2012.

[40] G. M. Morton. *A computer oriented geodetic data base and a new technique in file sequencing*. 1966.

[41] C.-D. Munz, M. Auweter-Kurtz, S. Fasoulas, A. Mirza, P. Ortwein, M. Pfeiffer, and T. Stindl. “Coupled Particle-In-Cell and Direct Simulation Monte Carlo method for simulating reactive plasma flows.” In: *Comptes Rendus Mécanique*, 342.10, pp. 662–670, 2014.

[42] J. Neudorfer, A. Stock, R. Schneider, S. Roller, and C.-D. Munz. “Efficient parallelization of a three-dimensional high-order particle-in-cell for the simulation of a 170 GHz gyrotron resonator.” In: *IEEE Transactions on Plasma Science*, 41, pp. 87–98, 2013.

[43] P. Ortwein, T. Binder, S. Copplestone, A. Mirza, P. Nizenkov, M. Pfeiffer, T. Stindl, S. Fasoulas, and C.-D. Munz. “Parallel Performance of a Discontinuous Galerkin Spectral Element Method Based PIC-DSMC Solver.” In: *High Performance Computing in Science and Engineering ‘14*, 2015.

[44] S. Osher and J. A. Sethian. “Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations.” In: *Journal of computational physics*, 79.1, pp. 12–49, 1988.

[45] D.-Y. Peng and D. B. Robinson. “A new two-constant equation of state.” In: *Industrial & Engineering Chemistry Fundamentals*, 15.1, pp. 59–64, 1976.

[46] P.-O. Persson and J. Peraire. “Sub-cell shock capturing for discontinuous Galerkin methods.” In: *AIAA paper*, 2006.

[47] B. E. Poling, J. M. Prausnitz, and J. P. O’Connel. *The properties of gases and liquids*. 5th edition. 2007.

[48] J. Qiu. “A Numerical Comparison of the Lax–Wendroff Discontinuous Galerkin Method Based on Different Numerical Fluxes.” In: *Journal of Scientific Computing*, 30.3, pp. 345–367, 2007.

[49] J. Qiu, B. C. Khoo, and C.-W. Shu. “A numerical study for the performance of the Runge–Kutta discontinuous Galerkin method based on different numerical fluxes.” In: *Journal of Computational Physics*, 212.2, pp. 540–565, 2006.

[50] P. L. Roe. “Characteristic-based schemes for the Euler equations.” In: *Annual Review of Fluid Mechanics*, 18, pp. 337–365, 1986.

[51] P. L. Roe. “Discrete models for the numerical analysis of time-dependent multidimensional gas dynamics.” In: *Upwind and High-Resolution Schemes*, pp. 451–469, 1986.

[52] K. Salari and P. Knupp. *Code verification by the method of manufactured solutions*. Tech. rep. Sandia National Labs., Albuquerque, NM (US), 2000.

[53] F. Salvador, J. Martinez-Lopez, J.-V. Romero, and M.-D. Roselló. “Computational study of the cavitation phenomenon and its interaction with the turbulence developed in diesel injector nozzles by Large Eddy Simulation (LES).” In: *Mathematical and Computer Modelling*, 57.7, 2013.

[54] R. Saurel, F. Petitpas, and R. Abgrall. “Modelling phase transition in metastable liquids: application to cavitating and flashing flows.” In: *Journal of Fluid Mechanics*, 607, pp. 313–350, 2008.

[55] R. Saurel and R. Abgrall. “A multiphase Godunov method for compressible multifluid and multiphase flows.” In: *Journal of Computational Physics*, 150.2, pp. 425–467, 1999.

[56] R. Saurel, P. Cocchi, and P. B. Butler. “Numerical study of cavitation in the wake of a hypervelocity underwater projectile.” In: *Journal of Propulsion and power*, 15.4, pp. 513–522, 1999.

[57] R. Saurel, E. Franquet, E. Daniel, and O. Le Metayer. “A relaxation-projection method for compressible flows. Part I: The numerical equation of state for the Euler equations.” In: *Journal of Computational Physics*, 223.2, pp. 822–845, 2007.

[58] U. Setzmann and W. Wagner. “A New Equation of State and Tables of Thermodynamic Properties for Methane Covering the Range from the Melting Line to 625 K at Pressures up to 100 MPa.” In: *Journal of Physical and Chemical Reference Data*, 20.6, pp. 1061–1155, 1991.

[59] U. Setzmann and W. Wagner. “A new equation of state and tables of thermodynamic properties for methane covering the range from the melting line to 625 K at pressures up to 1000 MPa.” In: *J. Phys. Chem. Ref. Data*, 20.6, pp. 1061–1155, 1991.

[60] U. Setzmann and W. Wagner. “A new method for optimizing the structure of thermodynamic correlation equations.” In: *International Journal of Thermophysics*, 10.6, pp. 1103–1126, 1989.

[61] M. Sonntag, M. Hoffmann, and C.-D. Munz. “Non-conforming interfaces for a Shock Capturing in Discontinuous Galerkin Methods using Finite Volume Sub-cells.” In: to be published.

[62] M. Sonntag and C.-D. Munz. “Efficient Parallelization of a Shock Capturing for Discontinuous Galerkin Methods using Finite Volume Subcells.” In: *Journal of Scientific Computing*, pp. 1–28, 2016.

[63] A. Sou, B. Biçer, and A. Tomiyama. “Numerical simulation of incipient cavitation flow in a nozzle of fuel injector.” In: *Computers & Fluids*, 103, pp. 42–48, 2014.

[64] R. Span, W. Wagner, E. Lemmon, and R. Jacobsen. “Multiparameter equations of state - recent trends and future challenges.” In: *Fluid Phase Equilibria*, 183-184, pp. 1–20, 2001.

[65] R. Span. *Multiparameter equations of state: an accurate source of thermodynamic property data*. Berlin: Springer, 2000, XVIII, 367 S.

[66] E. F. Toro, M. Spruce, and W. Speares. “Restoration of the contact surface in the HLL-Riemann solver.” In: *Shock Waves*, 4.1, pp. 25–34, 1994.

[67] E. F. Toro. *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*. 2013.

[68] M. Versluis, B. Schmitz, A. von der Heydt, and D. Lohse. “How snapping shrimp snap: through cavitating bubbles.” In: *Science*, 289.5487, pp. 2114–2117, 2000.

[69] J. D. van der Waals. “On the Continuity of the Gaseous and Liquid States.” PhD thesis. Universiteit Leiden, 1873.

[70] W. Wagner and A. Pruß. “The IAPWS formulation 1995 for the thermodynamic properties of ordinary water substance for general and scientific use.” In: *Journal of Physical and Chemical Reference Data*, 31.2, pp. 387–535, 2002.

[71] G. Wang and M. Ostoja-Starzewski. “Large eddy simulation of a sheet/cloud cavitation on a NACA0015 hydrofoil.” In: *Applied Mathematical Modelling*, 31.3, pp. 417–447, 2007.

[72] J. H. Wilkinson. *The algebraic eigenvalue problem*. Vol. 87. Clarendon Press Oxford, 1965.

[73] F. R. Young. *Cavitation*. 1999.

List of Tables

List of Figures

Lebenslauf

07.06.1985	Geboren in Goslar
1991 - 1995	Grundschule Hahndorf
1995 - 1997	Orientierungsstufe Goldene Aue, Goslar
1997 - 2004	Christian-von-Dohm-Gymnasium, Goslar
2004	Allgemeine Hochschulreife
2004 - 2005	Zivildienst: Rettungsdienst Goslar
2005 - 2012	Studium der Luft- und Raumfahrttechnik an der Universität Stuttgart
	Vertiefungsrichtungen: Strömungslehre, Datenverarbeitung
2012	Abschluß: Diplom-Ingenieur
2012 - 2017	Wissenschaftlicher Mitarbeiter am
	Institut für Aerodynamik und Gasdynamik