Solving the Level Set Equation using High-order Non-oscillatory Reconstruction

by Tobias Martin

Matr.Nr: 211203603

Study research project

Master of Science Ship Technology and Ocean Engineering

At the Chair of Modelling and Simulation

Prof. Dr.-Ing. habil. Nikolai Kornev

Date of submission: 14.03.2016

Task formulation of the study research project

Solving the level set equation using high-order non-oscillatory reconstruction

of Mr. Tobias Martin

The project shall include the following steps:

1. Implementation of (W)ENO interpolation scheme from scratch, including

   - Parallel stencil collection algorithm
   - Efficient computation of the resulting overdetermined system of equation
   - Solving the least-squares problem using QR-decomposition or SVD

   The scheme shall work at least up to the third order of accuracy and on three-dimensional, arbitrary meshes. Furthermore, the parallelisation of the code has to be achieved.

2. Showing the accuracy and proper functioning of the scheme on structured and unstructured meshes

   - Implementation of a high-order numerical integration for the calculation of the $\mathcal{L}_p$-norm of errors
   - Analysis of the $\mathcal{L}_p$-norm of errors for smooth functions in two and three dimensions
   - Reconstruction of a non-smooth function in two dimensions
   - Numerical convergence study

3. Application of the level set equation using a high-order reconstruction

   - Discretisation of the level set equation within the Finite-Volume-Method
   - Determination of the numerical fluxes by solving the corresponding Riemann problem

- Using high-order explicit time integration
- Verification by appropriate problems in two and three dimensions for different mesh types

Ivan Shevchuk, 21.10.2015

Preface

In this project, the first steps of the development of a level set method for incompressible two-phase flows are presented. The transport equation for the scalar level set is transformed using the finite volume method which results in a Riemann problem for the evaluation of fluxes at the cell faces. For this purpose, high-order (W)ENO schemes are implemented working on convex, polyhedral grids in 3D. First, a Quasi-ENO scheme with fixed central stencils is developed. This includes a parallel stencil collection algorithm, an algorithm for the computation of necessary volume integrals, and an efficient QR-decomposition for the resulting least-squares problems using Householder transformation and automatic treatment of rank-deficient problems. The verification process shows the principal potential of the scheme of providing high-order accurate solutions without the introduction of spurious oscillations. However, the weaknesses of this method reduce the efficiency and applicability for level set methods on arbitrary, three-dimensional meshes. Therefore, an additional WENO scheme is presented, which extends the Quasi-ENO scheme by computing the solution from a central and several sector stencils. The smoothness of the solution is mesh-independent and partially solution-independent which increases the efficiency in runtime. The WENO scheme appears as the appropriate method for the further work because the first verification shows improved results on unstructured grids.

Acknowledgement

First of all, I would like to thank my second supervisor Mr. Ivan Shevchuk. He inspired me to start with this topic and was always available for questions. I enjoyed the active discussions which pushed our project steady forward. I gladly look forward to our further cooperation. In addition, I want to express my gratitude to the Chair of Modelling and Simulation of Professor Nikolai Kornev for the hints they gave me on different aspects of this project. Finally, I want to thank Professor Carl Ollivier-Gooch from the University of British Columbia for an informative email exchange.

Contents

List of Figures

List of Tables

List of Abbreviations

ADER	Arbitrary high order schemes using Derivatives
CFL	Courant-Friedrichs-Lewy number
FVM	Finite Volume Method
IC	Initial Condition
MPI	Message Passing Interface
PDE	Partial Differential Equation
SVD	Singular Value Decomposition
TVB	Total Variation Bounded
TVD	Total Variation Diminishing
(W)ENO	(Weighted) Essentially Non Oscillatory

List of Symbols

a_k	degrees of freedom [-]
$\widetilde{a}_k$	modified degrees of freedom [-]
d_m	linear weight [-]
$\vec{n}$	outward normal vector [-]
$\vec{n}_l$	normal vector of a face [-]
p	polynomial representation of a variable [-]
r	order of the polynomial [-]
$\vec{u}$	velocity field [$\frac{m}{s}$]
u_{n_l}	velocity normal to a face [$\frac{m}{s}$]
w	geometric weight [-]
w^{ENO}	ENO weight [-]
$\vec{x}$	physical coordinates [-]
z	pseudo rank of the reconstruction matrix [-]
$\mathcal{A}$	reconstruction matrix [-]

$\mathcal{A}^+$	Moore-Penrose pseudoinverse [-]
$\mathcal{B}$	oscillation indicator matrix [-]
D_i	sub-domain [-]
$\vec{F}$	flux vector [$\frac{m}{s}$]
F_l	face of a control volume [-]
F_{n_l}	flux normal to a face [$\frac{m}{s}$]
$\mathcal{I}_S$	smoothness indicator [-]
$\mathcal{J}$	Jacobian matrix [-]
J_{max}	number of cells in each stencil [-]
K	required number of degrees of freedom [-]
$\mathcal{L}_p$	p-norm of error [-]
N	total number of cells in domain [-]
R	residuum [-]
S	stencil [-]
V_i	control volume [-]
$\|V\|$	volume [m^3]
$\|V_T\|$	total volume of domain [m^3]
$\vec{X}$	reference space of sectors [-]
α	order of accuracy [-]

$\vec{\xi}$	reference space of stencils [-]
τ	tolerance parameter [-]
φ	scalar level set [-]
ω_m	WENO weight [-]
Δ	characteristic cell size [m]
Φ	conservative variable [-]
$\overline{\Phi}$	cell-averaged value of a conservative variable [-]
Ψ_k	orthogonal basis function [-]
Ω_k	basis function [-]
∂V_i	boundary surface of control volume [-]
∇	divergence operator [-]

Chapter 1

Introduction

The level set method requires the solution of a linear hyperbolic equation, namely level set equation. After the discretisation with the finite volume method (FVM), fluxes have to be computed at all boundary faces of cells. In doing so, the field of the scalar variable has to be reconstructed on appropriate points. A suitable scheme has to be chosen which copes with the formation of large gradients in front propagating problems and preserve the sharpness of the interface without creating spurious oscillations near discontinuities at the interface [22]. This Gibbs-like phenomenon generates $\mathcal{O}(1)$ spurious oscillations at points of discontinuity, proportional to the size of the jump [6].

According to Godunov, linear schemes with such a monotonic behaviour are at most first-order accurate [28] leading to the conclusion that a high-order scheme can just be non-oscillatory near high gradients if it is non-linear, even for linear hyperbolic equations. Most famous high-order schemes fulfilling these properties are total variation diminishing (TVD) schemes which can reach second-order accuracy and additionally often satisfy boundedness even in 3D [33]. However, it is well known that TVD schemes degenerate to first-order near extrema. Therefore, (weighted) essentially non-oscillatory ((W)ENO) schemes, which introduce a non-linear weighting for preventing oscillations near discontinuities and simultaneously reach arbitrary order of accuracy in smooth re-

gions, were developed. However, they are not necessarily bounded [33].

The first ENO scheme based on adaptive stencils was introduced by Harten, Osher, Engquist, and Chakravarthy in 1987 [6]. The scheme searches for the appropriate cells dynamically in order to get the smoothest solution in each time step. Abgrall [1] expanded ENO for unstructured grids in 1994. The drawbacks of ENO schemes are the prevention of convergence in the case of frequent switching of the stencil from one time step to another [17] and costly operations in runtime reducing the overall performance. Therefore, WENO schemes were introduced first by Jiang, Shu [9], later expanded by Friedrich for two-dimensional, unstructured grids [4]. Recently, WENO schemes were extended to accomplish three-dimensional, unstructured grids with up to sixth order of accuracy [3, 29]. Pringuey [22] successfully combined this approach with a level set method in his Ph.D. thesis. In contrast to ENO, WENO schemes compute the solution on several fixed stencils and combine these by a non-linear weighting. The weights are calculated from an evaluation of the smoothness in each stencil. This theoretically leads to $(2r+1)^{th}$ order of accuracy from a polynomial of r^{th}-order at most [16]. However, they are computationally quite expensive, especially on unstructured meshes in 3D [3], due to the use of several stencils [3]. Furthermore, the reconstruction may fail near critical points, for very coarse grids, and in case not enough smooth data is provided [31]. To overcome these drawbacks, an alternative scheme named Quasi-ENO was introduced by Gooch [17]. In contrast to classical ENO schemes, this one is based on a high-order least-squares reconstruction which fulfils the ENO-property by a pointwise data-dependent weighting in one fixed, central stencil [16]. Classical least-squares reconstruction schemes are able to reach high-orders in smooth regions but fail near discontinuities. A common way of eliminating this problem is introducing a limiter as Wang [30] and McDonald [13] demonstrate. However, the underlying mathematical problem is the inclusion of the data from cells which are behind a discontinuity into the reconstruction. The Quasi-ENO scheme solves this problem in a more efficient way

as the classical ENO scheme and in addition offers the possibility of excluding cells of poor quality from the reconstruction. Nonetheless, Quasi-ENO displays problems of providing higher orders for three-dimensional cases, as it will be shown later.

In chapter 2 the level set equation and its discretisation using FVM will be introduced. Afterwards, the implementation of Gooch's [16] Quasi-ENO scheme using `C++` and the open source CFD toolbox OpenFOAM is described, including the stencil searching algorithm, solving the least-squares problem and parallelisation. The proper functioning of the scheme will be shown in chapter 4 for different reconstruction cases in 2D and 3D, including non-smooth functions. In chapter 5, the further procedure for solving the level set equation will be given in detail, in particular computation of fluxes, solving the Riemann problem, and time discretisation. This is followed by the application of the implemented solver for convection problems. Due to unsatisfying results, an advancement of the Quasi-ENO scheme to a WENO scheme is implemented and described in chapter 6. It is successfully applied to the level set equation in chapter 7. Final remarks can be found in chapter 8. Additionally, the code and all prepared calculations can be found on a DVD on the last page of the work.

Chapter 2

Level Set Equation

The scope of this thesis is limited to a linear hyperbolic partial differential equation (PDE) in the form of [11]

$$\frac{\partial \Phi}{\partial t} + \vec{u} \cdot (\nabla \cdot \Phi) = 0, \tag{2.1}$$

with Φ as a general conserved variable and ∇ the divergence operator. Since an incompressible fluid is assumed, equation (2.1) may be rewritten as a hyperbolic conservation law

$$\frac{\partial \Phi}{\partial t} + \nabla \cdot \vec{F}(\vec{u}, \Phi) = 0, \tag{2.2}$$

with $\vec{F} = (u_x\Phi, u_y\Phi, u_z\Phi)$ representing the fluxes in the three Cartesian coordinates. Based on the finite volume method (FVM), the fluxes are evaluated by integrating (2.2) over an arbitrary volume of a cell V_i

$$\int_{V_i} \frac{\partial \Phi}{\partial t} \, \mathrm{d}V + \int_{V_i} \nabla \cdot \vec{F} \, \mathrm{d}V = 0. \tag{2.3}$$

Introducing the cell-averaged value of Φ_i as $\overline{\Phi}_i$ and assuming the time-independence of Φ, the first term on the left-hand side yields

$$\int_{V_i} \frac{\partial \Phi}{\partial t} \, \mathrm{d}V = \frac{\mathrm{d}}{\mathrm{d}t} \overline{\Phi}_i |V_i|. \tag{2.4}$$

The second term of (2.3) is transformed to a surface integral over the boundary surface of the volume ∂V_i using Gauss's theorem. Thus, equation (2.2) leads to

$$\frac{\mathrm{d}}{\mathrm{d}t}\overline{\Phi}_i + \frac{1}{|V_i|}\int_{\partial V_i} \vec{F}\cdot\vec{n}\,\mathrm{d}\partial V_i = 0, \tag{2.5}$$

with the normal surface vector $\vec{n}$. Moreover, the surface integrals are divided into a sum of L_i integrals over the faces F_l of V_i

$$\frac{\mathrm{d}}{\mathrm{d}t}\overline{\Phi}_i + \frac{1}{|V_i|}\sum_{l=1}^{L_i}\int_{F_l} F_{n_l}\left(\Phi^-,\Phi^+\right)\,\mathrm{d}F_l = 0, \tag{2.6}$$

with $F_{n_l}\left(\Phi^-,\Phi^+\right) = \vec{F}\cdot\vec{n}_l$ the flux into the direction of $\vec{n}_l$, which is the outward normal vector of F_l. The flux depends on the solution of reconstruction on both sides of F_l: Φ^- and Φ^+. By definition, Φ^- is the reconstructed value from the cell V_i itself, while Φ^+ is obtained from the neighbour cell of the face $|V_{jl}|$. Since Φ^- and Φ^+ are different values, a Riemann problem has to be solved locally. The interpolation from the cell centres can reach arbitrary high-orders of accuracy in space by using a polynomial representation. In the next chapter, this representation is applied for the Quasi-ENO scheme in order to compute the reconstruction polynomials. The further handling of (2.6) can be found in chapter 5.

Chapter 3

High-Order Reconstruction Scheme

The aim of this chapter is the development of a high-order interpolation scheme which can be used for evaluating the fluxes at cell faces from a known cell-averaged field of any variable Φ. The scheme handles with arbitrary convex cells on unstructured, three-dimensional meshes. The efficiency of the implementation is increased by parallelising the code using Message Passing Interface (MPI). In general, the theoretical basis is taken from the Quasi-ENO scheme of Gooch [16].

3.1 Overview of the Method

The spatial domain is taken in its discretised form of N cells with the volumes $|V_i|$, $i = 1,..,N$, and the data of any variable Φ is stored in the cell centres $\vec{x}_i$ represented by the cell average value $\overline{\Phi}_i$. With the aid of a least-squares reconstruction, Φ is presented by a polynomial representation p_i in each cell V_i with the constraint of conservation of the mean value within V_i

$$\overline{\Phi}_i = \frac{1}{|V_i|}\int_{V_i} \Phi(\vec{x})\ \mathrm{d}V = \frac{1}{|V_i|}\int_{V_i} p_i(\vec{x})\ \mathrm{d}V. \tag{3.1}$$

The polynomial p_i is represented by an expansion over local polynomial basis functions Ω_k as [29]

$$p_i(\vec{x}) = \overline{\Phi}_i + \sum_{k=1}^{K} a_k \Omega_k, \tag{3.2}$$

where a_k are the degrees of freedom and K relates to the order of the polynomial r in 3D pursuant to

$$K = \frac{(r+1)(r+2)(r+3)}{6} - 1, \tag{3.3}$$

respectively in 2D to

$$K = \frac{(r+1)(r+2)}{2} - 1. \tag{3.4}$$

The degrees of freedom for a target cell V_i are determined using the cell averaged values of Φ from the neighbouring cells V_j. Hence, a stencil S_i has to be collected as

$$S_i = \bigcup_{j=0}^{J_{max}} V_j. \tag{3.5}$$

The choice of J_{max} is an important parameter, which will be discussed later. By definition, $j = 0$ corresponds to the target cell V_i. The basis functions Ω_k have to be chosen with the constraint of satisfying (3.1), equivalent to a zero mean value over V_i. Gooch [16] writes this condition explicitly in the resulting system of equation and includes it by an elimination technique, such as Gaussian eliminations. Here, equation (3.1) will be satisfied analytically by an appropriate definition of the basis functions

$$\Omega_k(\vec{x}) = \Psi_k(\vec{x}) - \frac{1}{|V_i|} \int_{V_i} \psi_k(\vec{x}) \, \mathrm{d}x\mathrm{d}y\mathrm{d}z, \quad k = 1, ..., K, \tag{3.6}$$

with arbitrary orthogonal polynomial basis functions Ψ_k. In accordance with Gooch [17] and Friedrich [4], a Taylor series expansion around the center of V_i is used and defined as

$$\Psi_k(\vec{x}) = (x - x_i)^n (y - y_i)^m (z - z_i)^o, \tag{3.7}$$

where k corresponds to one combination of n, m, o such that $0 \leq n + m + o \leq r$. For the sake of completeness, alternative definitions could be shifted Legendre polynomials or

a normal Taylor series [22, 29]. Finally, a system of equations has to be solved in order to calculate a_k. The system is computed with the aim of preserving the averaged values $\overline{\Phi}_j$ in all cells V_j of the stencil S_i by the corresponding cell averages of p_i. Hence, the following expression is valid

$$\overline{\Phi}_j = \frac{1}{|V_j|} \int_{V_j} p_i(\vec{x}) \, \mathrm{d}x\mathrm{d}y\mathrm{d}z, \quad j = 1, ..., J_{max}. \tag{3.8}$$

Substituting (3.8) in (3.2) yields the system of equations as

$$\overline{\Phi}_j - \overline{\Phi}_i = \sum_{k=1}^{K} a_k \cdot \left(\frac{1}{|V_j|} \int_{V_j} \Omega_k(\vec{x}) \, \mathrm{d}x\mathrm{d}y\mathrm{d}z \right), \quad j = 1, ..., J_{max} \tag{3.9}$$

$$b_j = \sum_{k=1}^{K} \mathcal{A}_{jk} a_k, \quad j = 1, ..., J_{max}. \tag{3.10}$$

In the following, the individual steps of the method are described in greater detail.

3.2 Stencil Collection Algorithm

In contrast to classical ENO schemes, the Quasi-ENO scheme computes the solutions on time-invariant stencils. Thus, the time consuming collection part simplifies and has to be done just once during the preprocessing step. The stencil should be compact in a topological sense to get physical reliable results. The most efficient collection for unstructured meshes is adding the neighbours of the target cell iteratively until the stencil has the required size. The most compact stencil could be achieved by using point-neighbours. On the contrary, the use of face-neighbours extends the dependent data further into the mesh which reduces the redundancy of data on anisotropic, structured meshes [22]. Therefore, the stencil S_i is collected iteratively through its face-neighbours until 1.5 J_{max} cells are gathered. The candidates are then sorted by center to center distances to the target cell, and the nearest J_{max} cells are collected. By collecting a surplus of possible stencil cells, the algorithm is independent of the starting point of an iteration and provides complete layers of new neighbours at a time. In figure 3.1, the

resulting stencils are shown. It may happen that more layers have to be considered until the necessary size is reached near boundaries. At this point, the iterative implementation is straightforward and advantageous.

The system of equations (3.10) provides a solution if the matrix $\mathcal{A}$ is at least squared, resulting in the condition $J_{max} \geq K$. As Pringuey [22] and Tsoutsanis [29] state, choosing $J_{max} = K$ leads to unstable solutions or eventually ill-conditioned systems. Additionally, the exclusion of data beyond discontinuities using a data-dependent ENO weighting is just possible in case of a surplus of useful cells [17]. Therefore, J_{max} should be approximately 2 K for three-dimensional problems and 1.5 K in 2D.

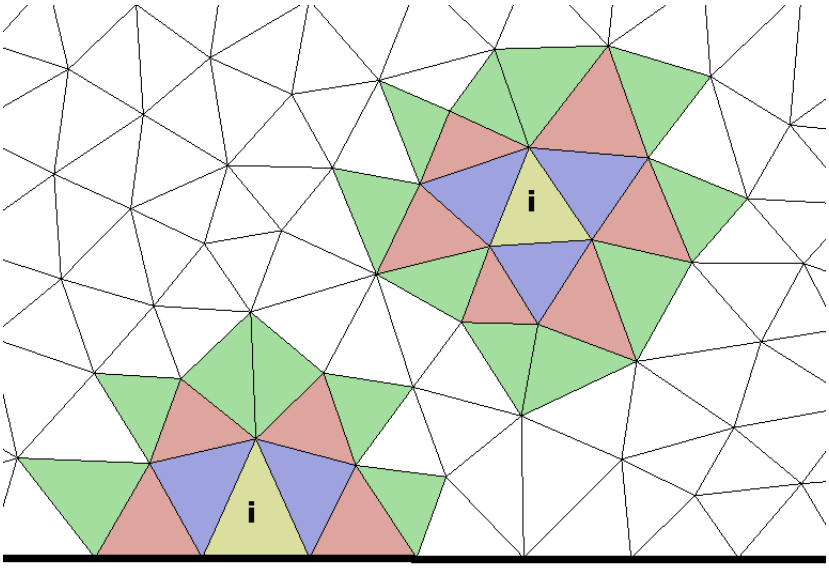


Figure 3.1: Layers of cells added at each iteration: in purple, initial neighbours; in red, the cells added at the first iteration; in green, the cells added at the second iteration - two-dimensional example

3.3 Determination of the Reconstruction Matrix

Recalling the derivation of the resulting system of equations in chapter 3.1, each entry of the reconstruction matrix $\mathcal{A}$ is obtained by an integration of a basis function over a cell V_j of S_i (see (3.9)). Combining (3.2), (3.6) and (3.9) yields

$$\mathcal{A}_{jk} = \frac{1}{|V_j|}\int_{V_j}\left(\Psi_k - \frac{1}{|V_i|}\int_{V_i}\Psi_k \,\mathrm{d}x\mathrm{d}y\mathrm{d}z\right)\,\mathrm{d}x\mathrm{d}y\mathrm{d}z. \tag{3.11}$$

The second integrand leads to a constant value for which reason (3.11) can be rewritten as

$$\mathcal{A}_{jk} = \frac{1}{|V_j|}\int_{V_j}\Psi_k \,\mathrm{d}x\mathrm{d}y\mathrm{d}z - \frac{1}{|V_i|}\int_{V_i}\Psi_k \,\mathrm{d}x\mathrm{d}y\mathrm{d}z. \tag{3.12}$$

The volume integrals have to be evaluated for each combination V_{j_i} since Ψ_k depends on the center of V_i. In order to avoid this, the Taylor series is transformed by replacing e.g. $(x-x_i)$ with $(x-x_j)+(x_j-x_i)$ at which x_j is the cell center of V_j. Inserting these expressions in (3.12), $\mathcal{A}_{jk}$ finally gets

$$\begin{aligned}\mathcal{A}_{jk} = &\sum_{p=0}^{l}\sum_{q=0}^{m}\sum_{g=0}^{n}\binom{l}{p}\binom{m}{q}\binom{n}{g}(x_j-x_i)^p(y_j-y_i)^q(z_j-z_i)^g\cdot\\ &\frac{1}{|V_j|}\int_{V_j}(x-x_j)^{(l-p)}(y-y_j)^{(m-q)}(z-z_j)^{(n-g)}\,\mathrm{d}x\mathrm{d}y\mathrm{d}z\\ &-\frac{1}{|V_i|}\int_{V_i}(x-x_i)^l(y-y_i)^m(z-z_i)^n\,\mathrm{d}x\mathrm{d}y\mathrm{d}z,\end{aligned} \tag{3.13}$$

where k corresponds to one combination of n,m,l such that $0 \leq n+m+l \leq r$. A more detailed derivation of (3.13) can be found in Gooch's paper [17]. Additionally, a geometrical weighting which increases the compactness by specifying the importance of the cells influence on the solution is introduced [16]. The weights are calculated from the inverse distance between centres

$$w_{ij} = \frac{1}{|\vec{x_j}-\vec{x_i}|^2}. \tag{3.14}$$

They are added to both sides of the system of equations which results in the weighted system

$$w_{ij}b_j = \sum_{k=1}^{K} w_{ij}\mathcal{A}_{jk}a_k, \quad j = 1,...,J_{max}. \tag{3.15}$$

The remaining task is the integration of the monomials Ψ_k over each cell in the domain. Under consideration of an arbitrary polyhedron, two efficient ways of evaluation can be provided. Pringuey [22] decomposes each cell into tetrahedra and applies a three-dimensional Gaussian quadrature formula for each of them. Here the volume integrals are transformed into surface integrals instead by using the divergence theorem [15]. Denoting $\vec{n}$ as the outward normal vector, it is transformed as

$$\int_{V_i} \nabla \cdot \Psi_k \, \mathrm{d}V_i = \int_{\partial V_i} \Psi_k \vec{n}\vec{e}_n \, \mathrm{d}\partial V_i. \tag{3.16}$$

The right-hand side of (3.16) can be further written as a sum of surface integrals over the L_i faces F_l of V_i. In this case, the normal vector of each face $\vec{n}_l$ is constant and can be taken out of the integrals. The desired volume integrals of the monomials are received from (3.16) by integrating $\Psi_k(\vec{x})$ analytically in one of the coordinates. Taking the general form of Ψ_k in (3.7) into account, it finally leads to

$$\int_{V_i} (x-x_j)^n (y-y_j)^m (z-z_j)^o \, \mathrm{d}V_i = \sum_{l=1}^{L_i} \frac{n_{l,x}}{n+1} \int_{F_l} (x-x_j)^{n+1} (y-y_j)^m (z-z_j)^o \, \mathrm{d}F_l. \tag{3.17}$$

The surface integrals are computed by decomposing the faces into triangles and using a Gaussian-quadrature rule for triangles. `OpenFOAM's polyMeshTetDecomposition`, which splits any plane polygon into a minimum number of triangles, is used for the decomposition. The experiences show accurate results for arbitrary cell shapes as long as it is convex and all faces are planar. Each triangle is transformed to a standard triangle by applying a linear mapping [2]. In order to compute the volume integrals of fourth order polynomials, surface integrals of fifth order monomials have to be considered,

which is provided by using seven quadrature points on each triangle. The appropriate coordinates of them are also taken from Deng [2].

3.4 Solving the System of Equations

As the next step the solution of the system of equations (3.10) has to provided. The robustness of the system is improved by considering more equations than unknowns, which results in an overdetermined, linear least-squares problem. Physically, this problem corresponds to the minimization of the $\mathcal{L}_2$-norm of the error in predicting the averaged values of the polynomial in all cells of the stencil [16]

$$min\|\mathcal{A}\cdot a - b\|_2, \quad \mathcal{A}\in\mathbb{R}^{m\times n}, \quad b\in\mathbb{R}^m, \quad m\geq n, \tag{3.18}$$

with a the solution vector, containing the degrees of freedom. The easiest way of solving (3.18) is using normal equations

$$\begin{aligned} \mathcal{A}^T\mathcal{A}a &= \mathcal{A}^T b \\ a &= (\mathcal{A}^T\mathcal{A})^{-1}\mathcal{A}^T b = \mathcal{A}^+ b. \end{aligned} \tag{3.19}$$

The so-called Moore-Penrose pseudoinverse $\mathcal{A}^+$ could be precomputed, as long as $\mathcal{A}$ is solution independent. However, according to Golub [5], the accuracy of the solution from (3.19) depends on the square of the condition number of $\mathcal{A}$. In practice, the resulting condition numbers are relatively high, particularly for higher order polynomials. Therefore, the singular value decomposition (SVD) is the usual way of calculating $\mathcal{A}^+$. As described in chapter 3.5, the ENO property is fulfilled by the introduction of additional data-dependent weights in $\mathcal{A}$, which makes the calculation of $\mathcal{A}^+$ useless. Quite the contrary, it will be shown that the upper triangular form of the matrix is necessary, which can be computed by a $\mathcal{QR}$-decomposition with Householder transformations very efficiently. According to Lawson [10], this approach needs just one quarter

of flops for obtaining a in comparison to SVD, which reduces the drawback of solving the whole system of equations continually.

In practice, the calculations are often characterised by the occurrence of rank-deficient matrices. Its is mainly caused by nearly linear-dependent lines in $\mathcal{A}$, typically arising if several cells of a stencil are on one line on structured grids [3], and by exclusion of too many cells from the reconstruction during the ENO weighting. These deficiencies are indicated by very small values on the main diagonal of $\mathcal{A}$ before the back-substitutions are performed. A possible solution is the addition of more cells to the stencil [16]. However, the condition number is linear dependent on J_{max} for which reason the accuracy of the solution could decrease. Following Lawson [10], the better solution is the negligence of the problem causing information from reconstructions, which is handled by replacing $\mathcal{A}$ by a nearby rank-deficient matrix $\tilde{\mathcal{A}}$.

The solution of (3.18) is computed by a $\mathcal{QR}$-decomposition of $\mathcal{A}$ using Householder transformations. Applying suitable orthogonal transformations, the matrix $\mathcal{A}$ with rank z can be decomposed as

$$\mathcal{A} = \mathcal{Q}^T \mathcal{R} \mathcal{P}^T, \tag{3.20}$$

with $\mathcal{Q}$ as a $m \times m$ orthogonal matrix, defined as the Householder matrix, $\mathcal{P}$ as a $n \times n$ orthogonal matrix, identified as the permutation matrix for a column interchange strategy and $\mathcal{R}$ as a $m \times n$ matrix of the form

$$\mathcal{R} = \begin{pmatrix} \mathcal{R}_{11} & \mathcal{R}_{12} \\ 0 & \mathcal{R}_{22} \end{pmatrix}. \tag{3.21}$$

Here, $\mathcal{R}_{11}$ is a $z \times z$ non-singular triangular matrix. In the case of a full rank problem $z = n$, $\mathcal{R}_{12}$ and $\mathcal{R}_{22}$ are zero matrices, while $\mathcal{R}_{11}$ is a $z \times z$ full-rank matrix. Hence, the

solution of (3.18) is obtained by [10]

$$\begin{aligned} \mathcal{Q}b = g = \begin{pmatrix} g_1 \\ g_2 \end{pmatrix} \\ \mathcal{P}^T a = y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \\ \mathcal{R}_{11} y_1 = g_1, \quad \textit{with solution } \widetilde{y_1} \\ a = \mathcal{P} \begin{pmatrix} \widetilde{y_1} \\ y_2 \end{pmatrix}. \end{aligned} \tag{3.22}$$

Since $\vec{y_2}$ is arbitrary, it can be shown that the unique solution of minimum length results from choosing $\vec{y_2} = \vec{0}$ as

$$a = \mathcal{P} \begin{pmatrix} \widetilde{y_1} \\ 0 \end{pmatrix}, \tag{3.23}$$

with the norm of the residuum r

$$\|r\| = \|b - \mathcal{A}a\| = \|g_2\|. \tag{3.24}$$

For the rank-deficient case, z gets a so called pseudorank with $z < n$ and the matrix $\mathcal{R}$ has to be considered in form (3.21). In this case, the reconstruction matrix $\mathcal{A}$ is replaced by

$$\tilde{\mathcal{A}} = \mathcal{Q}^T \begin{pmatrix} \mathcal{R}_{11} & \mathcal{R}_{12} \\ 0 & 0 \end{pmatrix} \mathcal{P}^T, \tag{3.25}$$

with an error depending on the tolerance parameter τ

$$\|\tilde{\mathcal{A}} - \mathcal{A}\| = \|\mathcal{R}_{22}\| \leq \tau\sqrt{n-z}. \tag{3.26}$$

Based on (3.25), the solution of the rank-deficient problem is [10]

$$\begin{aligned} \mathcal{Q}\tilde{\mathcal{A}}\mathcal{P} &= \mathcal{R} \\ \mathcal{Q}b = c &= \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \\ \begin{pmatrix} \mathcal{R}_{11} & \mathcal{R}_{12} \end{pmatrix} \mathcal{K} &= \begin{pmatrix} \mathcal{W} & 0 \end{pmatrix} \\ \mathcal{W}y_1 = c_1, &\quad \textit{with solution } y_1 \textit{ and } y_2 \textit{ arbitrary} \\ x = \mathcal{P}\mathcal{K} &\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}. \end{aligned} \tag{3.27}$$

Again, the minimum length solution, even for the rank-deficient case, results from $\vec{y_2} = \vec{0}$ with the norm of the residuum

$$\|r\| = \|b - \mathcal{A}a\| = \|c_2 - \mathcal{R}_{22}y_2\| = \|c_2\|. \tag{3.28}$$

The permutation matrix $\mathcal{P}$ results implicitly from a column interchange strategy in order to compute a well-conditioned $\mathcal{R}_{11}$. During the construction of the i^{th} Householder transformation, the $\mathcal{L}_2$-norm of the vectors of the column $j = i, i+1, ..., n$ are compared, and the column with the highest norm is interchanged with the i^{th} column. Additionally, a similar row interchange strategy is chosen to get the highest element on the main diagonal. The orthogonal matrix $\mathcal{Q}$ is the product of z Householder transformations which are calculated as

$$\mathcal{Q} = I_m + \left(\frac{\|u\|^2}{2} \right)^{-1} uu^T. \tag{3.29}$$

Each transformation i zeros the elements below the main diagonal of the i^{th} column of $\mathcal{Q}\mathcal{A}$ and $\mathcal{Q}\tilde{\mathcal{A}}$ respectively resulting in the upper triangular form of $\mathcal{R}$. After this triangulation, the pseudorank z is determined as the largest index j so that $|r_{jj}| > \tau$, where τ is the introduced tolerance parameter, taken as 10^{-5}. In the rank-deficient case $z < n$, additional Householder transformations have to be computed in order to get the orthogonal matrix $\mathcal{K}$ and $\mathcal{W}$ as a non-singular and upper triangular $z \times z$ matrix.

As a final remark, preconditioning can be introduced for treating bad conditioned problems which typically arise for higher order polynomials. Friedrich [4] removes the bad conditioning arising from scaling effects by multiplying the matrix by the inverses of the cell volumes. Another common way is the multiplication of a diagonal matrix from the right at which the entries are the inverses of the magnitudes of the highest value in the corresponding columns in $\mathcal{A}$. However, the condition number within the Quasi-ENO scheme gets worse after ENO weights are introduced in $\mathcal{A}$. In this case, both described ways of preconditioning have no positive effect since the bad condition arises from the very small singular values corresponding to lines of small ENO weights independent of scaling effects. This fact leads to stability problems during the verification process and reveals an important drawback of the scheme.

3.5 Satisfying the ENO Property

The Quasi-ENO scheme fulfils the ENO property completely data-dependent by weighting the data within each stencil according to there influence on a smooth solution. Lui, Osher and Chan [12] defined the necessary behaviour as the weighting of non-smooth data with $\mathcal{O}(\Delta^{(r+1)})$ where Δ is a characteristic size of the cell. At least one cell in a stencil V_j does not satisfy (3.8) exactly if a non-smooth function is reconstructed and if the stencil is located across a discontinuity [17]. In this case, it yields

$$\overline{\Phi}_j - \frac{1}{|V_j|}\int_{V_j} p_i(\vec{x})\,\mathrm{d}x\mathrm{d}y\mathrm{d}z = \mathcal{O}(1), \tag{3.30}$$

which corresponds to a residuum R of the least-squares problem of the same order. On the contrary, the least-squares problem will be solved with the appropriate order of truncation in smooth regions such that R is $\mathcal{O}(\Delta^{(r+1)})$. Thus, the residuum can provide an indicator for the smoothness of the solution in general. In order to filter the non-

smooth data, the connection between the target cell V_i and V_j can be investigated as

$$\frac{\Phi_j - \Phi_i}{|\vec{x}_j - \vec{x}_i|} = \begin{cases} \mathcal{O}(1) & \text{if smoothly connected} \\ \mathcal{O}(\Delta^{-1}) & \text{if not smoothly connected} \end{cases}, \tag{3.31}$$

with $\vec{x}_j$ and $\vec{x}_i$ respectively represents the coordinates of the cell centres of V_j and V_i. Hence, an appropriate weighting can be constructed from R as a smoothness indicator $\mathcal{I}_S$ detecting stencils with non-smooth data and (3.31) as a tool for indicating the cells causing the oscillatory solution [17]

$$w_{ij}^{ENO} = \frac{1}{10^{-10} + \mathcal{I}_S} \tag{3.32}$$

$$\mathcal{I}_S = \Delta^2 R \left| \frac{\Phi_j - \Phi_i}{|\vec{x}_j - \vec{x}_i|} \right|^{r+1}, \tag{3.33}$$

where $\Delta^2 = |V_i|^{\frac{2}{3}}$ removes the geometric weighting on R. This choice leads to the asymptotic behaviour

$$w_{ij}^{ENO} = \begin{cases} max(10^{10}, \mathcal{O}(\Delta^{-(r+1)})) & \text{if smooth polynomial} \\ \mathcal{O}(1) & \text{if smooth data, but stencil with non-smooth data} \\ \mathcal{O}(\Delta^{(r+1)}) & \text{if non-smooth data} \end{cases}.$$

All weights are similar for a smooth polynomial, which preserves the quality of the reconstruction. In the case of a non-smooth stencil, the weights for non-smooth data are by a factor of $\mathcal{O}(\Delta^{(r+1)})$ smaller than the weights for the smooth data which coincides with the required ENO property [12]. In practical computations, the weights are introduced after receiving the residual from the least-squares problem. Using the implemented $\mathcal{QR}$-decomposition from chapter 3.4, R is obtained from the right-hand side of the system after the upper triangulation of $\mathcal{A}$ without additional steps as

$$R = \sqrt{\sum_{i=z+1}^{J_{max}} b_i^2}. \tag{3.34}$$

Afterwards, each line in the system is multiplied by the ENO weight of the corresponding cell V_j from (3.32) which yields a new system of equations including both geometrical and ENO weighting

$$w_{ij} w_{ij}^{ENO} b_j = \sum_{k=1}^{K} w_{ij} w_{ij}^{ENO} \mathcal{A}_{jk} a_k, \quad j = 1, ..., J_{max}. \tag{3.35}$$

The final solution for a_k is obtained by solving (3.35) by another $\mathcal{QR}$-decomposition and back-substitution.

3.6 Boundary Conditions

A special requirement for ensuring high-order solutions is the introduction of boundary conditions in the reconstruction process. These conditions are enforced by adding them to the least squares problem as constraints and solving them by direct elimination during the upper triangulation. All stencils with boundary cells could theoretically include such constraints. However, the experience shows improved results for using boundary conditions just for target cells which are located next to a wall boundary. In accordance with Tsoutsanis [29], they are satisfied just at the boundary face centres to prevent a over-constraint system. A Dirichlet boundary condition $\Phi(\vec{x}) = f_1(\vec{x})$ is satisfied at a point $\vec{x}_g$ by

$$f_1(\vec{x}_g) = p_i(\vec{x}_g) \tag{3.36}$$

$$f_1(\vec{x}_g) - \overline{\Phi}_i = \sum_{k=1}^{K} a_k \Omega_k(\vec{x}_g). \tag{3.37}$$

Neumann boundary conditions $\dfrac{\partial \Phi(\vec{x})}{\partial n} = f_2(\vec{x})$, with $\vec{n}$ the outward normal vector of the wall, yield at the point $\vec{x}_g$

$$\begin{aligned} f_2(\vec{x}_g) &= \frac{\partial p_i(\vec{x}_g)}{\partial n} = \nabla p_i(\vec{x}_g)\vec{n} \\ &= \sum_{k=1}^{K} a_k \cdot \left(n_x \frac{\partial \Omega_k(\vec{x}_g)}{\partial x} + n_y \frac{\partial \Omega_k(\vec{x}_g)}{\partial y} + n_z \frac{\partial \Omega_k(\vec{x}_g)}{\partial z} \right). \end{aligned} \tag{3.38}$$

Since OpenFOAM is designed for second-order accuracy at most, values at boundaries with a Neumann boundary condition are calculated as [20]

$$\Phi(\vec{x}_g) = \Phi(\vec{x}_i) + f_2(\vec{x}_g) \cdot (\vec{x}_i - \vec{x}_g)\,, \tag{3.39}$$

whereas the correct implementation for achieving a higher order sets the values from the resulting polynomial of V_i. The code is not changed within this project although it results in a nominal decrease of the order of accuracy near boundaries. However, it can be taken into account for future investigations.

The resulting system of equations is a linear least squares problem of minimizing $\|\mathcal{A} \cdot a - b\|_2$ as above but with the equality constraint $\mathcal{C} \cdot a = d$. Thus, the problem leads to

$$\begin{pmatrix} \mathcal{C} \\ \mathcal{A} \end{pmatrix} a = \begin{pmatrix} d \\ b \end{pmatrix}, \tag{3.40}$$

with the partitioning

$$\begin{pmatrix} \mathcal{C} \\ \mathcal{A} \end{pmatrix} = \begin{pmatrix} \mathcal{C}_1 & \mathcal{C}_2 \\ \mathcal{A}_1 & \mathcal{A}_2 \end{pmatrix}. \tag{3.41}$$

The simplest way of solving (3.40) is using a heavy weight for the constraint lines and solve the system afterwards as described in chapter 3.4. As a result, it may lead to poorly conditioned matrices and a decrease of the solutions accuracy. The better choice is direct elimination of the constraints during the transformation process as described by Lawson [10]. For this purpose, the matrix $\mathcal{C}_1$ is triangulated by Householder transformations with the algorithm of chapter 3.4. Next, $\mathcal{A}_1$ is zeroed using Gaussian elimination steps and finally, $\mathcal{A}_2$ is triangulated by further Householder transformations. The resulting matrix has an upper triangular form and can be handled with the implemented algorithm for rank-deficient matrices. More details are provided in Lawson's book [10].

3.7 Parallelisation

In this chapter an outline of the parallelisation of the code is given. It seems to be a crucial step due to the time consuming reconstruction process. By default, OpenFOAM uses a 0-halo approach which divides the domain into several non-overlapping regions and Message passing interface (MPI) to transmit the information between the inter-processor boundaries. This leads to at best second-order accurate solutions at such boundaries [22]. In contrary, the stencils near processor patches of a high-order (W)ENO scheme need the geometrical and physical data from several layers of the neighbouring processors. Consequently, a n-halo approach with several overlapping sub-domains would be the proper choice. The implicit handling of the Navier-Stokes equations leads to algebraic systems of equations which are solved by linear, iterative solvers in OpenFOAM. Since these solvers just work for 0-halo approaches, a n-halo approach is discarded. The solution is the virtual extension of the sub-domains by collecting halo cells from neighbouring processors in additional lists. Besides, the field values of the halo cells are update in the beginning of each runtime step which is computed on non-overlapping domains.

The initialization of parallelisation starts in the preprocessing step after the local stencils are collected. At that point, several stencils with a deformed shape exist near processor boundaries such as the blue framed stencil in figure 3.2a. Appropriate halo cells from other processors provide the necessary correction of the stencils. All possible stencils with a deformed shape and therefore recipients for halo cells are included in the stencils of target cells next to processor boundaries. In addition, it implies that recipient cells are vice versa the only possible halo cells for stencils of other processors. Hence, the halo cells do not have to be collected separately but can just be taken from the prepared local stencil lists. This leads to the following modification for the stencil collection algorithm in the case of using several processors:

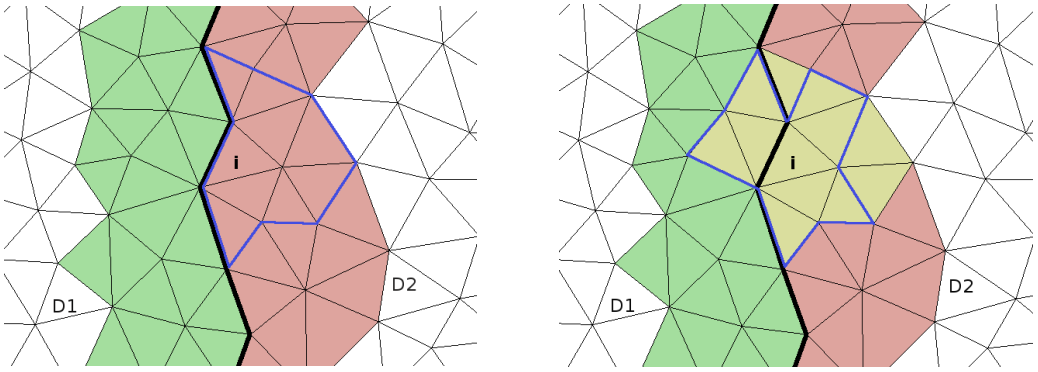
1. For each sub-domain D_i, all cells from the stencils of target cells next to a processor boundary are gathered in a list of halo cells together with the information of the target processor. Beyond, the stencils of these cells are marked as the possible recipients for halo cells from other sub-domains. In figure 3.2a, recipient cells of the sub-domain D_1 are coloured green while its halo cells from sub-domain D_2 are coloured red, and vice versa the green cells are the halo cells from sub-domain D_1 for the red recipient cells of D_2.

2. The lists of halo cells are further prepared by assigning them a new ID and additionally storing their cell center coordinates. Afterwards, the lists are transmitted to the appropriate target processor using MPI.

3. The required halo cells for each marked stencil S_i are determined by a geometrical selection due to missing face neighbour information through processor boundaries. For this purpose, a sphere is spanned around the target cell V_i of S_i with the distance from the center of V_i to the most outer cell center in S_i as the radius. All halo cells whose cell centres are located within this sphere are added to S_i. In figure 3.2b, this geometrical selection results in the yellow coloured global stencil for the originally blue framed stencil.

4. The final stencils are attained from sorting the global stencils by distance and pick the nearest J_{max} cells. In figure 3.2b, the new stencil is framed in blue.

Additional lists with the information of the origin processor of each cell in a stencil are generated in order to transmit any further data, such as volume integrals, between processors. These lists are also used to transmit the field data between processors before the local reconstruction starts in each runtime step.



(a) Local stencil (b) Global stencil

Figure 3.2: Correct stencil collection at a processor boundary

3.8 Practical Operations

The reconstruction of any function in multiple dimensions and on unstructured meshes can be computed using the above presented Quasi-ENO scheme. The computation is divided into a preprocessing and a runtime part. For preprocessing, the input is any field of variable Φ represented by the averaged value within each cell of the domain $\overline{\Phi}_i$. The following preprocessing steps are executed out of that:

1. Generation and storing of a central stencil list for each cell V_i. If several processors are included, halo cells are considered.

2. Calculation and storing of all volume integrals of the basis functions for each V_i from (3.17) using triangulation of the faces and Gaussian quadrature formulas.

3. Computation and storing of a reconstruction matrix $\mathcal{A}$ for each stencil S_i from (3.13) including geometrical weighting. If the target cell is next to a wall boundary, the appropriate boundary condition is written in the first lines of $\mathcal{A}$.

The most relevant lists of the preprocessing step are additionally written as files and stored in the `constant` folder which reduces the time for preprocessing if calculations

on the same mesh are executed several times. In runtime, the remaining steps for obtaining the solution at each time step are:

1. For each S_i, generating the vector b from $\overline{\Phi}$ as the right hand side of (3.10) and introducing the geometrical weighting.

2. Solving the resulting least-squares problem by $\mathcal{QR}$-decomposition with Householder transformations until the residuum can be calculated using (3.34).

3. Computing the ENO-weights with (3.32) and solving the resulting system (3.35) for obtaining the degrees of freedom a_k. Finally, the reconstructed value at any point in V_i can be calculated using the polynomial representation (3.2).

Chapter 4

Verification and Performance

The verification process has to demonstrate the correlation of the numerical results to their theoretically intended results [24]. For this purpose, the order of accuracy of the reconstruction is determined by a convergence study using different meshes and smooth functions in 2D and 3D. Furthermore, the capacities of the smoothness indicator is investigated for non-smooth functions.

4.1 Accuracy of Reconstruction

The order of accuracy α corresponds to the relation between the $\mathcal{L}_p$-norm of the error and different grid resolutions. α is represented as the slope of the line through the corresponding points of error and mesh size in a log-log plot. Knowing the error norm for two grids $\mathcal{L}_{p1}$ and $\mathcal{L}_{p2}$ with the characteristic sizes Δ_1 and Δ_2, the order of accuracy α is obtained as

$$\alpha = \frac{\log\left(\frac{\mathcal{L}_{p2}}{\mathcal{L}_{p1}}\right)}{\log\left(\frac{\Delta_2}{\Delta_1}\right)}. \tag{4.1}$$

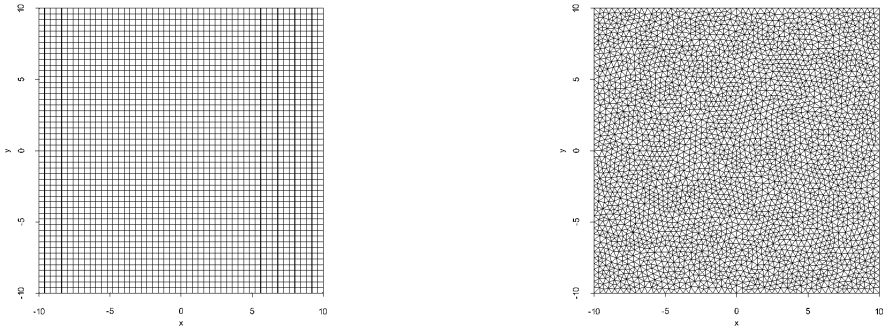
The characteristic size is taken as $\sqrt[3]{N}$ in 3D and $\sqrt{N}$ in 2D with N the total number of cells in the domain. The grids are generated without changing the topology during the

refinement steps to prevent the results from additional scaling effects. The considered error norms are calculated as

$$\mathcal{L}_2 = \sqrt{\frac{1}{|V_T|}\sum_{i=1}^{N}\iiint_{V_i}\left(p_i(\vec{x}) - \Phi_{exact}(\vec{x})\right)^2 \,\mathrm{d}x\mathrm{d}y\mathrm{d}z}$$
$$\mathcal{L}_\infty = \max_i\left(\frac{1}{|V_i|}\iiint_{V_i}\left(p_i(\vec{x}) - \Phi_{exact}(\vec{x})\right) \,\mathrm{d}x\mathrm{d}y\mathrm{d}z\right), \quad i = 1,..N, \tag{4.2}$$

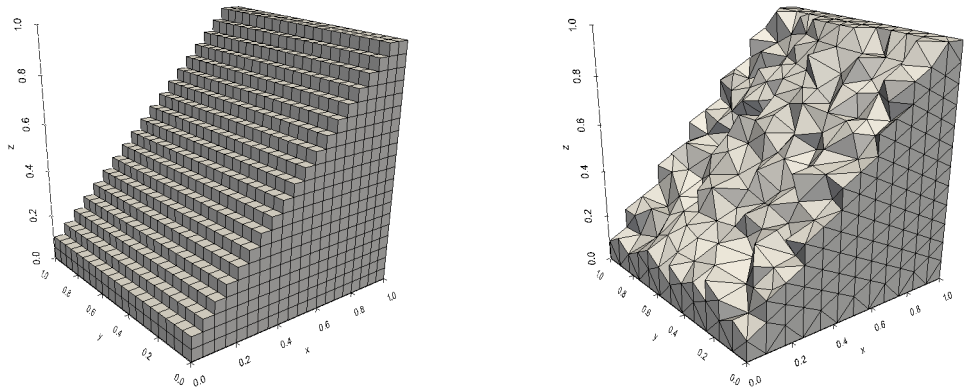
with $|V_T|$ the total volume of the domain and $\Phi_{exact}(\vec{x})$ the known exact solution in $|V_T|$. The polynomials $p_i(\vec{x})$ are computed from (3.2). The volume integrals in (4.2) are evaluated by a transformation into surface integrals and the use of Gaussian quadrature rules from above with a higher order than in the reconstruction. For this purpose, an integration of the differences over one of the coordinates has to be executed analytically. This can be computed by any mathematical software package for sufficient simple functions.

The considered grids are hexahedral and unstructured (triangular prisms) for two-dimensional calculations, and hexahedral and unstructured (tetrahedral, polyhedral) for three-dimensional calculations. The tetrahedral and polyhedral meshes are generated using StarCCM+ and Gmsh. All meshes are presented in form of their coarsest grid resolution in figure 4.1 and 4.2. The triangulation fails several times for the polyhedral meshes since it contains many concave cells and non-flat faces, which leads to an inaccurate reconstruction. The scheme is still able to get any solution but without convergence. Therefore, it is decided to take these meshes away from the verification and instead, suggest to pay heed to convexity and flatness of the generated cells. Furthermore, as it can be seen in figure 4.2b, the tetrahedral mesh is not regular, which is challenging for any scheme.

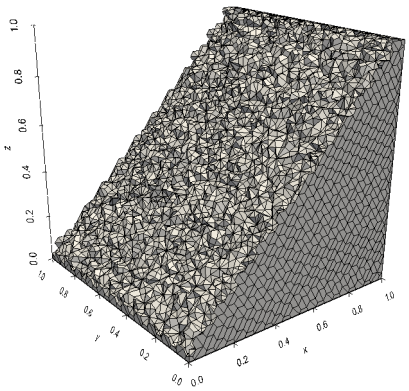


(a) 2,500 hexahedra (b) 6,172 triangular prisms

Figure 4.1: Two-dimensional meshes for the verification process



(a) 8,000 hexahedra (b) 7,982 tetrahedra



(c) 9,327 polyhedra

Figure 4.2: Three-dimensional meshes for the verification process

4.1.1 Smooth Function in 2D

First, a smooth, two-dimensional function is reconstructed using the Quasi-ENO scheme with polynomials of first- to fourth-order accuracy. No analytical integration has to be executed since the integrals in (4.2) simplify to surface integrals for two-dimensional cases. The considered function is taken from McDonald [13] which offers the possibility of comparing the results to another high-order approach. The so-called CENO scheme uses a least squares reconstruction and a limiter in order to prevent oscillations. The function is taken as

$$\Phi_{exact}(x,y) = 1 + \frac{1}{3} \cdot \cos\left(\sqrt{x^2+y^2}\right). \tag{4.3}$$

The solution is computed on a quadratic domain with an edge length of 20 and centred on $x = y = 0$. In figure 4.3, the initial data is shown on a structured grid with $2,500$ hexahedra.

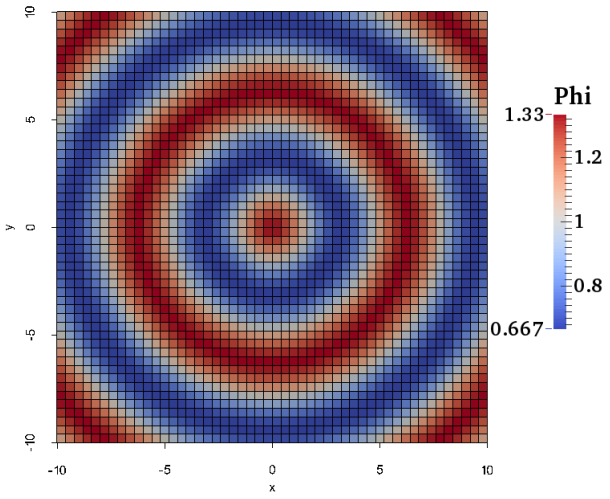
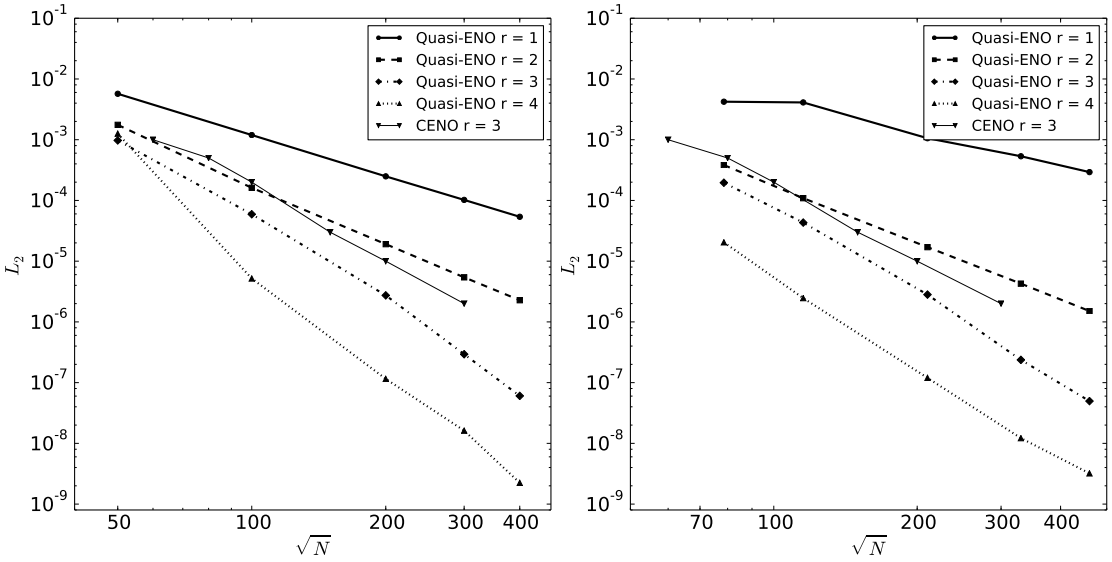


Figure 4.3: Initial data for the reconstruction of the smooth, two-dimensional function

The results of the reconstructions are shown in figure 4.4 where the $\mathcal{L}_2$-norm of the error is plotted over the characteristic size. As expected, the accuracy of the scheme increases with raising polynomial order and grid resolution. Further, the results are almost

the same for both cell shapes except for first-order. The fourth-order accurate CENO scheme from calculations on unstructured grids is outperformed by the corresponding Quasi-ENO scheme which indicates the conservation of an accurate smooth solution by the ENO weighting in comparison to limiting strategies. The order of accuracy is evaluated for the $\mathcal{L}_2$-norm of the error under consideration of (4.2) and presented in table 4.1. Obviously, the scheme reaches the nominal convergence rate and even exceeds it on two-dimensional, hexahedral grids.



(a) hexahedra (b) triangular prisms

Figure 4.4: $\mathcal{L}_2$-norm of the error for the smooth, two-dimensional function

Table 4.1: Order of accuracy from the numerical convergence study for the smooth, two-dimensional function on hexahedral α_{hex} and unstructured α_t grids using $\mathcal{L}_2$

Polynomial order r	α_{hex}	α_t
1	2.2	1.5
2	3.2	3.1
3	4.6	4.6
4	6.3	4.9

4.1.2 Smooth Function in 3D

A smooth, three-dimensional function is reconstructed using polynomials of first- to third-order. The function is harmonically with changes in all three coordinate directions

$$\Phi_{exact}(x,y,z) = y \cdot \cos(4x) + z \cdot \sin(10y) + x \cdot \cos(3z). \tag{4.4}$$

The solution is computed on a cubic domain with an edge length of 1 and centred on $x = y = z = 0.5$. The initial data on a hexahedral grid with $8,000$ hexahedra is shown in figure 4.5.

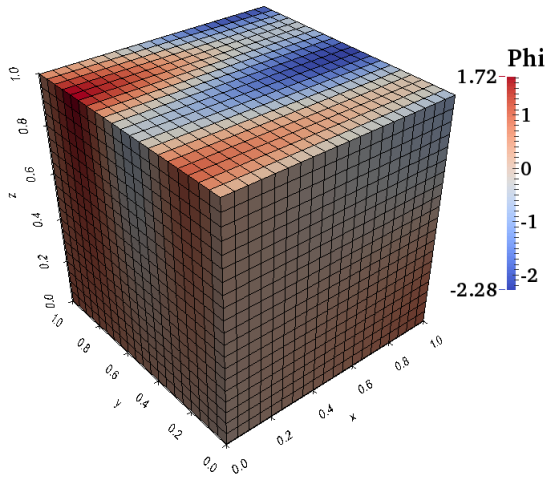
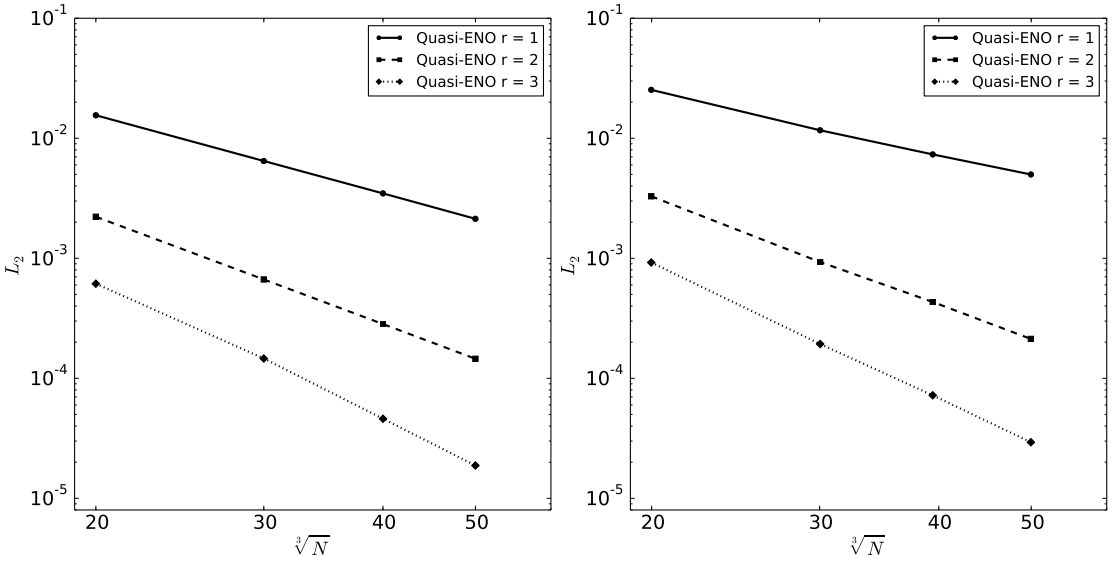


Figure 4.5: Initial data for the reconstruction of the smooth, three-dimensional function

In figure 4.6, the resulting $\mathcal{L}_2$-norm of the error is shown for increasing grid resolutions. Generally, the scheme provides the expected distribution of the norm for increasing order and resolution. In contrast to the two-dimensional function, the results on the structured grids are slightly improved in comparison to the ones on the tetrahedral grids. For the sake of completeness, the $\mathcal{L}_2$-norm of the error for polyhedral meshes arranges between 10^{-1} and 10^{-3} which is still acceptable in view of the poor mesh quality and shows the strength of the scheme for arbitrary meshes. The evaluation of the order of accuracy from the convergence study of the $\mathcal{L}_\infty$- and $\mathcal{L}_2$-norm of the error is shown in table 4.2 and 4.3. The nominal order can be reached for both errors on structured grids while on the unstructured ones the scheme tends to slightly smaller orders. Although they are still in an acceptable range, this problem indicates a general problem of the scheme for higher order polynomials in 3D. It may arise on irregular meshes from local scaling effects and deformed stencils as well as on regular meshes from many collinear cells and less compact stencils near boundaries.



(a) hexahedra (b) tetrahedra

Figure 4.6: $\mathcal{L}_2$-norm of the error for the smooth, three-dimensional function

Table 4.2: Order of accuracy from the numerical convergence study for the smooth, three-dimensional function on hexahedral α_{hex} and tetrahedral α_{tet} grids using $\mathcal{L}_\infty$

Polynomial order r	α_{hex}	α_{tet}
1	1.95	1.74
2	2.91	3.09
3	3.84	3.56

Table 4.3: Order of accuracy from the numerical convergence study for the smooth, three-dimensional function on hexahedral α_{hex} and tetrahedral α_{tet} grids using $\mathcal{L}_2$

Polynomial order r	α_{hex}	α_{tet}
1	2.17	1.77
2	2.97	2.99
3	3.96	3.76

4.1.3 Non-smooth Function

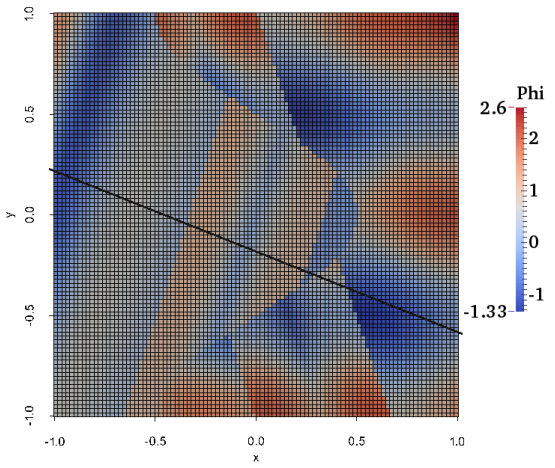
In the following, the reliability of the smoothness indicator is investigated by reconstructing a non-smooth function. In accordance with Gooch [17], Abgrall's function [1] is considered which is 2D and has several discontinuities. The initial data field can be found in figure 4.7a and 4.8a respectively. The function is calculated on a square with edge length 2, centred on $x = y = 0$, as [17]

$$\Phi_{exact}(x,y) = \begin{cases} f(x - y \cdot \cot\left(\sqrt{\frac{\pi}{2}}\right)) & \text{if} \quad x \leq \frac{\cos(\pi y)}{2} \\ f(x + y \cdot \cot\left(\sqrt{\frac{\pi}{2}}\right)) + \cos(2\pi y) & \text{else} \end{cases}, \tag{4.5}$$

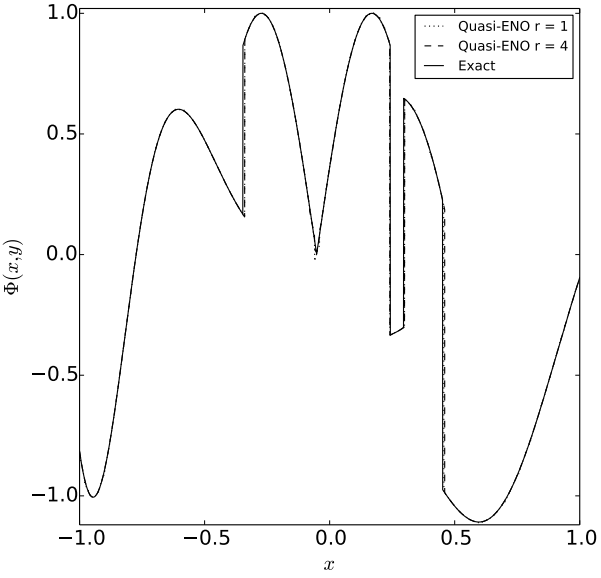
with

$$f(r) = \begin{cases} -r \cdot \sin\left(\dfrac{3\pi r^2}{2}\right) & if \quad r \leq -\frac{1}{3} \\ |\sin(2\pi r)| & if \quad |r| < \frac{1}{3} \\ 2r - 1 + \frac{1}{6} \cdot \sin(3\pi r) & if \quad r \geq \frac{1}{3} \end{cases} . \tag{4.6}$$

It is decided to verify the reconstruction graphically due to the difficulty of integrating a function through discontinuities in multiple dimensions and the focus of interest on the non-oscillatory behaviour. The results in figure 4.7b and 4.8b show the exact and reconstructed solution on a slice between the two points $(-1, 0.22)$ and $(1, -0.58)$. In the course of evaluation, grids with $2,500$ up to $90,000$ hexhedra and triangular prisms respectively are investigated using polynomials of first- to fourth-order. The presented results are taken to represent the typical behaviour since no significant differences occur for the different mesh sizes. On both grids, the scheme shows non-oscillatory behaviour irrespectively to the resolution. In the case of structured grids, the lines of the reconstructed solutions are slightly shifted near discontinuities through the discrete representation of jumps. The first-order polynomial overshoots the point of discontinuity near $x = 0$. For the unstructured grid, the first-order solution undershoots this point and has a small tip at $x = 0.3$. Increasing the order reduces the inaccuracy and presents the expected non-oscillatory behaviour for all considered orders of the polynomials.

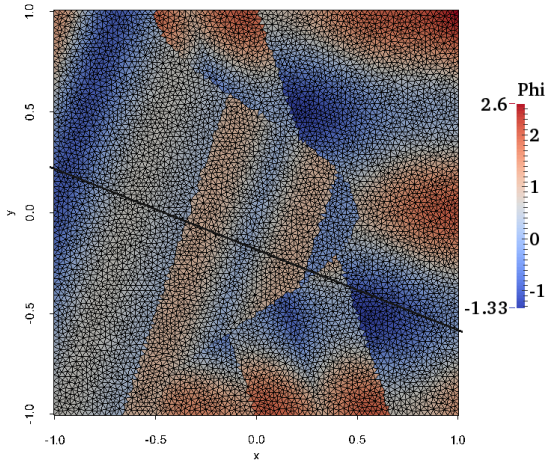


(a) Initial data

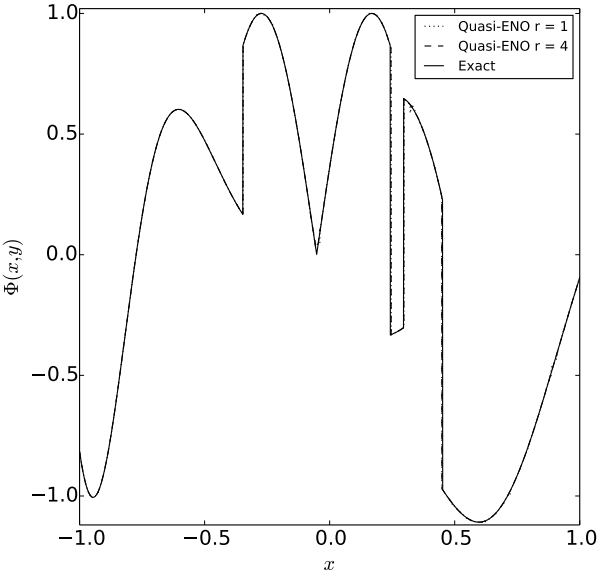


(b) Reconstructed solution

Figure 4.7: Abgrall's function for a grid with $10,000$ hexahedra



(a) Initial data



(b) Reconstructed solution

Figure 4.8: Abgrall's function for a grid with $13,360$ triangular prisms

4.2 Efficiency of the Implementation

The verification of the Quasi-ENO scheme shows its principal functioning for the reconstruction of smooth and non-smooth functions in multiple dimensions and on arbitrary grids. It provides a non-oscillatory behaviour near discontinuities and a high-order reconstruction in smooth regions. In practice, the increase of accuracy is counterbalanced by an increase of computational costs. Table 4.4 shows the times for preprocessing and reconstructing the three-dimensional, smooth function from above on an unstructured grid with $124,693$ tetrahedra on a single processor. It should be noticed that the preprocessing has to be executed just once for a mesh due to writing of relevant lists in files, which accelerates this step in further calculations. The runtime seems to be efficient for first and second-order polynomials while the third-order reconstruction takes more than $5\ s$. The main reason for the reduced efficiency is the enforcement of solving the system of equations in each time step twice. However, these measurements show the proper quality of implementation under consideration of the original implementation of Gooch [17].

The investigation of the performance points out the necessity to parallelise the code. Evaluating the same case as above but on two processors shows the increased efficiency. As it is shown in table 4.5, the times decreases up to 50% in both preprocessing and runtime. It is not exactly halved due to the modified stencil collecting algorithm in preprocessing and the data transmission in runtime. In order to complete the investigation of parallel computation, the $\mathcal{L}_2$-norm of the error of the reconstruction is compared to a computation on a single processor in table 4.6. The results are minimally different due to the slightly different stencils. The stencil collecting algorithm through processor boundaries uses geometrical searching while on a single processor just the face neighbours are gathered.

Polynomial order r	preprocessing	runtime
1	5.73	0.23
2	26.70	1.22
3	104.37	5.36

Table 4.4: Time measurements for the reconstruction of a three-dimensional, smooth function for $124,693$ tetrahedra (measured in seconds on Intel i5-4430 CPU, 8 GB memory, Linux Ubuntu 64-bit)

Polynomial order r	preprocessing	runtime
1	3.11	0.14
2	13.40	0.62
3	54.78	3.31

Table 4.5: Time measurements for the reconstruction of a three-dimensional, smooth function for $124,693$ tetrahedra on two processors (measured in seconds on Intel i5-4430 CPU, 8 GB memory, Linux Ubuntu 64-bit)

Polynomial order r	$\mathcal{L}_{2,s}$	$\mathcal{L}_{2,t}$
1	$4.99 \cdot 10^{-3}$	$4.98 \cdot 10^{-3}$
2	$2.13 \cdot 10^{-4}$	$2.05 \cdot 10^{-4}$
3	$2.93 \cdot 10^{-5}$	$2.95 \cdot 10^{-5}$

Table 4.6: Comparison of the $\mathcal{L}_2$-norm of the error for the reconstruction of a three-dimensional, smooth-function for $124,693$ tetrahedra on a single $\mathcal{L}_{2,s}$ and two processors $\mathcal{L}_{2,t}$

Chapter 5

Application to the Level Set Equation

The successful implementation of the Quasi-ENO scheme is shown in the last chapter. The final task is the application of the reconstruction for the flux evaluations within the solving algorithm for the level set equation. This equation arises in the level set method for capturing free surfaces in multiphase flows. In terms of physical coordinates, this equation is obtained as

$$\frac{\partial \varphi}{\partial t} + u_x \frac{\partial \varphi}{\partial x} + u_y \frac{\partial \varphi}{\partial y} + u_z \frac{\partial \varphi}{\partial z} = 0. \tag{5.1}$$

Under consideration of chapter 2, equation (5.1) is characterized as a linear hyperbolic conservation law. Thus, its representation in FVM is analogous to (2.6)

$$\frac{\mathrm{d}}{\mathrm{d}t}\overline{\varphi}_i + R_i = 0 \tag{5.2}$$

$$R_i = \frac{1}{|V_i|} \sum_{l=1}^{L_i} \int_{F_l} F_{n_l}(\varphi^-, \varphi^+)\, \mathrm{d}F_l, \tag{5.3}$$

with F_{n_l} the flux normal to the face F_l. A comparison with (5.1) yields

$$F_{n_l} = \vec{F} \cdot \vec{n}_l = \begin{pmatrix} u_x \varphi \\ u_y \varphi \\ u_z \varphi \end{pmatrix} \cdot \vec{n}_l = (\vec{u}\vec{n}_l) \cdot \varphi = u_{n_l} \varphi, \tag{5.4}$$

which represents the whole flux due to the rotational invariance of (5.1). The proof is provided by Toro [28]. Since the flux depends on both φ^-, obtained from the polynomial interpolation of $|V_i|$, and φ^+, the solution of the neighbouring cell of the face $|V_{jl}|$, a local Riemann problem has to be solved at the discontinuity.

The temporal discretisation of (5.2) is performed explicitly using a TVD third-order Runge-Kutta method. Thus, the progress in time from time step n to $n+1$ for any cell V_i is [11]

$$\begin{aligned}
\varphi_i^{(n,1)} &= \varphi_i^{(n)} - \Delta t \cdot R_i(\varphi_i^{(n)}) \\
\varphi_i^{(n,2)} &= \frac{3}{4}\varphi_i^{(n)} + \frac{1}{4}\varphi_i^{(n,1)} - \frac{\Delta t}{4} \cdot R_i(\varphi_i^{(n,1)}) \\
\varphi_i^{(n+1)} &= \frac{1}{3}\varphi_i^{(n)} + \frac{2}{3}\varphi_i^{(n,2)} - \frac{2\Delta t}{3} \cdot R_i(\varphi_i^{(n,2)}).
\end{aligned} \tag{5.5}$$

A Runge-Kutta method of even higher order is not considered here because the time-consuming reconstruction step has to be executed within each sub step anyway. In order to achieve arbitrary high-order in space and time, the ADER approach (Arbitrary high order schemes using derivatives) of Toro et al. [25] seems to be a better choice. Dumbser [3] applied this approach successfully for solving linear hyperbolic systems on unstructured grids in 3D. The time step in (5.5) has to be chosen without violating the stability of the scheme. A proper determination for a linear hyperbolic equation on structured grids is given by Toro [26] as

$$\Delta t = CFL \cdot \min_i \left(\frac{\Delta x_i}{|u_{xi}|}, \frac{\Delta y_i}{|u_{yi}|}, \frac{\Delta z_i}{|u_{zi}|} \right), \quad i = 1, ..N, \tag{5.6}$$

at which (W)ENO schemes are stable for $0 < CFL \leq \frac{1}{3}$ in 3D [26]. For unstructured grids, Δ_i is taken as $|V_i|^{\frac{1}{3}}$ and the normal velocity at the faces as $|u_i|$.

5.1 Solving the Riemann Problem

The level set equation (5.1) may be rewritten as an 1D-formulation in the direction of n_l under consideration of the rotational invariance

$$\frac{\partial \varphi}{\partial t} + \frac{\partial F_{n_l}}{\partial n_l} = 0, \tag{5.7}$$

which results in the Riemann problem for a linear advection equation [28]

$$\left.\begin{aligned} PDE: \quad & \frac{\partial \varphi}{\partial t} + u_{n_l} \frac{\partial \varphi}{\partial n_l} = 0, \\ IC: \quad & \varphi(n_l, 0) = \varphi_0(n_l) = \begin{cases} \varphi^- & if \quad n_l < 0 \\ \varphi^+ & if \quad n_l > 0 \end{cases} \end{aligned}\right\}. \tag{5.8}$$

In figure 5.1a the initial conditions of (5.8) are shown. The initial data shows a discontinuity at the face $n_l = 0$ resulting from the two polynomial reconstructions from two different cells. Progressing in time, the discontinuity at $n_l = 0$ propagates with the characteristic speed u_{n_l}. This is indicated by the characteristic curve $n_l = u_{n_l} t$ representing the locations without changes in φ. Hence, the exact solution of (5.8) is given as (see figure 5.1b and [28])

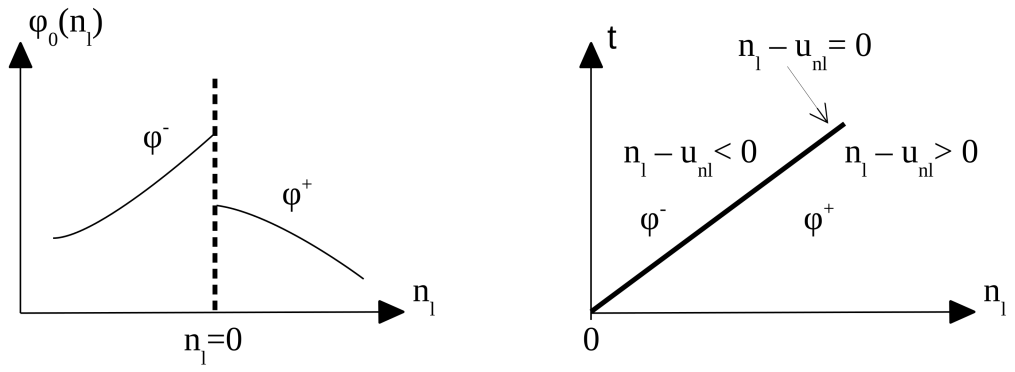
$$\varphi(n_l, t) = \varphi_0(n_l - u_{n_l} t) = \begin{cases} \varphi^- & if \quad n_l - u_{n_l} t < 0 \\ \varphi^+ & if \quad n_l - u_{n_l} t > 0 \end{cases}. \tag{5.9}$$

At any time $t > 0$, the flux from (5.4) yields at $n_l = 0$

$$F_{n_l} = \begin{cases} u_{n_l} \varphi^- & if \quad u_{n_l} > 0 \\ u_{n_l} \varphi^+ & if \quad u_{n_l} < 0 \end{cases}. \tag{5.10}$$

Following this procedure, the implemented scheme performs as a non-oscillatory high-order upwind scheme for the level set equation [27]. In practice, the result of (5.10) may be combined to one simple formulation for the flux [3, 29]

$$F_{n_l}(\varphi^-, \varphi^+) = \frac{1}{2} \cdot \left((u_{n_l} + |u_{n_l}|) \cdot \varphi^- + (u_{n_l} + |u_{n_l}|) \cdot \varphi^+ \right). \tag{5.11}$$



(a) Initial conditions (b) Solution

Figure 5.1: Illustrations of the Riemann problem

5.2 Computation of the Numerical Flux

Introducing the final formulation for the fluxes (5.11) in (5.3) leads to

$$R_i = \frac{1}{2|V_i|} \sum_{l=1}^{L_i} \left((u_{n_l} + |u_{n_l}|) \int_{F_l} \varphi^- \, \mathrm{d}F_l + (u_{n_l} - |u_{n_l}|) \int_{F_l} \varphi^+ \, \mathrm{d}F_l \right). \tag{5.12}$$

The velocities can be taken out of the integrals due to the linearity of the equation. Further simplifications are obtained by replacing the interpolated values of φ on F_l by their polynomial representation (3.2) which yields at a time step n

$$\begin{aligned} \varphi^-(\vec{x},t) &= \overline{\varphi}_i^{(n)} + \sum_{k=1}^{K} a_{k,i}^{(n)} \Omega_{k,i} \\ \varphi^+(\vec{x},t) &= \overline{\varphi}_{jl}^{(n)} + \sum_{k=1}^{K} a_{k,jl}^{(n)} \Omega_{k,jl}. \end{aligned} \tag{5.13}$$

Using (5.13), the surface integrals in (5.12) are rewritten as e.g.

$$\int_{F_l} \varphi^- \, \mathrm{d}F_l = |F_l| \overline{\varphi}_i^{(n)} + \sum_{k=1}^{K} \left(a_{k,i}^{(n)} \int_{F_l} \Omega_{k,i} \, \mathrm{d}F_l \right). \tag{5.14}$$

The resulting integrals on the right hand side are solution independent and can be pre-computed. Under consideration of (3.6), they are obtained as

$$\int_{F_l} \Omega_{k,i} \, \mathrm{d}F_l = \int_{A_l} \Psi_k \, \mathrm{d}F_l - \frac{|F_l|}{|V_i|} \int_{V_i} \Psi_k \, \mathrm{d}V_i. \tag{5.15}$$

The volume integrals in (5.15) are already computed during the reconstruction procedure. The surface integrals over the basis functions are evaluated in a similar way by decomposing the faces into triangles and using Gaussian quadrature rules of appropriate order. Since both cells sharing F_l have the same basis, the surface integrals have to be computed just once for each face in the domain.

5.3 Results for Test Cases

Two-dimensional Results

First, the solver is tested on two-dimensional, structured, and unstructured grids with $2,500$ and $2,832$ cells respectively. The convection test cases of Jasak [8] which are generated to investigate the numerical diffusion of different convection schemes are chosen. For this purpose, the left boundary of a squared domain is taken as an inlet with a predefined profile as shown in figure 5.2. Three different profiles are considered, namely a *sin*-profile

$$\varphi(x=0,y) = \begin{cases} \sin^2\left(3\pi\cdot(y-\frac{1}{6})\right) & \text{if} \quad \frac{1}{6}\leq y\leq\frac{1}{2} \\ 0 & \text{else} \end{cases}, \tag{5.16}$$

an *ellipse*-profile

$$\varphi(x=0,y) = \begin{cases} \sqrt{1-\left(\frac{y-\frac{1}{3}}{\frac{1}{6}}\right)^2} & \text{if} \quad \frac{1}{6}\leq y\leq\frac{1}{2} \\ 0 & \text{else} \end{cases}, \tag{5.17}$$

and a *step*-profile resulting in the transport of $\varphi = 0.5$ as a physical surface

$$\varphi(x=0,y) = \begin{cases} 1 & \text{if} \quad \frac{1}{6}\leq y \\ 0 & \text{else} \end{cases}. \tag{5.18}$$

The velocity is set as $\vec{u} = (\frac{\sqrt{3}}{2}, \frac{1}{2})$ in the whole domain which leads to a transport of φ from the left to the right with an angle of 30° (see also figure 5.3). The time step is

chosen from the stability criteria from above, and the simulations are completed after steady-state solutions are obtained. The evaluation of the results is carried out graphically by plotting the steady-state solution for different polynomial orders along the y-axis at $x = 0.5$ and a comparison with the analytical results from (5.16) to (5.18).

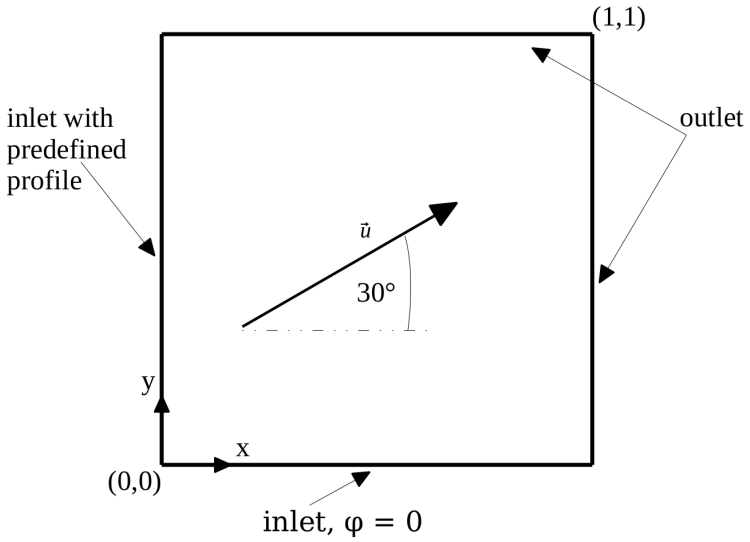
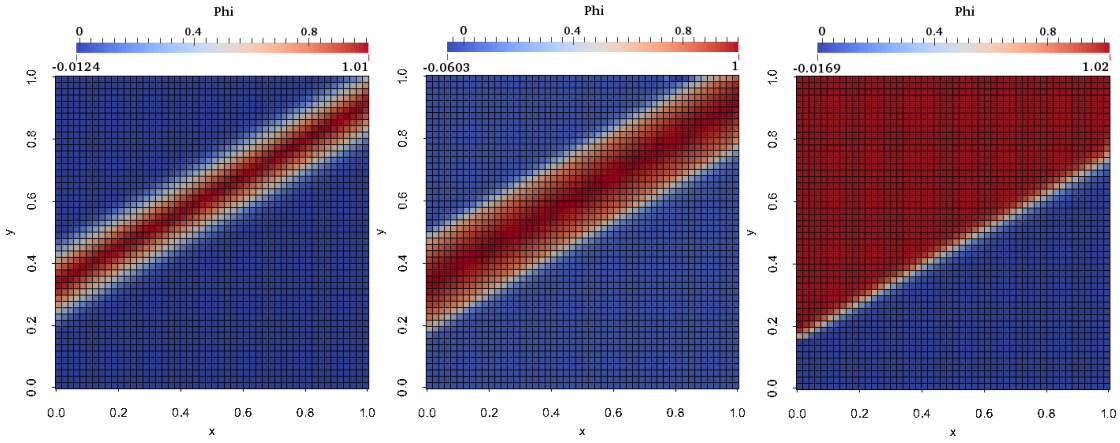


Figure 5.2: Setup for the two-dimensional test cases

In figure 5.4 the results for the *sin*-profile are presented. As can be seen from figure 5.4a, the Quasi-ENO scheme performance is accurate and non-oscillatory on the structured grid up to third-order polynomials. The peak of the profile is clipped using first-order polynomials whereas higher order reconstructions match this point accurately. At the base of the profile, the accuracy increases with the order. However, the scheme is not able to provide a stable fifth-order accurate solution near boundaries on this hexahedral grid in contrast to the unstructured one. This behaviour can be observed for all profiles which leads to the presumption that it is induced by the topology of the mesh. The analysis of ENO-weighted matrices of stencils near boundaries shows an increase of the condition number from $\approx 10^3$ for first-order polynomials to $\approx 10^{12}$ for fourth-order polynomials, equivalently to a considerable decrease of accuracy. In combination with

less compact stencils near boundaries and collinear data through the structured topology, the scheme is not able to compute a stable solution using fourth-order polynomials. An appropriate solution would particularly be a refinement of the grid. In contrast, the collinearity of cells is removed on the unstructured grids for which reason fourth-order polynomials can be applied. On the other hand, the solution on unstructured grids in figure 5.4b is generally less accurate at the peak for the same number of prismatic as hexahedral cells. Under consideration of the regarded similar number of cells for both meshes, the result corresponds to an expectable distribution because the characteristic size of a triangle is bigger than the size of a square. The mesh resolution near the peak is obviously insufficient for an accurate interpolation on this unstructured grid. Irrespectively to that, the fluxes provide a less smooth distribution of φ in comparison to them on a structured grid.

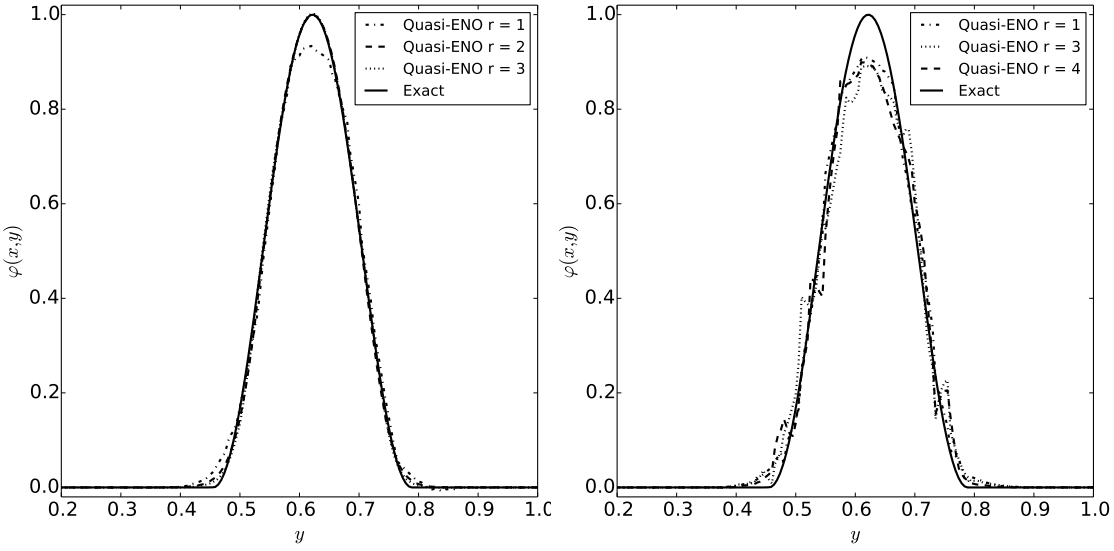


(a) *sin*-profile (b) *ellipse*-profile (c) *step*-profile

Figure 5.3: Steady-state solutions on the structured grid using 3^{rd}-order polynomials

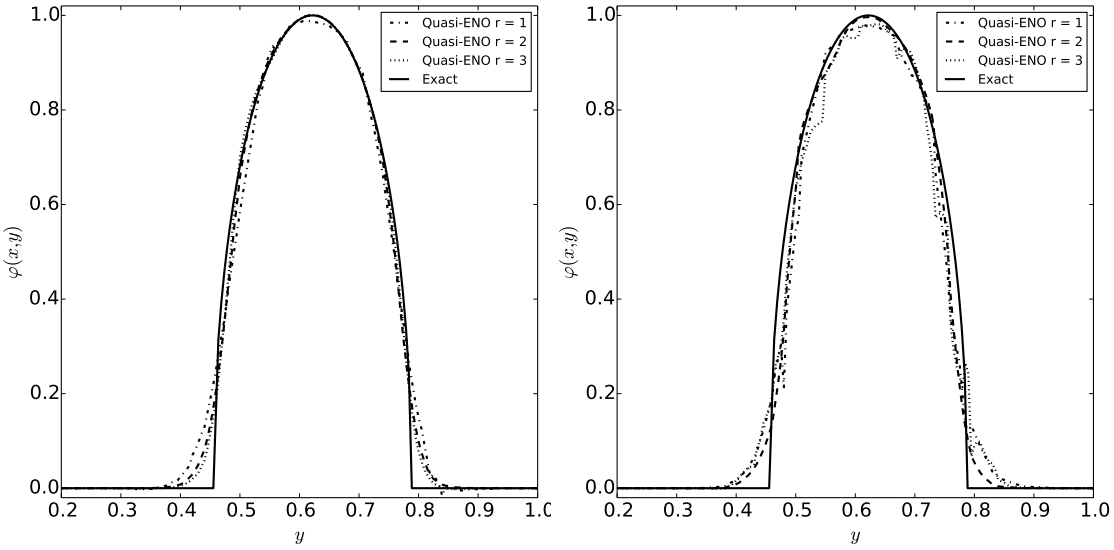
The *ellipse*-profile shows similar results for both types of meshes. As can be seen in figure 5.5, the solution is getting more accurate with increasing polynomial order on structured grids. Once again, the fourth-order reconstruction is unstable even for the unstructured grid. The best solution on this grid is obtained by the second-order recon-

struction while the third-order reconstruction falls back into a first-order reconstruction. The reason could be the increased application of low ENO-weights near the base of the profile resulting in an inaccurate computation of the degrees of freedom. In comparison to the *sin*-profile, the peak of the *ellipse*-profile is wider for which reason more cells are available for resolving this point accurately. Finally, the results for the *step*-profile are presented in figure 5.6. It is notable, that all stable computations produce non-oscillatory results. The distributions on the structured grid are getting more accurate with increasing order as seen above. For the unstructured grid, the scheme operates similar to the *ellipse*-profile. The fourth-order reconstruction operates as a first-order one and the third-order reconstruction provides the worst results. However, the solution from the fourth-order reconstruction is smoothest. As a final remark, it can be noticed from figure 5.3, that the solution is not bounded since the analytical solution postulates $0 \leq \varphi \leq 1$. It corresponds to the theory of (W)ENO schemes as Zhang [32] showed. From a practical point of view, this behaviour may break the calculation if e.g. the pressure or the density gets negative. Therefore, some kind of limiting should be introduced in order to ensure boundedness during future investigations.



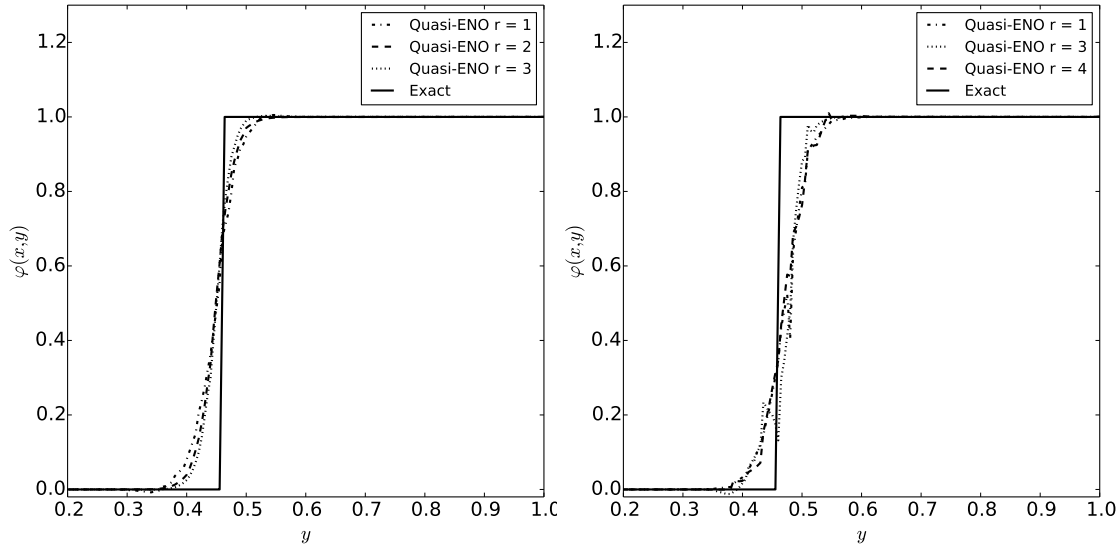
(a) 2,500 hexahedra (b) 2,832 triangular prisms

Figure 5.4: Slice of the *sinus*-profile at $x = 0.5$



(a) 2,500 hexahedra (b) 2,832 triangular prisms

Figure 5.5: Slice of the *ellipse*-profile at $x = 0.5$



(a) 2,500 hexahedra (b) 2,832 triangular prisms

Figure 5.6: Slice of the two-dimensional *step*-profile at $x = 0.5$

Step-profile in 3D

A three-dimensional case is investigated using an extension of the *step*-profile from (5.18). Here, the computation results in a three-dimensional surface at $\varphi = 0.5$ since the mesh is expanded in z-direction. It could be interpreted as a representation of the physical surface within a mass conservative level set method [19]. The investigated meshes are a structured grid with 27,000 hexahedra and an irregular, tetrahedral mesh with 27,092 cells. The resulting surface is shown in figure 5.7. Compared to the two-dimensional case, the overshoots of φ on unstructured grids are much higher resulting from inappropriate stencils in the irregular, three-dimensional mesh.

Figure 5.8 provides the plots along the line $x = z = 0.5$ on both meshes. The solution on the structured grid shows a non-oscillatory behaviour for the first-order reconstruction but gets unstable for higher orders due to less compact stencils near boundaries combined with collinear data and ENO-weighting near the interface. As a result, the

condition numbers of typical reconstruction matrices exceed 10^{20} in case of third-order polynomials. For the unstructured grid, the second-order reconstruction provides the best results while the first-order solution is similar to the corresponding solution on the structured grid. The use of third-order polynomials leads to an almost oscillatory solution on the upper corner of the step which indicates an unstable solution. The reasons are inappropriate stencils from a too coarse resolution, local scaling, and straining effects. The fluxes are less smooth on the unstructured than on the structured grid which is in accordance with the two-dimensional calculations.

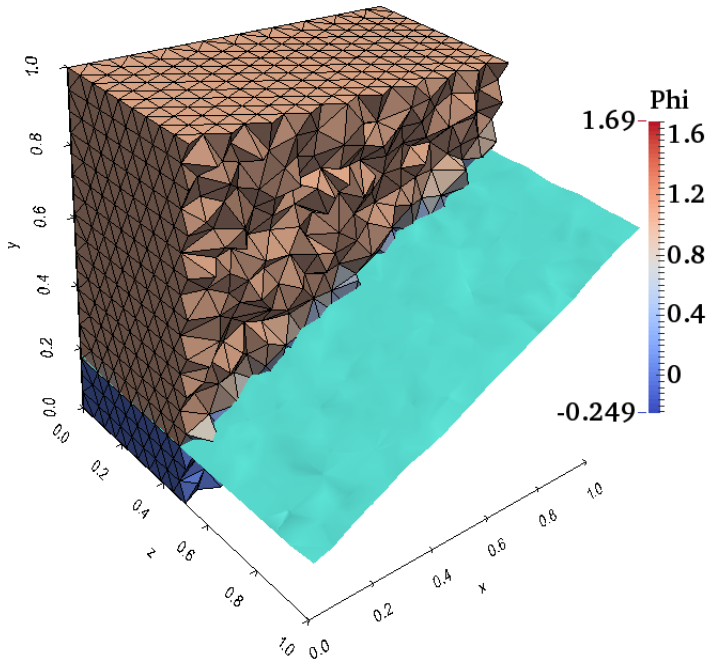
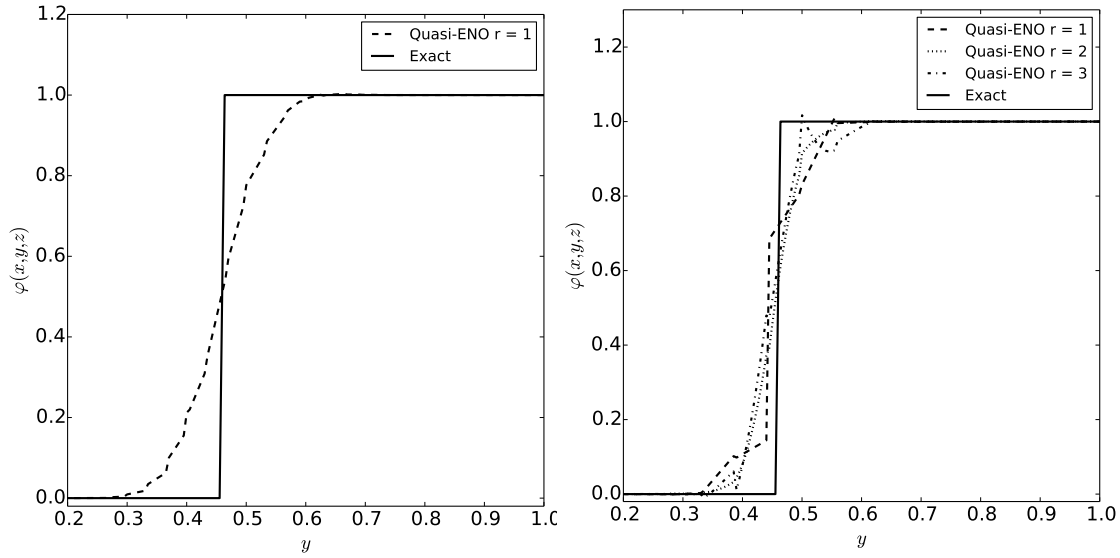


Figure 5.7: Resulting surface for the unstructured grid with 27,092 tetrahedra using 2^{nd}-order polynomials



(a) $27,000$ hexahedra

(b) $27,092$ tetrahedra

Figure 5.8: Slice of the three-dimensional *step*-profile at $x = z = 0.5$

Chapter 6

WENO Reconstruction Method

The investigation of the solution for the level set equation using the Quasi-ENO scheme shows fundamental limitations for this approach. It seems to provide accurate solutions as long as the topology allows compact stencils without scaling effects. But the solutions for the systems of equations have to be found for bad conditioned matrices, in particular for higher order polynomials, since preconditioning is not applicable for the ENO weighted reconstruction matrices. The stencils may loose too much information for reaching the nominal order of accuracy near discontinuities which results in a reduction of the order similar to TVD schemes. One opportunity for overcoming this deficit could be the inclusion of additional cells into the reconstruction. However, the compactness of the stencil has still to be ensured and further, the condition number of the matrices get worse which may lead to no improved results. Gooch [17] recommends to reduce the order of reconstruction until any nominal order can be reached which corresponds to a partially dynamic stencil searching algorithm as in common ENO-schemes. This would further reduce the performance of the scheme and drastically increase the costs in three-dimensional applications. The resulting algorithm has to work stable and efficient in 2D and 3D which is not supplied by the Quasi-ENO scheme.

Therefore, it is decided to extend the implemented scheme to a WENO scheme

because it provides a higher efficiency in runtime through computations of pseudoinverses during the preprocessing step. Furthermore, it yields better convergence due to smoother numerical fluxes [23]. The new scheme operates in a reference space without scaling effects in order to prevent bad conditioned matrices and to improve the accuracy on irregular meshes. The structure of the first implementation is preserved under consideration of the narrow connection between ENO and WENO schemes. The most relevant modifications are the computation of the degrees of freedom not just on one central stencil but on several sectoral stencils and achieving the ENO-property not by weighting the cells within a stencil but by a weighted combination of the solutions from a central and different sectoral stencils. In order to eliminate scaling effects, the whole procedure is mapped from the physical space $\vec{x}$ into a reference space $\vec{\xi}$ by an affine transformation $\vec{x} = \vec{x}(\xi, \eta, \zeta)$ and its inversion $\vec{\xi} = \vec{\xi}(x, y, z)$ respectively. In analogy to (3.2), the polynomial in V_i is represented in V_i', the cell in the new space, as

$$p_i(\vec{\xi}) = \overline{\Phi}_i + \sum_{k=1}^{K} a_k \Omega_k(\vec{\xi}). \tag{6.1}$$

The usage of an affine transformation preserves the conservation condition (3.1), which can be shown by keeping $\mathrm{d}x\mathrm{d}y\mathrm{d}z = |\mathcal{J}|\,\mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\zeta$ in mind where $\mathcal{J}$ is the later introduced Jacobian matrix

$$\begin{aligned}\overline{\Phi}_i &= \frac{1}{|V_i|}\int_{V_i} p_i(\vec{x})\,\mathrm{d}x\mathrm{d}y\mathrm{d}z = \frac{|\mathcal{J}|}{|\mathcal{J}||V_i'|}\int_{V_i'} p_i(\vec{\xi})\,\mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\zeta \\ &= \frac{1}{|V_i'|}\int_{V_i'} p_i(\vec{\xi})\,\mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\zeta. \end{aligned} \tag{6.2}$$

Thus, the same basis functions are applicable in the reference space and the resulting system of equations reads (compare (3.10) and (3.12))

$$b_j = \sum_{k=1}^{K} \mathcal{A}_{jk} a_k, \quad j = 1, \ldots, J_{max} \tag{6.3}$$

$$\mathcal{A}_{jk} = \frac{1}{|V_j'|}\int_{V_j'} \Psi_k\,\mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\zeta - \frac{1}{|V_i'|}\int_{V_i'} \Psi_k\,\mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\zeta. \tag{6.4}$$

For the sake of clarity, the integrations in (6.4) have to be computed in the reference space of the stencil of the target cell V_i'. The elements $\mathcal{A}_{jk}$ are evaluated using (3.13) in the new system. In contrast to ENO schemes, the degrees of freedom a_k are evaluated on a set of stencils U_{Si} for each cell V_i'

$$U_{Si} = \bigcup_{m=0}^{N_{Si}} S_m, \tag{6.5}$$

where the number of sector stencils N_{Si} depends on the cell shape and its closeness to boundaries. By definition, the stencil S_0 is the central stencil. For each S_m, the polynomial from (6.1) leads to

$$p_m(\vec{\xi}) = \overline{\Phi}_i + \sum_{k=1}^{K} a_k^{(m)} \Omega_k(\vec{\xi}). \tag{6.6}$$

The WENO reconstruction is obtained by a non-linear combination of several polynomials p_m as

$$p_{weno}(\vec{\xi}) = \sum_{m=0}^{N_{Si}} \omega_m p_m(\vec{\xi}), \tag{6.7}$$

with the non-linear weights

$$\omega_m = \frac{\gamma_m}{\sum_{m=0}^{N_{Si}} \gamma_m}, \tag{6.8}$$

and γ_m defined as

$$\gamma_m = \frac{d_m}{\left(\varepsilon + \mathcal{I}_{S,m}\right)^p}. \tag{6.9}$$

ε prevents the denominator from becoming zero and is typically taken as 10^{-6}. The index p enables the weights of non-smooth regions the possibility of tending to zero faster than the mesh size converges to zero and is usually chosen as $p = 4$ [7]. The linear weights d_m are taken to assign a higher weight to the central stencil in comparison to the sectoral stencil due to its precision in reconstructing smooth functions. In accordance with Dumbser [3], they are defined as

$$d_m = \begin{cases} 10^3 & \text{if} \quad m = 0 \\ 1 & \text{else} \end{cases} \tag{6.10}$$

It should be noticed that increasing d_0 leads to a higher weight for the central stencil, thus to a better solution in case of smooth problems. By contrast, a lower weight works better near discontinuities. Finally, $\mathcal{I}_{S,m}$ represents the indicator for the smoothness of the solution on the stencil S_m. The smoother the solution the smaller this indicator gets and the larger is ω_m. The matrix expression for $\mathcal{I}_{S,m}$ is provided by Pringuey [22] and reads

$$\mathcal{I}_{S,m} = \sum_{s=1}^{K} a_s^{(m)} \cdot \left(\sum_{t=1}^{K} \mathcal{B}_{st} a_t^{(m)} \right), \tag{6.11}$$

where $\mathcal{B}_{st}$ is an element of the mesh-independent oscillation indicator matrix $\mathcal{B}$, which is discussed in chapter 6.3.

It is convenient to represent the WENO polynomial p_{weno} in the form of (6.6) by substituting (6.6) in (6.7)

$$p_{weno}(\vec{\xi}) = \sum_{m=0}^{N_{Si}} \omega_m \left(\overline{\Phi}_i + \sum_{k=1}^{K} a_k^{(m)} \Omega_k(\vec{\xi}) \right). \tag{6.12}$$

In consideration of the partition of unity through the weights and a rearrangement, equation (6.12) finally results in

$$p_{weno}(\vec{\xi}) = \overline{\Phi}_i + \sum_{k=1}^{K} \left(\sum_{m=0}^{N_{Si}} \omega_m a_k^{(m)} \right) \cdot \Omega_k(\vec{\xi}) = \overline{\Phi}_i + \sum_{k=1}^{K} \tilde{a}_k \cdot \Omega_k(\vec{\xi}), \tag{6.13}$$

with $\tilde{a}_k$ denominated as modified degrees of freedom.

6.1 Modified Stencil Collection Algorithm

The modified degrees of freedom are calculated from weighted solutions of several stencils. Commonly, these solutions are provided from one central and several sectoral stencils which cover all spatial directions of the target cell. In the case of cells near boundaries, some stencils could be too small and have to be discarded. A compact stencil for isotropic, uniform meshes is collected by adding the nearest face neighbours iteratively. However, unstructured meshes and regions with highly anisotropic cells can

not ensure this compactness which is an important requirement to preserve an accurate solution. Further, the selection of the stencils in physical space may lead to a loss of information near walls along the boundary layer region. Therefore, the stencil is transformed to a reference system $\vec{\xi} = (\xi, \eta, \zeta)$ where no scaling effects from increasing grid resolution or deformed cells occur. Furthermore, the geometric weighting of the quasi-ENO scheme is no longer used.

The collection procedure for each stencil is identical to chapter 3.2 with the only difference of operating in the reference space $\vec{\xi}$. Denoting any vertex of V_i as $P_0 = (x_0, y_0, z_0)$, the basis of a reference frame can be spanned by this point and three other vertices of the cell $P_1 = (x_1, y_1, z_1), ..., P_3 = (x_3, y_3, z_3)$ which are connected to P_0 through an edge (see figure 6.1). This leads to the mapping $\vec{x} = \vec{x}(\xi, \eta, \zeta)$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} + \mathcal{J} \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix}, \tag{6.14}$$

with $\mathcal{J}$ as the Jacobian matrix, defined as

$$\mathcal{J} = \begin{pmatrix} x_1 - x_0 & x_2 - x_0 & x_3 - x_0 \\ y_1 - y_0 & y_2 - y_0 & y_3 - y_0 \\ z_1 - z_0 & z_2 - z_0 & z_3 - z_0 \end{pmatrix}. \tag{6.15}$$

Any point in the stencil $P_{\vec{x}} = (x_P, y_P, z_P)$ can then be mapped to its equivalent in the new space $P_{\vec{\xi}} = (\xi_P, \eta_P, \zeta_P)$ by

$$\begin{pmatrix} \xi_P \\ \eta_P \\ \zeta_P \end{pmatrix} = \mathcal{J}^{-1} \begin{pmatrix} x_P - x_0 \\ y_P - y_0 \\ z_P - z_0 \end{pmatrix}. \tag{6.16}$$

The transformed cells volume $|V_i'|$ results from

$$|V_i'| = |\det(\mathcal{J}^{-1})||V_i|. \tag{6.17}$$

The inverse Jacobians are precomputed for each target cell analytically which reduces the costs of mapping. Further, the affine transformation preserves the principal connections between the cells for which reason the collection of face neighbours is independent of space and can still be applied. The stencil lists are then sorted by distance in $\vec{\xi}$, and the J_{max} cells are collected resulting in a compact stencil in $\vec{\xi}$.

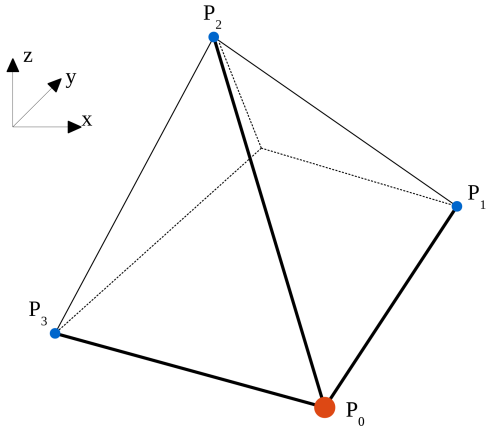


Figure 6.1: Selection of the vertices for the definition of the reference system $\vec{\xi}$

6.1.1 Generation of Sectoral Stencils

The sector stencils are constructed under consideration of several conditions. Besides the compactness, the union of all sectors has to cover the whole space around the target cell like one big central stencil at which they are not allowed to share another cell than the target cell [22]. The compactness is automatically fulfilled by the above procedure, while the remaining conditions have to be taken into account during the selection of sectors. A simple but nonetheless brilliant solution is provided by Pringuey [22], which is taken as a reference here. The starting point is a large central stencil list where the cells are already sorted by distance. The size of the list N_U relates to the number of internal faces of the target cell N_l by

$$N_U = (N_l + 1) \cdot J_{max}, \tag{6.18}$$

which offers a small surplus in case of convoluted boundaries. After the selection of sectoral stencils, the central stencil is simply obtained from cutting the specified list to the necessary size J_{max}. In general, the number of sectors equals the number of internal faces of V_i'. However, if some sector does not provide enough cells it is not taken into account for the runtime operations. Each sector is defined by a cone from the cell center of the target cell $P_0 = (x_0, y_0, z_0)$ as the apex and the contour of the related face as the base. The N_U cells are assigned to the sectors according to the position of their cell centres which results in a distribution as can be seen in figure 6.2.

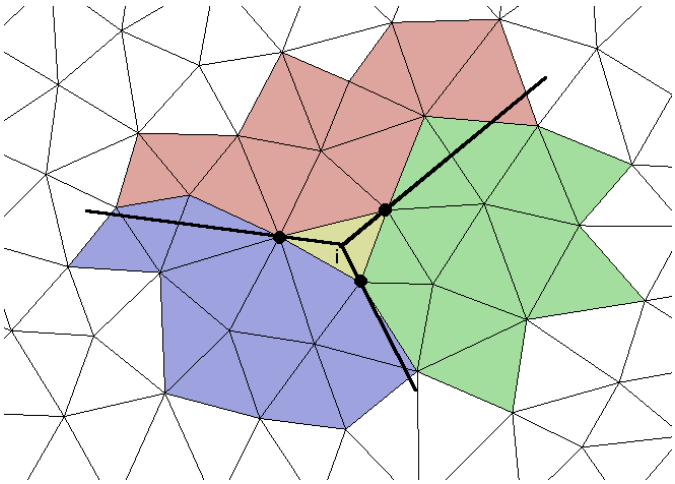


Figure 6.2: Definition of the three sectoral stencils for a triangular cell - two-dimensional example

In order to check the assignment of the cells, each sector is mapped to the first octant in a transformation space $\vec{X} = (X, Y, Z)$, at which the relevant cell centres have positive coordinates in $\vec{X}$. Under consideration of the cell center $P_0 = (x_0, y_0, z_0)$, such a mapping is obtained by writing

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} + \mathcal{J}_C \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}, \tag{6.19}$$

with $\mathcal{J}_C$ the Jacobian matrix built from the base of the cone P_1, P_2, P_3 (see figure 6.3).

Equation (6.19) is just applicable to triangles for which reason each internal face is divided into a minimum number of triangles N_T first. Thus, the sector of a face S_l is a union of several sub-sectors S_{lj}

$$S_l = \bigcup_{j=1}^{N_T} S_{lj} \tag{6.20}$$

The cell center of each of the N_U cells is transformed to the reference space $\vec{X}$ of each S_{lj} and is added to the sectoral stencil S_l if:

- All coordinates in the reference space are positive.
- The cell is not already a member of another stencil which may happen on Cartesian grids where the center lies on the boundary of two adjunct sectors.
- The target stencil list contains no more than J_{max} cells.

The first and last conditions are simple requests while the second condition is prevented by using a dynamic list from which cells are removed after being assigned to a sector. The received stencils are compact by itself since the central lists are presorted and scanned from the nearest to the farthest members.

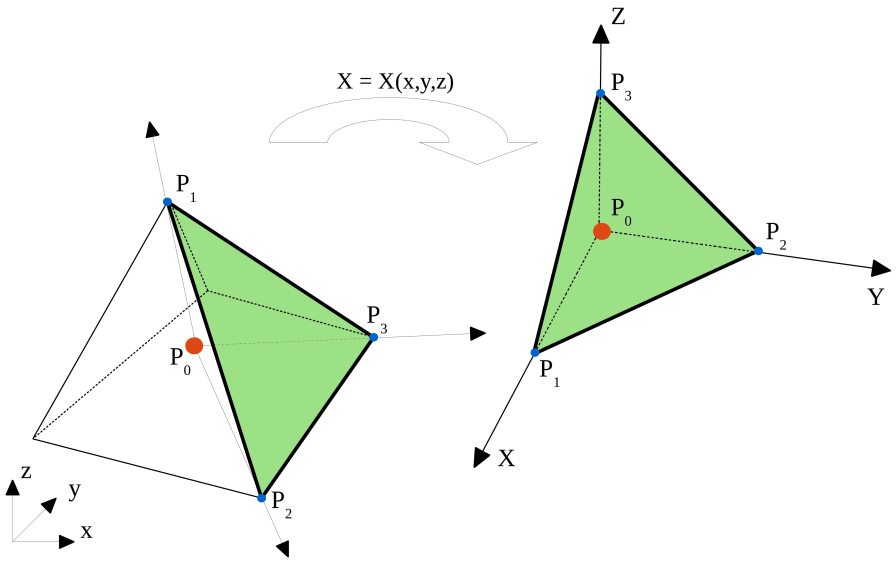


Figure 6.3: Mapping of a face to the first octant

6.2 Obtaining the Pseudoinverse

In contrast to the quasi-ENO scheme, it is no longer necessary to solve the whole system of equations in runtime. Instead, the Moore-Penrose pseudoinverse $\mathcal{A}^+$ can be precomputed and stored. The common way of calculating $\mathcal{A}^+$ is using a singular value decomposition (SVD), which will be shortly described here. For more details, see e.g. Lawson [10].

In SVD, the $m \times n$ matrix $\mathcal{A}$ is represented by its decomposition

$$\mathcal{A} = \mathcal{U}\mathcal{S}\mathcal{V}^T, \tag{6.21}$$

with $\mathcal{U}$ as a $m \times m$ orthogonal matrix of the orthonormalised eigenvectors of $\mathcal{A}\mathcal{A}^T$ and $\mathcal{V}$ as a $n \times n$ orthogonal matrix of the orthonormalised eigenvectors of $\mathcal{A}^T\mathcal{A}$. Lastly, $\mathcal{S}$ represents a $m \times n$ diagonal matrix, containing the non-zero singular values of $\mathcal{A}$. Considering the least-squares problem

$$min\|\mathcal{A} \cdot a - b\|_2, \quad \mathcal{A} \in \mathbb{R}^{m \times n}, \quad b \in \mathbb{R}^m, \quad m \geq n, \tag{6.22}$$

and replacing $\mathcal{A}$ by its decomposition from (6.21) leads to

$$\begin{aligned} (\mathcal{U}\mathcal{S}\mathcal{V}^T) \cdot a &= b \\ a &= (\mathcal{V}\mathcal{S}^+\mathcal{U}^T) \cdot b = \mathcal{A}^+ b, \end{aligned} \tag{6.23}$$

where $\mathcal{A}^+$ is the pseudoinverse. Similar to the rank-deficient problem in chapter 3.4, there may occur small singular values, which lead to high sensitivity of a to inaccurate data or round-off errors. Hence, a tolerance parameter τ is introduced for neglecting singular values smaller than τ. In either case, equation (6.23) provides the solution of minimum length.

6.2.1 Boundary Conditions

A detailed description of the boundary treatment is given in chapter 3.6. Again, these conditions are just considered for the target cell but added to the reconstruction of all

stencils. Dirichlet boundary conditions are calculated as written in (3.36), just in the reference space. Neumann boundary conditions have to be modified due to the given normal vectors in physical space. In accordance with Tsoutsanis [29], it yields (compare (3.38))

$$\begin{aligned} f_2(\vec{\xi}_g) &= \frac{\partial p_i(\vec{\xi}_g)}{\partial n} = \nabla p_i(\vec{\xi}_g)\vec{n}(\vec{\xi}) \\ &= \sum_{k=1}^{K} a_k \cdot \left(n_\xi \frac{\partial \Omega_k(\vec{\xi}_g)}{\partial \xi} + n_\eta \frac{\partial \Omega_k(\vec{\xi}_g)}{\partial \eta} + n_\zeta \frac{\partial \Omega_k(\vec{\xi}_g)}{\partial \zeta} \right), \end{aligned} \tag{6.24}$$

where $\vec{n}(\vec{\xi})$ is given by

$$\begin{aligned} n_\xi &= n_x \mathcal{J}_{1,1}^{-1} + n_y \mathcal{J}_{1,2}^{-1} + n_z \mathcal{J}_{1,3}^{-1} \\ n_\eta &= n_x \mathcal{J}_{2,1}^{-1} + n_y \mathcal{J}_{2,2}^{-1} + n_z \mathcal{J}_{2,3}^{-1} \\ n_\zeta &= n_x \mathcal{J}_{3,1}^{-1} + n_y \mathcal{J}_{3,2}^{-1} + n_z \mathcal{J}_{3,3}^{-1}. \end{aligned} \tag{6.25}$$

The conditions are introduced to the system of equations by Gaussian elimination steps before the SVD.

6.3 WENO Smoothness Indicator

Once the degrees of freedom are calculated for each stencil, the final coefficients are computed by a non-linear weighted combination. The weights have to determine to what extend the solution in a stencil provides a qualitative contribution for a smooth solution of the polynomial reconstruction in the target cell, which is indicated by the smoothness indicator $\mathcal{I}_S$. According to Jiang [9], the aim of the indicator is the minimization of the total variation for the sum of the $\mathcal{L}_2$-norms of all derivatives of the polynomial. Recalling the definition of $\mathcal{I}_S$ in (6.11), this property has to be satisfied by the oscillation indicator matrix $\mathcal{B}$. Therefore, denoting $\gamma = \lambda - \alpha - \beta$ and r as the polynomial order,

each element $\mathcal{B}_{st}$ is defined as

$$\mathcal{B}_{st} = \sum_{\lambda=1}^{r} \sum_{\alpha=0}^{\lambda} \sum_{\beta=0}^{\lambda-\alpha} \int_{V_i'} \frac{\partial^\lambda}{\partial\xi^\alpha \partial\eta^\beta \partial\zeta^\gamma} \Omega_s(\vec{\xi}) \cdot \frac{\partial^\lambda}{\partial\xi^\alpha \partial\eta^\beta \partial\zeta^\gamma} \Omega_t(\vec{\xi}) \, \mathrm{d}\xi \mathrm{d}\eta \mathrm{d}\zeta. \tag{6.26}$$

As it can be seen from (6.26), the matrix is solution independent and can be precomputed. Additionally, it is mesh independent through the calculation in the reference space. Under consideration of the definitions for the basis functions (3.6), equation (6.26) is simplified to

$$\mathcal{B}_{st} = \sum_{\lambda=1}^{r} \sum_{\alpha=0}^{\lambda} \sum_{\beta=0}^{\lambda-\alpha} \int_{V_i'} \frac{\partial^\lambda}{\partial\xi^\alpha \partial\eta^\beta \partial\zeta^\gamma} \Psi_s(\vec{\xi}) \cdot \frac{\partial^\lambda}{\partial\xi^\alpha \partial\eta^\beta \partial\zeta^\gamma} \Psi_t(\vec{\xi}) \, \mathrm{d}\xi \mathrm{d}\eta \mathrm{d}\zeta. \tag{6.27}$$

Further, the monomials can be expressed by their orthogonal basis functions as

$$\begin{aligned} \Psi_s(\vec{\xi}) &= (\xi - \xi_i)^{A_1} (\eta - \eta_i)^{B_1} (\zeta - \zeta_i)^{C_1} \\ \Psi_t(\vec{\xi}) &= (\xi - \xi_i)^{A_2} (\eta - \eta_i)^{B_2} (\zeta - \zeta_i)^{C_2}, \end{aligned} \tag{6.28}$$

with the condition $(A_1, B_1, C_1, A_2, B_2, C_2) \in [0, r]^6$ and $\{1 \leq A_i + B_i + C_i \leq r, i = 1, 2\}$. Applying the partial derivatives to (6.28), equation (6.27) yields

$$\mathcal{B}_{st} = \sum_{\lambda=1}^{r} \sum_{\alpha=0}^{\lambda} \sum_{\beta=0}^{\lambda-\alpha} \int_{V_i'} K \cdot \xi^{(A_1+A_2-\alpha)} \eta^{(B_1+B_2-\beta)} \zeta^{(C_1+C_2-\gamma)} \, \mathrm{d}\xi \mathrm{d}\eta \mathrm{d}\zeta. \tag{6.29}$$

As it is shown in Pringuey's thesis [22], the constant K is defined as

$$\begin{aligned} K &= \begin{cases} K_1 & if \quad \forall_{i=1,2} : (A_i \geq \alpha) \wedge (B_i \geq \beta) \wedge (C_i \geq \gamma) \\ 0 & else \end{cases} \\ K_1 &= \frac{A_1!}{(A_1-\alpha)!} \cdot \frac{A_2!}{(A_2-\alpha)!} \cdot \frac{B_1!}{(B_1-\beta)!} \cdot \frac{B_2!}{(B_2-\beta)!} \cdot \frac{C_1!}{(C_1-\gamma)!} \cdot \frac{C_2!}{(C_2-\gamma)!}. \end{aligned} \tag{6.30}$$

The evaluation of the volume integrals is carried out by transforming them into surface integrals and the integration over the faces of each cell as described in chapter 3.3. The highest order to consider is $2r-2$ since $\lambda_{min} = 1$. In practice, all volume integrals are calculated before the triple sum is evaluated using (6.29) and (6.30).

6.4 Practical Operations

All modifications influence the preprocessing and runtime operations in comparison to the Quasi-ENO scheme. The new preprocessing steps are (compare chapter 3.8):

1. Generation of a large central stencil list for each control volume V_i' in the transformed space. Halo cells are considered if several processors are included.

2. Generation of the sectoral stencils and the final central stencil using the algorithm from chapter 6.1.1.

3. Calculation and storing of all volume integrals of the basis functions for the target cells V_i' using triangularization of the faces and Gaussian quadrature formulas.

4. Computation and storing a pseudoinverse $\mathcal{A}^+$ for each stencil of V_i' using SVD. For this purpose, volume integrals of the basis functions for the cells in the stencils are calculated in the space of V_i'. In comparison to ENO, these integrals have to be calculated multiple times. In the case of multiple processors, transformation information and results from volume integrations have to be transmitted frequently.

5. Determination and storing of the oscillation indicator matrix $\mathcal{B}$ for each transformed target cell.

The most relevant lists of preprocessing are additionally written as files and stored in the `constant` folder, which reduces the time for preprocessing if calculations on the same mesh are executed several times. Here, WENO schemes need much more space than ENO schemes, due to several stencils, reconstruction matrices, and one indicator matrix per cell. In contrast, the runtime steps are reduced to three simple steps:

1. For each stencil of V_i', generation of the vector b as the right hand side of (3.10) from $\overline{\Phi}$. The degrees of freedom are directly computed from a matrix vector product using (6.23).

2. Inserting the coefficients in (6.11) for obtaining the smoothness indicator.

3. Calculation of the WENO weights from (6.8) and computing the modified degrees of freedom from (6.13)

Chapter 7

Application of the WENO Reconstruction to the Level Set Equation

In principal, the solution of the level set equation is obtained as described in chapter 5. However, the modified degrees of freedom of the WENO scheme are represented in a reference space of each cell which has to be considered during the solution procedure. The surface integrals of the basis functions in (5.15) differ for the two cells sharing each face since the representation of the face in the two reference systems may be different. A detailed derivation of the modifications within the resulting formulas is given by Pringuey [22]. Therefore, the focus of this chapter lies on the presentation of the results of the test cases from chapter 5.3. All meshes and adjustments are preserved in order to compare the results to the Quasi-ENO scheme.

First of all, the performance of the new implementation is shown in table 7.1. In comparison to the Quasi-ENO scheme (table 4.4), a runtime step using second- and third-order polynomials takes just one-third of the time while the preprocessing step takes more than 10 times as long. The reasons are the additional collection of multi-

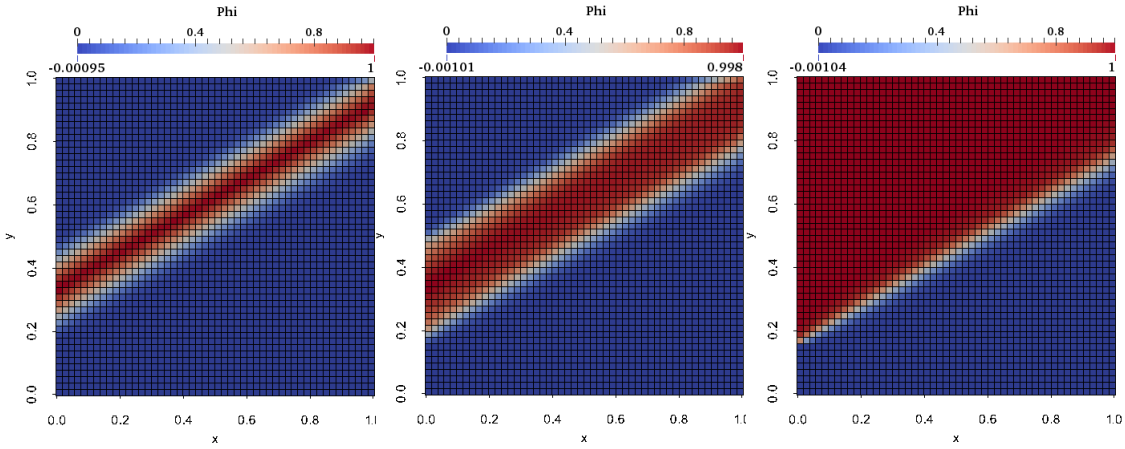
ple sectoral stencils, the pre-computation of $\mathcal{A}^+$, and the calculation of the oscillation indicator matrix $\mathcal{B}$. However, the overall efficiency is improved since the additional preprocessing time is outbalanced by the improved runtime after several time steps.

Polynomial order r	preprocessing	runtime
1	52.91	0.20
2	331.05	0.45
3	1535.55	1.68

Table 7.1: Time measurements for reconstruction of a three-dimensional, smooth function for $124,693$ tetrahedra using WENO (measured in seconds on Intel i5-4430 CPU, 8 GB memory, Linux Ubuntu 64-bit)

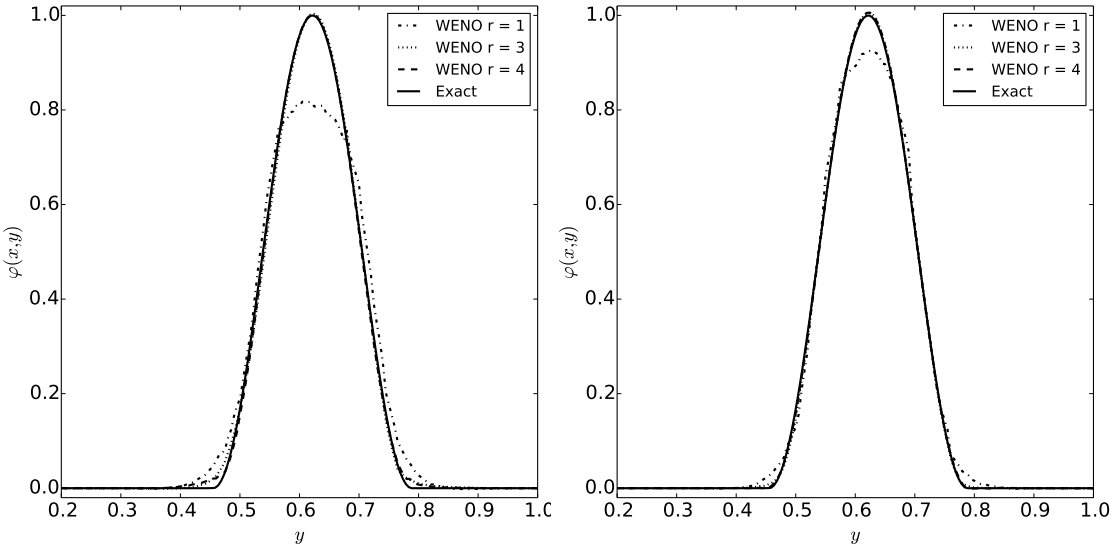
Two-dimensional Results

The steady-state solutions for the computations with a third-order reconstruction are presented for all three test cases in figure 7.1. In comparison to the Quasi-ENO scheme, the solutions are rather bounded, but limiting is still necessary for physical applications. The calculations of the *sin*-profile show improved results on the unstructured grid, particularly for the third- and fourth-order reconstruction. It is noticeable that the fourth-order solution equals mostly the third-order one which is obviously caused by the third-order time integration. Taking the results for the *ellipse*- and *step*-profile into account, it can be seen that all calculations provide an appropriate solution independent of the topology of the mesh. Quite the contrary to the Quasi-ENO reconstruction, the results on the unstructured grids are even better than on the structured grids which may result from more compact stencils with less collinear information [29]. Further, the results on the structured grids are similar to the ones using the Quasi-ENO scheme but the fourth-order reconstruction yields stable solutions here. Finally, it is remarkable that φ is generally smooth even on the unstructured grid.



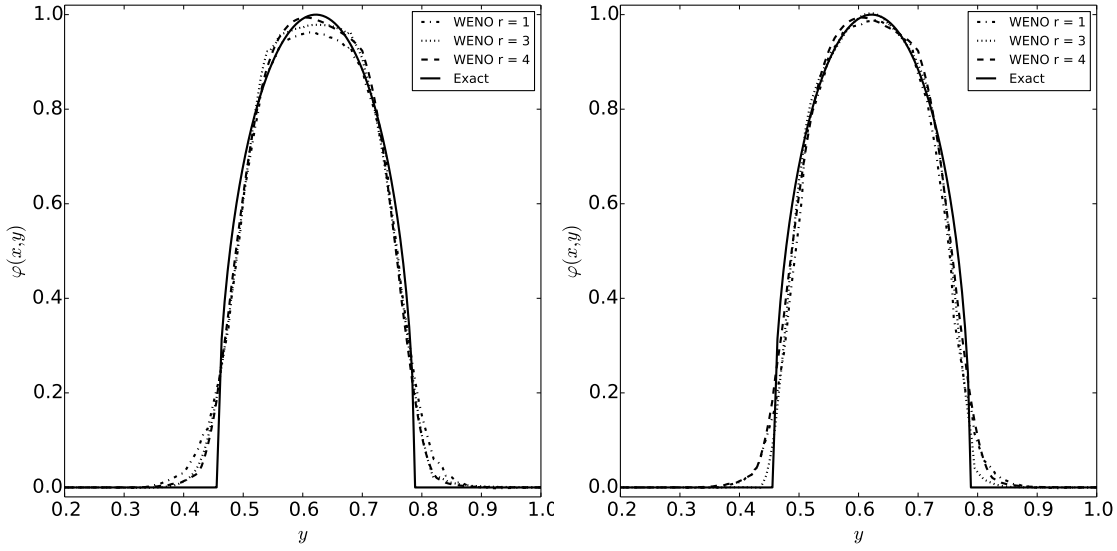
(a) *sin*-profile (b) *ellipse*-profile (c) *step*-profile

Figure 7.1: Steady-state solutions on the structured grid using 3^{rd}-order polynomials



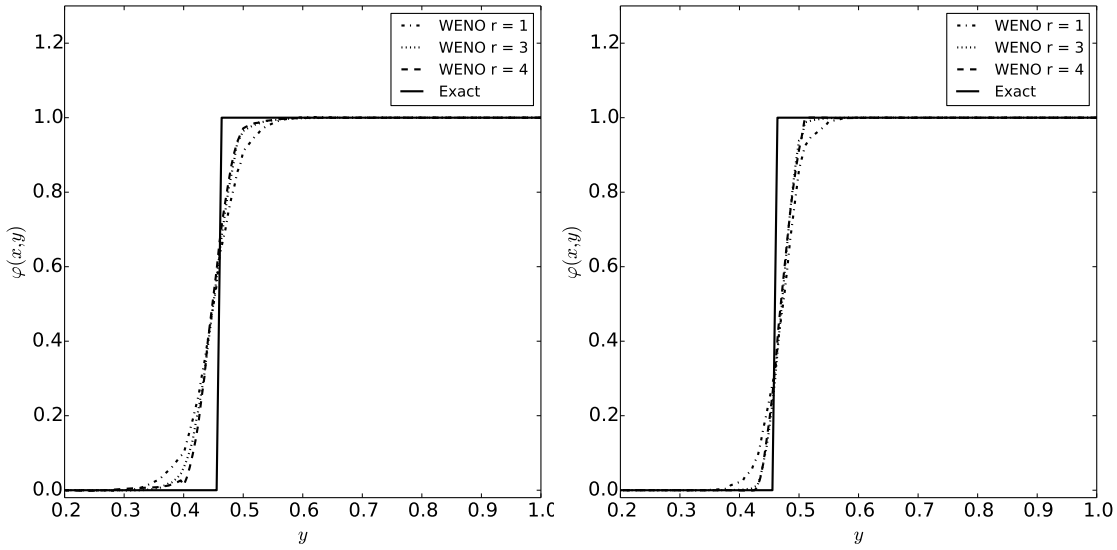
(a) $2,500$ hexahedra (b) $2,832$ triangular prisms

Figure 7.2: Slice of the *sinus*-profile at $x = 0.5$



(a) 2,500 hexahedra (b) 2,832 triangular prisms

Figure 7.3: Slice of the *ellipse*-profile at $x = 0.5$

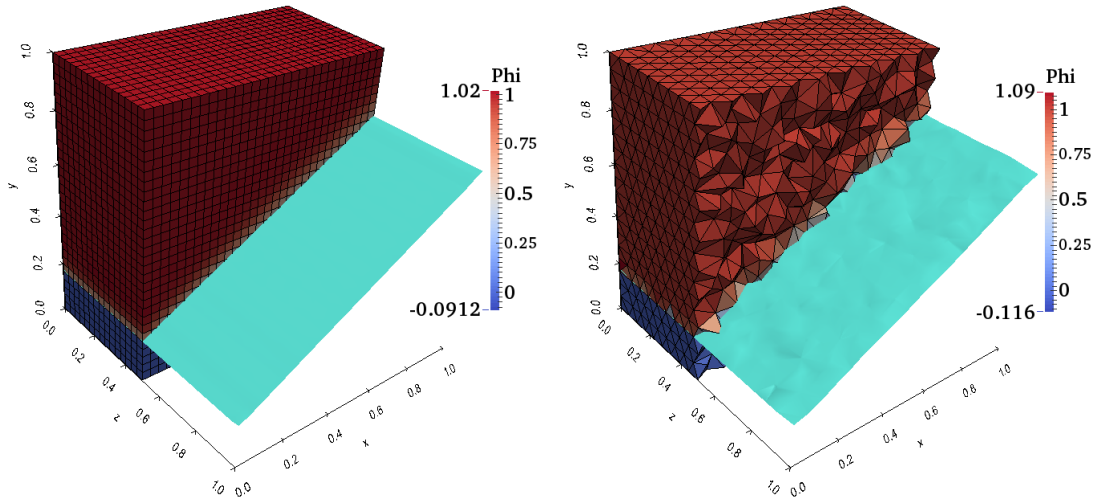


(a) 2,500 hexahedra (b) 2,832 triangular prisms

Figure 7.4: Slice of the two-dimensional *step*-profile at $x = 0.5$

Step-profile in 3D

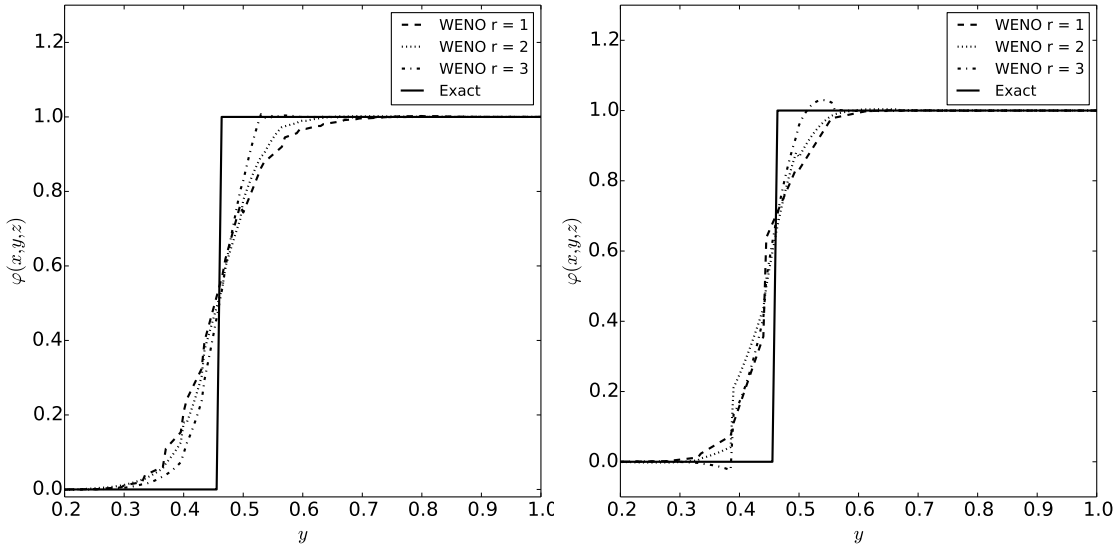
The results for the three-dimensional *step*-profile are presented in figure 7.5 and 7.6. Again, the solution provides less over- and undershoots in comparison to the Quasi-ENO scheme. The use of a WENO reconstruction leads to stable solutions for all tested polynomial orders on both grids. As can be seen in figure 7.5, the resulting distribution is non-oscillatory and the accuracy increases with the nominal order of accuracy. On the unstructured mesh, the third-order reconstruction leads to over- and undershoots near the discontinuity. The increase of the grid resolution should vanish this behaviour since the compactness of the stencils is increased at the same time. Finally, the fluxes provide a smoother distribution of φ for all considered computations.



(a) 27,000 hexahedra

(b) 27,092 tetrahedra

Figure 7.5: Resulting surface for different grids using 2^{nd}-order polynomials



(a) 27,000 hexahedra (b) 27,092 tetrahedra

Figure 7.6: Slice of the three-dimensional *step*-profile at $x = z = 0.5$

Chapter 8

Final Remarks

In this work, high-order non-oscillatory solutions of the level set equation on arbitrary meshes are obtained successfully. Furthermore, the implementation of two (W)ENO schemes yields the possibility to compare different approaches within this class of schemes. The Quasi-ENO scheme achieves the nominal orders of accuracy for the reconstruction of smooth two- and three-dimensional functions, as well as a non-oscillatory solution for Abgrall's function. For such a function, with many discontinuities, a WENO scheme would not be able to reproduce these results on coarse grids since it needs large stencils in all directions, at which at least one of them contains smooth data [3, 16]. Here, the Quasi-ENO scheme can at least produce an accurate solution by using the available smooth data. Apart from that, this approach reveals lower accuracy for convection problems in comparison to the WENO method. In 2D, the scheme computes accurate solutions on structured grids up to third-order polynomials but may lose the nominal order near discontinuities due to insufficient smooth data in the stencils. Here, the WENO scheme benefits from the sectoral stencils where at least one of them contains enough smooth data for an accurate reconstruction. On unstructured grids, the transformations into a reference space without scaling effects improves the results additionally. The Quasi-ENO scheme shows limitations in stability for higher orders for

the three-dimensional test case. The main reasons are bad conditioned matrices resulting from nearly linear-dependent lines and the ENO-weighting. As described in the solution algorithm, preconditioning is not helpful in this case. Moreover, even stable solutions are quite unbounded, and the fluxes are non-smooth, particularly on unstructured grids. The extension to WENO solves these problems and provides stable solutions up to third-order polynomials in 3D. The fluxes are smoother, and the efficiency of the code is crucial improved in runtime.

Resulting from these investigations, the implemented WENO scheme will be used for the further work. For this purpose, the convergence rate of the method has to be verified first, together with a parameter study of the constants of the non-linear weights. A limiter has to be introduced since the solution for the scalar level set should be bounded. Pringuey [22] uses OpenFOAM's multi-dimensional limiter for explicit solutions (MULES) which nevertheless reduces the nominal order and is time costly. Therefore, it is planned to use a simpler limiter as provided by Zhang [33], which bounds the solution within a stencil from the maximum and minimum values of the gathered data. The efficiency and accuracy of the temporal integration can eventually be improved by using the ADER approach of Toro [25]. Finally, the mass conservative level set method of Olsson [19] will be implemented coupled with the incompressible Navier-Stokes-equations. The high-order reconstruction of the gradient has to be computed, in which once again the WENO approach will be used.

Bibliography

[1] R. Abgrall. “On essentially non-oscillatory schemes on unstructured meshes: Analysis and implementation”. In: *Journal of Computational Physics* Volume 114 (1994), pp. 45–58.

[2] S. Deng. *Quadrature formulas in two fimensions, Lecture Math 5172 - Finite element method*. URL: http://math2.uncc.edu/~shaodeng/TEACHING/math5172/Lectures/Lect_15.PDF (Accessed: 2016).

[3] M. Dumbser and M. Käser. “Arbitrary high order non-oscillatory finite volume schemes on unstructured meshes for linear hyperbolic systems”. In: *Journal of Computational Physics* Volume 221 (2007), pp. 693–723.

[4] O. Friedrich. “Weighted essentially non-oscillatory schemes for the interpolation of mean values on unstructured grids”. In: *Journal of Computational Physics* Volume 114 (1998), pp. 194–212.

[5] G.H. Golub and C.F. van Loan. *Matrix computations*. John Hopkins University Press, 1996.

[6] A. Harten. “Uniformly high order accurate essentially non-oscillatory schemes, III”. In: *Journal of Computational Physics* Volume 71 (1987), pp. 231–303.

[7] A.K. Henrick, T.D. Aslam, and J.M. Powers. “Mapped weighted essentially non-oscillatory schemes: Achieving optimal order near critical points”. In: *Journal of Computational Physics* Volume 207 (2005), pp. 542–567.

[8] H. Jasak. *Error analysis and estimation for the finite volume method with applications to fluid flows*. PH.D.-thesis, Department of Mechanical Engineering, Imperial College, London. (last access: 12.03.2016). 1996.

[9] G.-S. Jiang and C.-W. Shu. "Efficient implementation of weighted ENO schemes". In: *Journal of Computational Physics* Volume 126 (1996), pp. 202–228.

[10] C.L. Lawson and R.J. Hanson. *Solving least squares problems*. SIAM, 1995.

[11] R.J. Leveque. *Finite-volume methods for hyperbolic problems*. Cambridge University Press, 2004.

[12] X. Liu, S. Osher, and T. Chan. "Weighted essentially non-oscillatory schemes". In: *Journal of Computational Physics* Volume 115,1 (1994), pp. 200–212.

[13] S. McDonald. *Development of a high-order finite-volume method for unstructured meshes*. Master's thesis, University of Toronto. (last access: 12.03.2016). 2011.

[14] C. Michalak and C.F. Ollivier-Gooch. "Accuracy preserving limiter for the high-order accurate solution of the Euler equations". In: *Journal of Computational Physics* Volume 228 (2009), pp. 8693–8711.

[15] B. Mirtich. "Fast and accurate computation of polyhedral mass properties". In: *Journal of graphics tools* Volume 1,2 (1996), pp. 31–50.

[16] C.F. Ollivier-Gooch. *Higher-order least-squares reconstruction for turbulent aerodynamic flows*. 11th World Congress on Computational Mechanics. (last access: 12.03.2016). 2003.

[17] C.F. Ollivier-Gooch. "Quasi-ENO schemes for unstructured meshes based on umlimited data-dependent least-squares reconstruction". In: *Journal of Computational Physics* Volume 133 (1997), pp. 6–17.

[18] C.F. Ollivier-Gooch and M. van Altena. “A high-order-accurate unstructured mesh finite-volume scheme for the advection-diffusion equation”. In: *Journal of Computational Physics* Volume 118 (1994), pp. 729–752.

[19] E. Olsson and G. Kreiss. “A conservative level set method for two phase flow”. In: *Journal of Computational Physics* Volume 210 (2005), pp. 225–246.

[20] OpenFOAM. *OpenFOAM Programmer’s C++ documentation*. URL: `http://foam.sourceforge.net/docs/cpp/a00813.html#details` (Accessed: 2016).

[21] T. Pringuey. “High-order schemes on three-dimensional general polyhedra meshes - Application to the level set method”. In: *Computer Physics Communications* Volume 12,1 (2012), pp. 1–41.

[22] T. Pringuey. *Large eddy simulation of primary liqiud-sheet breakup*. PH.D.-thesis, Hopkinson Laboratory, University of Cambridge. (last access: 12.03.2016). 2012.

[23] C.-W. Shu. *Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws*. Institute for Computer Applications in Science and Engineering, NASA/CR-97-206253, ICASE Report No. 97-65. (last access: 12.03.2016). 1997.

[24] F. Stern et al. *Verification and validation of CFD simulations*. IHHR Report, No. 407. (last access: 12.03.2016). 1999.

[25] V.A. Titarev and E.F. Toro. “ADER schemes for three-dimensional non-linear hyperbolic systems”. In: *Journal of Computational Physics* Volume 204 (2005), pp. 715–736.

[26] V.A. Titarev and E.F. Toro. “Finite-volume WENO schemes for three-dimensional conservation laws”. In: *Journal of Computational Physics* Volume 201 (2004), pp. 238–260.

[27] E.F. Toro. “MUSTA: A multi-stage numerical flux”. In: *Applied Numerical Mathematics* Volume 56 (10-11) (2006), pp. 1464–1479.

[28] E.F. Toro. *Riemann solvers and numerical methods for fluid dynamics*. Springer Verlag, 1997.

[29] P. Tsoutsanis, A.F. Antoniadis, and D. Drikakis. “WENO schemes on arbitrary unstructured meshes for laminar, transitional and turbulent flows”. In: *Journal of Computational Physics* Volume 256 (2014), pp. 254–276.

[30] Z.J. Wang. *Adaptive high-order methods in computational fluid dynamics, Vol.2*. World Scientific Publishing, 2011.

[31] X. Wu, J. Liang, and Y. Zhao. “A new smoothness indicator for third-order WENO scheme”. In: *Journal of Numerical Methods in fluids* (2015).

[32] X. Zhang and C.-W. Shu. “Linear instability of the fifth-order WENO method”. In: *SIAM Journal on Numerical Analysis* Volume 45(5) (2007), pp. 1871–1901.

[33] X. Zhang and C.-W. Shu. “On maximum-principle-satisfying high order schemes for scalar conservation laws”. In: *Journal of Computational Physics* Volume 229 (2010), pp. 3091–3120.

Ich versichere eidesstattlich durch eigenhändige Unterschrift, dass ich die Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen sind, habe ich als solche kenntlich gemacht. Ich weiß, dass bei Abgabe einer falschen Versicherung die Prüfung als nicht bestanden zu gelten hat.

Rostock, 14.03.2016 _____