710 | März 2019

SCHRIFTENREIHE SCHIFFBAU

Manuel Manzke

Development of a Scalable Method for the Efficient Simulation of Flows using Dynamic Goal-Oriented Local Grid-Adaptation

TUHH
Technische Universität Hamburg

Development of a Scalable Method for the Efficient Simulation of Flows using Dynamic Goal-Oriented Local Grid-Adaptation

Vom Promotionsausschuss der
Technischen Universität Hamburg
zur Erlangung des akademischen Grades
Doktor-Ingenieur (Dr.-Ing.)
genehmigte Dissertation

von
Manuel Manzke
aus Hannover

2019

Vorsitzender des Prüfungsausschusses
Prof. Dr.-Ing. habil. Alexander Düster

Gutachter
1. Gutachter: Prof. Dr.-Ing. Thomas Rung
2. Gutachter: Prof. Michel Visonneau

Tag der mündlichen Prüfung
24. Oktober 2018

Schriftenreihe Schiffbau
Am Schwarzenberg - Campus 4
D-21073 Hamburg
http://www.tuhh.de/vss

Bericht Nr. 710
ISBN 978-3-89220-710-8
DOI 10.15480/882.2057

Acknowledgements

I am very grateful that I had the opportunity to work at the Institute for Fluid Dynamics and Ship Theory within a young, inspiring and talented team under the supervision of Prof. D.-Ing. Thomas Rung. His inimitable devotion towards the field of viscous fluid dynamics and his strong commitment towards every single researcher within his group always motivated myself. Our numerous in-depth discussions starting from theoretical backgrounds up to detailed implementation details were invaluable for my work. I also would like to thank Prof. Michel Visonneau for being my second assessor and Prof. Dr.-Ing. Alexander Düster for chairing the examination board.

Special thanks are directed to my colleagues from the Institute for Fluid Dynamics and Ship Theory as well as the developer group of FreSCo$^+$. In particular I would like to thank Jörg Brunswig, Jörn Kröger, Arthur Stück and Dieke Hafermann for their continuous support.

Finally and most important I thank my family and especially Mihaela and Isabel for their patiently and continuous support.

Abstract

Flows in marine applications are characterized by very high Reynolds numbers and complex three-dimensional geometries, which in combination often lead to localized turbulent structures. Furthermore, most flows include free surfaces. For both features, localized turbulent structures and free surfaces, a sufficiently fine local grid resolution is indispensable to achieve accurate solutions. On the other hand the effort for solving flow problems increases with increasing numbers of cells and the locations for the local grid refinement depends on the solution of the flow. Therefore, it is hardly possible to generate optimal grids, meaning using a minimum number of cells to solve a problem accurately, before knowing the solution for the flow problem. A well known technique to overcome this problem is dynamic local grid-adaptation, where the grid is adapted to the flow during the solution process. Although being a well known technique, the number of applications that are feasible to perform local grid-adaptation in turbulent flows for complex three-dimensional geometries is limited, which is caused by the high complexity of the software implementation and the lack of appropriate grid refinement indicators. Within this work the algorithm to perform local anisotropic grid-adaptation for hexahedral cells in a parallel unstructured grid environment is described in detail. The developed grid-adaptation technique is combined with different grid refinement indicators, ranging from simple feature based indicators via error estimators to sophisticated goal-oriented indicators for three-dimensional turbulent flows. Simple feature based indicators refine the grid at specific flow features without any link to an error or a stopping criterion for the refinement. Error estimator based refinement indicators try to estimate the error for each cell in the domain and to minimise the error by refining the grid at locations associated with large errors. However, they do not provide a link between the local errors and their influence on a scalar quantity of interest (e.g. drag). Goal-oriented error estimators link the local errors to the error in a scalar (global) quantity of interest and hence indicate refinement only in those cells where the local error has an influence on the global quantity of interest. Furthermore, it is possible to define a desired range of accuracy for the global quantity of interest to provide a stopping criterion for the refinement. In addition to the developed grid refinement technique, methods to enhance the accuracy of the VoF method, widely used for free-surface flows within finite volume flow solvers, are developed. The accuracy of the method is enhanced by the application of an Explicit Interface Sharpening (EIS) technique, which is able to resharpen blurred surfaces. The efficiency is enhanced by means of a sub-cycling technique, solving the transport equation for the mixture fraction with smaller time steps than the other transport equations. For the sub-cycling technique two dedicated modes associated to flow problems which lead to steady and transient solutions are developed.

Contents

Nomenclature

Abbreviations

AIAA	American Institute of Aeronautics and Astronautics
ALE	Arbitrary Lagrangian Eulerian (method)
BRICS	Blended Reconstructed Interface Capturing Scheme
CAD	Computer Aided Design
CAE	Computer Aided Engineering
CBC	Convective Boundedness Criteria
CDS	central differencing scheme
CFD	Computational Fluid Dynamics
CGNS	CFD General Notation System
CICSAM	Compressive Interface Capturing Scheme for Arbitrary Meshes
CLSMOF	Combined Level Set MoF (method)
CLSVOF	Combined Level-Set VoF (method)
CRS	Compact Row Storage
CV	control volume
DDS	downwind discretisation scheme
DNS	Direct Numerical Simulation

EIS	Explicit Interface Sharpening
FB	flux blending
FEM	Finite Element Methods
FV	Finite Volume
HPC	High Performance Computing
HRIC	High Resolution Interface Capturing (scheme)
IGDS	Inter-Gamma Differencing Scheme
ITTC	International Towing Tank Conference
LES	Large Eddy Simulation
LUDS	linear upwind differencing scheme
MAC	Marker and Cell (method)
MDAO	Multidisciplinary Design Analysis and Optimisation
MoF	Moment of Fluid (method)
MPI	Message Passing Interface
MUSCL	Monotone Upstream-centred Schemes for Conservation Laws
NVD	normalised variable diagram
PETSc	Portable, Extensible Toolkit for Scientific computation
PLIC	Piecewise Linear Interface Reconstruction
QUICK	quadratic upwind differencing scheme
RANS(E)	Reynolds Averagerd Navier-Stokes (Equations)
RMS	Root Mean Square
SIMPLE	Semi-Implicit Method for Pressure-Linked Equations

SLIC	Simple Line Interface Calculation
SPMD	Single Program Multiple Data programming model
STL	Surface Tesselation Language
TVD	Total Variation Diminishing
UDS	upwind differencing scheme
VoF	Volume of Fluid

Lower-case Greek

α	blending factor for flux blending scheme ($\alpha = 0 \rightarrow$ UDS, $\alpha = 1 \rightarrow$ CDS)
α	turbulence model constant
α_1	turbulence model constant
α_2	turbulence model constant
$\beta 1$	turbulence model constant
β	turbulence model constant
β^*	turbulence model constant
β_1	turbulence model constant
δ_F	face-based liquid-level sensor for EIS
δ_{ij}	Kronecker delta
ϵ	dissipation rate, local error estimate
$\epsilon_P(\phi)$	normalised error estimate for ϕ in cell P having the same dimensions as ϕ
γ	exponent controlling the decomposition of the diffusive flux operator
γ_{sharp_F}	face-based concentration transfer directionality factor for EIS

κ	turbulence model constant, TVD scheme parameter
(λ)	linear weighting factor for face interpolation $(0 < \lambda < 1)$
μ	dynamic viscosity
μ_t	turbulent eddy-viscosity
μ_{eff}	effective viscosity $(\mu + \mu_t)$
ν	kinematic viscosity
ν_t	eddy viscosity
ω	specific dissipation rate
ϕ	general variable of state
ψ_F	face-based final concentration transfer for EIS
ρ	density
σ_ϵ	turbulence model constant
σ_k	turbulence model constant
$\sigma_{\omega 1}$	turbulence model constant
$\sigma_{\omega 2}$	turbulence model constant
σ_{k1}	turbulence model constant
σ_{k2}	turbulence model constant
τ_w	shear stress at wall
τ_{ij}	viscous stress tensor
$\zeta(r)$	limiter for TVD scheme
ζ_i	unit vector in the direction of d_i $(\zeta_i = \frac{d_i}{\sqrt{d_j^2}})$

Upper-case Greek

Δ_F	largest physically admissible concentration transfer for EIS
Γ	diffusion coefficient
$\Psi(r)$	TVD scheme damping function

Sub- & Superscripts

$(\cdot)_B$	marker for boundary face
$(\cdot)_D$	downstream CV
$(\cdot)_F$	marker for face
$(\cdot)_{F'}$	auxiliary point for face interpolation (orthogonal projection)
$(\cdot)_{F*}$	auxiliary point for face interpolation (piercing point)
$(\cdot)_i$	vectorial quantity
$(\cdot)_{ij}$	tensorial quantity
$(\cdot)_L$	marker for cell on left-hand
$(\cdot)_{(N)}$	non-orthogonal part of vector
$(\cdot)_t$	marker for turbulent quantities
$(\cdot)_{NB}$	neighbouring CV
$(\cdot)_{(O)}$	orthogonal part of vector
$(\cdot)_P$	marker for internal CV
$(\cdot)_R$	marker for cell on right-hand
$(\cdot)'$	fluctuating component of a value
$(\cdot)_U$	upstream CV
$(\cdot)_{UU}$	far upstream CV
$(\cdot)_{wall}$	marker for quantities at wall boundaries

Lower-case Latin

a_1	turbulence model constant
$\overline{c}_\mu$	turbulent viscosity constant
c	mixture fraction
d_i	vector connecting CV centres P and NB ($d_i = x_{i,(NB)} - x_{i,(P)}$)
d_i^*	direction vector indicating the force optimisation direction
e	error
e_{c1}, e_{c2}, e_{c3}	unit vectors of the local cell coordinate system
f_i	body force vector
j_F	surface-declared objective functional integrand
j_V	volume-declared objective functional integrand
k	turbulent kinetic energy
$\dot{m}$	mass flux
n_i	unit face vector($n_i = \frac{\Delta F_i}{\sqrt{\Delta F_j^2}}$)
y	normal distance to nearest wall
$\hat{p}$	adjoint pressure
p	pressure
p'	pressure correction
r	TVD scheme sensor
t	time
$\hat{u}_i$	adjoint velocity vector
u_i	velocity vector

u^+	non-dimensional velocity
x_i	cartesian coordinate
y	closest wall distance
y^+	non-dimensional wall distance

Upper-case Latin

A_P	main diagonal equation system coefficient
A_{NB}	off diagonal equation system coefficient
$\tilde{P}$	limited production term
$C_{\epsilon 1}$	turbulence model constant
$C_{\epsilon 2}$	turbulence model constant
$\frac{D\cdot}{Dt}$	substantial derivative
$\mathcal{E}_P$	upper bound of local error e_P in cell P
E	total error $E = \sum_{k=1}^{N} \mathcal{E}_k$
E_P	error estimate in cell P
F_i	surface vector
F_o	Part of surface on which the objective function is evaluated
Fn	Froude number
$\mathcal{Z}$	arbitrary quantity
$\mathcal{I}_i$	refinement indicator vector
J	scalar quantity of interest = objective functional
$\mathcal{L}_h^H$	prolongation operator from coarse grid H to fine grid h, being of lower order than $\mathcal{M}$

L	differential operator
L_t	turbulent length scale
$\mathcal{M}_h^H$	prolongation operator from coarse grid H to fine grid h, being of higher order than $\mathcal{L}$
$N_{\Delta t}$	number of discrete time steps
P	turbulent production
Q	residual of continuity equation
$\mathcal{R}$	discrete residual operator
R_i	residuals of momentum equations
Re	Reynolds number
$\mathcal{S}$	rate of strain $\mathcal{S} = \sqrt{2S_{ij}S_{ij}}$, ideal sub-cycling speed-up
S	source term
S_{ij}	rate of strain tensor ($S_{ij} = 0.5\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)$)
$\mathcal{T}$	refinement indicator threshold
T	temperature
$\mathcal{V}$	material volume
V	volume
V_t	turbulent velocity scale

Symbols- & Specials

$\breve{(\cdot)}$	marker indicating linear part of term
$\breve{(\cdot)}$	marker indicating constant part of term
$\dot{(\cdot)}$	marker for flux

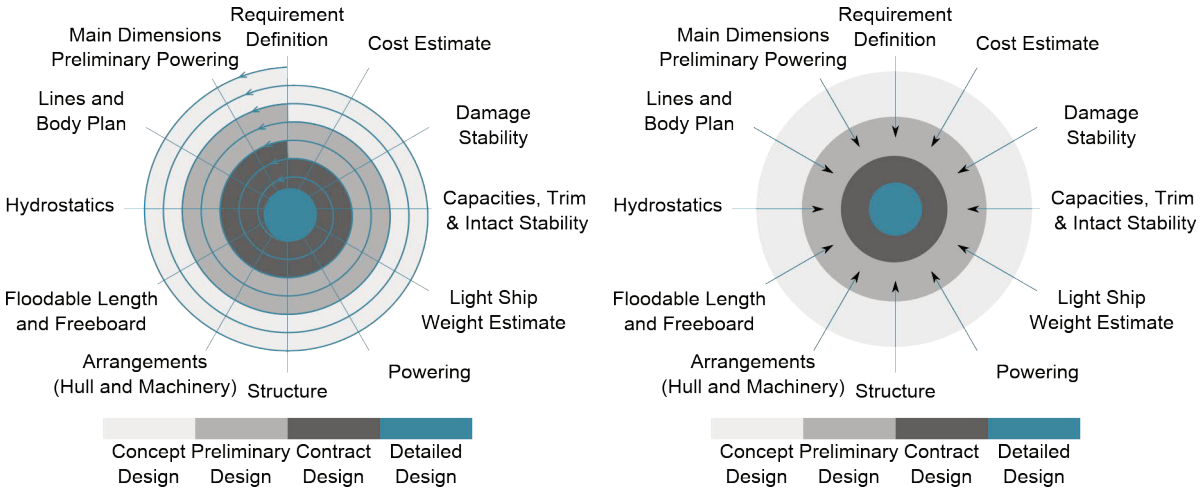
$(\cdot)^n$	upper index indicating time level
$(\cdot)^{n,t,s}$	upper indices indicating bases of local boundary coordinate system
$O(\mathsf{V})$	closed surface of volume V
$(\cdot)'$	turbulent fluctuations, marker indicating correction quantities
$\hat{(\cdot)}$	marker for adjoint variables
$\bar{(\cdot)}$	average value
$\tilde{(\cdot)}$	instantaneous value, normalised variable

1 Introduction

1.1 Background and Motivation

In the past decades the usage of Computational Fluid Dynamic (CFD) tools in industrial projects increased continuously. In 2012 Keith Hannah reported an annual growth rate for the CFD market of 13% [68]. The main drivers for this growth are the increasing computational power and continuous development of CFD methods. Due to the available computational power the usage of CFD is no longer limited to reduced flow models e.g inviscid flow problems. A large amount of the CFD tools used in the industry is able to solve viscous turbulent flow problems at feasible turn-around times. Therefore, CFD methods nowadays allow the investigation of several complicated flow phenomena with a reasonable commercial and temporal effort. The results of these investigations can be used to provide superior designs. Accordingly, CFD tools are widely used in the design phase. Weinhold [198] stated in 2013 that in recent years a paradigm shift to simulation-based design approaches has been observed. Currently the use of CFD tools within the design process in a multi-disciplinary environment is increasing, which leads to Multidisciplinary Design Analysis and Optimisation (MDAO) applications. Hence, *CFD results become the sole basis for making design decisions with serious business and financial impact* (Weinhold [198]). These developments have two main consequences. First the robustness and uncertainty of CFD methods is a main issue and secondly the CFD user profile changes from the CFD specialist to the design engineer.

The same trends as in the overall industry are also visible in the maritime industry. CFD tools are used within integrated design environments to enhance the design process [2]. This leads to a change in the design approach. The traditional design spiral gets more and more substituted by an integrated design approach, as depicted in Figure 1.1. The consequences are the same as in the overall industry – the user profile is changing from



(a) Design Spiral

(b) Integrated Design Approach

Figure 1.1: Design Approaches.

the CFD specialist to the design engineer and the robustness and uncertainty is a main issue for CFD methods. This is also reflected by the results of a questionnaire that has been distributed by the specialist committee on Computational Fluid Dynamics of the International Towing Tank Conference (ITTC) within the maritime industry. According to the results the main difficulties of CFD usage from the industrial point of view are the accuracy of CFD results, the grid generation and the confidence in CFD results (Figure 1.2).

The successful usage of CFD tools in the industry hinges on the integration of the CFD tools in the design process. The main requirements to integrate CFD tools in the design process are:

- High robustness
- Low uncertainty
- (Short) time to solution

To identify how these requirements can be satisfied, the typical process of a CFD simulation is reviewed, and the relation of these steps to the requirements is analysed. Each CFD simulation starts with the geometry preparation. Within the design process the geometry is usually defined within CAD systems. The requirements on the geometry representation

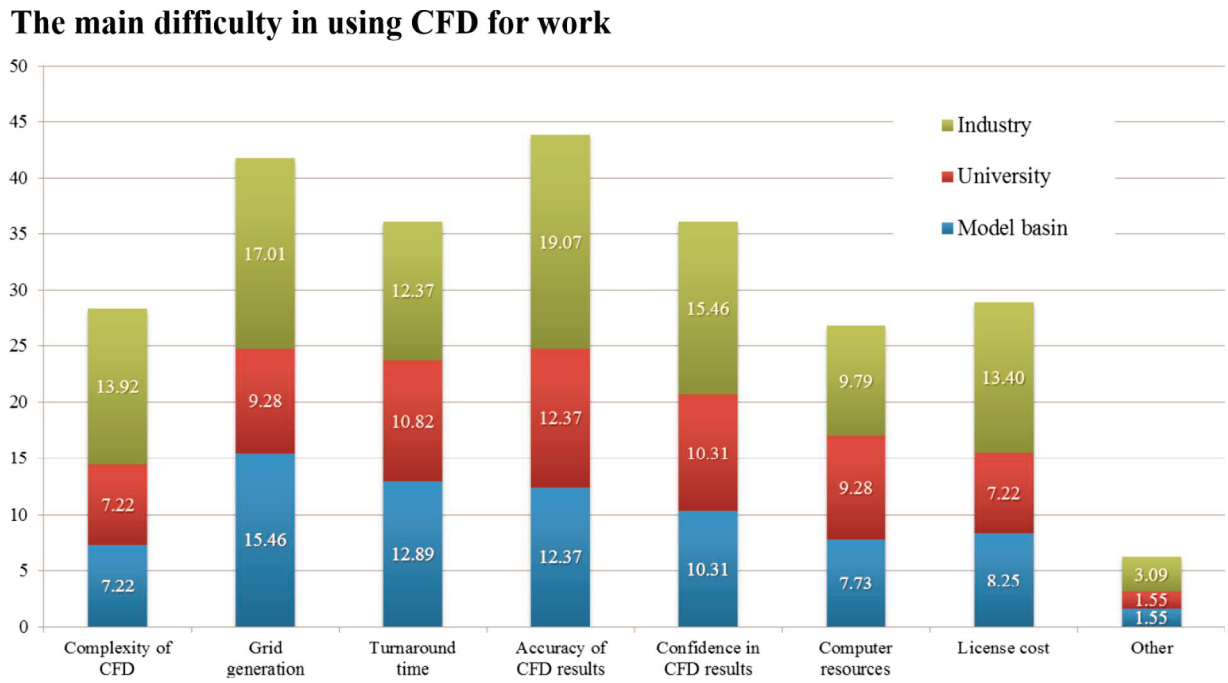


Figure 1.2: Main Difficulties in the Usage of CFD in Maritime Community. From [30].

within the CAD system are different to the requirements of a geometry representation for a CFD simulation. Most of the times the geometry in the CAD system is modelled very detailed down to a single screw of the assembly, which however is much to detailed for the CFD representation. On the other hand a CFD simulation usually requires a closed surface, which is usually not necessary within the CAD representation. The preparation of the CAD geometry to fulfil the CFD requirements can be time consuming. To improve this process the seamless integration of CFD tools in the CAE workbenches is required, where only one geometry representation is present or the transformation between different geometry representations is automatised. Based on the geometry a numerical mesh on which the flow problem is solved has to be created. Depending on the type of mesh (structured / block-structured / unstructured) the creation of the mesh can be a time consuming task. However, the numerical mesh has major impacts on the quality of the results and the robustness of the simulation. The creation of an appropriate mesh requires the knowledge of the solution of the flow problem. On the other hand, the solution of the flow problem is only possible with a numerical mesh. Therefore, the creation of the numerical mesh often turns into an iterative process, which is of course time consuming. This

is underlined by Slotnick et.al. [168] who state: *Today, the generation of suitable meshes for CFD simulations about complex configurations constitutes a principal bottleneck in the simulation workflow process. Often the mesh generation phase constitutes the dominant cost in terms of human intervention, and concerns about the cost and robustness of mesh generation were raised repeatedly [...]* . Accordingly, the mesh generation requires a sound knowledge, which is usually provided by expert users. The mesh generation process may be improved by automated mesh adaptation procedures, which adjust the mesh during the simulation to the specific requirements. When the mesh has been generated the flow problem is solved numerically. Depending on the flow problem and the selected flow models, this may be a time consuming task. The choice of the flow models moreover affects the accuracy of the results and the stability of the simulation. The numerical simulation can be enhanced by the selection of the appropriate flow models, which requires expert knowledge, and by improved flow models which are adapted to specific problems. Once the flow problem has been numerically solved the results need to be post-processed to extract the required informations. Depending on the mesh size and the required informations this task may be time consuming. However, it is possible to automatise large parts of the post-processing process to minimise the required time. Table 1.1 summarises the impact of the individual steps of the CFD process on the above mentioned requirements for the integration of CFD in the design process and the associated optimisation tasks.

Table 1.1: The CFD process and the associated optimisation potential.

Step	Affects	Optimisation Tasks
Geometry Preparation	Time to Solution	CAD/CAE Integration
Grid Generation	Time to Solution	Adaptive Mesh Refinement
	Result Quality	
	Stability	
Numerical Solution	Time to Solution	Improved CFD models
	Result Quality	Improved Hardware
	Stability	
Post-Processing	Time to Solution	Automation

From the optimisation tasks identified in Table 1.1, the following two are covered within

this work, because it is deemed that they have a major impact:

- Adaptive Mesh Refinement
- Improved CFD Models

According to Slotnick et. al. [168] *adaptive mesh techniques offer great potential, but have not seen widespread use due to issues related to software complexity, inadequate error estimation capabilities, and complex geometries.* The potential of adaptive mesh techniques is enormous, especially in the marine industry where many problems involve dynamic or even violent free-surface flows, which require a substantial grid resolution localised to the free-surface. Therefore, the integration of automated adaptive mesh refinement techniques into a flow solver capable to solve flows around complex geometries is demonstrated within this work. Even more important than the mesh adaptation itself is question where to refine the mesh. Typically accurate results for the quantities in question are required. However, the accuracy of the results of the simulation hinges on the utilised numerical mesh. Simply said, sufficient mesh resolution is required in the areas of interest and in the areas with large numerical errors that influence the areas of interest. A technique to provide a refinement indicator that identifies these regions and also provides an approximation of the uncertainty in the result in question is goal orient mesh refinement, which is applied in the present work. The ultimate goal of adaptive mesh refinement techniques can be summarised as follows: *Given a suitable geometry representation and a desired level of solution accuracy, a fully automated meshing capability would construct a suitable mesh and adaptively refine this mesh throughout the solution process with minimal user intervention until the final accuracy levels are met.* [168]

The improvement of CFD models within this work is confined to the modelling of the free-surface flow. Within RANS based flow solvers the Volume of Fluid (VoF) method is most frequently used to simulate free-surface flows. However, compared to single phase flows the required computational effort for these methods is much larger. This is mainly caused by the limitations of the underlying discretisation practices. To include free-surface effects using the VoF method transient simulations are required, even when the required result is of quasi-static nature (e.g wave pattern of a ship in still water). Furthermore, the VoF discretisation scheme requires extremely small time steps to fulfil the Courant number limitations, which significantly increases the required numerical effort. When the Courant number limitation is not fulfilled the VoF method either produces largely blurred surfaces

or gets unstable. Within the present work the above mentioned problems associated to the VoF method are tackled by improved and additional models.

1.2 Starting Point and Aim of the Thesis

The overall goal of this thesis is the improvement of the the accuracy and efficiency of RANS based finite volume schemes especially for maritime flow problems. Therefore, methods to enhance the grid generation process and the free-surface flow prediction have been developed.

The developed methods have been implemented in the in-house flow solver FreSCo$^+$, which uses a second-order accurate finite volume discretisation of the Reynolds Averaged Navier Stokes (RANS) equations in space and time. It supports arbitrary shaped three-dimensional cells in an unstructured grid environment. The pressure-velocity coupling is achieved by a SIMPLE-type projection scheme. The code is parallelised using the MPI-protocol in the Single Program Multiple Data (SPMD) paradigm. Domain decomposition is achieved via the PARMETIS library. Base-line implementations of the Volume of Fluid (VOF) method, to include multiple phases are available. Furthermore, the code is supplemented by an adjoint solver, which has been developed in a previous work by Stück [171].

1.3 Present Contributions

The specific contributions to the prediction of free-surface flows with RANS based flow solvers in the marine context are summarised as follows:

- A sub-cycling technique, which can operate in an unsynchronised mode to speed up the time to solution, is developed for the sequential computational procedure. The time integration of the mixture fraction equation and the remaining considered equations is decoupled to achieve a maximum speedup for simulations, which aim at quasi steady results. A theoretically achievable speedup is derived to verify and validate the implemented procedure.

- Two main sources that lead to (partly) blurred interfaces in free-surface simulations using standard compressive interface capturing schemes are commonly identified.

These sources are high Courant numbers and unfavourable alignments of the flow and the numerical grid. Both sources can hardly be avoided in the standard unstructured grid environment with the use of reasonable timesteps particularly for violent flows which feature dynamic interfaces. Hence, an Explicit Interface Sharpening (EIS) technique is developed, which is designed as an add-on technique. The approach analyses the sharpness of the interface and redistributes the mixture fraction in a locally and globally conservative manner where necessary, to obtain a sharp interface. Any convection scheme supplemented with EIS is at least partly turned into a compressive scheme.

Specific contributions to the field of automatic grid refinement are:

- A procedure to adapt cells in the numerical mesh in an anisotropic manner is developed. The procedure ensures topological and geometrical validity of the mesh at all instants of time. The mesh adaptation procedure is fully integrated into the parallel environment of the flow solver.

- Refinement of boundary layer cells requires special attention in order to be consistent with the boundary layer approximation. The present mesh adaptation technique is adjusted to not disturb the mesh in the boundary layer depending on the applied boundary layer approximation.

- Mesh refinement in the vicinity of complex curved three-dimensional bodies needs to take into account the shape of the body. The present technique is able to align the mesh to the geometrical representation of the body, which needs to be present in the simulation.

- A new refinement indicator for free-surface flows has been developed. Unlike other published refinement indicators, it is not based on the gradients of the mixture fraction, because gradients especially in unstructured grids are afflicted with uncertainties.

- A technique to combine several different refinement indicators is developed.

- A refinement indicator for goal-oriented grid refinement in three-dimensional turbulent high Reynolds number flows in the finite volume framework is developed.

All developed methods are intensively verified and validated using a wide range of suitable test cases.

1.4 Outline of the Thesis

The mathematical background of the flow solver is introduced in Chapter 2. Suitable discretisation techniques within the finite volume framework are introduced in Chapter 3. The chapter is followed by a discussion of error sources in the finite volume framework in Chapter 4, which is necessary to identify areas afflicted with rather large errors and hence possibilities for improvement. Chapter 5 starts with a review of the state of the art for free-surface flows in the finite volume framework and concludes with the introduction of the sub-cycling technique and the Explicit Interface Sharpening (EIS) technique. The dynamic grid adaptation procedure and the goal-oriented grid adaptation is introduced in Chapter 6. The chapter also starts with a review of the state of the art in dynamic grid adaptation. Chapter 7 presents the verification and validation studies for the developed techniques, while Chapter 8 shows application examples in which the developed techniques are used to enhance the accuracy and efficiency. The closure of the thesis provides a summary and gives an outlook with regard to the developed techniques.

Throughout the thesis index notation is used for formulas – Einstein's sum convention applies to small-type Latin subscripts, unless declared differently. A glossary of terms and abbreviations used within the thesis is provided on page iv to xii.

2 Mathematical Model

For the simulation of fluid flow the fluid is considered as a continuum, because the characteristic time- and length-scales for flow problems are usually much larger than the scales associated with the individual molecules of the matter. The mechanics of continua are described by balance equations, which are introduced in the first part of this chapter. For the closure of these balance equations constitutive relations are necessary. They describe the relation between two physical properties of a matter and approximate the response to an external force. Constitutive relations for Newtonian fluids are introduced in the second part of this chapter. The last part of this chapter introduces the employed basics for the statistical modelling of turbulent flows.

2.1 Transport Equations

In fluid mechanics the balance laws are usually considered in the Eulerian reference frame, rather than in the Lagrangian reference frame. An Eulerian representation of the material derivative that describes the rate of change of an intensive physical property ϕ within a material volume $\mathcal{V}$ in time reads

$$\frac{d}{dt}\int_{\mathcal{V}} \phi dV \rightarrow \int_{\mathsf{V}} \frac{\partial \phi}{\partial t}\, dV + \oint_{O(\mathsf{V})} (u_i \phi)\, dF_i = S_\phi. \tag{2.1}$$

Here, the source term S_ϕ describes the production, dissipation or redistribution of ϕ. Mind that Einsteins summation convention and Cartesian coordinates are used. The Leibnitz integration rule is applied because the borders of the integral and integrand are time-dependant. V is the volume instantaneously coincident with the material volume, u_i is velocity vector and dF_i an outside pointing unit normal vector on a surface of the material volume $O(\mathsf{V})$.

Using the Gauss' theorem an alternative formulation can be written as

$$\int_{\mathsf{V}} \left[\frac{\partial \phi}{\partial t} + \frac{\partial}{\partial x_i} \left(u_i \phi \right) \right] \, dV = S_\phi. \tag{2.2}$$

In differential form it reads

$$\frac{\partial \phi}{\partial t} + \frac{\partial}{\partial x_i} \left(u_i \phi \right) = \frac{D\phi}{Dt} + \phi \frac{\partial u_i}{\partial x_i} = s_\phi, \quad \text{with} \quad s_\phi = \frac{S_\phi}{dV} \tag{2.3}$$

because Equation (2.2) is valid for arbitrary control volumes V in the framework of continuum mechanics. The substantial derivative $\frac{D\phi}{Dt}$ used in the above equation is defined as

$$\frac{D\phi}{Dt} = \frac{\partial \phi}{\partial t} + u_i \frac{\partial \phi}{\partial x_i}. \tag{2.4}$$

Conservation of Mass

Conservation of mass in a source free flow-domain ($S_\phi = 0$) is described by the transport of density ρ and reads

$$\int_{\mathsf{V}} \frac{\partial \rho}{\partial t} \, dV + \oint_{O(\mathsf{V})} \left(\rho u_i \right) dF_i = 0 \tag{2.5}$$

in integral formulation and

$$\frac{\partial \rho}{\partial t} + \frac{\partial u_i \rho}{\partial x_i} = 0 \tag{2.6}$$

in differential formulation.

Conservation of Linear Momentum

The change of linear momentum is equal to sum of external forces acting on the fluid according to Newtons first law.

$$\int_{\mathsf{V}} \frac{\partial (\rho u_i)}{\partial t} \, dV + \oint_{O(\mathsf{V})} \left(\rho u_i u_j \right) dF_j = - \oint_{O(\mathsf{V})} p dF_i + \oint_{O(\mathsf{V})} \tau_{ij} dF_j + \int_{\mathsf{V}} f_i \, dV \tag{2.7}$$

External forces are the static pressure p, the still unknown shear stress tensor τ_{ij}, which will be defined by the constitutive relations, and arbitrary body forces f_i, which may account for e.g. gravity. The differential formulation of the momentum equation reads

$$\rho \frac{Du_i}{Dt} = - \frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} + f_i. \tag{2.8}$$

Generic Transport Equation

Generic properties are transported according to the following equation

$$\rho \frac{D\phi}{Dt} = \frac{\partial}{\partial x_i}\left(\Gamma_\phi \frac{\partial \phi}{\partial x_i}\right) + s_\phi \tag{2.9}$$

with a property dependant (isotropic) diffusion coefficient Γ_ϕ.

2.2 Constitutive Relations

The set of balance equations introduced in the previous section is valid for any continuum. However, the number of unknown quantities is larger than the number of available equations. This gap is closed using the matter-dependant constitutive relations. In the present work only Newtonian fluids are considered. The viscous stress tensor τ_{ij} for Newtonian fluids in a Cartesian reference frame reads

$$\tau_{ij} = 2\mu S_{ij} - \frac{2}{3}\mu\left(\frac{\partial u_k}{\partial x_k}\right)\delta_{ij} \quad \text{, with} \quad S_{ij} = 0.5\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \quad \text{and} \quad \mu = \rho\nu \tag{2.10}$$

with μ being the dynamic viscosity and δ_{ij} the Kronecker delta

$$\delta_{ij} = \begin{cases} 1 & , \ i = j \\ 0 & , \ \text{otherwise.} \end{cases} \tag{2.11}$$

The transport coefficient μ is usually a function of the thermodynamic state variables

$$\mu = \mu(p, T) \tag{2.12}$$

where T is the temperature.

2.3 Turbulence Modelling

Most flows which are relevant for engineers are turbulent. Turbulent flow might be characterised as irregular and chaotic with stochastic property changes. They are always three-dimensional and unsteady even if the boundary conditions are steady. Unsteady interacting vortices appear on many scales ranging from the Kolmogorov micro-scales to

scales comparable with the size of the geometry. However, it is often still possible to separate the fluctuations from the mean (transient) flows, and for most technical applications only the statistic mean values of a flow are relevant.

Turbulent flows can be simulated using the Direct Numerical Simulation (DNS) technique in conjunction with Equations (2.1) to (2.10). The governing equations are integrated for the whole range of turbulent scales which puts high demands on the required spatial and temporal resolution. Therefore, this technique is not feasible for engineering applications.

Another possibility is to use low-pass filtering to separate different length scales in a turbulent flow domain. Large scale structures can be resolved by integration of the governing equations, while the influence of all other unresolved or filtered scales is modelled. This technique is usually known as Large Eddy Simulation (LES). Small scales of turbulence are deemed more homogeneous and isotropic and thus easier to model.

The last possibility presented within this work is statistical modelling. In fluid dynamics two popular averaging techniques are used. Reynolds averaging (unweighted) is most used for incompressible flow while Favré averaging (density-weighted) is usually used for compressible flows. This work considers only incompressible flow, therefore only Reynolds averaging is presented.

The Reynolds average of an arbitrary quantity $\mathcal{Z}$ reads

$$\tilde{\mathcal{Z}} = \mathcal{Z} + \mathcal{Z}' \quad \text{with} \quad \overline{\tilde{\mathcal{Z}}} = \mathcal{Z} \quad , \quad \overline{\mathcal{Z}'} = 0 \tag{2.13}$$

with $\tilde{\mathcal{Z}}$ being the instantaneous value, $\mathcal{Z}$ the average value and $\mathcal{Z}'$ the fluctuation. The average value of the ensemble is defined as

$$\mathcal{Z}\left(x_i, t\right) = \lim_{N\to\infty}\left(\frac{1}{N}\sum_{n=1}^{N}\tilde{\mathcal{Z}}_n\left(x_i, t\right)\right). \tag{2.14}$$

The decomposition of the most important variables within this work read

$$\tilde{p} = p + p', \quad \tilde{u_i} = u_i + u_i', \quad \tilde{\rho} = \rho + \rho', \quad \tilde{\phi} = \phi + \phi'. \tag{2.15}$$

Substituting all variables in the balance equations given in Section 2.1 with the statistical equivalent and averaging the balance equations, assuming non-fluctuating densities, results in the Reynolds-averaged transport equations

$$\frac{D\rho}{Dt} = \frac{\partial\rho}{\partial t} + u_i\frac{\partial\rho}{\partial x_i} = 0 \qquad \text{and} \quad \frac{\partial u_i'}{\partial x_i} \equiv 0 \tag{2.16}$$

$$\rho\frac{Du_i}{Dt} = -\frac{\partial p}{\partial x_i} + \frac{\partial\overline{\tau_{ij}}}{\partial x_j} - \rho\frac{\partial\overline{u_i'u_j'}}{\partial x_j} + \overline{f_i}. \tag{2.17}$$

The difference compared to the transport equations for the instantaneous values are the second moments $\overline{u_i' u_j'}$. The Reynolds-averaged shear stress tensor reads

$$\overline{\tau_{ij}} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) = 2\mu S_{ij}. \tag{2.18}$$

Further modelling is necessary to close the system. The second moments can be either closed using a correlation between the Reynolds stresses $\rho\overline{u_i' u_j'}$ and the gradients of the mean velocity by an eddy-viscosity model or solving the exact transport equations for the second moments.

In the present work eddy viscosity turbulence models have been used. The most popular approximation for this approach is the Boussinesq approximation, describing a linear relation as

$$\overline{u_i' u_j'} = \nu_t \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) + \frac{2}{3} k \delta_{ij} \tag{2.19}$$

with

$$k = \frac{1}{2} \overline{u_i' u_i'}. \tag{2.20}$$

In order to close the system the isotropic eddy-viscosity ν_t has to be modelled. The modelling process is often divided in a turbulent length scale L_t and a turbulent velocity scale V_t

$$\nu_t \sim L_t \cdot V_t. \tag{2.21}$$

There are many ways to approximate L_t and V_t. If ν_t is approximated by a purely algebraic function, the turbulence model is called an algebraic turbulence model. Models which are based on the transport of one or two additional variables are called one-equation-models or two-equation-models, respectively. Some popular models which have been used within this work will be briefly presented.

$k - \epsilon$ Model

Using the $k - \epsilon$ model from Jones and Launder [96], ν_t is expressed as a function of the turbulent kinetic energy k and its dissipation rate ϵ

$$\nu_t = \overline{c}_\mu \frac{k^2}{\epsilon} \tag{2.22}$$

with $\overline{c}_\mu$ being a model constant. The transport equation for the turbulent kinetic energy – omitting buoyancy effects – reads

$$\frac{Dk}{Dt} = P - \epsilon + \frac{\partial}{\partial x_k}\left[\left(\nu + \frac{\nu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_k}\right] \tag{2.23}$$

where P is the production of turbulent kinetic energy and σ_k a model constant. The production of term P reads

$$P = -\overline{u_i' u_j'}\frac{\partial u_i}{\partial x_j} = \nu_t \mathcal{S}^2 \quad \text{with} \quad \mathcal{S} = \sqrt{2S_{ij}S_{ij}}. \tag{2.24}$$

The modelling of the energy transfer relates to the transport equation of the turbulent kinetic energy and reads

$$\frac{D\epsilon}{Dt} = \frac{\epsilon}{k}\left(C_{\epsilon 1}P - C_{\epsilon 2}\epsilon\right) + \frac{\partial}{\partial x_k}\left[\left(\nu + \frac{\nu_t}{\sigma_\epsilon}\right)\frac{\partial \epsilon}{\partial x_k}\right] \tag{2.25}$$

with $C_{\epsilon 1}$, $C_{\epsilon 2}$ and σ_ϵ being coefficients which are calibrated based on analytical solutions for fundamental flows. The most popular coefficients found in literature are

$$\sigma_k = 1.0, \quad \overline{c}_\mu = 0.09, \quad C_{\epsilon 1} = 1.44, \quad C_{\epsilon 2} = 1.92, \quad \sigma_\epsilon = 1.3.$$

Equation (2.25) is only valid for fully turbulent flow (so called high Reynolds number flow) fields which are located acceptably far away from no-slip walls. The viscous boundary layer regime at the wall is not resolved using this approach but modelled using wall functions. Wall functions mimic the near-wall behaviour of the velocity, the turbulent kinetic energy and its dissipation rate and assume that the near wall flow behaves uniquely, e.g. like a fully developed flat plate boundary layer.

So called low-Reynolds number versions of the $k-\epsilon$ model, which are able to resolve the near wall field are available in literature, but have not been used within this work.

The standard $k-\epsilon$ model performs well for free-shear layer flows with moderate pressure gradients [18], however its accuracy decreases when the flow contains large adverse pressure gradients [199].

Wilcox $k-\omega$ Model

The $k-\omega$ model developed by Wilcox [199] is another popular turbulence model. The eddy-viscosity is defined as

$$\nu_t = \frac{k}{\omega} \tag{2.26}$$

within this model. The transport of turbulent kinetic energy follows Equation (2.23), where ϵ can be rewritten as

$$\epsilon = \beta^* \omega k. \tag{2.27}$$

The transport equation for the specific dissipation rate ω reads

$$\frac{D\omega}{Dt} = \alpha \frac{\omega}{k} P - \beta \omega^2 + \frac{\partial}{\partial x_k} \left[(\nu + \sigma_\omega \nu_t) \frac{\partial \omega}{\partial x_k} \right]. \tag{2.28}$$

The coefficients can be determined along the route of the $k - \epsilon$ framework. Wilcox recommends to use

$$\alpha = \frac{5}{9}, \quad \beta = \frac{3}{40}, \quad \beta^* = 0.09, \quad \sigma_k = 2, \quad \sigma_\omega = 0.5.$$

The $k-\omega$ model has its advantages in modelling the flow near the wall. It is able to capture complex boundary layer flows with adverse pressure gradients. However, it is sensitive to the definition of the far field and boundary conditions of the length-scale variables, which is not trivial.

Menter Shear Stress Transport Model

The Shear Stress Transport Model, developed by Menter[118, 119], is a zonal model which aims on using the $k - \omega$ model near the wall and the $k - \epsilon$ model in the outer boundary layers and the free-stream and thus combines the advantages of both models. Moreover, it addresses non-equilibrium effects associated to high non-dimensional strain rates using the Bradshaw hypothesis [28]. It was developed to accurately predict aeronautical flows with separation and large adverse pressure gradients. The following formulas refer to the newer version of the MSST model published by Menter [119].

The transport equation for the turbulent kinetic energy reads

$$\frac{Dk}{Dt} = \tilde{P} - \beta^* \omega k + \frac{\partial}{\partial x_k} \left[\left(\nu + \sigma_{(k)} \nu_t \right) \frac{\partial k}{\partial x_k} \right] \tag{2.29}$$

where, $\tilde{P}$ is the limited production term

$$\tilde{P} = min\left(P, 10\beta^* \rho \omega k\right). \tag{2.30}$$

The specific dissipation rate is transported by

$$\frac{D\omega}{Dt} = \frac{\alpha}{\nu_t \rho} \tilde{P} - \beta \omega^2 + \frac{\partial}{\partial x_k} \left[\left(\nu + \sigma_{(\omega)} \nu_t \right) \frac{\partial \omega}{\partial x_k} \right] + 2\left(1 - F_1\right) \frac{\sigma_{(\omega)2}}{\omega} \frac{\partial k}{\partial x_k} \frac{\partial \omega}{\partial x_k} \tag{2.31}$$

with σ_k, α, β and $\sigma_{(\omega)}$ being responsible to blend between the $k-\epsilon$ and $k-\omega$ model.

$\sigma_{(k)}$, $\sigma_{(\omega)}$, α and β are constants for the two base models, and are blended between inner (1) and outer (2) regions by

$$\Phi = F_1 \Phi_1 + (1 - F_1)\, \Phi_2. \tag{2.32}$$

The function F_1 which controls the blending reads

$$F_1 = tanh\left(arg_1^4\right) \tag{2.33}$$

with

$$arg_1 = min\left[max\left(\frac{\sqrt{k}}{\beta^* \omega y}, \frac{500\nu}{y^2\omega}\right), \frac{4\rho\sigma_{(\omega)2}k}{CD_{k\omega}y^2}\right] \tag{2.34}$$

and

$$CD_{k\omega} = max\left(2\rho\sigma_{(\omega)2}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}, 10^{-10}\right), \tag{2.35}$$

where y is the near wall distance.

The turbulent eddy-viscosity μ_t is determined by

$$\mu_t = \frac{\rho a_1 k}{max\left(a_1\omega, \mathcal{S}F_2\right)} \quad \text{with} \quad \mathcal{S} = \sqrt{2S_{ij}S_{ij}} \tag{2.36}$$

where F_2 is another zonal blending function that reads

$$F_2 = tanh\left(arg_2^2\right) \tag{2.37}$$

with

$$arg_2 = max\left(2\frac{\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right). \tag{2.38}$$

The constants for the Shear Stress Transport model read

$$\begin{aligned} \alpha_1 &= \frac{5}{9} & \alpha_2 &= 0.44 \\ \sigma_{k1} &= 0.85 & \sigma_{k2} &= 1.0 \\ \sigma_{\omega 1} &= 0.5 & \sigma_{\omega 2} &= 0.856 \\ \beta_1 &= 0.075 & \beta_2 &= 0.0828 \\ \beta^* &= 0.09 & \kappa &= 0.41 \quad a_1 = 0.31. \end{aligned}$$

2.4 The Dual Problem

A CFD simulation is usually used to estimate a functional output (e.g. drag) for a physical problem (e.g. flow around a vessel). Typically the functional output has to be estimated with a certain accuracy, preferably using the lowest effort. The achievable accuracy is directly coupled to the discretisation of the solution domain – using smaller elements reduces the error. However, using smaller elements the number of elements and therefore the effort increases. The distribution of the error in the solution domain especially for complex flows is not known a priori, therefore the selection of a reasonable discretisation of the solution domain is a challenging task. Introducing techniques that can estimate the local error during the solution process, and combining them with procedures to automatically adapt the discretisation of the solution domain, is an efficient technique to reduce the difficulties in creating a suitable discretisation of the solution domain. However, this technique does not relate the discretisation error to the error in the predicted functional and therefore may not lead to an optimal discretisation of the solution domain. A relationship between the error and its influence on a specific output functional is provided by the adjoint equations, which are widely used in the field of optimisation.

Optimisation strategies aim to evaluate the variation of the output functional with respected to the input variables (e.g. discretised shape of the vessel). Direct methods perform flow calculations for each perturbed input parameter (e.g each face of the discretised hull form) and therefore the effort scales with the number of input parameters. On the other hand adjoint methods evaluate the variation of the output functional with respect to all input parameters by solving the adjoint problem. The effort of solving the adjoint Navier-Stokes equations is in the same order as the effort of one flow calculation. The derivation of the adjoint Navier-Stokes solver follows two main approaches published in literature. The *discrete* approach (e.g. [59]) is based on the *discretise-then-derive* paradigm. The adjoint code is derived from the discretised Navier-Stokes equations, by transposing the linearised operators. This technique is the basis for the *automatic differentiation* approach (e.g. [64, 122]), which aims to generate the adjoint flow solver automatically by source-to-source pre-compilation. The *continuous* (e.g. [87]) approach follows the *derive-then-discretise* philosophy. The adjoint Navier-Stokes equations are derived and subsequently discretised. The advantages and shortcomings of both approaches have been discussed by many authors [60, 126, 144, 171]. Using the *discrete* approach

the exact gradient of the discrete objective function is obtained, while the *continuous* approach provides an approximation of the continuous gradient. Furthermore, the *discrete* approach allows an automatic derivation of the adjoint Navier-Stokes solver. However, it requires the complete Jacobian matrix of the discretisation, which is usually not available for face-based finite-volume schemes or cumbersome manual summation by parts operations, to provide the exact gradient. The *continuous* approach facilitates the mathematical and physical understanding of the adjoint equations. Code obtained from this approach has generally a lower operation count, lower memory requirements and a simpler code structure. Large portions of the code for the primal problem can be reused, including structures for parallel simulations and dedicated libraries (e.g. for solving the equation system). However, the derivation of the continuous adjoint equations for transport equations associated with complex source terms or singularities at boundaries (e.g. turbulence equations) and bounded field variables (e.g. concentration equation) remains a challenging task. The *hybrid* differentiation technique (e.g. [126]) aims to combine the merits of both approaches. The *discrete* adjoint approach can be used to derive consistent discretisation methods for the *continuous* adjoint equations. Furthermore, the *discrete* approach can be used to include transport equations or functionals that can hardly be handled by the *continuous* approach. The adjoint Navier-Stokes solver, which has been developed by Stück [171] and forms the basis for the goal-oriented mesh refinement technique, follows the *hybrid* approach. It uses the *frozen turbulence* assumption and hence neglects the influence of the variation of the turbulence quantities, caused by a perturbation of the input variables (e.g. shape), on the functional. A discussion of the consequences for the goal-oriented grid refinement technique follows in Section 6.3.3. This section briefly describes the derivation of the continuous adjoint Navier-Stokes equations using Lagrangian multipliers. A detailed derivation is provided by Stück [171].

The starting point is the optimisation problem for a specific objective functional

$$J = \int_{F_o} j_F dF + \int_V j_V dV \tag{2.39}$$

under the constraint that the steady state incompressible Navier-Stokes equations

$$R_i = \rho u_j \frac{\partial u_i}{\partial x_j} + \frac{\partial p}{\partial x_i} - \frac{\partial \tau_{ij}}{\partial x_j} - f_i = 0 \tag{2.40}$$

and

$$Q = -\frac{\partial u_i}{\partial x_i} = 0, \tag{2.41}$$

presented in their local residual formulation, are satisfied. The objective functional may be an arbitrary function defined in the Volume V and/or the surface F_o. Prominent examples for objective functionals in marine hydrodynamics refer to drag or wake homogeneity objective functions. To derive the adjoint Navier-Stokes equations, the objective functional is augmented with the volume integral of the Navier-Stokes equations weighted by the Lagrangian multipliers $\hat{u_i}$ and $\hat{p}$

$$L = J + \int (\hat{u_i} R_i + \hat{p} Q) dV \tag{2.42}$$

also known as adjoint variables. The adjoint Navier-Stokes equations are derived through a variation of the extended objective functional and a subsequent integration by parts, which are provided in detail by Stück [171]. Assuming frozen turbulence properties, the resulting steady adjoint Navier-Stokes equations for incompressible flow read

$$-\rho u_j \frac{\partial \hat{u}_i}{\partial x_j} = \frac{\partial}{\partial x_j} \left(2\mu_{eff} \hat{S}_{ij} - \hat{p}\delta_{ij} \right) - \rho \hat{u}_j \frac{\partial u_j}{\partial x_i} - \frac{\partial j_\Omega}{\partial u_i} \tag{2.43}$$

and

$$\frac{\partial \hat{u}_i}{\partial x_i} = \frac{\partial j_\Omega}{\partial p} \tag{2.44}$$

where $\frac{\partial j_\Omega}{\partial u_i}$ and represent $\frac{\partial j_\Omega}{\partial p}$ the source terms originating from the objective functional. As these equations are derived using integrations by parts, they are supplemented by a number of boundary integrals [173].

3 Finite-Volume Technique

3.1 Discretisation of the Solution Domain

The discretisation of the solution domain is decomposed of the discretisation of space and time. The result of the discretisation is a set of vertices in space and time, to which the numerical grid is connected.

Time is a parabolic coordinate, accordingly the solution is obtained marching in time from prescribed initial values. For the discretisation of the time marching procedure a time step has to be defined.

For the finite volume (FV) technique, the domain is subdivided into a finite number of control volumes, which do not overlap and completely fill the domain. The control volumes can have an arbitrary polyhedral shape depicted in Figure 3.1. They are bounded by faces

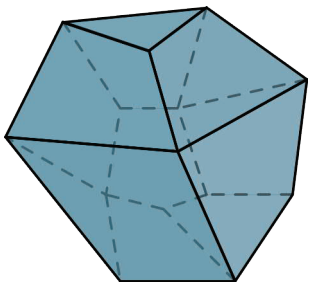


Figure 3.1: A control volume represented by an arbitrary polyhedron.

which have a polygonal shape. Faces are either classified as internal faces or boundary faces. Boundary faces are located at the boundary of the domain and have exactly one adjacent cell, while internal faces are located inside the domain and have exactly two adjacent cells. Both types of faces with its face normal vectors are presented in Figure

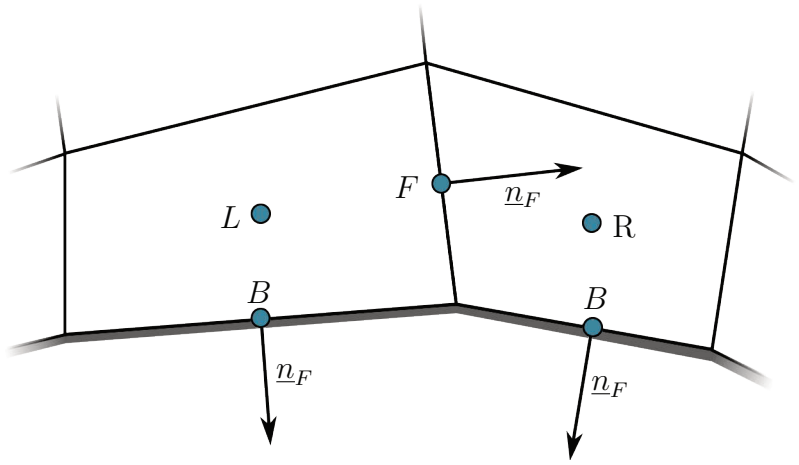
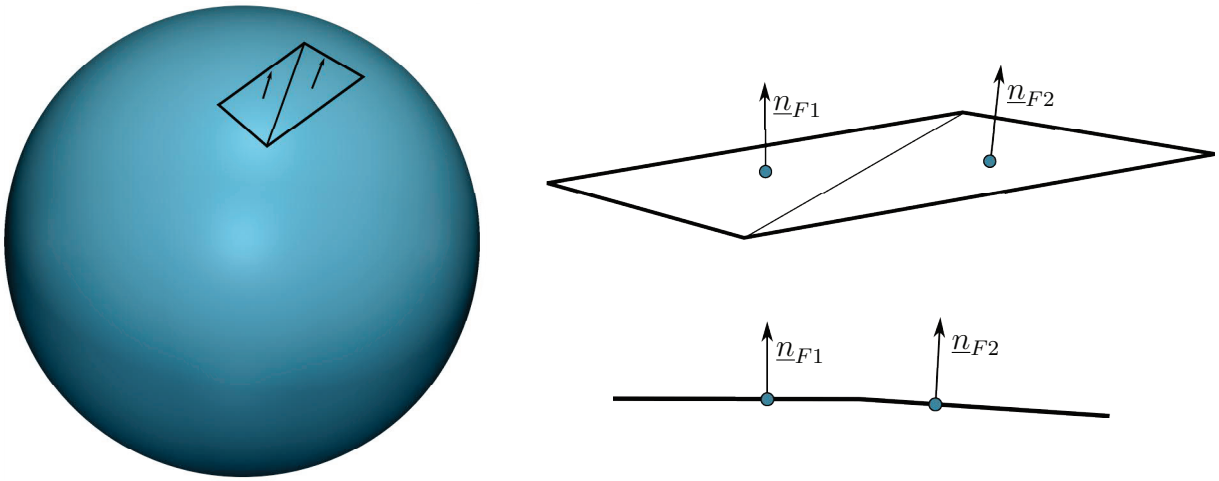


Figure 3.2: Illustration of internal boundary faces.

3.2. Face normal vectors of boundary faces point outside the domain by definition. The discretisation of the governing equations relies on face normal vectors pointing outward of the cell. However, the direction of internal face normal vectors is random. Therefore, it is necessary for each cell to know if the face normal vector is pointing inside or outside the cell. For each cell a list of surrounding faces exists, where faces pointing outside the cell have a positive sign and faces pointing inside the domain a negative sign. Polygonal faces are not always perfectly flat. Especially boundary faces, where the vertices are placed on an arbitrarily shaped domain boundary, are usually non-planar as shown in Figure 3.1. Hence, it can not be ensured that all cells have a convex shape and the evaluation of geometric properties has to be able to deal with non-planar faces and concave cells.

The centre of a face is located in its barycentre. To account for the non-planarity of faces the evaluation of the barycentre is a two-step process. First a temporary face centre is evaluated as the arithmetic mean of the coordinates of the connected vertices. Then the barycentre's for all triangles, that can be constructed by the edges of the face and the temporary face centre, weighted by their area are summed up, and divided by the sum of all triangle areas. The same procedure is used to determine the face normal. A similar two step approach is used to calculate the cell centres. A preliminary cell centre is first assigned to the arithmetic mean of all corner vertices. Subsequently the preliminary cell centre is used to build up tetrahedrons from which the final cell centre is computed.

Flow variables are stored in the cell centres in a collocated manner. The use of unstruc-



(a) Arbitrary Face on sphere.

(b) Isometric and side view of face.

Figure 3.3: Non-planar face formed by 4 vertices split into 2 triangles.

tured grids simplifies the grid generation for complex geometries. Furthermore, it provides the flexibility to enrich the grid in regions of interest.

3.2 Approximation Techniques

The numerical evaluation of the flow field needs a discretised form of the transport equations. The transport equations involve integrals over faces and cells and contain the flow variables and their gradients.

Approximation of Integrals

Integrals are approximated to second-order accuracy using the mid point rule

$$\int_V \phi \, dV \approx \phi_P \Delta V \text{ and } \int_F \phi dF \approx \phi_F \Delta F. \tag{3.1}$$

The discretisation of an integral of a flow variable over a cell is straightforward, as the flow variables are stored in the cell centres. However, for the surface integrals the flow variables have to be interpolated to the face centre. Interpolation techniques for face values are presented in the next section.

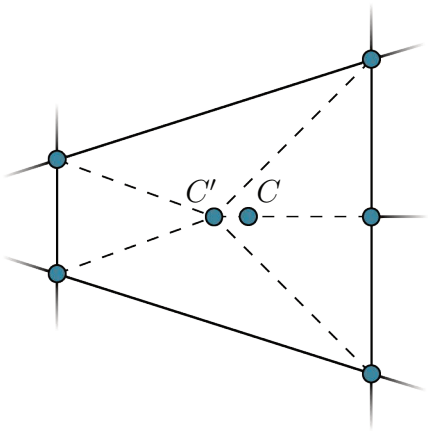
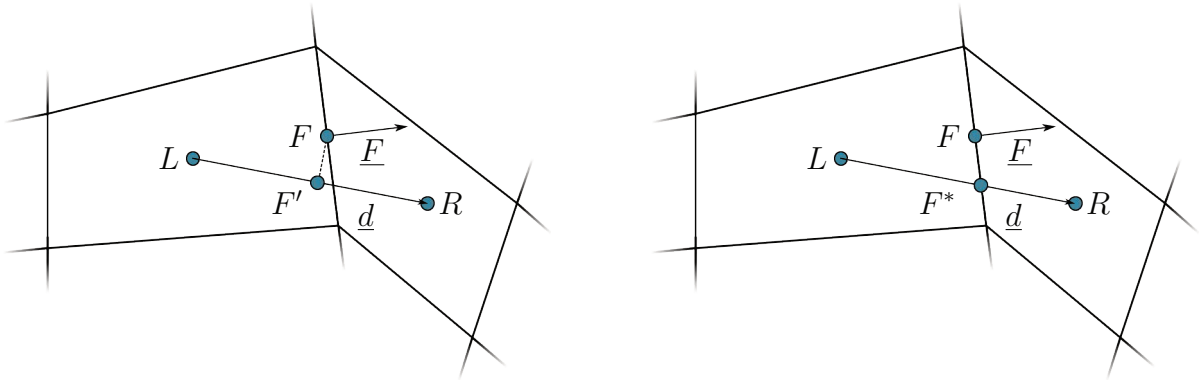


Figure 3.4: Subdivision of a face into triangles to determine its barycenter.

Interpolation of Face Values

The value at the face is evaluated by a simple, second-order accurate, linear 1D-interpolation from the known left and right neighbour cells to an intermediate point located on the line connecting the two cell centres. This intermediate point does not necessarily coincide with the face centre as shown in Figure 3.5, accordingly a subsequent "non-orthogonality" correction is introduced.



(a) Orthogonal projection.

(b) Intersection point approach.

Figure 3.5: Interpolation to face center F from adjacent cells L and R.

$$\phi_F = \underbrace{(1-\lambda)\,\phi_L + \lambda\phi_R}_{\text{linear interpolation}} + \underbrace{\left(x_{i(F)} - x_{i(F')}\right)\left(\frac{\partial\phi}{\partial x_i}\right)_{F'}}_{\text{non-orthogonality correction}} \tag{3.2}$$

The gradient used for the correction is interpolated to the intermediate point using the linear interpolation scheme.

$$\left(\frac{\partial\phi}{\partial x_i}\right)_{F'} = (1-\lambda)\left(\frac{\partial\phi}{\partial x_i}\right)_L + \lambda\left(\frac{\partial\phi}{\partial x_i}\right)_R \tag{3.3}$$

The choice of the intermediate point on the connecting line is not unique. An intuitive choice is the point which has the smallest distance to the face centre. The vector from this intermediate point to the face centre is orthogonal to d_i connecting the cell centres. The corresponding interpolation weight λ reads

$$\lambda = \frac{\left(x_{i,(F)} - x_{i,(L)}\right) d_i}{d_i^2}. \tag{3.4}$$

An alternative choice is the intersection point of d_i and the face. This point is located on the connecting line d_i

$$x_{i(F^*)} = x_{i(L)} + \lambda d_i \tag{3.5}$$

and on the surface

$$\Delta F_i \left(x_{i(F^*)} - x_{i(F)}\right). \tag{3.6}$$

The combination of these criteria yields the definition of the interpolation weight

$$\lambda = \frac{\Delta F_i \left(x_{i(F)} - x_{i(L)}\right)}{\Delta F_i \; d_i}. \tag{3.7}$$

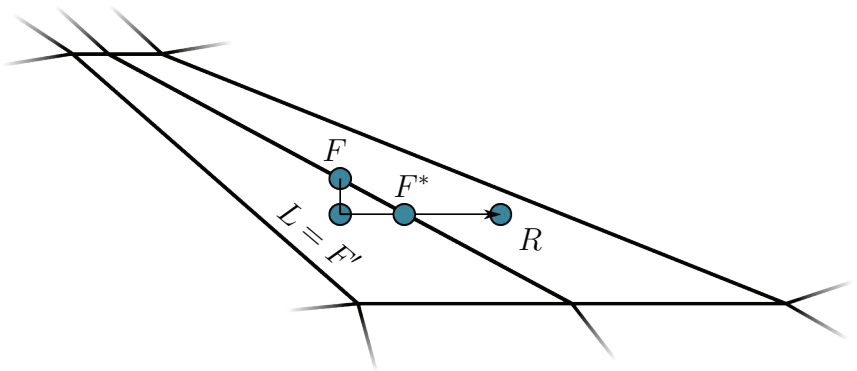


Figure 3.6: Exaggerated illustration of distort mesh situation.

Although the distance from the intersection point to the face centre is larger, the interpolation using the piercing point has shown to be more stable. Figure 3.6 shows an

exaggerated example of a distorted mesh. The value at the face centre would be completely determined by values of the left cell when the minimal distance correction technique is applied. The correction technique based on the intersection point determines the face value from values of both neighbour cells.

Approximation of Gradients

A second-order accurate approximation of the gradient in a control volume centre, that obeys the conservation principle, can be derived from the application of the mid point rule to the Gauss-Green theorem

$$\int_{\mathsf{V}} \frac{\partial \phi}{\partial x_i}\, dV = \oint_{O(\mathsf{V})} \phi dF_i. \tag{3.8}$$

The discrete gradient is evaluated by

$$\frac{\partial \phi}{\partial x_i} \approx \frac{1}{\Delta V} \sum_{F(P)} \left(\phi_F \Delta F_i\right) \tag{3.9}$$

and depends on the values of ϕ at the face centres. As shown in the previous section the values at the face centres depend on the gradient in the cell centres. Therefore, an iterative procedure, which is not necessarily converging, is necessary to accurately evaluate the face values and gradients.

An alternative method to approximate the gradients, which is also second-order accurate and does not require an iterative procedure is provided by a pre-conditioned Least-Square based method. To evaluate the gradient, an asymmetric over-determined equation system for $\frac{\partial \phi}{\partial x_i}$ at the control volume centre, involving all neighbouring control volume locations i is assembled.

$$\frac{1}{d_{(i)}} \underbrace{\left[\phi_i - \phi_P\right]}_{(\Delta\phi)_i} = \frac{1}{d_{(i)}} \underbrace{\left[x_{k(i)} - x_{k(P)}\right]}_{A_{ik}} \left(\frac{\partial \phi}{\partial x_k}\right)_P \tag{3.10}$$

It is regularised by an algebraic manipulation

$$\frac{1}{d_{(i)}} A^T_{mi} \left(\Delta\phi\right)_i = \underbrace{\frac{1}{d_{(i)}} A^T_{mi} A_{ik}}_{B_{mk}} \left(\frac{\partial \phi}{\partial x_k}\right)_P \tag{3.11}$$

and finally solved

$$\frac{1}{d_{(i)}} B^{-1}_{km} A^T_{mi} \left(\Delta\phi\right)_i = \left(\frac{\partial \phi}{\partial x_k}\right)_P. \tag{3.12}$$

To counteract singularities during the solution of the equation system, which can be caused by distorted grids, the equation system is pre-conditioned by the reciprocal distance $\frac{1}{d_{(i)}}$ between the neighbouring cell centre and the cell centre of the considered control volume. This technique is known as the weighted Least-Square based approach.

In the present method both gradient approximation techniques are available. By default the approximation based on the Gauss-Green theorem is used for the pressure-correction equation and the Least-Square based method for all other transport equations. When the Gauss-Green theorem is used, the gradient is initialised using the Least-Square based method.

3.3 Discretisation of Transport Equations

The generic transport equation in its weak integral form reads

$$\underbrace{\int_{\mathsf{V}} \frac{\partial(\rho\phi)}{\partial t}\, dV}_{\text{temporal derivative}} + \underbrace{\oint_{O(\mathsf{V})} (\phi \rho u_i)\, dF_i}_{\text{convection term}} - \underbrace{\oint_{O(\mathsf{V})} \left(\rho \Gamma_\phi \frac{\partial \phi}{\partial x_i}\right) dF_i}_{\text{diffusion term}} = \underbrace{\int_{\mathsf{V}} f\, dV}_{\text{source term}} \tag{3.13}$$

and is a second-order equation because it includes the second derivative of ϕ in space. Accurate solutions of this equation can therefore be computed using an second-order or higher-order discretisation. Second-order accurate discretisations for each component will be discussed in the following sections.

Convection Term

The convective term is of parabolic nature. The quantity ϕ is transported in the direction of the flow. Therefore, a physically consistent approximation should rely only on neighbours, which are located up-stream. Within unstructured grids only the direct neighbour cells for a face are known by the data structure. However, a physically consistent approximation that can only use one neighbour value, results in a first-order accurate approximation. The error introduced due to the missing second-order term acts like an additional diffusive flux [88] and is known as numerical diffusion. It can be of the same magnitude as the solution itself. The second-order accurate interpolation practice introduced in Chapter 3.2 avoids numerical diffusion. However, it is not physically consistent and not bounded causing oscillations within the solution [80, 138]. Furthermore, it deteri-

orates the condition of the coefficient matrix, because the diagonal contribution is usually weakened.

Due to the difficulties described above and the importance of the convective flux in high Reynolds number flows, the discretisation of the convective term has been an area of intense research within the past decades [39, 71, 80, 90, 105, 106, 107, 110, 131, 138, 141, 156, 176, 185, 196]. In the following the convection schemes that have been used within this work will be briefly presented.

The convective flux over a face is approximated using the mid-point rule (3.1) and reads

$$\int_F (\phi \rho u_i)\, dF_i \approx [\rho \Delta F_i u_i]_F\, \phi_F = \dot{m} \phi_F \tag{3.14}$$

where $\dot{m}$ is the mass flux over the face. Details on the evaluation of the mass flux will be given in Section 3.5. The upwind discretisation scheme (UDS) of the convective flux is given by

$$\phi_{F_{UDS}} = \begin{cases} \phi_L & , \dot{m} \geq 0 \\ \phi_R & , \dot{m} < 0 \end{cases} \tag{3.15}$$

under the assumption that flow from the left hand cell of the face, denoted by the lower index $_L$, to the right hand cell, denoted by the lower index $_R$, leads to a positive mass flux $\dot{m}$ at the face. It is treated completely implicit. All higher order differencing schemes are considered by the deferred correction technique. Using linear interpolation according to Equation (3.2) yields the central differencing scheme (CDS). The linear combination of UDS and CDS

$$\phi_{F_{FB}} = \phi_{F,UDS} + \alpha(\phi_{F,CDS} - \phi_{F,UDS}) \tag{3.16}$$

is known as flux blending scheme. Upwind, central and blended differencing schemes feature a compact interpolation stencil. For the remainder of this section cells will be denoted as located down-stream $(\cdot)_D$, up-stream $(\cdot)_U$ and far up-stream $(\cdot)_{UU}$ from the face of interest.

A number of higher order interpolation practices, that intent to increase the order by an additional point upstream shown in Figure 3.7, are published [105, 196]. As they all share the same principle an unified formulation can be constructed which reads

$$\phi_F \approx \phi_U + \frac{1}{4}\left[(1+\kappa)\,(\phi_D - \phi_U) + (1-\kappa)\,(\phi_U - \phi_{UU})\right] \tag{3.17}$$

where the parameter κ controls which convection scheme is mimicked. It shall be noted that the far up-stream cell value ϕ_{UU} is not known in the data structure of an unstructured

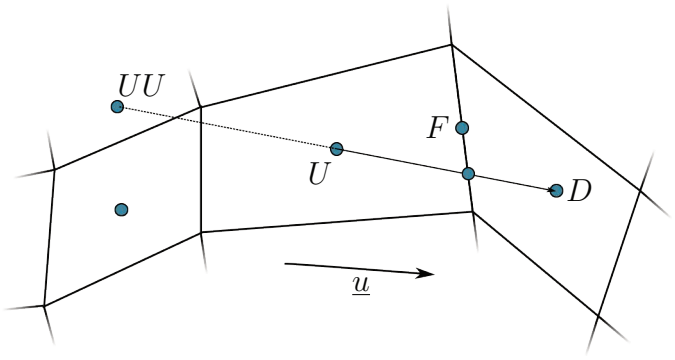


Figure 3.7: Reconstruction of virtual point UU up-stream from cell centre U.

grid framework and hence needs to be reconstructed. The higher-order correction terms tend to predict values at the face which are not bounded by the neighbouring cell values. This leads to new extrema at the faces and causes oscillations within the solution. Different methods are published which account for this problem. A well known approach by Harten [71], requires that limited schemes neither should produce new extrema nor amplify existing extrema. TVD (Total Variation Diminishing) schemes fulfil these requirements. In a TVD scheme the total variation, which is defined as sum of the absolute differences of adjacent cell values, is reduced for every new time step. The unified formulation can be extended by limiters, which locally decrease the order of accuracy and introduce a non-linearity, thus that the TVD is fulfilled. The resulting Monotone Upstream-centred Schemes for Conservation Laws (MUSCL) from van Leer [184] reads

$$\begin{aligned}\phi_F \approx & \phi_U + \frac{1}{4}\left[(1+\kappa)\,\Psi(r)\,(\phi_D - \phi_U) + (1-\kappa)\,\Psi\left(\frac{1}{r}\right)(\phi_U - \phi_{UU})\right] \\ \approx & \phi_U + \frac{1}{2}\,(\phi_D - \phi_U)\,\zeta(r).\end{aligned} \tag{3.18}$$

The limiter $\zeta(r)$ reads

$$\zeta(r) = \left[\frac{1+\kappa}{2}\Psi(r) + \frac{1-\kappa}{2} r \Psi\left(r^{-1}\right)\right] \quad \text{with} \quad r = \frac{\phi_U - \phi_{UU}}{\phi_D - \phi_U}. \tag{3.19}$$

Table 3.1 shows the coefficients for some prominent convection schemes and its limiters. Although the TVD approach is widely accepted, schemes based on the TVD approach which are either too diffusive (e.g. MINMOD [202]) or too compressive (e.g. SUPERBEE [156]), are reported.

Table 3.1: Coefficients and limiters for unified convection.

scheme	κ	$\zeta(r)$
CDS	1	1
LUDS	-1	r
QUICK	$\frac{1}{2}$	$\frac{3+r}{4}$

Another approach to satisfy boundedness has been introduced by Leonard [107] and Gaskell and Lau [56]. It is based on the normalised variables which read

$$\tilde{\phi}_F = \frac{\phi_F - \phi_{UU}}{\phi_D - \phi_{UU}} \quad \tilde{\phi}_U = \frac{\phi_U - \phi_{UU}}{\phi_D - \phi_{UU}} \tag{3.20}$$

in the one-dimensional case. Non-physical oscillations can be avoided when ϕ_F is bounded by ϕ_D and ϕ_{UU}. In the normalised variable notation this requirement is fulfilled when

$$\begin{aligned} \tilde{\phi}_F &= \tilde{\phi}_U \quad \text{for} \quad \tilde{\phi}_U < 0 \text{ or } \tilde{\phi}_U > 1 \\ \tilde{\phi}_U \;<\; \tilde{\phi}_F \;<\; 1 &\quad \text{for} \quad 0 \;<\; \tilde{\phi}_U \;<\; 1. \end{aligned} \tag{3.21}$$

The normalised variable notion has been graphically presented by Gaskell and Lau [56] in the normalised variable diagram, which is presented in Figure 3.8. The shaded region including the line $\tilde{\phi}_F = \tilde{\phi}_U$ marks the area, where the convective boundedness criterion is satisfied. The unbounded versions of some popular convection schemes are also depicted in Figure 3.8.

An extension of both, the TVD and the NVD approach to unstructured grids has been presented by Jasak [88]. The original definition depends on the far upstream node ϕ_{UU}, which is not known in an unstructured grid framework. Moreover, it has been developed for uniform meshes, which implies that NVD based convection schemes are sensitive to local changes in mesh spacing. To overcome these shortcomings a definition of the normalised variable based on the gradient field has been introduced by Jasak [88], which reads

$$\tilde{\phi}_U = 1 - \frac{\left(\frac{\partial \phi}{\partial x_i}\right)_F d_i}{2\left(\frac{\partial \phi}{\partial x_j}\right)_U d_j} \quad \text{or} \quad \phi_{UU} = \phi_D - 2\left(\frac{\partial \phi}{\partial x_j}\right)_U d_j. \tag{3.22}$$

The gradient at the face $\left(\frac{\partial \phi}{\partial x_i}\right)_F$ is interpolated by the gradients of the adjacent cells. Therefore, the new definition just depends on values in the cells adjacent to the face and enables the use of NVD schemes on arbitrary unstructured grids. Mind that the attainable order of accuracy for ϕ_F is restricted by the gradient evaluation in Equation (3.22).

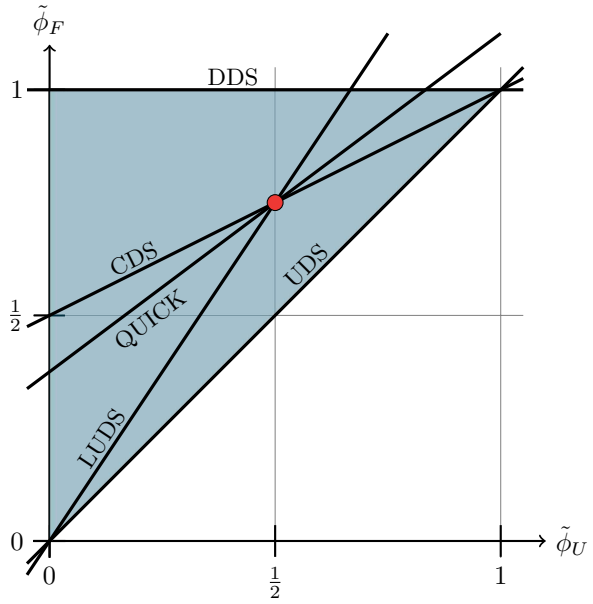


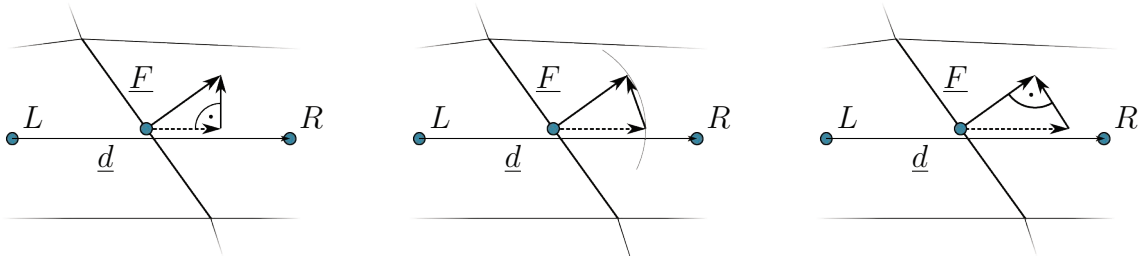
Figure 3.8: NVD diagram with some prominent convection schemes.

Diffusion Term

The evaluation of the diffusive flux

$$-\int_F \left(\rho\Gamma_\phi \frac{\partial\phi}{\partial x_i}\right) dF_i \approx -\left[\rho\Gamma_\phi \frac{\partial\phi}{\partial x_i}\right]_F \Delta F_i \tag{3.23}$$

on an orthogonal grid ($\Delta\underline{F}$ and $\underline{d}$ in Figure 3.9 are parallel) is straightforward and reads



(a) Minimal correction. (b) Orthogonal correction. (c) Over-relaxed correction.

Figure 3.9: Different evaluation modes for diffusive fluxes.

$$\frac{\partial\phi}{\partial x_i}\Delta F_i = \Delta F \frac{\phi_R - \phi_L}{\sqrt{d_i^2}}. \tag{3.24}$$

On non-orthogonal grids the face vector can be split into a orthogonal part $\Delta F_{i(O)}$ aligned with d_i and a non-orthogonal part $\Delta F_{i(N)}$ such that

$$\Delta F_i = \Delta F_{i(O)} + \Delta F_{i(N)} \tag{3.25}$$

is satisfied. Inserting the decomposed face vector into the diffusive flux evaluation yields

$$\frac{\partial \phi}{\partial x_i} \Delta F_i = \underbrace{\Delta F_{i(O)} \left(\frac{\partial \phi}{\partial x_i} \right)_F}_{\text{orthogonal part}} + \underbrace{\Delta F_{i(N)} \left(\frac{\partial \phi}{\partial x_i} \right)_F}_{\text{non-orthogonal part}} . \tag{3.26}$$

Some usual decomposition types for the face vector are depicted in Figure 3.9.

The minimal correction approach with $\Delta F_{i(O)} = (\zeta_j \Delta F_j)\zeta_i$, where $\zeta_i = \frac{d_i}{\sqrt{d_j^2}}$, minimises the contribution from the non-orthogonal part. The contribution from the orthogonal part decreases with increasing non-orthogonality.

Within the orthogonal correction approach the contribution from the orthogonal part is the same as on an orthogonal grid, irrespective of the degree of non-orthogonality. The associated orthogonal face vector reads $\Delta F_{i(O)} = \sqrt{\Delta F_j^2}\zeta_i$.

When the orthogonal face vector is evaluated as $\Delta F_{i(O)} = \left(\frac{\sqrt{\Delta F_j^2}}{\zeta_k n_k} \right) \zeta_i$, with $n_k = \frac{\Delta F_k}{\sqrt{\Delta F_j^2}}$, the contribution of the orthogonal part increases with increasing non-orthogonality. This technique is labelled as over-relaxed correction.

A unified expression for the diffusive flux, where the correction type depends on the exponent γ reads

$$\frac{\partial \phi}{\partial x_i} \Delta F_i = \Delta F_{i(O)} \left(\frac{\partial \phi}{\partial x_i} \right)_F + \Delta F_{i(N)} \left(\frac{\partial \phi}{\partial x_i} \right)_F \tag{3.27}$$

$$= \frac{\sqrt{\Delta F_j^2}}{\sqrt{d_k^2}} \left[\frac{1}{(\zeta_m n_m)^\gamma} (\phi_R - \phi_L) + \left(\frac{\partial \phi}{\partial x_i} \right)_F \left(\Delta F_i \frac{\sqrt{d_j^2}}{\sqrt{\Delta F_k^2}} - \frac{d_i}{(\zeta_m n_m)^\gamma} \right) \right] . \tag{3.28}$$

For $\gamma = -1$ the unified expression is equal to the minimal correction approach, while it resembles the orthogonal and over-relaxed correction approach for $\gamma = 0$ and $\gamma = 1$, respectively. During the solution process the orthogonal part is treated implicitly, while the non-orthogonal part is treated as a deferred correction. A large contribution to the implicit part enhances the stability of the equation system, which has been proven by Jasak [88]. Therefore, the over-relaxed correction approach improves the condition of the equation system and is used to discretise the pressure correction equation. For the momentum equation the orthogonal correction approach is chosen, because the matrix

coefficients of the pressure correction equation depend on the reciprocal matrix coefficients of the momentum equation.

Source Terms

Source terms

$$\int_{\mathsf{V}} f \, dV \tag{3.29}$$

can be general functions of ϕ. To improve the condition and stability of the equation system source terms should be treated as implicit as possible. Therefore, they have to be linearised before the discretisation with $\breve{f}$ being defined as inherently negative:

$$f(\phi) = \breve{f} + \breve{f}\phi. \tag{3.30}$$

Following Equation (3.1) the discretised form reads

$$\int_{\mathsf{V}} f \, dV = \breve{f}\Delta V + \breve{f}\Delta V \phi_P. \tag{3.31}$$

Temporal Term

Within this work only implicit approximations of the temporal derivative have been used. In contrast to explicit time integration schemes they are unconditionally stable and allow larger time steps. Within an implicit approximation the solution on the new time level n depends at least partly on the unknown variables at time level n. Therefore, an implicit approximation leads to an algebraic equation system which has to be solved.

The Implicit Euler approximation, which can be derived by a first-order truncation of a Taylor series expansion of ϕ around t^n reads

$$\int_{\mathsf{V}} \frac{\partial(\rho\phi)}{\partial t} \, dV \approx \frac{(\rho\phi\Delta V)^n - (\rho\phi\Delta V)^{n-1}}{\Delta t} \tag{3.32}$$

where the upper index denotes the time level and Δt the time step. A second-order accurate discretisation, based on three time levels and constant time steps, reads

$$\int_{\mathsf{V}} \frac{\partial(\rho\phi)}{\partial t} \, dV \approx \frac{3\,(\rho\phi\Delta V)^n - 4\,(\rho\phi\Delta V)^{n-1} + (\rho\phi\Delta V)^{n-2}}{2\Delta t} \tag{3.33}$$

for constant time steps. A formulation based on varying time steps reads

$$\int_{\mathsf{V}} \frac{\partial(\rho\phi)}{\partial t} \, dV \approx \frac{a\,(\rho\phi\Delta V)^n - (a+b)\,(\rho\phi\Delta V)^{n-1} + b\,(\rho\phi\Delta V)^{n-2}}{\Delta t^n(\Delta t^{n-1})^2 + (\Delta t^n)^2\Delta t^{n-1}} \tag{3.34}$$

with

$$a = (\Delta t^{n-1})^2 + 2\Delta t^n \Delta t^{n-1} \quad \text{and} \quad b = (\Delta t^n)^2 \tag{3.35}$$

which is especially useful when the time step is automatically adapted to certain flow features (e.g. Courant number).

3.4 Boundary Conditions

For many flow quantities the boundary behaviour may be modelled by either setting the flow quantity at the boundary to a specific value (Dirichlet condition) or setting its derivative normal to the boundary to a specific value (Neumann condition). Furthermore, both boundary conditions may be linearly combined yielding the Robin condition. For the sake of brevity the implementation of the boundary conditions into the equation system, explained in detail by Ferziger et al. [54], is omitted.

Before the boundary condition types for all flow quantities used within this work at different physical boundary types are presented in Table 3.2, details on the treatment of flow quantities at boundaries associated with specifics are given. The implementation of the boundary conditions is based on a local coordinate system (n, t, s), depicted in Figure 3.10, defined at each boundary of a cell. The decomposition of the velocity vector in

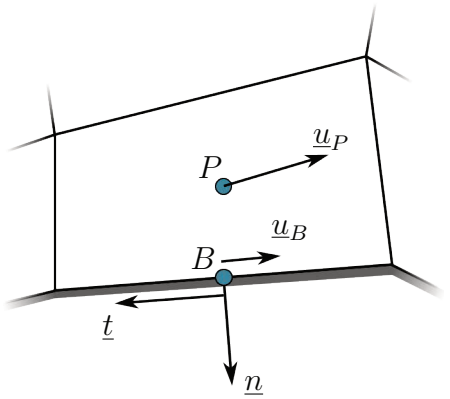


Figure 3.10: Definition of local coordinate system at boundary cells.

local boundary coordinates reads

$$u_i = u^n n_i + u^t t_i + u^s s_i. \tag{3.36}$$

The first in-plane unit vector t_i, which is the unit vector of the velocity vector projected on the boundary, reads

$$t_i = \frac{\Delta u_i - (\Delta u_j n_j)\, n_i}{\sqrt{\left[\Delta u_i - (\Delta u_j n_j)\, n_i\right]^2}} \tag{3.37}$$

with n_i being the unit normal vector of the boundary face and

$$\Delta u_i \approx u_{i,B} - u_{i,P}. \tag{3.38}$$

The second in-plane vector s_k reads

$$s_k = \epsilon_{ijk} n_i t_j. \tag{3.39}$$

In the following sections the specifics for different physical boundary conditions and flow variables are presented.

No-Slip Walls

At impermeable no-slip walls the flow adheres to the wall, hence a zero relative velocity

$$u_i \approx u_{i,wall} \rightarrow u_{i,rel} \approx u_{i,wall} - u_i = 0 \tag{3.40}$$

occurs at the wall. In the local boundary coordinate system this condition reads

$$u_i t_i = u_{i,wall} t_i \quad , \quad u_i s_i = u_{i,wall} s_i \quad \text{and} \quad u_i n_i = 0. \tag{3.41}$$

Therefore, the convective flux through the wall boundary face is zero. Instead of directly prescribing the velocities at the wall via a Dirichlet boundary condition, a cut-free technique is used to prescribe the diffusive flux, which ensures that the viscous normal stress at the wall is zero. The forces acting on the boundary face $\sigma_{i,F}$ are obtained from the flux of momentum through the face

$$\sigma_{i,F} = \int_{wall} \left[p \Delta F_i - \tau_{ij} \Delta F_j \right]. \tag{3.42}$$

The evaluation of τ_{ij} in the local boundary coordinate system is based on the decomposition given in Equation (3.36) and

$$\frac{\partial}{\partial x_i} = s_i \frac{\partial}{\partial s} + t_i \frac{\partial}{\partial t} + n_i \frac{\partial}{\partial n}. \tag{3.43}$$

Based on the hypothesis of a plane Couette flow, the changes of the local base coordinates (t_i, s_i, n_i) are omitted. Therefore, curvature effects are neglected, which implies that strongly curved surfaces rely on a fine discretisation to minimise the discretisation error. In such situations the velocity vector has solely one component $u_{i,rel} = u^t_{rel} t_i$ (cf. Eqn. (3.41)) and the gradient operator is approximated by $\frac{\partial}{\partial x_i} = n_i \frac{\partial}{\partial n}$. Hence, the components of the viscous stress tensor in the local boundary coordinate system reads

$$\tau_{ij} = \left[\mu_B \frac{\partial u^t_{rel}}{\partial n}\right](n_i t_j + t_i n_j) \approx -\left[\mu_B \frac{u_{k,rel} t_k}{y}\right](n_i t_j + t_i n_j) \tag{3.44}$$

with y being the normal distance from the boundary to the cell centre. The discrete form of the wall force expressed in the local wall reference system reads

$$\sigma_{i,F} = -\left[p\delta_{in} - \tau_{in}\right]\Delta F_n = p\delta_{in}\Delta F_n - \tau_{in}\Delta F_n \quad \text{with} \quad i \in n, t, s. \tag{3.45}$$

Inserting the viscous stress tensor leads to the following expression

$$\tau_{ij}\Delta F_j = \tau_{ij} n_j \Delta F \approx -\left[\frac{\Delta F \mu_B}{y}\right](u_{k,rel} t_k)\, t_i, \tag{3.46}$$

because $n_i t_i = 0$ and $n_j n_j = 1$.

In turbulent flows the determination of the viscosity at the wall μ_b depends on the employed wall model. If the boundary layer is fully resolved the equations are integrated up to the wall. This requires a fine grid resolution in the normal direction. A measure of the necessary grid resolution is introduced through the non-dimensional wall distance y^+ which is defined as

$$y^+ = y\frac{u_\tau}{\nu} \quad \text{with} \quad u_\tau = \sqrt{\frac{\tau_w}{\rho}} \tag{3.47}$$

with τ_w being the wall shear stress. To accurately resolve the boundary layer y^+ for the first cell layer adjacent to the wall should be smaller than one. Furthermore, a sufficient number of cells (e.g. 30 cells with a stretching factor of 1.2) needs to be placed within the boundary layer. The viscosity at the wall is defined as the sum of the material viscosity μ and the turbulent eddy-viscosity μ_t:

$$\mu_B = \mu_{eff} = \mu + \mu_t \tag{3.48}$$

In principle a Dirichlet boundary condition could be used for the turbulent kinetic energy k, however a Neumann boundary condition is applied

$$k_B \sim y^2 \rightarrow \lim_{y \to 0} \frac{\partial k}{\partial n} = 0 \quad \text{assuming} \quad k_B \approx k_P \tag{3.49}$$

to ensure consistency between boundary conditions for fully resolved boundary layers, and modelled boundary layers. The dissipation rate ϵ is fixed at the first cell layer above the boundary using an asymptotic expansion, i.e.

$$\epsilon_P \approx 2\nu \frac{k_P}{y_P^2}. \tag{3.50}$$

The definition of the specification dissipation rate ω

$$\omega = \frac{\epsilon}{C_\mu k} \tag{3.51}$$

has a singularity at the wall boundary. Therefore, the dissipation rate is set to

$$\omega_P = 80 \frac{\nu}{y_P^2} \tag{3.52}$$

at the first cell layer, based on its analytical near-wall solution in absence of convection production and turbulent diffusion.

An alternative approach is the usage of wall functions. The velocity distribution in the boundary layer is modelled according to the law of the wall. Based on the law of the wall, depicted by Figure 3.11, the non-dimensional velocity u^+ in the logarithmic region

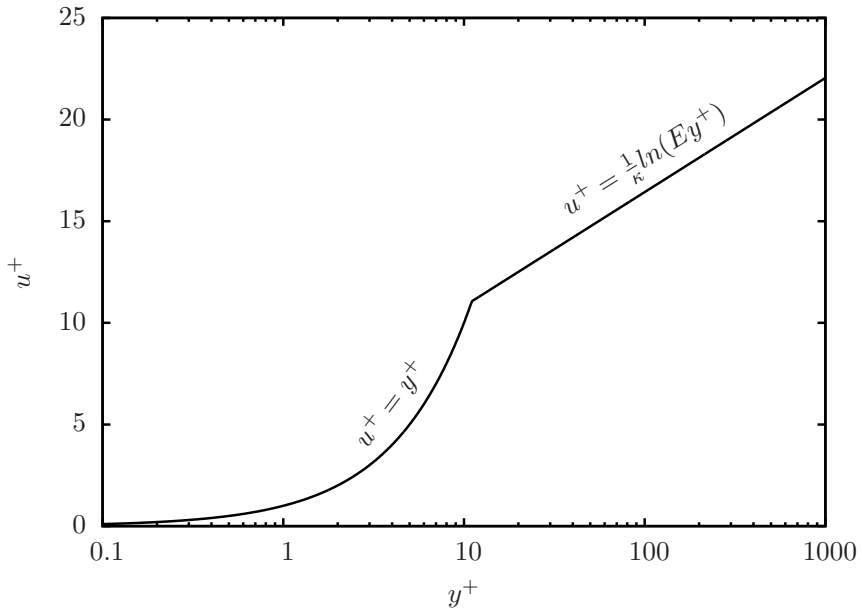


Figure 3.11: Law of the wall.

($y^+ > 20$) can be prescribed by

$$u^+ = -\frac{u_i t_i}{u_\tau} = \frac{1}{\kappa} ln\left(Ey^+\right) \quad \text{with} \quad \kappa = 0.41 \quad \text{and} \quad E = 8.432. \tag{3.53}$$

The viscosity in the cell centre P near the wall can be expressed using the following relation

$$\frac{\tau_w}{\rho} = u_\tau^2 = \nu_P \frac{u}{y} \rightarrow \mu_P = \rho u_\tau \frac{u_\tau}{u} y = \frac{\rho \kappa \sqrt[4]{C_\mu}\sqrt{k}}{ln(Ey^+)} y_P \tag{3.54}$$

which is based on the definition of u_τ obtained from the law of the wall (3.53)

$$u_\tau = -\frac{u_i t_i \kappa}{ln\left(Ey^+\right)} \tag{3.55}$$

and the turbulent kinetic energy

$$u_\tau = \sqrt[4]{C_\mu}\sqrt{k} \tag{3.56}$$

to obtain a close coupling between the momentum and turbulence equations. While a standard Neumann boundary condition type for the turbulent kinetic energy k

$$\frac{\partial k}{\partial n} = 0 \tag{3.57}$$

can be applied in line with Equation (3.56), the associated production term P is manipulated at the first cell layer above the wall. The manipulated production term follows from $P = \frac{\mu_\tau^3}{\kappa y}$ by means of

$$P = \frac{\tau_w}{\rho}\left(\frac{\sqrt{k}\sqrt[4]{c_\mu}}{\kappa y}\right). \tag{3.58}$$

The dissipation rate is fixed in the first cell layer to the local equilibrium expression $\epsilon \equiv P$ viz.

$$\epsilon_P = \frac{C_\mu^{\frac{3}{4}} k_P^{\frac{3}{2}}}{\kappa y_P} \tag{3.59}$$

and the specific dissipation rate in the first cell layer reads

$$\omega_P = \frac{\sqrt{k_P}}{\sqrt[4]{C_\mu}\kappa y_P}. \tag{3.60}$$

More details about the derivation of the boundary conditions for the turbulence quantities are given by Rung [159].

Symmetry Planes and Slip-Walls

At symmetry planes and slip walls the normal velocity component is zero under the assumption of zero curvature. However, slightly above the wall the normal velocity component may be non-zero. The flow does not adhere to symmetry planes or slip walls, accordingly the in-plane gradients are zero. Hence, the near wall velocity vector and gradient operator read

$$u_i = u^n n_i + u^t t_i + u^s s_i \quad \text{and} \quad \frac{\partial}{\partial x_i} = n_i \frac{\partial}{\partial n} \tag{3.61}$$

in the local boundary coordinate system. Inserting the viscous stress tensor (2.18) based on these definitions into the discrete form of the wall force (3.45) yields

$$\tau_{ij} \Delta F_j = \tau_{ij} n_j \Delta F \approx - \left[\frac{\Delta F \mu_B}{y} \right] (u_{k,rel} n_k)\, n_i \tag{3.62}$$

for the viscous part of the wall force.

Boundary Conditions for Pressure-Correction

For all boundaries where the mass flux over the face is prescribed a Neumann boundary condition $\frac{\partial p'}{\partial n} = 0$ for the pressure correction is applied. A non-zero derivative of the pressure correction would imply a change of the mass flux, which would deteriorate the prescribed mass flux. On the other hand a Dirichlet boundary condition for the pressure correction equation, in general $p' = 0$, leads to a correction of the velocity and mass flux at the boundary face. The evaluation of the velocity and mass flux correction will be presented in Section 3.5.

3.5 Derivation of Pressure-Correction Scheme

After the momentum equations are solved to estimate the new velocities at the control volume centres, the continuity equation is left to determine the pressure. However, the pressure does not occur in the continuity equation. This problem can be solved using the SIMPLE algorithm by Patankar et al. [139], which will be briefly outlined in the following.

Having determined the velocities from an estimated pressure by the momentum equations, new mass fluxes at the faces can be evaluated as

$$\dot{m}_F = \rho_F u_{i,F} \Delta F_i = \rho_F u_{i,F} n_i \Delta F. \tag{3.63}$$

Table 3.2: Boundary condition types for flow quantities at physical boundaries.

	Inlet	Pressure Outlet	Symmetry Plane	Slip Wall	No-Slip Wall
u_i	D	N	*	*	*
p'	N	D	N	N	N
k	D	N	N	N	N(*)
ϵ	D	N	N	N	*
ω	D	N	N	N	*
c	D	N	N	N	N

N refers to Neuman, D to Dirichlet.

** needs special attention and has been discussed in Section 3.4*

Mind that a linear 1D interpolation scheme according to Equation (3.2) is used to compute $u_{i,F}$ from cell centred velocities. In case of multi-phase flows, where the phases have different densities ρ_F is determined using the interpolation schemes dedicated to the transport of the individual phases. Due to the continuity equation the sum of all mass fluxes of a control volume needs to vanish

$$\int_V \rho \frac{\partial u_i}{\partial x_i}\, dV \rightarrow \sum_{F(V)} \dot{m}_F = 0, \tag{3.64}$$

which is usually not the case. To counteract the mass defect $\Delta\dot{m}_V = \sum_{F(V)} \dot{m}_F \neq 0$, the mass fluxes at the faces have to be modified such that

$$\Delta\dot{m}_V + \sum_{F(V)} \dot{m}'_F = 0. \tag{3.65}$$

The modification of the mass fluxes $\dot{m}'_F$ shall be pressure driven. A relationship between the pressure driven change of the face mass fluxes and the gradient of the pressure over the face ($\left(\frac{\partial p}{\partial n_i}\right)_F$) is introduced via the pressure correction scheme. It can be derived from the discrete momentum equation of a control volume, i.e.

$$\begin{aligned} A_P^{u_i} u_i &= -\left(\frac{\partial p}{\partial x_i}\right)_P \Delta V_P \underbrace{- \sum_{NB} A_{NB}^{u_i} u_{i,NB} + f_{i,P}\Delta V_P}_{\text{this part is neglected}} \Rightarrow \\ u_{i,F} n_i &\approx -\left(\frac{\Delta V}{A_P^{u_i}}\right)_F \left(\frac{\partial p}{\partial x_i} n_i\right)_F \end{aligned} \tag{3.66}$$

where $A_P^{u_i}$ refers to the central coefficient, $A_{NB}^{u_i}$ to the neighbour coefficients and f_i to the source term of the momentum equation. The correction is applied locally for each face, where the sum of the local face corrections for one cell needs to assure a negligible mass defect for the cell. Each internal face has two adjacent cells, therefore a coupled equation system for the pressure correction p' is derived

$$\sum_{CV} \left[A_P^{p'} p'_P + \sum_{NB} A_{NB}^{p'} p'_{NB} = -\Delta \dot{m}_V \right]. \tag{3.67}$$

The equation system coefficients can be derived considering the desired mass flux correction $\dot{m}'_F$

$$\begin{aligned} \dot{m}'_F =& \rho_F \Delta F u'_i n_i \\ \approx& -\rho_F \Delta F \left(\frac{\Delta V}{A_P^{u'_i}} \right)_F \left(\frac{\partial p'}{\partial x_i} n_i \right)_F , \end{aligned} \tag{3.68}$$

where the artificial volume at the face $(\Delta V)_F$ is approximated as

$$(\Delta V)_F = d_i \Delta F i. \tag{3.69}$$

The approximation of the pressure correction gradient at the face $\left(\frac{\partial p'}{\partial x_i}\right)_F$ is done according to the discretisation practice of the diffusive fluxes. Therefore, the term is split into an orthogonal (implicit) and a non-orthogonal (explicit) part. However, the explicit part depends on the gradient of the pressure correction field, which is zero because the initial pressure correction is zero. Therefore, the solution of the pressure correction equation usually exists of at least two stages. Within the first stage only the implicit contribution is considered. Sorting the equation according to central and neighbouring coefficients yields the following equation system:

$$\sum_{CV} \left[\underbrace{\left(-\frac{\rho \Delta F^2}{A_p^{u_i}} \right)_F}_{A_P^{p'}} p'_P + \sum_{NB} \underbrace{\left(\frac{\rho \Delta F^2}{A_p^{u_i}} \right)_F}_{A_{NB}^{p'}} p'_{NB} \right] = -\Delta \dot{m}_V \tag{3.70}$$

Based on the obtained pressure correction field p' the mass fluxes at the faces are corrected via

$$\dot{m}_{F,1} = \dot{m}_F + A_{NB}^{p'} \left(p'_R - p'_L \right). \tag{3.71}$$

A supplementary correction of the velocities at the cell centres reads

$$u_{i,1} = u_i + \left(-\frac{\Delta V}{A_P^{u_i}} \right)_P \left(\frac{\partial p'}{\partial x_i} \right)_P . \tag{3.72}$$

The pressure field is corrected via

$$p_{P,1} = p_P + \alpha p'_P, \tag{3.73}$$

where $\alpha \in [0, 1]$ denotes a factor by which the pressure correction is scaled. This factor replaces the usual under-relaxation of the equation system, which would lead to erroneous non-zero mass defects.

During subsequent stages non-orthogonality effects are taken into account. The obtained pressure correction field from the first stage is used to explicitly calculate the pressure correction gradient field. Based on this field the explicit contributions from the gradient approximation at the face are considered on the right hand side of the equation system. Because the mass defect has been eliminated within the first stage, it no longer contributes to the right hand side of the equation system. The coefficient matrix for the second stage is identical to the first stage. The resulting second pressure corrections are used to update the pressure field. The velocity field and the mass fluxes remain however unchanged.

Rhie and Chow Interpolation

The combination of discretised first derivatives, which are included in the gradient of the pressure (3.66) and divergence of the velocities (3.64), tend to produce oscillations when a collocated variable arrangement at the cell centres is used. These oscillations are known as chequerboard pattern. Considering an oscillating solution shown in Figure 3.12, a straight-

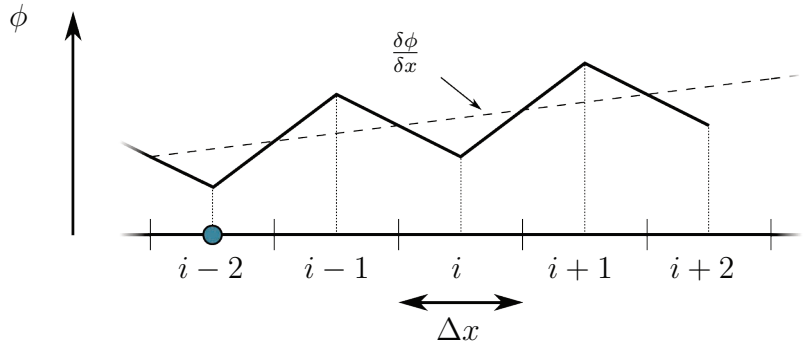


Figure 3.12: Chequerboard pattern of ϕ on collocated grid (solid line) and gradient of ϕ based on $2\Delta x$ approximation (dashed line) .

forward approximation based upon central differences would not be able to detect such

oscillations. Following an approach suggested by Rhie and Chow [151], the interpolated velocity at each face is augmented by a derivative term, i.e.

$$u_{i,F} = \underline{u}_{i,F} + (d_j \Delta F_j)_F \underline{\left(\frac{1}{A_P^{u_i}}\right)_F} \left[\left(\frac{\partial p}{\partial x_i}\right)_F - \underline{\left(\frac{\partial p}{\partial x_i}\right)_F} \right]. \tag{3.74}$$

The underlined terms are linearly interpolated from the adjacent cell centres, while $\left(\frac{\partial p}{\partial x_i}\right)_F$ is a CDS approximation based on the adjacent cell centres of the face. Peraire et al. [140] have shown that the proposed procedure can be interpreted as a fourth-order artificial damping term applied to the pressure correction equation. Therefore, the associated error is reduced quadratically when the grid is refined.

3.6 Numerical Procedure

The algebraic equation system for each transport equation is obtained from the discretised transport equation, written in terms of unknowns at the cell centres. A compact discretisation stencil for cell P, contains the contribution from the cell itself and its direct neighbours NB:

$$A_P \phi_P + \sum_{NB(P)} A_{NB} \phi_{NB} = S_P \tag{3.75}$$

Contributions that depend on unknowns in other cells are considered through the deferred correction approach in the source term S_P. The algebraic equation system for all cells n reads

$$A_{ii}\phi_i + \sum_{j \in NB(i)} A_{ij}\phi_j = \sum_j A_{ij}\phi_j = S_i, \quad i = 1, \ldots, n, \tag{3.76}$$

and yields a sparse coefficient matrix A_{ij}. On unstructured grids the non-zero pattern of the coefficient matrix is irregular.

The solution procedure for an unsteady flow problem is outlined in Algorithm 1. First all flow variables including the mass fluxes at the faces are initialised. The old values for all flow variables are initially set to the current values. Then the segregated solution algorithm is employed to iteratively solve the coupled equation system, in so called outer iterations for each time step. Within an outer iteration first the Picard-linearised momentum equations are solved to obtain new velocities at the cell centres. The predicted velocities do not necessarily satisfy the continuity equation. Therefore, the first stage of the pressure correction equation is solved and subsequently the mass fluxes, velocities

```
initialise flow variables and mass fluxes
repeat
    repeat
        solve momentum equation
        solve first stage of pressure correction equation
        correct fluxes, velocities and pressures
        repeat
            solve n'th stage of pressure correction equation
            correct pressures
        until stage = nMaxStages
        optionally solve equations for turbulence
        optionally solve equation for concentration
        optionally solve equations for other fields
    until (nOuterIteration≥maxOuterIteration) or (residual < residualThreshold)
    old flow variables=current flow variables
until nTimeSteps≥maxTimeSteps
```

Algorithm 1: Pseudo-code illustrating the basic segregated solution procedure of the RANS solver.

and pressures are corrected, to obtain a solution without mass defects. In the following pressure correction loops the n'th stage of the pressure correction equation is solved to account for non-orthogonality effects in the grid. The solution is used to update the pressures at the cell centres. The mass-fluxes and velocities are left unchanged because they already satisfy the continuity equation. Based on the corrected mass fluxes, all additional equations (e.g. turbulence, concentration) are solved afterwards. When a convergence criterion for the coupled system of equations is satisfied the outer loop is finished. The old values of the flow variable are set to the current values before the algorithm proceeds to the next time step.

Parallelisation

The algorithm is parallelised using a domain-decomposition technique based on a Single Program Multiple Data message-passing model. Inter-processor communication employs the MPI communications protocol [120]. Load balancing is achieved using the ParMETIS partitioning software [98]. The prescribed discretisation approach features a compact computational stencil, e.g. all terms in a cell depend on values of the cell itself and its direct neighbours. Contributions from other than direct neighbours are incorporated through the gradients of the cell itself and its direct neighbours. During the assembly stage the information of all direct neighbours have to be available. Therefore, ghost cells are introduced at the partition boundaries, where the direct neighbours are located on another process as depicted in Figure 3.13. Ghost cells just serve as source for the

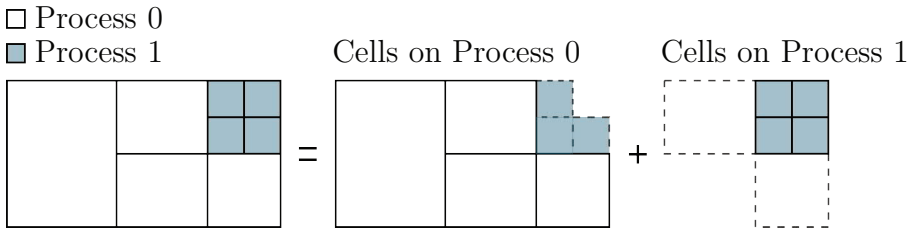


Figure 3.13: Illustration of domain decomposition with 2 processes. Cells with dashed grey lines are ghost cells.

evaluation of terms in the adjacent cells, terms are never evaluated at the ghost cells. Hence, no information other than the values of the dependant variables, its gradients, the cell volume and the barycentre of the cell have to be present at ghost-cells. They receive their values through parallel exchange routines, which are built upon the knowledge of the global cell id's and the association of the cells to the processes. During the solution stage all issues associated to the parallel solution of equation systems are handled by the PETSc toolkit [13, 14, 15] , which was provided with the global cell id's and the cell adjacencies.

Solution of Linear Equation System

Iterative Krylov subspace methods, provided by the PETSc toolkit [13, 14, 15], are used to solve the linearised equation systems. The solution process consists of a preconditioning and a subsequent iterative solution of the equation system in so-called inner iterations.

To ensure convergence of the linear equation system it has to be diagonal dominant. The condition for diagonal dominance reads

$$|A_p| \geq \sum_{NB(P)} |A_{NB}|. \tag{3.77}$$

Furthermore, the magnitude of the central coefficient is required to be larger than the sum of the magnitudes of the neighbouring coefficients for at least one cell.

The convergence criterion for the coupled system of equations is based on the algebraic integral residual $\overline{R}$ for each transport equation considered in the solution process. It is usually calculated by the L_1-norm of the local residuals R_i of the algebraic equations system and normalised by the residual of the first outer iteration $\overline{R}^1$. It reads

$$\overline{R}^k = \left| \frac{1}{\overline{R}^1} \sum_{i=1}^{n} R_i^k \right| = \left| \frac{1}{\overline{R}^1} \sum_{i=1}^{n} \left[A_{ii}^k \phi_i^{k-1} + \sum_{j \in NB(i)} A_{ij}^k \phi_j^{k-1} - S_i^k \right] \right| \tag{3.78}$$

using the matrix coefficients of the current outer iteration k and the field values of the previous outer iteration $k-1$. Mind that the above expression is only valid for $k > 1$. Other norms and normalisation techniques may be chosen alternatively. Typically the convergence criterion depends on the largest residual of all solved transport equations.

Under-Relaxation

While the diagonal dominance ensures the convergence of the linear equation system, it does not guarantee that the coupled equation system converges during the outer iterations. To promote the convergence of the coupled equation system, the change of the unknowns through the outer iterations is slowed down using an under-relaxation. The new value of the unknown is written as a mixture of old value and predicted value from the solution of the linear equation system

$$\phi^k = \phi^{k-1} + \alpha_\phi \left(\phi^* - \phi^{k-1} \right), \tag{3.79}$$

where the upper index k denotes the outer iteration, and ϕ^* is the predicted value from the linear equation system

$$\phi^* = \frac{S - \sum_{NB(P)} A_{NB} \phi_{NB}^k}{A_P}. \tag{3.80}$$

The under-relaxation factor $\alpha \in [0 : 1]$ defines how much of the new solution is used. According to Patankar [138] an implicit formulation of the under-relaxation technique can be derived by substituting Equation (3.80) in Equation (3.79) and rearranging it

$$\underbrace{\frac{A_P}{\alpha}}_{A_P^*} \phi_P^k + \sum_{NB(P)} A_{NB} \phi_{NB}^k = \underbrace{S + \frac{1-\alpha}{\alpha} A_P \phi^{k-1}}_{S^*} . \tag{3.81}$$

Thus the implicit under-relaxation also increases the diagonal dominance, because the central coefficient grows when α is chosen smaller than unity and the off-diagonal coefficients remain unchanged. The implicit under-relaxation technique is applied for all transport equations except the pressure-correction equation where the relaxation is directly applied to the update of the pressure as explained in Section 3.5.

Typical values for the under-relaxation factors in three-dimensional simulations on unstructured grids are $\alpha_u = 0.4, \ldots, 0.7$ for the momentum and turbulence equations, $\alpha_c = 0.7, \ldots, 0.9$ for the concentration transport and $\alpha_p = 0.1, \ldots, 0.3$ for the pressure-correction equation. Note that a theoretical investigation of Ferziger and Peric [54] recommends to use $\alpha_p + \alpha_u = 1.0$.

4 Error Classification

Any numerical solution scheme is afflicted with different types of errors that may deteriorate the accuracy of the method. The dynamic grid adaptation technique aims to improve the accuracy of the results of the numerical procedure by local grid refinement. To understand to which degree local grid refinement can improve the accuracy, a sound knowledge of possible sources of errors is necessary. In the following errors will be classified into different sources and analysed. Furthermore, methods to counteract the individual error types will be discussed.

4.1 Modelling Error

The *modelling error* is the difference between the real flow and the exact solution of the mathematical model [54]. The mathematical model used to describe the fluid flow within this work is based on the Navier-Stokes equations. The 'simple' direct numerical simulation of the Navier-Stokes equations for turbulent flows at high Reynolds numbers is impossible, due to the enormous numerical effort arising from the large spectrum of scales. Therefore, turbulence is modelled through simpler statistical approaches. Furthermore, the simulation of free-surface flows is based on a simple mixture-fraction model, which treats the flow as homogeneous even at the free-surface and deliberately ignores surface tension and any diffusion or reaction terms. Additional errors are introduced by the boundary conditions, which are either not exactly known (e.g. inflow/outflow) or simplified to a certain flow model (e.g. walls). Moreover, a finite computational domain is used which is only a cut-out of the real world.

The analysis of the *modelling error* is a challenging task, because it requires the knowledge of the exact flow. The determination of the real flow can either be based on simulations using Direct Numerical Simulations (only suitable for simple problems) or mea-

surements from experiments, where both of which are usually afflicted with uncertainties. Even when the real flow is known, other sources of errors in the simulations need to be negligible.

Modelling errors can be reduced using a more sophisticated model that is a better approximation of the real flow. The development of a sophisticated model requires a deep knowledge of the underlying physics. Usually a more sophisticated model is associated with an increase in the overall computational effort.

The choice of the mathematical model strongly depends on the purpose of the numerical method. It should capture all relevant physics for the specific purpose of the investigation, but neglect computationally expensive models that merely effect this purpose.

4.2 Discretisation Error

When it is impossible to solve the mathematical model exactly, which is usually the case, it needs to be discretised to provide an algebraic equation system that can be solved numerically. The difference between the exact solution of the mathematical model and the exact solution of algebraic equation system is the *discretisation error*. The discretisation error depends on the discretisation method for both the equations and the solution domain. In the following the discretisation errors associated with the presented method will be discussed.

The quality of the discretisation is judged by its order of accuracy, which relates the truncation error of the approximation to some power of a characteristic size d (e.g. grid spacing for spatial approximation). For a p^{th} order method the truncation error is proportional to d^p. The order defines the rate of change in the error due to a change in the characteristic size. It is not a direct measure for the magnitude of the error. The present approach generally assumes linear variation in space and time

$$\phi(x_i) = \phi_P + (x_i - x_{i,P}) \left(\frac{\partial \phi}{\partial x_i} \right)_P \quad \text{with} \quad \phi^P = \phi(x_{i,P}) \tag{4.1}$$

$$\phi(t + \Delta t) = \phi^t + \Delta t \left(\frac{\partial \phi}{\partial t} \right)^t \quad \text{with} \quad \phi^t = \phi(t). \tag{4.2}$$

A comparison with the Taylor series expansion in space of a function around $x_{i,P}$

$$\begin{aligned}\phi(x_i) =& \phi_P + (x_i - x_{i,P}) \left(\frac{\partial \phi}{\partial x_i}\right)_P \\ &+ \frac{1}{2}(x_i - x_{i,P})(x_j - x_{j,P}) \left(\frac{\partial \phi}{\partial x_j}\frac{\partial \phi}{\partial x_i}\right)_P \\ &+ \mathcal{O}(3)\end{aligned} \tag{4.3}$$

proves second-order accuracy for the assumed linear variation, because the truncation error scales with $|(x_i - x_{i,P})(x_j - x_{j,P})|$. Second-order accuracy for the temporal term is proven by a similar analysis. The theoretical second-order accuracy of the method can be proven through grid refinement studies. The error for each quantity should decrease by a factor of four, when the grid spacing is halved.

The following discussion of discretisation error is based on the findings of Jasak [88]. Discretisation errors can be divided in two groups. The first group contains errors due discretisation practices less accurate than second-order. In case of the convective flux discretisation usually higher order differencing schemes are used, which are limited to ensure boundedness of the solution. Due to the limiters the order is locally decreased as discussed in Section 3.3. Any higher order convection scheme can be written as the sum of a central differencing scheme, a diffusion term with an associated numerical diffusion tensor and a source term, which inheres contributions from approximations based on other cells than the direct neighbours of the face in question [88]. To analyse the numerical diffusion tensor the difference of an interpolation to the face $(\delta\phi)_F$ based on the second-order accurate central differencing scheme and the flux blending scheme is examined.

$$\begin{aligned}(\delta\phi)_F &= \phi_{F,CDS} - \phi_{F,FB} \\ &= \phi_{F,CDS} - (1-\alpha)\phi_{F,UDS} - \alpha\phi_{F,CDS} \\ &= (1-\alpha)(\phi_{F,CDS} - \phi_{F,UDS}) \\ &= (1-\alpha)[(\lambda - max(sgn(\dot{m}_F), 0))\phi_P + \\ &\quad ((1-\lambda) - min(sgn(\dot{m}_F), 0))\phi_N]\end{aligned} \tag{4.4}$$

Assuming $\dot{m}_F > 0$ the expression simplifies to

$$(\delta\phi)_F = (1-\alpha)(1-\lambda)(\phi_N - \phi_P) = (1-\alpha)(1-\lambda)d_i \left(\frac{\partial \phi}{\partial x_i}\right)_F . \tag{4.5}$$

Summing up the differences on all faces of a cell yields

$$\begin{aligned} E_P = \sum_{F(P)} \dot{m}_F (\delta\phi)_F &= \sum_{F(P)} \left[\rho \Delta F_i u_i\right]_F (1-\alpha)(1-\lambda) d_j \left(\frac{\partial \phi}{\partial x_j}\right)_F \\ &= \sum_{F(P)} \Delta F_i \left[[(1-\alpha)(1-\lambda)(\rho u_i)_F d_j] \left(\frac{\partial \phi}{\partial x_j}\right)_F \right] \\ &= \sum_{F(P)} \Delta F_i \left[\tau_{ij,N} \frac{\partial \phi}{\partial x_j} \right]_F \end{aligned} \tag{4.6}$$

where $\tau_{ij,N}$ is the numerical diffusion tensor. The analysis of the diffusion term E_P and the numerical diffusion tensor

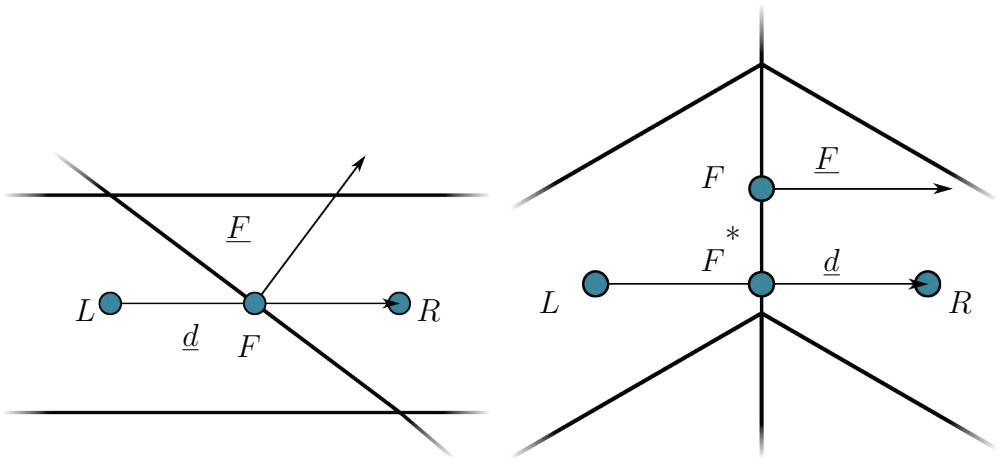
$$(1-\alpha)(1-\lambda)(\rho u_i)_F d_j \tag{4.7}$$

reveals that it tends to zero when central differencing is used ($\alpha = 1$) or the weighting factor λ from the linear interpolation scheme is close to one. This is obvious since a high interpolation factor implies a small distance of the face to the cell centre. On the other hand the interpolation factor is $1 - \lambda$ for the other cell associated to the face which implies a larger numerical diffusion. The diffusion tensor scales with the velocity u_i and the distance between two adjacent cells d_j, which indicates the grid spacing. Furthermore, the numerical diffusion vanishes for a uniform distribution of ϕ which implies $\frac{\partial \phi}{\partial x_j} = 0$ and gets large in case of steep gradients. When the flow is aligned with the mesh the contribution of all faces where the velocity is orthogonal to the face normal disappears. To effectively reduce the numerical diffusion, convection schemes close to second-order should be used. A reduction of the flow velocity is impossible because it is the solution of the flow problem. Mesh alignment is only possible when the mean flow direction is known a priori and requires attention during the mesh generation. The most general way to minimise the numerical diffusion is to minimise the grid spacing. Jasak [88] has shown a similar derivation of the numerical diffusion caused by temporal discretisation that are not at least second-order accurate. The numerical diffusion from an implicit first order Euler approximation scales with the Courant number Co. For $Co = 1$ it resembles the same order as the numerical diffusion from upwind differencing.

The second group contains errors due to the discretisation of the solution domain. This includes errors from *insufficient grid resolution*, *excentricity* and *non-orthogonality* of cells. The most obvious mesh induced error originates from an *insufficient grid resolution*. The second-order accurate method assumes a linear variation of the variables of

state over a cell. Accordingly, a sufficient number cells have to be placed in the field to obtain sufficiently accurate approximations of the shape of the solution with piecewise linear functions. Usually the rate of change of the solution is not uniform over the entire domain. Hence, an appropriate grid resolution needs to place enough cells only in regions afflicted with a large local change of the resolution. The obvious way to counteract errors originating from insufficient grid resolution is to add cells to regions where they are necessary. Unfortunately, the locations of these regions often display large temporal changes.

When the face normal vector is not parallel to the vector connecting the two adjacent cells as depicted in Figure 4.1(a) the face is afflicted with *non-orthogonality* issues. The non-orthogonality influences the discretisation of the diffusion term. If the non-



(a) Non-orthogonality. (b) Excentricity.

Figure 4.1: Explanation of non-orthogonality and excentricity.

orthogonality term in the diffusion approximation is used *non-orthogonality* does not deteriorate the order of the methods. However, the non-orthogonal term inheres terms not only from the cell and its direct neighbours but also from the neighbours of the neighbours. The larger computational molecule implies a larger leading term in the Taylor series expansion and larger errors.

When the piercing point of the line connecting the two adjacent cells with the face does not coincide with the barycentre of the face, depicted in Figure 4.1(b), the order of the

face integral using the midpoint rule

$$\int_F dF_i\phi = \mathrm{F}_i\phi_F \tag{4.8}$$

is lower than two. Summing up the error due to a face integral based on ϕ_{F^*} rather than ϕ_F for the convection term reads

$$E_E = \sum_F F_i \left(\rho u_i \delta\phi\right)_F \quad \text{with} \quad \delta\phi_F = \phi_F - \phi_{F^*} = \left(x_{i,F} - x_{i,F^*}\right)\left(\frac{\partial\phi}{\partial x_i}\right)_F . \tag{4.9}$$

It can be transformed to a volume integral

$$\begin{aligned} E_E &= \sum_F F_i \left[\left(\rho u_i\right)_F \left(x_{j,F} - x_{j,F^*}\right)\left(\frac{\partial\phi}{\partial x_j}\right)_F\right] \\ &= \frac{\partial}{\partial x_i}\left(\tau_{ij,E}\left(\frac{\partial\phi}{\partial x_j}\right)\right) \quad \text{with} \quad \tau_{ij,E} = \left(\rho u_i\right)_F \left(x_{j,F} - x_{j,F^*}\right) \end{aligned} \tag{4.10}$$

where $\tau_{ij,E}$ is a diffusion tensor due to excentricity. The diffusion tensor scales with the distance from the piercing point to the barycentre of the cell and the flow velocity. The numerical diffusion tensor $\tau_{ij,N}$ scales with the length of the connecting line between the adjacent cell centres, which is usually larger. Therefore, the influence of the diffusion caused by excentricity is deemed to be less important than the influence from numerical diffusion for large portions of the mesh. However, many automatic grid generators tend to produce grids, which are afflicted with a high excentricity in the region of sharp convex edges. In these regions the diffusion introduced by excentricity can be significant. The reduction of the order caused by excentricity can be avoided when, an explicit correction from F^* to F is applied as proposed in Equation (3.2). However, the correction term increases the numerical stencil, implying a larger leading terms in Taylor series expansions and hence larger errors. Furthermore, they can possibly violate boundedness of the solution due to the extrapolation character of the correction in Equation (3.2).

4.3 Iterative Error

The *iterative error* is the difference between a numerical, usually iterative, and exact solution of an algebraic equation system. The direct solution of linearised algebraic equation systems originating from RANS procedures for engineering problems is usually not preferred, due to the associated enormous numerical effort. Therefore, the equation systems are solved iteratively, where the iteration process has to be stopped at some point.

The stopping criterion is usually based on a prescribed reduction of the residuals of the algebraic equation system. Whatever stopping criterion is used, the obtained solution will always deviate from the exact solution at least due to *round-off errors*, that will be discussed in the next section. Lowering the iterative error to a specific value requires a significantly larger effort when the size of the equations system grows. The choice of the stopping criterion is a non-trivial task, since it needs to ensure that the remaining iterative error is negligible at least compared to the other errors, while the solution has to be computed with an appropriate effort. Ferziger et al. [54] advise to reduce the iterative error to a level at least one order lower than the discretisation error.

The reduction of the *iterative error* can be estimated by monitoring the residuals of the algebraic equation system. Investigations by Ferziger et al. [53] have shown that the norm of the residual, the norm of the difference between successive iterations and the actual iteration error are reduced at the same rate, at least for iterations after the initial stage. Therefore, a reduction of the residual by n orders implies a reduction of the iterative error by approximately n orders. Assuming that at the initial stage of a simulation all fields are initialised with zero, the error at the initial stage is the solution itself. When the residual in the simulation is reduced by n orders the *iterative error* is most likely reduced by at least $n-1$ orders. Therefore, a residual drop of five orders of accuracy implies that the solution is accurate within approximately 0.001% regarding the iterative error.

4.4 Round-off Error

The *round-off error* describes the error due to truncation of a floating point number which has an unlimited amount of digits in the mathematical description, but is reduced to a finite number of digits on a computer. In ill-conditioned systems the round off error may accumulate and get important. However, for the present method using double precision variables to store floating point numbers, the influence of the round-off error is deemed to be negligible compared to the other errors.

Concluding Remarks

Modelling errors are introduced when a mathematical model is selected to represent the real world. When a choice is made the only way to counteract these errors, is to select

another mathematical model, which has to be discretised subsequently. The influence of the *iterative error* is controlled through the selected convergence criterion. To reduce the error a stricter convergence criterion has to be used, which increases the necessary computational effort. *Round-off errors* are deemed to be negligible for the present study. *Discretisation errors* due the discretisation practice lower than second-order can be minimised selecting the appropriate discretisation practices. All grid-related errors can be decreased by a reduction of the cell size. Reducing the cell size in the whole computational domain, which is the easiest methodology, leads to huge numerical effort. On the other hand, usually only a small number of cells that are afflicted with large errors need to be refined to enhance the overall solution accuracy. Refining only these parts of the computational domain yields an efficient method. However, regions where the cell size needs to be reduced are usually not known a priori and might also vary over time. Therefore, an automatic grid adaptation procedure, that detects the regions with large errors depending on the cell size, and refines the grid in the respective regions is a powerful tool. Grid quality measures that influence the accuracy of the solution (e.g. non-orthogonality and excentricity) have to be considered within the grid refinement procedure. An automatic grid adaptation procedure that fulfils the above mentioned requirements will be presented in Chapter 6.

5 Immiscible Two-Phase Flows

Maritime two-phase flows are characterised by immiscible fluids which can be treated as incompressible. In marine engineering flows the interface between immiscible fluids always remains sharp. Therefore, the simulation of two-phase flows requires a model which is able to maintain a sharp interface. Furthermore, the method should be as general as possible to treat flows with large interface deformations and topology changes like they occur for breaking waves. In most maritime flows the effect of surface tension is negligible. The simulation of such two-phase flows has been a research topic since many decades. Some popular approaches used in the finite-volume discretisation and their strengths and weaknesses will be briefly presented. The discussion will show that none of the proposed methods is able to combine the requirements of efficiency, accuracy and generality. Techniques to improve a general model with respect to its efficiency and accuracy have been developed and will be presented subsequently.

5.1 State of the Art

The developed approaches can be classified into two major groups – *interface-tracking* methods and *interface-capturing* methods [54]. *Interface-tracking* methods explicitly define the interface which is dynamically followed by the discretised domain. On the other hand *interface-capturing* need to reconstruct the free-surface position from marked particles or indicator functions on a static or dynamic mesh.

5.1.1 Interface Tracking Methods

Interface-tracking methods follow the explicitly defined free-surface by adjusting the domain. The advancement of the free-surface is based on the Lagrangian method, while the

discretisation of the underlying flow field can be treated by the Lagrangian, Eulerian or Arbitrary Lagrangian Eulerian (ALE) method.

a) ALE

Interface tracking methods based on the ALE representation [38, 52, 69, 179] align the interface between the fluids with a domain boundary, which is deformed during simulation to satisfy the free-surface boundary conditions, namely the dynamic boundary condition ($p = p_{atmospheric}$ at the free-surface) and the kinematic boundary condition ($\dot{m}_F = 0$, no net mass flux through the faces forming the free-surface). Hence, the interface between the fluids is inherently kept sharp, and only one fluid is modelled which enhances the efficiency of the method. The inner grid has to be deformed according to the deformation of the domain boundaries. Large free-surface deformations lead to decreased mesh quality, in particular with an increased deviation of the wetted domain from the initial domain. Furthermore, topological changes in the interface require re-gridding or at least intensive grid optimizations. Re-gridding and intensive optimization are computationally expensive operations. Accordingly, these methods are limited to flows with moderate free-surface deformations.

b) Front Tracking

In the front tracking approach presented by Daly [41] the free-surface is represent by ordered and connected massless particles. The particles are advanced according to the velocity of the underlying cells. The deformation of the free-surface leads an inhomogeneous distribution of particles which in turn yields an inhomogeneous resolution of the free-surface. Therefore, particles need to be added or deleted dynamically and the connections need to be updated. This process is straightforward in two dimensions, but gets complicated in three dimensions [177]. Due to the ordered sequence of particles the treatment of merging and rupturing interface shapes is almost impossible. Furthermore, the method offers no implicit conservatism. For instance, mass conservatism can only be ensured by subsequent iterative front correction methods.

5.1.2 Interface Capturing Methods

Interface capturing methods define the free-surface position from marked particles or indicator functions defined on both sides of the interface. Accordingly, the interface position is not explicitly defined but has to be reconstructed.

a) Marker and Cell

The Marker and Cell (MAC) method presented by Harlow and Welch [69] mostly used in conjunction with finite difference discretisations uses massless particles introduced in the fluid domain at the initial time, which are convected according to the fluid velocities. The fluid flow is only solved in cells which are marked as liquid phase. The free-surface is located in cells containing markers and having neighbouring cells without markers. Examples showing that the method is capable to treat complex interface deformations and topological changes are shown by Daly [40] and Hirt et al.[82], however the computational effort to track the huge number of necessary particles is large [9]. Furthermore, the distribution of markers tends to become irregular with progressing simulation time and needs to be reinitialised. Moreover, the conservation properties are not implicitly satisfied as 'newly' marked fluid cells need to be initialised and require special consideration.

b) Volume of Fluid

The Volume-of-Fluid (VoF) method first used by Noh et al.[127] and refined by Hirt et al. [81] uses a scalar fraction function c defined at each cell. It ranges from zero to one with zero indicating the first fluid and one the second fluid. The interface is characterised by values greater than zero and lower than one. The fluid properties in interface cells are defined as a mixture from both fluids according to the value of c. Contrary to the MAC method flow equations in regions filled by both phases are solved. The accuracy of the VoF method hinges on the discretisation of the concentration transport equation, because the transport of the mixture fraction function principally allows mixing of both phases which is physically incorrect. The method is capable to predict merging and rupturing of free-surfaces, while the numerical effort is moderate. A sharp representation of the interface implies a concentration distribution according to a step function, posing challenges for the discretisation. A major advantage of the VoF method is the inherited conservatism. The VoF method will be discussed in more detail in Section 5.1.3.

c) Level Set

The signed distance function introduced by Osher et al. [132] is used to characterise the interface. It is located at the zero iso-surface of the signed distance function. The continuous distribution of the signed distance function simplifies the higher order discretisation of the scalar advection equation. Furthermore, the interface normal and curvature can be determined with higher accuracy. It has been proven by Osher [132] that the interface remains at the zero distance iso-surface, however the distance function tends to get 'irregular'. This behaviour prevents merging of interfaces, which can be overcome by reinitialising the distance function after each time step as shown by Sussmann et al. [175]. To compute the fluid properties in cells cut by the interface, it has to be reconstructed after each time to determine the volume fractions. A major drawback of the Level Set method is that it does not guarantee mass conservation. Olsson [128] presented a Level Set method using a smeared heaviside function as level set, which can be advected using conservative methods. Therefore, mass conserving properties have been improved, however the method is not strictly conservative. The reinitialisation step is modified to resharpen the surface. The resharpening method has been continuously improved by Olsson [129] and Sato [162] and the model has been extended to 2-fluid models by Štrubelj et al. [170]. An overview about level set methods is given by Sethian et al. [165].

d) Moment of Fluid

Within the Moment of Fluid (MoF) method presented by Shashkov et al. [3, 4, 49] the concentration and the local concentration centroids are utilised. This enables the MoF method to reconstruct the interface using only local cell values. The location is determined such that the volumes are preserved and the concentration centroids are best approximated. The concentration and the first moment of the concentration are transported by a Lagrangian remap scheme described in [49]. The technique is able to resolve interface details as small as a cell, and therefore requires less cells than the VoF method. The additional transport of the first moment and the iterative procedure to reconstruct the interface imply that the method is afflicted with a high computational effort.

e) Hybrid Models

A number of hybrid models combining some of the aforementioned methods to benefit from their particular strengths are published. Combined Level Set VoF (CLSVOF) methods aim to inhere the mass conservation property of the VoF scheme and the surface representation accuracy of the Level Set scheme. Examples for CLSVOF methods are presented by Sussmann et al. [174], Son [169] , Van der Pijl [183] and Park et al. [133]. CLSVOF methods are able to accurately predict the interface with an improved mass conservation behaviour compared to pure Level-Set at the cost of increased computational demands.

The combination of particle methods and Level Set methods presented by Enright et al. [51] and Aulisa et al. [8] aims on combining the mass conservation properties of particle methods with the accurate geometrical surface information obtained from Level Set methods. Furthermore, the particles can be used to enrich under-resolved reasons. The method is shown to have a better treatment of small filaments compared to the CLSVOF method. Again, the increased accuracy is accompanied by an increased numerical effort.

A combination of the Level Set method with the MoF (CLSMOF) has been proposed by Jemison et al. [91]. The method used the Level Set information to speed up the interface reconstruction of the MoF scheme, and for some special interpolations. Although slight performance improvements could be shown, the method is still afflicted with high computational costs.

5.1.3 The VoF model

For maritime free-surface flows surface tension effects are negligible. Therefore, the VoF model seems to be the best compromise with respect to efficiency and accuracy. In the following the basics of the Volume of Fluid model will be introduced, subsequently some popular discretisation practices of the VoF model will be described in more detail and shortcomings of the model will be highlighted.

Although the respective fluids are immiscible, the flow field is computationally modelled by a locally varying mixture of the two phases. The mixture state is governed by the transient evolution of a mixture fraction from its initial state. In this regard, the volumetric concentration c of the first phase is defined by

$$c = \frac{\Delta V_1}{\Delta V} = \frac{\Delta V_1}{\Delta V_2 + \Delta V_1} \,, \tag{5.1}$$

where ΔV_1 denotes the fraction of the discrete control volume ΔV filled with the first phase and ΔV_2 is the respective fraction filled by second phase. The local properties of the mixture, density ρ and molecular viscosity μ, are computed from simple algebraic equations of state. The most frequently employed linear version reads

$$\rho = c\,\rho_1 + (1-c)\,\rho_2 \qquad \text{and} \qquad \mu = c\,\mu_1 + (1-c)\,\mu_2\ . \tag{5.2}$$

Since the fluids are immiscible, the initial material-state will never change, viz.

$$\frac{Dc}{Dt} = \frac{\partial c}{\partial t} + U_j\,\frac{\partial c}{\partial x_j} = 0\ . \tag{5.3}$$

In conjunction with VoF-methods, the continuity Equation (2.6) is subdivided into two parts. Substituting the density by the equation of state (5.2), the continuity expression for incompressible phases yields

$$\left[(\rho_1 - \rho_2)\,\frac{Dc}{Dt}\right] + \rho\frac{\partial U_j}{\partial x_j} = 0\ . \tag{5.4}$$

The transient first part of (5.4) vanishes due to the immiscibility relation (5.3). The steady-state second part refers to the traditional volume conservation for incompressible flows, which is often used to compute the pressure from a pressure-correction or pressure-projection scheme [36, 54]. In conclusion, (5.2) and (5.3) are the only multi-phase related modifications of the computational procedure.

Using a second-order accurate, cell-centred finite-volume approach, a modified version of (5.3) for divergence-free velocity fields, i.e.

$$\int_{\Delta V}\left(\frac{\partial c}{\partial t} + \frac{\partial\,(c\,U_j)}{\partial\,x_j}\right)\,dV = 0\ , \tag{5.5}$$

is integrated over a sequence of control volumes ΔV which are used to discretise the domain.

A discretisation of the transport equation using standard convection schemes is not straightforward. Boundedness of the field could be guaranteed using upwind differencing schemes, however the diffusive character of upwind biased schemes blurs the surface. Higher order differencing is better suited to maintain the sharpness of the interface, however boundedness is not guaranteed. Furthermore, a correct discretisation of the concentration flux depends on the slope of the free-surface as shown in the exemplary flow in Figure 5.1. The Figure depicts a droplet which is transported downward. Predicted con-

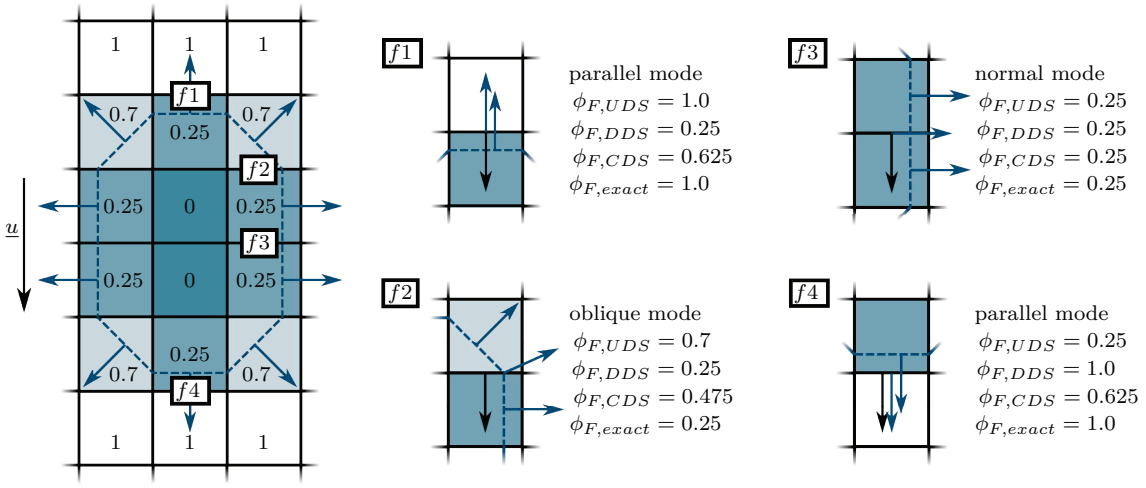


Figure 5.1: Example illustrating the different modes.

centration values at selected faces using different interpolation practices are given. None of the classical interpolation practices is able to predict concentration values which are at least close to the theoretical values for all cases. This simple example reveals that a successful interpolation practice for the concentration value at the faces has to include information about the surface slope.

Within the class of VoF methods several approaches use a geometric reconstruction of the interface which can be used to approximate the concentration value at faces and to determine the interface normals and curvature. Within the Simple Line Interface Calculation (SLIC) algorithm presented by Noh [127] the interface in each cell has been constructed as a line parallel to the Cartesian axis, depending on the the volume fraction distribution of the neighbouring cells. An explicit operator-splitting technique has been used for the advection step. The SLIC algorithm has been replaced by the Piecewise Linear Interface Reconstruction (PLIC) in the method presented by Hirt and Nichols [81]. Since the several improvements of the geometric reconstruction schemes aiming on an improved geometrical representation of the interface are published [7, 114, 116, 123, 147, 152, 201]. All proposed geometric reconstruction methods are afflicted with a high algorithmic complexity even on structured grids composed of rectangular or parallelepipedic cells. Early versions apply the explicit split operator, while recently unsplit operators are developed [111]. The extension of the methods to arbitrary unstructured grids is complicated. Furthermore, the geomet-

ric reconstruction is computationally demanding, and the explicit nature of the advection process limits the time-step. Parallel versions for unstructured three-dimensional grids are published in [93, 94]. However, the computational efficiency in parallel simulations is still an issue, since the geometrical reconstruction only takes place in the free-surface regions which may be unevenly distributed over the processes.

The donor-acceptor scheme published by Ramshaw and Trapp [150] sets the face value for the concentration according to the concentration value in the downwind (acceptor) cell. Accordingly, no explicit reconstruction of the interface is necessary. This approach changes any finite gradient into a step and thus compresses the interface. This behaviour becomes obvious when analysing the resulting matrix coefficients for different convection discretisations and the discretisation of the diffusive flux. The discretised upwinding scheme reads

$$\sum_{O(V)} \max(\dot{m}_F, 0)\, \phi_{NB} - \phi_P \left[\sum_{O(V)} \max(-\dot{m}_F, 0) \right], \tag{5.6}$$

where the sum runs over all faces of the considered cell and all face normals are pointing outward. Discretised central differencing reads

$$\sum_{O(V)} 0.5\dot{m}_F\, \phi_{NB} - \phi_P \left[\sum O(V)\, 0.5\dot{m}_F \right] \tag{5.7}$$

and the discretisation of the diffusive fluxes reads

$$\sum_{O(V)} \frac{\Gamma_F \rho_F \Delta F}{\Delta d_F}\, \phi_{NB} - \phi_P \left[\sum_{O(V)} \frac{\Gamma_F \rho_F \Delta F}{\Delta d_F} \right]. \tag{5.8}$$

The coefficients for a simple one-dimensional equidistant grid given in Table 5.1 reveal

Table 5.1: Matrix coefficients for a 1-dimensional equidistant finite volume discretisation

$$\dot{m}_{F,PD} = -\dot{m}_{F,UP} = \alpha > 0 \quad ; \quad \left(\tfrac{\Gamma\rho\Delta F}{\Delta d}\right)_{F,PD} = \left(\tfrac{\Gamma\rho\Delta F}{\Delta d}\right)_{F,UP} = \beta$$

	A_U	A_P	A_D
$F_{c,DD}$		$-\alpha$	α
$F_{c,UD}$	$-\alpha$	α	
$F_{c,CD}$	$-0.5\ \alpha$		$0.5\ \alpha$
F_d	$-\beta$	2β	$-\beta$

that the coefficients for downwind differencing can be constructed by a sum of central

differencing and diffusive fluxes, the latter although with opposite signs. Therefore, these diffusive fluxes with an opposite sign have a compressive character. However, the approach is unbounded and can create concentrations below zero and above one, e. g. the requested concentration from the acceptor cell may not be available in the donor cell. Therefore, boundedness is considered using the available concentration in the upwind (donor) cell to correct the value predicted by the downwind cell. The compressive character of this discretisation leads to un-physical free-surface shapes when the free surface is aligned with the flow direction. This shortcoming was removed by Hirt and Nichols [81], who introduced informations about the slope of the interface to switch between upwind and downwind discretisations. Further improvements of the method to handle merging and fragmenting interfaces have been presented by Lafaurie et al. [102]. However, the use of first-order accurate upwind differencing introduced additional numerical diffusion, as the matrix coefficients for upwind differencing

$$\sum_{O(V)} \max(-\dot{m}_F, 0)\, \phi_{NB} - \phi_P \left[\sum_{O(V)} \max(\dot{m}_F, 0) \right] \tag{5.9}$$

shown in Table 5.1 can be composed of central differencing coefficients and positive diffusion coefficients.

The difference between the available compressive schemes was outlined by Davis [42] and led to the development of a number of compressive discretisation schemes, based on the blending of downwind and higher order upwind approaches e.g. CICSAM [180], HRIC [125], IGDS [90], BRICS [192], which are state of the art in today's commercial codes. These schemes are developed on the basis of the Normalised Variable Diagram (NVD) which has been presented in Section 3.3. Basically all methods aim to be as compressive as possible, while being numerically stable and satisfying boundedness. To obtain the compressive behaviour the downwind is chosen as the baseline discretisation scheme. Gaskell and Lau [56] have shown that the convective boundedness criteria (CBC) for implicit flow calculation is satisfied if

$$\begin{aligned} \tilde{\phi}_F &= \tilde{\phi}_D \quad \text{for} \quad \tilde{\phi}_D < 0 \text{ or } \tilde{\phi}_D > 1 \\ \tilde{\phi}_D \;<\; \tilde{\phi}_F \;<\; 1 \quad &\text{for} \quad 0 \;<\; \tilde{\phi}_D \;<\; 1. \end{aligned} \tag{5.10}$$

Therefore, all compressive interface capturing schemes use first order upwind differencing in regions where $\tilde{\phi}_D < 0$ or $\tilde{\phi}_D > 1$. Discontinuities in the slope of $\tilde{\phi}_F$ cause a discontinuous

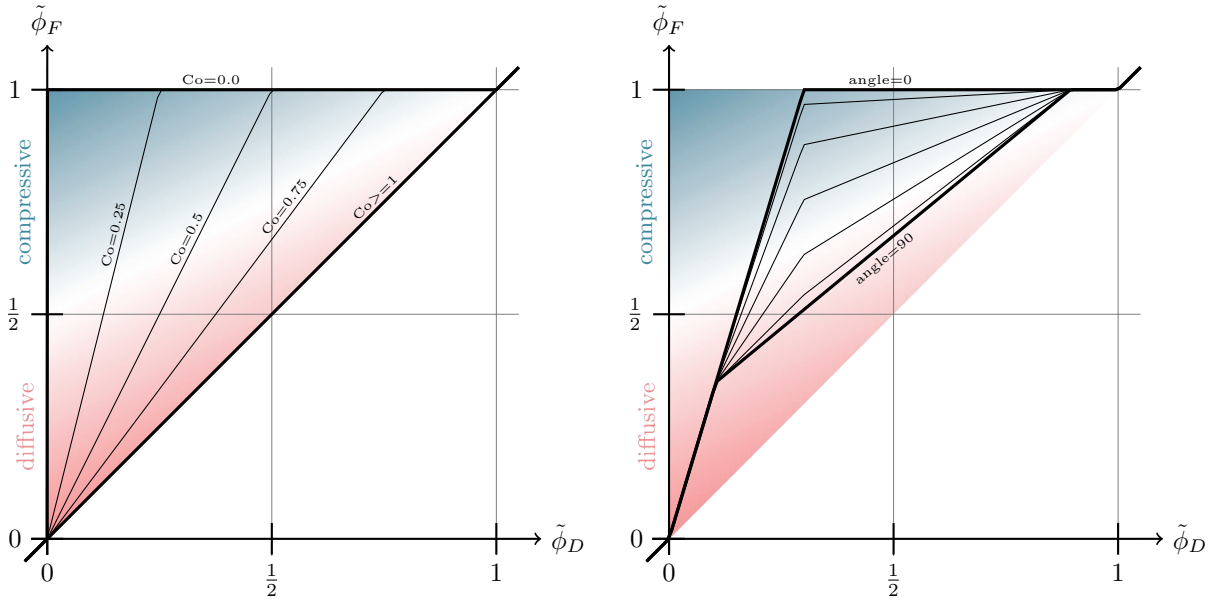
change in the discretisation practice depending on $\tilde{\phi}_D$ which is a function of the field variables. Jasak [88] mentioned that occasionally non-linear NVD schemes are not able to produce unique solutions for steady-state problems, caused by the mutual recursion – the discretisation practice depends on the flow field, which itself depends on the discretisation practice. Therefore, all presented discretisations are at least C_0 continuous, except the CICSAM scheme which has a C_0 discontinuity for Courant numbers equal zero as depicted in Figure 5.2. The normalised variable diagrams introduced in Figure 5.2 illustrate the Courant number and and free-surface to flow direction alignment (angle between free-surface normal and flow direction) dependency of CICSAM. As previously shown, upwind differences can be constructed by a sum of central differences and diffusion terms, while downwind differences can be constructed by a sum of central differences and compression terms. Figure 3.8 shows that central differences in the NVD diagram are indicated by a straight line from $\left(\tilde{\phi}_U = 0\ ;\ \tilde{\phi}_F = 0.5\right)$ to $\left(\tilde{\phi}_U = 1\ ;\ \tilde{\phi}_F = 1\right)$. Hence, the region between this line and the upwind differences ($\tilde{\phi}_F = \tilde{\phi}_U$) is characterised as diffusive while the region in between the CDS line in the NVD diagram and the DDS line ($\tilde{\phi}_F = 1.0$ for $0.0 \leq \tilde{\phi}_U \leq 1.0$) is characterised as compressive. The C_0 discontinuity of the CICSAM scheme is due to the convective boundedness criterion for explicit flow calculations [180]

$$\begin{aligned} \tilde{\phi}_F &= \tilde{\phi}_D \quad \text{for} \quad \tilde{\phi}_D < 0 \text{ or } \tilde{\phi}_D > 1 \\ \tilde{\phi}_D \ < \ \tilde{\phi}_F \ &< \ min\left[1, \frac{\tilde{\phi_D}}{Co_F}\right] \quad \text{for} \quad 0 \ < \tilde{\phi}_D \ < \ 1 \end{aligned} \tag{5.11}$$

obeyed by CICSAM, with Co_F being the face Courant number. Due to the CBC for explicit flow calculation the CICSAM scheme inheres a strong Courant number dependency. However, all presented high resolution schemes depend on the Courant number. Whilst being of importance to the predictive accuracy, an influence of the time step on the stability of an implicit approximation is usually not expected. Employing the Gauss-Green theorem to Equation (5.5), the resulting volume and surface integrals are usually approximated by the 2^{nd}-order accurate mid-point rule. In case of stationary grids and an exemplary implicit 1^{st}-order temporal approximation one obtains the discrete form of (5.5) for the control volume around the cell centre P

$$\Delta V \left[\frac{c_P^n - c_P^{n-1}}{\Delta t}\right] + \sum_F \dot{V}_F \, c_F^n = 0 \, . \tag{5.12}$$

Here, the sum runs over all discrete faces of the control volume ΔV which are represented by their face centroid values. The superscript $n-1$ marks values known from the pre-



(a) Courant number dependency

(b) Angle dependency (angle spacing=15°)

Figure 5.2: Courant number (a) and angle dependency (b) of the CICSAM scheme shown in NVD diagram.

vious time step and $\dot{V}_F$ refers to the discrete volume flux across a face. Using downwind differences to approximate the face values the discretised equation reads

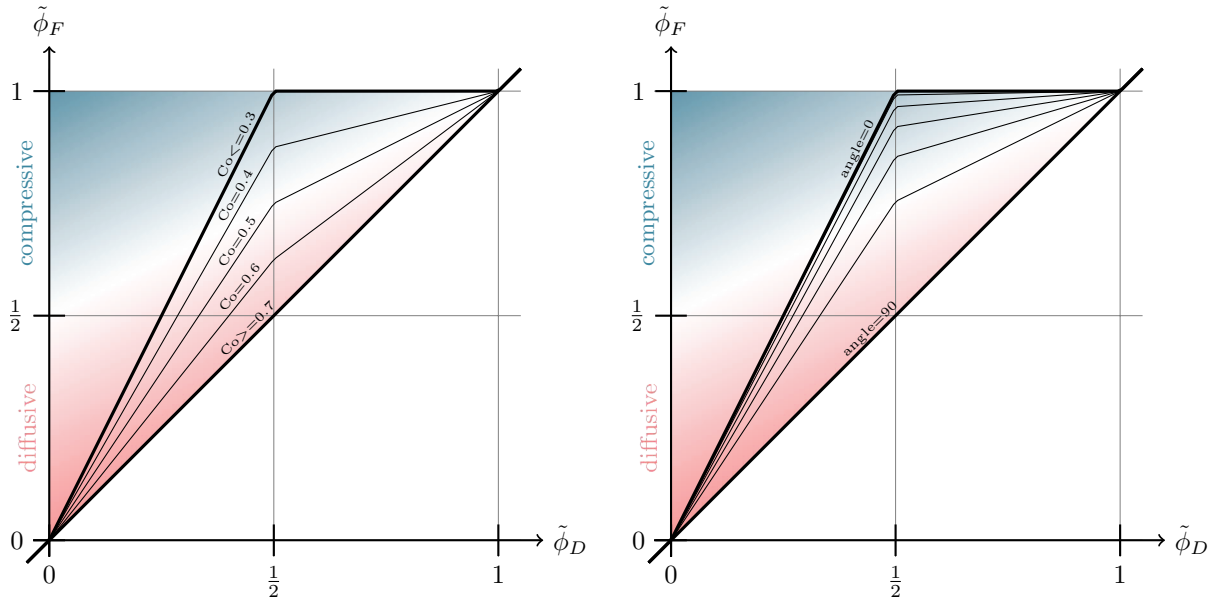
$$\Delta V \left[\frac{c_P^n - c_P^{n-1}}{\Delta t}\right] + \sum_F \max(\dot{V}_F, 0)\, c_{NB}^n - c_P^n \left[\sum \max(-\dot{V}_F, 0)\right] = 0 \,, \tag{5.13}$$

where the subscript NB marks the location of the neighbouring control-volume centre adjacent to the face F. A time-step limitation occurs when looking at the coefficient matrix. Rearranging Equation (5.13) one obtains

$$c_P^n \left[1 - \sum_F \max\left(-\frac{\Delta t\, \dot{V}_F}{\Delta V}, 0\right)\right] + \left[\sum_F \max\left(\frac{\Delta t\, \dot{V}_F}{\Delta V}, 0\right) c_{NB}^n\right] = c_C^{n-1} \,. \tag{5.14}$$

Due to the downwind bias of the interpolation scheme, transient and spatial contributions to the main-diagonal coefficient have opposite sign. The main-diagonal coefficient vanishes for Courant numbers $Co = -\Delta t\, \dot{V}_F / \Delta V = 1$ and diagonal dominance is only preserved for small Courant number magnitudes below 0.5. In 3D applications the Courant number limit gets even lower. Therefore, the presented high resolution schemes smoothly switch to upwind or central differencing biased interpolation schemes with an increasing Courant

number. The Courant number dependencies of CICSAM, HRIC and BRIC are shown in Figure 5.2(a), 5.3(a) and 5.4(a). As for the original donor acceptor scheme downwind



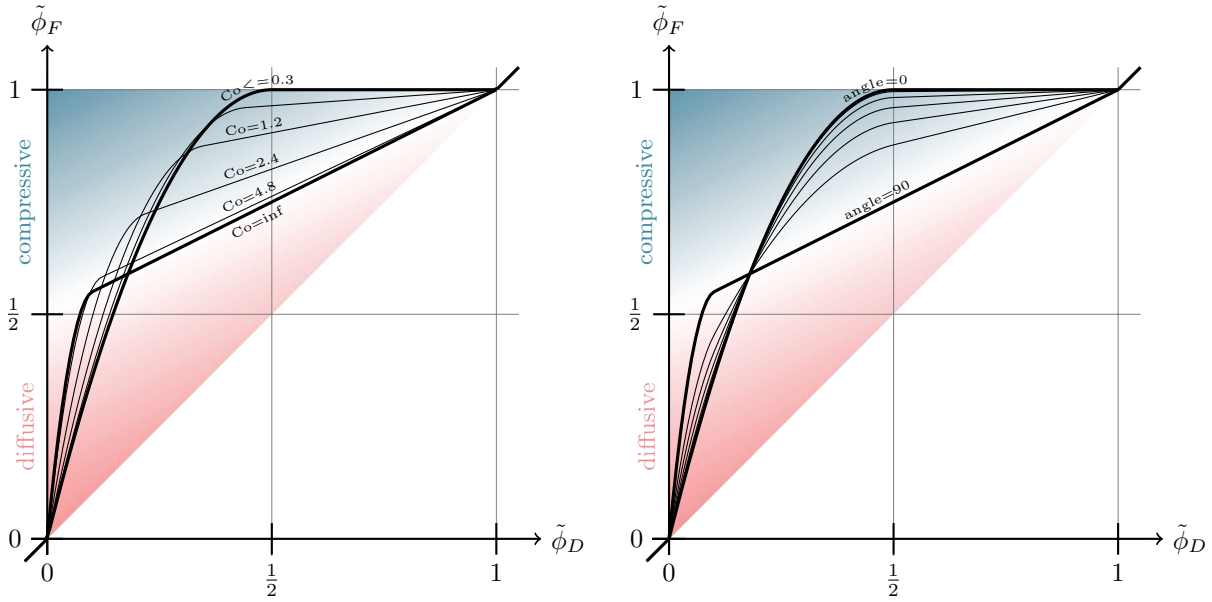
(a) Courant number dependency

(b) Angle dependency (angle spacing=15°)

Figure 5.3: Courant number (a) and angle (b) dependency of HRIC shown in NVD diagram.

differences lead to non-physical interface deformations when the free-surface is aligned with the flow direction, due to its unconditional compressive character. Therefore, all high resolution schemes use a non-compressive fall-back discretisation scheme when the interface is aligned with the flow direction. In the original work of Hirt and Nichols [81], a certain angle has been defined as switching point. Again the sudden switch between discretisation schemes can lead to convergence problems. A smooth transition between the proposed downwind-biased base scheme and the non-compressive fall-back scheme depending on the angle between the surface normal and the face normal is used by all high resolution schemes (Figure 5.2(b), 5.3(b) and 5.4(b)).

All presented schemes try to maintain a sharp interface, by using downwind biased interpolation schemes. However, all schemes occasionally switch to upwind differencing, which introduces numerical diffusion,to fulfil the CBC, stability requirements or to avoid non-physical interface deformations. Therefore, all schemes have the tendency to blur



(a) Courant number dependency

(b) Angle dependency (angle spacing=15°)

Figure 5.4: Courant number (a) and angle (b) dependency of BRIC shown in NVD diagram.

the interface under certain conditions, especially for high Courant numbers. It can be summarised that the presented methods are afflicted with two shortcomings. To maintain their compressive character low Courant numbers are necessary increasing the computational effort and even then they occasionally introduce numerical diffusion leading to partly blurred interfaces.

A methodology to counteract the numerical diffusion based on the artificial compression method of Harten [70] has been developed by Weller and is presented by Rusche [160]. An additional compression term is added to the convection equation for the volume fraction

$$\frac{\partial c}{\partial t} + U_j \frac{\partial c}{\partial x_j} + R_j \frac{\partial}{\partial x_j}[c(1-c)] = 0 \,, \tag{5.15}$$

where R_j is an artificial velocity field suitable to compress the blurred surface, and $c(1-c)$ ensures that the compression is only active in the free-surface region. The direction of R_j is determined by the unit interface normal. However, due to the poor approximation of the interface normal in VoF schemes a smoothed concentration is determined, where smoothing is accomplished using an elliptic relaxation approach, and its gradients are used. The level of compressibility is governed by the magnitude of the artificial velocity

R_j, which is determined according to the maximum velocity magnitude in the cell multiplied by an arbitrary factor. The choice of the arbitrary factor is based on numerical experiments. The factor chosen by Rusche [160] is 1.5 which means that the compression flux can be larger than the usual flux. More recently publications [26, 100] state that the compression flux should be evaluated face-wise and that the magnitude of the compressive flux should be in the order of the usual flux. Gopala [63] showed that the inclusion of the interface compression term can lead to interface deformations. The inclusion of the artificial interface compression term has been adopted by Heyns et al. [77] within their proposed high resolution scheme. However, they report results which are able to maintain sharp interfaces using a factor of 0.1. Although for all proposed factors results keeping the interface sharp are presented the choice of the factor to scale the compressive flux is somehow arbitrary.

The use of downwind-biased interpolation practices, which are necessary to maintain a sharp interface, cause a dependency on the Courant number. Whenever downwind-biased schemes are used a Courant number higher than $Co = 0.3$ will deteriorate the main diagonal dominance. To improve the efficiency of the VoF approach the use of sub-cycles, which employ smaller time steps for the evolution of the mixture fraction than for the other transported quantities, has been proposed by Ubbink [180]. The use of the sub-cycling technique is stated by various authors [23, 77, 160], however the time reduction capability of sub-cycling has only been present by Leroyer et. al. [108] for flows converging to steady state.

In the next section the sub-cycling method is presented in detail, and improvements of the technique when approaching steady state solutions will be shown. In Section 5.3 a newly developed Explicit Interface Sharpening (EIS) scheme will be presented, which aims on an explicit redistribution of the concentration field to maintain a sharp interface and resharpen small inaccuracies of the convective approximations.

5.2 Sub-Cycling Technique

The aim of the sub-cycling technique is to improve the overall efficiency of the solution algorithm for two-phase flows, while maintaining the accuracy. In the previous section the time step dependency of compressive discretisation schemes used for flows with immiscible fluids was highlighted. The time-step limitation retards the transport within numerically

admissible bounds and assures a physically realizable evolution of the mixture fraction. Violations of the limitation introduce spurious non-physical gravity waves, which might persist in the simulation domain and interact in a non-linear manner with physical waves. The time-step limitation, however, pertains only to the mixture-fraction transport. Other properties, e.g. momentum or turbulence parameters, run at much larger time scales.

The idea of the sub-cycling strategy is rather simple and illustrated in the pseudo-code displayed in Algorithm 2. The evolution of the mixture fraction (5.14) follows from an

repeat
  repeat
    solve momentum equation
    solve pressure correction equation
    solve equations for auxiliary fields
    $\Delta t = \frac{\Delta t}{N_{\Delta t}}$
    for *sub-cycle=1,*$N_{\Delta t}$ **do** solve concentration equation
    $\Delta t = \Delta t N_{\Delta t}$
  until *(nOuterIteration≥maxOuterIteration)* **or** *(residual < residualThreshold)*
until *nTimeSteps≥maxTimeSteps*

Algorithm 2: Pseudo-code illustrating the basic integration of the sub-cycling procedure into the RANS solver

embedded sub-cycling approach which is executed over $N_{\Delta t}$ reduced discrete time steps $\Delta t / N_{\Delta t}$

$$c_P \left[1 - \sum_F \max\left(-\frac{\Delta t \, \dot{V}_F}{N_{\Delta t} \, \Delta V}, 0 \right) \right] + \left[\sum_F \max\left(\frac{\Delta t \, \dot{V}_F}{N_{\Delta t} \, \Delta V}, 0 \right) c_{NB} \right] = c_C^{n-1} \, . \tag{5.16}$$

The global cycle of the segregated algorithm remains unchanged. A sequence of outer iterations is performed for each time step. The outer iterations start with the solution of the momentum Equations (2.3), followed by the pressure-correction scheme and an update of the auxiliary fields. The final step refers to the solution of the mixture-fraction Equation (5.5) and the determination of the related fluid properties (ρ, μ) from Equation

(5.2). Utilizing the sub-cycling approach, the solution of the mixture-fraction equation employs not one but multiple time steps to reach the new iterative value.

In practice, the integration of the mixture fraction equation obeys to the Courant-number limit , e.g. $\Delta t/N_{\Delta t} \leq \Delta V/(3\dot{V}_F)$, while the remainder is advanced with a much larger global step (Δt). As usual, coupling terms are mostly lagged during the iterative process, in particular the volume fluxes employed in (5.16). In conjunction with the sub-cycling approach, this practice deserves attention when moving geometries are involved in an Arbitrary Lagrangian Eulerian (ALE) frame, e.g. when simulating floating-body motions. Synchronisation is only reached at the end of the sub-cycling procedure and requires that the history of the mixture fraction is re-initialised before beginning the next outer iteration. For transient flow simulations, synchronisation is obviously necessary. When attention is confined to converged steady state solutions, time-accurate intermediate states are not required and an unsynchronised integration — which omits the reinitialisation step — helps to reduce the computational effort further (cf. Section 7.2.1). The design of the synchronised and unsynchronised version of the sub-cycling algorithm is shown in Algorithm 3.

High-resolution interpolation schemes for convective kinematics frequently employ gradients of the concentration itself to derive the equation system in the framework of unstructured grids. Accurate gradient computations can only be performed if the discrete data is evaluated at the same sub-cycle. On the contrary, the predictive accuracy deteriorates if the concentration field computed at the last sub-cycle of the previous outer iteration enters the evaluation of gradients. Therefore, the concentration field of each sub-cycle has to be either reconstructed linearly from the old and current concentration or saved in temporary fields. The latter approach has been employed, since the reconstruction approach features a non-satisfactory predictive accuracy in conjunction with highly non-linear free-surface motion (e.g. during tank sloshing).

Since most of the equations employ the larger global time step, the total effort should be reduced by an approximate factor of $N_{\Delta t}$. In order to verify the attainable effort reductions, the employed algorithm has been instrumented for profiling the fraction of the execution times T associated to the prediction of the mixture-fraction, i.e. αT. The latter is used to derive an expected ideal sub-cycling speed-up $\mathcal{S}$ for transient cases defined as

$$\mathcal{S} = \frac{T}{\alpha T + \frac{T(1-\alpha)}{N_{\Delta t}}} \tag{5.17}$$

and serves as a reference for the assessment of the achieved speed-up using different numbers of sub-cycles.

if *synchronise* **and** *outer iteration*= 1 **then**
  $c_{save}^{n-1} \leftarrow c^{n-1}$
  if *time discretisation scheme=Implicit Three Time Level* **then** $c_{save}^{n-2} \leftarrow c^{n-2}$
end
for *sub-cycle=1,*$N_{\Delta t}$ **do**
  if *outer iteration>1* **then** $c^n \leftarrow c_{subcycle,save}$
  solve concentration equation
  $c_{subcycle,save} \leftarrow c^n$
  if *sub-cycle<* $N_{\Delta t}$ **then**
    switch *time discretisation scheme* **do**
      case *Implicit Euler* **do** $c^{n-1} \leftarrow c^n$
      case *Implicit Three Time Level* **do** $c^{n-2} \leftarrow c^{n-1}$; $c^{n-1} \leftarrow c^n$
    end
  end
end
if *synchronise* **then**
  $c^{n-1} \leftarrow c_{save}^{n-1}$
  if *time discretisation scheme=Implicit Three Time Level* **then** $c^{n-2} \leftarrow c_{save}^{n-2}$
end

Algorithm 3: Pseudo-code showing the synchronised and unsynchronised version of the sub-cycling procedure within an outer iteration

5.3 Explicit Interface Sharpening Technique

The Explicit Interface Sharpening (EIS) technique developed herein aims to improve the accuracy of the interface prediction while maintaining its efficiency. In contrast to the artificial compression method no user-prescribed factors influence the technique.

It is composed of two major steps:

1. Identify areas associated with a blurred interface.
2. Explicitly redistribute the concentration to sharpen the interface.

Within the first step the sharpness of the interface is face-wise analysed as follows:

A blurred interface is identified when both cells adjacent to the face are partly filled,

A sharp interface is identified if at least one cell is completely filled with one phase.

An illustrative one-dimensional example of a blurred interface is provided in Figure 5.5 where an explicit redistribution leads to a sharp(er) interface after two steps. The admis-

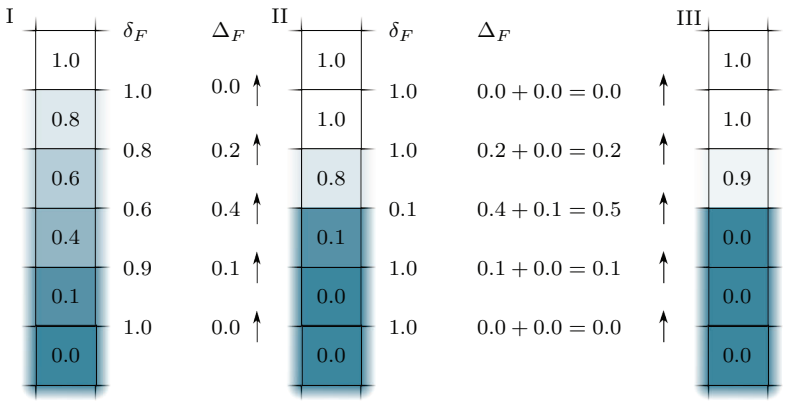


Figure 5.5: Illustration of the basic idea of explicit interface sharpening.

sible transfer is evaluated face-wise considering physically valid liquid-levels in the donor and acceptor cell. The direction of the transfer is evaluated according to the local interface shape. The conservation properties are satisfied by an additional book-keeping instance. The proposed redistribution is added to the right-hand side of the concentration transport equation.

5.3.1 Liquid-Level Sensor and Transfer Rate

An interface is blurred if more than one partly filled control volume occurs between two control volumes completely filled with different phases. Hence, at least two partly filled volumes are direct neighbours along a blurred interface. This serves for the definition of a face-based liquid-level sensor

$$\delta_F = \max\left(\max\left[\left(1-c_L\right), c_L\right], \max\left[\left(1-c_R\right), c_R\right]\right) \quad \in \quad [0.5, 1] \tag{5.18}$$

where the subscripts R and L denote the left and right neighbouring control volumes for the face F. The face-based liquid-level sensor quantifies the largest physically admissible transfer viz:

$$\Delta_F = (1-\delta_F) \quad \in \quad [0, 0.5] \tag{5.19}$$

Experience shows that even if the predicted interface is slightly blurred the iso-surface of the concentration $c = 0.5$, which indicates the interface position, is usually correct. Therefore, a transfer dependant under-relaxation is introduced to avoid spurious interface deformations via

$$\Delta_F = (1-\Delta_F)(2\Delta_F - 1) \quad \in \quad [0, 0.125]. \tag{5.20}$$

This modification reduces the admissible transfer in regions where a large transfer is within the physically admissible bounds. However, due to the iterative solution process for all transport-equations a sharp interface is usually obtained.

The concentration is transferred in the direction of the concentration gradient $\underline{\nabla}(c)$. However, the gradient evaluation of the concentration is difficult, due to its step function character. The proposed procedure is sensible to small variations in the transfer direction. Introduced errors may be accumulated and distort the interface. Therefore, the gradient has to be a sufficiently smooth field. An implicit discrete smoothing operator is applied to the concentration gradient which reads

$$\phi_P^* = \alpha\phi_P + \frac{1-\alpha}{n_{NB}} \sum_F \phi_{NB} \tag{5.21}$$

where ϕ_P^* is the smoothed quantity in cell p, α is a weighting factor and n_{NB} is the number of neighbouring control volumes. The weighting factor α is set to $\alpha = 0.9$.

For each face the transfer oblique to the interface gradient is attenuated by a directionality factor

$$\gamma_{sharp_F} = \frac{\underline{d}_F}{d} \frac{\underline{\nabla}(c^*)}{|\underline{\nabla}(c^*)|} \quad \in \quad [-1, 1]. \tag{5.22}$$

The final concentration transfer for each cell reads

$$\psi_F = \gamma_{sharp_F}\, \Delta_F \quad \in \quad [-0.125, 0.125]. \tag{5.23}$$

5.3.2 Conservation Properties

The explicit sharpening operates face-wise by means of a source term to the mixture-fraction equation for cells adjacent to faces featuring $\delta_F < 1$. The source term aims to explicitly transfer mixture fraction between two cells. Since this correction influences the status of the liquid levels of these cells, it also has an impact on the face sensors δ_F , in particular with respect to potential other transfer candidates that will be treated subsequently.

The above mentioned procedure could yield an 'overbooking' of cells, which in turn would be the origin of liquid sources or sinks. Resulting values could always finally be limited, however, the approach would not be conservative any more and might cause a continuous leakage or convergence problems. In order to obtain a conservative approach and avoid 'overbooking', the procedure requires an additional book-keeping.

For all cells an intermediate concentration due to the sharpening transfer is evaluated

$$c_P^* = c_P + \sum_{F(P)} \psi_F \tag{5.24}$$

and if it is outside the physical bounds a scale factor s based on the concentration defect δc_P

$$\delta c_P = \max\left[\max\left(c_P^* - 1, 0\right), \max\left(-c_P^*, 0\right)\right] \tag{5.25}$$

is evaluated for all face transfers

$$s_F = \begin{cases} 1 \quad \text{if} & c < 0 \text{ or } c > 1 \\ \frac{\delta c_P}{\sum_{F(P)} \psi_F} \end{cases} \tag{5.26}$$

which ensures the boundedness of the concentration. For each face the maximal transfer scale factor s_{max_F} is evaluated. The final contribution to the right hand side of the concentration equation for each cell reads

$$s_P = \sum_F (P) \pm \frac{s_{max_P}\, \psi_F}{\Delta t} \Delta \hat{V}_F, \quad \text{with} \quad \Delta \hat{V}_F = min\left[\Delta V_L, \Delta V_R\right] \tag{5.27}$$

where the source is added for interior and subtracted for exterior cells.

5.3.3 Concluding Remarks

Due to the explicit nature of the developed sharpening algorithm it is able to compress a blurred interface, where the blurred region extends by one cell on each side of the interface. It is possible to apply the explicit interface sharpening technique multiple times within one outer iteration by using the virtually modified concentration levels for each cell as the basis for the subsequent run. Hence, the blurred interface can be compressed by the number of applied cycles. However, tests have shown that the temporal discretisation error is dominant when multiple runs of the explicit interface sharpening algorithm would be necessary. Therefore, only one explicit interface sharpening loop is executed for each outer iteration.

6 Dynamic Grid-Adaptation

Following the discussion of Chapter 4 the accurate representation of flows hinges on an appropriate spatial discretisation of the domain, given that the numerical method is able to predict results at the desired level of accuracy. However, the generation of an adequate (static) mesh requires at least a rough estimate of the simulation results, since cells usually need to be clustered in regions afflicted with large gradients assuming that these regions inherit large resolution dependant errors. The problem gets more challenging in transient simulations, such as violent flows, where the solution changes and an appropriate mesh may be different for each instant of time. A convenient technique to tackle these problems is adaptive mesh refinement. The mesh is adjusted during the simulation to satisfy user-defined criteria. Methods for adaptive mesh refinement have been developed since the early 1980's and will be reviewed in the following section.

6.1 Review of Dynamic Grid-Adaptation

Any mesh refinement technique consists of the following three building blocks:

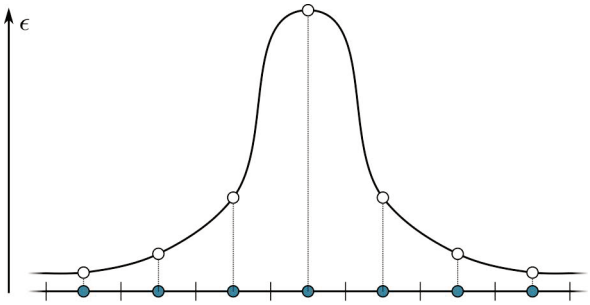
- Mesh optimality criterion,
- Adaptation indicator,
- Mesh (de-)refinement algorithm.

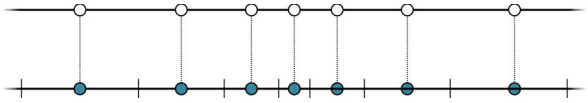
Relevant works for each building block are presented in the following sections.

6.1.1 Mesh Optimality Criterion

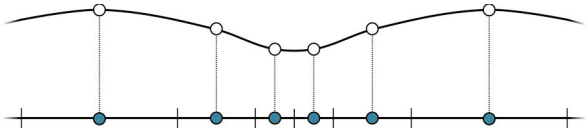
When a grid-adaptation technique is available, the question arises where to refine the grid to obtain an optimal grid and how to measure the optimality. Assuming the local error

ϵ can be estimated, a homogeneous spatial distribution of the local error through grid-adaptation leads to a meaningful optimal grid for the given number of grid points [203]. The left-hand part of Figure 6.1 illustrates an in-homogeneously distributed error. For a





(a) Adaptation through mesh deformation / point movement.



(b) Adaptation through point insertion and deletion.

Figure 6.1: Illustration of the homogeneous error distribution concept.

fixed number of grid points only the location of cell centroids or the size and shape of the cells may be adapted in a cell centred finite volume method. As the error scales with the element sizes, a homogeneously distributed error is achieved through adapted element sizes as shown in Figure 6.1(a). Babuška and Rheinboldt [10] have shown that this optimum is comparably insensitive to small error inhomogeneities. Hence, it is sufficient to have an approximately homogeneous distribution of the local error. The error can be reduced when elements afflicted with large errors are replaced by smaller elements and elements afflicted with small errors are merged to obtain larger cells as depicted in Figure 6.1(b). The error may not only be minimised but reduced below a prescribed threshold when the total number of cells may be changed. Furthermore, the error tolerance can vary over the domain leading to the class of *local absolute error tolerances* [113]. A low error tolerance may be used in regions of interest for the specific simulation, while a larger error tolerance may be assigned to other regions. This approach leads to a lower total number of cells reducing the computational effort. whilst being attractive from an efficiency point of view, the local absolute error tolerance approach requires knowledge about regions of importance and unimportance.

6.1.2 Adaptation Indicator

A wide range of adaptation indicators, which differ substantially in their associated complexity and success in approaching an optimal grid, have been developed in the past decades. A brief overview about adaptation indicators applicable in the finite volume framework is provided in the following sections.

a) Feature-Based Criteria

Many flows inherit highly localised flow features, e.g. interfaces in multiphase flows, shocks in compressible flows, boundary layers, tip vortices and many more. Assuming that large gradients, which can be attributed to localised flow features, indicate regions associated with large errors leads to the class of feature-based adaptation criteria (e.g. [19, 92, 197]). However, continuous refinement in these regions does not necessarily reduce the global error in the solution. Some examples of increasing errors due to feature-based grid-adaptation are reported by Dwight [48] and Warren et al. [197]. Feature based criteria flag a certain region for refinement, however they usually don't provide any information about a target cell size. Therefore, a stopping criterion for the adaptation process has to be provided. Frequently employed criteria are the maximum number of refinements or the minimal cell size. Although the deficits of feature-based adaptation criteria are well known, it is often used in marine free-surface flows [191, 193] due to its simplicity.

b) Local Error Estimates

Another class of adaptation indicators aims on an estimate for the local error. The most popular error estimation technique in the finite volume framework is Richardson extrapolation [155]. Richardson extrapolation is based on the evaluation of the same problem on at least two different grids of increasing resolution. Reasonably accurate results are achieved ([24, 124, 178]), assuming that all utilized grids are fine enough to observe error reduction in the asymptotic range. However, the associated effort to determine solutions on grids of increasing resolution is large. The direct Taylor Series Error estimate by Jasak [88] aims to provide an error estimate based on a single mesh within a single run. Therefore, the second gradient tensor of the Taylor series is estimated applying the Gauss' theorem twice. It has been shown that the direct Taylor Series Error estimate consistently underestimates the error, which is an undesired feature. Furthermore, it is afflicted with

some uncertainties in second-order accurate discretisation frameworks, due to the repeated application of the Gauss Theorem.

Simple estimates for the local error based on the solution obtained from only one grid are derived using interpolation theory. Under the assumption of a smooth solution the second derivatives of a scalar field, also known as Hessian, are proportional to the error associated with a piecewise-linear interpolation of that field ([12],[55],[66],[112]). In an unstructured grid framework for finite volume schemes special care has to be taken to accurately evaluate the second derivatives [193]. For a second-order accurate method, a second-order accurate estimation of the second derivative must be based on more than just the direct neighbours.

Another method proposed by Jasak [88] is based on the transport of higher moments of the variables. When the original transport equations would be solved without errors, all higher moment equations would be satisfied automatically. A measure of the local error based on the imbalance of the higher moment equation, calculated on a cell-by-cell basis has been introduced.

A technique inspired by error estimates used in the finite element framework is the residual error estimate by Jasak [88]. Second-order accurate finite volume discretisation assumes a linear change of the variable over the control volume. However, the solution usually is associated with non-linear changes. Using the value and gradient of a variable at one cell, values at the faces, which satisfy the assumption of the linear change, can be assembled. On the other hand values at faces can be obtained from linear interpolation of the adjacent cells. The difference in the values obtained from both methods is used to assemble the residual error estimate.

All proposed methods estimate the error for a cell, based on the solution in the cell itself and its near vicinity. In this sense they are local error estimates. Following a classification from Roache [155] these methods can be called algebraic. However, it is well known that the error for an unknown is transported in the same way as the unknown itself [157]. Therefore, errors introduced at one location of the mesh may be transported to other regions. Local error estimates would give an indication to refine the mesh in the area where the error is located at one instant of time, but not necessarily at the origin of the error. Published methods that are able to account for these effects will be presented in the subsequent paragraph.

c) Global Error Estimates

Methods that account for effects due to the transport of the error rely on the solution of partial differential equations that mimic the transport. A method to derive a transport equations for the error was introduced by Babuška and Rheinboldt [11] for finite element methods (FEM). Considering the transport of ϕ, through a differential operator L, and a source term S

$$L(\phi) = S \tag{6.1}$$

the insertion of an approximate solution ϕ^*

$$L(\phi^*) - S = R \tag{6.2}$$

produces a residual R. Therefore, a transport equation for the error e reads

$$L(e) = -R \quad \text{with} \quad e = \phi - \phi^*. \tag{6.3}$$

Extensions of this method to the finite volume framework were provided by Qin et al. [149]. The evaluation of the error transport equations is quite demanding in terms of the discretisation practice, which should be of higher order than the discretisation of the flow variables. Applications in an unstructured grid framework and extensions to turbulent flows were published by Hay et al. [75]. However, no application of the error transport equation in finite volume frameworks to complex three-dimensional flows have been reported, possibly due to the complexity of the required higher order discretisation for arbitrary control volumes. Two alternatives to a higher order approximation are proposed by Roy [157]. The first one relies on an approximation of the exact solution through Richardson extrapolation. This however requires the solution on at least two grids of increasing and sufficient resolution. The second one is based on continuous approximations of the discrete solutions, which can be found through curve-fitting techniques (e.g. Roy and Sinclair [158]). However, the application of this technique to complex three-dimensional turbulent flows has not been demonstrated and is deemed to be cumbersome. An interesting combination of this technique with the adjoint correction technique has been published by Pierce and Giles [146].

Applications of CFD in industrial environments usually aim to predict a single objective, e.g. the drag or the lift force. Though, error estimates based on the solution of partial differential equations for the transport of the error provide an accurate estimate

of the error, there is no indication whether the error has an influence on the accuracy of the predicted objective functional. A link between the error and the functional of interest can be established by adjoint methods. Giles [58] introduced the concept of goal-oriented grid-adaptation, which is based on *a posteriori* error estimates presented by Becker and Rannacher [21]. The solution of the adjoint partial differential equations with inhomogeneous source terms and/or boundary conditions appropriate for the particular objective functional defines the relationship between the error in the functional and the local error in the discretisation. Using this relationship grid refinement can be applied to locations where the local error has an influence on the functional of interest. The concept presented by Giles [58] has been the starting point of intensive research activities for goal-oriented grid refinement especially in the finite element framework applied to aerodynamics.

Venditti and Darmofal [187, 188] presented a technique based on the work of Pierce and Giles [145], where the adjoint error correction is used to correct the objective functional, and the adaptation is driven by the remaining error in the corrected functional. The developed technique has been applied to one- and two-dimensional viscous flows. An extension towards multiple objectives was presented by Hartmann et al. [72, 74]. The method is based on the calculation of the primal problem, the adjoint problem and an adjoint adjoint problem, taking into account an unlimited number of different functionals. The error indicator provided from goal-oriented refinement strategies is a scalar value at each cell. Therefore, it does not incorporate any information for anisotropic refinement. On the other hand, anisotropic refinement can drastically reduce the number of necessary cells especially for three-dimensional flow problems. Venditti and Darmofal [189] substituted additional information from the Hessian of a primal scalar flow quantity to the refinement decision to enable anisotropic refinement for goal-oriented approaches. They successfully applied the methodology to two-dimensional flows. Further developments of this approach for parallel computing environments have been reported by Lee-Rausch [104] for three-dimensional inviscid flows. Park [134] developed a direct control approach to drive the anisotropic adaptation. It takes into account anisotropic effects from both, the primal and the adjoint solution, blended relative to their importance for the functional. A method, based on simplexes, targeting the mesh quality for the anisotropy decision has been proposed by Houston [83]. Another approach to incorporate the mesh quality in the refinement decision has been proposed by Ceze et al. [33] for quadrilateral elements in two dimensions and hexahedrons in three dimensions. Loseille et al. [115] derived a

different strategy for anisotropic goal-oriented grid-adaptation. It is derived by an *a priori* version of the adjoint error estimator and its resolution in an abstract continuous framework, representing the grid. Few attempts for transient goal-oriented mesh adaptation are published. This is possibly due to the fact that the adjoint solution travels back in time and therefore relies on the complete time history of the primal advancing in time. A combination of the anisotropic goal-oriented error estimate from Loseille et al. [115] with the transient fixed-point mesh adaptation algorithm [6, 65] has been proposed by Belme et al. [22] and applied to a two-dimensional problem. However, nowadays applications of this method in three-dimensional complex flows seem to be too demanding from the computational point of view. The goal-oriented grid-adaptation procedure has been extended to turbulent flows by Hartman et al. [73] using the $k-\omega$ two-equation model and Park [135] using the Spalart-Allmaras one-equation model. The challenges associated to turbulent flows are the derivation of the adjoint of the turbulent transport equations and a robust grid-adaptation procedure, as the grid is usually very fine in the boundary layer. All previously mentioned goal-oriented grid-adaptation procedures use a discrete adjoint approach, where the discrete adjoint equations are derived by transposing the primal discrete transport equations. While the discrete adjoint approach ensures duality it can be very difficult to derive, especially when the Jacobian of the complete system matrix is not stored. Discrete adjoint methods using automatic differentiation techniques are fairly difficult to apply to industrial procedures, which heavily employ scripting, hybrid coding languages, operator overloading and pre-compiled third party libraries. They are also extremely storage demanding and not well suited for HPC . The continuous adjoint approach, which is based on the continuous adjoint equations derived from the continuous primal equations, offers a large flexibility in implementation. However, special care has to be taken to ensure duality. The only published work of goal-oriented grid-adaptation based on the continuous adjoint approach is from Li, Allaneau and Jameson [109]. It is applied to the Euler equations for a two-dimensional airfoil in sub- and transonic flow. Results indicate that goal-oriented refinement procedures based on either discrete or continuous adjoints are able to refine the mesh in suitable regions.

6.1.3 Mesh (De-)Refinement Algorithm

Mesh (de-)refinement algorithms can be divided into three groups:

p-refinement Within p-refinement techniques the order of the discretisation is locally increased in regions with large errors. It is usually used for finite element discretisations, where the shape function can vary locally. Increasing the order of the shape function introduces new computational nodes. Therefore, the mesh topology or at least the topology of the unknowns is changed. As shape functions are usually smooth, the method suits better to flows with smooth changes of the solution than to flows with discontinuities (e.g. shocks, free-surface).

r-refinement A refinement technique which maintains the grid topology and number of computational points is called r-refinement. It locally reduces the error by clustering computational nodes in regions afflicted with large errors. The number of computational nodes stays constant, therefore the reachable accuracy is limited. Special care has to be taken to ensure sufficient grid quality.

h-refinement Adding new computational nodes, while maintaining the order of discretisation and the locations of the initial computational nodes is known as h-refinement. In regions afflicted with large errors new computational nodes are added to obtain a better spatial resolution and minimise the error. Furthermore, in regions with low computational errors computational nodes may be removed. In contrast to *p-refinement* this technique is well suited for problems with discontinuities.

Within the present work only *h-refinement* has been used, because it is suitable for flows with discontinuities and allows to reduce the error to a desired level. Therefore, the subsequent review of grid refinement and coarsening algorithms focuses on h-refinement techniques.

Published h-refinement techniques can be classified as *structured mesh refinement techniques* and *unstructured mesh refinement techniques* [45]. *Structured mesh refinement techniques* use a hierarchy of rectangular grids and exchange the solution through the boundaries of the grids to higher or lower levels of the hierarchy. *Unstructured mesh refinement techniques* are based on a local change of the grid topology and a mapping of the solution from the initial grid to the adapted grid.

a) Structured Mesh Refinement Techniques

The first published developments of mesh adaptation algorithms were based on the multi-grid approach. Brandt [29] adopted the idea not to work with a single grid, but with a sequence of local grid patches at different levels of resolution. Levels may be introduced and changed during the solution process and interact with each other. The interaction of the different levels is managed through boundary conditions. Further enhancements of this technique were presented by Caruso [31]. The patches introduced at each level may be rotated relative to the base grid. Optimisations of the patch positioning and size have been presented by Berger et al.[25], using a clustering algorithm. In a later work, Berger and Colella [24] focused on unsteady flows with moving shocks, where mass conservation was a critical issue. Errors concerning mass conservation are introduced at the boundaries of the patches, especially when rotated against each other. An efficient cure was the alignment of grid lines of overlapping patches. Another work of Thompson and Ferziger [178] aimed on solving the mass conservation issues at the boundaries of patches through modified and improved interpolation techniques. However, the problem has never been solved appropriately.

An attractive feature of *structured mesh refinement techniques* is the use of efficient dedicated solution algorithms for structured rectangular grids. Furthermore, these methods have low storage requirements as no explicit topological data structure is necessary. On the other hand the usage of rectangular grids poses limits on the geometric complexity of the flow problem.

Parallelisation of *structured mesh refinement techniques* can borough basic techniques that exist for non-adaptive structured mesh methods. In both cases a patch is associated to a processor, and inter-processor communication is established through the boundaries of the patches. Adaptive mesh refinement adds complexity to the communication patterns as the grid configuration is strongly irregular and changes throughout the simulation.

A number of toolkits which provide the infrastructure for *structured mesh refinement techniques* exist. Prominent examples are BoxLib [27] and Chombo [35] from Lawrence Berkeley National Laboratory.

b) Unstructured Mesh Refinement Techniques

Unstructured mesh refinement techniques are based on the nested splitting or merging of existing topological entities. Bank et al. [17] provided a general description of refinement algorithms for two-dimensional problems discretised by triangles or quads. An essential problem associated with nested refinement is that certain grid quality measures, which depend on the underlying discretisation procedure to solve the governing equations, have to be satisfied. Nested refinement usually changes these quality measures, and when they are violated, the refinement has to be propagated to neighbouring cells to maintain the grid quality. Additional refinement of a neighbouring cells again changes the quality measures, therefore refinement may be propagated over several cell layers. A widely used criterion to judge the mesh quality and drive the propagation of the refinement information is the *1-irregular* rule for two-dimensional elements. It says that any un-refined element having a side with more than one irregular vertex needs to be refined. Bank et al. [17] focused on isotropic refinement, where each edge associated to the element marked for refinement is split. Extensions to anisotropic refinement for simplexes in two dimensions have been proposed by several authors [153, 154, 166]. A comparison of these methods by Mitchell [121] revealed that their performance applied to the solution of elliptic problems using the finite element scheme was quite similar and clearly superior to uniform refinement. Vilsmeier and Hänel [190] developed a finite volume method for the solution of Euler and Navier-Stokes equations that was supplemented by a grid-adaptation procedure for simplexes in three dimensions. The h-refinement was based on a cell-by-cell method, emphasizing on grid quality measures. A combination of multi-grid acceleration and local grid refinement was developed by Muzaferija [124]. The grid refinement algorithm was based on quadrilateral and hexahedral elements, rather than on simplexes. Results for three-dimensional turbulent flows were produced with the proposed method. A three-dimensional numerical method using the finite volume discretisation based on polyhedrons has been presented by Jasak [89]. Unstructured adaptive mesh refinement for hexahedrons is incorporated in the method. It allows anisotropic splits of hexahedrons, and uses an adopted 1-irregular concept inspired by Berger [25], where the number of neighbours for each cell is limited to seven. A coarsening procedure based on lists of cell pairs was implemented, where pairs with minimum surface areas are preferred. The refinement algorithm is implemented in serial. Parallel simulations are based on a sequence of parallel

execution of the flow solver, collection of the results, serial execution of the refinement procedure and redistribution of the new grid for the next calculation cycle.

Since the early 1990's research focused on the development of *parallel* unstructured grid refinement techniques. In parallel implementations of grid refinement algorithms the uniqueness of a topological entity has to be ensured. In the widely used Single Program Multiple Data (SPMD) approach the computational domain is decomposed according to the number of processes. If a vertex is introduced on an edge at a process boundary, where the cells adjacent to the edge always also belong to different processes, then the insertion has to be done on all processes adjacent to the edge and the same global ID has to be assigned to the vertex. Furthermore, it has to be ensured that the propagation of refinement decisions, necessary to ensure grid quality measures, can cross process boundaries. An important issue in parallel local grid-adaptation is load-balancing. Local grid refinement usually adds different amounts of work load to each processor. Without re-balancing of the workload the overall performance of the numerical method is severely deteriorated. One of the first published approaches from Williams [200] was based on a voxel database approach for parallelisation of the refinement algorithm for two-dimensional simplexes. Later research from Kallinderis et al. [97] focused on the development of dedicated data-structures and generic primitives to manage the connections of the faces over process boundaries. They showed that the developed primitives were suitable for shared- and distributed memory systems. An implementation from Jones [95] divided the refinement procedure for two-dimensional simplexes into two phases. Within the first phase the sub-meshes, located entirely on one process, are refined independently. Therefore, the shared faces at partition boundaries are treated as if they were boundary faces. The resulting overall mesh of the first step is composed of locally conforming sub-meshes with non-conforming shared faces. In the second phase, simplexes containing one or more shared faces, which have been bisected during the first phase, are exchanged between neighbour sub-meshes. Subsequently all simplexes having one or more non-conforming shared faces are bisected. The process is repeated until the global conformity of the mesh is reached, i.e., no more non-conforming shared faces exist. A re-partitioning algorithm was implemented to ensure load-balance. A method capable to adapt simplexes in two and three dimensions from Castanos et al. [32] used the same principle for a parallel implementation of the bisection method from Rivara [154]. It has been proven that serial and parallel runs of the code on the same problem lead to identical grids. Facing the challenge of load bal-

ancing Bank et al. [16] proposed a different grid refinement philosophy consisting of three steps. First the problem is solved in serial using a coarse mesh following a partitioning that ensures that the estimated error is equally distributed over all partitions, which may largely differ in their number of grid points. Afterwards each process solves the whole problem on the complete coarse mesh, where the process is allowed to adapt the mesh only in the region identified by the first step. Finally the refined portions of the meshes on each process are merged and regularised, and the resulting mesh is used for the final simulation, which is based on standard domain de-composition techniques. Although the proposed scheme eases some issues related to parallel grid-adaptation, it is clearly limited in the maximum number of processes, due to the solution of the whole problem on coarse meshes during the first step.

Wackers et al. [194] developed a finite volume method which incorporates anisotropic grid-adaptation of hexahedral cells in parallel. They provided a brief description of the refinement procedure, which is based on an initial flagging of cells for refinement depending on the indicator and a subsequent propagation of the refinement flags to the neighbouring cells in case that certain cell qualities are violated. The refinement itself is performed on a cell by cell basis, leaving a valid grid after each refinement. Each partition is refined independently based on the refinement flags from the previous step. No additional communication across partition boundaries is necessary during the refinement process. Various applications of the developed adaptation procedure mainly to marine free-surface flows with complex three-dimensional flow features are provided (e.g.[191, 193]).

When a grid is adapted at a curved boundary, new inserted vertices on an edge located at the boundary have to be projected to the curved boundary, to generate an accurate geometry representation. However, this may lead to invalid inverted elements. To ensure element validity the displacement of the boundary vertex has to be propagated into the volume mesh. Various methods to propagate the displacement have been proposed, e.g. Laplacian smoothing, linear springs, linear elasticity. However, as noted by Park [134], none of these methods is able to guarantee a valid grid. Therefore, he proposed another concept based on the cut-cell method. Within this approach the background grid is generated without any knowledge of the curved boundaries, and subsequently all cells that intersect with the boundary are cut. Though this method is very powerful in producing geometrically valid grids, it is very hard to provide grids at curved boundaries suitable to accurately resolve turbulent boundary layers.

c) Concluding remarks

Balasubramanian et. al [12] published a comparison of feature-based grid-adaptation to goal-oriented grid-adaptation. The study revealed that grids obtained from goal-oriented grid adaptation are able to provide the same level of accuracy – for the specific objective – as feature-based grid-adaptation. However, grids obtained from feature-based adaptation had three to five times more cells. Furthermore, in contrast to feature-based grid-adaptation goal-oriented grid-adaptation does not suffer from poor initial grid resolution. Therefore, many goal-oriented grid-adaptation techniques have been developed. The developments have focused on aerodynamic compressible flows, usually afflicted with shocks. Shocks require a substantial grid resolution and are very localised. Therefore, they are a very suitable problem for mesh adaptation. Mesh adaptation for flows which have less localised features are more challenging, especially for feature-based grid-adaptation algorithms. Goal-oriented grid adaptation for these kind of flows is expected to be clearly superior to feature-based grid adaptation algorithms. However, to the authors knowledge no applications of methods using goal-oriented grid-adaptation in complex turbulent three dimensional flows, without shocks, are present in literature.

The majority of developed and published grid-adaptation techniques is based on simplexes. The use of simplexes simplifies the algorithmic implementation of grid-adaptation. However, using simplexes often introduces numerical diffusion, because the cells can hardly be aligned with the flow. Due to this reason grids based on quadric elements in two dimensions and hexahedrons in three dimensions are often preferred, especially in free-surface flows. Considering free-surface flows anisotropic grid-adaptation is an essential feature to generate adequate grids. However, published literature lacks a complete description of grid-adaptation algorithms for anisotropic adaptation of hexahedrons.

Nowadays parallel computation of complex problems based on unstructured cells is supposed to be standard approach. Accordingly, a grid-adaptation feature should be fully integrated into the parallel computation procedure without posing any addition requirements, like e.g. first merging the mesh from the various processes to a single process on which the actual adaptation takes place and a subsequent redistribution of the mesh to the processes. Furthermore, the adaptation technique should also be scalable. From the reviewed methods, only the method presented by Wackers et al. [194] is able to fulfil these requirements. Again, available literature does not provide a complete description of the

parallelisation.

6.2 Mesh Optimality Criterion

The dynamic mesh adaptation method developed herein can be combined with adaptation indicators from feature- and error-based criteria as well as indicators from goal-oriented refinement. Furthermore, anisotropic refinement is supported. The indicator $\mathcal{I}_i$ in each cell is a vectorial quantity to account for anisotropy. It contains the indicator information in the directions of the Cartesian reference frame unit vectors. However, the refinement of the cell is aligned to its local coordinate system which usually does not coincide with the Cartesian reference frame (Figure 6.2). Therefore, the indicator vector is projected to the

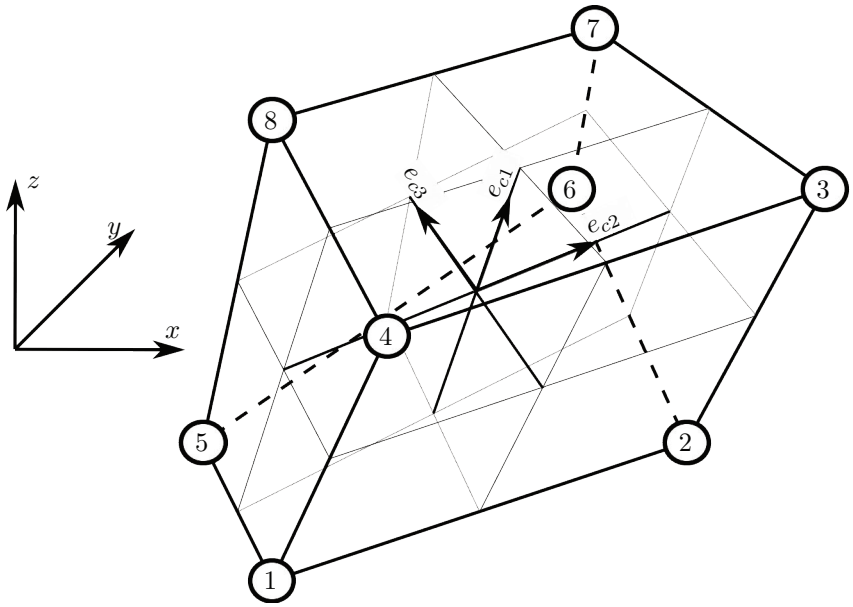


Figure 6.2: Local coordinate frame of a hexahedron.

subspaces defined by the unit vectors e_{c1}, e_{c2} and e_{c3} of the local cell coordinate system, to derive a refinement indicator in the direction of the local cell unit vectors. When the indicator is above an user defined threshold the cell is marked for refinement in the specific direction. Refinement indicators that provide information for isotropic refinement set all values of the vector to the indicator value.

In some cases it is desirable to combine several refinement criteria. The different criteria

may be associated with different refinement thresholds. To allow a combined analysis of error indicators with different thresholds each indicator $\mathcal{I}_i$ is normalised by the associated threshold $\mathcal{T}$ and the maximum normalised indicator for each direction is stored. Because all indicators are scaled to their associated threshold, cells that have indicator values above one are flagged for refinement.

The developed dynamic mesh adaptation method in general obeys the principal of spatial equal distribution of the local error. However, in certain situations it is known in advance that the errors in specific locations will probably have a negligible influence on the results that have to be obtained from the analysis. In these situations the refinement can be restricted to certain geometrically defined regions of the mesh. Moreover, limitations for the resulting cell sizes can be prescribed. They are given in the Cartesian reference frame and projected into the cell coordinate system of each cell.

Special care needs to be taken when refinement occurs at walls in turbulent flows. The boundary layer at no-slip walls reacts very sensible to changes in the normal distance of the first cell layer to the wall. Therefore, the refinement in the normal direction of the wall should be avoided, unless all cells at the wall are refined in that direction. Furthermore, when the law of the wall is employed, the centre of the first cell should be located considerably outside the viscous sub-layer. The easiest way to ensure that both criteria are met is to exclude cells in the boundary layer from refinement. Simulations where cells in the boundary layer are not allowed to be refined, rely on a sufficiently fine mesh at the wall to capture all relevant flow features. On the other hand it is often not known which regions at the wall need to be sufficiently fine and what sufficiently fine means. Moreover, using a fine discretisation of wall surfaces in the simulations counteracts potential benefits from automatic grid-adaptation due to the lower computational effort associated to a grid with less cells. Therefore, a second methodology to cope with boundary layers is to prevent the refinement of boundary layer cells just in the direction of the wall normal, which automatically leads to anisotropic refinement. The remaining issue is the detection of boundary layer cells. Although an automatic detection based on turbulence quantities of the flow could be developed, in the present contribution all cells that have a wall-distance lower than an user-prescribed threshold are treated as boundary layer cells.

Solely using the adaptation criterion to identify cells for refinement would usually result in grids associated with a poor mesh quality. Therefore, the adaptation algorithm propagates refinement decisions to neighbouring cells to satisfy certain mesh quality criteria.

Quality criteria that are taken into account during the adaptation procedure are:

- mesh irregularity,
- cell aspect ratios and
- mesh orthogonality.

A detailed description of the procedure to flag neighbouring cells for refinement follows in the description of the refinement procedure.

6.3 Adaptation Indicator

Adaptation indicators that have been used within this work will be briefly presented in this section. The description will focus on implementation issues rather than the mathematical background which can be found in the given references.

6.3.1 Feature-Based Criteria

Many maritime flows are two-phase flows. The accurate prediction of these flows depends on suitable convection schemes, possibly with extensions that have been introduced in Chapter 5, and a sufficient resolution of the free-surface. Refinement at the free-surface is a natural candidate for anisotropic refinement, because a good resolution is mostly required normal to the free-surface. The normalised gradient of the mixture fraction seems to be a natural choice for the anisotropic refinement indicator. However, as outlined in Chapter 5 the computation of the gradient is usually afflicted with a low quality due to the step-function character of the mixture fraction. Therefore, an alternative indicator based on the absolute difference of the mixture fraction in adjacent cells has been derived. The difference is evaluated at each face and multiplied by the normalised connection vector of the adjacent cells ζ_i

$$\Delta c_{F,i} = |c_L - c_R|\zeta_i. \tag{6.4}$$

The adaptation indicator in cell P is the maximum value of the face indicator $\Delta c_{F,i}$ from all surrounding faces

$$\mathcal{I}_{P,i} = max(\Delta c_{\alpha,i}) \quad \alpha \in F(P) \tag{6.5}$$

separately evaluated in each coordinate direction.

6.3.2 Local Error Estimator

The residual error estimator from Jasak [88] has been implemented to compare adaptation procedures based on local and global error estimators. In finite element methods the residuals measure how good the local solutions satisfy the original governing equations. Therefore, the level of the residuals is associated with the local solution error. The residual error in a finite volume discretisation can be derived along a similar route. If the exact solution of a problem varies over a control volume at a higher order than the numerical order of accuracy (p), a control volume imbalance exists. In second-order accurate schemes, the residual error E_P in a control volume P for a steady-state scalar-transport equation is given by

$$E_P(\phi) = \sum_{F(V_P)} F_i \left[(\rho u_i \phi)_F - (\rho \Gamma_\phi)_F \left(\frac{\partial \phi}{\partial x_i} \right)_F \right] - \breve{S} V_P - \breve{S} \phi_P V_P \tag{6.6}$$

with F_i denoting the face vector coordinates, ρ the density and Γ_ϕ, $\breve{S}$, $\breve{S}$ and V_P the diffusion coefficient, the constant part of the source term, the linear part of the source term and the control volume of cell P. For a second-order accurate implementation the face values of ϕ and $\frac{\partial \phi}{\partial x_i}$ are determined via

$$\phi_F = \phi_P + (x_{F_i} - x_{P_i}) \left(\frac{\partial \phi}{\partial x_i} \right)_P \quad \text{and} \quad \left(\frac{\partial \phi}{\partial x_i} \right)_F = \left(\frac{\partial \phi}{\partial x_i} \right)_P . \tag{6.7}$$

This error estimate is then normalised by

$$F_{norm,P} = \sum_F \left[\Delta F \frac{(\rho \Gamma_\phi)}{d} \right]_F + \sum_F max(\dot{m}, 0) + \breve{S} \tag{6.8}$$

with d being the distance between the cell centres of the two cells adjacent to face F, $\dot{m}$ the flux through the face and ΔF the face area. The final error estimate reads

$$\epsilon_P(\phi) = \frac{E_P(\phi)}{F_{norm_P}} \tag{6.9}$$

and has the same dimension as ϕ.

6.3.3 Goal-Oriented Criteria

As shown by Giles [60] the error in a scalar quantity of interest $J(\phi_i)$ that depends on the primal solution variables ϕ_i can be expressed as

$$J_h(\phi_{i,h}) - J_h(\phi_i(x_h)) \approx \hat{\phi}^T_{i,h} \mathcal{R}_h(\phi_i(x_h)) \tag{6.10}$$

where the subscript h denotes a grid of density h and $\phi_i(x_h)$ and $\phi_{i,h}$ are the analytical and the numerical solution on this grid. The error can be expressed as the inner product of the adjoint solution variables $\hat{\phi}_{i,h}$ and the vector of truncation errors $\mathcal{R}_h(\phi_i(x_h))$, obtained by substituting the analytical solution $\phi_i(x_h)$ into the discrete residual operator $\mathcal{R}_h$. Assuming that an approximate analytical solution can be obtained, an approximate error can be evaluated, which can be used as a grid refinement criterion.

A more elaborate procedure to estimate the error is described by Venditti and Darmofal [187]. It is based on the correction technique presented by Pierce and Giles [145]. A corrected value for the scalar quantity of interest $\tilde{J}_h(\phi_{i,H})$ is given as

$$\tilde{J}_h(\phi_{i,H}) = J_h(\phi^H_{i,h}) - \underbrace{(\hat{\phi}^H_{i,h})^T \mathcal{R}_h(\phi^H_{i,h})}_{AdjointCorrection} - \underbrace{(\hat{\phi}_{i,h} - \hat{\phi}^H_{i,h})^T \mathcal{R}_h(\phi^H_{i,h})}_{RemainingError} \tag{6.11}$$

where the subscript h indicates a systematically refined grid originating of grid H, e.g for hexahedrons one cell in H is split to eight cells in h, and $\phi^H_{i,h}$ and $\hat{\phi}^H_{i,h}$ are reconstructions of the coarse grid primal and adjoint variables on the fine embedded mesh. In regions where the remaining error is zero, but the adjoint correction not, no refinement is necessary, because the adjoint correction is sufficient. In regions where the remaining error is not zero a refinement criterion is evaluated based on the remaining error. In the remaining error term of Equation (6.11) the solution of the adjoint problem on the fine mesh $\hat{\phi}_{i,h}$ is required. However, it is not desirable to compute this solution, therefore $\hat{\phi}_h$ is reconstructed from the coarse grid solution using a reconstruction operator with a higher order than the operator used for $\phi^H_{i,h}$ and $\hat{\phi}^H_{i,h}$. This approach yields

$$\begin{aligned} \tilde{J}_h(\phi_{i,H}) =& J_h(\mathcal{L}^H_h \phi_{i,H}) - (\mathcal{L}^H_h \hat{\phi}_{i,H})^T \mathcal{R}_h(\mathcal{L}^H_h \phi_{i,H}) \\ & - (\mathcal{M}^H_h \hat{\phi}_{i,H} - \mathcal{L}^H_h \hat{\phi}_{i,H})^T \mathcal{R}_h(\mathcal{L}^H_h \phi_{i,H}) \end{aligned} \tag{6.12}$$

with $\mathcal{L}^H_h$ and $\mathcal{M}^H_h$ as reconstruction operators and $\mathcal{M}^H_h$ being of higher order than $\mathcal{L}^H_h$. A bound on the local error $\mathcal{E}_P$ in each cell P of the coarse mesh can be expressed as

$$\mathcal{E}_P = \left((\mathcal{M}^H_h \hat{\phi}_{i,H} - \mathcal{L}^H_h \hat{\phi}_{i,H})^T \mathcal{R}_h(\mathcal{L}^H_h \phi_{i,H}) \right)_P \tag{6.13}$$

and the total error E is the sum of the local errors $E = \sum_{P=1}^{N} \mathcal{E}_P$. A refinement criterion can then be derived by an user prescribed tolerance for the total error E in the scalar value of interest J.

Some details on the implementation of the goal-oriented grid-adaptation procedure are given in the following.

The evaluation of the residual of the coarse mesh solution on the fine grid can be done either on a cell by cell basis or for the complete grid at once. Performing the evaluation for the complete grid at once requires that the whole mesh is refined isotropically. For three-dimensional flows based on hexahedral cells the total number of cells is increased by a factor of eight for each refinement level. The required computer memory scales with the total number of cells. Therefore, this procedure demands large computational memory resources. For an evaluation on a cell by cell basis the cell in question is refined isotropically and the coarse grid solution is projected to the refined cell. However, the residual evaluation in the finite volume framework requires not only values from the cell in question. For a complete evaluation, values from three layers of neighbour cells on the refined grid are required as depicted in Figure 6.3. Hence, the cell in question, its neighbour cells and its neighbours neighbourhood need to be refined.

In both cases the evaluation of the residual depends on mass fluxes through the faces. Finite volume methods require conservative mass fluxes – the sum of the mass fluxes over all faces needs to be zero. However, when the coarse grid solution is projected onto the embedded fine grid, mass fluxes are no longer conservative. To achieve conservative mass fluxes the pressure correction equation is solved once and only the mass fluxes are updated. When the residual evaluation is performed for the whole grid at once, this operation has to be performed only one time. Within a cell by cell approach this operation needs to be conducted for each cell. Therefore, an equation system which covers the whole grid needs to be solved for each cell. The computational effort for such a procedure would be prohibitive. A feasible cell by cell approach requires an alternative local solution technique to achieve conservative mass fluxes for the cells in question. In the present work the residual evaluation for the embedded fine grid is performed for the whole grid, rather than using a cell by cell approach.

After the complete mesh is isotropically adapted, the mass fluxes are corrected once by solving the pressure correction equation and applying only the mass flux correction. The pressure correction equation employs the reciprocal central coefficient from the momentum equation. Therefore, the momentum equation is assembled but not solved. When the mass fluxes are corrected, the algebraic equation systems for the transport equations are assembled using the standard procedure and the residuals of the equation systems are evaluated. During that process no solution of the equation system is necessary. The high order prolongation operator $\mathcal{M}$ uses third order accurate simplex based interpolation while

The figure on the right hand side shows the coarse grid, where the residual has to be evaluated for the shaded cell on the embedded fine grid.

The cell is refined isotropically, to create the embedded fine grid. The necessary stencil size will only be displayed for one of the resulting cells, i.e. the shaded cell.

The evaluation of the pressure correction equation residual requires the reciprocal diagonal coefficients $\frac{1}{A_p^{u_i}}$ on the cell faces. These are determined by interpolation between the cells adjacent to the face, leading to additional refined cells indicated by the solid black lines.

The evaluation of $A_p^{u_i}$ in the neighbour cells requires in turn information from the faces forming the cells. Hence, additional refined neighbouring cells, as indicated by the black solid lines, occur.

The evaluation of the diffusion term, being part of $A_p^{u_i}$, requires the face gradient. Hence, the gradient in the cells adjacent to the faces need to be evaluated. The cell gradient evaluation in turn uses values from adjacent cells. Therefore, an additional third refinement is involved.

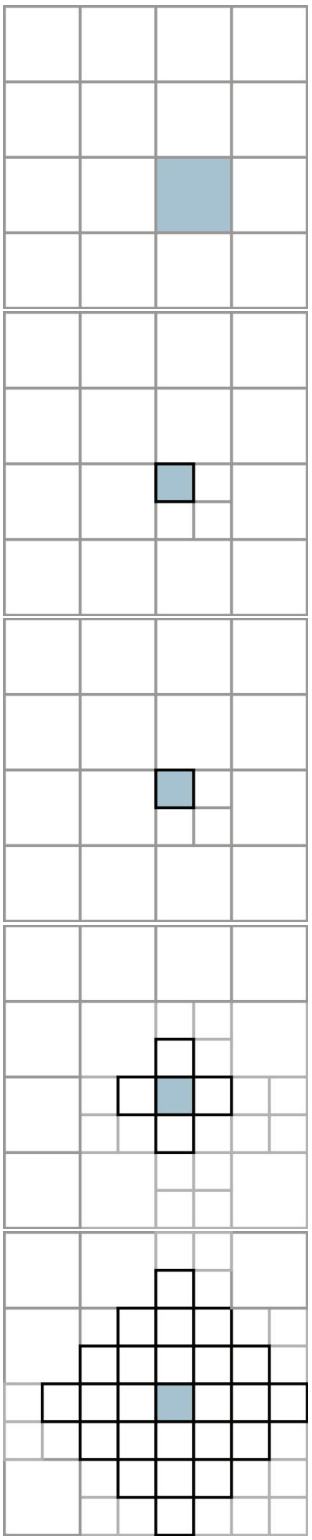
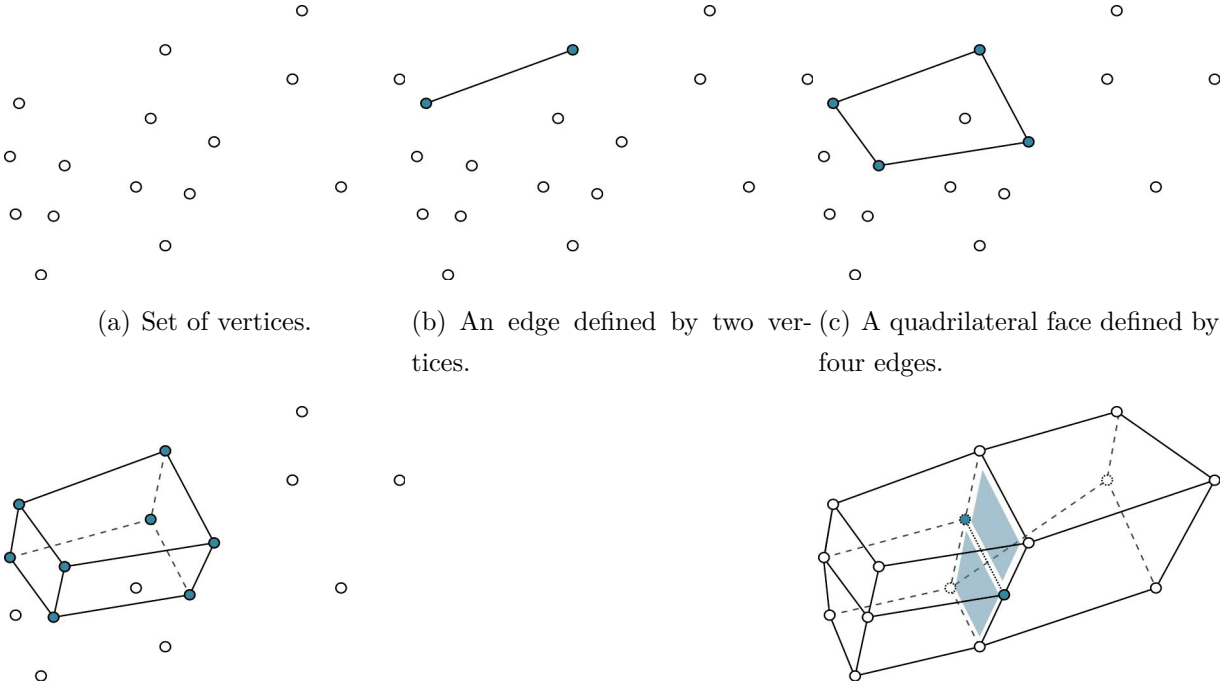


Figure 6.3: Illustration of required stencil size for the evaluation of the embedded fine grid residual.

the low order prolongation operator $\mathcal{L}$ is based on second-order simplex interpolation. A detailed description of simplex based interpolation is provided in Section 6.4.3. After the functional, the adjoint correction and the remaining error have been evaluated on the fine grid, the complete grid is coarsened. During that process the local remaining error of the corresponding fine cells are simply agglomerated for each coarse cell.

6.4 Refinement Algorithm

Prior to the description of the refinement algorithm a nomenclature for all relevant geometrical and topological entities is introduced and depicted in Figure 6.4. Each cell is



(a) Set of vertices.

(b) An edge defined by two vertices.

(c) A quadrilateral face defined by four edges.

(d) A hexahedral cell defined by six faces.

(e) Hanging Nodes (filled circles) and a side for the bigger cell consisting of 2 faces (shaded area).

Figure 6.4: Geometrical and topological entities.

bounded by its faces, which are built upon edges. Edges are defined by two vertices at their ends. A hexahedron is defined by eight corner vertices. These vertices will be called base vertices in the following. A hanging node is a vertex which is not present in the list

of base vertices of all hexahedrons adjacent to the vertex. A hexahedron has six sides. Within the unstructured grid environment each side may consist of one, two or four faces and each face of four to eight edges.

The refinement algorithm should provide a topological valid grid at all instants of time. Within this work a topologically valid grid is defined as a grid where each edge has at most one hanging node. Figure 6.5 depicts an example of an invalid hanging node.

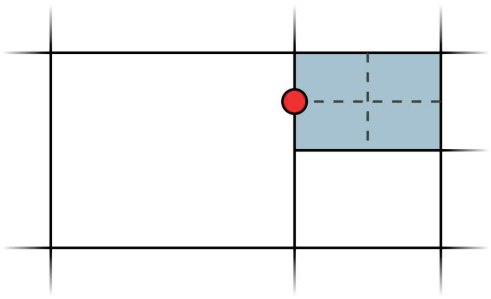


Figure 6.5: The refinement of the shaded cell would lead to an invalid (i.e. second) hanging node, as indicated by the red dot.

The refinement algorithm is composed of two building blocks:

- Propagation of refinement information
- Actual refinement

Details on all steps are described in the following sections. For the sake of clarity and improved understanding, illustrations are limited to two dimensions whenever possible.

6.4.1 Propagation of Refinement Information

After the refinement criterion has been evaluated, all cells are marked according to their refinement indicator value as described in Section 6.2. Subsequently it has to be verified that the marked cells lead to a valid grid. During this procedure, additional cells are marked for refinement until a valid grid can be constructed. Furthermore, the grid quality should not be deteriorated due to the refinement. Within the propagation step the following quality measures are considered:

- Irregularity

- Anisotropy
- Non-orthogonality
- Three-neighbour splits

To ensure grid validity at all instants of time the refinement sequence has to be prescribed. In the following sections a detailed description of the propagation step is provided.

a) Irregularity

After the cells have been marked for refinement all base vertices of a marked cell are flagged. A flagged hanging node in a cell definition indicates that one of the "smaller" adjacent cells is marked. If a common edge has to be split due to the marker of the "smaller" cell, an additonal split dividing the edge of the "larger" cell has to be added. Subsequently the corner vertices of this cell are flagged, unless they have been flagged before. The whole process is repeated until no further splits are required. Figure 6.6 depicts this process.

In case of isotropic refinement, splits in all cell directions are added if a flagged hanging node vertex is detected in a cell definition.

b) Anisotropy

Using only the irregularity criterion to preserve grid quality, it is likely to achieve pronounced anisotropic local refinement, which may have a bad effect on the solution quality. To suppress the pronounced anisotropy of the grid, the following rationale is used:

When two neighbouring cells are marked for refinement, but only one of the markers would split a common edge, the refinement splitting the common edge is propagated to the other cell. The depth of this propagation should be limited. By default it is limited to three cell layers.

c) Three-Neighbour Splits

Special care has to be taken when anisotropic refinement is used in three dimensional spaces. It may happen that a cell gets three neighbours on one of its sides due to the refinement. Within the present algorithm, such a situation is considered topologically invalid. To prevent this situation, the algorithm walks through all cells which have two

Shaded cells are marked for refinement, dashed lines indicate refinement direction

In a loop over the vertices of the smaller marked cell a vertex which is currently a hanging node (rhombus symbol) is found. The refinement is propagated to the adjacent cell.

Inspected vertices are marked (red circles). In a loop over the vertices of the newly marked cell, another hanging node is identified and the refinement is propagated further.

All vertices of marked cells have been processed. No further necessity for refinement propagation is detected.

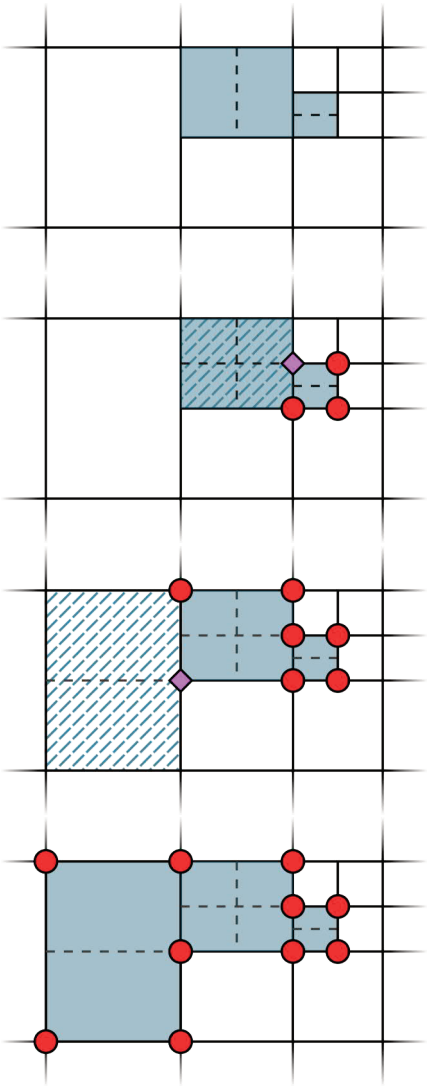


Figure 6.6: Illustration of the recursive refinement propagation process for the example of an anisotropic refinement to ensure a valid grid topology.

Marking before anisotropy check: Dashed lines indicate marked refinement direction.

Marking after anisotropy check:: Dashed lines with triangles at their ends indicate additional refinements due to anisotropy check.

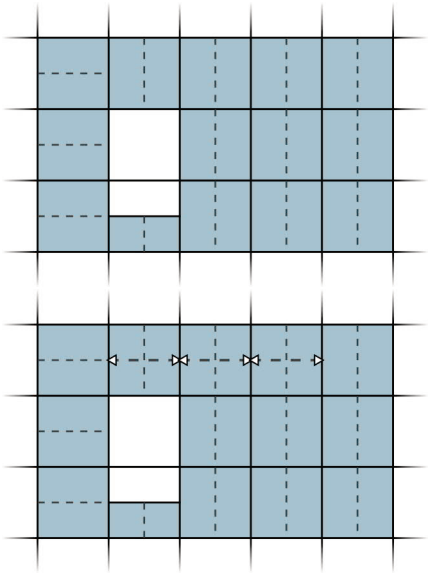


Figure 6.7: Illustration of grid quality enhancement due to the suppression of pronounced anisotropy.

neighbours on one of their sides. If the cell itself or one of its two neighbours are marked for refinement and a common edge of all three cells has to be split, a split that divides this edge is added to all three cells. The process is sketched in Figure 6.8. The figure shows a view on the common faces of the three cells. The shaded cells are marked for refinement and the dotted lines indicate split direction. On the left-hand side of Figure 6.8 two situations which lead to three-neighbour splits are shown. To avoid three-neighbour splits in both situations, all three cells need to be marked for refinement, as shown on the right-hand side of the figure.

d) Non-Orthogonality

An important aspect which influences the stability and accuracy of simulations is the non-orthogonality of the grid (c.f. Section 4.2). Especially anisotropic mesh adaptation can significantly reduce mesh orthogonality, which is measured as the angle between the face normal vector $\underline{\Delta F}$ and the connecting line of the two adjacent cell centres $\underline{d}$. During the refinement process the face normal vector will not change, however the location of the cell centres will change. The calculation of the non-orthogonality on the refined mesh requires the new cell centre positions, which are unfortunately not available at this stage of the

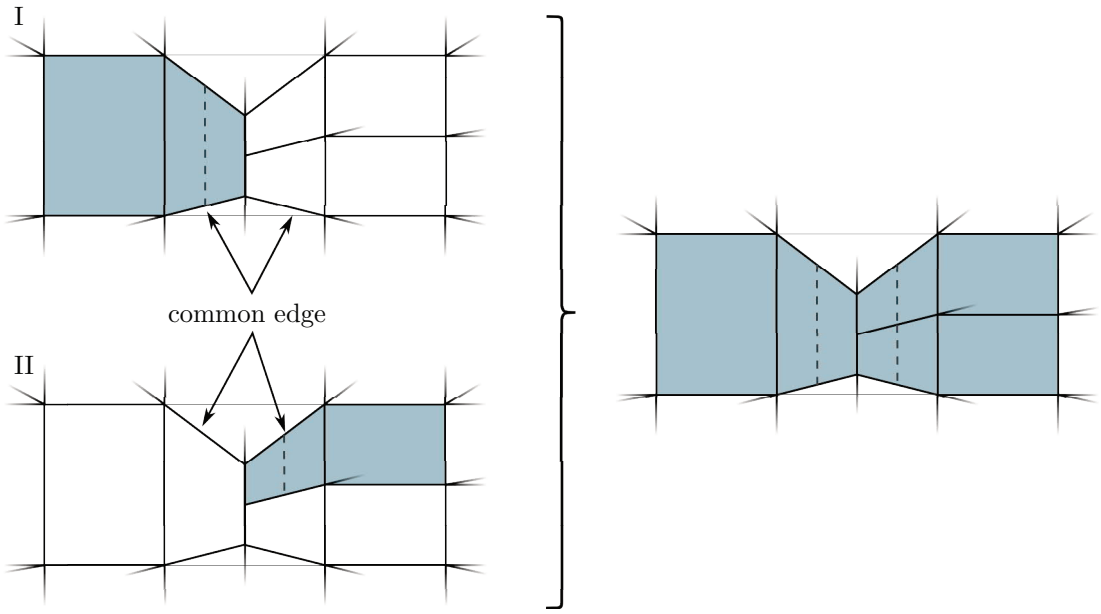


Figure 6.8: Avoidance of three-neighbour splits.

refinement process. Therefore, an approximate non-orthogonality measure based on the angle between the face normal and the connecting line between an intermediate point on the edge and the adjacent cell is used, as depicted in Figure 6.9. Within the figure, the angles α and β denote the exact and the approximated non-orthogonality. The new cell centres due to refined cells are marked by rhombic symbols while the original cell centre is marked by a circle. The intermediate points denoted by triangles are located at one and three quarters of the edge length, and serve as an approximation for the new face centres. Whenever a cell is flagged for refinement, the resulting non-orthogonality between the cell and all neighbours that share an edge of the cell which needs to be split is assessed. If the resulting non-orthogonality is below an user-prescribed threshold, the neighbouring cell will be flagged for refinement in the direction of the edge. The non-orthogonality check is recursively repeated for the neighbouring cell.

e) Refinement Sequence

Even if all cells are correctly marked for refinement, an invalid intermediate mesh can occur depending on the sequence in which the cells are refined. An example is shown in Figure 6.10.

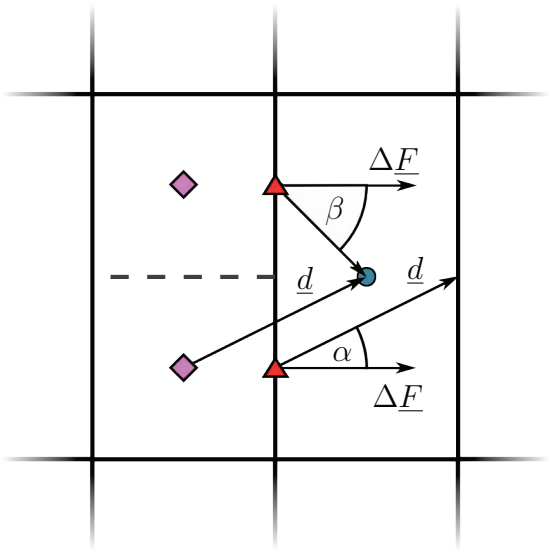


Figure 6.9: Orthogonality approximation.

Parts of the developed algorithm rely on topologically valid intermediate grids, therefore the sequence in which the cells have to be refined, has to be defined. Each cell which is marked for refinement will get a priority level. Cells with higher priority levels are refined prior to cells with lower priority levels. To determine the priority levels, the highest priority level of all adjacent cells is stored for each vertex. If a vertex of a cell, which is not a base vertex, is a hanging node vertex and has the same or higher priority level than the cell, the cell priority level has to be increased. The priority levels of the vertices

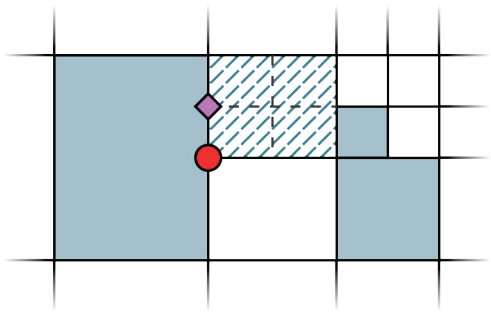


Figure 6.10: If the ruled cell would be refined first the invalid hanging node marked with a rhomb would be created on an intermediate mesh.

adjacent to the cells are updated, and the whole process is repeated until no more changes have to be applied. An illustration of this process is given in Figure 6.11

Initial priority levels for cells and adjacent vertices are shown. Cells with higher priority levels will be refined first based on the final priority levels.

Marked vertex is a hanging node vertex, which is not a base vertex for the ruled cell. The ruled cell and the adjacent marked cell have to be refined, but the priority of the ruled cell is not higher.

Marked vertex is a hanging node vertex, which is not a base vertex for the ruled cell. The ruled cell and the adjacent marked cell have to be refined, but the priority of the ruled cell is not higher.

Valid distribution of priority levels.

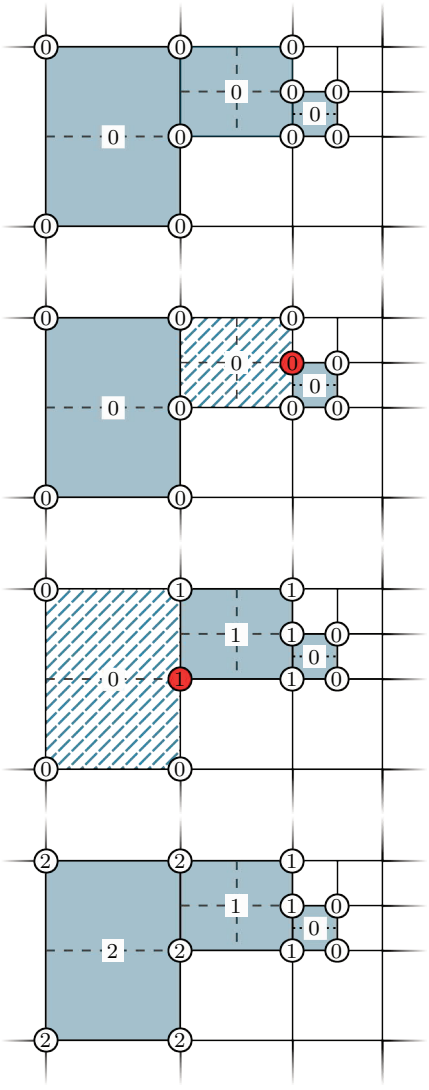


Figure 6.11: Illustration of determination of refinement sequence for anisotropic refinement to ensure a valid grid topology at any instant in time.

After completion of this step a valid grid is obtained from the marked cells and the associated priority levels.

f) Overview of Complete Refinement Propagation Algorithm

The complete sequence of the propagation step is shown in Algorithm 4. It consists of a

```
repeat
    for icell=1,nCells do
        add hanging node splits
        add buffer layer splits (recursive)
        add angle splits (recursive)
        add three neighbour splits
    end
until No more splits added
repeat
    for icell=1,nCells do
        check refinement priority
    end
until No changes in priority
```

Algorithm 4: Pseudo-code illustrating the sequence of the refinement propagation step.

loop over all cells, where for each cell all previously described mesh quality measures are handled. Due to performance reasons the propagation of additional refinement markers to suppress anisotropy and non-orthogonality are designed as recursive algorithms. The loop over all cells is repeated until no additional refinement markers have to be added. Subsequently the refinement sequence is determined. Therefore, a loop over all cells is executed, where the refinement sequence is determined for each cell. This loop is repeated until no more changes are applied to the refinement sequence.

6.4.2 Refinement Phase

When the refinement decision step is completed the actual refinement is initiated. Prior to the description of the refinement required dedicated data-structures are introduced in this subsection.

a) Data-Structures

For the refinement step the topological data set is transferred into an intermediate data set which contains a specialised data structure to handle the changing topology. Connectivities between different topological entities are stored in the compact row storage (CRS) format. It consists of two one-dimensional arrays. The first array (index array) stores an index, which indicates the position where the stored data set starts in the second array (data array). The principle is shown in Figure 6.12. The last start pointer refers to the end of the data array. It is a widely used memory efficient data structure. However, it is not efficient for data where the lengths of the stored data sets are changing. If the length of a data set that is stored at an arbitrary position of the data-array changes, all following entries have to be copied to other positions. This is an expensive operation, especially when the data set that changes is located somewhere in the beginning or middle of the data array and the data array is large.

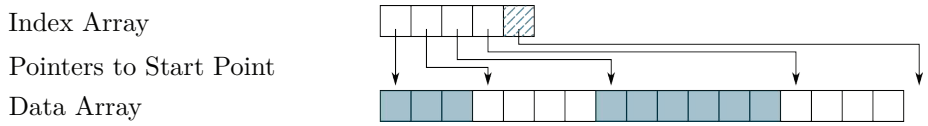


Figure 6.12: Illustration of the CRS data format.

A data structure that is designed for changing data set lengths is the linked list. It consists of a pointer pointing to the start index of the first data set and pointers in each data set pointing to the next data set. Therefore, it is possible to reallocate the sizes of the individual data sets, without performing any copy operations. However, even the reallocation operations are computationally expensive. Furthermore, linked lists supply no direct access to a specific element. Hence, a modified version of the compact row storage, which is efficient for changing data set sizes and at the same time provides direct access to specific data sets, has been designed for the present study.

The new data structure consists of three one-dimensional arrays, where the first one stores the start indices (start index array), the second one the end indices (end index array) and the third one the data (data array). When the array is initially filled void spaces are introduced between the different data sets. A schematic of the new data structure is depicted in Figure 6.13. When the length of a data set increases (e.g. from 6 entries to

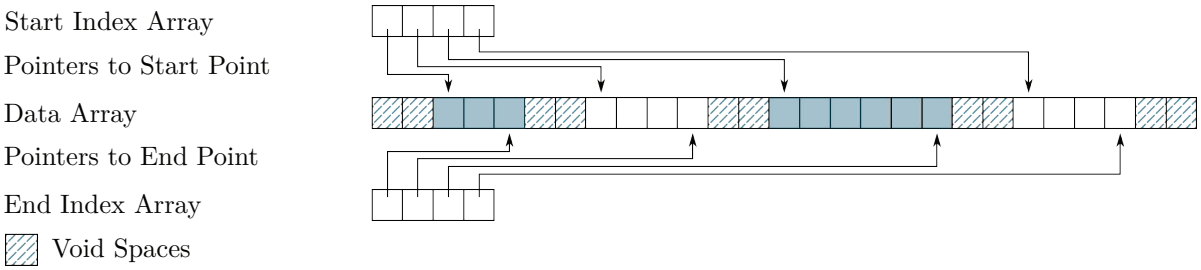


Figure 6.13: Illustration of the new data structure.

7 entries) the void spaces surrounding the data set are used to store the data set. If the data set is larger than the combined length of the original data set and the surrounding void spaces, the neighbouring data sets are moved, using their preceding and following void spaces, to create enough space. However, this copy operations are typically localised and therefore significantly less copy operations have to be executed than for the same data insertion in the compressed row storage data format. During the initialisation of the new data-structure a default void space of typically two voids is used. In case all voids are filled a new (larger) copy of the data-structure is created and all data is transferred to the new data structure before the old data structure is deleted.

The intermediate data structure contains the following information from the original data structure:

Vertex Locations Cartesian coordinates of vertices

Cell to Face Connectivity The neighbouring cells of a face

Face Family The assignment of faces to geometrical surfaces

Vertex to Cell Connectivity The base vertex definition for each cell

Face to Cell Connectivity The faces for each cell

Vertex to Face Connectivity The vertex definition for each face

During the refinement process all intermediate data structures have to be updated permanently, because the information stored in these structures is used within the refinement process.

Vertex and face indices for all cells that are marked for refinement and for their neighbouring cells are stored in two arrays for each cell. The first array contains 27 entries for the vertices. The vertex indices are stored in this array according to the CGNS HEXA_27 format [34, 148] shown in Figure 6.14. Indices of existing vertices are stored at the appropriate position in the array, while a zero is stored for all non-existing vertices. The second array consists of 24 entries for face indices in a format related to the CGNS HEXA_27 format. The face numbering is depicted in Figure 6.15. The face index array has four

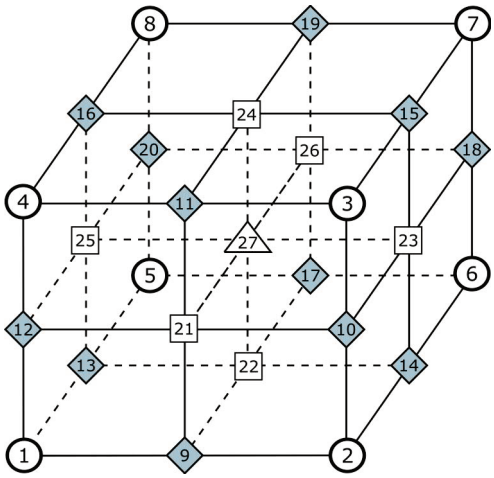


Figure 6.14: CGNS HEXA_27 notation.

entries for each side of a cell. For sides consisting of one face the face index is stored to all four entries associated to the side. If a side consists of more than one face, the face indices are stored at the appropriate positions of the face index array for this side. The appropriate positions are determined using a two-dimensional local coordinate system, which is defined for each side. The first principal axis points from the first face defined for this side to the second face defined for this side, according to the face definition shown in Figure 6.15. The second principal axes points from the first face to the third face. Accordingly, it can be determined if the side is split normal to the first and/or second principal axis of the side.

The vertex and face index array contain non-unique informations. Two adjacent cells that share one face have the same face index in their face index arrays identifying the shared face. If the shared face has to be split due to the refinement marker of one of the cells, the indices of newly created faces have to be stored at the appropriate locations of

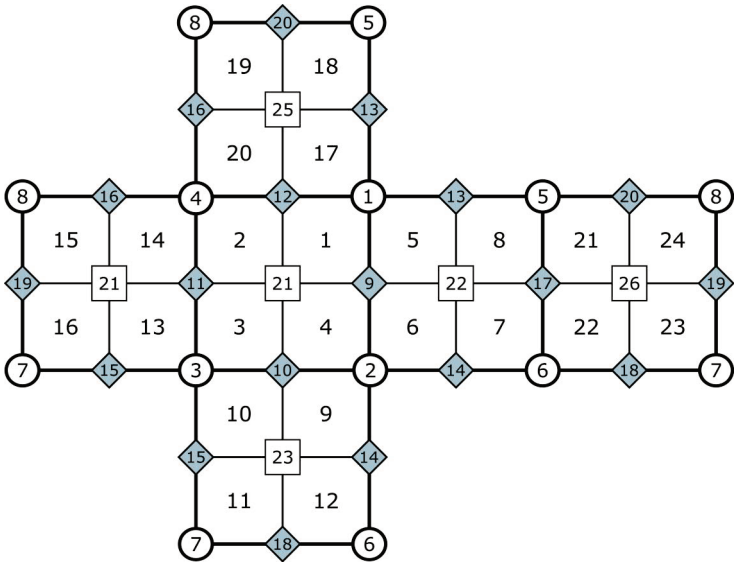


Figure 6.15: Developed view of a cube with the vertices in HEXA_27 notation and face numbers.

the face index arrays of both cells. The same holds for the vertex index arrays.

b) Refinement

The basic refinement algorithm is shown in Algorithm 5. It consists of a loop over the refinement priorities, starting at the highest priority and proceeding to the lowest priority. Within this loop the refinement of all cells, that have a refinement priority according to the priority level of the actual loop, are refined.

During all these steps it has to be ensured that the connectivity information for all cells is kept up to date. The individual steps will be explained the following.

During refinement new vertices are created where necessary and their indices are stored in the vertex array for the appropriate cells. Within this step, it has to be ensured that no coincident vertices are created. The algorithm walks through all cells and introduces new vertices for a cell, where necessary. When a new vertex is introduced for a cell, the new vertex is also communicated to the neighbouring cells and the vertex list for the neighbouring cell is updated.

Afterwards, faces that need to be split are modified. For the original face the vertex to face connectivity is manipulated to get the first new faces, and the additional new faces

```
for priorityLevel=maxPriorityLevel,0,-1 do
    for icell=1,nAllCells do
        if icell to be refined then
            if priorityLevel(icell)=priorityLevel then
                Store vertex and face indices of icell
                Store vertex and face indices of all neighbour cells of icell
                Create new vertices for icell
                Create new faces for icell
                Split icell
            end
        end
    end
end
```

Algorithm 5: Pseudo-code illustrating the basic refinement procedure

are added to the vertex-face connection list. At the same time additional information like the family, a face belongs to, or its cell neighbours are set. The new face information is then stored to the 24-entry face index array for each cell. Furthermore, new necessary faces inside the original cell are created. Afterwards the information in the face index array is communicated to the neighbouring cells.

Finally the new cells are created. The vertex to cell connectivity for the original cell is updated and the new vertex-cell connectivities are added to the list. Additionally, the face to cell connectivity is updated for the refined cell, the new cells and the neighbouring cells.

6.4.3 Post Refinement Phase

After the refinement step is completed, the intermediate data structure is transferred back to the standard data structure. All geometric entities are recomputed using the vertex positions and the topological information. Arrays depending on the grid size are reallocated.

Furthermore, the refined grid is aligned to the geometry and all field data is transferred

from the original cells to the new cells. Special care has to be taken to estimate the face fluxes on the refined grid. Details on the geometry alignment, prolongation operators and face flux adjustment are presented in the following sections.

a) Geometry Alignment

Refining cells adjacent to a curved geometrical boundary usually leads to new vertices which are not located on the boundary surface because new vertices are always introduced in the middle of the edge that has been split. However, a precise representation of the boundary surface is crucial to achieve accurate results and the locations of vertices, which are not located on the boundary, need to be corrected. To locate the new vertices on the geometrical boundary, informations about the geometrical boundary need to be available. The developed approach uses an STL discretisation of the geometrical boundary. The STL discretisation consists of conforming triangles, where all vertices of the triangles are located on the curved boundary. Furthermore, for each triangle the associated geometrical family is available.

For the geometry alignment first the triangle of the STL surface discretisation, which has the same family as the vertex to be aligned and is located closest to this vertex, needs to be found. For an efficient determination of the nearest triangle a kd-tree based search algorithm is employed. A separate kd-tree for each family of the STL-discretised surface is used to ensure that the family associated to the vertex and the surface discretisation match. However, the nearest triangle of the STL discretisation is not necessarily the best approximation as depicted in Figure 6.16 (assuming that the whole boundary has the same family). Therefore, a second criterion regards the angle between the outward pointing facet and the surface normal. If the angle is larger than a prescribed threshold (40° is used in the developed approach) the result of the kd-tree search is deemed to be wrong and a geometric search algorithm is used. The geometric search algorithm walks through all triangles of the STL discretisation which have the same family as the vertex. It finds the triangle which is located closest to the vertex and fulfils the above described angle criterion at the same time. In parallel simulations the whole STL surface boundary discretisation is available on all processes.

Figure 6.17(a) depicts a refined grid where the new boundary vertices are not aligned with the boundary surface. Just aligning the boundary vertices to the boundary surface would lead to cells with poor grid quality or even negative volume. Hence, the adjustment

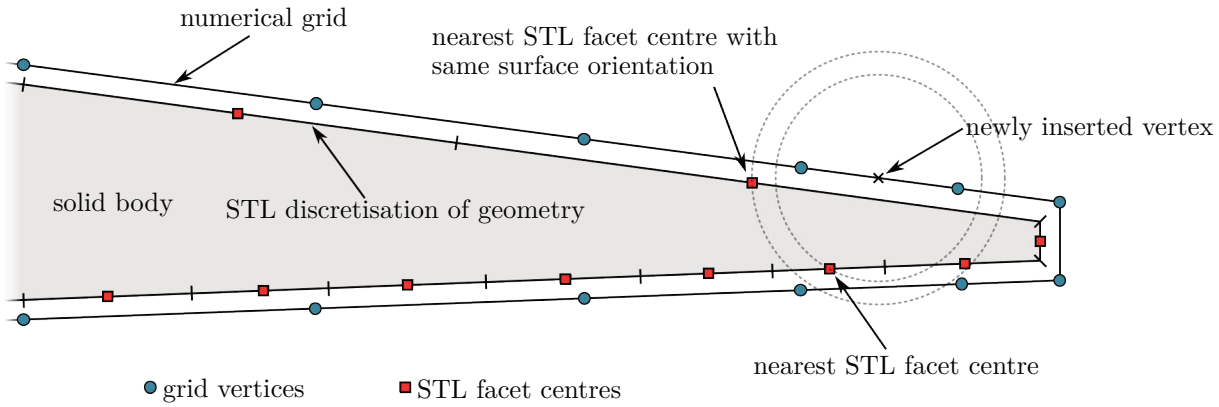


Figure 6.16: 2D example of geometry alignment for vertices.

of the boundary vertices needs to be propagated into the volume mesh.

The propagation of the boundary vertex displacement into the volume mesh uses a front marching algorithm. The vertex displacements are propagated along edges that are oriented approximately normal to the boundary surface. Four different statuses are used to classify vertices during this process:

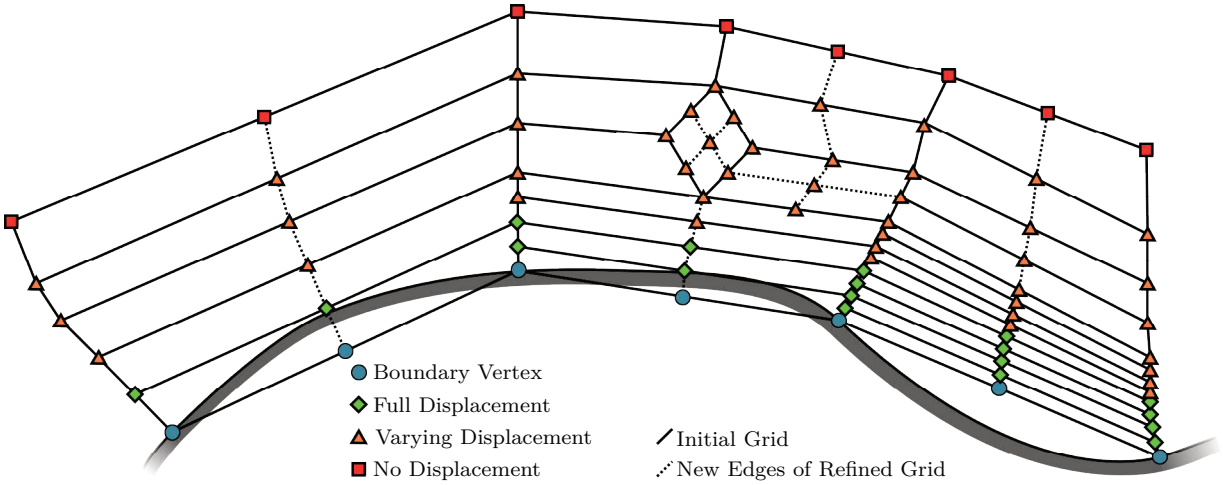
source vertices that serve as source for a displacement

target vertices that will receive the displacement from the source vertices

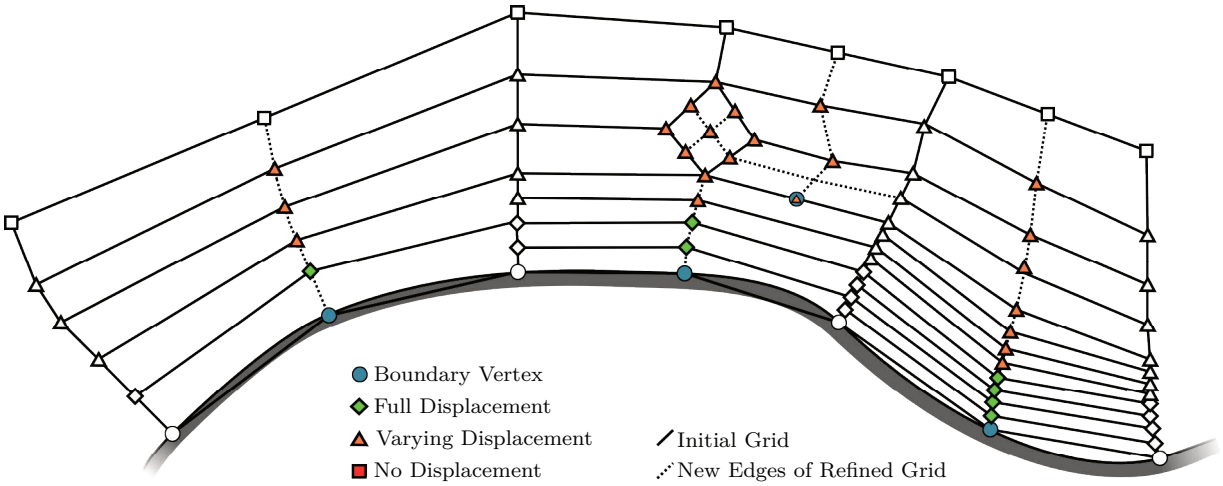
finished vertices for which the final position is reached

pending vertices for which the displacement operation is not yet finished

Initially all vertices are marked as pending and the boundary vertices are aligned with the geometry. Afterwards all vertices that currently have a prescribed displacement are marked as sources. Vertices are marked as target, if they are part of an edge definition with the other vertex marked as source and the angle between the surface normal associated to the source vertex and the edge is smaller than a prescribed threshold. When all vertices are marked, the displacements are propagated from the source to the target vertices. Subsequently the source vertices are marked as finished and the target vertices as source vertices. Pending vertices which are part of an edge definition with a source vertex are marked as target vertex, if the angle between the edge and the source edge is below a prescribed threshold. The whole process is repeated until all vertices are marked



(a) Grid after refinement before geometry adaptation. Symbols denote status of vertices.



(b) Grid after refinement and geometry adaptation. Vertices with filled symbols have been moved.

Figure 6.17: Illustrative example of geometry alignment.

as finished. For parallel simulations the vertex displacements and vertex statuses are communicated over all processes after each step of the front marching algorithm.

In unstructured meshes vertices may get displacement contributions from more than one edge. Hence, the displacements and source edge directions are accumulated for each vertex and finally normalised by the total number of displacements each vertex received. Furthermore, hanging node vertices may receive displacement contributions from target vertices. Usually hanging node vertices are located on the edge for which they present the hanging node vertex. Therefore, displacement contributions for hanging node vertices

are initially ignored. Their position is projected to the edge, for which they present the hanging node vertex, if any of the base-vertices of the edge is modified.

The developed boundary alignment procedure deploys a zonal approach to achieve a good grid quality even in boundary layer meshes and minimise the associated computational effort. The zones are determined using the minimal wall distance of the vertices and characterised by a zonal weight factor. The vertex displacement is multiplied by the zonal weight factor. Two thresholds, defined by the user, serve as boundary definition for the different zones. All vertices that have a minimal wall distance lower than y_{rigid} receive the full displacement of their respective source vertices, hence the weight factor for all vertices in this region is set to unity. All vertices with a larger wall distance than y_{fixed} get a displacement of zero – the weight factor is set to zero. For all vertices in between the thresholds the zonal weight factor is determined by a cubic blending function that ensures a smooth transition between the rigidly moving zone and the fixed zone.

The complete geometry alignment algorithm is depicted in Algorithm 6.

b) Prolongation Operators

After the grid is refined, the primal variables need to be transferred from the original grid to the refined grid. Within the developed approach a number of different prolongation operators, which differ in their order and complexity, are available.

The most obvious prolongation operator is to transfer the cell centre values of the original coarse cell to the cell centres of the nested refined cells via

$$\phi_{P^*} = \phi_P \tag{6.14}$$

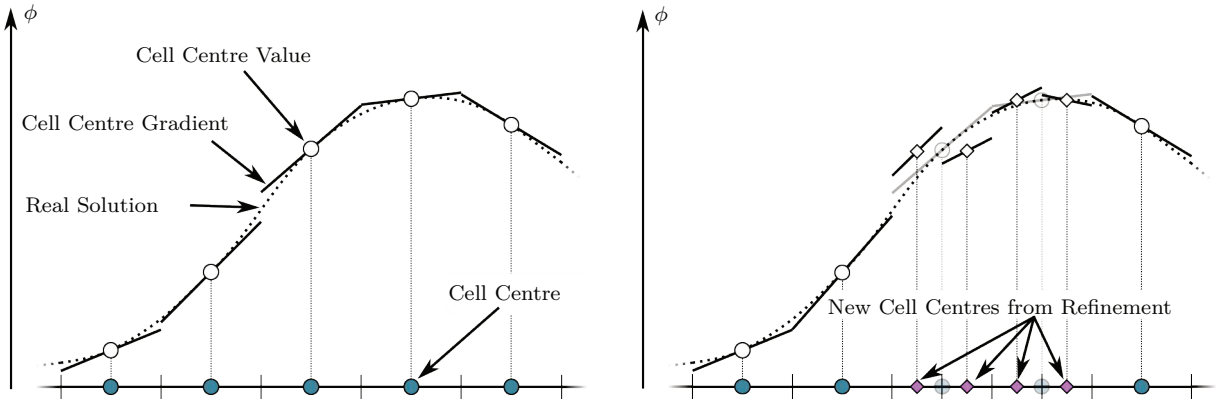
with P^* denoting the cell centre of the refined cells and P the cell centre of the original cell. This approach is easy to implement, however it deteriorates the formal second-order accuracy of the finite volume approach. Any smooth profile will be turned into a step profile. To obtain a smooth profile, additional iterations are necessary. A one-dimensional example of this mapping algorithm is shown in Figure 6.18.

A prolongation operator which is second-order accurate can be assembled using the original cell centre values and gradients as input. The new cell centre values are the determined as

$$\phi_{P^*} = \phi_P + (x_{i,P} - x_{i,P^*}) \frac{\partial \phi}{\partial x_i}. \tag{6.15}$$

```
for iBoundaryVertex,nBoundaryVertices do
    project vertex on boundary surface
    if Displacement of Projection != 0 then
        flag vertex as source vertex
    end
end
mark all vertices with wall distance ¿ y_fixed as finished
set nDisplacements for all vertices to zero
repeat
    for iEdge,nEdges do
        if iEdge contains source vertex then
            mark other edge vertex as target vertex
            if angle between edge and wall normal ¡ angleThreshold then
                add source vertex displacement to target vertex
                increase nDisplacements for target vertex by one
                mark target vertex as modified
                mark source vertex as finished
            end
        end
    end
    for iVert,nAllVert do
        if vertex is marked as modified then
            divide displacement by nDisplacments and multiply by zonal weight factor
            mark vertex as source vertex
        end
        if vertex is marked as target vertex then
            mark vertex as finished
        end
        if vertex is hanging node vertex then
            Adjust diplacment if necessary
        end
    end
until all vertices marked as finished
multiply target vertex displacement by zonal weight factor
```

Algorithm 6: Pseudo-code illustrating geometry alignment process.



(a) Initial one-dimensional grid and solution.

(b) Refined one-dimensional grid. Simple solution mapping

Figure 6.18: One-dimensional example simple prolongation operator.

Although this approach is more accurate, it is unbounded and prone to produce new local extrema as shown in Figure 6.19. Dealing with flow variables that are bounded, e.g.

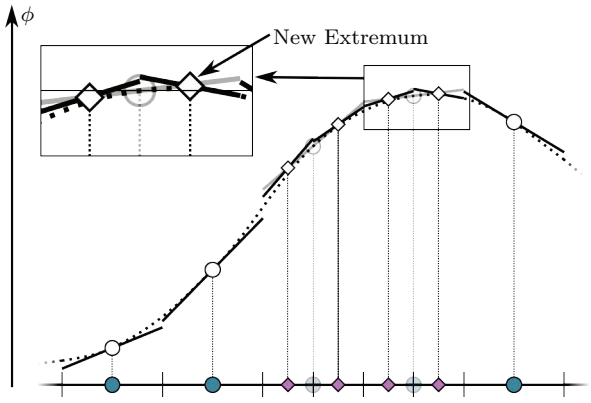


Figure 6.19: One dimensional example of gradient extrapolation prolongation operator.

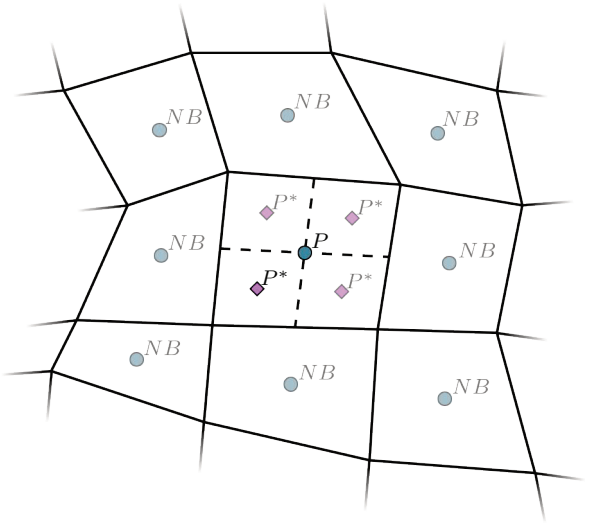
volume mixture fraction c, non-physical solutions are introduced due to the unbounded prolongation operator. Hence, either a bounded prolongation operator has to be used for these flow variables or the variables need to be corrected after prolongation. Furthermore, the new extrema can cause oscillations, especially when the grid refinement procedure is used frequently.

A second-order accurate prolongation operator that is bounded, is based on interpo-

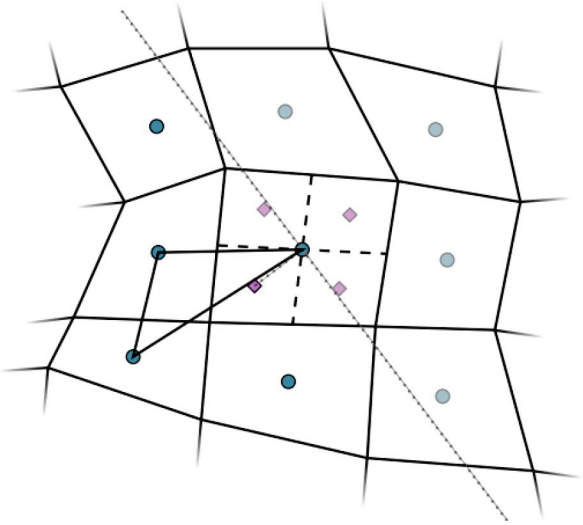
lation inside simplexes. The algebraic interpolation operator for simplexes has a low computational effort. However, the operator requires that the target point to which the solution is prolonged is located inside the simplex. The base points of the simplex need to be cell centres of the original grid. To achieve a good accuracy these base points should be located close to the target point. In an unstructured grid environment such simplexes are not necessarily available. Therefore, they have to be determined for each point to which the solution has to be prolonged. Within the grid refinement procedure the prolongation usually only needs to be performed for a small sub-set of the complete grid. Hence, the required simplexes are determined ad-hoc to reduce the computational effort. The first base point of which the simplex is composed is always the cell centre of the original cell that has been refined. To find the other base points of the simplex, all neighbouring cell centres (named NB in Figure 6.20(a)) are stored in a list. The entries are ordered according to their projection of the normalised connection vector from original cell centre to the refined cell centre onto the connection vector from the original cell centre onto the neighbour cell centre. The list only includes neighbour cell centres with a positive projection. The determination of the simplexes is a trial and error method. Simplexes are created from the original cell centre and the neighbour cell centres depending on their position in the list. The probability, that a simplex composed of the neighbouring cell centres with the largest projections, contains the new cell centre is high. Accordingly, the neighbouring cell centres with the largest projections are preferred in the list. If the new cell centre is located inside the simplex, the required simplex has been determined (example shown in Figure 6.20(c)). Otherwise (exemplary shown in Figure 6.20(b)) a new simplex using the next value in the list is created and it is checked if the new cell centre is located inside the simplex. This procedure is repeated until a valid simplex has been found.

To prolong the field data to the new cell centres the barycentric coordinate system of the simplex is used. The barycentric coordinate system for a three-dimensional simplex is depicted in Figure 6.21. Therefore, the first vertex $P1$ is mapped to the barycentric coordinate $\lambda = (1, 0, 0, 0)^T$, the second vertex $P2$ to the barycentric coordinate $\lambda = (0, 1, 0, 0)^T$ and the other vertices are mapped accordingly. The transformation from the Cartesian reference frame to the barycentric coordinates λ_1, λ_2 and λ_3 reads as follows:

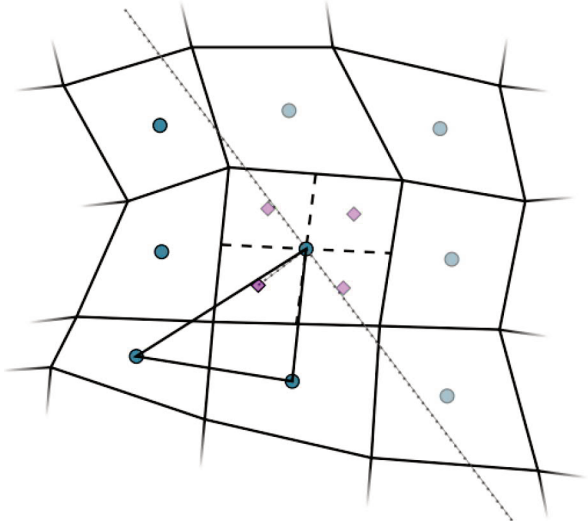
$$\begin{aligned} \lambda_i =& T_{ij}^{-1}(x_{j,P} - x_{j,P_4}) \quad \text{with} \\ T_{ij} =& (x_{i,P_1} - x_{i,P_4}; x_{i,P_2} - x_{i,P_4}; x_{i,P_3} - x_{i,P_4}) \end{aligned} \tag{6.16}$$



(a) Two-dimensional grid with refined centre cell.



(b) First Simplex candidate. New cell centre not inside simplex.



(c) Second Simplex candidate. New cell centre inside simplex.

Figure 6.20: Two-dimensional example of ad-hoc simplex determination.

λ_4 is determined via the compatibility condition

$$\lambda_4 = 1 - \lambda_1 - \lambda_2 - \lambda_3. \tag{6.17}$$

Since the barycentric coordinates are a normalised partition of unity (6.17), they can be

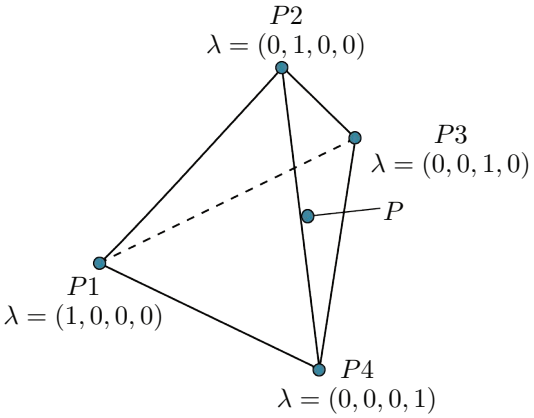


Figure 6.21: Barycentric coordinate system of a three-dimensional simplex.

used as interpolation weights and the prolongation to point P reads

$$\phi_P = \lambda_1 \phi_{P_1} + \lambda_2 \phi_{P_2} + \lambda_3 \phi_{P_3} + \lambda_4 \phi_{P_4}. \tag{6.18}$$

An one-dimensional example of the second-order accurate simplex-based prolongation operator used within the grid refinement procedure is depicted in Figure 6.22. Comparing the resulting prolonged values to the gradient extrapolation operator depicted in Figure 6.19 it is obvious that no new extrema are create using the simplex-based prolongation operator.

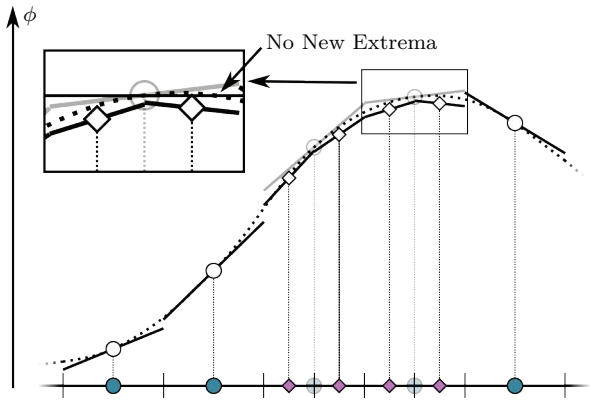


Figure 6.22: One dimensional example of the 2nd order accurate simplex-based prolongation operator.

A third order accurate prolongation operator can be build upon the basis of the second-order accurate simplex-based prolongation operator. It does not only utilise the cell centre values but also the gradients at the cell centres. Valid simplexes are determined as described above. The prolongation operator is composed of the second-order accurate simplex interpolation and additional terms. It reads

$$\begin{aligned}\phi_P = & \underbrace{\lambda_1\phi_{P_1} + \lambda_2\phi_{P_2} + \lambda_3\phi_{P_3} + \lambda_4\phi_{P_4}}_{\text{Second-order accurate simplex interpolation}} \\ & + 4[dA\lambda_2 + dB\lambda_3 + dC\lambda_4 - dA\lambda_2^2 - dB\lambda_3^2 - dC\lambda_4^2 \\ & + (dD - dB - dA)\,\lambda_2\lambda_3 \\ & + (dE - dC - dA)\,\lambda_2\lambda_4 \\ & + (dF - dC - dB)\,\lambda_3\lambda_4]\end{aligned} \tag{6.19}$$

with

$$\begin{aligned}dA =& \frac{1}{8}\left[\frac{\partial\phi_{P_1}}{\partial x_i}\left(x_{i,P_2} - x_{i,P_1}\right) - \frac{\partial\phi_{P_2}}{\partial x_i}\left(x_{i,P_2} - x_{i,P_1}\right)\right] \\ dB =& \frac{1}{8}\left[\frac{\partial\phi_{P_1}}{\partial x_i}\left(x_{i,P_3} - x_{i,P_1}\right) - \frac{\partial\phi_{P_3}}{\partial x_i}\left(x_{i,P_3} - x_{i,P_1}\right)\right] \\ dC =& \frac{1}{8}\left[\frac{\partial\phi_{P_1}}{\partial x_i}\left(x_{i,P_4} - x_{i,P_1}\right) - \frac{\partial\phi_{P_4}}{\partial x_i}\left(x_{i,P_4} - x_{i,P_1}\right)\right] \\ dD =& \frac{1}{8}\left[\frac{\partial\phi_{P_2}}{\partial x_i}\left(x_{i,P_3} - x_{i,P_2}\right) - \frac{\partial\phi_{P_3}}{\partial x_i}\left(x_{i,P_3} - x_{i,P_2}\right)\right] \\ dE =& \frac{1}{8}\left[\frac{\partial\phi_{P_2}}{\partial x_i}\left(x_{i,P_4} - x_{i,P_2}\right) - \frac{\partial\phi_{P_4}}{\partial x_i}\left(x_{i,P_4} - x_{i,P_2}\right)\right] \\ dF =& \frac{1}{8}\left[\frac{\partial\phi_{P_3}}{\partial x_i}\left(x_{i,P_4} - x_{i,P_3}\right) - \frac{\partial\phi_{P_4}}{\partial x_i}\left(x_{i,P_4} - x_{i,P_3}\right)\right].\end{aligned} \tag{6.20}$$

c) Face Flux Adjustment

To transfer the fluxes from the old grid to the new grid, the pressure correction equation is solved once and the fluxes are updated. As the pressure correction equation depends on the matrix coefficients of the momentum equation, the momentum equation is assembled – not solved – before.

6.4.4 Parallelisation

The ghost cells for an exemplary domain decomposition of a two-dimensional grid distributed over two processes are shown in Figure 6.23. For the ghost cells the complete topological information is available. The refinement criterion is evaluated similar to serial

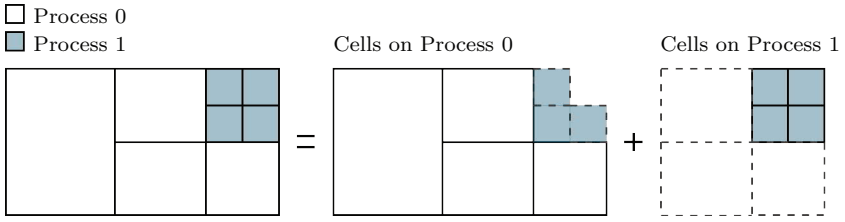


Figure 6.23: Illustration of domain decomposition with 2 processes. Cells with dotted lines are ghost cells.

computations. It is evaluated only for the internal cells. Within the refinement decision step only internal cells get additional markers. However, the decision can depend on values of ghost cells. At each loop of the refinement decision step, the markers are exchanged between the processes. Again the loops are executed until nothing changes any more. During the refinement step the refinement is done for marked internal and ghost cells. However, some information might be missing after that step, as illustrated in Figure 6.24. The considered cell in the figure is marked for refinement on process 1. After the refinement, process 0 misses the information about the hanging nodes, which are indicated by the closed dots. This information is gathered for the ghost cells after the refinement is completed and the ghost cells are updated with the missing information. During the transfer from the intermediate to the new data structure some cells, created by refining ghost-cells, have to be removed as it is shown in Figure 6.25. The considered cells, an internal cell on process 0 and a ghost cell on process 1, are marked for refinement. After refinement the cells marked by a red cross are superfluous ghost cells for process 1. Finally the exchange pointers, on which the mechanism to exchange data between partitions is based, have to be rebuilt.

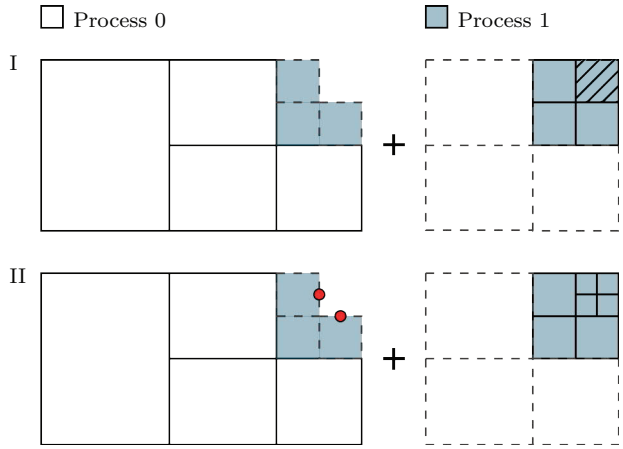


Figure 6.24: Example to show which information is missing after parallel refinement.

Load Balancing

The optimal performance of a parallel simulation hinges on a proper load balancing. Within the employed SPMD approach the whole computational grid is split into as many parts as processors are used. The computational load mainly scales with the number of unknowns, therefore a balanced load is achieved when the number of unknowns is equal for each part.

The dynamic grid-adaptation procedure refines the grid locally. Hence, the number of unknowns in each part of the grid may change. To maintain a balanced load, cells have to be redistributed over the processes. The employed dynamic load balancing algorithm uses the ParMetis library to determine which cell belongs to which processor. Whenever the grid is refined in a parallel simulation the load balancing algorithm is used after the actual grid refinement. After it has been determined to which processor each cell belongs the cells are redistributed over the processes. The redistribution scheme has been developed by D. Hafermann and is described in detail in [67].

6.4.5 Concluding Remarks

The present goal-oriented grid-adaptation technique is based on the adjoint flow solver developed by Stück [171]. The adjoint flow solver uses the *frozen turbulence* approach which is the most commonly used technique, where the variation of the turbulence trans-

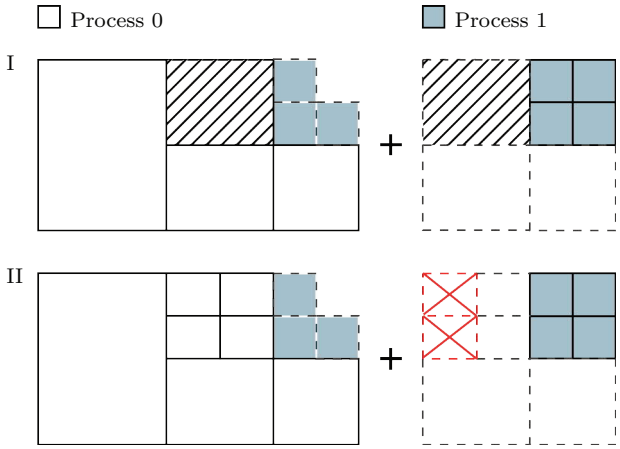


Figure 6.25: Exemplary update of ghost cells after refinement.

port equation is neglected. This leads to an inaccuracy of the adjoint solution of turbulent flows. Peter and Dwight [144] reviewed adjoint techniques for optimisation purposes. Several authors analysed the influence of the *frozen turbulence* approach on the accuracy of the results. However, due to the complexity and stability issues associated to adjoint turbulence transport equations, these examples are limited. However, it has been shown that the deviations in the gradients have only a minor impact on the optimisation result, when robust optimisation techniques are used. The derivation of the gradients is based on the adjoint solution, which also forms the basis for the adjoint error correction procedure and hence the refinement indicators for goal-oriented mesh refinement. Accordingly, a large uncertainty remains in the predicted adjoint correction terms and refinement indicators for goal-oriented refinement for turbulent flows with the present grid-adaptation technique.

7 Verification and Validation

The purpose and definition on the verification and validation according to the AIAA definitions [1] reads:

Verification The process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution to the model.

Validation The process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model.

All developed methods have been intensively verified and validated in this sense. Accordingly, the techniques are assessed individually. This chapter provides the description and results of selected verification and validation test-cases for the developed techniques.

The chapter starts with a brief description of some verification tools that have been used with the verification and validation and proceeds with the description of selected examples for each developed technique.

7.1 Verification Tools

To verify the developed code some features have to be assessed in detail. Tools for this purpose are presented within this section.

7.1.1 Spatial Order of Convergence

For the verification of the developed code it is necessary to assess the spatial order of convergence and the uncertainty in the predicted results. A convenient technique to assess the spatial order of convergence of numerical simulations in the finite volume context is

inhered in the Richardson extrapolation [155]. When the simulation results with refined grids converge asymptotically, Richardson extrapolation can be used to extrapolate the result to an infinitely fine grid and to provide an uncertainty contained in the result. Richardson extrapolation is usually based on solutions obtained for three successively refined grids. The technique has been enhanced by Eça and Hoekstra to be applied for solutions obtained from more than three grids. In the following only the procedure with the associated formulas is presented. The theoretical background is described in more detail by Eça and Hoekstra in [50]. The observed order of accuracy and convergence condition is estimated by a least square technique. The following function

$$S(\phi_0, a, p) = \sqrt{\sum_{i=1}^{n_g} [\phi_i - (\phi_0 + a h_i^p)]^2}, \tag{7.1}$$

is minimised in a least squares sense. Within the function ϕ_i is the numerical solution of any local or integral scalar quantity on a given grid, ϕ_0 the estimated exact solution, a a constant, p the observed order of accuracy, h_i a parameter identifying the representative grid cell size and n_g the number of available solutions from different grids. The above given formula yields the same results for $n_g = 3$ as the standard definition of the Richardson extrapolation. Richardson extrapolation is only applicable for monotonic convergence behaviour. Hence, the convergence behaviour has to be identified. When only solutions on three systematically refined grids are available the convergence behaviour is identified by

$$R = \frac{\phi_2 - \phi_1}{\phi_3 - \phi_2} \tag{7.2}$$

where the subscript $_1$ identifies the finest grid, $_2$ the medium grid and $_3$ the coarsest grid. Depending on R the following convergence behaviour is found:

$$\begin{aligned} 0 < R < 1 \quad &\rightarrow \quad \text{Monotonic convergence} \\ -1 < R < 0 \quad &\rightarrow \quad \text{Oscillatory convergence} \\ R > 1 \quad &\rightarrow \quad \text{Monotonic divergence} \\ R < -1 \quad &\rightarrow \quad \text{Oscillatory divergence} \end{aligned} \tag{7.3}$$

When solutions on more grids are available, the above given method is not applicable. To identify the convergence behaviour, the occurrence n_{ch} for a sign change of the difference between consecutive solutions $[(\phi_{i+1} - \phi_i)(\phi_i - \phi_{i-1}) < 0]$ is counted. The convergence

behaviour is classified using the estimated order of convergence p and n_{ch} via:

$$\begin{aligned}
&1. \quad p > 0 \quad \rightarrow \quad \text{Monotonic convergence} \\
&2. \quad p < 0 \quad \rightarrow \quad \text{Monotonic divergence} \\
&3. \quad n_{ch} \geq INT\left(\frac{n_g}{3}\right) \quad \rightarrow \quad \text{Oscillatory convergence or divergence} \\
&4. \quad \text{Otherwise} \quad \rightarrow \quad \text{Unknown}
\end{aligned} \tag{7.4}$$

The method is not able to distinguish between convergence and divergence in case of oscillatory behaviour. However, Richardson extrapolation is only applicable for convergent behaviour, hence according to [50] only the following classifications are relevant:

1. Monotonic convergence

2. Oscillatory behaviour

3. Anomalous behaviour

When the convergence behaviour is classified, the uncertainty in the results can be estimated. In case of monotonic convergence the uncertainty U_ϕ is estimated from

$$\mathrm{U}_\phi = \begin{cases} \min\left(A, 1.25\,\min\left(1.6, \frac{2.28}{p} - 1.4\right) B\right) & \text{for} \quad 0 \leq p < 0.95 \\ 1.25A & \text{for} \quad 0.95 \leq p < 2.05 \\ \max\left(A, 1.25\,\min\left(1.6, 3p - 5.15\right) C\right) & \text{for} \quad p \geq 2.05 \end{cases} \tag{7.5}$$

with

$$\begin{aligned} A &= \delta_{RE} + U_S \\ B &= \delta_{RE}^{12} + U_S^{12} \\ C &= \delta_{RE}^{02} + U_S^{02}. \end{aligned} \tag{7.6}$$

Herein δ_{RE}, δ_{RE}^{12}, δ_{RE}^{02}, U_S, U_S^{12} and U_S^{02} are defined as follows:

$$\begin{aligned}
\delta_{RE} =& \phi_i - \phi_0 = \alpha h_i^p \quad \text{with} \quad \alpha = const \\
\delta_{RE}^{12} =& \phi_i - \phi_0 = \alpha_1 h_i + \alpha_2 h_i^2 \quad \text{with} \quad \alpha_1 = const,\ \alpha_2 = const \\
\delta_{RE}^{02} =& \phi_i - \phi_0 = \lambda_1 h_i^2 \quad \text{with} \quad \lambda_1 = const \\
U_S =& \sqrt{\frac{\sum_{i=1}^{n_g} \left(\phi_i - (\phi_0 + \alpha h_i^p)\right)^2}{n_g - 3}} \\
U_S^{12} =& \sqrt{\frac{\sum_{i=1}^{n_g} \left(\phi_i - (\phi_0 + \alpha_1 h_i + \alpha_2 h_i^2)\right)^2}{n_g - 3}} \\
U_S^{02} =& \sqrt{\frac{\sum_{i=1}^{n_g} \left(\phi_i - (\phi_0 + \lambda_1 h_i^2)\right)^2}{n_g - 3}}
\end{aligned} \tag{7.7}$$

In case of oscillatory convergence the uncertainty is estimated via

$$\begin{aligned}
U_\phi &= 3\Delta_M \quad \text{with} \\
\Delta_M &= \max\left(|\phi_i - \phi_j|\right) \quad \text{with} \quad 1 \leq i \leq n_g \wedge 1 \leq j \leq n_g.
\end{aligned} \tag{7.8}$$

For all other cases the uncertainty is determined as

$$U_\phi = \min\left(3\Delta_M, 3B\right). \tag{7.9}$$

7.1.2 Approximation of Interface Sharpness

The approximation of the interface sharpness is based on the ratio of the present interface thickness to the optimal interface thickness. As stated in Section 5.3 an interface is blurred if more than one partly filled control volume occurs between cells completely filled with different phases. Hence, the maximum interface thickness for a perfectly sharp interface extents over three cell layers. The interface sharpness is estimated by means of the interface sharpness index S. The building blocks for this estimate are the approximation of the present interface thickness and the optimal interface thickness at a given location of the interface. These serve as basis for the local interface sharpness. Furthermore, a suitable normalisation of the local interface sharpness to judge the interface sharpness in the entire computational domain is presented.

The local interface thickness $\tilde{t}$ at a face F is estimated by

$$\tilde{t}_F = \begin{cases} ||\eta_i||_F^{-1} & \text{if} \quad c_L < 0.5 \leq c_R \quad \text{or} \quad c_R \leq 0.5 < c_L \\ 0 & \text{otherwise} \end{cases} \tag{7.10}$$

with $\eta_{i,F} = \left(\frac{\partial c}{\partial xi}\right)_F$ being the mixture fraction gradient, interpolated to the face. The approximate grid spacing $\tilde{h}$ evaluated at the face F in the direction of the mixture fraction gradient reads

$$\tilde{h}_F = \begin{cases} \left[\left(\frac{|d_i\ \eta_i|}{||d_i||\ ||\eta_i||}\right)^{-1} ||d_i||\right]_F & \text{if} \quad \frac{d_i\ \eta_i}{|d_i||\eta_i|} > 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{7.11}$$

where only faces featuring an angle between the face normal and the mixture fraction normal lower than 60° are considered. The local interface sharpness index $\tilde{S}_F$ at a given face reads

$$\tilde{S}_F = \begin{cases} \frac{\tilde{t}_F}{2\tilde{h}_F} & \text{if} \quad \tilde{h}_F > 0 \\ 0 & \text{otherwise} \end{cases}. \tag{7.12}$$

Accordingly, the local interface is considered perfectly sharp when the index is below one. The interface sharpness index S providing a measure for the interface sharpness in the whole computational domain is the arithmetic average of all local non-zero interface sharpness indices.

7.2 Sub-Cycling Technique

The sub-cycling technique aims to improve the efficiency while maintaining the accuracy. The efficiency is measured by the time to solution, while the accuracy is measured by the interface position and its sharpness.

7.2.1 Duncan Foil

The validation of the unsynchronised sub-cycling procedure focuses upon the Duncan Foil. In 1981 Duncan experimentally studied breaking waves produced by a towed hydrofoil [47]. The NACA0012 hydrofoil has been fully submerged at an angle of attack of 5°. The submergence depth has been varied to study the effect on the free-surface elevation. In the following simulations a submergence depth leading to non-breaking quasi-steady waves is chosen. This test case has been widely use to validate free-surface codes [43, 79, 101, 186].

Case Description

The Froude and Reynolds numbers based on the chord length c of the foil are $Fn = 0.5669$ and $Re = 1.43 \times 10^5$. For the present study a non-dimensional submergence depth of $s/c = 1.246$ measured from the leading edge of the foil is chosen. The employed computational domain and boundary conditions are depicted in Figure 7.1. Simulations

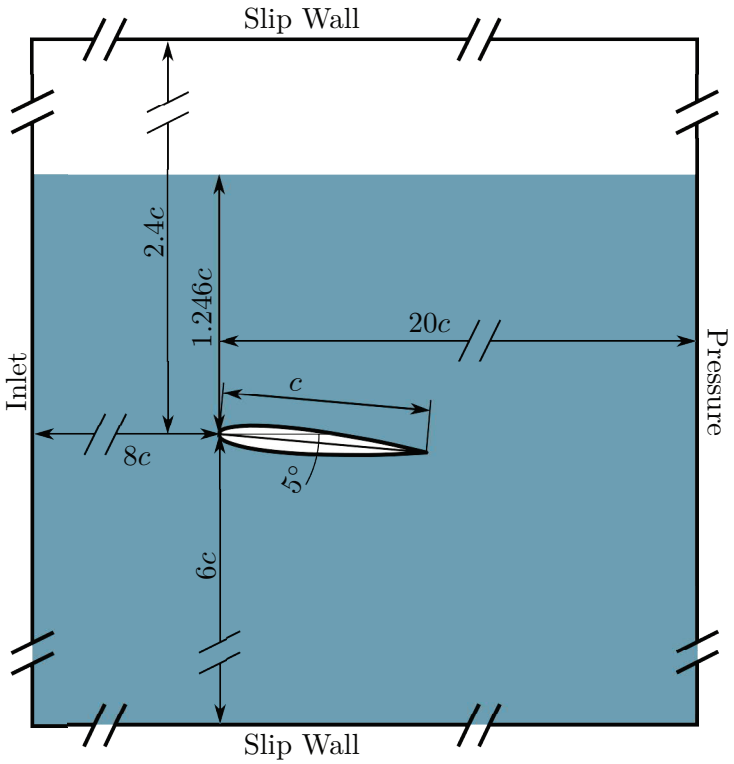
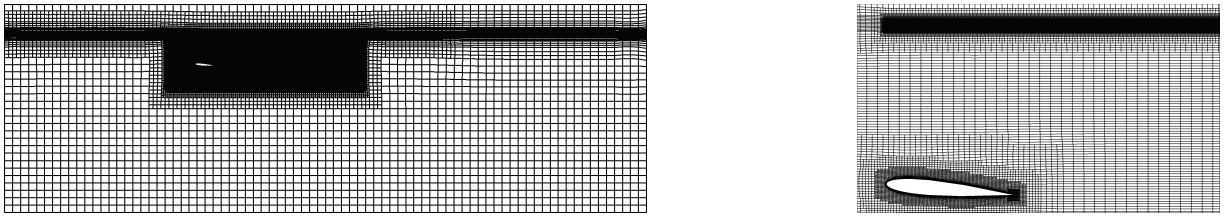


Figure 7.1: Computational domain and boundary conditions employed for the Duncan foil.

have been carried out using a two-dimensional setup. The computational domain starts 8 chord lengths upstream of the foil and ends 19 chord lengths downstream of the foil, where the last 12 chord lengths are used as numerical beach. The bottom is located 6 chord lengths below the foil and the top 2.4 chord lengths above the foil. The unstructured numerical grid, which is refined in the free-surface region, consists of $93,397$ hexahedrons. The origin of the coordinate system is located in the leading edge of the foil. The free-surface region above and 7 chord lengths downstream the airfoil is refined using roughly 50 cells per wave lengths and 20 cells per wave height. The boundary layer cells are

designed to satisfy a theoretical $y^+ < 1$. Hence, the boundary layer is fully resolved. The numerical grid is shown in Figure 7.2. Due to the low Reynolds number the flow is deemed



(a) Whole grid (b) Detail view

Figure 7.2: Full grid and detailed view of grid in the vicinity of the foil and the free-surface.

to be largely laminar, and turbulence effects have been neglected. This is supported by results of other authors [186]. The convective fluxes of the momentum equations have been discretised with the QUICK scheme, while the convective fluxes of the mixture fraction transport have been discretised with an HRIC approach.

To study the influence of the unsynchronised sub-cycling mode on the time to solution, simulations with varying numbers of sub-cycles in the synchronised and unsynchronised mode have been performed. In the unsynchronised sub-cycling mode, the rate of time advancement of the mixture fraction depends on the number of outer iterations used per time step. Therefore, the number of outer iterations per time step in the unsynchronised sub-cycling mode has been varied.

Results

The time to solution is measured by the convergence level of the drag of the submerged foil. The drag of the submerged foil oscillatory converges to steady state. The required computational time to reach steady state is quite long. Therefore, simulations are deemed to be converged when the oscillation amplitude in the predicted drag is less than 0.5% of the mean drag. Figure 7.3 illustrates this procedure. The computational time, which has been used to reach a converged solution, is normalised by the computational time of a simulation without sub-cycling, reaching the same convergence level.

The speed-up in time to solution for the synchronised and unsynchronised sub-cycling mode with varying numbers of sub-cycles using 10 outer iterations is shown in Figure 7.4. For the synchronised mode the achieved speed-up is close to the ideally achievable

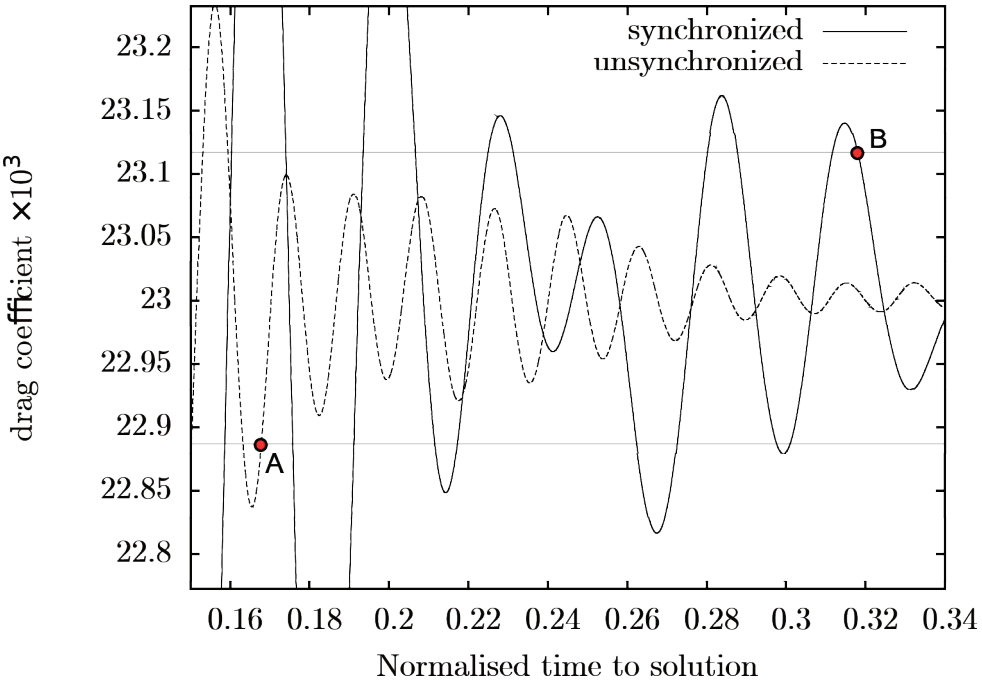


Figure 7.3: Definition of converged solution to measure time to solution. Point A: unsynchronised simulation reaches convergence, Point B: synchronised simulation reaches convergence.

speed-up as defined by Equation (5.17) in Section 5.2, while it is roughly 2 times larger for the simulation in unsynchronised mode. The achievable speed-up in the unsynchronised mode depends on the number of outer iterations. An upper bound can be determined by multiplying the achievable speed-up in the synchronised mode with the number of outer iterations. However, it is usually much smaller, because only the mixture fraction equation virtually advances faster in time. Hence, the achieved speed-up times are within the expected range. The predicted surface elevations for these simulations are shown in Figure 7.5 and compared to experimental results. The numerical results obtained from different sub-cycling modes with varying numbers of sub-cycles are on top of each other. They are afflicted with a lower wave amplitude compared to experimental results, which is due to the grid resolution.

Achieved speedups from simulations using 8 sub-cycles and a varying numbers of outer iterations are depicted in Figure 7.6. The speedup is determined with respect to simulations without sub-cycling using the same number of outer iterations. The ideal speedup for the synchronised sub-cycling mode has been determined for each investigated prescribed

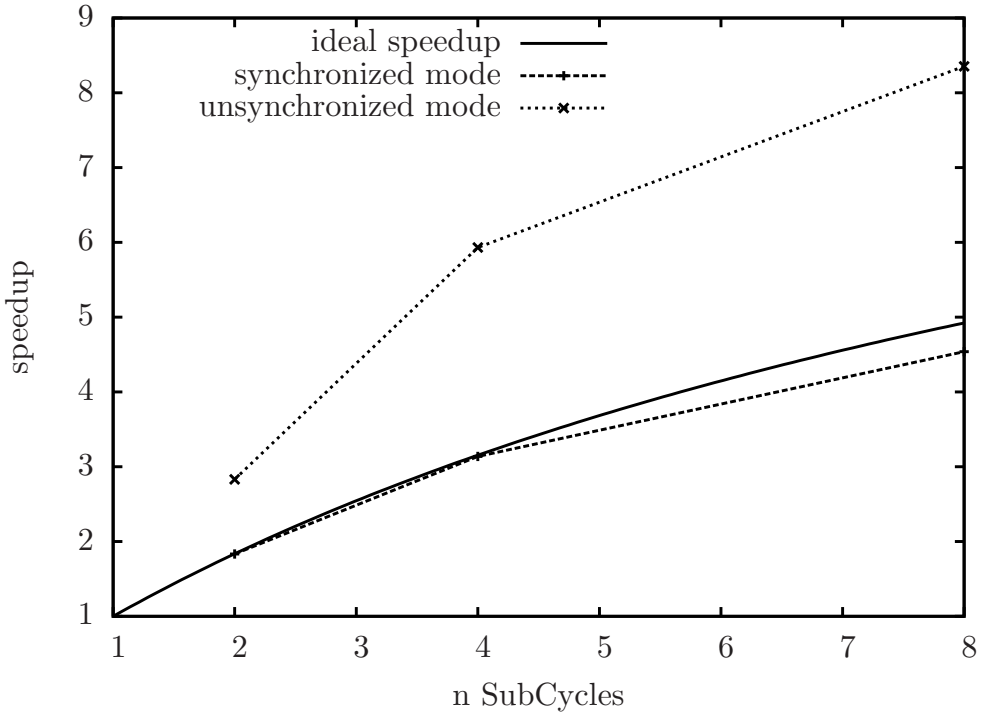
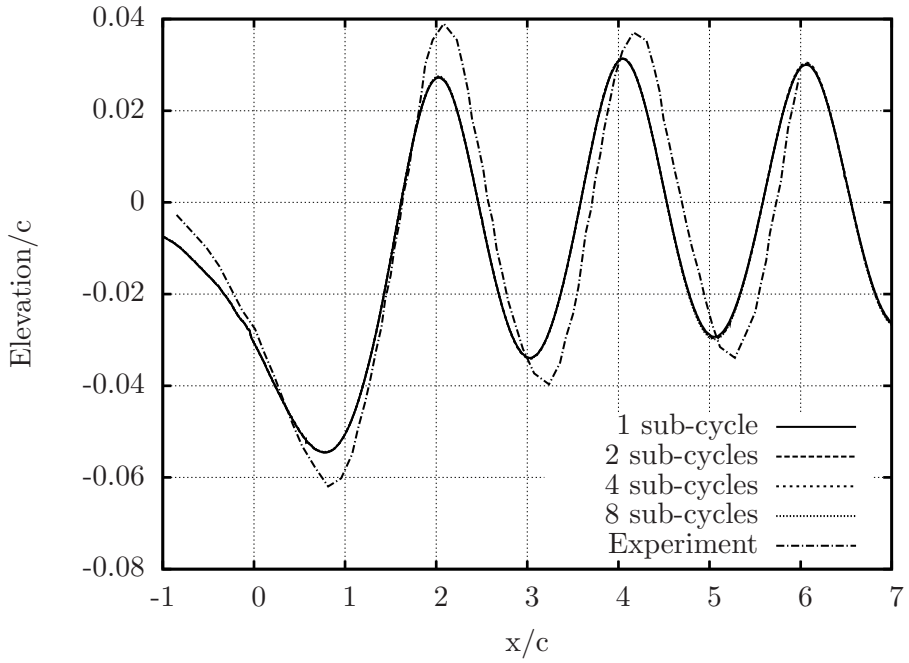


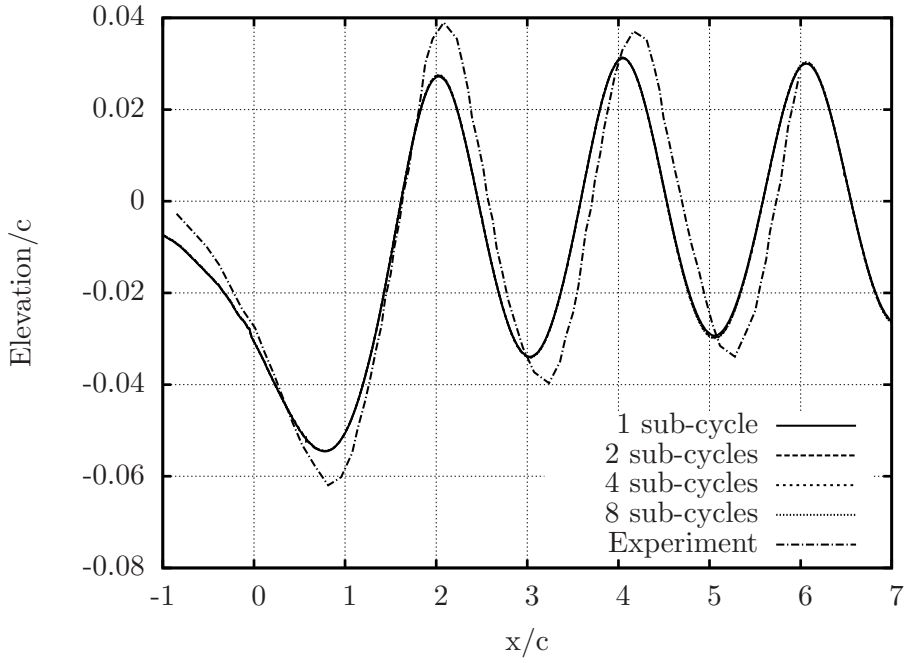
Figure 7.4: Achieved speed-up for the synchronised and unsynchronised sub-cycling mode compared to ideally achievable speed-up for the synchronised mode.

number of outer iterations. As expected it does not depend on the number of outer iterations. The same holds for the attainable speedup in simulations using the synchronised sub-cycling mode, which is almost constant. Using the unsynchronised sub-cycling mode the speed up gets larger when more outer iterations are used. For one outer iteration the achievable speedup is the same for both sub-cycling modes. When the number of outer iterations is increased the speedup for the unsynchronised mode asymptotically converges towards a limit. This behaviour is expected, because only the advancement in time of the mixture fraction equation is affected by the sub-cycling methodology.

The time to solution directly depends on the number of outer iterations per time step. Doubling the number of outer iterations theoretically doubles the time to solution, assuming that the convergence behaviour of the functional of interest with respect to the simulation time is independent on the number of outer iterations. However, the convergence behaviour is not independent on the number of outer iterations. Especially using only a limited number of outer iterations the convergence over the simulation time is slower



(a) synchronised mode



(b) unsynchronised mode

Figure 7.5: Predicted surface elevations in synchronised and unsynchronised mode with varying sub-cycles compared to experimental results.

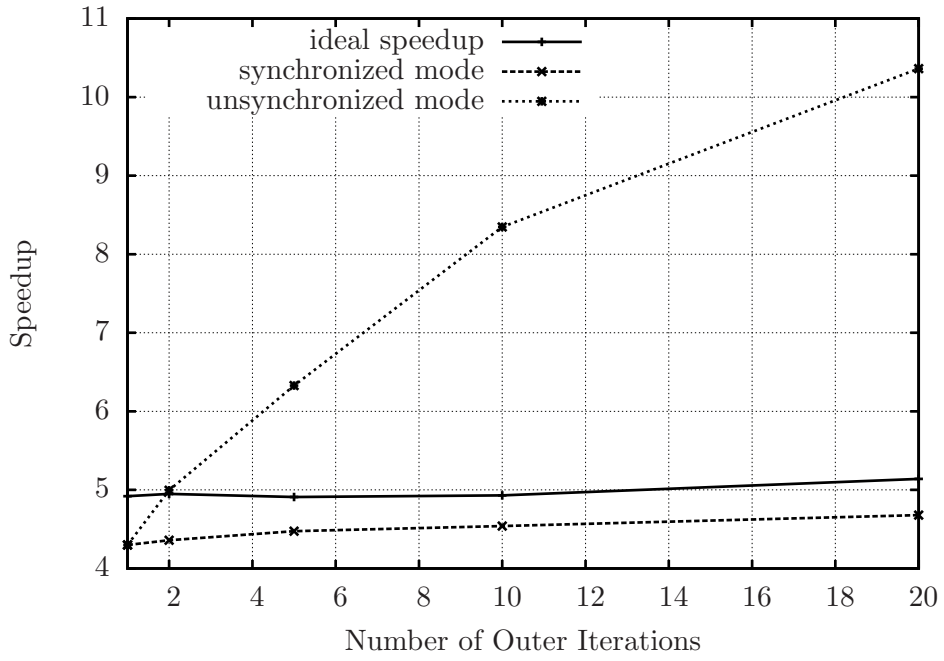


Figure 7.6: Speedup for the synchronised and unsynchronised sub-cycling mode with varying numbers of outer iterations.

than for a simulation where the solution is fully converged in each time step. The absolute time to solution has been compared in Figure 7.7, to evaluate the effect of the number of outer iterations on the time to solution for the unsynchronised sub-cycling mode. The depicted time to solution is normalised by the longest time to solution that occurred for all investigated cases. The increase in the attainable speedup in the unsynchronised mode is less than the increase in the time to solution due to more outer iterations. Hence, a simulation with only one outer iteration yields the lowest time to solution.

The unsynchronised sub-cycling mode is only applicable for simulations where no time-accurate solutions are required. Hence, it is not necessary to converge the solution for each time step and the use of one outer iteration would lead to the best time to solution. However, the implicit time integration schemes tend to loose their stability when only one outer iteration is used [54], and only small time steps can be realised. Hence, it is advisable to use at least 2 outer iterations. When using more than one outer iteration the unsynchronised sub-cycling mode is able to further reduce the required time to solution.

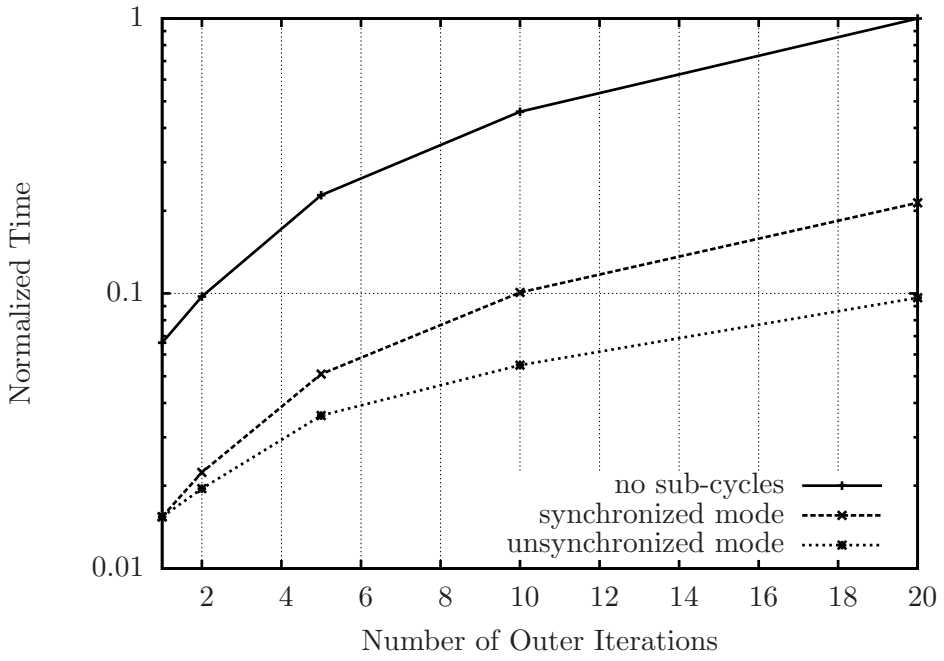


Figure 7.7: Normalised time to solution for simulations without sub-cycling and with sub-cycling in the synchronised and unsynchronised mode.

7.2.2 Dam Break

The two-dimensional dam break, experimentally studied by Martin and Moyce [117], involves pronounced dynamics of the interface and has been used intensively to scrutinise the predictive accuracy of numerical methods in the past [46, 161, 163]. In the present study, it is used to validate the capabilities of the sub-cycling technique for simulations which require time-accurate solutions.

Case Description

The test-case investigates the collapse of a rectangular water column with an initial height of $h = 292\,\mathrm{mm}$ and an initial width of $a = h/2$ into a box-shaped domain. The flow is considered laminar and the (liquid phase) Reynolds number reads $Re = 1.2 \times 10^5$. No-slip boundary conditions are applied along the walls, thus the column collapses exclusively under gravity. The global time scale refers to $\frac{1}{\tau} = \sqrt{2g/a}$, where t is the time, $g = 9.81\,\mathrm{m/s^2}$ is the gravitational acceleration and a is the initial width of the water column.

The size of the domain is 584 mm × 365 mm. The investigation is confined to the period before the surge front hits the opposite wall. The 2D grid involves 8,400 control volumes using an equidistant spacing in vertical and horizontal direction featuring $\Delta x = 4.9\,\text{mm}$ and $\Delta y = 5.2\,\text{mm}$. Figure 7.8 displays the investigated configuration and the employed grid.

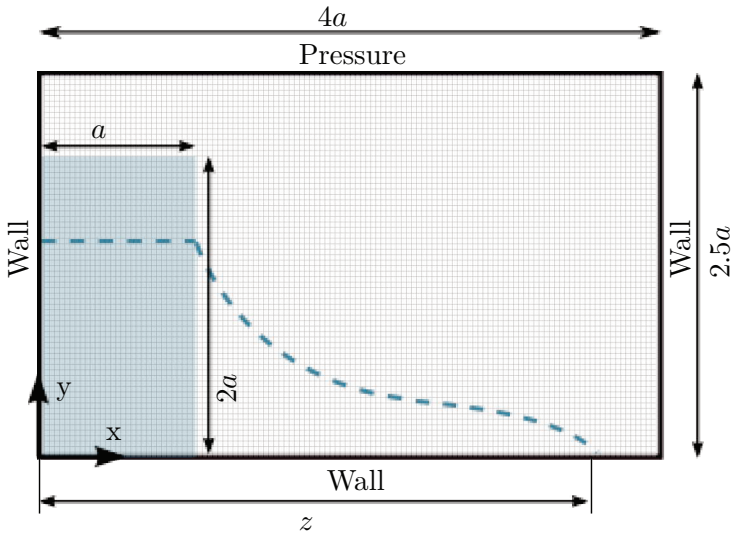
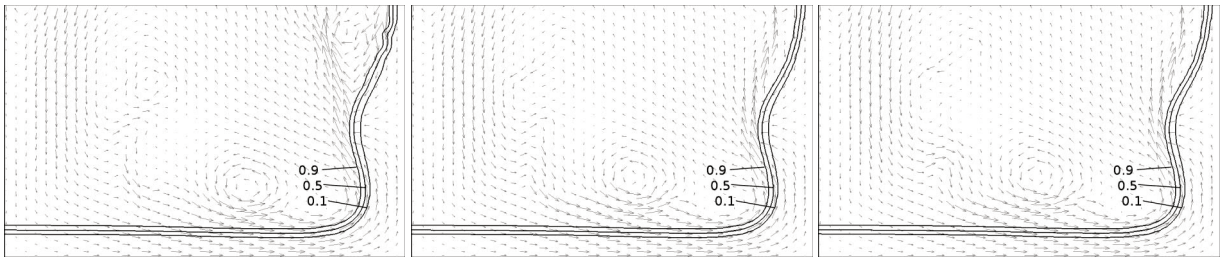


Figure 7.8: Numerical grid, computational domain and boundary conditions employed to study the 2D dam break test case.

Examples included in this validation exercise refer to a constant global time step of $\Delta t = 0.001\,\text{s}$ $(1.59 \times 10^{-4}\,\tau)$ with various amounts of sub-cycles using the synchronised sub-cycling technique. Furthermore, the influence of varying numbers of sub-cycles using a constant time step applied to the concentration transport equation has been studied.

Results

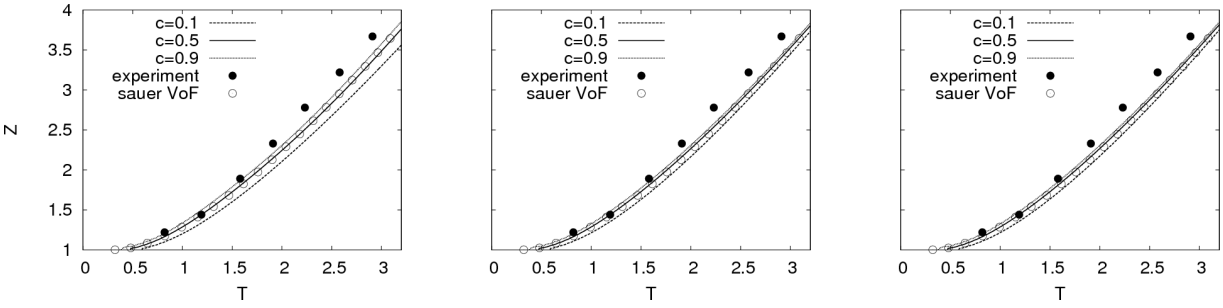
Figure 7.9 displays the predicted contour lines for air-volume mixture fraction levels of 10%, 50% and 90% from simulations with varying numbers of sub-cycles obtained $0.6s$ after the collapse of the water column. Furthermore, the velocity field is depicted by arrows. Results obtained for 4 and 8 sub-cycles are in fair agreement with each other, while small differences remain when compared to the simulation without sub-cycles. These differences are attributed to a slightly blurred free-surface since the Courant number without sub-cycling is partly above 0.5.



(a) $N_{\Delta t} = 1$ (b) $N_{\Delta t} = 4$ (c) $N_{\Delta t} = 8$

Figure 7.9: Dam break test case: Contour lines of volume concentration and velocity field examined at t=0.6s using a global time step of $\Delta t = 0.001\,\mathrm{s}$ for different numbers of sub-cycles.

Figure 7.10 provides a comparison of predicted surge front positions by means of normalised horizontal distances $Z = z/a$ over non-dimensional times $T = t/\tau$ for various numbers of sub-cycles in conjunction with a global time step of $\Delta t = 1 \cdot 10^{-3}\,\mathrm{s}$. The position of the surge front z is difficult to measure, thus the computed surge front is traced by a sequence of three alternative volume-fraction values, i.e. $c = 0.1$, $c = 0.5$ and $c = 0.9$. Results are compared against experimental values and predictions published by Sauer [163] using a similar numerical approach. Displayed results clearly reveal the



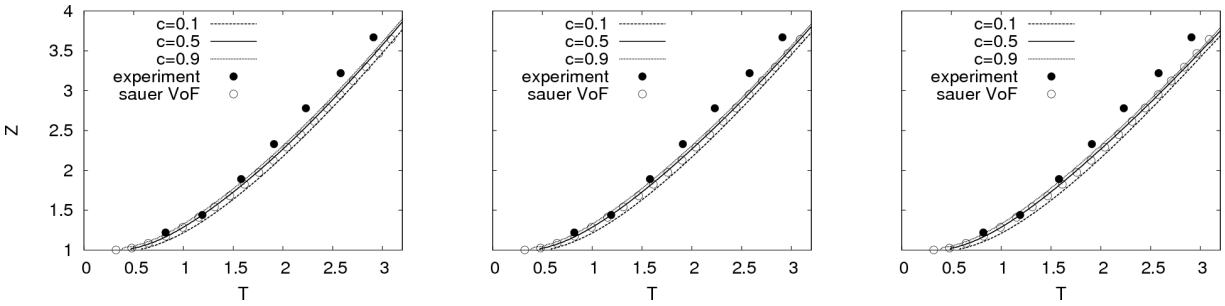
(a) $N_{\Delta t} = 1$ (b) $N_{\Delta t} = 4$ (c) $N_{\Delta t} = 8$

Figure 7.10: Dam break test case: Temporal evolution of the surge front for a constant global time step.

predictive benefits offered by the use of sub-cycling. Without sub-cycling the interface rapidly tends to get blurred, indicated by the spreading of concentration isolines used to identify the surge front. This effect is attributed to the Courant number dependency of

the high resolution interface capturing schemes. Performing 4 sub-cycles — which yields a time step of $\Delta t = 2.5 \cdot 10^{-4}$ s for the concentration equation — provides already sufficiently accurate results (cf. Figure 7.10(b)).

The attainable speed-up of the sub-cycling approach has been verified for a sequence of global time steps in conjunction with a constant sub-cycling time step for the concentration equation ($\Delta t = 2.5 \cdot 10^{-4}$ s). Results displayed in Figure 7.11 reveal only minor changes of the predicted surge front evolution when using sub-cycles. All predictions are still close to results obtained by Sauer as well as experimental values. Figure 7.12 illustrates the achieved speed-up in comparison to the ideal speed-up curve (5.17). The observed speed-up falls slightly below the ideal curve, which indicates that the sub-cycling procedure performs as expected.



(a) $N_{\Delta t} = 1$ (b) $N_{\Delta t} = 4$ (c) $N_{\Delta t} = 8$

Figure 7.11: Dam break test case: Temporal evolution of the surge front for a constant time step of the concentration equation.

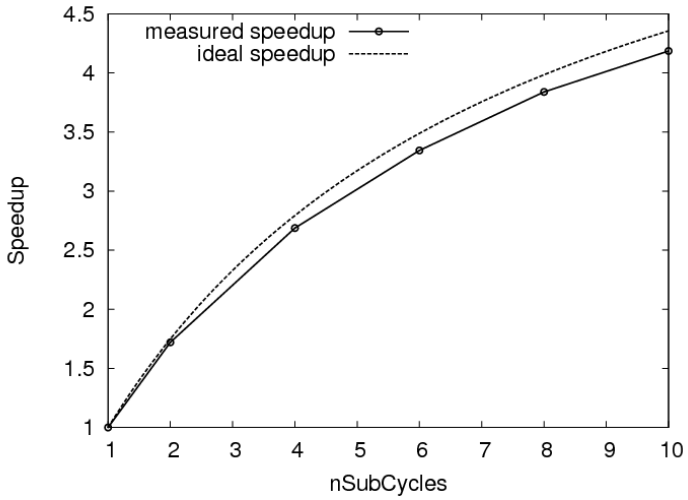


Figure 7.12: Comparison of observed and ideal speed-up due to the use of the sub-cycling technique for the dam break case.

7.3 Explicit Interface Sharpening

7.3.1 Rising Bubble

The rise of a circular bubble in a liquid column has been set up as a benchmark case by Hysing et al. in 2009 [86]. Since then, the case has been used by various groups using different numerical methods [5, 76, 100]. Due to the abscence of experimental data results will be compared to published numerical datasets. Two of them (TP2D [84, 85], freeLIFE [44, 136, 137]) have been presented in [86] and are computed with academic finite element methods using the level-set technique for two-phase flows. Results from a finite volume method, i.e. openFOAM [130], using the VoF approach with a dedicated surface compression technique have been presented in [100].

Case Description

The geometrical configuration of the testcase is shown in Figure 7.13. A circular bubble with a radius of $r = \frac{1}{4}a$, where a is the width of the channel, is initially positioned at $x = y = 0.5a$. The origin of the coordinate system is located in the lower left corner. The liquid column has a height of $2a$. The density of the liquid forming the bubble is

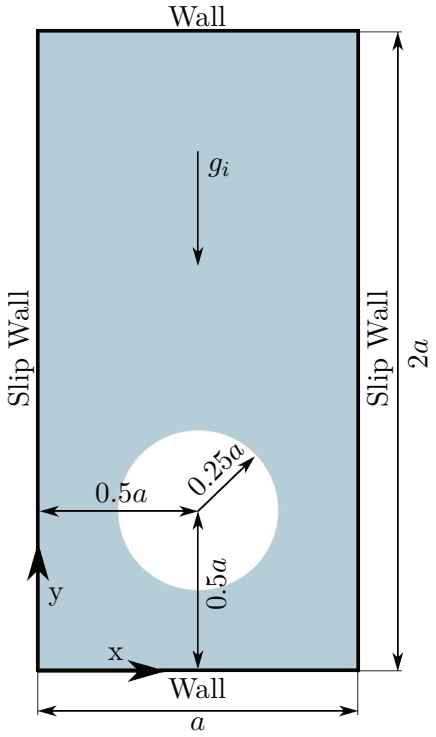


Figure 7.13: Definition of rising bubble test case.

lower than the density of the liquid surrounding the bubble. For the benchmark two cases have been defined, which differ in their material definitions. The case incorporating higher density and viscosity ratios has been assessed because it provides an interface evolution with strong topological changes. Physical quantities defining this test case are given in Table 7.1. The test case features a Reynolds number of $Re = 35$, an Eötvös number

$a[m]$	$\rho_{bubble}\left[\frac{kg}{m^3}\right]$	$\rho_{column}\left[\frac{kg}{m^3}\right]$	$\mu_{bubble}\left[\frac{kg}{sm}\right]$	$\mu_{column}\left[\frac{kg}{sm}\right]$	$g\left[\frac{m}{s^2}\right]$	$\sigma\left[\frac{N}{m}\right]$
1	1	1,000	0.1	10	0.98	1.96

Table 7.1: Physical quantities defining the Rising Bubble test case.

of $Eo = \frac{\Delta p g L^2}{\sigma} = 125$, where Δp is the difference in density of the two phases, L a characteristic length and σ the surface tension. The Weber number W associated to the test case reads $W = \frac{\rho u^2 L}{\sigma} = 125$. Surface tension effects are taken into account by a surface tension model according to Ubbink [180]. Within this model surface tension effects are accounted for by an additional source term for the momentum equation. The cell-wise

local source term reads $S_P = \sigma \mathcal{K}_P \left(\frac{\partial c}{\partial x_i}\right)_P V_P$, with $\mathcal{K}_P$ being the free-surface curvature. The free-surface curvature is evaluated based on a smoothed volume fraction field by $\mathcal{K}_P = -\frac{1}{V_P} \sum_{F(P)} \left(\frac{\frac{\partial c_F}{\partial x_i}}{\frac{\partial c_F}{\partial x_j} \frac{\partial c_F}{\partial x_j}} \Delta F_i \right)$. All simulations have been carried out on regular grids with square-shaped cells, ranging from 40 times 80 cells to 320 times 640 cells. Unless declared differently, a grid resolution dependant time step of $\Delta t = \frac{h}{2a}[s]$, with h being the grid spacing, has been used throughout the simulations, which yields an average Courant number of $Co = 0.12$.

For verification purposes the position of the centre of mass X_i, the rise velocity V and the circularity of the bubble $\tilde{c}$ have been computed at each timestep. The centre of mass X_i is defined as

$$X_i = \frac{\int_V c x_i \, dV}{\int_V c \, dV} \tag{7.13}$$

with c being the volume-fraction of the lower density material. The rise velocity $\mathcal{V}$ is evaluated as

$$\mathcal{V} = \frac{\int_V c u_2 \, dV}{\int_V c \, dV} \tag{7.14}$$

with u_2 being the vertical velocity component. The definition of circularity $\tilde{c}$ reads

$$\tilde{c} = \frac{\pi d}{P_b} \quad \text{with} \quad P_b = \text{actual perimeter of bubble.} \tag{7.15}$$

The actual perimeter of the bubble is taken as the length of the iso-line for $c = 0.5$. For each parameter studied in the following, simulations on four different grids, with systematically refined cells, have been carried out. The solution on the finest grid is taken as reference solution and the L_1 norm of the absolute deviation from the reference solution $|q_{t,ref} - q_t|$ for each verification quantity q is evaluated. Solutions on the coarse grids do not exist for every time step of the reference grid, due to the coupling of the time step to the grid size. Hence, the coarse grid solutions are interpolated to intermediate time steps, where necessary, using cubic splines. The effect of the Explicit Interface Sharpening technique on the interface sharpness is appraised by means of the interface sharpness index presented in Section 7.1.2.

Reference results from TP2D obtained on a grid of 5,120 x 10,240 cells, show that the bubble develops a strong convex shape during its rise. Afterwards thin filaments appear and break off. The break-off is predicted between $2.2s$ and $2.4s$. After break-off small satellite droplets trail the bulk of the main bubble which eventually reaches the shape of a dimpled cap. These results are in line with experimental studies from Clift et. al.

[37] indicating that the bubble is somewhere between the skirted and dimpled ellipsoidal-cap regimes. The challenging part of this benchmark is the accurate prediction of the development of the thin filaments after the break-up of the bubble. Within the results obtained from freeLIFE and openFOAM at least small portions of the filaments remain until the end of the simulation time. The predicted size and shape of the satellite droplets is afflicted with a huge variation, while all codes generally predict the same shape for the main bubble. The time of the break-up is indicated by an inflection point of the predicted circularity, when the thin filaments retract after break-up. This behaviour is only observed within the TP2D results. The other benchmark quantities are less sensitive to the exact temporal evolution of the thin filaments.

Results

The temporal evolution of the rising bubble on the finest grid using CICSAM without EIS is depicted in Figure 7.14 with isolines of the mixture volume fraction for 5%, 50% and 95%.

The break-off of the bubble between $2.2s$ and $2.4s$ and the subsequent development of thin filaments are clearly visible. At $t = 2.6s$ the satellite droplets are separated and small fragmented portions of the thin filaments remain until the end of the simulation. The final shape of the bubble at $t = 3s$ compared to results obtained with other codes is shown in Figure 7.15. While comparing the results it should be kept in mind that the results are obtained on grids with largely varying grid resolutions. Results of TP2D have been obtained on a grid of 5,120 x 10,240 cells while results obtained with FreeLIFE are based on a grid of 160 x 80 cells and the results shown for FreSCo$^+$base on a grid of 320 x 160 cells. The largest differences in the shapes are found in the region of the thin filaments. Results obtained from TP2D indicate that the satellite droplets are completely separated, and the thin filaments have retracted. The thin filaments are clearly visible in results obtained with FreeLIFE, however the satellite droplets are no longer connected to the main bubble. The shape of the satellite droplets seems to be distorted. Within results from FreSCo$^+$ the satellite droplets are clearly visible and slightly larger than the droplets predicted by TP2D while the thin filaments are strongly fragmented. The predicted shape of the main bubble shows only minor differences for all codes.

Results for the benchmark quantities and the interface sharpness, using CICSAM as concentration advection scheme without explicit interface sharpening are shown in Figure

7.16 for systematically refind grids and the lowest time step. The predicted vertical centre of gravity shows a strong grid dependency and visually converges to the reference solution using finer grids. The same holds for the predicted rise velocity, however the deviations from the reference results are larger. The reference result for the circularity achieved by TP2D shows a $C1$-discontinuity at $t = 2.4s$ indicating the separation of droplets from the skirted bubble. This discontinuity is also predicted by FreSCo$^+$. However, the effect gets less pronounced with increasing grid density which indicates that a portion of concentration remains in the area of the thin filaments. The interface of this remaining concentration can only be resolved on a grid which is sufficiently fine. No clear retraction of the thin filaments is visible which may indicate that the surface tension model, which has not been reviewed within this work, is not working properly. In general, results obtained with FreSCo$^+$ predict a lower circularity than the reference results obtained with TP2D but a higher circularity after the break-up than openFOAM. This may also be attributed to deficits in the surface tension model. The mean interface sharpness represented by the sharpness index is shown in Figure 7.16(d). It is below 1, indicating a sharp interface, until $t = 2.2s$ for all simulations. When the bubble breaks up and the thin filaments start to develop, the sharpness index is increasing. This is attributed to an insufficient grid resolution to resolve the thin filaments with a sharp interface. Hence, the increase of the sharpness index is lower for finer grids. Despite the deviations in the circularity the predicted results agree well with the reference results.

The temporal evolution of the rising bubble on the finest grid with activated EIS is shown in Figure 7.17. No differences in the predicted bubble shapes are visible compared to the simulations without EIS presented in Figure 7.14.

Results for the benchmark quantities from simulations with EIS are presented in Figure 7.18. Though the interface has been resolved well without EIS, due to a sufficiently small time step, the inclusion of EIS improves the results for all predicted quantities slightly. The largest differences can be seen for the predicted rise velocity. The overall physical behaviour is not changed due to the sharpening algorithm, the method is still able to predict separating droplets. As the grid resolution is insufficient to resolve the thin filaments, the EIS methodology is not able to increase the interface sharpness significantly after the break-up of the bubble.

The formal order of accuracy for the proposed EIS scheme, is evaluated using the Least Squares version of the Grid Convergence Index proposed by Eça and Hoekstra [50], as

described in Section 7.1.1. The procedure is applied to the root mean square of the proposed benchmark quantities for the lowest Courant number $Co = 0.12$. Although the general discretisation practice is second-order accurate, a lower order of convergence due to the first order accuracy of down-winding is expected when using dedicated concentration convection schemes. Depending on the selected benchmark quantity the observed order of accuracy varies between 1.4 and 1.65 using CICSAM (Table 7.2). The order of accuracy using CICSAM and EIS is slightly lower and varies between 1.32 and 1.6. The uncertainty is in the same range for both methods, being slightly lower for the approach considering EIS.

Table 7.2: Estimated order of accuracy at $Co_{mean} = 0.12$ for the RMS of the proposed quantities, for different convection mechanisms.

Method	Quantity	Order	Uncertainty [%]
CICSAM	X_2	1.625	0.0739%
	V	1.420	0.2332%
	$\tilde{c}$	1.654	0.0610%
CICSAM with EIS	X_2	1.575	0.0701%
	V	1.315	0.1429%
	$\tilde{c}$	1.449	0.0003%

As shown in Section 5.1.3 high resolution schemes have a strong Courant number dependency and tend to blur the interface with increasing Courant numbers. The mean Courant number averaged over all cells and time steps is $Co_{mean} = 0.12$, using the proposed time step. The maximum Courant number in the interface region during the rise of the bubble is $Co = 0.25$. Therefore, CICSAM is working in its proposed Courant number regime and is able to predict a sharp interface. Simulation with a timestep two and four times larger than the standard time step are carried out to show the capabilities of the developed sharpening algorithm. Results obtained with and without interface sharpening for the benchmark quantities are depicted in Figure 7.19. A larger timestep does not only affect the interface sharpness but also increases the temporal discretisation error. This can been seen in Figure 7.19(a), where the predicted vertical position for the centre of mass is lower using a larger timestep. Results obtained with EIS are nearly similar to results

obtained without EIS and the small Courant number, while results obtained without EIS and the large Courant number clearly deviate from the reference results. The largest effect of the interface sharpening algorithm is seen for the rise velocity. Using the large timestep the separation point of the droplets is no longer visible in the predicted circularity. The effect of the sharpening algorithm on the interface sharpness is clearly visible in Figure 7.19(d), where the sharpness index for the large time step shows a strong dependency on the inclusion of EIS. Figure 7.20 reveals that the large timestep leads to a distorted shape of the main bubble in the lower region using the CICSAM convection scheme without sharpening and that the interface of the droplets is blurred. Applying the sharpening algorithm the predicted shapes of the bubble agree well with results obtained from simulations with smaller time steps. Slight differences are only visible for the remaining thin filaments that connect the droplets to the main bubble.

The $L1$ norm of the error for the predicted rise velocity relative to results obtained on the finest grid with the smallest time step is shown in Table 7.3 for CICSAM as concentration advection scheme with and without EIS. Using CICSAM without EIS, the temporal and

	Number of cells			
Courant number	3,200	12,800	51,200	204,800
CICSAM without EIS				
0.48	0.079	0.057	0.040	0.030
0.24	0.060	0.032	0.015	0.004
0.12	0.058	0.025	0.010	0.000
CICSAM with EIS				
0.48	0.074	0.037	0.015	0.002
0.24	0.071	0.035	0.014	0.001
0.12	0.071	0.034	0.014	0.000

Table 7.3: L1 error norm for the predicted rise velocity in $\frac{m}{s}$ using CICSAM without and with EIS.

spatial errors are of the same magnitude doubling the time step or spacing. The absolute error level due to larger time steps is reduced by an order of magnitude including the EIS

algorithm in the solution process on the finest grid. Including EIS the error on the coarser grids is dominated by the spatial discretisation error. The results indicate that due to the use of EIS significantly larger time steps can be realised, compared to pure high resolution interface capturing schemes, while the same level of accuracy is maintained.

The Courant number dependency of high-resolution advection scheme is caused by the inclusion of downwind differences in these schemes. The downwind interpolation technique is applied to include the compressive character, necessary for the prediction of sharp interfaces. However, a sharp interface can be predicted with any convection scheme, when the EIS technique is applied. Hence, the Courant number dependency can be removed. In the following results obtained using limited QUICK as advection scheme combined with interface sharpening are presented.

Predicted benchmark quantities for different time steps are shown in Figure 7.21. The vertical centre of gravity and the rise velocity are slightly lower compared to results obtained with CICSAM as concentration advection scheme supplemented by EIS. The break-off of the satellite droplets is not visible in the circularity plot, contrary to the reference result. Despite the missing break-off the predicted circularities are on top of each user using the same time steps. The sharpness index reveals that the interface is less sharp than the interface in reference calculations and gets more blurred using larger time steps. However, the interface sharpness is still close to 1, which is a satisfactory sharpness level. The temporal evolution of the bubble shape is presented in Figure 7.22 for the smallest and largest time step. A slightly lower final position of the main bubble using the larger time step is observed. The general temporal evolution of the bubble shape is not afflicted by the times step. The interface remains visually sharp for both cases.

Table 7.4 shows the $L1$ error norm of the rise velocity using QUICK as concentration advection scheme with and without supplementary EIS. In contrast to results obtained with CICSAM the spatial error dominates even without sharpening. However, the interface is already blurred for the lowest Courant number as shown in Figure 7.22 and therefore the method is not applicable to predict sharp interfaces. Due to the inclusion of EIS the interface remains sharp and the temporal and spatial error develops in a similar way like for CICSAM with supplementary EIS.

The order of accuracy, using QUICK as advection scheme, has been evaluated by the same procedure as for the simulations with CICSAM as advection scheme. The results are shown in Table 7.5, where also the results for CICSAM are repeated for comparison.

	Number of cells			
Courant number	3,200	12,800	51,200	204,800
QUICK without EIS				
0.48	0.074	0.037	0.015	0.002
0.24	0.071	0.035	0.014	0.001
0.12	0.071	0.034	0.014	0.000
QUICK with EIS				
0.48	0.057	0.031	0.012	0.007
0.24	0.046	0.027	0.011	0.005
0.12	0.039	0.024	0.012	0.000

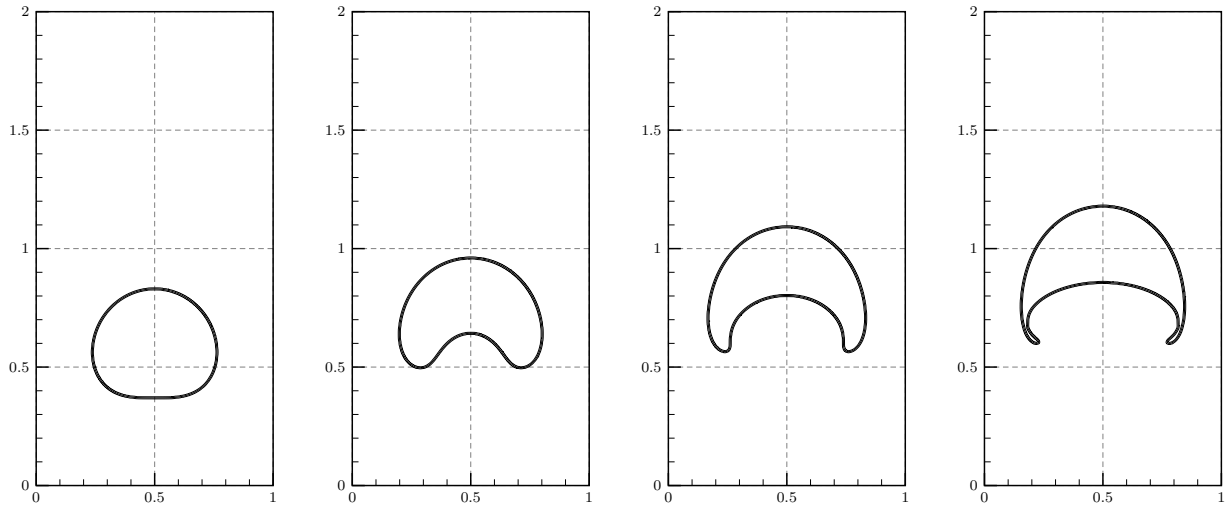
Table 7.4: Relative L1 error norm for the predicted rise velocity using QUICK without and with EIS.

A slightly higher order of accuracy ranging from 1.43 to 1.64 is observed for QUICK with supplementary explicit interface sharpening. This result is expected due to the formal higher order of QUICK compared to CICSAM. In general, the inclusion of EIS has a minor influence on the order of accuracy for the overall algorithm. However, the absolute error level can be reduced significantly.

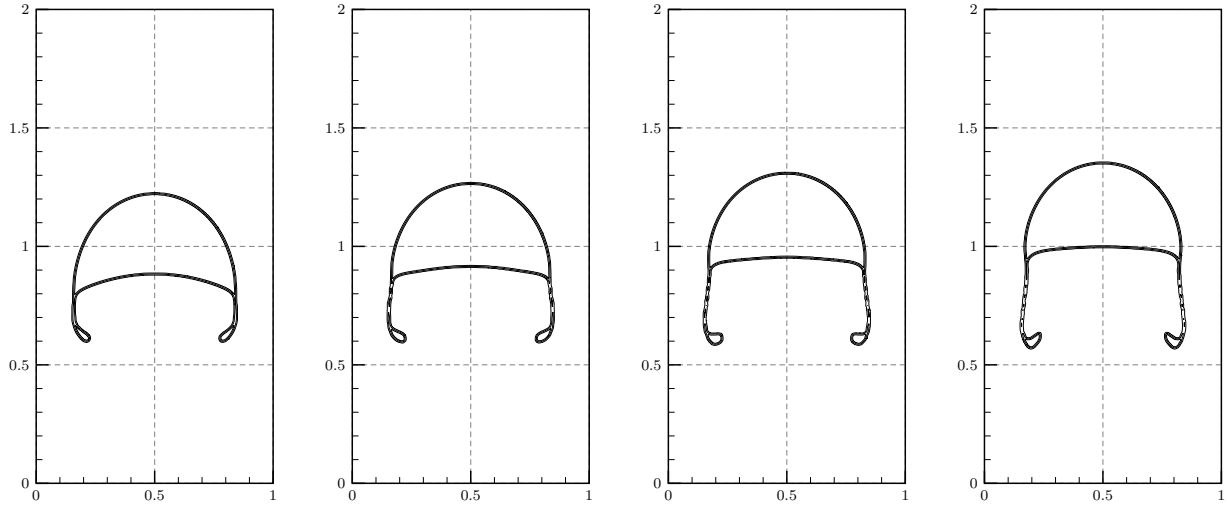
Although slight differences in the predictions obtained from CICSAM and QUCIK as concentration convection schemes are visible, it has been demonstrated that also non-dedicated advection schemes can be used for the concentration transport when the explicit interface sharpening technique is applied. The predictive accuracy is slightly deteriorated using non-dedicated convection schemes. However, the combination of non-dedicated advection schemes and EIS removing the Courant number dependency, may be a beneficial procedure for simulations approaching to steady state, at least for the initial phase of these simulations.

Table 7.5: Estimated order of accuracy at $Co_{mean} = 0.12$ for the RMS of the proposed quantities, for different convection mechanisms.

Method	Quantity	Order	Uncertainty [%]
CICSAM	X_2	1.625	0.0739%
	V	1.420	0.2332%
	$\tilde{c}$	1.654	0.0610%
CICSAM with EIS	X_2	1.575	0.0701%
	V	1.315	0.1429%
	$\tilde{c}$	1.449	0.0003%
QUICK with EIS	X_2	1.639	0.1155%
	V	1.433	0.0050%
	$\tilde{c}$	1.572	0.0868%

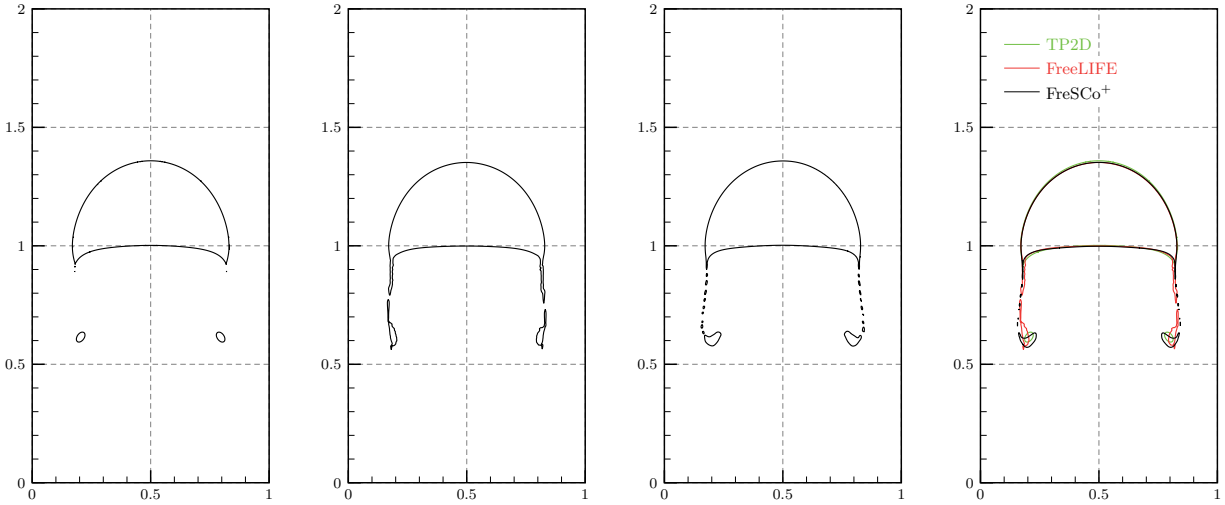


(a) $t = 0.6s$ (b) $t = 1.2s$ (c) $t = 1.8s$ (d) $t = 2.2s$



(e) $t = 2.4s$ (f) $t = 2.6s$ (g) $t = 2.8s$ (h) $t = 3.0s$

Figure 7.14: Temporal evolution of the rising bubble obtained from the finest grid (320 x 160 cells) without EIS.



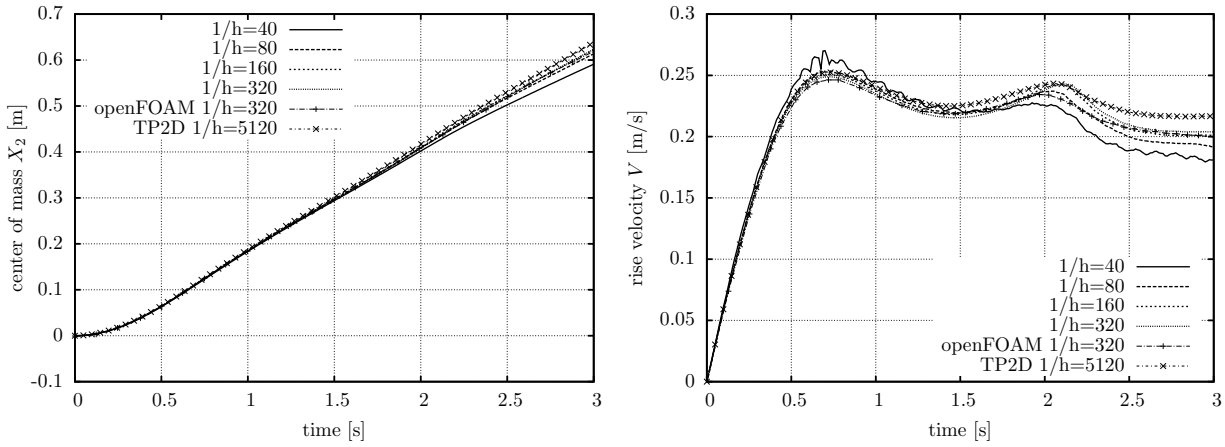
(a) TP2D (5,120 x 10,240 cells)

(b) FreeLIFE (160 x 80 cells)

(c) FreSCo$^+$ (320 x 160 cells)

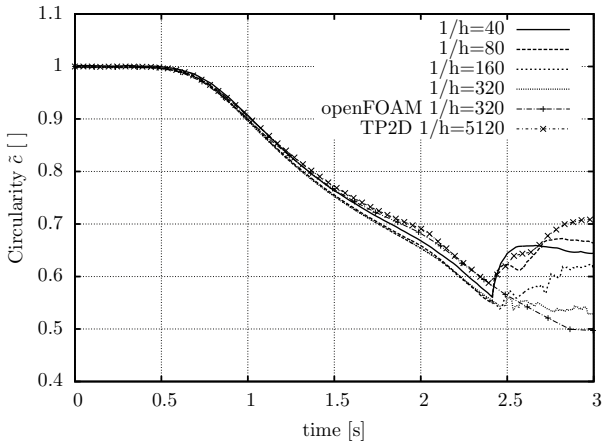
(d) All results.

Figure 7.15: Final shapes of bubble obtained with different codes.

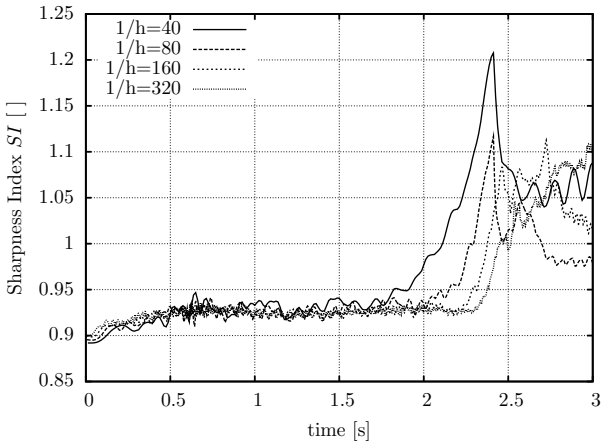


(a) Vertical position of centre of gravity

(b) Rise velocity

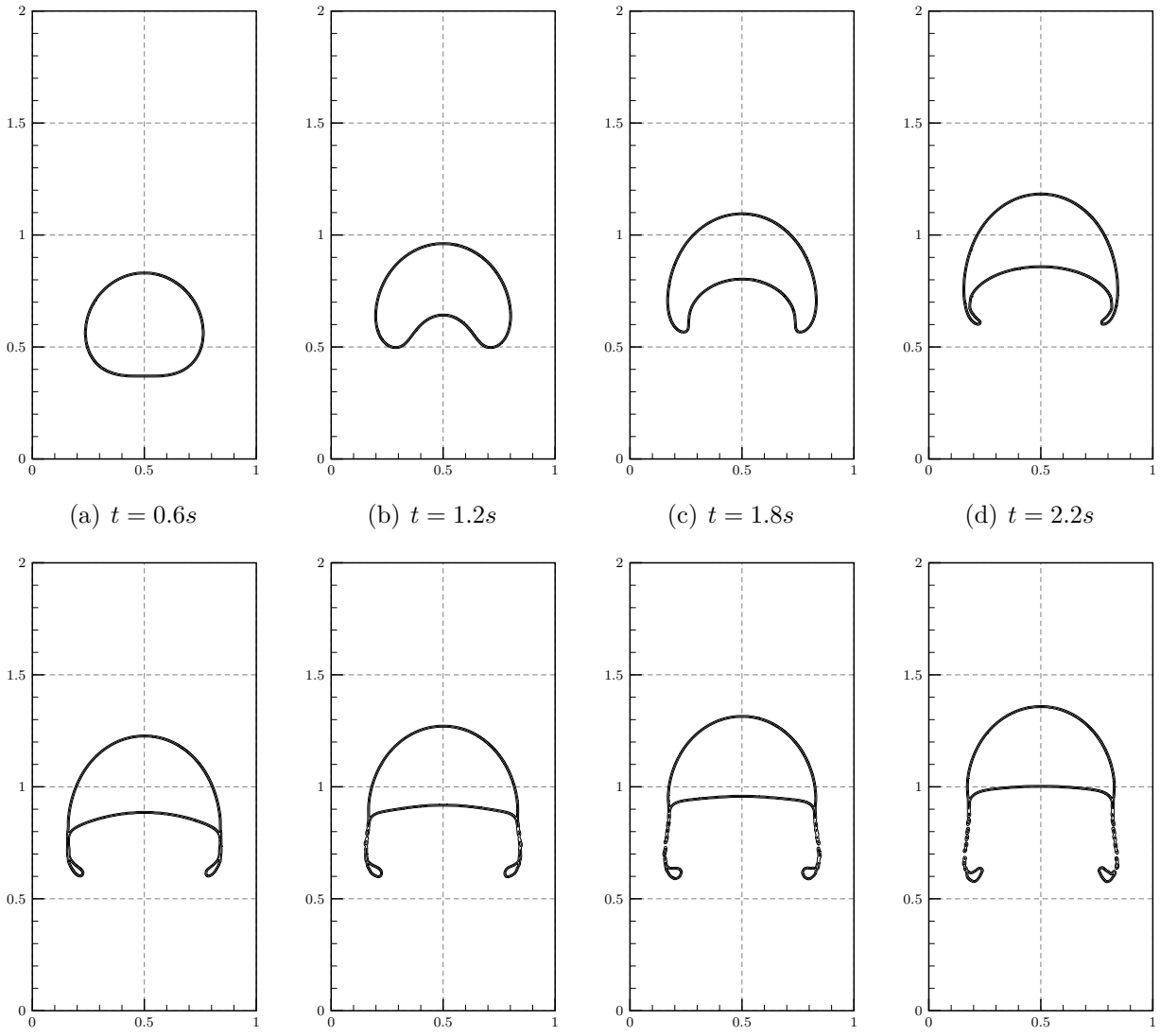


(c) Circularity



(d) Interface Sharpness

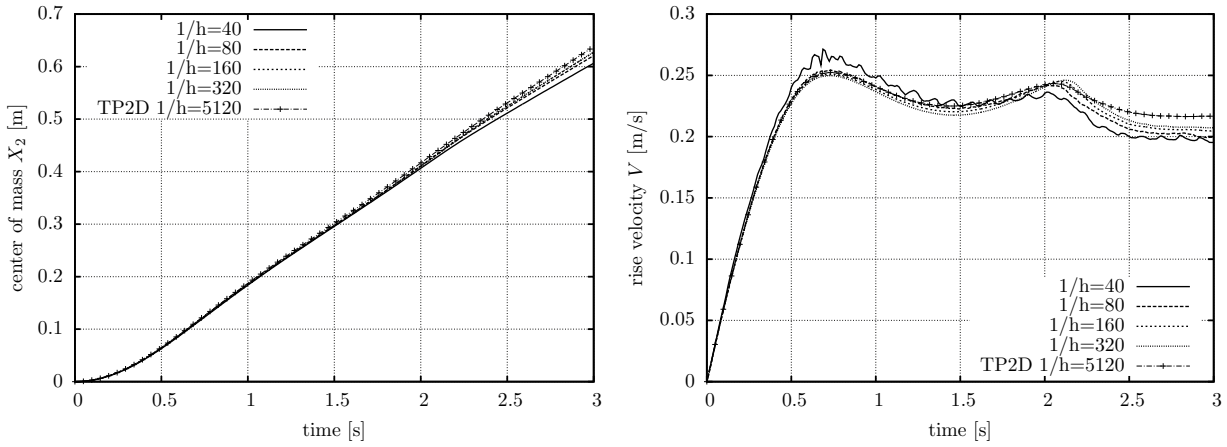
Figure 7.16: Temporal evolution of benchmark quantities and interface sharpness without sharpening for systematically refined grids and the lowest time step.



(a) $t = 0.6s$ (b) $t = 1.2s$ (c) $t = 1.8s$ (d) $t = 2.2s$

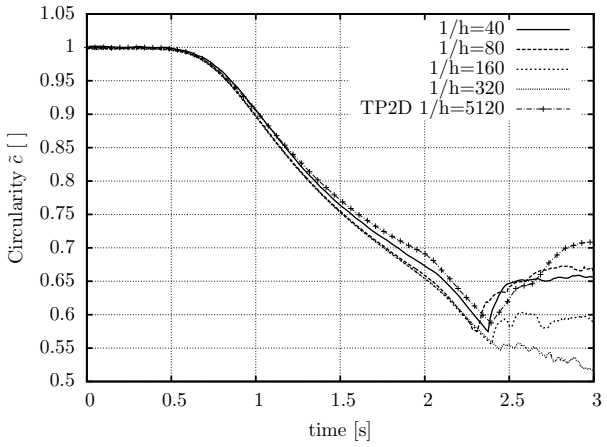
(e) $t = 2.4s$ (f) $t = 2.6s$ (g) $t = 2.8s$ (h) $t = 3.0s$

Figure 7.17: Temporal evolution of the rising bubble obtained from the finest grid (320 x 160 cells) with activated EIS and the lowest timestep.

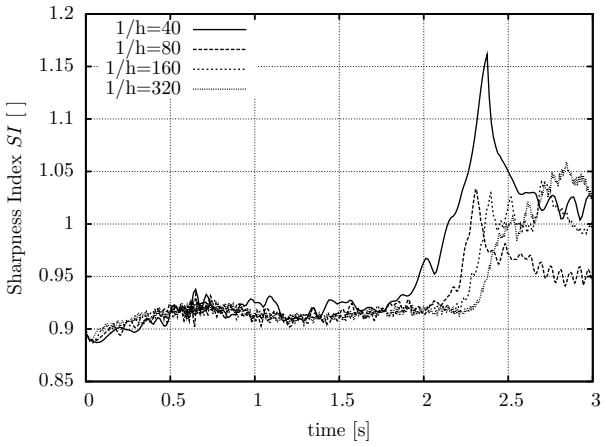


(a) Vertical position of centre of gravity

(b) Rise velocity

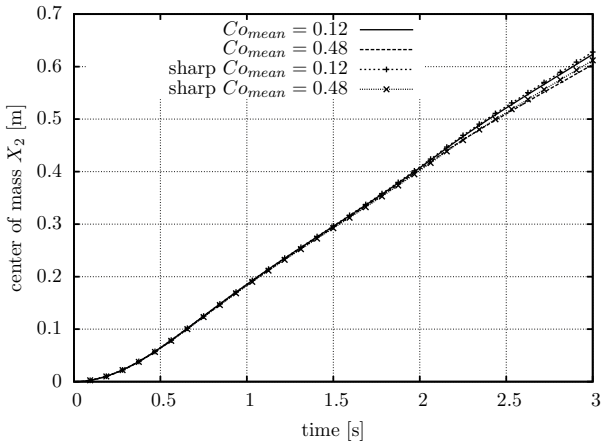


(c) Circularity

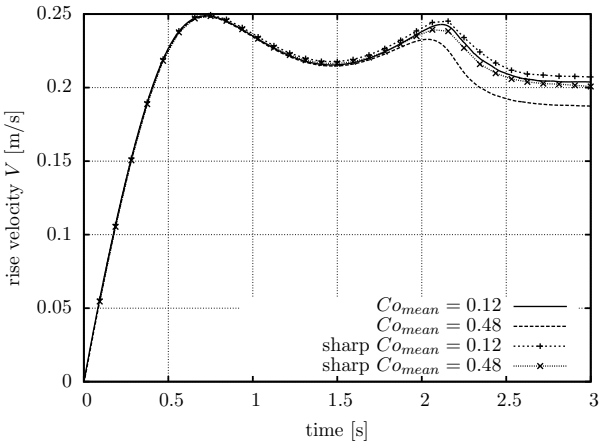


(d) Interface Sharpness

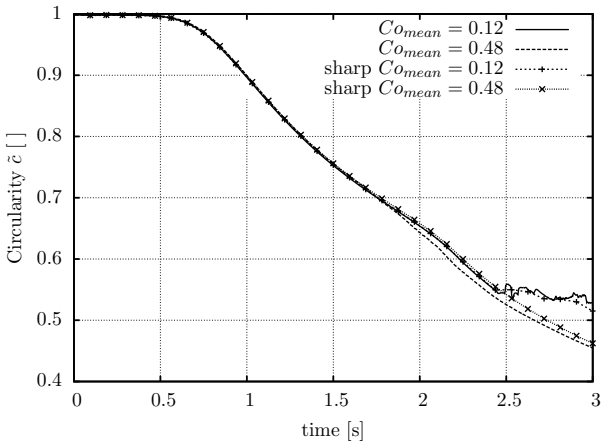
Figure 7.18: Temporal evolution of benchmark quantities and interface sharpness with sharpening for systematically refined grids and the lowest time step.



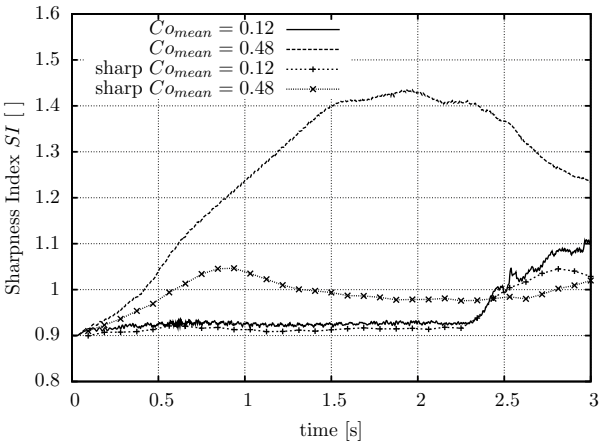
(a) Vertical position of centre of gravity



(b) Rise velocity

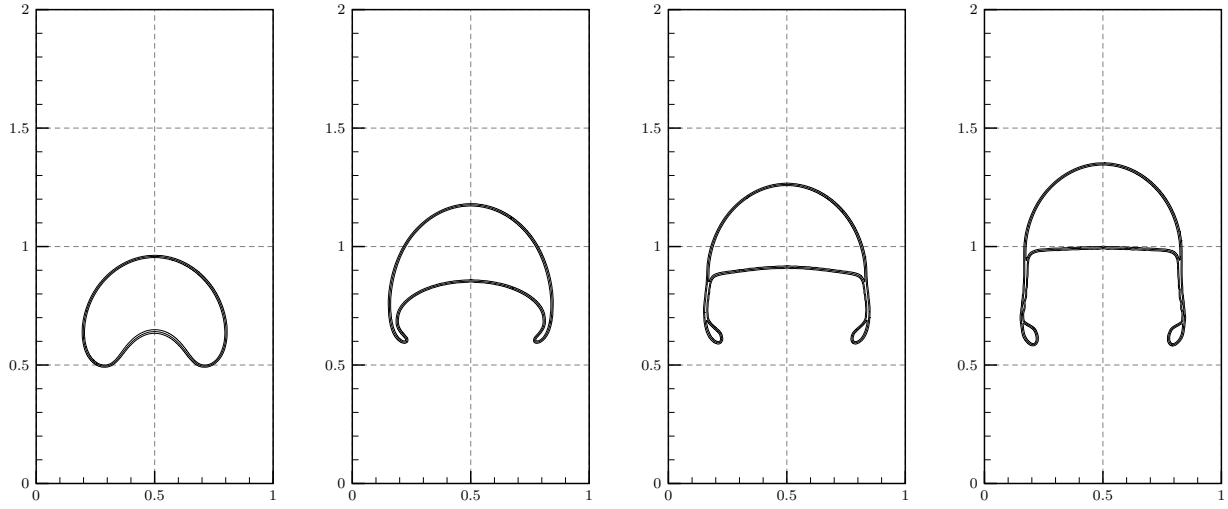


(c) Circularity

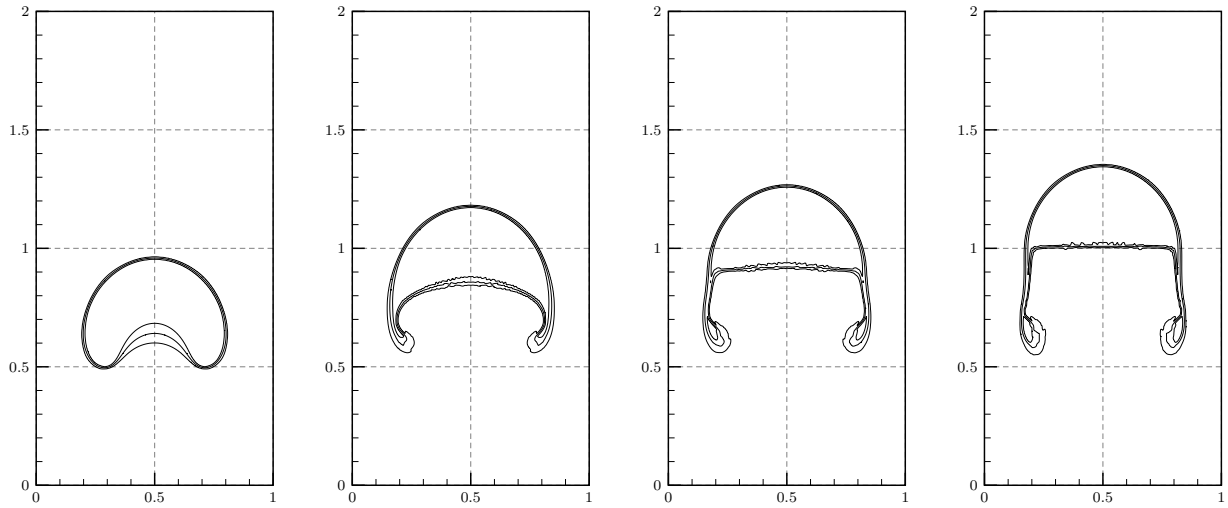


(d) Interface Sharpness

Figure 7.19: Temporal evolution of benchmark quantities and interface sharpness on finest grid (320 x 160 cells) with varying timestep with and without sharpening.

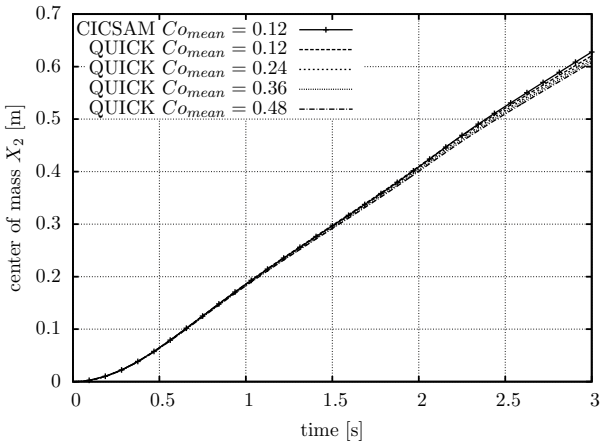


(a) $t = 1.2s$ (b) $t = 2.2s$ (c) $t = 2.6s$ (d) $t = 3.0s$

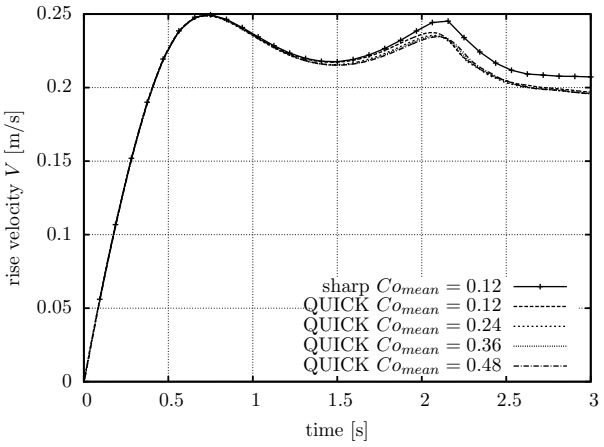


(e) $t = 1.2s$ (f) $t = 2.2s$ (g) $t = 2.6s$ (h) $t = 3.0s$

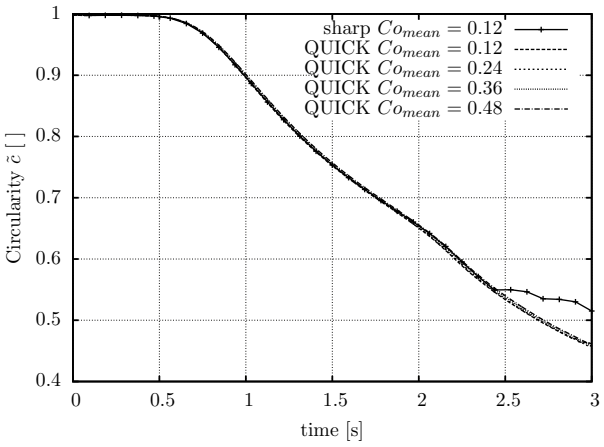
Figure 7.20: Temporal evolution of the bubble shape for the large time step (Isolines $c = 0.05; c = 0.5; c = 0.95$) on the finest grid (320 x 160 cells). Upper row CICSAM scheme with EIS, lower row CICSAM scheme without EIS.



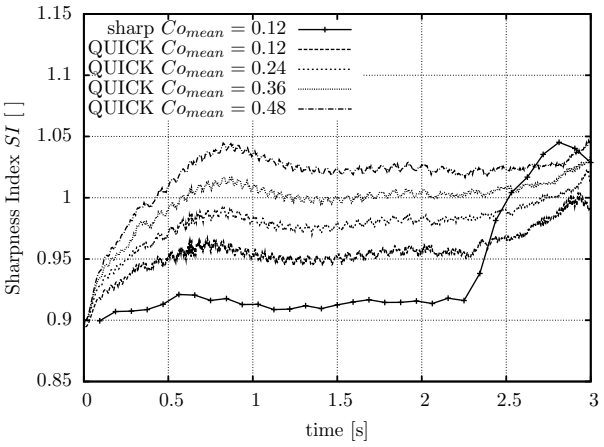
(a) Vertical position of centre of gravity



(b) Rise velocity

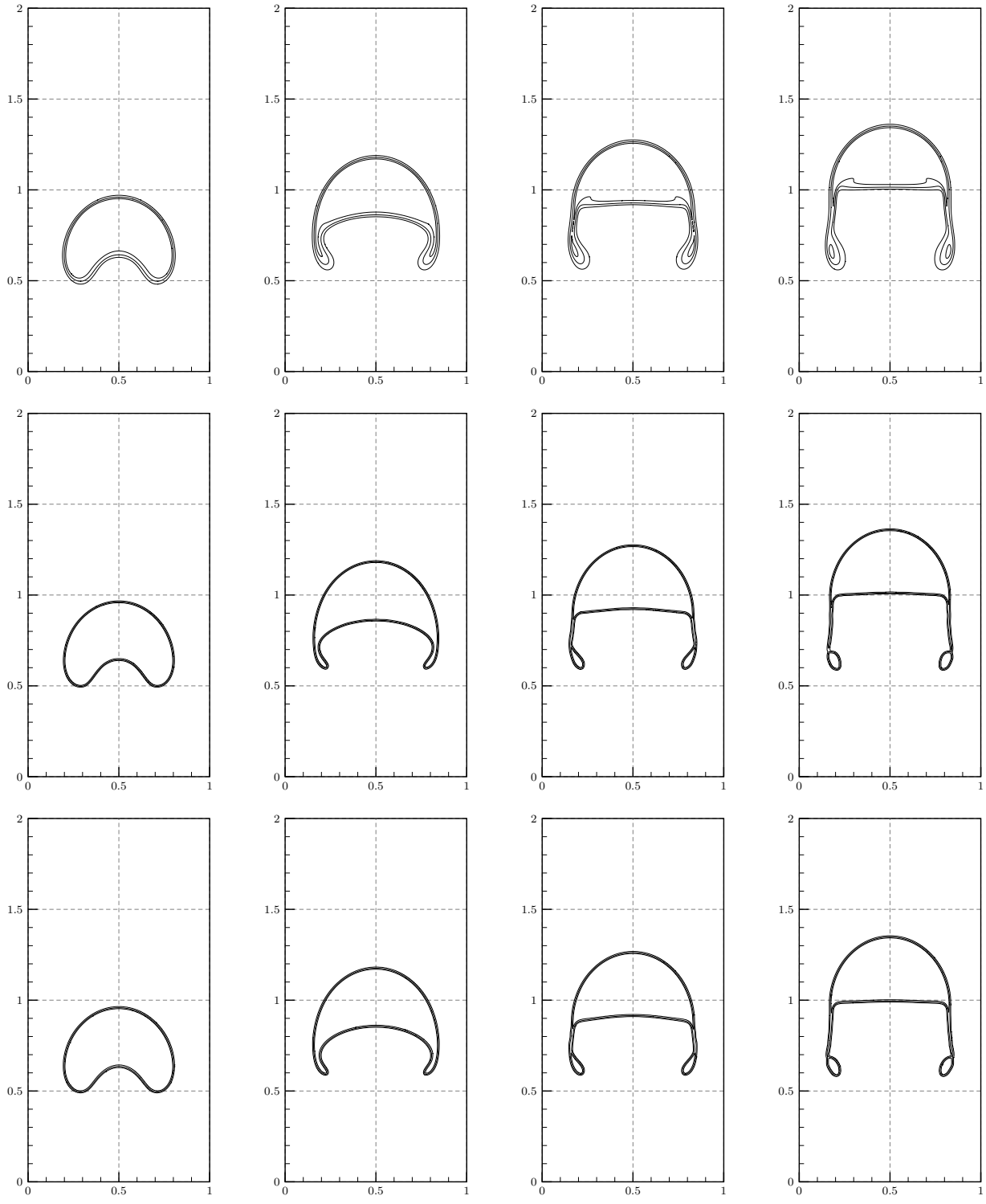


(c) Circularity



(d) Interface Sharpness

Figure 7.21: Temporal evolution of benchmark quantities and interface sharpness using QUICK and EIS on the finest grid (320 x 160 cells).



(i) $t = 1.2s$ (j) $t = 2.2s$ (k) $t = 2.6s$ (l) $t = 3.0s$

Figure 7.22: Predicted bubble rise using QUICK on finest grid (320 x 160 cells). Upper row $Co_{mean} = 0.12$ without EIS, mid row $Co_{mean} = 0.12$ with EIS, lower row $Co_{mean} = 0.48$ with EIS.

7.3.2 Sloshing

Predicting pressure peaks at probe locations of sloshing in tanks requires the preservation of a sharp interface, especially when several consecutive sloshing events are assessed. Experimental long term observations of sloshing in a rectangular tank have been published by Hinatsu et al.[78]. Their results have been intensively used to validate numerical methods (e.g. [61, 62, 142, 143, 195]).

Case Description

The geometrical test setup and the position of the pressure probes is shown in Figure 7.23. Within this validation exercise only sloshing due to sway motion at the resonance period

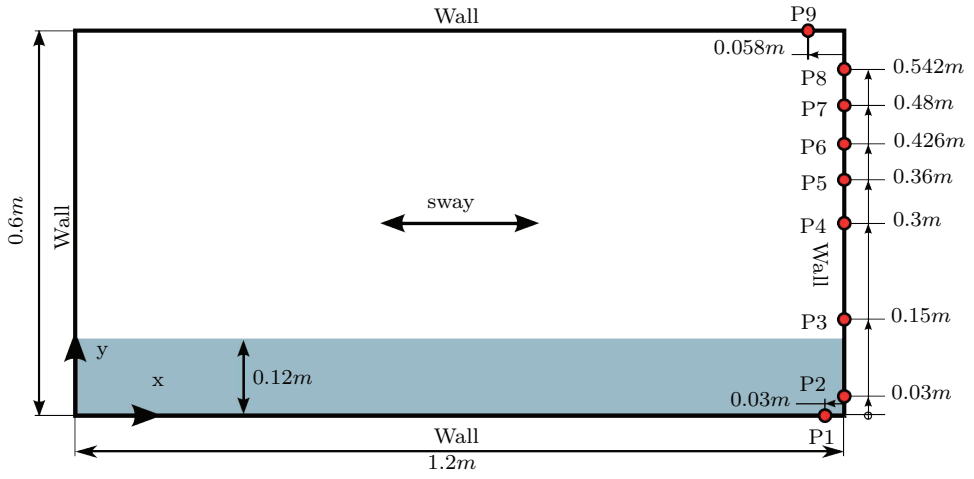


Figure 7.23: Computational domain, boundary conditions and pressure probes employed to study a 2D sloshing case.

of the tank with a filling level of 20% is simulated. Wacławczyk et al. [195] proposed modified motion parameters shown in Table 7.6 which fit to the published tank motion [78] and have also been used within this study. The utilised material properties read $\rho_{air} = 1.205 \frac{kg}{m^3}$, $\mu_{air} = 1.8 \times 10^{-5} \frac{kg}{ms}$, $\rho_{water} = 998 \frac{kg}{m^3}$ and $\mu_{water} = 1 \times 10^{-3} \frac{kg}{ms}$.

Previous studies from Perić et al. [142] revealed that a structured grid consisting of at least 160 x 80 control volumes provides a sufficient resolution. Within their study a maximum time step of $\Delta t = \frac{T}{1{,}000}$ has been shown to provide accurate results in conjunction with the proposed grid. Although, the experiment features some three-dimensional effects,

Table 7.6: Nominal and modified tank motion parameters.

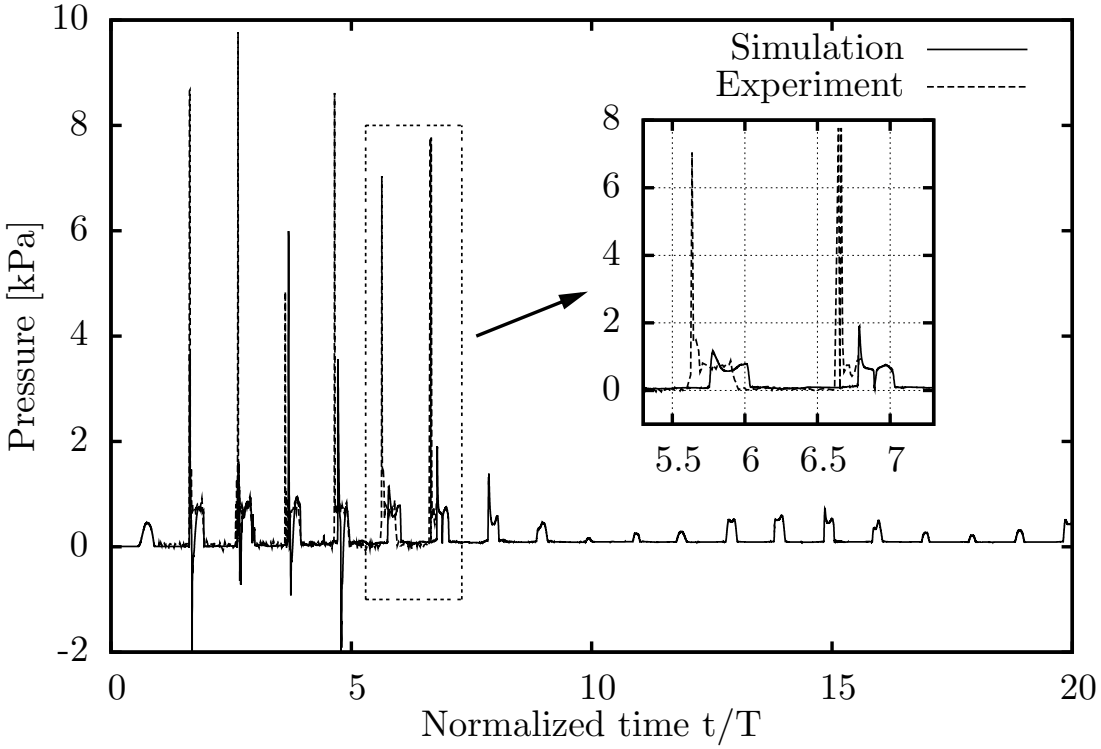
	$A[m]$	$T[s]$	β
original	0.06	1.74	0
modified	0.0605	1.74	0.41

simulations carried out in two dimensions are able to capture most physics accurately [142]. The main difference between two-dimensional and three-dimensional simulations is that air bubbles, which are created during the impact of the fluid on the wall, persist longer in two-dimensional simulations. They have a damping effect on the whole system, which leads to slightly longer periods between the impacts. Although the flow is turbulent, the inclusion of turbulence modelling has a hardly measurable influence on the results. Accordingly, turbulence effects are neglected within the performed simulations.

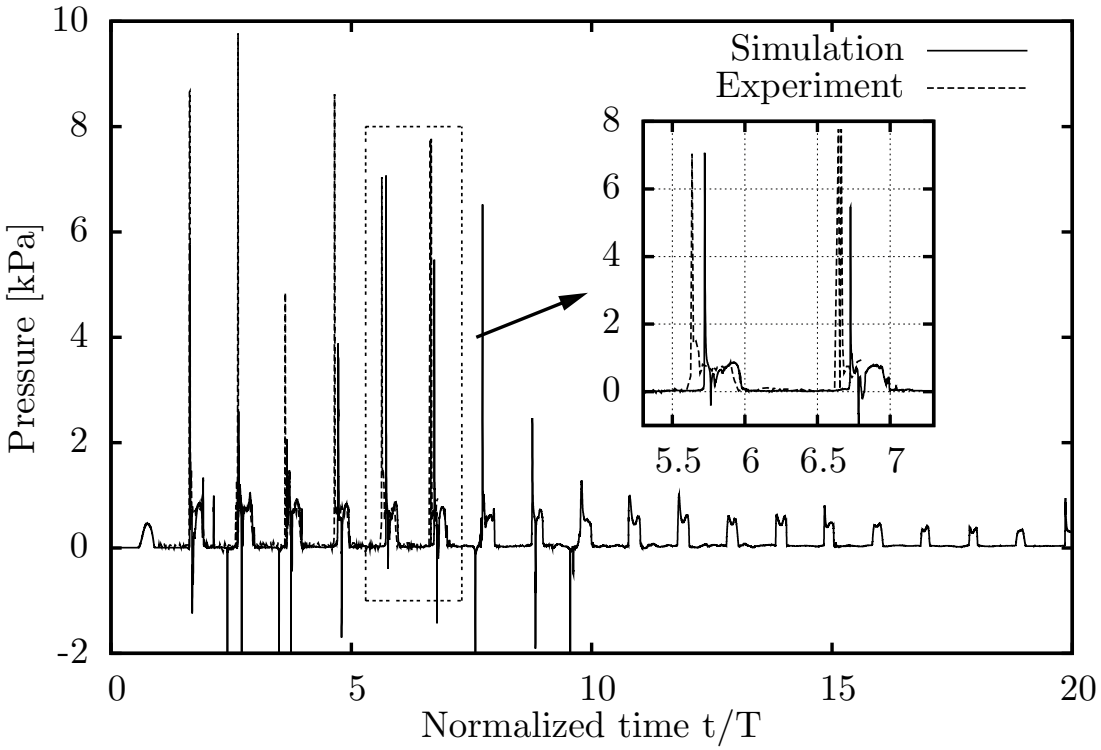
The influence of including EIS is studied in the two-dimensional case with grids consisting of 160 x 80 and 240 x 120 control volumes. The time step has been varied between 500 and 2,000 time steps per period. The advection term of the mixture fraction transport has been discretised using CICSAM. Time advancement is achieved using a second-order accurate implicit three-time level scheme. Furthermore, three-dimensional simulations with and without EIS have been carried out on a grid with 160 x 80 x 16 control volumes. Within these simulations the time step is automatically adapted thus that the RMS value of the cell Courant number is $Co_{RMS} = 0.3$.

Results

The time history of the pressure probe at P3 is shown in Figure 7.24 for the coarse grid and the smallest time step. Results obtained with and without EIS are presented and compared to experimental results which are available for the first seven periods. The time of the first high pressure peak impact is well captured with both methods. In contrast to the experimental results, the pressure then drops significantly and negative pressure peaks are observed. These low pressure peaks are due to air bubbles that have been formed during the impact, and persist in the liquid phase. Figure 7.25 shows the temporal evolution of the free-surface during the second impact on the wall. The low pressure pulses are clearly aligned with enclosed air bubbles at the position of the pressure probe. When



(a) *CICSAM*



(b) *CICSAM + EIS*

Figure 7.24: Temporal evolution of the pressure at pressure probe P3, using 2,000 time steps per period on a grid containing 160 x 80 cells.

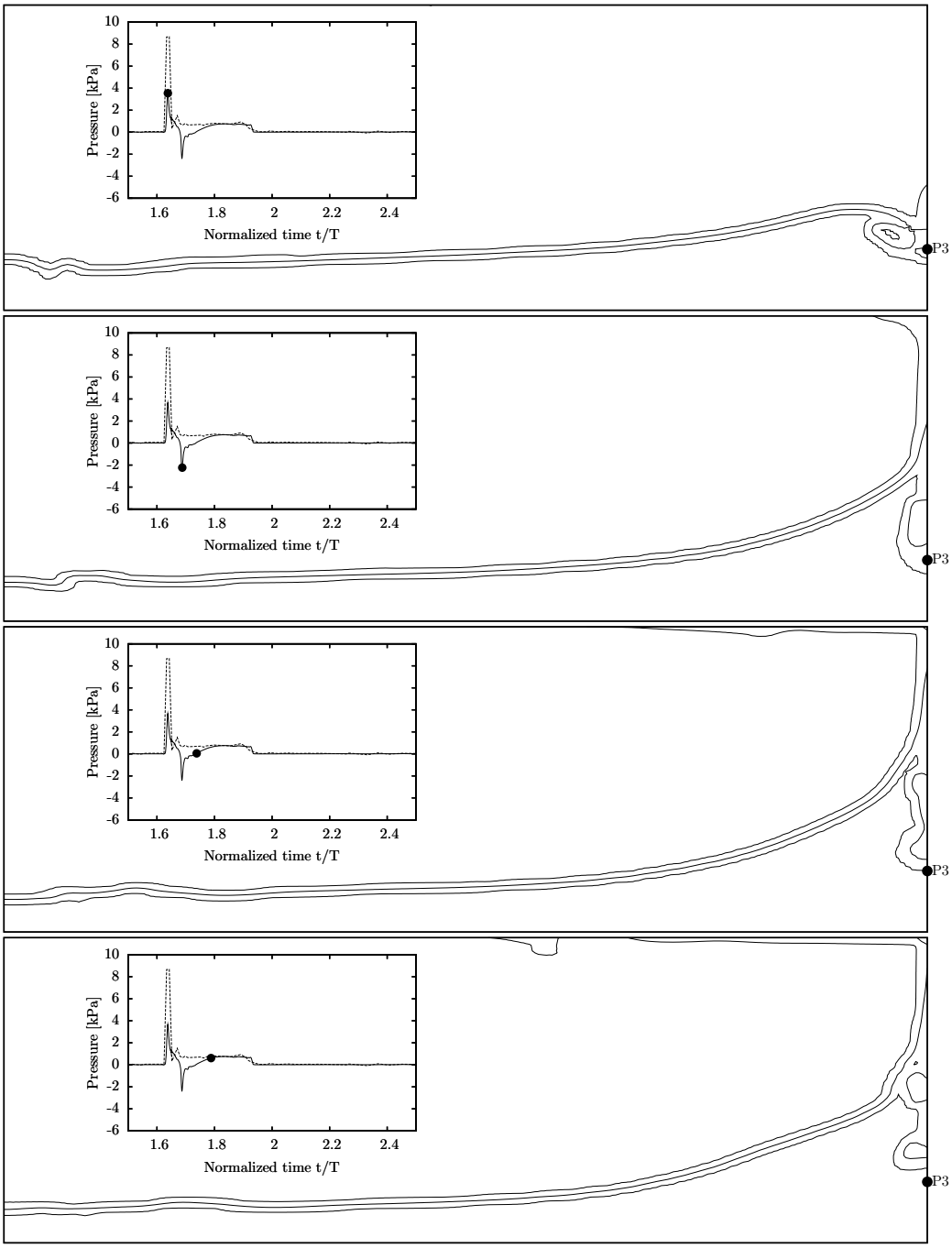


Figure 7.25: Temporal evolution of the surface during the second impact on the coarse grid using the smallest time step without EIS. (Isolines $c = 0.05; c = 0.5; c = 0.95$).

the entrapped air bubbles have passed the pressure probe, predicted pressures are in line with experimentally reported values. The difference in the predicted time for the impact grows with each period, which is in line with other published results from Perić et al. [142] and Wacławczyk et al. [195]. This effect has been assigned to partly blurred interfaces and entrapped air. Results obtained with EIS show a lower discrepancy for the predicted impact times, due to the sharper interface representation. However, the sharper interface representation does not afflict the entrapped air. Later impact events from simulations with EIS show higher pressure peaks than from simulations without EIS (Figure 7.24).

To further analyse the differences in the pressure signal obtained from the different simulations and the experiments, three quantities are assessed. The first quantity is the impact frequency, which is calculated based on the incident time of the pressure peaks. The mean pressure of an impact event, including the following plateau is evaluated as a second quantity, and the maximum pressure as the third quantity. All quantities are averaged over the last 5 periods available in experimental data. The deviations of these quantities from the experimental data are given in Table 7.7. The error levels obtained for the impact frequency on the different grids are almost identical, while the error shows a large dependency on the time step. Using EIS the error is at least one percent point smaller than in simulations without EIS. This difference grows using larger time steps. The differences observed for the other quantities are much larger. The large deviation in the mean pressure is mainly driven by the entrapped air bubbles, causing the low pressure peaks. The application of EIS reduces the error for low Courant numbers, however the errors for large Courant numbers are bigger. The underestimated peak pressures may be partly caused by different sampling rates in the experiment and the simulation. However, the entrapped air bubbles introduce a large uncertainty to the simulation results from two-dimensional simulations.

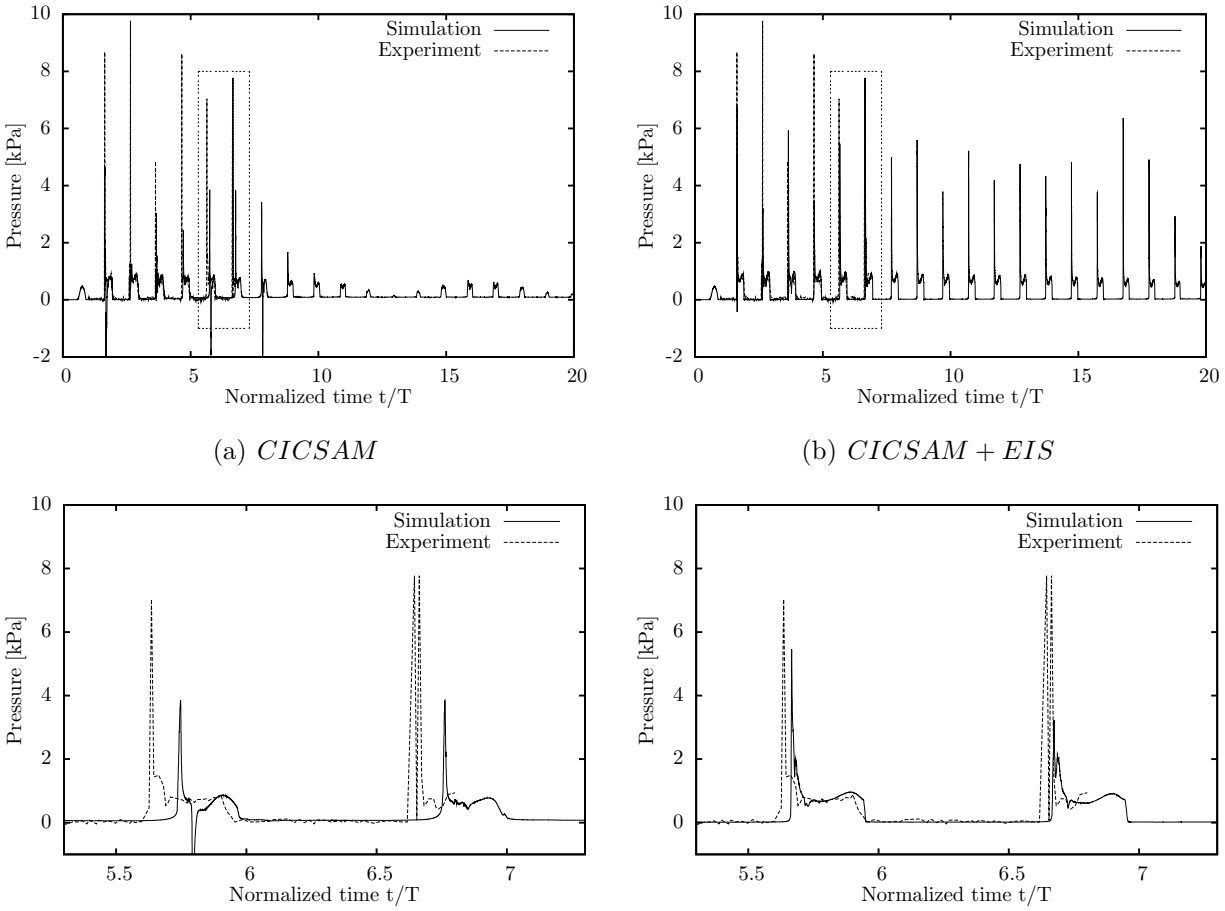
Figure 7.26 depicts the predicted pressure signal from pressure probe three using three-dimensional grids. An adaptive time step has been employed for the concentration equation which ensures that the RMS-value of the Courant number yields $Co_{RMS} = 0.3$. The pressure signal obtained from simulations using CICSAM without EIS still misses some of the later impact events. Applying EIS a nearly perfect periodic pressure signal is predicted. Furthermore, pressure undershoots are significantly reduced. The difference in the predicted impact frequency compared to the experiment is reduced from 2.33% to 0.6% as shown in Table 7.8. The reduction of the error in the other quantities is even larger. How-

Table 7.7: Errors $\left(\frac{\phi-\phi_{exp}}{\phi_{exp}}\right)$ of predicted impact pressure parameters at probe P3 for two-dimensional sloshing simulations.

	12,800 cells		28,800 cells	
Co_{mean}	CICSAM	CICSAM+EIS	CICSAM	CICSAM+EIS
Impact frequency				
0.2	3.33%	2.18%	3.19%	2.18%
0.4	4.77%	3.05%	4.05%	2.90%
0.8	7.36%	4.48%	5.92%	4.34%
Average pressure during impact				
0.2	-47.02%	-36.87%	-47.69%	-37.00%
0.4	-45.63%	-48.79%	-45.83%	-41.27%
0.8	-48.45%	-54.78%	-50.48%	-58.31%
Peak pressure during impact				
0.2	-59.03%	-45.08%	-61.41%	-41.96%
0.4	-60.63%	-61.65%	-51.32%	-52.17%
0.8	-83.72%	-85.09%	-52.17%	-77.12%

ever, the predicted mean peak pressure of all impact events is still about 36% lower than in the experiment. The accurate prediction of the peak pressure hinges on the temporal resolution of the impact and the spatial resolution at the vicinity of the pressure sensor. Both quantities may contribute to the difference but are not further studied. Furthermore, also compressibility effects, neglected within this study, may contribute to the differences.

The temporal evolution of the interface is compared for simulations obtained with and without EIS in Figure 7.27. The delay of the impact event in the simulations without EIS is clearly visible. The breaking wave reaches the side wall significantly later and shows a more chaotic surface. When the fluid rises along the side wall after the impact, the difference in the interface sharpness gets visible. The iso-surface of $c = 0.95$ reaches the top of the tank and stays above the water level of $c = 0.5$ even after the fluid has dropped back to the bottom of the tank. In general the interface looks more chaotic within the simulation without EIS.



(a) $CICSAM$ (b) $CICSAM + EIS$

(c) $CICSAM$ (d) $CICSAM + EIS$

Figure 7.26: Temporal evolution of the pressure at pressure probe three, obtained from the three-dimensional simulation.

Table 7.8: Errors $\left(\frac{\phi - \phi_{exp}}{\phi_{exp}}\right)$ of predicted impact pressure parameters at probe P3 for three-dimensional sloshing simulations.

	CICSAM	CICSAM+EIS
Impact frequency	2.33%	0.60%
Average pressure	-38.72%	-3.61%
Peak pressure	-60.69%	-36.04%

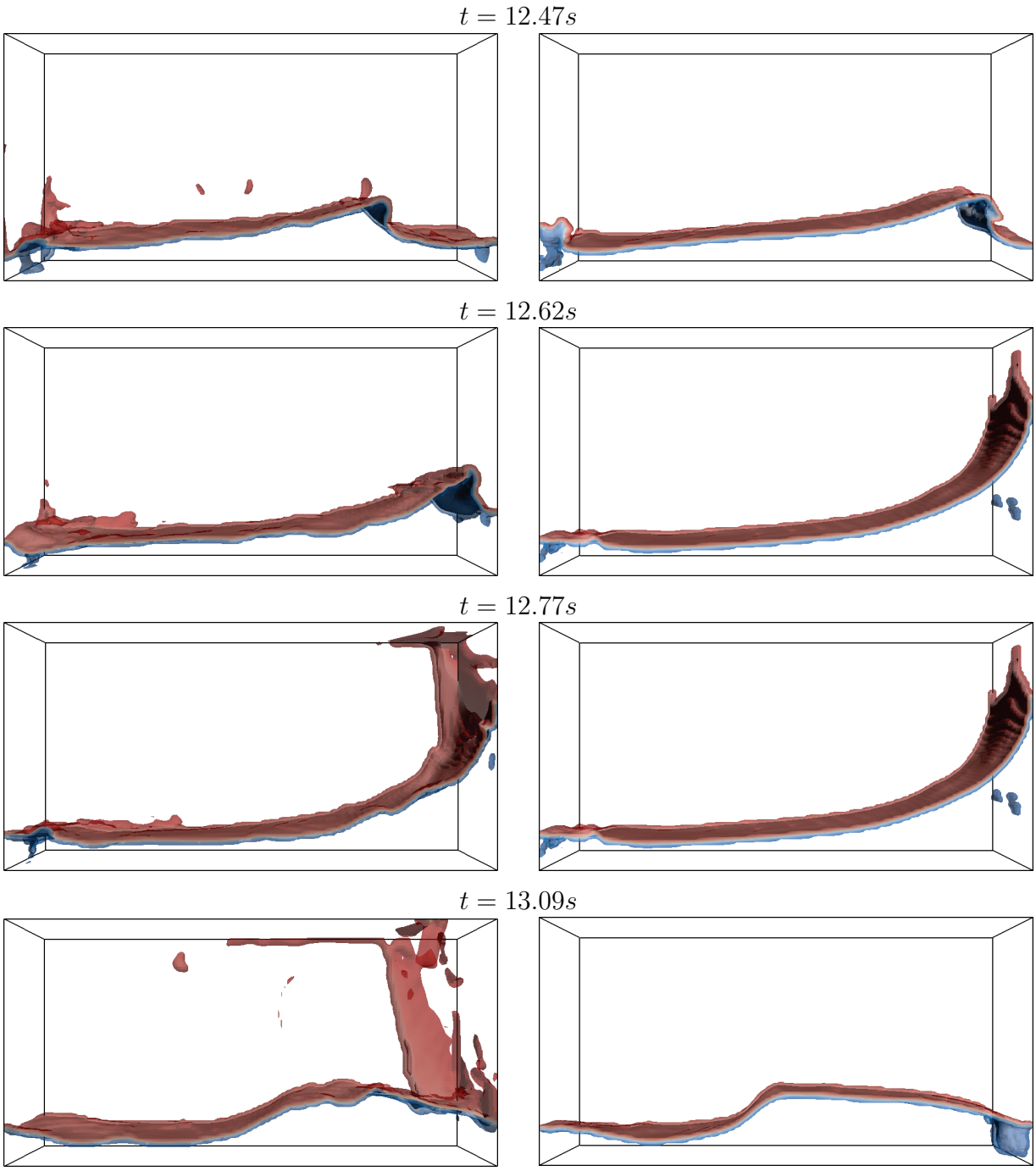


Figure 7.27: Temporal evolution of the interface obtained for simulations with and without EIS on the three-dimensional grid. Iso-surfaces of $c = 0.05$,$c = 0.5$ and $c = 0.95$ are shown.

7.4 Automatic Grid Refinement

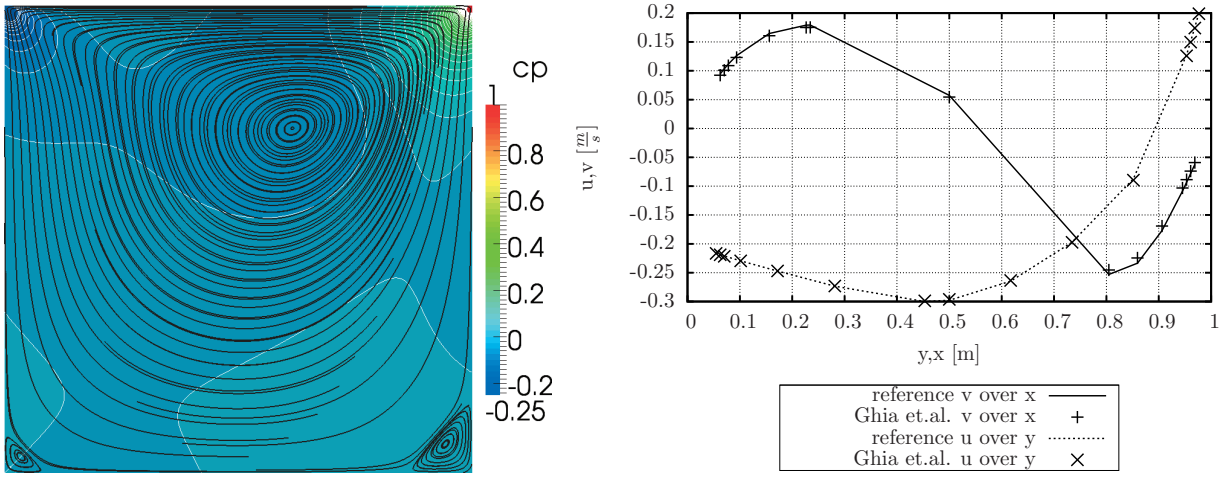
7.4.1 Lid Driven Cavity Flow

A widely used simple 2-dimensional benchmark test-case for CFD methods is the lid driven cavity flow at low Reynolds numbers. The top of a squared 2-dimensional domain has a constant tangential velocity as boundary condition, while the sides and the bottom are simulated as no-slip walls. The lid driven cavity flow for this study is examined at a Reynolds number of 100 giving a laminar steady flow. The flow is studied on systematically refined grids starting with 30 x 30 cells using up to 240 x 240 cells. To minimise the influence of the iterative error the residuals have been reduced by at least 9 orders for all equations. Results for the finest grid are shown in Figure 7.28 and compared against results published by Ghia et.al. [57]. A primary vortex displaced from the middle of the squared domain and two secondary vortices in the lower corners are observed. The pressure field has high gradients in the upper corners of the domain, while it is smooth in the other parts of the domain. The results published by Ghia et.al. are the vertical velocities on a horizontal line in the centre of the domain and the horizontal velocities on a vertical line in the centre of the domain. Results obtained with FreSCo$^+$ on the fine grid agree fair with those published by Ghia et.al. as shown in Figure 7.28.

In the following the result obtained on the finest grid is used as reference result. The errors of the velocity components in horizontal (u), vertical direction (v) and the pressure are evaluated for all cases in the grid refinement study. The results obtained on the automatically refined grids are interpolated to the reference grid using a second-order accurate interpolation method, and the L1, L2 and L∞ norm of the local errors $\phi_{ref} - \phi$ are computed.

Grid Refinement Studies

Grids with different resolutions have been used as starting point for the simulations to investigate the sensitivity of the refinement criterion towards the grid resolution. The maximum number of refinement steps has been varied and the highest number of allowed refinement steps leads to cells which have the same size as the cells of the reference grid. The residual error estimator for the velocity components has been used as refinement criterion for all simulations using a threshold of $\mathcal{T} = 0.001$.



(a) Streamlines and pressure coefficient contour.

(b) Comparison to results of Ghia et al. [57]

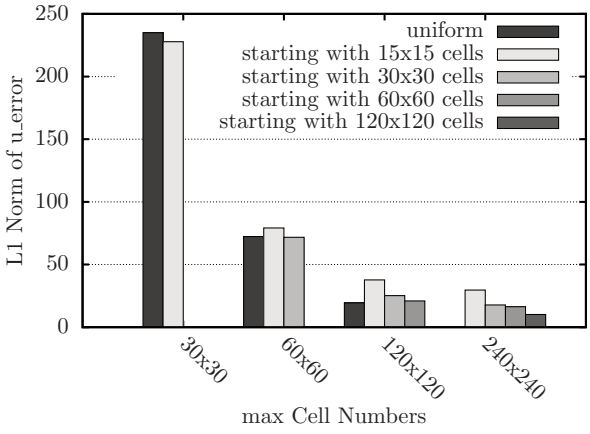
Figure 7.28: Results for the 2D lid driven cavity flow obtained on finest grid (240 x 240 cells) which are used as reference solution.

Isotropic Refinement in Serial Computations

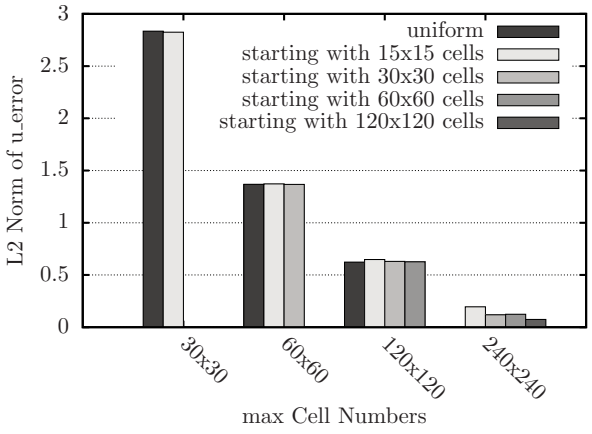
Figure 7.29 shows results from serial computations with isotropic refinement. The bar marked with 'uniform' indicates the error for an uniformly refined grid towards the finest grid. The other bars show results from automatic grid refinement starting with different grid resolutions compared to the reference solution. Ideally the error originating from simulations with automatic grid refinement should be minimised to the same level as the error from the reference computation using the same number of refinement steps. This behaviour is observed for the error of the horizontal velocities shown in Figure 7.29 in both error norms, while the error for the vertical velocities in the L2 norm is even slightly lower than the reference error.

The pressure has not been used as an error sensor, anyhow the error drops using grid refinement as depicted in Figure 7.29. Regarding the L1 norm the error drops in the same way as the error of the reference simulations, while the error in L2 norm is higher than in the reference simulations.

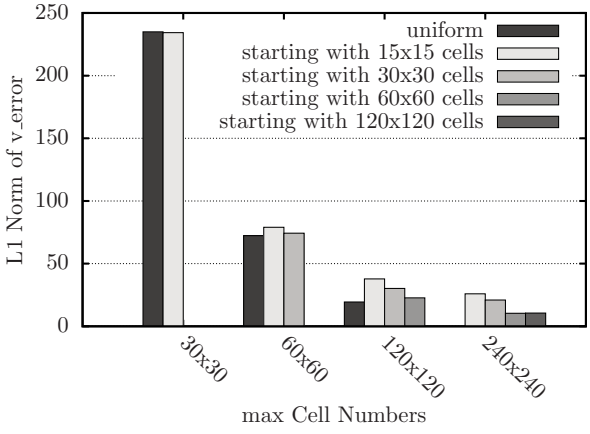
Figure 7.30 depicts grids obtained from automatic grid refinement starting with different grid resolutions. It is obvious that for this case the refinement criterion works independent



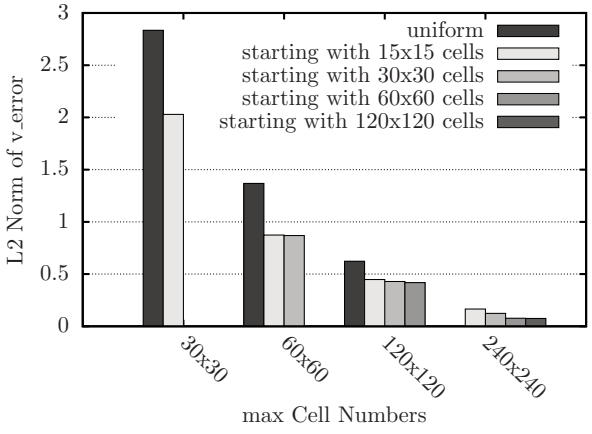
(a) $||u_{ref} - u||_1$



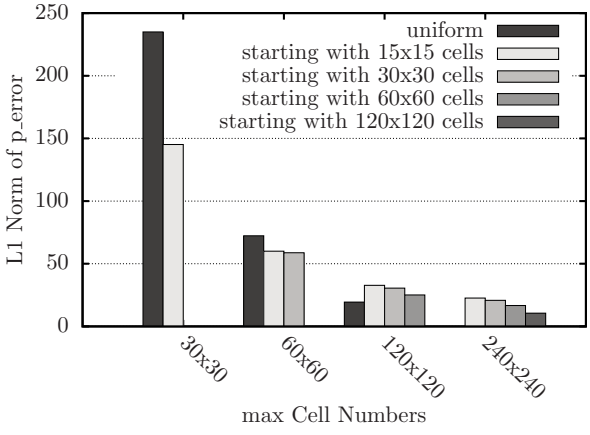
(b) $||u_{ref} - u||_2$



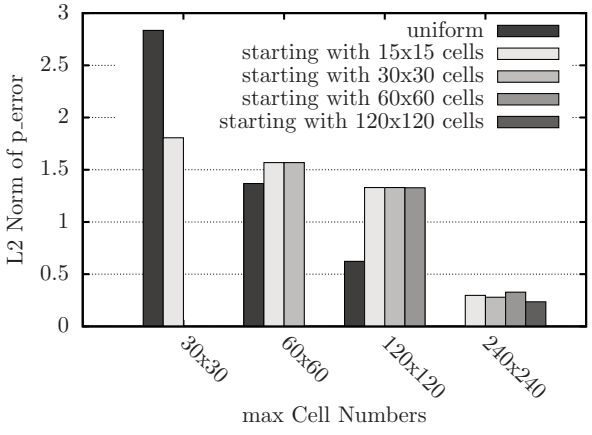
(c) $||v_{ref} - v||_1$



(d) $||v_{ref} - v||_2$



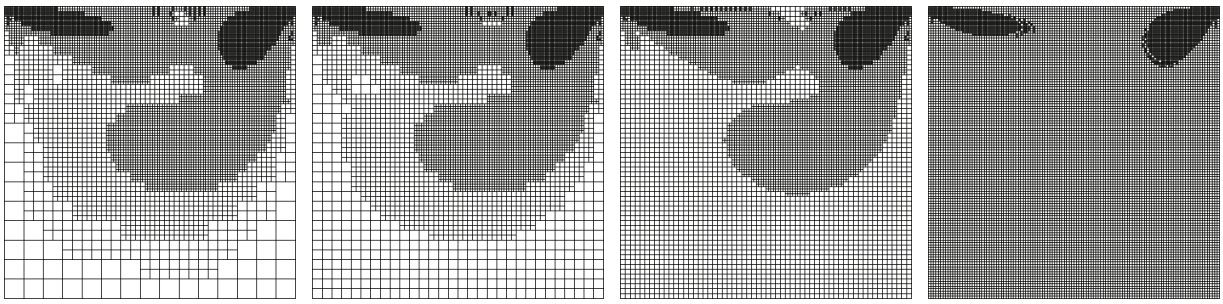
(e) $||p_{ref} - p||_1$



(f) $||p_{ref} - p||_2$

Figure 7.29: Errors for u, v, p in L1 and L2 norm for isotropic refinement in serial computations.

of the initial grid density, which is favourable.



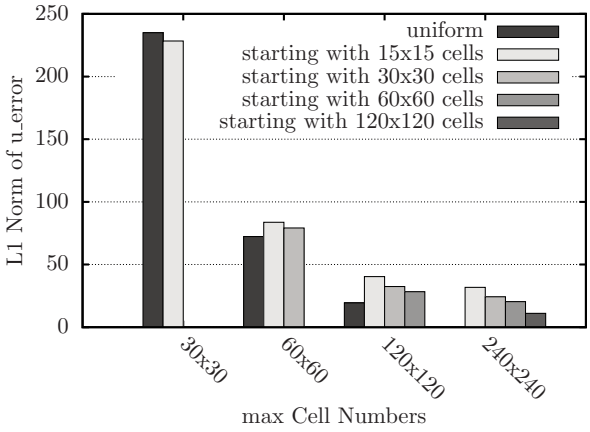
(a) Starting with 15x15 cells. (b) Starting with 30x30 cells. (c) Starting with 60x60 cells. (d) Starting with 120x120 cells.

Figure 7.30: Grids after isotropic refinement starting with different resolutions in serial computations.

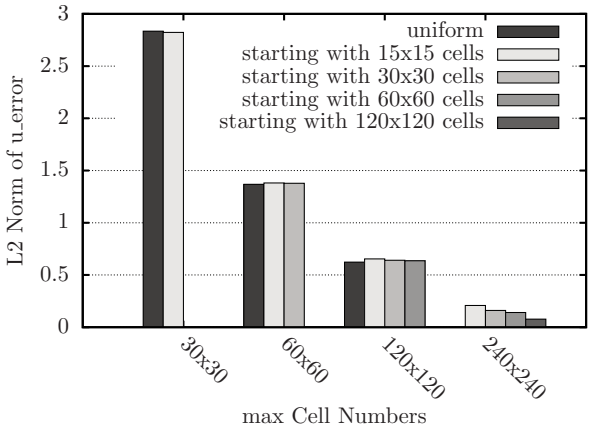
Anisotropic Refinement in Serial Computations

The same simulations have been carried out using anisotropic refinement. Figure 7.31 shows the estimated errors obtained with anisotropic refinement. The behaviour in all error norms for all quantities is similar to the behaviour observed for isotropic refinement. This indicates that the error indicators for anisotropic refinement work as expected.

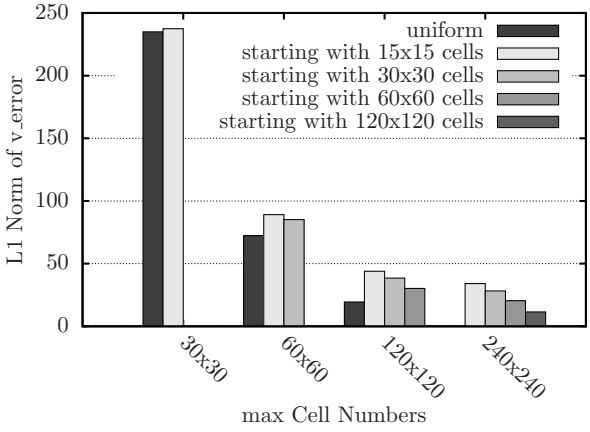
The resulting grids shown in Figure 7.32 are quite similar to those obtained from isotropic refinement. However, some slight anistropically refined regions are visible.



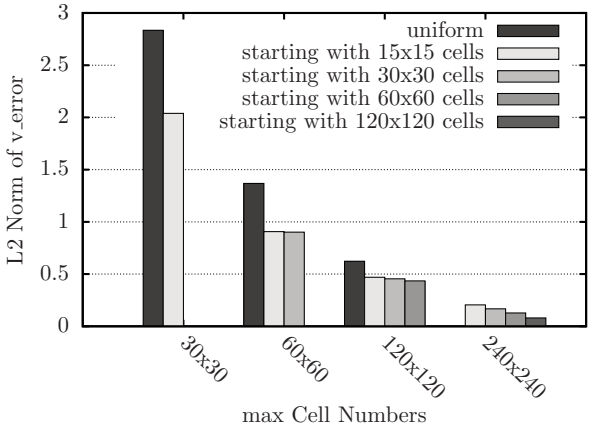
(a) $||u_{ref} - u||_1$



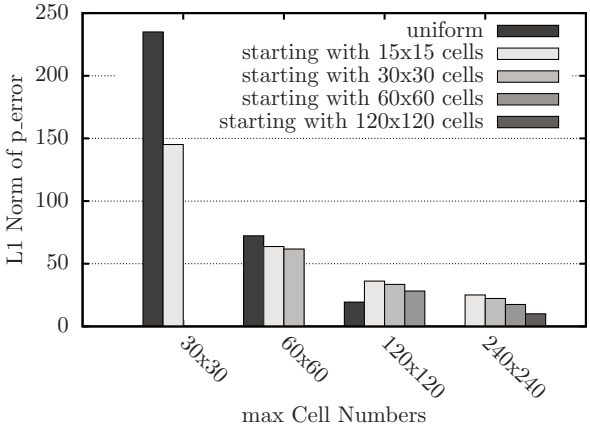
(b) $||u_{ref} - u||_2$



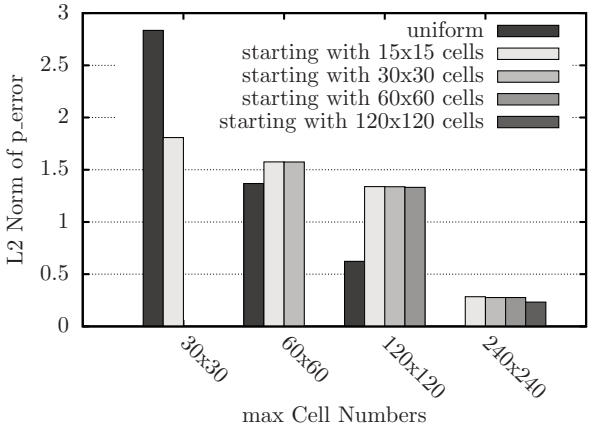
(c) $||v_{ref} - v||_1$



(d) $||v_{ref} - v||_2$

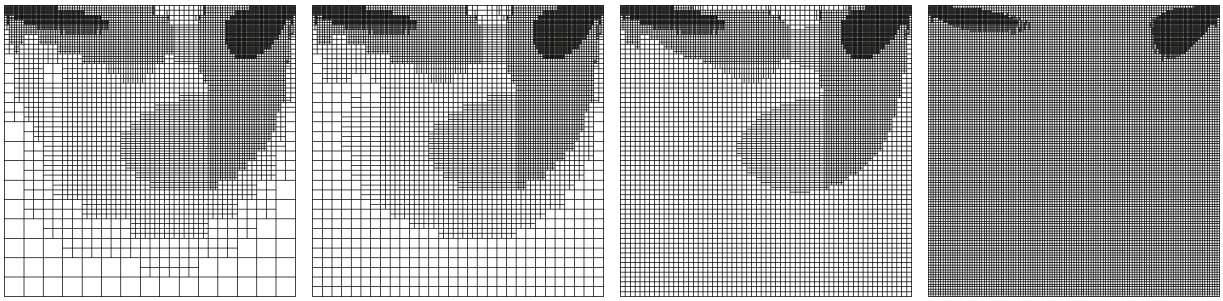


(e) $||p_{ref} - p||_1$



(f) $||p_{ref} - p||_2$

Figure 7.31: Errors for u, v, p in L1 and L2 norm for anisotropic refinement in serial computations.



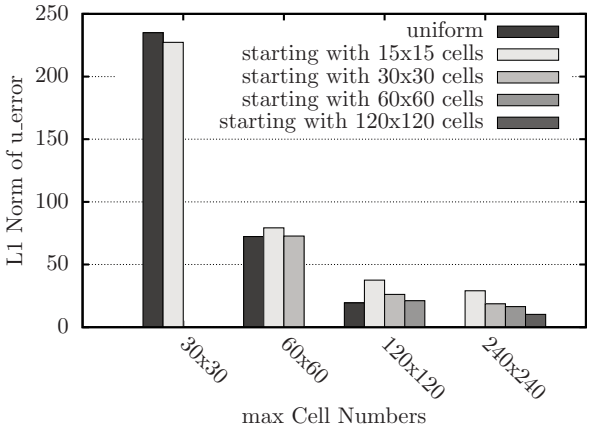
(a) Starting with 15x15 cells. (b) Starting with 30x30 cells. (c) Starting with 60x60 cells. (d) Starting with 120x120 cells.

Figure 7.32: Grids after anisotropic refinement starting with different resolutions in serial computations.

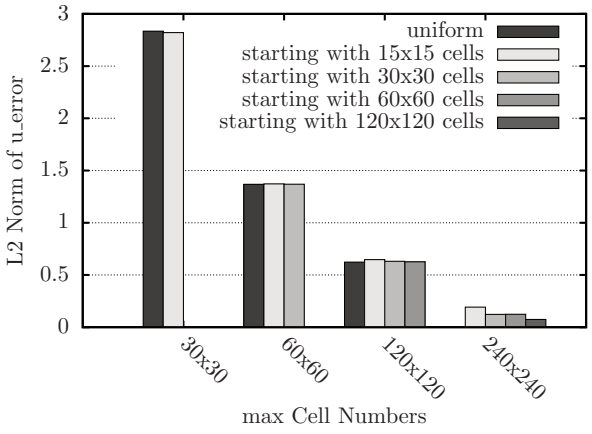
Isotropic Refinement in Parallel Computations

To verify the developed automatic grid-adaptation method for parallel simulations, the domain has been decomposed into 10 parts. The errors are expected to be exactly the same as for the serial simulation to prove that the algorithm works in parallel. Figure 7.33 presents the errors obtained from these simulations. They are identical to those obtained from serial simulations and thus prove that the algorithm works as expected.

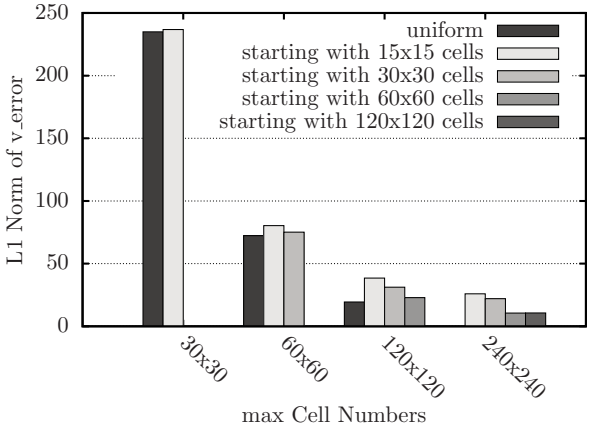
The resulting numerical grids starting with different grid densities are shown in Figure 7.34. The colours indicate the partitioning of the grid. The grids are identical to those obtained with serial simulation and no disturbing influence from the domain borders is visible.



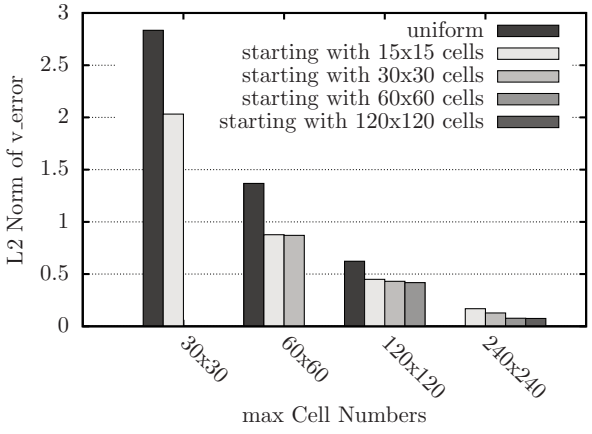
(a) $||u_{ref} - u||_1$



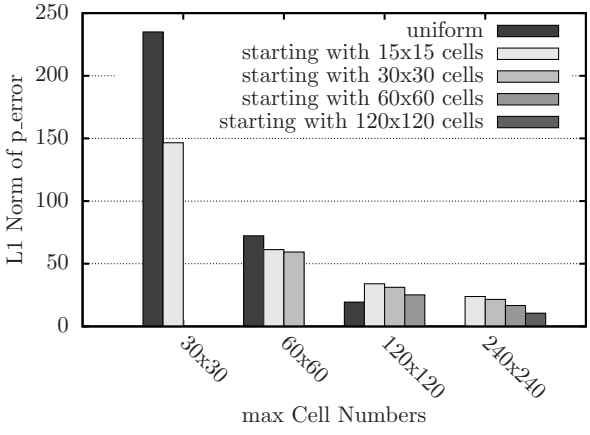
(b) $||u_{ref} - u||_2$



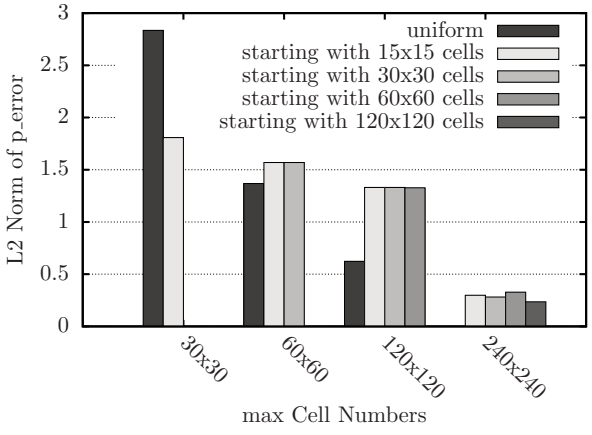
(c) $||v_{ref} - v||_1$



(d) $||v_{ref} - v||_2$

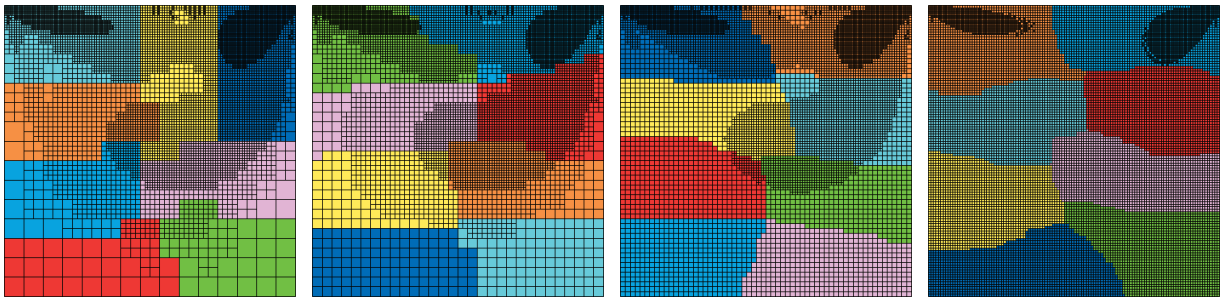


(e) $||p_{ref} - p||_1$



(f) $||p_{ref} - p||_2$

Figure 7.33: Errors for u, v, p in L1 and L2 norm for isotropic refinement in parallel computations.



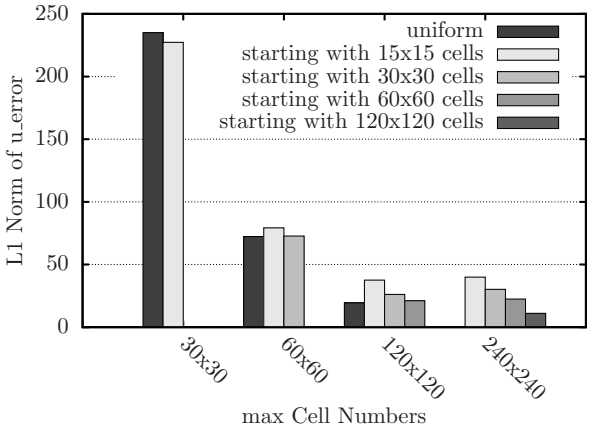
(a) Starting with 15x15 cells. (b) Starting with 30x30 cells. (c) Starting with 60x60 cells. (d) Starting with 120x120 cells.

Figure 7.34: Grids after isotropic refinement starting with different resolutions in parallel computations.

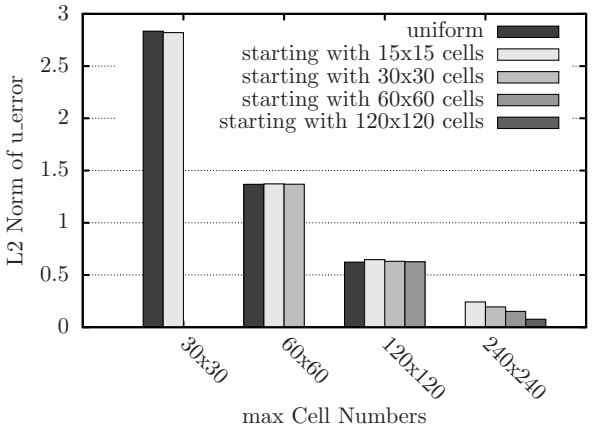
Anisotropic Refinement in Parallel Computations

Parallel simulations using anisotropic refinement have been carried out to verify the correctness of the developed algorithm for anisotropic refinement in parallel simulations. The domain has been split into 10 parts. The errors are expected to be very close to those from the serial simulations using anisotropic refinement. Figure 7.35, which depicts the errors, reveals that the expected behaviour is found.

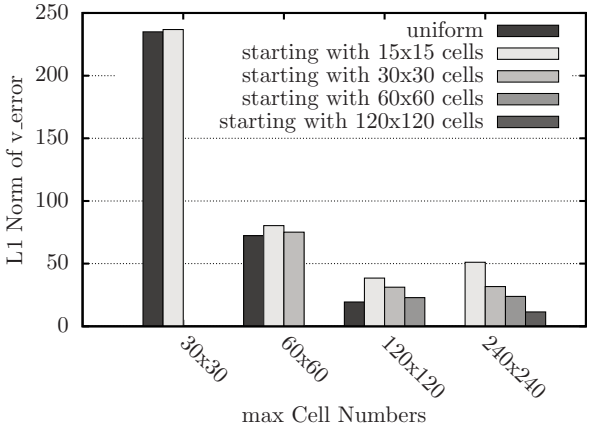
The resulting grids shown in Figure 7.36 are identical to those obtained with serial computations and anisotropic refinement. The simulations have shown that the developed algorithm works as expected.



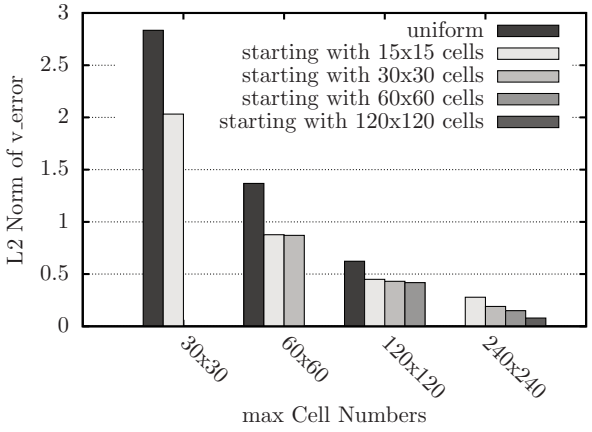
(a) $||u_{ref} - u||_1$



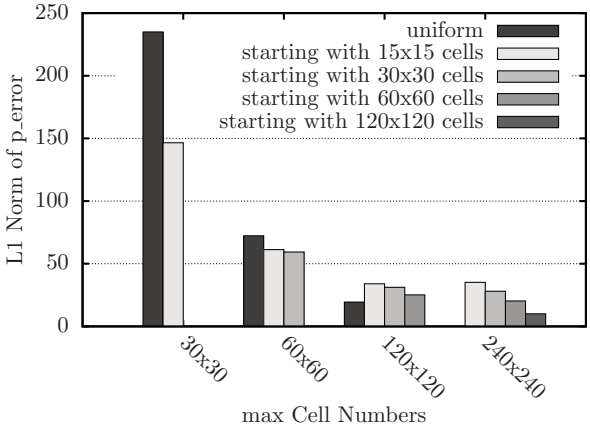
(b) $||u_{ref} - u||_2$



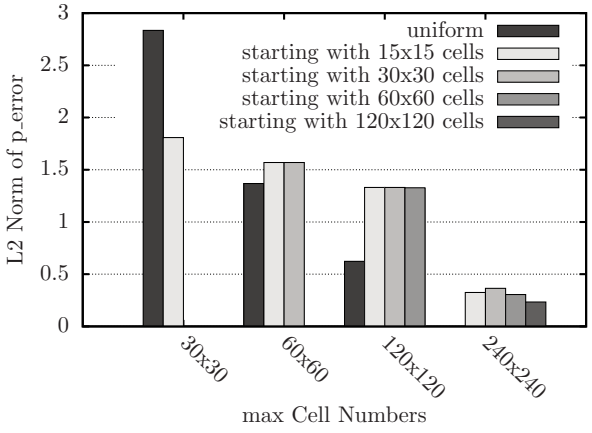
(c) $||v_{ref} - v||_1$



(d) $||v_{ref} - v||_2$

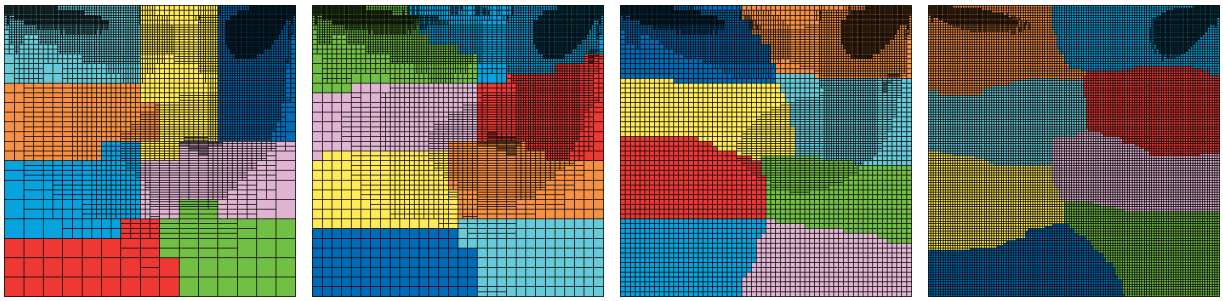


(e) $||p_{ref} - p||_1$



(f) $||p_{ref} - p||_2$

Figure 7.35: Errors for u, v, p in L1 and L2 norm for anisotropic refinement in parallel computations.



(a) Starting with 15x15 cells. (b) Starting with 30x30 cells. (c) Starting with 60x60 cells. (d) Starting with 120x120 cells.

Figure 7.36: Grids after anisotropic refinement starting with different resolutions in parallel computations.

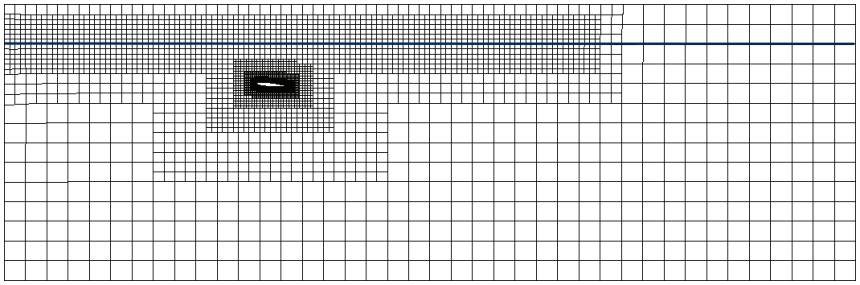
7.4.2 Duncan Foil

To test the basic capabilities of the grid-adaptation technique for free-surface flows, the Duncan foil test case is used.

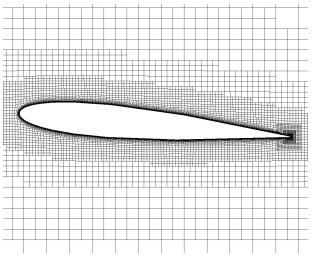
Case Description

The detailed flow conditions for this test cases are given in Section 7.2.1. The numerical grid, used as starting point, is shown in Figure 7.37. Obviously the grid resolution is insufficient in the free-surface region, however the grid is used as a starting point for automatic grid-adaptation, which will enrich the grid in the free-surface region. The target y-plus value is below one, therefore many cells are clustered in the vicinity of the foil. The base grid consists of about 9,000 cells.

Two refinement indicators have been used within this study. The first indicator is the free-surface criterion, while the second indicator is the free-surface criterion supplemented by the residual error estimator for the velocity. The refinement threshold for the free-surface criterion has been set to 0.15 and the threshold for the residual error estimator to 0.01. The maximum number of refinement steps for both criteria have been varied. Anisotropic refinement has been used for this study because it is the natural choice for free-surface flows. To counteract the stability issues associated to the varying Courant number due to grid-adaptation, the time step has been automatically adjusted during the



(a) Complete computational grid and domain for Duncan Foil. Bold blue line indicates the undisturbed free-surface.



(b) Grid in the near vicinity of the foil.

Figure 7.37: Base grid.

simulation to a time step, which gives a maximum Courant number of 0.3.

The predicted free-surface elevation has been compared to the experiments from Duncan [47]. To quantify the deviation from the experiments, the experimental and numerically predicted free-surface shapes have been fitted to a function of a damped oscillation:

$$z^*(x^*) = A^* \exp\left(-\beta x^*\right) sin\left[\left(\sqrt{\left(\frac{2\pi}{l^*}\right)^2 - \beta^2}\right) x^* - b^* \frac{2\pi}{l^*}\right] \tag{7.16}$$

Herein, $z^* = \frac{Elevation}{c}$ is the normalised free-surface elevation, $x^* = \frac{x}{c}$ the normalised ordinate, $A^* = \frac{A}{z}$ the normalised oscillation amplitude, β the damping coefficient, $l^* = \frac{l}{c}$ the normalised wave length and $b^* = \frac{b}{c}$ the normalised phase shift. At its creation point the wave does not follow the shape of a damped oscillation, due to strong non-linear effects during the generation of the wave. Hence, for the curve fitting only the data after the first positive maximum, which occurs approximately at $x^* \approx 2$ is used.

Results

Figure 7.38 shows the predicted free-surface elevation from simulations with the free-surface sensor as given in Section 6.3.1 as adaptation criterion, compared to experimental results. It is clearly seen that the refinement in the free-surface region drastically improves the quality of the predicted free-surface elevation. The number of cells for each grid and the associated predicted curve fit coefficients compared to the experiment are provided in Table 7.9. The predicted amplitude coefficient converges to a deviation of 3% from the coefficient obtained from the experimentally predicted free-surface. However, the convergence is not

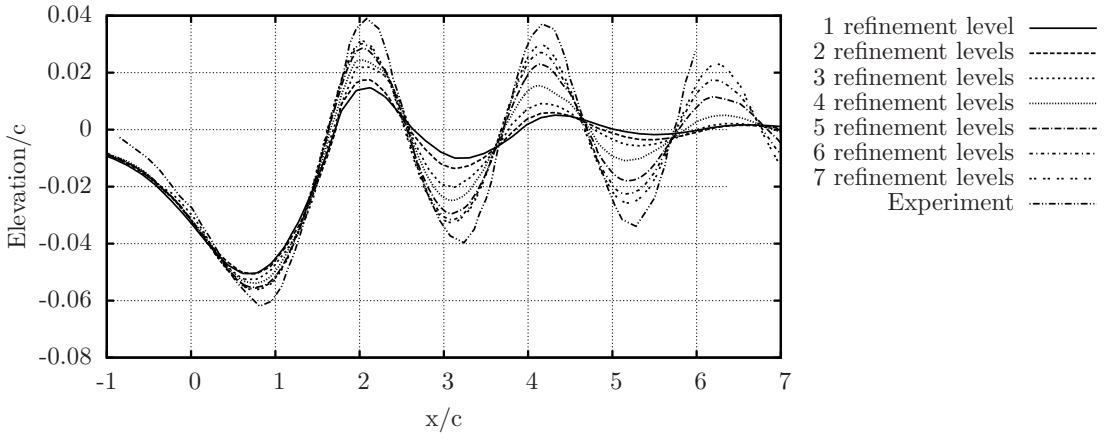


Figure 7.38: Wave elevation above foil for variation of the number of refinement steps compared to experiment.

monotonic. Up to a maximum refinement level of three the deviation from the experiment grows. This is attributed to the high damping coefficient, which is about 20 times larger than in the experiment for these maximum refinement levels. The predicted normalised wave length converges monotonically up to a deviation of 1% compared to the experiment. Large deviations of more than 250% remain for the predicted damping coefficients. Even though most coefficients converge monotonically no grid convergence has been achieved for the free-surface elevation using seven refinement steps. The approximate local resolution of the wave length and height for the seventh refinement reads 60 cells per wave height respectively 230 cells per wave length. Simulations with eight or more refinement steps were afflicted with stability issues and did not converge. Due to the anisotropic refinement the number of cells increases only moderately. Figure 7.39 shows some details of the finest grid.

Using the free-surface criterion as refinement indicator, errors in the region between the foil and the free-surface are ignored. However, in this region the wave is generated and errors may have a large effect on the predicted surface shape. To identify these regions two different refinement criteria are combined. A refinement indicator based on the residual error estimator for the velocity is added below the surface within a prescribed rectangular box, shown in Figure 7.40. Additionally the free-surface refinement indicator is active with a maximum number of seven refinement steps. Figure 7.41 shows the predicted free-surface evaluation for a variation of the allowed number of refinement steps for the

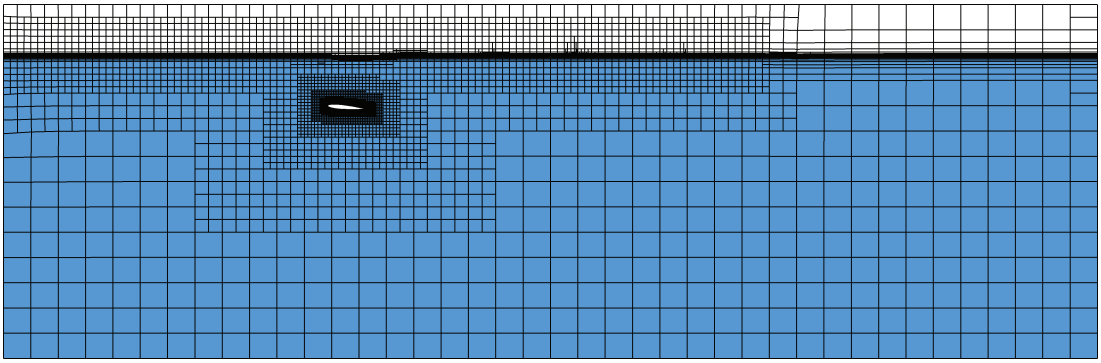
Table 7.9: Final number of cells for variation of the number of refinement steps and curve fit parameters compared to experimental results.

		Curve Fit Parameter		
Refinement Steps	Cells	A^*	l^*	β
Experiment		0.0403	2.087	0.0254
1	11,254	0.0498 [+23%]	2.373 [+14%]	0.5218 [+1951%]
2	11,659	0.0642 [+59%]	2.347 [+12%]	0.5229 [+1955%]
3	12,139	0.0862 [+114%]	2.236 [+7%]	0.5058 [+1888%]
4	12,822	0.0606 [+50%]	2.162 [+4%]	0.3309 [+1201%]
5	14,110	0.0519 [+29%]	2.142 [+3%]	0.2100 [+725%]
6	16,666	0.0463 [+15%]	2.126 [+2%]	0.1429 [+462%]
7	23,459	0.0415 [+3%]	2.119 [+1%]	0.0919 [+261%]

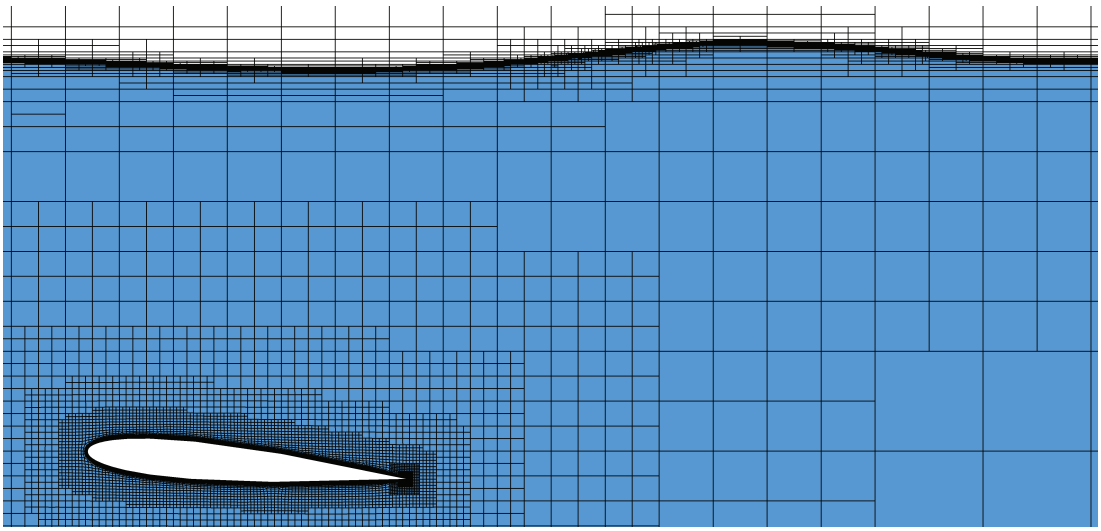
residual error estimator. It can be seen that the graphs for the predicted free-surface from simulations using 3 and 4 refinement levels for the residual error estimator are on top of each other. They show significant improvements of the predicted free-surface towards the experimental results, compared to the simulations using only the free-surface criterion as indicator. The resulting grid for three refinement steps of the residual error estimator is shown in Figure 7.42. Cells have been added near the stagnation point of the profile and below the crests and troughs of the predicted free-surface elevation. Table 7.10 shows the deviations for the curve fit coefficients associated to the predicted free-surface shapes with the combined refinement criteria. The inclusion of the residual error estimator in the region where the wave is generated increases the predictive accuracy. This essentially supports the importance of an adequate resolution of the orbital motion. The deviations for the predicted wave amplitude and wave length are below 1% and the deviation in the predicted damping is further reduced. Results obtained with 3 and 4 maximum refinement steps for the residual error estimator are almost identical.

Table 7.10: Final number of cells for variation of the number of refinement steps for the residual error estimator and curve fit parameters compared to experimental results. (FS=Free Surface Criterion, REE=Residual Error Estimator Criterion)

Refinement Steps		Cells	Curve Fit Parameter		
FS	REE		A^*	l^*	β
Experiment			0.0403	2.087	0.0254
7	0	23,459	0.0415 [+3%]	2.119 [+1%]	0.0919 [+261%]
7	2	31,308	0.0429 [+6%]	2.102 [+1%]	0.0753 [+196%]
7	3	34,278	0.0403 [+0%]	2.088 [+0%]	0.0496 [+95%]
7	4	38,445	0.0404 [+0%]	2.081 [+0%]	0.0499 [+96%]



(a) Overview of adapted grid.



(b) Details of the grid in the free-surface and close to the foil.

Figure 7.39: Grids for Duncan foil simulation after 7 refinement steps using the free-surface criterion.

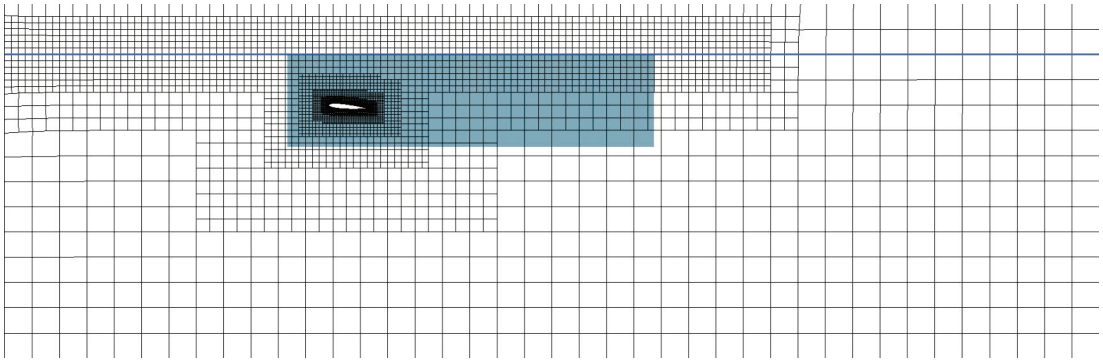


Figure 7.40: The shaded box outlines the region in which additional error-estimator based refinement is allowed.

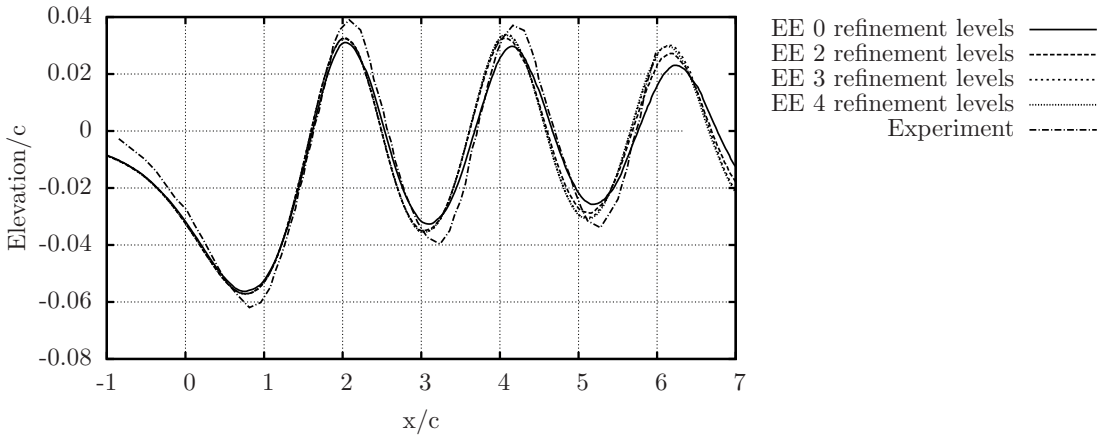
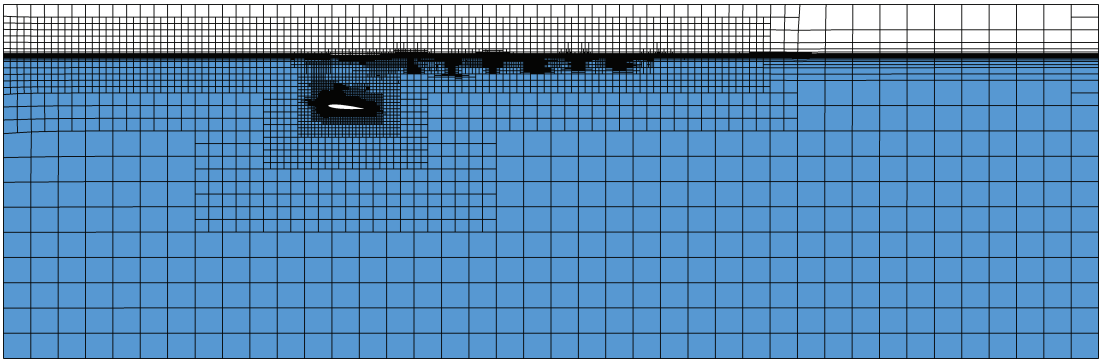
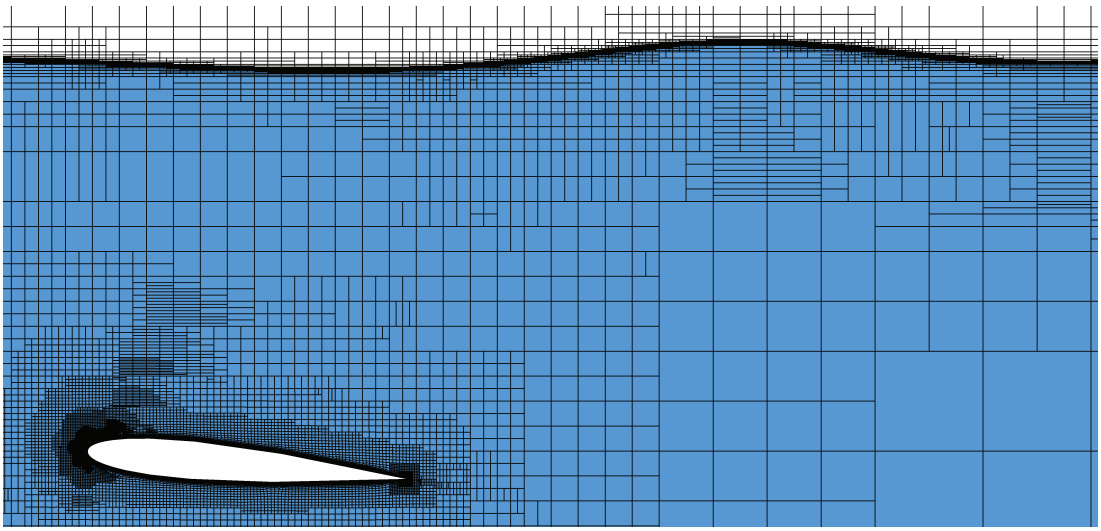


Figure 7.41: Wave elevation above foil for seven refinement steps in free-surface and variation of the number of refinement steps for the error estimator compared to experiment.



(a) Overview of adapted grid.



(b) Details of the grid in the free-surface and close to the foil.

Figure 7.42: Grids for Duncan foil simulation after 7 refinement steps using the free-surface criterion and 3 refinement steps for the error estimator.

7.4.3 DFG Benchmark Flow

The two-dimensional laminar flow around a cylinder in a channel has been presented as a benchmark testcase by Schäfer et al. [164]. The report contains results from 15 groups for the steady flow at Reynolds number $Re = 20$, obtained from finite element, finite differences and finite volume methods on structured and unstructured grids. The benchmark case has been previously used by Becker [20] to validate a goal-oriented mesh adaptation strategy.

Case Description

Figure 7.43 shows the geometrical setup and applied boundary conditions. To stabilise the

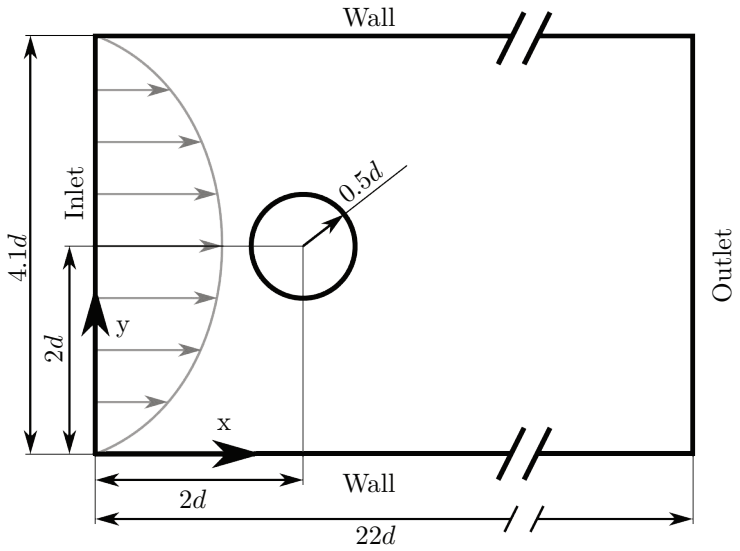


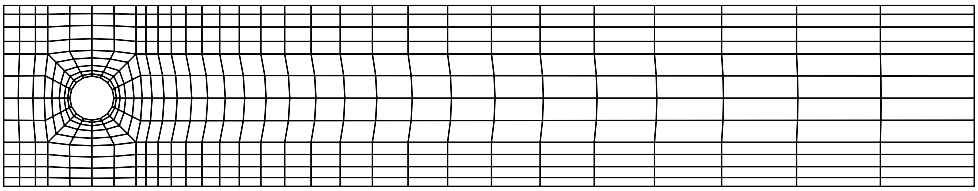
Figure 7.43: Definition of DFG flow around cylinder validation case.

flow the cylinder is place slightly asymmetrically. The Reynolds number of the considered flow reads $Re = 20$ based on the mean velocity $u_{mean} = 0.2\frac{m}{s}$ of the parabolic inflow profile given by

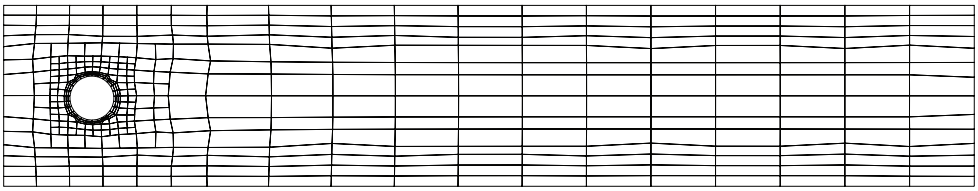
$$u_1(0, y) = 6u_{mean}\frac{y(H - y)}{H^2}, \quad u_2(0, y) = 0. \tag{7.17}$$

The 2D drag $c_D = \frac{2F_D}{\rho u_{mean}^2 D}$ and lift $c_L = \frac{2F_L}{\rho u_{mean}^2 D}$ coefficient as well as the pressure difference between two points – (0.15m,0.2m) and (0.25m,0.2m) – are provided as reference solutions in [164].

The developed goal-oriented grid-adaptation technique has been applied to the channel flow focussing on the drag as output functional. Results are compared to results from formal grid convergence studies using different grid types. The initial grids, which are the base for the grid convergence study and the grid refinement procedures, are depicted in Figure 7.44. Convective fluxes have been discretised with QUICK, and the residual has



(a) Block-structured grid (412 cells).



(b) Unstructured grid (423 cells).

Figure 7.44: Initial grids for the DFG Channel Flow Benchmark.

been reduced by 6 orders of magnitude.

Results

The formal order of accuracy for initially block-structured and unstructured grids is evaluated using the Least Squares version of the grid convergence index proposed by Eça and Hoekstra [50]. Within the grid convergence study five grid levels with nested refinement leading to $105,472$ cells for the block-structured and $108,288$ cells for the unstructured grid have been used. Table 7.11 shows the estimated order of accuracy, extrapolated drag coefficient and associated uncertainty for the block structured grid. Schäfer et al. [164] define the valid range for the predicted drag coefficient to be within $5.57 < c_D < 5.59$, based on the benchmark results. The extrapolated drag coefficient shows a deviation of about 1% from the valid range, although the predicted drag coefficients and associated uncertainties from the different sets are consistent with each other. The predicted order

Table 7.11: Grid convergence study for DFG-Channel benchmark using a block-structured grid.

Set	Convergence	Extrapolated c_D	Order	Uncertainty
1-3	monotonic	5.521	1.734	8.15×10^{-2} (1.479%)
1-4	monotonic	5.523	1.750	2.55×10^{-2} (0.458%)
1-5	monotonic	5.521	1.735	1.00×10^{-2} (0.179%)
2-4	monotonic	5.526	1.819	2.15×10^{-2} (0.380%)
2-5	monotonic	5.520	1.707	1.10×10^{-2} (0.204%)
3-5	monotonic	5.513	1.284	1.55×10^{-2} (0.280%)

of accuracy is close to 2 and within the expected range. The initial grids are depicted in Figure 7.44. The results from the grid convergence study of the unstructured grid are presented in Table 7.12. The predicted order of accuracy is in the same range as for the

Table 7.12: Grid convergence study for DFG-Channel benchmark using an unstructured grid.

Set	Convergence	Extrapolated c_D	Order	Uncertainty
1-3	monotonic	5.556	1.702	5.25×10^{-2} (0.941%)
1-4	monotonic	5.556	1.701	1.60×10^{-2} (0.291%)
1-5	monotonic	5.555	1.690	0.60×10^{-2} (0.110%)
2-4	monotonic	5.556	1.698	1.60×10^{-2} (0.291%)
2-5	monotonic	5.554	1.647	0.65×10^{-2} (0.119%)
3-5	monotonic	5.552	1.454	0.75×10^{-2} (0.137%)

block-structured grid. The extrapolated drag coefficient is slightly higher and approximately 0.3% lower than the valid range.

Although the simulations on both grid types show a monotonic grid convergence behaviour, the extrapolated drag coefficients are afflicted with a difference of about 0.5%, which is larger than the estimated uncertainty. Hence, the grid type (structured / unstructured) has a significant influence on the results. When local grid-adaptation is applied any structured or block-structured grid is changed to an unstructured grid. Furthermore,

local refinement of an unstructured grid usually introduces additional hanging nodes. The differences may be also afflicted by the different initial discretisation/resolution of the cylinder and the different alignment of the cells to the streamlines. Therefore, in the following local grid refinement studies a deviation in the order of 1% from the extrapolated solution is deemed to be acceptable, based on the previous observations.

The predicted drag coefficient from goal-oriented grid refinement simulations with different refinement thresholds is shown in Figures 7.45 to 7.48 for initially block-structured grids. The hatched area shows the valid range according to Schäfer et al. [164] and the shaded area is the area covered by the extrapolated result plus/minus the refinement tolerance. For the highest refinement threshold 0.25 the target area is already reached after one adaptation step. Since no stopping criterion is implemented for the goal-oriented refinement the refinement procedure proceeds. However, the number of new cells in the next step is significantly lower, than for the stricter refinement thresholds. The correction of the functional according to (6.11) increases the accuracy of the prediction. Surprisingly the corrected functional converges to a value, which is inside the reference solution area. A higher refinement threshold leads to more additional cells and the functional approaches the target area faster. Comparing the predicted drag coefficients for different refinement thresholds the achievable drag coefficient converges to a value higher than the extrapolated drag coefficient from the grid refinement study.

The same refinement thresholds have been applied to the goal-oriented grid-adaptation study based on the initially unstructured grid. The predicted drag values, depicted in Figures 7.49 to 7.52 show the same behaviour as for the block-structured grid. Interestingly, the predicted drag coefficient without correction rises with increasing numbers of cells using the stricter refinement thresholds, but the corrected functional converges to a value within the valid range.

The resulting grids are shown in Figure 7.53 for the highest refinement threshold after the second adaptation step for both grid types. In both cases the grid has been refined in the vicinity of the cylinder, at the inlet and at the outlet. The refinement near the inlet is more intense close to the channel walls. Furthermore, some refinement at the locations where the grid quality is poor is visible, especially where bad angles and high volume ratios are present in the initial grids.

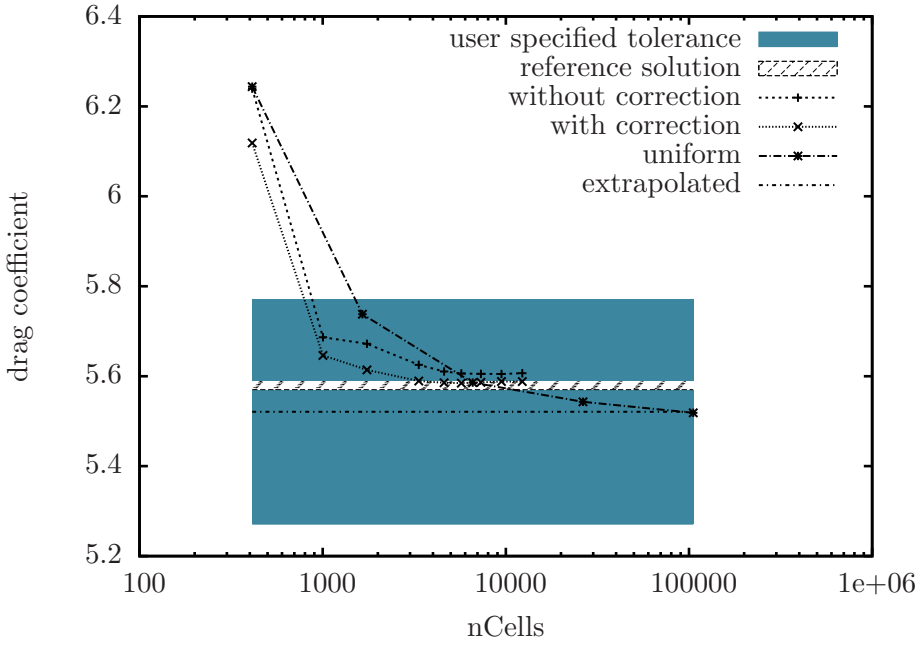


Figure 7.45: Goal-oriented grid-adaptation for drag on initially block-structured grid - Threshold $0.25c_D$.

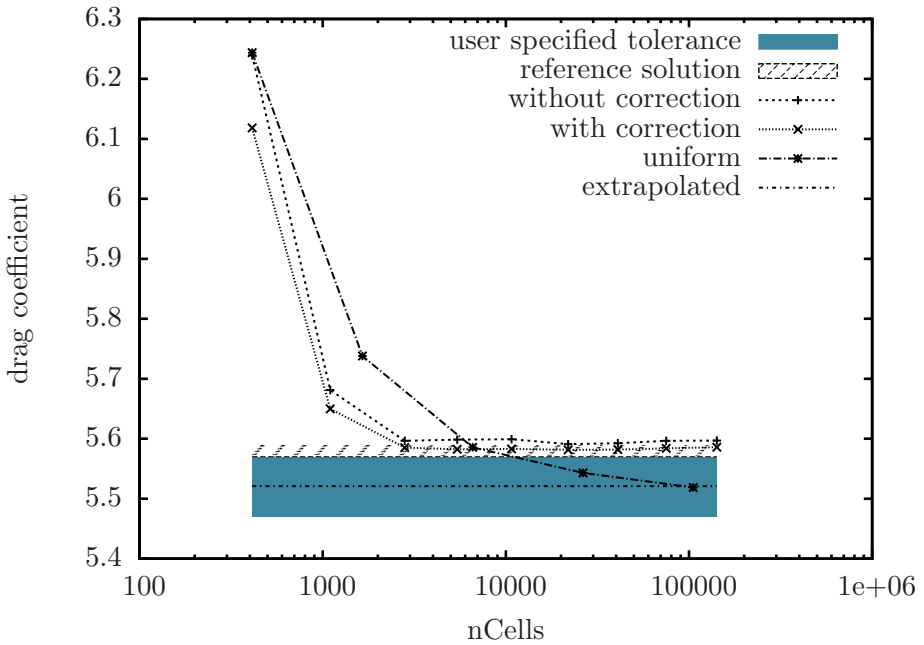


Figure 7.46: Goal-oriented grid-adaptation for drag on initially block-structured grid - Threshold $0.05c_D$.

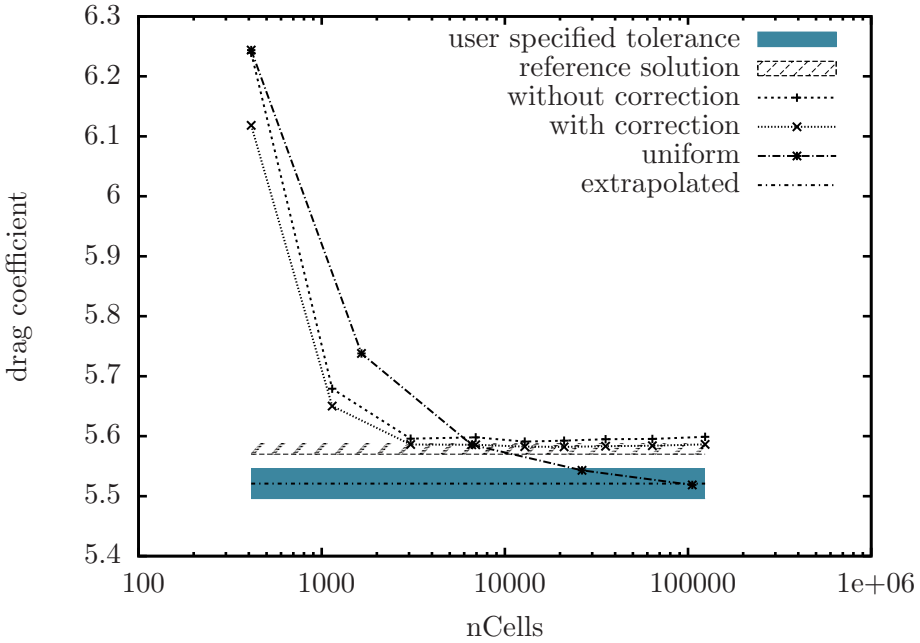


Figure 7.47: Goal-oriented grid-adaptation for drag on initially block-structured grid - Threshold $0.025c_D$.

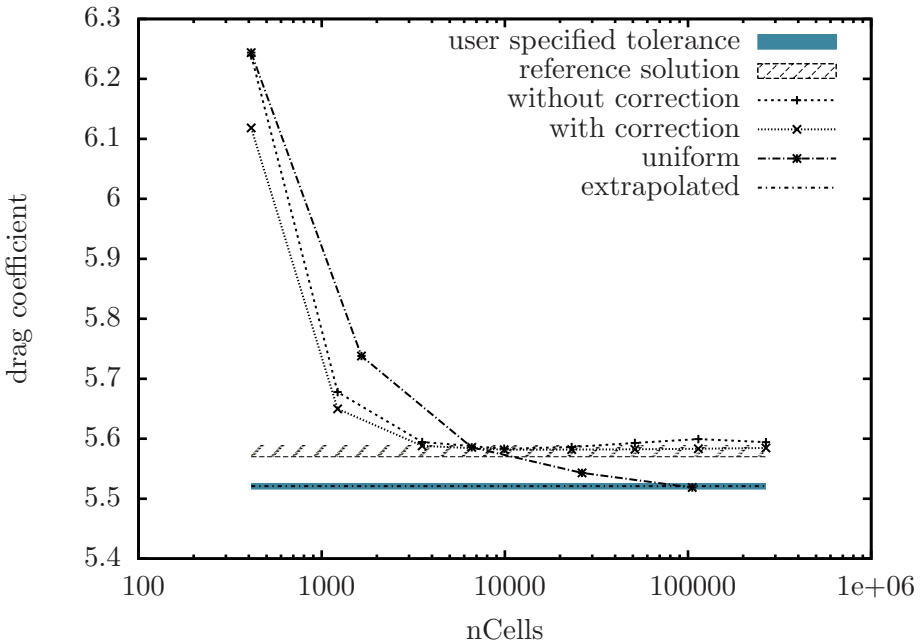


Figure 7.48: Goal-oriented grid-adaptation for drag on initially block-structured grid - Threshold $0.005c_D$.

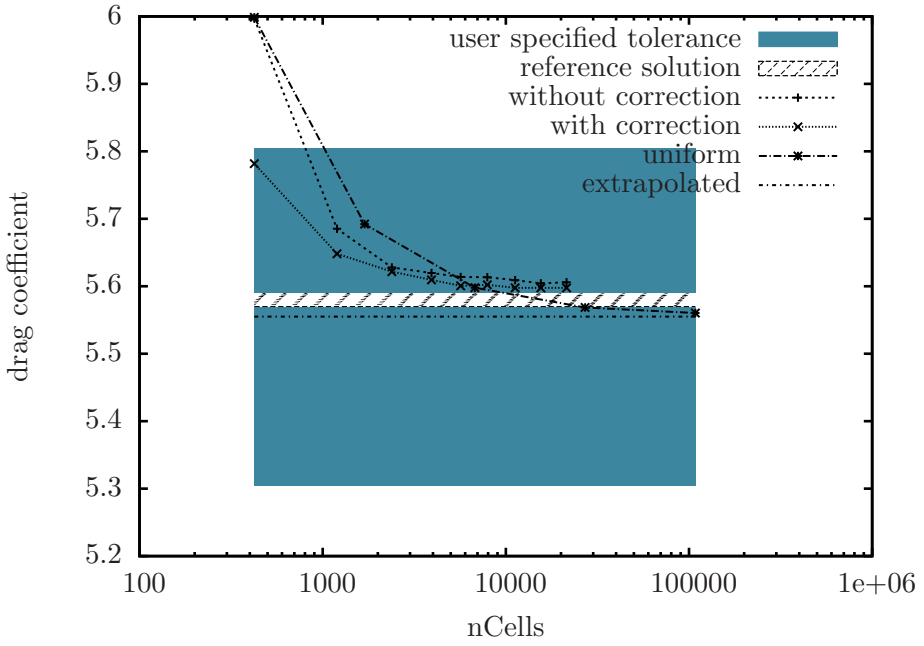


Figure 7.49: Goal-oriented grid-adaptation for drag on initially unstructured grid - Threshold $0.25c_D$.

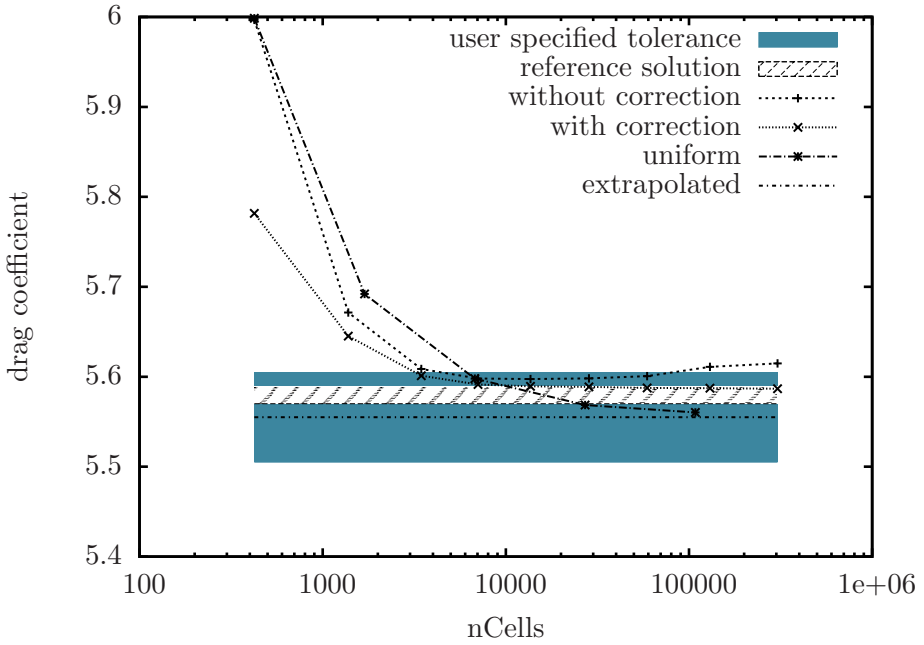


Figure 7.50: Goal-oriented grid-adaptation for drag on initially unstructured grid - Threshold $0.05c_D$.

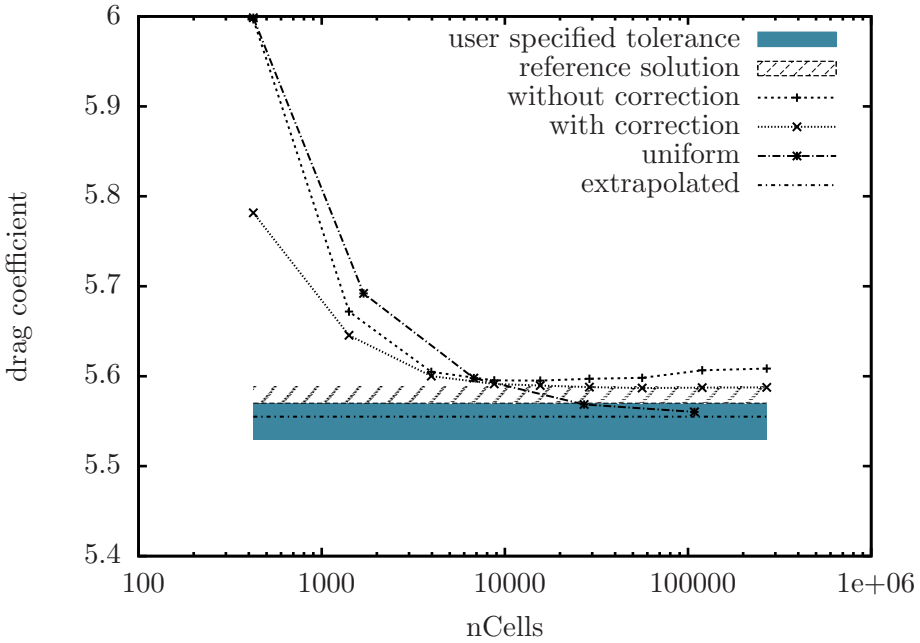


Figure 7.51: Goal-oriented grid-adaptation for drag on initially unstructured grid - Threshold $0.025c_D$.

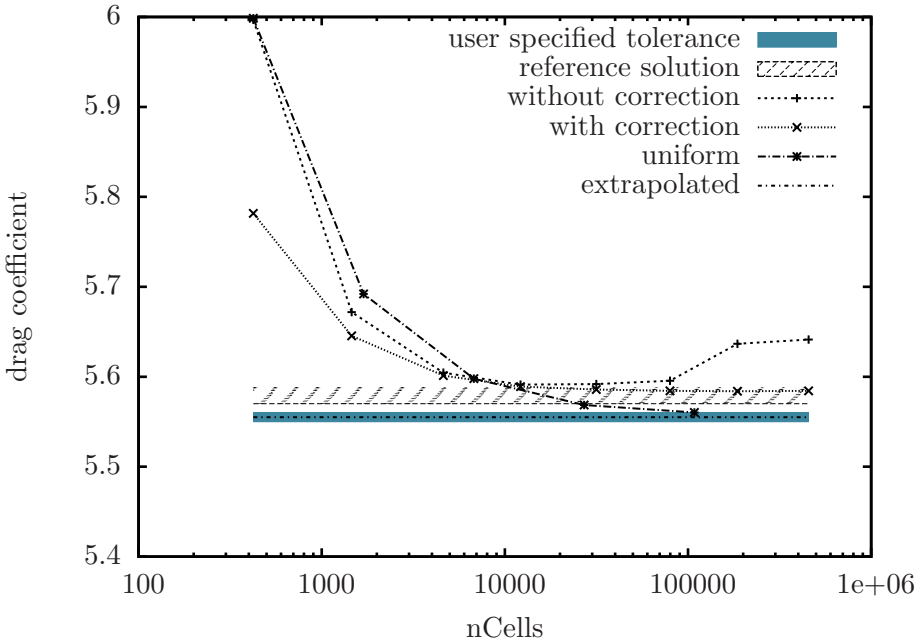
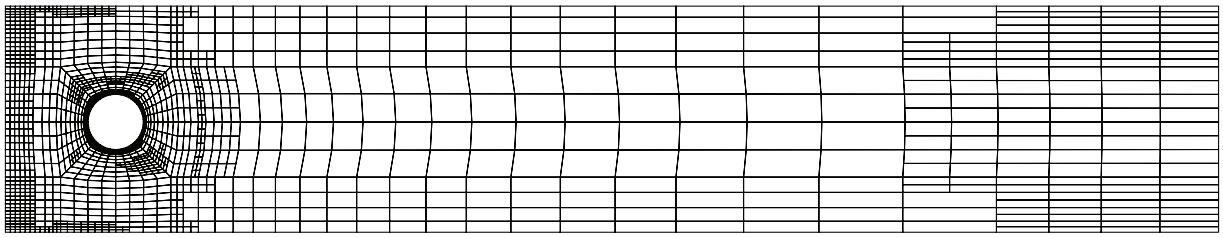
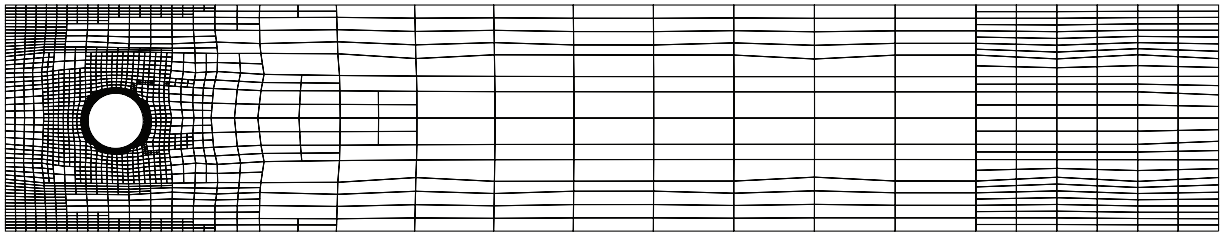


Figure 7.52: Goal-oriented grid-adaptation for drag on initially unstructured grid - Threshold $0.005c_D$.



(a) Block-structured grid (1744 cells).



(b) Unstructured grid (2385 cells).

Figure 7.53: Grids for the DFG Channel Flow Benchmark after 2 refinement steps (Threshold 0.025).

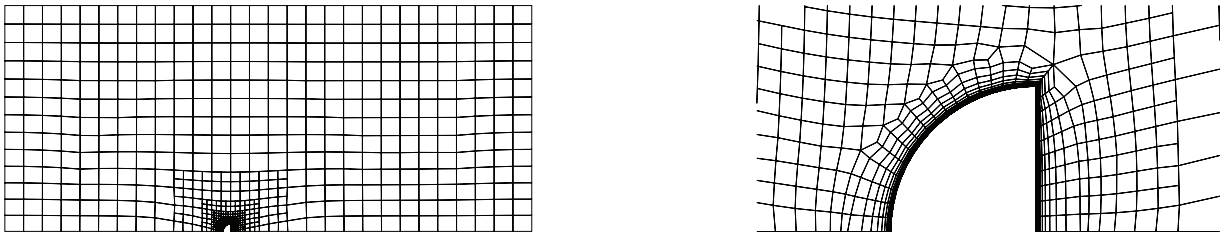
7.4.4 Quarter Cylinder in Free-Stream

The validation of the goal-oriented refinement procedure for turbulent flows, refers to a 2D quarter cylinder in turbulent free-stream conditions. It has been used by Stück [171] to validate the underlying Adjoint Navier Stokes method. It features a blunt body with a distinct separation point.

Case Description

The computational domain, with the initial grid containing $1,267$ cells is depicted in Figure 7.54. The boundary layer at the cylinder is fully resolved with a target $y^+ < 1$. The inlet is located 15 diameters d upstream the cylinder. The extensions to the side and outlet are $15d$ and $20d$, respectively. A symmetry plane has been defined on the centreline. The flow features a Reynolds number of $Re = 1 \times 10^6$ based on the uniform inflow velocity and the cylinder diameter. Turbulence has been modelled by the $k - \omega$ model of Wilcox. Convective fluxes were discretised using the limited QUICK advection scheme.

The goal-oriented refinement strategy with two different functionals has been compared



(a) Overview of initial grid.

(b) Detail of adapted grid.

Figure 7.54: Initial grid for the quarter cylinder in free-stream.

to a residual-error based refinement strategy. The considered functionals were the drag of the cylinder and the axial flow homogeneity at a specified region behind the cylinder. The region considered for flow homogeneity starts $0.9d$ downstream the cylinder and ends at $1.1d$. The lateral extend is defined from the centreline to $0.5d$. The quadratic deviation from the mean axial flow velocity has been chosen as measure for the homogeneity. As a basis for the comparison, a formal grid convergence study with 5 grid levels, yielding $324,352$ control volumes for the finest grid, has been conducted.

Results

Results from the grid convergence study based the least squares version of the grid convergence index proposed by Eça and Hoekstra [50] are presented in Table 7.13. The estimated

Table 7.13: Grid convergence study for the quarter cylinder in turbulent free-stream.

Set	Convergence	Extrapolated c_D	Order	Uncertainty
1-3	monotonic	0.5336	0.989	8.521×10^{-2} (15.966%)
1-4	monotonic	0.5502	1.107	3.599×10^{-2} (6.536%)
1-5	monotonic	0.5552	1.156	1.521×10^{-2} (3.003%)
2-4	monotonic	0.5620	1.426	1.667×10^{-2} (3.298%)
2-5	monotonic	0.5614	1.408	7.464×10^{-3} (1.329%)
3-5	monotonic	0.5608	1.353	7.872×10^{-3} (1.404%)

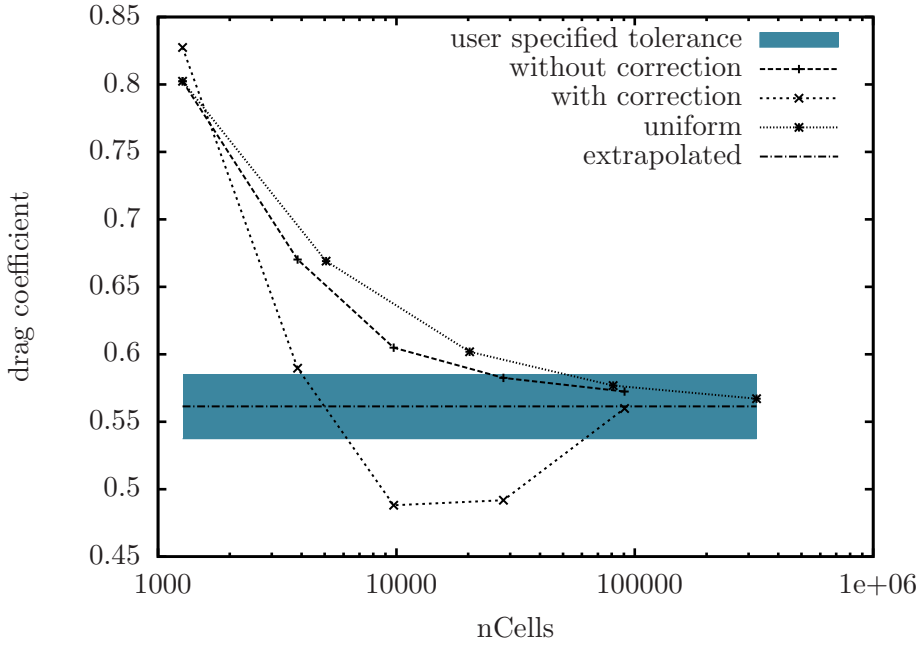
order of accuracy is lower than for the laminar flow examples. For the convective fluxes

of the turbulence equations upwind differences have been used, which influence the order of accuracy. Anyhow an order of accuracy near 1.5 is reached using sufficiently fine grids. The extrapolated drag coefficient is afflicted with uncertainties of 1.5% using sufficiently fine grids.

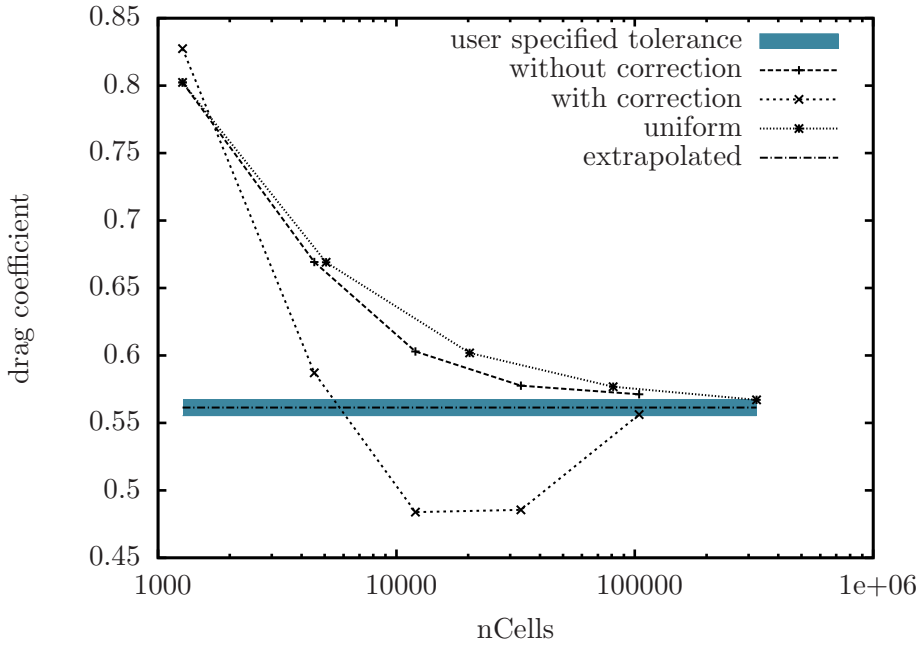
Results from the goal-oriented refinement strategy, using the drag as functional, are compared to the uniform grid refinement in Figure 7.55. Obviously the functional correction procedure does not work reliable for the considered test case. This behaviour of the correction procedure is attributed to the frozen turbulence approach used for the adjoint Navier Stokes method. The negligence of the variation of the turbulent quantities leads to a larger duality gap. However, the derivation of the correction procedure is based on the strict duality of the primal and adjoint solution. Hence, the correction procedure suffers from a larger duality gap. The determination of the correction term is based on the local correction terms in each cell. Therefore, these are afflicted with the same uncertainties. Accordingly, the definition of the refinement threshold is not reliable. However, the goal-oriented refinement procedure leads to grids which are able to predict the drag coefficient with a comparable accuracy as the grids originating from uniform refinement using less cells.

Figures 7.56 to 7.59 depict the quadratic deviation from the axial mean flow velocity in the defined wake region of the cylinder, for different user-prescribed adaptation thresholds. Even for the uniform grid refinement study the functional is not converging asymptotically. Therefore, the result from the finest grid is taken as reference result. Within the results obtained from goal-oriented grid-adaptation the correction term has a small influence on the results. The functional converges towards the target region, defined by the user-specified tolerance, for all prescribed thresholds. However, the convergence is not monotonic for the goal-oriented grid-adaptation and the uniform grid-adaptation. For all prescribed refinement thresholds the predicted WOF after one refinement step is of comparable accuracy like the WOF obtained after one step of uniform refinement. The number of cells from goal-oriented refinement ranges from 2,857 to 3,136 after one refinement step, while uniform refinement leads to 5,068 cells after one step. For the most strict threshold a comparable accuracy of the WOF compared to uniform refinement is achieved after two refinement steps using 4,496 cells compared to 20,272 cells using uniform refinement.

Figure 7.60 depicts the grids obtained from goal-oriented refinement for the different functionals. For both functionals, refinement is mainly located close to the symmetry plane



(a) Threshold $6.0N(2.4 \times 10^{-2} c_D)$.



(b) Threshold $1.5N(6 \times 10^{-3} c_D)$.

Figure 7.55: Estimated drag from goal-oriented refinement using drag as functional.

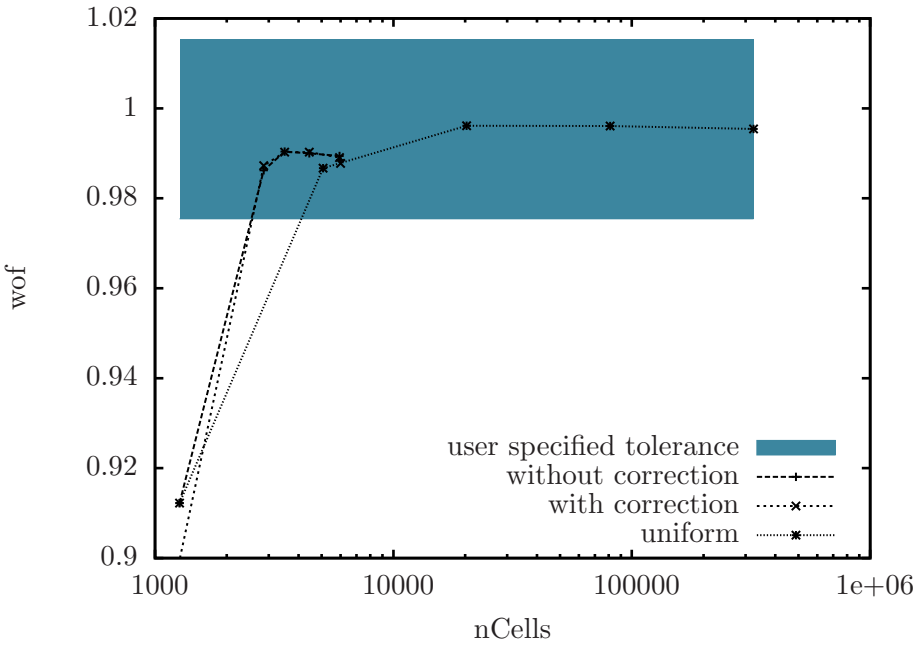


Figure 7.56: Quadratic deviation from the mean flow velocity in the wake from goal-oriented refinement using the homogeneity objective - Threshold 0.02.

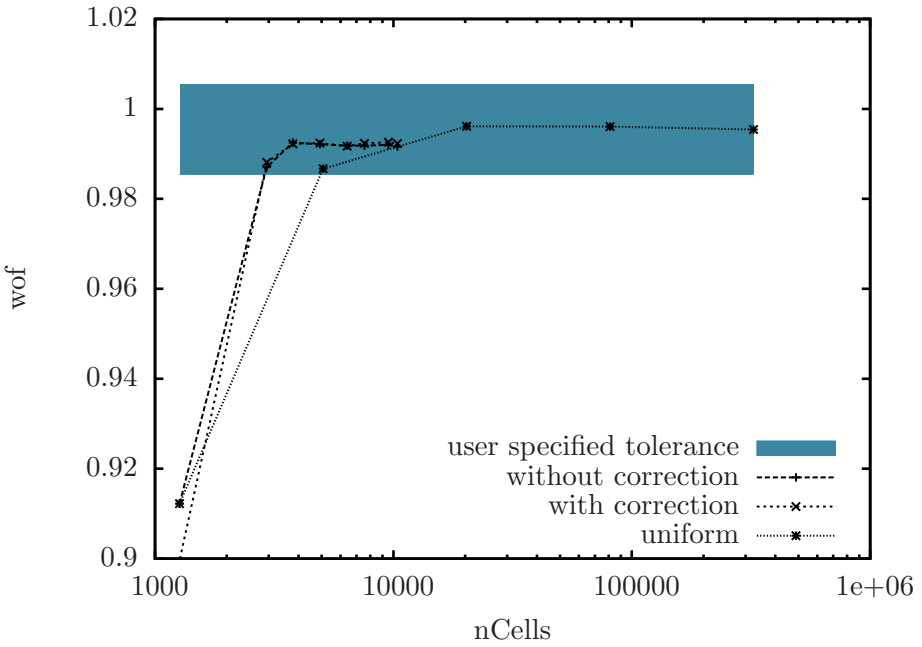


Figure 7.57: Quadratic deviation from the mean flow velocity in the wake from goal-oriented refinement using the homogeneity objective - Threshold 0.01.

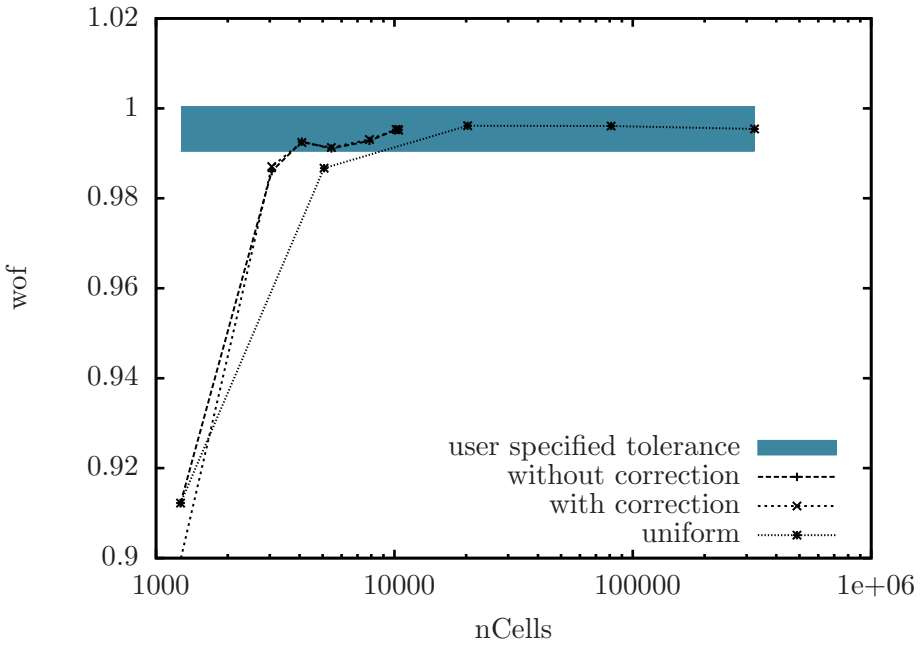


Figure 7.58: Quadratic deviation from the mean flow velocity in the wake from goal-oriented refinement using the homogeneity objective - Threshold 0.005.

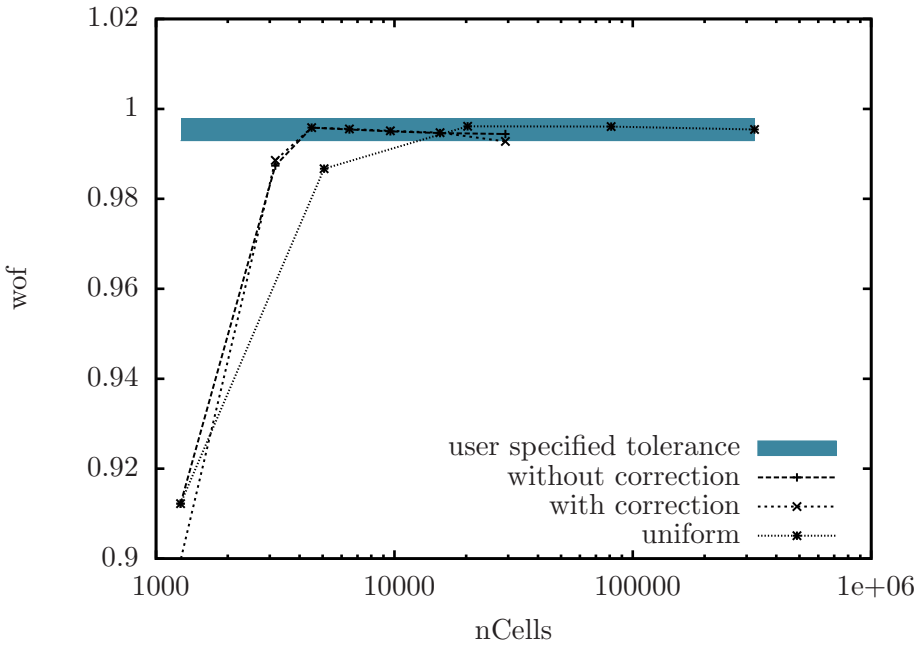
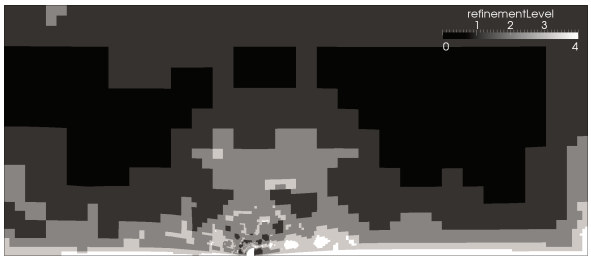
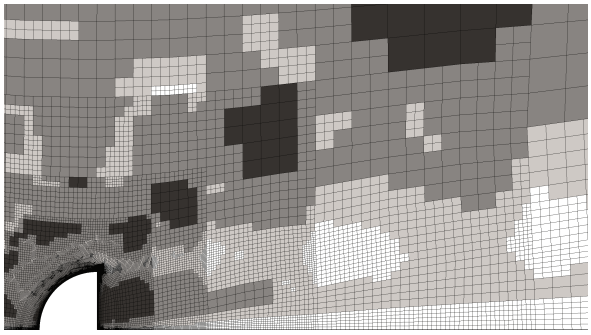


Figure 7.59: Quadratic deviation from the mean flow velocity in the wake from goal-oriented refinement using the homogeneity objective - Threshold 0.0025.

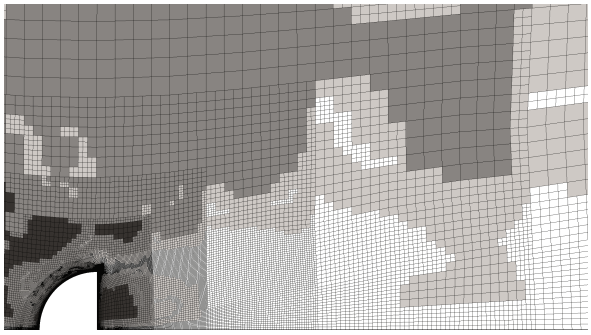


(a) Resistance Objective - Grid Overview.

(b) Wake Objective - Grid Overview.



(c) Resistance Objective - Grid Detail.

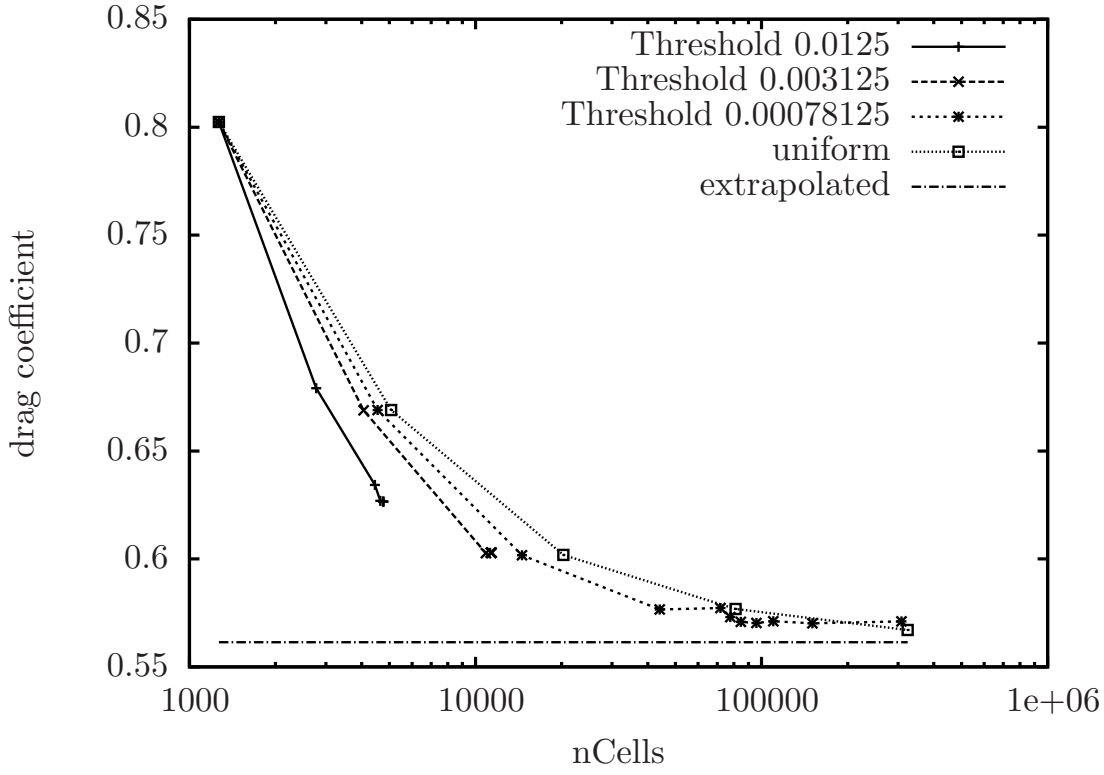


(d) Wake Objective - Grid Detail.

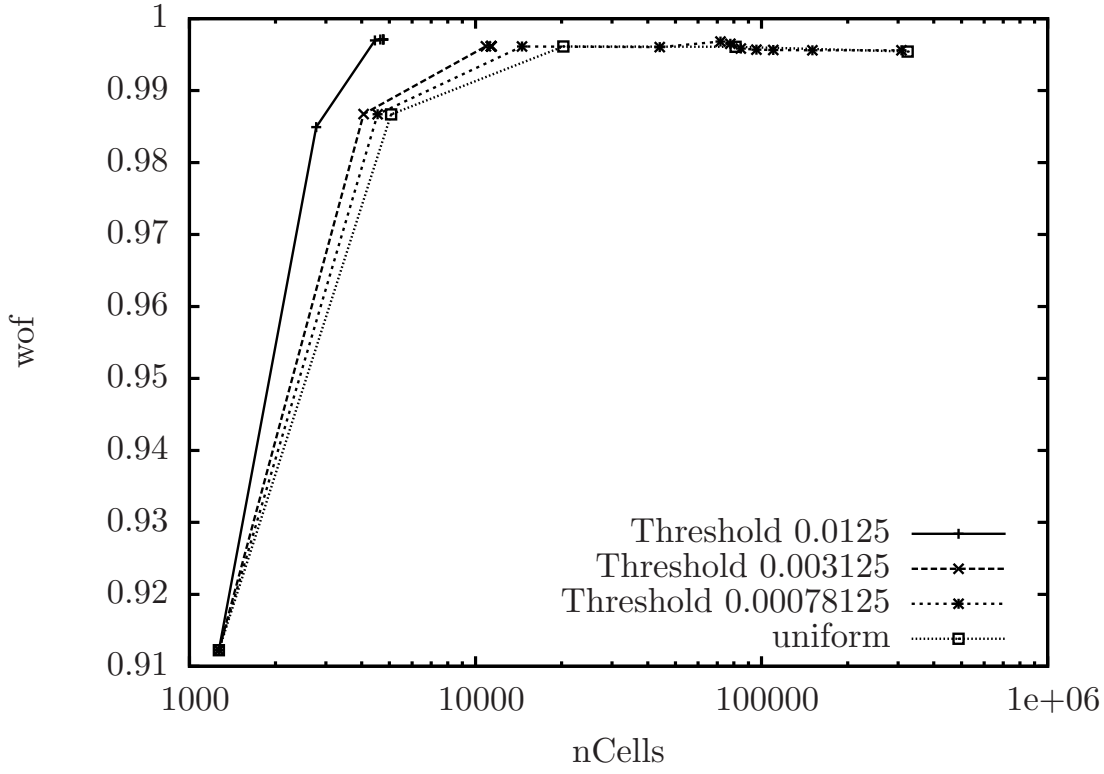
Figure 7.60: Grids obtained from goal-oriented refinement using different functionals for flow past quarter cylinder.

and in a relatively large area around the obstacle. Furthermore, additional refinement at the outflow boundary close to the symmetry plane occurs. Comparing the grid in the vicinity of the quarter cylinder, it is obvious that goal-oriented refinement based on the wake objective leads to a finer mesh in the wake region of the cylinder.

Results from simulations with automatic grid-adaptation based on the residual error estimator are presented in Figure 7.61. Both functionals converge towards the reference solutions, while the accuracy of the final result depends on the user-prescribed threshold, as expected. Interestingly, refinement based on the residual error estimator is afflicted with some stability issues, which have not been noticed for goal-oriented refinement. For all prescribed refinement thresholds the last stable solution is depicted. An additional refinement step led to diverging simulations. A comparison of goal-oriented and residual error estimator based refinement depicted in Figure 7.62, reveals that goal-oriented refinement leads to results of same accuracy as residual error estimator based refinement using



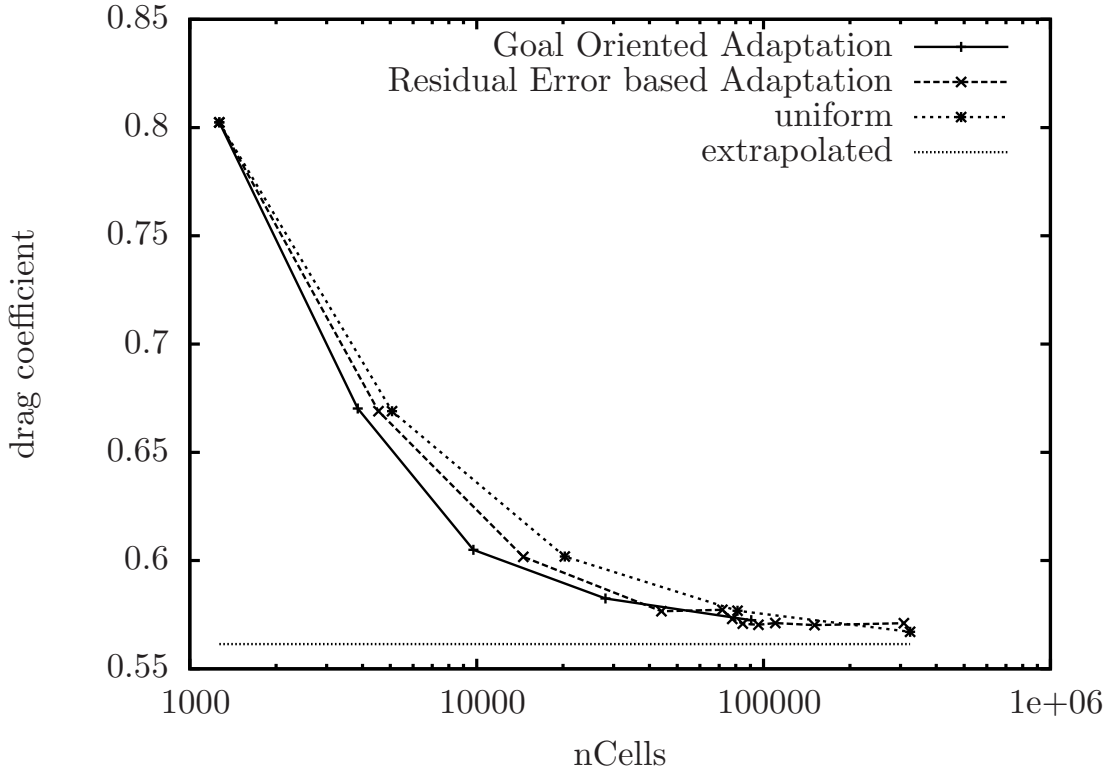
(a) Drag coefficient.



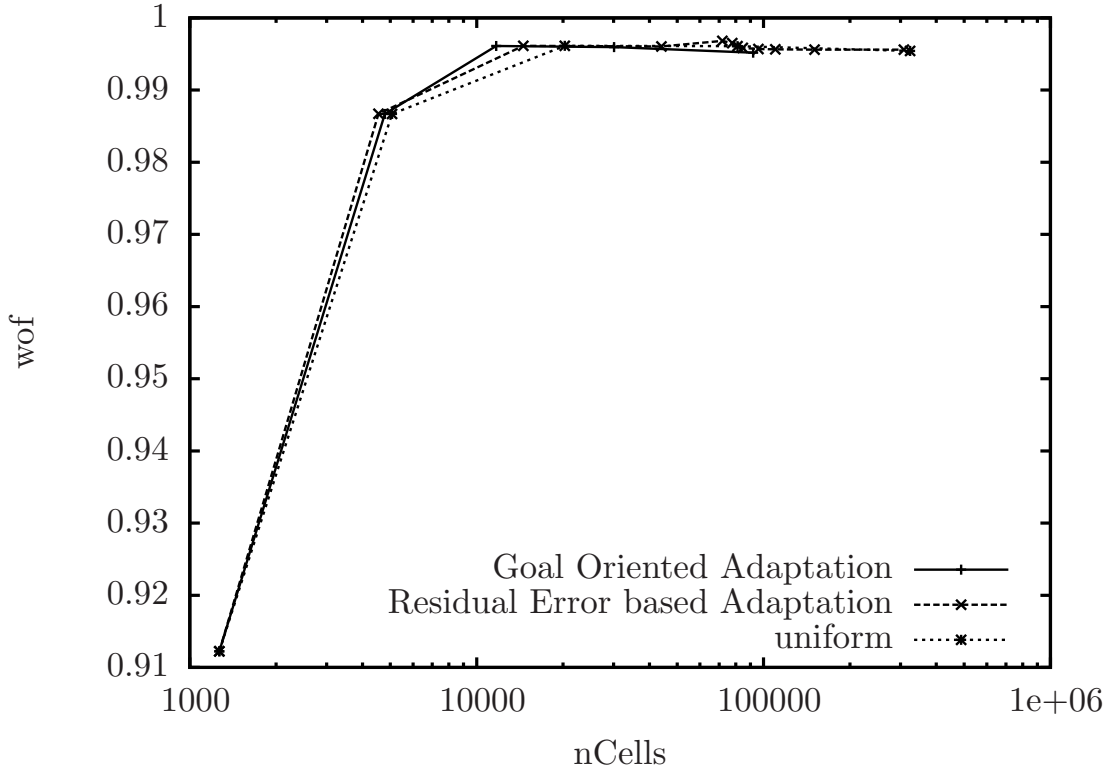
(b) Wake objective function.

Figure 7.61: Results from automatic grid-adaptation using the residual error estimator.

less cells for both considered functionals.



(a) Drag coefficient.



(b) Wake objective function.

Figure 7.62: Comparison of results from goal-oriented and residual error estimator based adaptation.

8 Case Studies

Within this chapter the application of the developed techniques to common maritime flows is presented. The first applications of a steady wave resistance analysis of a tanker in Section 8.1 and a motion prediction for a tanker in Section 8.2 are denoted to the application of the sub-cycling procedure presented in Section 5.2 and the explicit interface sharpening technique EIS described in Section 5.3. Within the example of the steady wave resistance prediction the sub-cycling technique is employed in the unsynchronised mode while it is employed in the synchronised mode for the motion prediction, where time accurate results are required. The last example of a wake prediction in Section 8.3 demonstrates the use of the dynamic grid-adaptation technique using the goal-oriented grid-adaptation indicators according to Section 6.3.3 with different functionals.

8.1 Steady Wave Resistance of a Tanker

Case Description

The application example of the unsynchronised sub-cycling procedure focuses upon the KVLCC2-tanker in model scale at Fn=0.142 and Re=4.6 $\cdot 10^6$, where a vast amount of numerical and experimental data is available [103, 182]. The case aims to predict the steady state wave pattern when exposed to calm water conditions with frozen trim and sinkage. Reported results are obtained on an unstructured mesh using local grid refinement in the vicinity of the hull. Figure 8.1 outlines the employed computational grid, which consist of 3.5 Mio. control-volumes. Five sub-cycles have been used to advance the concentration equation in time. An adaptive time step has been employed for the concentration equation which ensures that the RMS-value of the Courant number yields $Co_{RMS} = 0.3$.

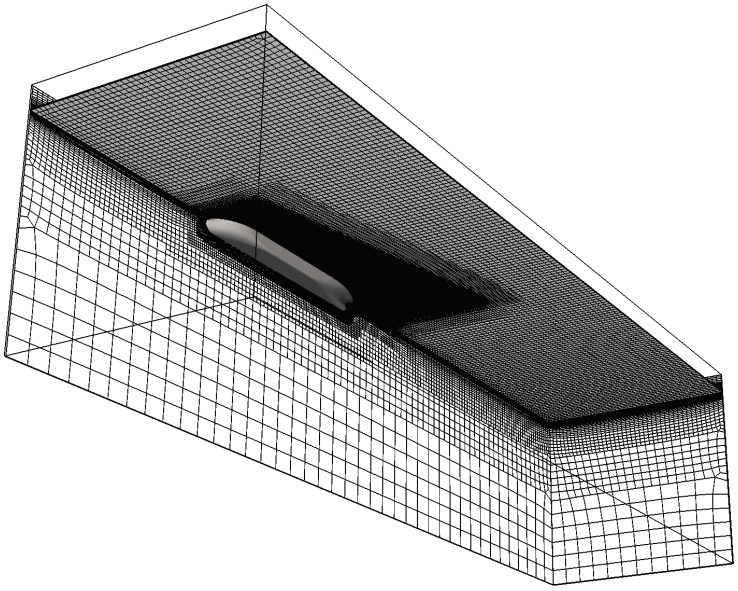
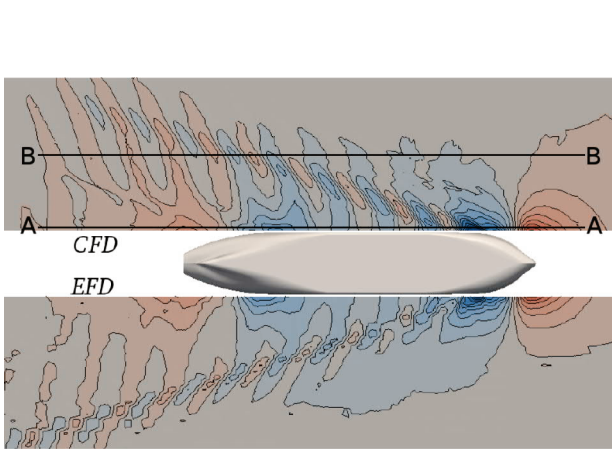


Figure 8.1: Numerical grid employed for the free-surface flow computation around the KVLCC tanker exposed to calm-water conditions

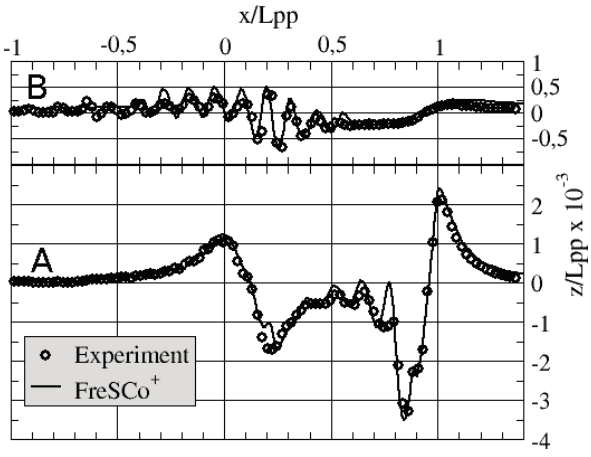
Results

The predicted wave pattern — indicated by the iso-surface $c = 0.5$ — is illustrated in Figure 8.2(a) in comparison to measured EFD data reported by Kim, Van and Kim [99]. The pattern is in remarkable agreement with experiments as depicted by the wave cuts in Figure 8.2(b). In conjunction with the unsynchronised sub-cycling technique, force convergence is typically obtained after 2,500 iterations. Moreover the approach provides a very rapid initial convergence as outlined by the comparison of wave contours displayed in Figure 8.3. The essentials of the flow field are already captured after 650 time steps, when the standard technique is still afflicted by large amounts of artificial initial transient.

Figure 8.4 provides an overview of the evolution of the predicted drag-force deviation from experiments for the standard approach and the unsynchronised sub-cycling technique. It is seen, that the sub-cycling approach exhibits an improved convergence behaviour. Force convergence is achieved after 3% of the wall-clock time of the standard approach which yields an impressive speed-up factor of 33. Mind that the ideal speed-up for a synchronised transient approach is $\mathcal{S} = 3.7$.



(a) Predicted and measured wave pattern & wave cuts positions



(b) Comparison of predicted and measured wave height along wave cut A (y=0.0964/Lpp) and B (y=0.2993/Lpp)

Figure 8.2: Calm-water predictions of the KVLCC2 tanker

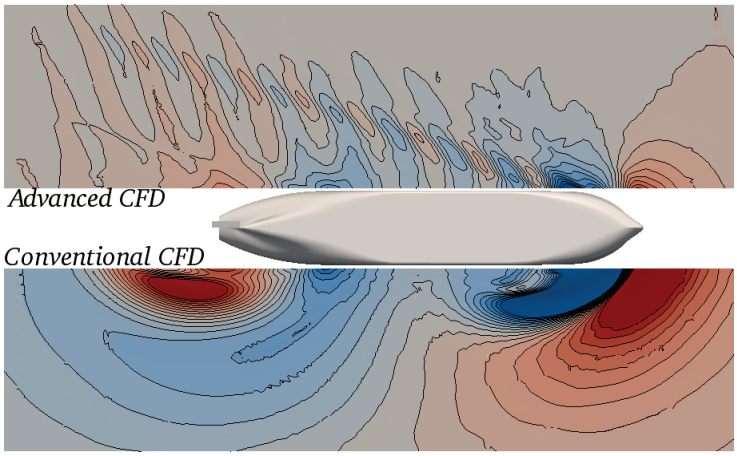


Figure 8.3: Predicted and measured wave pattern for the KVLCC2 tanker after 650 time steps

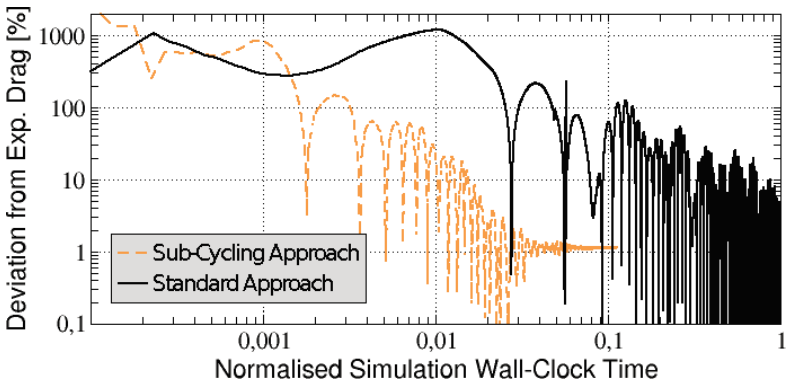


Figure 8.4: Deviation of predicted drag for the KVLCC2 tanker from experimental results for the standard and the sub-cycling approach

8.2 Pitch and Heave Prediction for a Container Vessel

Case Description

The synchronised sub-cycling technique has been applied to transient investigations of a free floating Kriso Container Ship (KCS) in head waves at Re=6.52 $\cdot 10^6$ and Fn=0.26. The investigated model-scale hull features a length of $Lpp = 4.3671\,\mathrm{m}$. The incoming waves had a significant height of $H_s/Lpp = 0.03343$ and a wave-length of $\lambda = 2Lpp$. Model tests have been carried out at FORCE Technology[167]. To account for the body motion a six degree of freedom quaternion-motion solver is coupled to the flow analysis. All degrees of freedom except pitch and heave have been suppressed in the present study and the grid is rigidly moved according to the predicted motion of the ship. The employed numerical grid takes advantage of the symmetry plane at $y = 0\,\mathrm{m}$ and consists of 0.83 Mio control-volumes (cf. Figure 8.5). Cells are refined in the free-surface region. The global

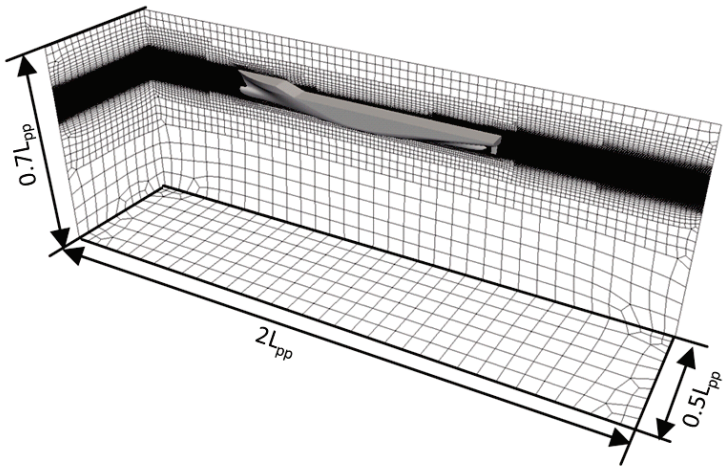


Figure 8.5: Numerical grid employed for the free-floating KCS container vessel (0.83 Mio cells)

time step has been assigned to a constant value of $\Delta t = 1 \cdot 10^{-3}\,\mathrm{s}$, which corresponds to a maximum Courant number of $Co = 5.4$ for a few cells and to $Co < 0.3$ for the majority of the control volumes for the simulation set-up without sub-cycles. For the set-ups using sub-cycles the global time step is multiplied with the number of sub-cycles. Simulations have been carried out without sub-cycling and with 4 and 8 sub-cycles.

Results

Time histories of the heave and pitch motions reported by the experiment and the present simulations are compared in Figure 8.6. The predicted heave motions are generally in fair agreement with measured data. The amplitude of the pitch motion is slightly under-predicted. To analyse the predictive differences returned by the sub-cycling approach,

(a) Heave

(b) Pitch

Figure 8.6: Simulation of a free-floating KCS container vessel: Comparison of measured and computed heave and pitch motion time records using $N_{\Delta t} = 1, 4, 8$

the amplitudes of the 0^{th}- and 1^{st}-order harmonics as well as the phase of the 1^{st}-order harmonic are assessed. The resulting coefficients are summarised in Table 8.1. The data indicates that 0^{th}- and 1^{st}-order harmonic amplitudes are not very sensitive to the increase of the global time step due to the use of sub-cycles. On the contrary, the 1^{st}-order harmonic phase displays a reduction of more than 10% comparing simulations with 8 and 1 sub-cycles. Figure 8.7 compares the achieved speed-up with the ideal speed-up. Computations without sub-cycling required approximately 10% of the total CPU effort for the free-surface prediction, which results in significant achievable speed-up factors for the use of sub-cycles.

Table 8.1: Comparison of measured and predicted 0^{th}- and 1^{st}-order harmonic amplitudes and 1^{st}-order harmonic phase for pitch and heave motions of the KCS container ship

		Experiment	$N_{\Delta t} = 1$	$N_{\Delta t} = 4$	$N_{\Delta t} = 8$
		Heave			
0^{th} Amplitude	(mm)	-12.021	-15.943	-15.213	-15.337
1^{st} Amplitude	(mm)	64.798	63.404	63.507	63.917
1^{st} Phase	(°)	15.418	14.852	13.817	10.510
		Pitch			
0^{th} Amplitude	(°)	0.032	0.283	0.282	0.392
1^{st} Amplitude	(°)	3.071	2.803	2.803	2.839
1^{st} Phase	(°)	70.418	77.812	76.276	72.064

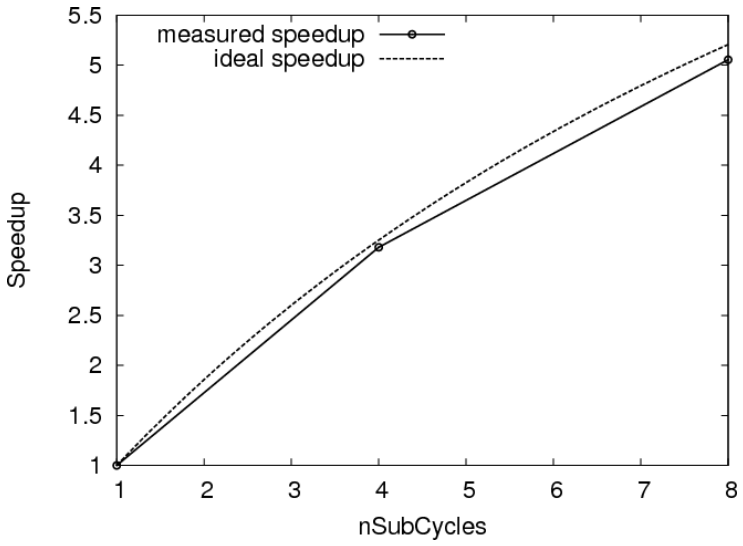


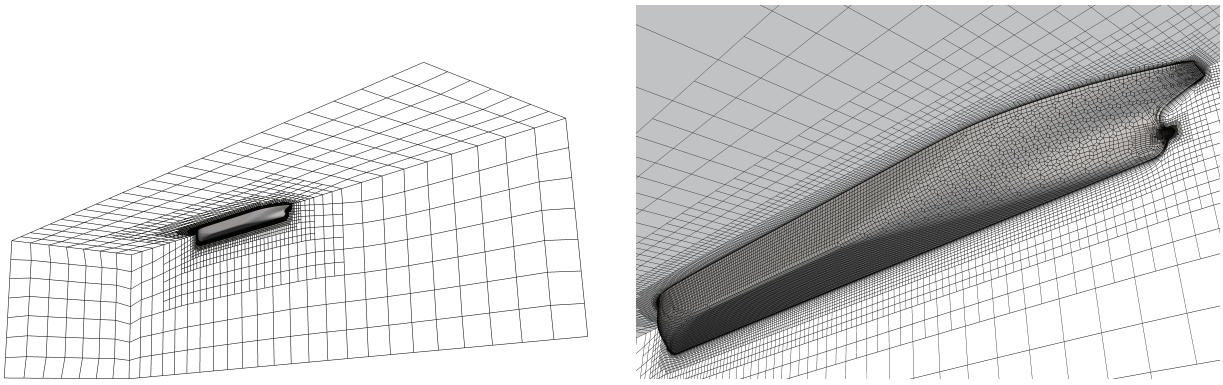
Figure 8.7: Simulation of a free-floating KCS container vessel: Comparison of ideal and realised speed-up due to the use of the sub-cycling techniques

8.3 Wake Field Prediction

The availability of detailed wake-field informations during the initial design phase of a ship provides a huge opportunity to optimise the design with respect to propeller induced pressure pulses. To support the initial design process, the respective wake-field predictions are requested to be fast and robust. Inviscid potential-flow methods can not be used for the optimisation of the wake, which is dominated by viscous effects. The topic can be addressed by viscous methods, e.g. RANS or LES, which are traditionally deemed to be afflicted by a prohibitive effort. The accuracy of the wake-field predictions is mainly governed by a fair development of the boundary layer, including separation and transport of vortices created in the aft-body region. Both, the accurate prediction of separation and the transport of vortices hinges on a numerical grid providing sufficient grid resolution. As the locations where separation occurs and vortices are generated and transported are usually not known beforehand, standard numerical grids for wake field predictions are refined in the whole aft-body region. The developed goal-oriented grid-adaptation procedure is applied to the wake-field prediction and results are compared to results from uniformly refined grids and refinement based on local error indicators.

Case description

Results refer to double-body mode computations due to the low Froude number. Interest is confined to model-scale ($\lambda = 58$) experiments, which feature a model length of $L_{ref} = 5.517m$, an approach velocity $V = 1.047\frac{m}{s}$ and a Reynolds number of $Re = 5.088 \times 10^6$. The $k-\omega$ Shear-Stress Transport turbulence model of Menter [119] has been used to model turbulence. Wall functions have been applied to reduce the computational effort. Convective fluxes for the momentum equation have been approximated using a monotonicity preserving QUICK scheme. The very coarse initial grid, which is the starting point for the grid refinement procedure, consists of roughly 120k hexahedrons and takes advantage of the symmetry plane at $y = 0$. The mesh, depicted in Figure 8.8, is locally refined at the hub in order to provide a grid resolution capable to represent the hub geometry at least roughly. The testcase offers a broad body of CFD-experience and detailed information about the flow in the propeller plane and several upstream planes [181, 182]. The locations for the different measurement planes relative to the aft perpendicular are given in Table 8.2. The experimental results, depicted in Figure 8.10, reveal that a vortex is generated in



(a) Overview (b) At hull

Figure 8.8: Initial numerical grid for the wake field prediciton for the KVLCC2.

Table 8.2: Measurement stations for KVLCC2 experiment.

Station	St3	St2	St1	Stpp
$\frac{x}{L_{pp}}$	0.15	0.1	0.05	0.0175

the region of convex hull-curvature in the aft-body (cf. St1 in Figure 8.9). The fingerprint

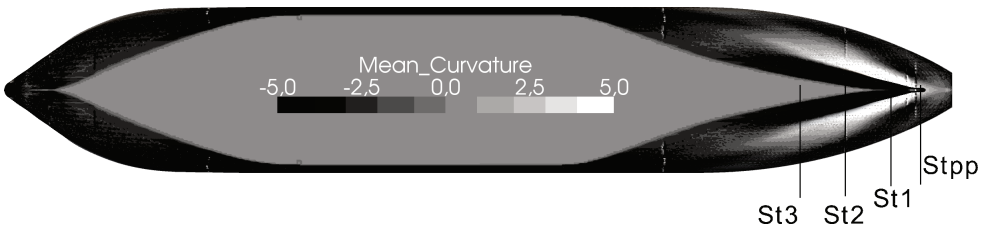


Figure 8.9: Locations of measurement planes for KVLCC2 experiment.

of this vortex, which is often suppressed in RANS simulations, can be clearly seen in the propeller plane. The impact of goal-oriented grid refinement on the prediction of wake fields will be studied using two different objective functions. The first objective function refers to the equi-distribution of the axial velocity in the wake field and was derived by

Stück [172]. The associated wake objective function reads

$$J = 1 - C \int_{v(prop)} \frac{dV}{2r} \left[U^a - \overline{U}^a(r)\right]^2 \quad \text{with} \quad C = \left[U_S^2 \int_{v(prop)} \frac{dV}{r}\right]^{-1} \tag{8.1}$$

which integrates the quadratic deviation of the axial velocity U^a from its radial mean velocity $\overline{U}^a$ in the propeller disk volume. The propeller plane is located at station Stpp, its diameter is $0.17m$ and its vertical distance from the keel of the tanker is $0.1m$. The adjoint source term derived from the objective function reads

$$\delta J = \int_{v(prop)} \delta u_i \frac{\partial j}{\partial u_i} dV \quad \text{with} \quad \frac{\partial j}{\partial u_i} = -\tilde{e}_i^a C \frac{u^a - \overline{u}^a(r)}{r}. \tag{8.2}$$

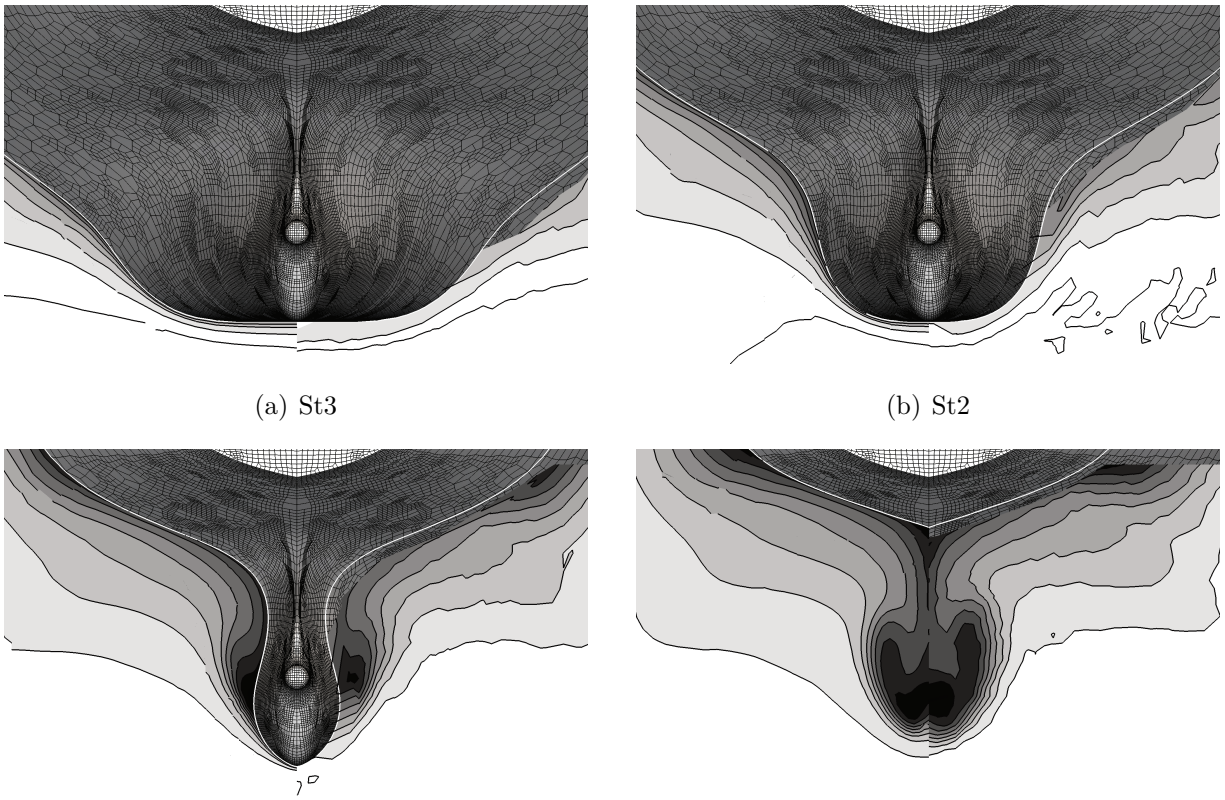
To demonstrate the influence of the objective function on the grid refinement a second objective function which regards the minimisation of the drag has been employed, for which the objective function reads

$$J = \int_{F_o} j_f dF \quad \text{with} \quad j_F = \left(p\delta_{ij} + \tau_{ij}\right) n_j d_i^*. \tag{8.3}$$

with F_o being the part of the surface on which the objective function is evaluated,n_j the boundary normal unit vector and d_i^* the direction vector indicating the force optimisation direction.

Results

Figure 8.10 shows the predicted axial flow velocity obtained from simulations with goal-oriented grid refinement based on the wake objective function compared to the experimental results at different measurement stations. The refinement threshold has been set to $t = 0.05$. The generation of the vortex in the vicinity of measurement station St1 is clearly captured, however the vortex has not yet been fully separated from the hull as indicated in the experimental results. Furthermore, the vertical position of the vortex is slightly lower than in the experiments. The vortex is transported downstream to the propeller-plane, where its fingerprint is clearly visible leading to a horseshoe shape of the wake. Its shape is less pronounced compared to the experiments, however this is an excellent result for a numerical grid consisting of 400k cells. The predicted boundary layer thickness is in excellent agreement with the experimental results for the first two measurement planes and slightly thicker than in the experiment for the last two measurement planes. However, the applied refinement strategy focuses on the homogenisation of the propeller inflow.



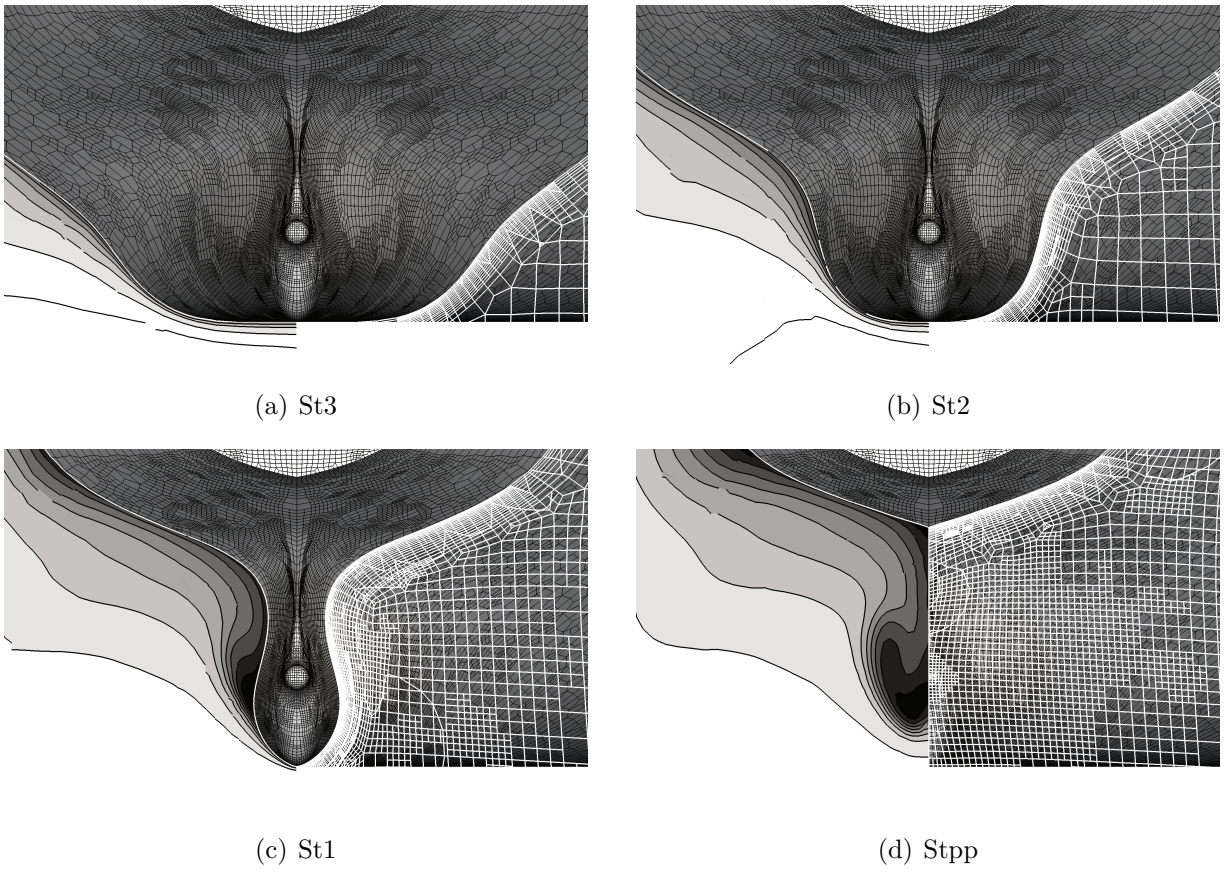
(a) St3 (b) St2

(c) St1 (d) Stpp

Figure 8.10: Comparison of normalised axial velocity obtained from simulations with goal-oriented refinement for wake equi-distribution (left side) and experiment (right side) for KVLCC2 at measurement stations.

The adapted numerical grid at the hull surface indicates that the grid has mainly been refined in the vicinity of the origin of the vortex. The numerical grid in the volume at the measurement stations is depicted in Figure 8.11. For the first two measurement stations the grid in the volume has not been adapted compared to the initial grid leading to coarse cells. At the second measurement station refinement at the wall focuses to regions afflicted with high curvature. At station St1 intensive refinement in the boundary layer and at the location of the vortex is visible. The grid at the propeller plane is refined over a larger area including the area of the propeller.

The development of the wake objective function over successive refinements, for goal-oriented refinement with the wake objective and with a resistance objective are compared



(a) St3 (b) St2

(c) St1 (d) Stpp

Figure 8.11: Normalised axial velocity obtained from simulations with goal-oriented refinement for wake equi-distribution (left side) and the underlying numerical grid (right side) for KVLCC2 at measurement stations.

to the development from uniform refinement in Figure 8.12. The wake objective function obtained from goal-oriented refinement with the wake objective shows a monotonic convergence behaviour. However, a kink after the first refinement indicates that the chosen refinement threshold is relatively large. The shape obtained from goal-oriented refinement focussing on the resistance is more irregular, while the trend is the same. The objective function obtained from uniform refinement is converged to a lower value, however the finest grid has about 20 times more cells than the finest grid from goal-oriented refinement.

The influence in the resulting grid, due to different objective functions, is exemplary shown for the grid on the hull in Figure 8.13. The grid obtained from simulations with the wake objective has been intensively refined in the propeller region and at the aft-body

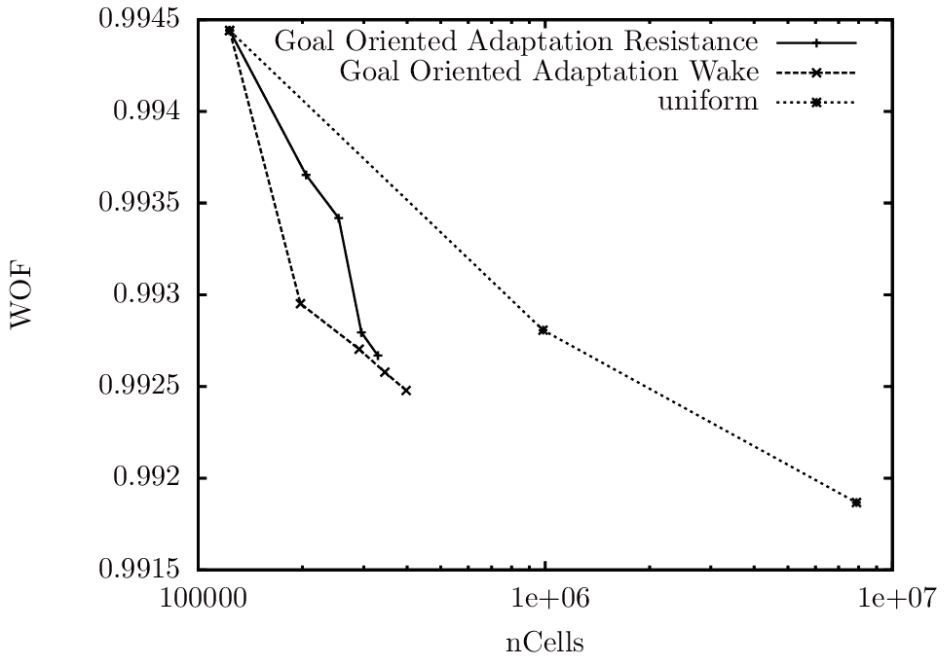
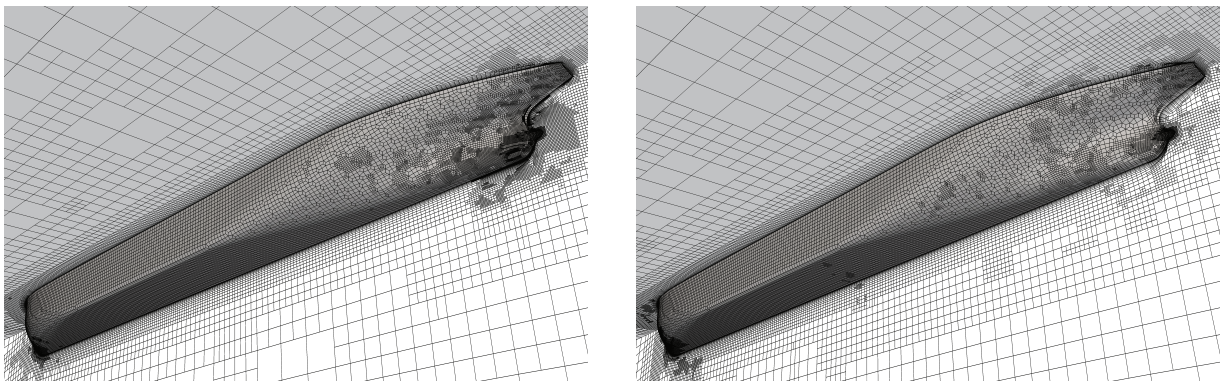


Figure 8.12: Evolution of wake objective function for different grid-adaptation strategies.



(a) Objective: wake

(b) Objective: resistance

Figure 8.13: Grids on hull surface obtained from goal-oriented refinement with different objective functions.

region close to the propeller. Some refinement occurred at the bow of the vessel. In contrast, the bow has been more intensively refined for the resistance objective while only a few refined cells occur in the aft-ship region. Interestingly, the bilge has not been refined for both cases.

The predicted wake in the propeller disk after each refinement step is depicted in Figure 8.14 for both goal-oriented adaptation strategies and uniform refinement. The initially predicted wake without refinement is of course the same. The hook shape, which is the main feature of the wake, is only adumbrated due to the coarse grid. After one level of refinement the hook shape gets clearly more pronounced for the wake objective based refinement. The predicted shape is visually the same as for uniform refinement using 197k cells instead of 1m cells for uniform refinement. The hook gets slightly more pronounced for the resistance objective where the grid consist of 205k cells. After the second refinement step the predicted wake from the wake objective based adaptation converges towards the experimental result on a grid consisting of 255k cells. Again, only slight changes in the predicted wake from the simulation focussing on resistance are visible. Interestingly, the wake from uniform refinement diverges from the experimental result. This may be attributed to the uniform refinement procedure, which also refines the cells in the boundary layer uniformly affecting the initially more homogeneous distribution of the non-dimensional wall distance y^+. Therefore, the distance of the first cell centre to the wall is halved with each refinement step. The initial grid was designed with a target $y^+ = 100$ leading to a target $y^+ = 25$ for the level 2 uniformly refined grid. This pushes the applied wall function to its limits and may probably explain the divergence from the experiment. The wake obtained from simulations with the wake objective is visually converged after the second refinement step, while some slight changes are visible in the results from the resistance based refinement strategy.

The remaining differences between the numerically predicted wake field using the wake field homogeneity objective function and the experimental results can be explained by two different root causes. One reason for the remaining differences is the applied turbulence model, which even on excessively refined grids, is not able to capture all details of the vortical structures. Especially in the region of the vortex core it tends to overestimate the turbulent viscosity and hence the diffusion of the vortex. Explicit algebraic stress models offer a possibility for a better modelling of the turbulent flow for this region. The second reason is that the applied objective function is designed to minimise the quadratic

deviation of the radial mean velocity. Hence, the approach will try to modify the grid in a way that this objective, a radially homogeneous flow, is reached. Strictly speaking the deviation from the experimental result should have been applied as objective function to produce the optimal grid for the prediction of the experimentally measured wake field. This could have been done by mapping the experimental results to the relevant discrete grid points, and use these as basis for the evaluation of the objective function considering the deviation of the numerically predicted axial velocity to the experimentally measured axial velocity. By the use of this strategy a better and more controlled convergence behaviour of wake objective function towards the experimental result over the refinement steps can be expected. Nevertheless, usually the experimental results are not available during the design phase and the numerical simulation shall be used during the design phase to predict and improve the wake field. Hence, the applied strategy is driven by the practical optimisation approach and has been shown to be able to locally refine the grid in relevant regions for a better prediction of the wake field. The remaining differences between the numerically predicted wake field using the wake field homogeneity objective function and the experimental results can be explained by two different root causes. One reason for the remaining differences is the applied turbulence model, which even on excessively refined grids, is not able to capture all details of the vortical structures. Especially in the region of the vortex core it tends to overestimate the turbulent viscosity and hence the diffusion of the vortex. Explicit algebraic stress models offer a possibility for a better modelling of the turbulent flow for this region. The second reason is that the applied objective function is designed to minimise the quadratic deviation of the radial mean velocity. Hence, the approach will try to modify the grid in a way that this objective, a radially homogeneous flow, is reached. Strictly speaking the deviation from the experimental result should have been applied as objective function to produce the optimal grid for the prediction of the experimentally measured wake field. This could have been done by mapping the experimental results to the relevant discrete grid points, and use these as basis for the evaluation of the objective function considering the deviation of the numerically predicted axial velocity to the experimentally measured axial velocity. By the use of this strategy a better and more controlled convergence behaviour of wake objective function towards the experimental result over the refinement steps can be expected. Nevertheless, usually the experimental results are not available during the design phase and the numerical simulation shall be used during the design phase to predict and improve the wake field.

Hence, the applied strategy is driven by the practical optimisation approach and has been shown to be able to locally refine the grid in relevant regions for a better prediction of the wake field.

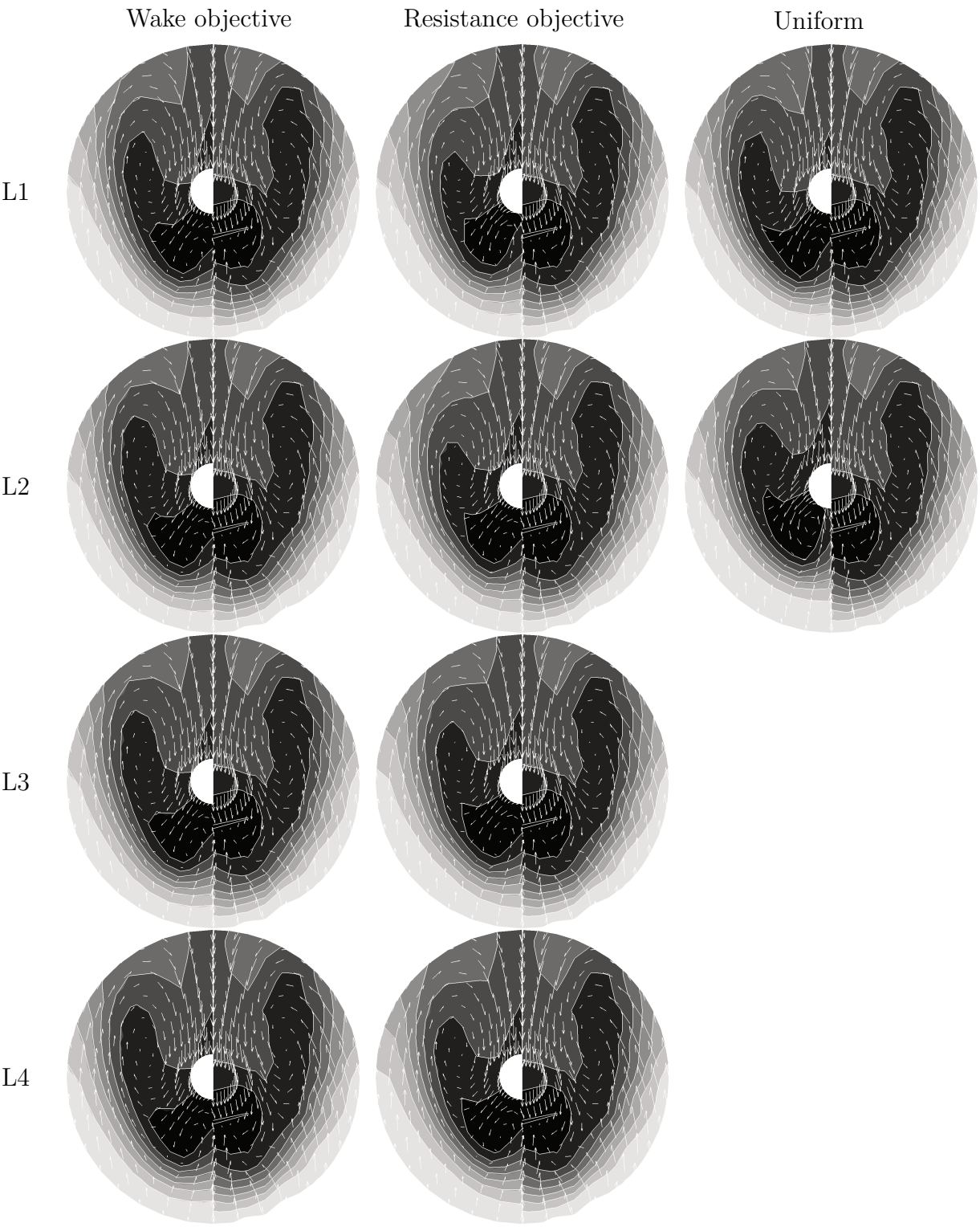


Figure 8.14: Predicted wake fields (left) compared to experiment (right) for different refinement methods with different maximum refinement levels.

9 Summary and Perspectives

The present thesis is confined to the improvement of the efficiency and accuracy of RANS-based finite volume flow solvers for marine flow problems. Two major items with significant impact on the efficiency and accuracy are identified.

The first item regards the discretisation of the free-surface flow discretisation. Usually VoF models, where an additional mixture fraction transport equation is solved, are used to represent free-surface flows in the finite volume framework. Within the VoF models compressive advection schemes are used to maintain a sharp interface representation. This is usually achieved by the use of downwind-biased advection schemes. However, the use of downwind biased schemes introduces the so called Courant number limitation, which requires small time steps and therefore increases the required computational effort. The small time steps are required to maintain the diagonal dominance of the system matrix, which is required for a stable solution of the equation system. When the time step is higher than required by the Courant number limitation the free-surface tends to get blurred and the simulation gets unstable. Furthermore, VoF based discretisations require transient simulation runs, even when the result is quasi-static, which introduces additional computational effort.

To counteract at least partly blurred interfaces an Explicit Interface Sharpening (EIS) technique has been developed. The EIS technique is able to detect blurred interfaces and to resharpen these interfaces. The sharpening is achieved by a conservative redistribution of the mixture fraction which is added to the right-hand side of the equation system for the mixture fraction transport equation.

The above described Courant number limitation is only valid for the transport equation of the mixture fraction. Therefore, a sub-cycling technique is introduced. Within the sub-cycling technique only the transport equation for the mixture fraction is solved using a small time step fulfilling the Courant number limitation. All other transport

equations are solved using larger time steps. In order to have the same advancement in time for all transport equations, the mixture fraction equation is solved multiple times within one outer iteration while all other transport equations are only solved once. Two different modes have been developed for the sub-cycling technique. The modes differ in their global advancement in time. The time discretisation requires values of the present time step and the previous time steps. Within the synchronised mode the old values to discretise the temporal term of the mixture fraction equation are taken from the previous time step. Therefore, the advancement in time of the mixture fraction equation and all other equations are synchronised. This mode is applicable for simulations where transient results are required. In the unsynchronised mode the old values used for the discretisation of the temporal term are taken from the previous solution of the the mixture fraction equation. Therefore, the mixture fraction equation has a faster advancement in time than the other transport equations. This mode is applicable for problems that lead to quasi-static solutions.

To appraise the developed interface sharpening technique, an interface sharpness index has been developed, which gives an indication of the interface sharpness in the whole domain as a single number. The interface sharpness is the arithmetic mean of present local approximate interface sharpness to the optimal interface sharpness ratio over the whole interface. The developed techniques have been intensively verified and validated. A detailed description of the verification and validation is given in Section 7.2 and 7.3. Here only the main results of the verification and validation are summarised.

The achievable speed-up using the sub-cycling technique in the synchronised mode is almost identical to the theoretical maximum speedup for quasi-steady and transient flow problems. The theoretical maximum speedup mainly depends on the ratio of solution time required to solve the mixture fraction transport equation and the other transport equations. It is therefore a simulation specific value. Within the verification and validation test cases speed-ups around 4.5 have been achieved using 8 sub-cycles. Using the unsynchronised sub-cycling mode for quasi-steady flow problems even higher speed-ups of about 8.5 are achieved. It should however be noted, that the achievable speed-up in the unsynchronised sub-cycling mode depends strongly on the number of employed outer iterations. The influence of the sub-cycling technique on the accuracy of the results is negligible.

The Explicit Interface Sharpening technique is shown to be able to maintain sharp

interfaces and even to resharpen blurred interfaces, which may occur even when dedicated compressive advection schemes are used for the mixture fraction transport equation. The influence of applying the EIS technique on the order of accuracy for the mixture fraction transport equation is insignificant. The accuracy of the results especially for long term simulations is drastically enhanced by the EIS technique, due to its ability to resharpen already blurred interfaces. Furthermore, it is shown that even non dedicated convection schemes can be used in combination with EIS for the transport of the mixture fraction and sharp interfaces are maintained.

The second identified item regards the mesh generation which has a significant impact on the accuracy and efficiency. Accurate results require high resolution meshes, however high resolution meshes lead to long answer times. On the other hand high resolution meshes are only required in regions where the high resolution has an impact on the result in question. Regions where high resolutions are required can usually only be identified on basis of the results of the flow problem. This leads to an iterative time consuming process. A convenient technique to improve this process is automatic mesh adaptation. The main building blocks are the mesh adaptation itself and an appropriate refinement indicator identifying the regions where refinement is required. In the present work an automatic mesh adaptation procedure is developed. It is based on the usage of unstructured grids, and features the anisotropic refinement of hexahedral cells. The refinement procedure is fully integrated into the parallel flow solver including a load balancing technique. A special treatment for the refinement of cells in the boundary layer is developed, to fulfil the requirements on the non-dimensional wall distance y^+ which depends on the employed wall resolution model. Furthermore, a technique to align the grid to curved boundaries, when the grid is refined at the boundaries, has been developed. It is based on a STL representation of the curved boundary which has to be provided. Different techniques to map the flow variables from the coarse grid to the refined grid are developed. Special care is taken to ensure conservative fluxes after the mapping procedure.

A number of different refinement indicators are developed, which all can be used to trigger the refinement algorithm as described above. The developed refinement indicators can be attributed to different classes. The most simple criterion is the free-surface refinement criterion, which is a pure feature based refinement criterion. It selects the regions for refinement only based on specific flow features, regardless of associated errors in the solution or the impact of errors in the solution towards the result in question. A more

elaborated indicator is the residual error estimator, which is designed to identify regions which inherit large numerical errors. However, no appraisal of the impact of the error towards the result in question is contained in the residual error estimator. The goal-oriented refinement indicator is able to detect the regions within the mesh, which are afflicted with errors that influence the result in question. To the authors knowledge, within this work the first application of goal-oriented refinement in the finite volume framework for complex three-dimensional turbulent flows has been presented.

The developed automatic grid-adaptation procedure is intensively verified and validated. It is shown that the refinement algorithm delivers similar refined meshes in serial and parallel runs. Furthermore, the ability to perform anisotropic refinement and boundary alignment in the refined regions are demonstrated. It is shown that the application of the residual error estimator leads to grids on which results converge towards reference solutions. For free-surface flow problems the combination of the free-surface criterion and the residual error estimator are shown to be beneficial. The goal-oriented mesh refinement technique is applied to laminar and turbulent flow problems. For laminar flows the adjoint error correction technique enhances the quality of the predicted quantities. However, the application of the adjoint error correction technique for turbulent flows is shown to be not reliable for the developed method. This is attributed to the frozen turbulence technique used in the adjoint flow solver. The comparison of grid-adaptation using the residual error estimator as criterion and the goal-oriented refinement indicator reveals that grid-adaptation using the goal-oriented refinement indicator is able to predict results of the same accuracy as grid-adaptation using the residual error estimator with significantly less cells.

All developed techniques have been applied to case studies, featuring complex three-dimensional turbulent flows. The EIS technique in conjunction with the sub-cycling technique in the unsynchronised mode is applied to the free-surface resistance prediction of the well known Kriso Tanker (KVLCC2). The predicted resistance shows a deviation from the experimentally predicted value of 1%, which is an excellent result. Furthermore, the predicted wave field shows a high accuracy even in a considerable distance from the hull. Due to the use of the sub-cycling technique an impressive speed-up of 33 has been realised in comparison to simulations without the application of EIS and sub-cycling. The sub-cycling technique in the synchronised mode in conjunction with the EIS technique are used within the prediction of the motion of the Kriso container vessel (KCS) in head

waves. The predicted pitch and heave motions are in fair agreement with the experimentally predicted values. A speed up of 5 compared to simulations without sub-cycling and EIS is achieved. The goal-oriented grid refinement technique is applied to the wake prediction for the Kriso tanker (KVLCC2). High quality results compared to the experimental results are achieved on a final grid using only 400k cells derived from a goal-oriented grid refinement strategy with the wake objective. To demonstrate the influence of the objective on the resulting grids an additional goal-oriented refinement study with a resistance objective is performed. It is clearly shown the grid-adaptation resulting from the goal-oriented refinement with the wake objective is superior for the prediction of the wake field. Results for simulations, which combine the grid-adaptation technique with the techniques for free-surface flows, are not presented due to minor stability issues associated to strongly anisotropic grid refinement in parallel.

Therefore, future work should at first concentrate on further stability enhancements for the developed grid-adaptation technique, especially considering highly anisotropic refinement in parallel environments. Within unsteady simulations, the regions where refined meshes are required, are usually non-residential. Hence, the grid refinement technique should be supplement by a grid de-refinement technique which provides access to a wide range of additional applications. Ideally the grid-refinement technique should be independent on the refinement history and therefore be able to refine even the initial grid. A result of the present thesis is that the adjoint error correction approach is not reliable for turbulent flows in conjunction with the used adjoint approximation. The present adjoint approximation is based on the frozen turbulence approach for the turbulence transport equations. It is assumed that the unreliability in the adjoint error prediction results from this approach. Therefore, an extension of the frozen turbulence approach by a complete differentiation should be beneficial.

Bibliography

[1] *Guide for the Verification and Validation of Computational Fluid Dynamics Simulations*. American Institute of Aeronautics & Astronautics, 1999.

[2] C. Abt and S. Harries. A New Approach to Integration of CAD and CFD for Naval Architects. In *International Conference on Computer Applications and Information Technology in the Maritime Industry*, 2007.

[3] H. Ahn and M. Shashkov. Adaptive Moment-of-Fluid Method. *Journal of Computational Physics*, 228(8):2792–2821, 2009.

[4] H. Ahn, M. Shashkov, and M. Christon. The Moment-of-Fluid Method in Action. *Communications in Numerical Methods in Engineering*, 25(10):1009–1018, 2009.

[5] S. Aland and A. Voigt. Benchmark Computations of Diffuse Interface Models for Two-Dimensional Bubble Dynamics. *International Journal for Numerical Methods in Fluids*, 69(3):747–761, 2011.

[6] F. Alauzet, P. Frey, P. George, and B. Mohammadi. 3D Transient Fixed Point Mesh Adaptation for Time-Dependent Problems: Application to CFD Simulations. *Journal of Computational Physics*, 222(2):592–623, 2007.

[7] N. Ashgriz and J. Poo. FLAIR: Flux Line-Segment Model for Advection and Interface Reconstruction. *Journal of Computational Physics*, 93(2):449–468, 1991.

[8] E. Aulisa, S. Manservisi, and R. Scardovelli. A mixed Markers and Volume-of-Fluid Method for the Reconstruction and Advection of Interfaces in Two-Phase and Free-Boundary Flows. *Journal of Computational Physics*, 188(2):611–639, 2003.

[9] R. Babaei, J. Abdollahi, P. Homayonifar, N. Varahram, and P. Davami. Improved Advection Algorithm of Computational Modeling of Free Surface Flow using Struc-

tured Grids. *Computer Methods in Applied Mechanics and Engineering*, 195(7-8):775–795, 2006.

[10] I. Babuška and W. C. Rheinboldt. A-posteriori Error Estimates for the Finite Element Method. *International Journal for Numerical Methods in Engineering*, 12(10):1597–1615, 1978.

[11] I. Babuška and W. C. Rheinboldt. Error Estimates for Adaptive Finite Element Computations. *SIAM Journal on Numerical Analysis*, 15(4):736–754, 1978.

[12] R. Balasubramanian and J. Newman III. Comparison of Adjoint-based and Feature-based Grid Adaptation for Functional Outputs. *International Journal for Numerical Methods in Fluids*, 53(53):1541–1569, 2007.

[13] S. Balay, J. Brown, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang. PETSc Users Manual. Technical Report ANL-95/11 - Revision 3.3, Argonne National Laboratory, 2012.

[14] S. Balay, J. Brown, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang. PETSc Web page, 2012. http://www.mcs.anl.gov/petsc.

[15] S. Balay, W. Gropp, L. Curfman McInnes, and B. Smith. Efficient Management of Parallelism in Object Oriented Numerical Software Libraries. In *Modern Software Tools in Scientific Computing*, pages 163–202, 1997.

[16] R. E. Bank and M. Holst. A new Paradigm for Parallel Adaptive Meshing Algorithms. *SIAM review*, pages 291–323, 2003.

[17] R. E. Bank, A. H. Sherman, and A. Weiser. Some Refinement Algorithms and Data Structures for Eegular Local Mesh Refinement. *Scientific Computing, Applications of Mathematics and Computing to the Physical Sciences*, 1:3–17, 1983.

[18] J. Bardin, P. Huang, and T. Coakley. Turbulence Modeling Validation, Testing, and Development. Technical Memorandum 110446, NASA, 1997.

[19] L. Baskett and R. Haimes. Feature Extraction of Shear Layers. In *AIAA Computational Fluid Dynamics Conference, 15 th, Anaheim, CA*, 2001.

[20] R. Becker. Weighted Error Estimators for the Incompressible Navier-Stokes Equations. Rapport de recherche 3458, INRIA, 1998.

[21] R. Becker and R. Rannacher. Weighted A Posteriori Error Control in Finite Element Methods. In *ENUMATH-97*, pages 621–637, 1998.

[22] A. Belme, A. Dervieux, and F. Alauzet. Fully Anisotropic Goal-Oriented Mesh Adaptation for Unsteady Flows. In *V European Conference on Computational Fluid Dynamics*, 2010. Papers2.

[23] E. Berberović. *Investigation of Free-surface Flow Associated with Drop Impact: Numerical Simulations and Theoretical Modeling*. PhD thesis, Technische Universität Darmstadt, 2010.

[24] M. J. Berger and P. Colella. Local Adaptive Mesh Refinement for Shock Hydrodynamics. *Journal of Computational Physics*, 82(1):64–84, 1989.

[25] M. J. Berger and J. Oliger. Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations. *Journal of Computational Physics*, 53(3):484–512, 1984.

[26] P. Bohorquez. *Study and Numerical Simulation of Sediment Transport in Free-Surface Flow*. PhD thesis, University of Malage, 2008.

[27] BoxLib. `https://ccse.lbl.gov/BoxLib/index.html`.

[28] P. Bradshaw. The turbulence structure of equilibrium boundary layers. *Journal of Fluid Mechanics*, 29(4):625–645, 1967.

[29] A. Brandt. Multi-Level Adaptive Solutions. *Mathematics of Computation. v31 i138*, pages 333–390, 1977.

[30] E. F. Campana, T. Hino, P. Bull, P. Carrica, J. Kim, S.-E. Kim, L. D.-Q., I. Saisto, and B. Starke. The Specialist Committee on Computational Fluid Dynamics - Final Report and Recommendations to the 26th ITTC. In *Proccedings of the 26th ITTC - Volume II*, 2011.

[31] S. C. Caruso, J. Ferziger, and J. Oliger. Adaptive Grid Techniques for Elliptic Fluid-Flow Problems. *NASA STI/Recon Technical Report N*, 87:12814, 1985.

[32] J. G. Castanos and J. E. Savage. Parallel Eefinement of Unstructured Meshes. In *Proc. of IASTED International Conference Parallel and Distributed Computing and Systems*. Citeseer, 1999.

[33] M. Ceze and K. J. Fidkowski. Output-driven Anisotropic Mesh Adaptation for Viscous Flows using Discrete Choice Optimization. *AIAA*, 170:2010, 2010.

[34] CGNS. `www.cgns.org`.

[35] Chombo. `https://commons.lbl.gov/display/chombo/Chombo+-+Software+for+Adaptive+Solutions+of+Partial+Differential+Equations`.

[36] A. Chorin. Numerical Solution of the Navier-Stokes Equations. *Mathematics of Computation*, 22(104):745–762, 1968.

[37] R. Clift, G. J. R., and W. M. E. *Bubbles Drops and Particles*. Academic Press, New York :, 1978.

[38] R. Coleman. Nonlinear Flow about a Three-Dimensional Transom Stern. In *International Conference on Numerical Ship Hydrodynamics, 4th*, 1981.

[39] R. Courant, E. Isaacson, and M. Rees. On the Solution of Nonlinear Hyperbolic Differential Equations by Finite Differences. *Communications on Pure and Applied Mathematics*, 5(3):243–255, 1952.

[40] B. Daly. Numerical Study of Two Fluid Rayleigh-Taylor Instability. *Physics of Fluids*, 10(2):297–307, 1967.

[41] B. Daly. A Technique for Including Surface Tension Effects in Hydrodynamic Calculations. *Journal of Computational Physics*, 4(1):97–117, 1969.

[42] S. F. Davis. Flux Difference Splittings and Limiters for the Resolution of Contact Discontinuities. *Applied Mathematics and Computation*, 65(1-3):3 – 18, 1994.

[43] A. Di Mascio, R. Muscari, and R. Broglia. A Level Set Approach for Naval Applications. In *Proceedings of 13th International Offshore and Polar Engineering Conference*, volume 3, pages 220–6, 2003.

[44] D. Di Pietro, S. Lo Forte, and N. Parolini. Mass Preserving Finite Element Implementations of the Level Set Method. *Applied Numerical Mathematics*, 56(9):1179–1195, 2006.

[45] L. F. Diachin, R. Hornung, P. Plassmann, and A. Wissink. *Parallel Adaptive Mesh Refinement*. SIAM, 2006.

[46] M. Doring, Y. Andrillon, B. Alessandrini, and P. Ferrant. SPH Free Surface Flow Simulation. In *18th International Workshop on Water Waves and Floating Bodies, Le Croisic, 6th-9th April*, 2003.

[47] J. Duncan. An Experimental Investigation of Breaking Waves produced by a Towed Hydrofoil. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 377(1770):331–348, 1981.

[48] R. P. Dwight. Heuristic a Posteriori Estimation of Error due to Dissipation in Finite Volume Schemes and Application to Mesh Adaptation. *Journal of Computational Physics*, 227(5):2845–2863, 2008.

[49] V. Dyadechko and M. Shashkov. Moment-of-Fluid Interface Reconstruction. *Los Alamos report LA-UR-05-7571*, 2006.

[50] L. Eça and M. Hoekstra. Testing Uncertainty Estimation and Validation Procedures in the Flow around a Backward Facing Step. In *3rd Workshop on CFD*, 2008.

[51] D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell. A Hybrid Particle Level Set Method for Improved Interface Capturing. *Journal of Computational Physics*, 183(1):83–116, 2002.

[52] J. Farmer, L. Martinelli, and A. Jameson. A fast Multigrid Method for Solving the Nonlinear Ship Wave Problem with a Free Surface. In *Proceedings 6th International Conference on Numerical Ship Hydrodynamics, Iowa*, 1993.

[53] J. H. Ferziger and M. Perić. Further Discussion of Numerical Errors in CFD. *International Journal for Numerical Methods in Fluids*, 23(12):1263–1274, 1996.

[54] J. H. Ferziger and M. Perić. *Computational Methods for Fluid Dynamics*, volume 3. Springer Berlin etc, 1999.

[55] P. Frey and F. Alauzet. Anisotropic Mesh Adaptation for CFD Computations. *Computational Methods in Applied Mechanics and Engineering*, 194(194):5068–5082, 2005.

[56] P. Gaskell and A. Lau. Curvature-compensated Convective Transport: Smart, a new Boundedness-Preserving Transport Algorithm. *International Journal for Numerical Methods in Fluids*, 8(6):617–641, 1988.

[57] U. Ghia, K. Ghia, and C. Shin. High-Re Solutions for Incompressible Flow using the Navier-Stokes Equations and a Multigrid Method. *Journal of Computational Physics*, 48(3):387–411, 1982.

[58] M. Giles. On Adjoint Equations for Error Analysis and Optimal Grid Adaptation in CFD, 1997.

[59] M. Giles and N. Pierce. An Introduction to the Adjoint Approach to Design. *Flow, Turbulence and Combustion*, 65(3):393–415, 2000.

[60] M. Giles and N. A. Pierce. Adjoint Equations in CFD: Duality, Boundary Conditions and Solution Behaviour. In *AIAA Computational Fluid Dynamics Conference*, 1997.

[61] B. Godderidge, M. Tan, S. Turnock, and C. Earl. A Verification and Validation Study of the Application of Computational Fluid Dynamics to the Modelling of Lateral Sloshing. Ship Science Report 140, University of Southampton, 2006.

[62] B. Godderidge, S. Turnock, M. Tan, and C. Earl. An Investigation of Multiphase CFD Modelling of a Lateral Sloshing Tank. *Computers & Fluids*, 38(2):183–193, 2009.

[63] V. Gopala and B. van Wachem. Volume of Fluid Methods for Immiscible-Fluid and Free-Surface Flows. *Chemical Engineering Journal*, 141(1):204–221, 2008.

[64] A. Griewank and A. Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Society for Industrial and Applied Mathematics (SIAM), 2008.

[65] D. Guégan, O. Allain, A. Dervieux, and F. Alauzet. An $l^\infty - l^p$ Mesh-Adaptive Method for Computing Unsteady Bi-fluid Flows. *International Journal for Numerical Methods in Engineering*, 84:1376–1406, 2010.

[66] W. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, and M. Vallet. Anisotropic Mesh Adaptation: Towards User-Independent Mesh-Independent and Solver-Independent CFD. Part I: General Principles. *International Journal for Numerical Methods in Fluids*, 32(32):725–744, 2000.

[67] D. Hafermann. STREAMLINE - D32.2 - Parallelisation and Load Balancing in Dynamic Grid Control. 2012.

[68] K. Hannah. Back to the Future: Trends in Commercial CFD. In *Proceedings of the International CAE Conference 2012*, 2012.

[69] F. Harlow and J. Welch. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *Physics of Fluids*, 8:2182, 1965.

[70] A. Harten. The Artificial Compression Method for Computation of Shocks and Contact Discontinuities. I-Single Conservation Laws. *Communications in Pure Applied Mathematics*, 30:611–638, 1977.

[71] A. Harten. High Resolution Schemes for Hyperbolic Conservation Laws. *Journal of Computational Physics*, 49(3):357–393, 1983.

[72] R. Hartmann. Multitarget Error Estimation and Adaptivity in Aerodynamic Flow Simulations. *SIAM Journal on Scientific Computing*, 31(1):708–731, 2008.

[73] R. Hartmann, J. Held, and T. Leicht. Adjoint-based Error Estimation and Adaptive Mesh Refinement for the RANS and k-ω Turbulence Model Equations. *Journal of Computational Physics*, 230(11):4268–4284, 2011.

[74] R. Hartmann and P. Houston. Goal-Oriented A Posteriori Error Estimation for Multiple Target Functionals. In *Hyperbolic Problems: Theory, Numerics, Applications*, pages 579–588. Springer Berlin Heidelberg, 2003.

[75] A. Hay and M. Visonneau. CFD Uncertainty Analysis for Turbulent Flows using Error Equation Method. In *Workshop on CFD Uncertainty Analysis Lisbon*, 2004.

[76] F. Heimann, C. Engwer, O. Ippisch, and P. Bastian. An Unfitted Interior Penalty Discontinuous Galerkin Method for Incompressible Navier-Stokes Two-Phase Flow. *International Journal for Numerical Methods in Fluids*, 71(3):269–293, 2012.

[77] J. Heyns, A. Malan, T. Harms, and O. Oxtoby. Development of a Compressive Surface Capturing Formulation for Modelling Free-Surface Flow by using the Volume-of-Fluid Approach. *International Journal for Numerical Methods in Fluids*, 71(6):788–804, 2012.

[78] M. Hinatsu, Y. Tsukada, R. Fusakawa, and Y. Tanaka. Two-phase Flows for Joint Research. In *Proceedings of the SRI-TUHH Mini Workshop on Numerical Simulation of Two-Phase Flows.*, 2001.

[79] T. Hino, L. Martinelli, and A. Jameson. A Finite-volume Method with Unstructured Grid for Free Surface Flow Simulations. In *Sixth International Conference on Numerical Ship Hydrodynamics*, 1993.

[80] C. Hirsch. *Numerical Computation of Internal and External Flows: The Fundamentals of Computational Fluid Dynamics*, volume 1. Butterworth-Heinemann, 2007.

[81] C. Hirt and B. Nichols. Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries. *Journal of Computational Physics*, 39(1):201–225, 1981.

[82] C. Hirt and J. Shannon. Free-Surface Stress Conditions for Incompressible-Flow Calculations. *Journal of Computational Physics*, 2(4):403–411, 1968.

[83] P. Houston, E. H. Georgoulis, and E. Hall. Adaptivity and A Posteriori Error Estimation for DG Methods on Anisotropic Meshes. In *Proceedings of the International Conference on Boundary and Interior Layers (BAIL)-Computational and Asymptotic Methods*, 2006.

[84] S. Hysing. *Performance in Two-Phase Flow Simulations: Finite Elements and the Level Set Method*. PhD thesis, Dortmund University, Institute of Applied Mathematics and Numerics (LS3), 2007.

[85] S. Hysing. Mixed Element FEM Level Set Method for Numerical Simulation of Immiscible Fluids. *Journal of Computational Physics*, 2011.

[86] S. Hysing, S. Turek, D. Kuzmin, N. Parolini, E. Burman, S. Ganesan, and L. Tobiska. Quantitative Benchmark Computations of Two-Dimensional Bubble Dynamics. *International Journal for Numerical Methods in Fluids*, 60(11):1259–1288, 2009.

[87] A. Jameson. Aerodynamic Design via Control Theory. *Journal of Scientific Computing*, 3(3):233–260, 1988.

[88] H. Jasak. *Error Analysis and Estimation for the Finite Volume Method with Applications to Fluid Flows*. PhD thesis, University of London, 1996.

[89] H. Jasak and A. Gosman. Automatic Resolution Control for the Finite-Volume Method, Part 2: Adaptive Mesh Refinement and Coarsening. *Numerical Heat Transfer: Part B: Fundamentals*, 38(3):257–271, 2000.

[90] H. Jasak, H. Weller, and A. Gosman. High Resolution NVD Differencing Scheme for Arbitrarily Unstructured Meshes. *International Journal for Numerical Methods in Fluids*, 31(2):431–449, 1999.

[91] M. Jemison, E. Loch, M. Sussman, M. Shashkov, M. Arienti, M. Ohta, and Y. Wang. A Coupled Level Set-Moment of Fluid Method for Incompressible Two-Phase Flows. *Journal of Scientific Computing*, pages 1–38, 2012.

[92] J. Jeong and D. Yang. Finite Element Analysis of Transient Fluid Flow with Free Surface using VOF (Volume-of-Fluid) Method and Adaptive Grid. *International Journal for Numerical Methods in Fluids*, 26(10):1127–1154, 1998.

[93] L. Jofre, O. Lehmkuhl, J. Castro, and A. Oliva. A PLIC-VOF Implementation on Parallel 3D Unstructured Meshes. In *Proceedings of the Fifth European Conference on Computational Fluid Dynamics ECCOMAS CFD*, volume 2010, 2010.

[94] L. Jofre Cruanyes, O. Lehmkuhl Barba, R. Borrell Pol, J. Castro González, A. Oliva Llena, et al. Parallelization Study of a VOF/Navier-Stokes Model for 3D Unstructured Staggered Meshes. In *Parallel CFD 2011 : 23rd International Conference on Parallel Computational Fluid Dynamics*, 2012.

[95] M. T. Jones and P. E. Plassmann. Parallel Algorithms for Adaptive Mesh Refinement. *SIAM Journal on Scientific Computing*, 18(3):686–708, 1997.

[96] W. Jones and B. Launder. Predicition of Laminarization with a Two-Equation Turbulence Model. *International Journal of Heat and Mass Transfer*, 1972.

[97] Y. Kallinderis and A. Vidwans. Generic Parallel Adaptive-grid Navier-Stokes Algorithm. *AIAA Journal*, 32(1):54–61, 1994.

[98] G. Karypis and V. Kumar. A Parallel Algorithm for Multilevel Graph Partitioning and Sparse Matrix Ordering. *Journal of Parallel and Distributed Computing*, 48:71 – 85, 1998.

[99] W. Kim, S. Van, and D. Kim. Measurement of Flows around Modern commercial Ship Models. *Experiments in Fluids*, 31:567–578., 2001.

[100] J. Klostermann, K. Schaake, and R. Schwarze. Numerical Simulation of a Single Rising Bubble by VOF with Surface Compression. *International Journal for Numerical Methods in Fluids*, 2012.

[101] S.-H. Kwag. Computation of Water and Air Flow with Submerged Hydrofoil by Interface Capturing Method. *KSME International Journal*, 14(7):789–795, 2000.

[102] B. Lafaurie, C. Nardone, R. Scardovelli, S. Zaleski, and G. Zanetti. Modelling Merging and Fragmentation in Multiphase Flows with SURFER. *Journal of Computational Physics*, 113(1):134–147, 1994.

[103] L. Larsson, F. Stern, and M. Visonneau. CFD in Ship Hydrodynamics - Results of the Gothenburg 2010 Workshop. In *MARINE 2011, Computational Methods in Marine Engineering IV*, 2011.

[104] E. Lee-Rausch, M. Park, W. Jones, D. Hammond, and E. Nielsen. Application of Parallel Adjoint-Based Error Estimation and Anisotropic Grid Adaptation for Three-Dimensional Aerospace Configurations. In *23rd AIAA Applied Aerodynamics Conference*. American Institute of Aeronautics and Astronautics, 2005.

[105] B. Leonard. A stable and accurate Convective Modelling Procedure based on Quadratic Upstream Interpolation. *Computer Methods in Applied Mechanics and Engineering*, 19(1):59–98, 1979.

[106] B. Leonard. The ULTIMATE Conservative Difference Scheme applied to unsteady One-dimensional Advection. *Computer Methods in Applied Mechanics and Engineering*, 88(1):17–74, 1991.

[107] B. P. Leonard. Simple High-accuracy Resolution Program for Convective Modelling of Discontinuities. *International Journal for Numerical Methods in Fluids*, 8(10):1291–1318, 1988.

[108] A. Leroyer, P. Queutey, E. Guilmineau, G. Deng, and M. Visonneau. New Algorithms to speed up RANSE Computations in Hydrodynamics. In *12th Numerical Towing Tank Symposium*, 2009.

[109] L. Y. Li, Y. Allaneau, and A. Jameson. Continuous Adjoint Approach for Adaptive Mesh Refinement. *AIAA Paper*, pages 11–3982, 2011.

[110] F. Lien and M. Leschziner. Upstream Monotonic Interpolation for Scalar Transport with Application to Complex turbulent Flows. *International Journal for Numerical Methods in Fluids*, 19(6):527–548, 1994.

[111] P. Liovic, M. Rudman, J. Liow, D. Lakehal, and D. Kothe. A 3D Unsplit-Advection Volume Tracking Algorithm with Planarity-preserving Interface Reconstruction. *Computers & Fluids*, 35(10):1011–1032, 2006.

[112] K. Lipnikov and Y. Vassilevski. Error Estimates for Hessian-based Mesh Adaptation Algorithms with Control of Adaptivity. In *13th International Meshing Roundtable*, 2004.

[113] R. Löhner. *Adaptive Mesh Refinement*, pages 269–297. John Wiley & Sons, Ltd, 2008.

[114] J. López, J. Hernández, P. Gómez, and F. Faura. A Volume of Fluid Method based on Multidimensional Advection and Spline Interface Reconstruction. *Journal of Computational Physics*, 195(2):718–742, 2004.

[115] A. Loseille, A. Dervieux, and F. Alauzet. Fully Anisotropic Goal-oriented Mesh Adaptation for 3D Steady Euler Equations. *Journal of Computational Physics*, 229(8):2866–2897, 2010. Papers2.

[116] P. Lötstedt. A Front Tracking Method applied to Burgers' Equation and Two-Phase Porous Flow. *Journal of Computational Physics*, 47(2):211–228, 1982.

[117] J. Martin and W. Moyce. An Experimental Study of the Collapse of liquid Columns on a rigid Horizontal Plane. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 244(882):312–324, 1952.

[118] F. Menter. Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications. *AIAA Journal*, 32:1598–1605, 1994.

[119] F. Menter, M. Kuntz, and R. Langtry. Ten Years of Industrial Experience with the SST Turbulence Model. In *4th Int. Symp. Turbulence, Heat Mass Transfer*, pages 625–632, Egell House, New York, 2003.

[120] Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard, Version 2.2.* High Performance Computing Center Stuttgart (HLRS), September 2009.

[121] W. F. Mitchell. A Comparison of Adaptive Refinement Techniques for Elliptic Problems. *ACM Transactions on Mathematical Software (TOMS)*, 15(4):326–347, 1989.

[122] B. Mohammadi. Optimal Shape Design, Reverse Mode of Automatic Differentiation and Turbulence. In *Aerospace Sciences Meetings*, pages –. American Institute of Aeronautics and Astronautics, Jan. 1997.

[123] K. W. Morton. *Numnerical Methods for Fluid Dynamics*, chapter Time Dependant Mulit-Matrial Flow With large Fluid Distortion, pages 273–285. Academic Press, 1982.

[124] S. Muzaferija. *Adaptive Finite Volume Method for Flow Prediction using Unstructured Meshes and Multigrid Approach.* PhD thesis, Imperial College London, 1994.

[125] S. Muzaferija, M. Peric, P. Sames, and T. Schelin. A Two-fluid Navier-Stokes Solver to simulate Water Entry. In *Twenty-Second Symposium on Naval Hydrodynamics*, 1998.

[126] S. Nadarajah and A. Jameson. Studies of the Continuous and Discrete Adjoint Approaches to Viscous Automatic Aerodynamic Shape Optimization. *AIAA Paper*, 2530:2001, 2001.

[127] W. Noh and P. Woodward. SLIC (Simple Line Interface Calculation). In *Proceedings of the Fifth International Conference on Numerical Methods in Fluid Dynamics June 28 - July 2, 1976 Twente University, Enschede*, volume 59 of *Lecture Notes in Physics*, pages 330–340. Springer Berlin Heidelberg, 1976.

[128] E. Olsson and G. Kreiss. A conservative Level Set Method for Two Phase Flow. *Journal of Computational Physics*, 210(1):225–246, 2005.

[129] E. Olsson, G. Kreiss, and S. Zahedi. A conservative Level Set Method for Two Phase Flow II. *Journal of Computational Physics*, 225(1):785–807, 2007.

[130] openFOAM. `www.openfoam.org`.

[131] S. Osher and S. Chakravarthy. High Resolution Schemes and the Entropy Condition. *SIAM Journal on Numerical Analysis*, 21(5):955–984, 1984.

[132] S. Osher and J. Sethian. Fronts propagating with Curvature-dependent Speed: Algorithms based on Hamilton-Jacobi Formulations. *Journal of Computational Physics*, 79(1):12–49, 1988.

[133] I. R. Park, K. S. Kim, J. Kim, and S. H. Van. A Volume-of-Fluid Method for Incompressible Free Surface Flows. *International Journal for Numerical Methods in Fluids*, 61(12):1331–1362, 2009.

[134] M. A. Park. *Anisotropic Output-based Adaptation with Tetrahedral Cut Cells for Compressible Flows*. PhD thesis, Massachusetts Institute of Technology, 2008.

[135] M. A. Park and J.-R. Carlson. Turbulent Output-based Anisotropic Adaptation. *AIAA Paper*, 168:2010, 2010.

[136] N. Parolini. *Computational Fluid Dynamics for Naval Engineering Problems*. PhD thesis, Section de Mathématiques pour l'obtention du grade de Docteur ès Sciences par Laurea in Ingegneria Aerospaziale, Politecnico di Milano, 2004.

[137] N. Parolini and E. Burman. A Finite Element Level Set Method for Viscous Free-surface Flows. *Series on Advances in Mathematics for Applied Sciences*, 69:416, 2005.

[138] S. Patankar. *Numerical Heat Transfer and Fluid Flow*. Taylor and Francis, 1980.

[139] S. Patankar and D. Spalding. A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-dimensional Parabolic Flows. *International Journal of Heat and Mass Transfer*, 15(10):1787–1806, 1972.

[140] J. Peraire, J. Peiro, and K. Morgan. A 3D Finite Element Multigrid Solver for the Euler Equations. In *AIAA Materials Specialist Conference-Coating Technology for Aerospace Systems*, volume 1, 1992.

[141] M. Perić. *A Finite Volume Method for the Prediction of Three-dimensional Fluid Flow in Complex Ducts.* PhD thesis, London University, 1985.

[142] M. Perić and T. Zorn. Simulation of Sloshing Loads on Moving Tanks. In *ASME 2005 24th International Conference on Offshore Mechanics and Arctic Engineering*, volume 3. ASME, 2005.

[143] M. Perić, T. Zorn, O. el Moctar, T. Schellin, and Y. Kim. Simulation of Sloshing in LNG-tanks. *Journal of Offshore Mechanics and Arctic Engineering*, 131:031101, 2009.

[144] J. E. Peter and R. P. Dwight. Numerical Sensitivity Analysis for Aerodynamic Optimization: A Survey of Approaches. *Computers & Fluids*, 39(3):373–391, 2010.

[145] N. A. Pierce and M. Giles. Adjoint Recovery of Superconvergent Functionals from PDE Approximations. *SIAM Rev.*, 42(2):247–264, 2000.

[146] N. A. Pierce and M. B. Giles. Adjoint and Defect Error Bounding and Correction for Functional Estimates. *Journal of Computational Physics*, 200(2):769–794, 2004.

[147] J. Pilliod Jr and E. Puckett. Second-order accurate Volume-of-Fluid Algorithms for Tracking Material Interfaces. *Journal of Computational Physics*, 199(2):465–502, 2004.

[148] D. Poirier, S. R. Allmaras, D. R. McCarthy, M. F. Smith, and F. Y. Enomoto. The CGNS System. *AIAA Paper*, pages 98–3007, 1998.

[149] Y. Qin and T. Shih. A Discrete Transport Equation for Error Estimation in CFD. *AIAA Paper*, 906, 2002.

[150] J. Ramshaw and J. Trapp. A Numerical Technique for Low-speed Homogeneous Two-phase Flow with Sharp Interfaces. *Journal of Computational Physics*, 21(4):438–453, 1976.

[151] C. M. Rhie and W. L. Chow. Numerical Study of the turbulent Flow past an Airfoil with Trailing Edge Separation. *AIAA Journal*, 21:1525–1532, Nov. 1983.

[152] W. J. Rider and D. B. Kothe. Reconstructing Volume Tracking. *Journal of Computational Physics*, 141(2):112 – 152, 1998.

[153] M. C. Rivara. Algorithms for Refining Triangular Grids suitable for Adaptive and Multigrid Techniques. *International Journal for Numerical Methods in Engineering*, 20:745–756, 1984.

[154] M.-C. Rivara. Mesh Refinement Processes based on the Generalized Bisection of Simplices. *SIAM Journal on Numerical Analysis*, 21(3):604–613, 1984.

[155] P. Roache. *Verification and Validation in Computational Science and Engineering*. Bertrams, 1998.

[156] P. Roe. Characteristic-based Schemes for the Euler Equations. *Annual Review of Fluid Mechanics*, 18(1):337–365, 1986.

[157] C. J. Roy. Strategies for driving Mesh Adaptation in CFD. *AIAA*, 1302:5–8, 2009.

[158] C. J. Roy and A. J. Sinclair. On the Generation of Exact Solutions for Evaluating Numerical Schemes and Estimating Discretization Error. *Journal of Computational Physics*, 228(5):1790–1802, 2009.

[159] T. Rung. Vorlesungsmanuskript: Statistische Turbulenzmodellierung. Technical report, Institut für Fluiddynamik und Schiffstheorie, Technische Universität Hamburg-Harburg, 2008.

[160] H. Rusche. *Computational Fluid Dynamics of Dispersed Two-phase Flows at High Phase Fractions*. PhD thesis, Imperial College, London., 2002.

[161] A. Salih and S. Moulic. A Level Set Formulation for the Numerical Simulation of Impact of Surge Fronts. *Sadhana*, 31(6):697–707, 2006.

[162] Y. Sato and B. Ničeno. A Conservative local Interface Sharpening Scheme for the onstrained Interpolation Profile Method. *International Journal for Numerical Methods in Fluids*, 2012.

[163] J. Sauer. *Instationär kavitierende Strömungen-Ein neues Modell, basierend auf Front Capturing (VOF) und Blasendynamik*. PhD thesis, Universität Karlsruhe, 2000.

[164] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher. Benchmark Computations of Laminar Flow around a Cylinder. *Notes on Numerical Fluid Mechanics*, 52:547–566, 1996.

[165] J. Sethian and P. Smereka. Level Set Methods for Fluid Interfaces. *Annual Review of Fluid Mechanics*, 35(1):341, 2003.

[166] E. Sewell. *Automatic Generation of Triangulations for Piecewise Polynomial Approximation.* PhD thesis, Purdue University, 1972.

[167] C. Simonsen, J. Otzen, and F. Stern. EFD and CFD for KCS Heaving and Pitching in Regular Head Waves. In *Proc. 27th Symp. Naval Hydrodynamics*, 2008.

[168] J. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, and D. Mavriplis. CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences. Technical report, NASA Langley Research Center, 2013.

[169] G. Son. Efficient Implementation of a coupled Level-Set and Volume-of-Fluid Method for Three-dimensional incompressible Two-phase Flows. *Numerical Heat Transfer: Part B: Fundamentals*, 43(6):549–565, 2003.

[170] L. Štrubelj and I. Tiselj. Two-fluid model with interface sharpening. *International Journal for Numerical Methods in Engineering*, 85(5):575–590, 2011.

[171] A. Stück. *Adjoint Navier Stokes Methods for Hydrodynamic Shape Optimisation.* PhD thesis, Hamburg University of Technology, 2012.

[172] A. Stück, J. Kröger, and T. Rung. Adjoint-based Hull Design for Wake Optimisation. *Ship Technology Research*, 58(1):34–44, 2011.

[173] A. Stück and T. Rung. Adjoint Complement to Viscous Finite-Volume Pressure-Correction Methods. *Journal of Computational Physics*, 2013.

[174] M. Sussman and E. Puckett. A coupled Kevel Set and Volume-of-Fluid Method for Computing 3D and axisymmetric Incompressible Two-phase Flows. *Journal of Computational Physics*, 162(2):301–337, 2000.

[175] M. Sussman, P. Smereka, and S. Osher. *A Level Set Approach for Computing Solutions to Incompressible Two-phase Flow.* Department of Mathematics, University of California, Los Angeles, 1994.

[176] P. Sweby. High resolution Schemes using Flux Limiters for Hyperbolic Conservation Laws. *SIAM Journal on Numerical Analysis*, 21(5):995–1011, 1984.

[177] A. Takizawa, S. Koshizuka, and S. Kondo. Generalization of Physical Component Boundary fitted Co-ordinate (PCBFC) Method for the Analysis of Free-surface Flow. *International Journal for Numerical Methods in Fluids*, 15(10):1213–1237, 1992.

[178] M. Thompson and J. Ferziger. An Adaptive Multigrid Technique for the Incompressible Navier-Stokes Equations. *Journal of Computational Physics*, 82(1):94–121, 1989.

[179] G. Tzabiras. A Numerical Investigation of 2D, Steady Free Surface Flows. *International Journal for Numerical Methods in Fluids*, 25(5):567–598, 1997.

[180] O. Ubbink. *Numerical Prediction of two fluid Systems with sharp Interfaces.* PhD thesis, University of London, 1997.

[181] S. H. Van, K. W. J., D. H. Kim, G. T. Yim, and C. J. Lee. Experimental Investigation of the Flow Characteristics around Practical Hull Forms. In *3rd Osaka Colloquium on Advanced CFD Applications to Ship Flow and Hull Form Design*, 1998.

[182] S. H. Van, K. W. J., D. H. Kim, G. T. Yim, C. J. Lee, and J. Y. Eom. Flow Measurement around a 300K VLCC Model. In *Annual Spring Meeting SNAK*, 1998.

[183] S. Van der Pijl, A. Segal, C. Vuik, and P. Wesseling. A mass-conserving Level-Set Method for Modelling of Multi-phase Flows. *International Journal for Numerical Methods in Fluids*, 47(4):339–361, 2005.

[184] B. van Leer. Towards the Ultimate Conservative Difference Scheme. V. A Second-order Sequel to Godunov's Method. *Journal of Computational Physics*, 32(1):101–136, 1979.

[185] B. van Leer. Towards the Ultimate Conservative Difference Scheme. *Journal of Computational Physics*, 135(2):229–248, 1997.

[186] G. Vaz, F. Jaouen, and M. Hoekstra. Free Surface Viscous Flow Computations. Validation of URANS Code FreSCo. In *Proceedings of OMAE2009, Honolulu, Hawaii, USA, June*, 2009.

[187] D. Venditti and D. Darmofal. Grid Adaptation for Functional Outputs: Application to Two-Dimensional Inviscid Flows. *Journal of Computational Physics*, 176(176):40–69, 2002. Papers2.

[188] D. A. Venditti and D. L. Darmofal. Adjoint Error Estimation and Grid Adaptation for Functional Outputs: Application to Quasi-one-dimensional Flow. *Journal of Computational Physics*, 164(1):204–227, 2000.

[189] D. A. Venditti and D. L. Darmofal. Anisotropic Grid Adaptation for Functional Outputs: Application to Two-dimensional Viscous Flows. *Journal of Computational Physics*, 187(1):22–46, 2003.

[190] R. Vilsmeier and D. Hänel. Adaptive Methods on Unstructured Grids for Euler and Navier-Stokes Equations. *Computers & Fluids*, 22(4):485–499, 1993.

[191] M. Visonneau, G. B. Deng, P. Queutey, J. Wackers, and B. Mallol. Anisotropic Grid Adaptation for RANS Simulation of a fast manoeuvring Catamaran. In *4th High Performance Yacht Design Conference*, 2012.

[192] J. Wackers, B. Koren, H. Raven, A. van der Ploeg, A. Starke, G. Deng, P. Queutey, M. Visonneau, T. Hino, and K. Ohashi. Free-Surface Viscous Flow Solution Methods for Ship Hydrodynamics. *Archives of Computational Methods in Engineering*, 18(1):1–41, Mar. 2011.

[193] J. Wackers, K. A. Said, and M. Visonneau. Automatic Grid Refinement for the accurate Computation of Free-Surface Flow around Ships. In *V European Conference on Computational Fluid Dynamics*, 2010.

[194] J. Wackers and M. Visonneau. Adaptive Grid Refinement for ISIS-CFD. In *Proceedings of the 10th Numerical Towing Tank Symposium (NuTTS'08), Landéda, France*, 2008.

[195] T. Wacławczyk and T. Koronowicz. Comparison of CICSAM and HRIC High-resolution Schemes for Interface Capturing. *Journal of Theoretical and Applied Mechanics*, 46(2):325–345, 2008.

[196] R. F. Warming and R. M. Beam. Upwind Second-Order Difference Schemes and Applications in Aerodynamic Flows. *AIAA Journal*, 14(9):1241–1249, Sept. 1976.

[197] G. Warren, W. K. Anderson, J. L. Thomas, and S. Krist. Grid Convergence for Adaptive Methods. In *AIAA Computational Fluid Dynamics Conference, 10 th, Honolulu, HI*, pages 729–741, 1991.

[198] I. Weinhold and J. Parry. The third Wave of CFD. In *Proceedings of NAFEMS World Congress 2013*, 2013.

[199] D. Wilcox. *Turbulence Modeling for CFD*. Second edition. DCW Industries, Anaheim, 1998.

[200] R. Williams. A dynamic Solution-adaptive Unstructured Parallel Solver. In *Computational Fluid Dynamics' 92*, volume 1, pages 155–162, 1992.

[201] D. Youngs. *Numerical Methods for Fluid Dynamics.*, chapter Time dependent multi-material flow with large fluid distortion., pages 273–285. Academic Press, 1982.

[202] J. Zhu and W. Rodi. A low Dispersion and bounded Convection Scheme. *Computer Methods in Applied Mechanics and Engineering*, 92(1):87–96, 1991.

[203] O. Zienkiewicz. *The Finite Element Method*. McGraw-Hill (London and New York), 1977.