Development and Application of a Finite Volume Method for the Computation of Flows Around Moving Bodies on Unstructured, Overlapping Grids

Vom Promotionsausschuss der
Technischen Universität Hamburg-Harburg
zur Erlangung des akademischen Grades
Doktor-Ingenieur
genehmigte Dissertation

von

Hidajet HADŽIĆ

Bosnien und Herzegowina

2005

Development and Application of a Finite Volume Method for the Computation of Flows Around Moving Bodies on Unstructured, Overlapping Grids

Hidajet Hadžić, 1. Auflage, Hamburg: Arbeitsbereiche Schiffbau, 2006
ISBN 3-89220-633-3

Gutachter: Prof. Dr. Milovan Perić
Prof. Dr.-Ing. Gerhard Jensen
Prof. Dr.-Ing. Heinz Herwig

Tag der mündlichen Prüfung: 15.12.2005

Technische Universität Hamburg-Harburg
Schwarzenbergstrasse 95 C
21073 Hamburg

DEDICATED TO MY PARENTS

Abstract

In this thesis the development and application of an overlapping grid technique for the numerical computation of viscous incompressible flows around moving bodies is presented.

A fully-implicit second-order finite volume method is used to discretize and solve the unsteady fluid-flow equations on unstructured grids composed of cells of arbitrary shape. The computational domain is covered by a number of grids which overlap with each other and can move relative to each other in an arbitrary fashion.

For the treatment of grid movement, besides the standard method which is based on the arbitrary Lagrangian-Eulerian formulation of the governing equations, a novel method based on the solution of the governing equations in their Eulerian formulation was developed. Thus, instead the computation of grid fluxes, the grid motion is taken into account by appropriate approximation of the local time derivative in the unsteady term of the governing equations and by adding mass sources/sinks produced by moving walls in the near-wall region. The new method allows the change in grid topology and can be conveniently used with a re-meshing technique.

A special implicit procedure for coupling of the solution on overlapping grids is developed. The interpolation equations used to compute the variable values at interpolation cells distributed along grid interfaces are involved in the global system of linearized equations that arise from discretization. Such a modified linear equation system is solved for the whole domain providing that the solution is obtained on all grids simultaneously. In this way a strong inter-grid coupling characterized by smooth and unique solution in the whole overlapping region and a good convergence rate is achieved. The mass conservation, which is violated by interpolation, is enforced by adjusting the interface mass fluxes.

For a successful handling of body motion, the computational cells are allowed to be active or passive, depending on their position relative to the computational domain. The grid cells which are at the current time step outside the computational domain (e.g. covered by a body) are temporarily deactivated. These cells are reactivated when they reenter the computational domain. In this way a motion of grid components of arbitrary large scales can be achieved.

The method developed in the present study was verified by applying it to some flows for which either the numerical solution or experimental data were known or the solution could be obtained using another numerical technique available in the commercial software. The accuracy of the method was assessed through the systematical grid refinement. The potential of the proposed overlapping grid method and its advantages over other available techniques for handling moving bodies was demonstrated on a number of flows which involve complex and large-scale body motion.

Acknowledgement

The research for this thesis I performed at Fluid Dynamics and Ship Theory Section of the Technical University Hamburg-Harburg.

I wish to express my gratitude to Professor Milovan Perić and Professor Gerhard Jensen for their continuous support, guidance and encouragement throughout this study. I wish also to thank all my colleagues in the Fluid Dynamics and Ship Theory Section with whom I shared many pleasant moments and useful discussions, in particular to Muris Torlak and Dr. Eberhard Gerlach.

Financial support from the German Academic Exchange Service (DAAD), Technical University Hamburg-Harburg and ICCM GmbH, that made this study possible, is greatly appreciated.

My special thanks goes to my family for their love, support and encouragement during all these years.

Hidajet Hadžić

Contents

CHAPTER 1

Introduction

The numerical solving of the mathematical model which describes the fluid flows, known as Computational Fluid Dynamics (CFD), is nowadays increasingly becoming a design tool in various parts of industrial product development. Numerous examples include flow around a car or a ship or flow in an internal combustion engine etc. This is a field of large expansion mainly due to the progress in computer technologies and the computational algorithms.

Fluid flows involving relative motion between device components is an important class of problems which require advanced computational techniques for their simulation. There are many important applications of this kind such as flying or/and floating bodies, rotating machinery (in particular extruders, mixers, pumps, propellers), approaching/diverting objects (e.g. car overtaking), navigation devices (rudders, flapping wings), fluid-solid interaction, etc. These examples cover a wide range of flow regimes and component sizes, and they all have in common the fact that both the flow fields and geometries are complex. Understanding the unique unsteady flow features associated with moving bodies is crucial for design purposes. Therefore, accurate and efficient solution methods for such flows are required. Due to continuous development in computer hardware (increase of both, the speed and the memory) and the progress in CFD techniques, numerical simulations of such problems have become feasible. However, the computational costs are still quite considerable because the moving bodies greatly increase the complexity of the problem.

The main characteristic of industrial applications is the geometrical complexity of the solution domain. Therefore, the use of unstructured meshes for computational fluid dynamics problems has become widespread. The main reason for this is the ability of unstructured meshes to discretize arbitrarily complex geometrical domains and the ease of local and adaptive grid refinement which enhances the efficiency of the solution as well as solution accuracy. In parallel, solution algorithms for computing flows on unstructured grids have been continuously developed. Among a number of discretization methods available, the finite volume methods are most widely used for engineering CFD applications. This is mostly due to the inherent conservativeness and ease of understanding, development, and use of such methods. These methods are capable to accommodate arbitrary polyhedral grids composed of cells of different topology. Such grids have gained recently popularity because of the improved efficiency and accuracy over pure tetrahedral

grids. For a successful simulation of flows around moving bodies, the discretization in time plays a key role, both from the efficiency and accuracy point of view. Due to prohibitively low time-step limitations of explicit schemes, implicit time-integration algorithms are usually preferred.

To compute flows around moving bodies, the numerical grid needs to be adapted to the moving body and therefore move with it. Special treatment is required for the governing equations to account for grid movement. The commonly used approach is to use the definition of the governing equations in a moving frame of reference – the so called arbitrary Lagrangian-Eulerian (ALE) formulation. Methods that use moving grids are well established in commercial software. However, there are problems when bodies move relative to each other in an arbitrary manner. Large motion leads inevitably to high grid distortions and, due to pure mesh quality, the use of a single domain-fitted grid of specified topology becomes no longer possible. These problems could be overcome by introducing re-meshing and/or overlapping grids. Commercial codes do not offer such features[1].

1.1 Previous related studies

A number of numerical techniques to handle fluid flows involving moving bodies have appeared over the last two decades. The three major ones are: i) the grid deformation approach, ii) the grid re-meshing approach, iii) the overlapping grid approach.

In *the grid deformation approach*, the computational grid around a moving body is ”adjusted” at each time step such that it conforms to the new position of the body. The grid topology and the total number of control volumes are preserved. It has been used in conjunction with structured grids [21] as well as with unstructured grids [5]. The advantage of this approach is that the flow solver can be easily made fully conservative. The disadvantage of this approach is that the scale of the motion of a moving body cannot be large in comparison to the body size and usually the rotations are not allowed. The scale of the body motion can be increased and the rotation can be achieved by using so called sliding grids. In that case a part of the grid is attached to body and moves with it, while the remaining part of the grid is stationary. Between the fixed and the moving part of the grid there is a sliding interface, which is a predefined surface (usually plain, cylindrical or spherical surface). Combining the grid deformation with sliding grids a higher level of body motion can be achieved [30], but it is still limited by the grid deformation and by sliding interfaces which require a common interface (allow no overlapping) between moving and stationary grid blocks.

In *the grid re-meshing approach*, the grid near the moving body is regenerated at each time step according to the new position of the body. This approach eliminates the limitation on grid topology and thus grid quality around the moving body can be maintained. Furthermore grid motions of arbitrary scales are possible. The main drawback of this approach is that flow variables

[1]Only recently, shortly before finishing-up the thesis, one commercial CFD software provider announced the possibility of re-meshing in the future software versions.

must be interpolated from the old to the new grid at each time step and it is not easy to interpolate the flow variables in the conservative manner. Another drawback is that the grid has to be generated many times which is a time consuming and expensive operation. Usually, the re-meshing is combined with the grid deformation so that the grid is deformed while the body moves for a certain distance which is followed by a re-meshing [46]. Another possibility is to consider only a part of the grid in immediate vicinity of the body for the movement and re-meshing, while the rest of the grid remains stationary [94].

The previous two approaches fall in the category of domain-conforming grid methods, in which a single grid or several grid blocks which do not overlap with each other cover the computational domain. In *the overlapping grid approach*, also called *Chimera grid approach*, the computational domain is covered by a number of overlapping grids. Grid components associated with moving bodies move with the bodies while the other grid components remain stationary. The component grids are not required to match in any special way, but they have to overlap sufficiently to provide the means of coupling the solutions on each of them. This method allows the component grids to move relative to each other in an arbitrary fashion, making them perfect for use in applications with moving bodies. Grid adjustments or grid regeneration are thus not necessary[2]. The grid components are usually geometrically simple and allow for independent griding of higher quality than would be possible in the case of a single grid. Flow variables are interpolated between the overlapped grids to exchange the information; however, the interpolation takes place only in a limited number of cells distributed along grid interfaces rather then in the whole domain, as is the case in re-meshing approaches. The grid interfaces can be placed in regions where the variables vary more smoothly than in the vicinity of the body, thus making interpolation errors smaller. The major drawback of this approach is that it is difficult to ensure conservation of the computed variables/quantities across grid interfaces.

Obviously for applications with bodies moving in an arbitrary fashion and involving large-scale body motion, the first above-mentioned technique is not quite suitable due to limitations imposed on the level of grid deformation and the shape of sliding interfaces. Among other two techniques, the overlapping grid approach offers more flexibility and has the following two important advantages over the re-meshing technique: i) no requirement for grid re-generation, ii) possibility to achieve a better grid quality since the boundary of the overlapping grids can be arbitrarily chosen.

The overlapping grid approach has been used by a number of authors in the past. It has been applied mostly with structured grids. The first overlapping grid computations were performed by Starius who solved elliptic and hyperbolic problems [72, 73]. Steger *et al.*[75] and Benek *et al.*[8] used the overlapping structured grids for computation of flows in complex geometries. A review article about overlapping grid computations in gas dynamics presenting a number of

[2]Under assumption that the bodies do not deform. If this is not the case, the grid around the body needs to be adapted. Usually the deformation of the body is small so that the grid can be adjusted without changing its topology (grid deformation approach).

applications in complex steady geometrical configurations and with multiple moving bodies is given by Steger and Benek [74]. Further applications of the overlapping grids to problems with moving bodies are presented in references [23, 48]. Recently, the overlapping grid techniques have been used also with unstructured grids. A few publications appeared which present the overlapping grid methods for unstructured tetrahedral grids [55, 44]. A more detailed review of publications related to overlapping grid techniques will be given later in chapter 5.

The main objective of the present research was to develop a method for computation of flows around moving bodies using the overlapping grid technique and arbitrary unstructured meshes. The unstructured meshes are preferred due to reasons mentioned above. The limitations of structured grids concerning the grid topology and controlling of the grid resolution can thus be overcome. Although, in general, a set of structured overlapping grids can be used to cover the domains of a high level of complexity, the complexity of such overlapping grid systems increases significantly due to the increasing number of component grids required. Combination of unstructured and overlapping grids provides an optimum in the treatment of problems with moving bodies. The highest flexibility can be achieved in both treatment of complex geometry and grid motion, while the complexity of the overlapping grid systems can be kept at low level since the number of overlapping grid components required is much smaller in comparison to structured grids.

When the ALE approach is used for treatment of grid movement, in most instances the effort has to be made to ensure that the grid movement itself does not influence the flow field[3]. To ensure this the so called space conservation law (SCL) has to be satisfied [20]. On arbitrary polyhedral unstructured grids the enforcement of the space conservation law may become difficult due to complex grid structure. Furthermore, the application of re-meshing is also difficult due to data transfer between two grids of different topology.

Another concern of the present study was to investigate the possibility of the treatment of the grid movement in an alternative way which would be easier for implementation on arbitrary unstructured grids and would facilitate the implementation of the re-meshing technique. The overall method would thus have no restrictions regarding the complexity of solution domains and body motion. For the sake of simplicity but without loss of generality, development has been performed in two spatial dimensions. The extension to three-dimensional flows is straightforward and the necessary steps are explained were appropriate.

1.2 Present contributions

The major contributions of the present work can be summarized as follows:

- A new discretization and solution practice that uses a fully-implicit method and approximation of local time derivative in the Eulerian formulation of the governing equations,

[3] It has to be ensured that no artificial flow appears due to grid motion. A convenient test case is a uniform stream flow which has to remain unaffected by the grid motion [69].

rather than the space-conservation law and the grid velocity in the arbitrary Lagrangian-Eulerian (ALE) formulation, for the treatment of the grid movement has been developed. It was shown that the treatment of the grid movement in such a way produces results of comparable accuracy as the conventional ALE approach, while offering some advantages over the ALE method, especially on arbitrary unstructured grids. Namely, the variable values at specified points (centers of control volumes), required for approximation of the local time derivative, are much easier to compute and can be computed independently of the grid topology. On the other hand, the computation of grid velocities (enforcement of the space conservation low) for unstructured grids composed of cells with arbitrary number of cell-faces may be very complicated. Another very important feature of this new technique is that it allows the change of the grid topology during the computation, making the method suitable for application with re-meshing technique.

- The re-meshing technique as a special case of the moving grid has been incorporated in the spirit of the new discretization technique. Interpolation of the dependent variables at the control-volume centers from the old to the new grid was shown to be enough for continuation of the computation after re-meshing. The accuracy was also found to be not significantly affected by the data transfer.

- An overlapping grid technique based on a finite volume discretization on arbitrary unstructured grids has been developed. A special implicit technique for the treatment of grid interfaces, based on the simultaneous solution of interpolation equations in the global linear equation system, was developed. It is shown that such a treatment provides a strong inter-grid coupling and smooth and unique solution over the whole computational domain and achieves a convergence rate in the range of comparable single grids. The method is applicable to stationary as well as moving overlapping grids and allows a nearly arbitrary large scale motion between the grid components providing an efficient tool for handling of moving bodies.

- Due to Neumann boundary conditions imposed on the pressure-correction equation when the mass flow rate over the boundaries is given, the strict mass conservation is essential for the solution of this equation. Since the interpolation technique used for the solution coupling between overlapping grids is not strictly conservative, additional treatment is necessary to fulfill this requirement. Two possible techniques to enforce the mass conservation were examined here, and the strategy based on the global correction of interface fluxes was adopted. It was shown that this global correction also provides the mass conservation on each grid as the converged solution is approached, as expected due to flow field continuity.

- The present method is embodied in a computer program for two-dimensional problems in a general way appropriate for the use of grids of arbitrary topology. The performance and accuracy of the method was first extensively assessed on a number of flows for which the reference solutions exist. The flexibility of the overlapping grid technique was clearly demonstrated on a number of flows which involve moving bodies with large scales of body motion.

1.3 Outline of the thesis

The thesis is subdivided into seven chapters. Chapter 2 summarizes the conservation equations governing the incompressible fluid flow. The equations are given in their integral form suitable for the discretization method used for their solution. A generic transport equation for an arbitrary scalar variable is introduced as a model equation upon which the discretization technique is derived.

The subsequent three chapters consider different aspects of the numerical methodology concerning the discretization of the governing equations, treatment of the grid movement and overlapping grids technique. Chapter 3 presents the second-order finite volume discretization method applicable to unstructured meshes of arbitrary topology. The discretization of the generic transport equation for a scalar variable is described term by term. This is followed by the derivation of a pressure-correction equation, which is used for resolving the velocity-pressure coupling and calculation of pressure. Finally, a segregated solution approach for the coupled system of equations along with steps required to obtain the solution is outlined.

In chapter 4 two methods for the grid movement implemented in the present study are described. First the method based on the Lagrangian-Eulerian formulation of the governing equations and the use of the space conservation law is outlined. Then the novel method based on the approximation of the local time derivative at the fixed location in space is described. The accuracy and performance of both methods is assessed on a piston-driven flow in a pipe contraction. In addition, the new method for grid movement is tested in combination with re-meshing technique.

Chapter 5 presents the overlapping grid technique developed in the present study. First, an outline of the overlapping grid technique is given, followed by description of the algorithms for hole cutting and donor searching adopted in this study. Next the methodology for the solution of governing equations on overlapping grids is explained, with special emphases on inter-grid coupling. An implicit method is developed, which provides strong inter-grid coupling and smooth and unique solution across overlapping interfaces. The necessity of the mass conservation is also discussed and two possible approaches to enforce the mass conservation on overlapping grids are described.

In chapter 6 the present method is assessed on a number of examples for which either benchmark numerical solutions or experimental data are available or the solution could be obtained with another technique available in the commercial software. Special attention is paid to the assessment of the overall accuracy, inter-grid coupling and conservation errors caused by interpolation of the solution between overlapping grids. In addition predictions are given of some complex flows involving multiple moving bodies for which no experimental or numerical data are available. These computations are shown in order to demonstrate the flexibility and the range of applicability of the overlapping grid method in computation of flows around moving bodies.

Finally, conclusions drown from the studies performed in this thesis and suggestions for future work are given in chapter 7.

A CD-ROM with a number of animations presenting the results of computation of flows around moving bodies is attached.

CHAPTER 2

Governing Equations

This chapter contains an overview of the equations governing the incompressible Newtonian fluid flow. The conservation equations for mass, momentum and energy are used in their integral form as the mathematical basis for the numerical method adopted in this study. The conservation equations are given in the coordinate-free form and the momentum equation is resolved in terms of Cartesian vector components. A generic conservation equation for an arbitrary scalar variable is introduced as a model equation upon which the discretization procedure is derived. Finally, the boundary and initial conditions necessary for completion of the mathematical model are discussed.

2.1 Introduction

The derivation of basic equations of fluid dynamics is based on the fact that the dynamical behavior of a fluid is determined by the conservation laws of mass, momentum and energy. The conservation of a certain flow quantity means that its variation inside an arbitrary control volume can be expressed as the net effect of the amount of the quantity being transported across the boundary by convection and diffusion and any sources or sinks within the control volume. The fluid is regarded as continuum, which assumes that the matter is continuously distributed in space. The concept of continuum enables us to define velocity, pressure, temperature, density and other important quantities as continuous functions of space and time. In addition, special mathematical tools (such as field theory) for the analysis of the problems of continuum mechanics can be used.

In the derivation of the governing equations of fluid dynamics the Eulerian or *control volume approach* is conveniently used, rather then Lagrangian or *material* approach. The formal derivation of equations follows from the *Reynolds' transport theorem*[1] [92] which facilitates the implementation of laws and principles concerning the behavior of a system (made of the same fluid particles) in an arbitrary control volume.

[1] Also called the *generalized transport theorem* [70].

2.2 Conservation equations

In this section the conservation equations of mass, momentum and energy for an arbitrary control volume V fixed in space[2] bounded by a closed surface S, (see figure 2.1), will be presented.

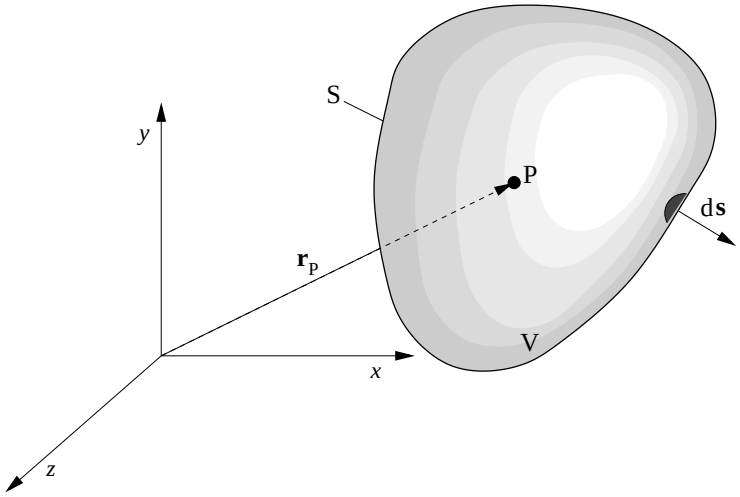


Figure 2.1: Arbitrary control volume V.

2.2.1 Conservation of mass

The conservation equation for mass or the continuity equation, for a control volume states that the rate of change of the mass inside the control volume V is equal to the difference between inflow and outflow mass fluxes across the volume surface S. In integral form the continuity equation reads:

$$\frac{\partial}{\partial t}\int_V \rho \, \mathrm{d}V + \oint_S \rho \mathbf{v} \cdot \mathrm{d}\mathbf{s} = 0 \,, \tag{2.1}$$

where, ρ is the fluid density and $\mathbf{v}$ is the fluid velocity. Note that if the fluid is regarded as incompressible, which is the case in this study, the first term on the left-hand side in equation (2.1) is zero since the density is constant. However, for problems with moving grids, which involve changes of the control volume, this term might be considered depending on the numerical method employed (see chapter 4).

[2]The equations for a moving control volume in conjunction with different solution methods are considered in chapter 4.

2.2.2 Conservation of momentum

The conservation equation for momentum states that the total variation of momentum, represented by the time variation of momentum within the control volume and the transfer of momentum across the boundary of the control volume by fluid motion (called convection or advection), is caused by the net force acting on the fluid in the control volume. In the integral coordinate-free form this equation reads:

$$\frac{\partial}{\partial t}\int\limits_V \rho\mathbf{v}\,\mathrm{d}V + \oint\limits_S \rho\,\mathbf{v}\mathbf{v}\cdot\mathrm{d}\mathbf{s} = \oint\limits_S \boldsymbol{\sigma}\cdot\mathrm{d}\mathbf{s} + \int\limits_V \rho\,\mathbf{f}_\mathrm{b}\,\mathrm{d}V\,, \tag{2.2}$$

where $\boldsymbol{\sigma}$ is the stress tensor representing the surface forces and $\mathbf{f}_\mathrm{b}$ represents the vector of body forces acting on the fluid. The stress tensor can be expressed in terms of basic dependent variables and for Newtonian incompressible fluids is defined as:

$$\boldsymbol{\sigma} = \mu\,\left[\mathrm{grad}\,\mathbf{v} + (\mathrm{grad}\,\mathbf{v})^T\right] - p\,\mathbf{I}\,, \tag{2.3}$$

where p is the pressure, μ is the dynamic viscosity of the fluid and $\mathbf{I}$ is the unit tensor.

In order to be solved, the vector equation (2.2) has to be resolved into specific directions resulting in three equations in terms of vector components. In this study the Cartesian coordinate system is used, resulting in the simplest form of momentum equations and insuring strong conservation of momentum components [18, 59]. Substituting equation (2.3) into equation (2.2) and taking dot product with the Cartesian base vector $\mathbf{i}_i$, the corresponding equation for the ith Cartesian component is obtained:

$$\begin{aligned}\frac{\partial}{\partial t}\int\limits_V \rho u_i\,\mathrm{d}V + \oint\limits_S \rho\,u_i\mathbf{v}\cdot\mathrm{d}\mathbf{s} \;&=\; \oint\limits_S \mu\,\mathrm{grad}\,u_i\cdot\mathrm{d}\mathbf{s} + \oint\limits_S \mu\,\left[\mathbf{i}_i\cdot(\mathrm{grad}\,\mathbf{v})^T\right]\cdot\mathrm{d}\mathbf{s}\\ &-\; \oint\limits_S p\,\mathbf{i}_i\cdot\mathrm{d}\mathbf{s} + \int\limits_V \rho\,\mathrm{f}_{\mathrm{b}_i}\,\mathrm{d}V\,,\end{aligned} \tag{2.4}$$

where u_i stands for the ith velocity component and $\mathrm{f}_{\mathrm{b}_i}$ stands for ith component of the body force.

2.2.3 Conservation of energy

The underlying principle upon which the energy equation is derived, is the first law of thermodynamics. It states that any changes in time of the total energy inside control volume are caused by the rate of work of forces acting on the volume and by the net heat flux into it. This equation in its most general form contains a large number of influences whose importance depends on the problem considered. For most engineering flows the energy equation can be written in term of specific enthalpy as:

$$\frac{\partial}{\partial t}\int\limits_V \rho h\,\mathrm{d}V + \oint\limits_S \rho h\,\mathbf{v}\cdot\mathrm{d}\mathbf{s} = \oint\limits_S \mathbf{q}\cdot\mathrm{d}\mathbf{s} + \int\limits_V \left(\boldsymbol{\sigma}:\mathrm{grad}\,\mathbf{v} + q_T\right)\,\mathrm{d}V\,, \tag{2.5}$$

where h is the specific enthalpy, $\mathbf{q}$ is the heat flux vector and q_T is the heat source or sink. If the fluid is considered to be thermally perfect, specific enthalpy depends only on temperature via the following relation:

$$h = c_{\mathrm{p}} T \,, \tag{2.6}$$

where, c_{p} is the specific heat at constant pressure and T is the temperature. The heat flux vector is related to the temperature gradient by the Fourier's law:

$$\mathbf{q} = -k \,\mathrm{grad}\, T \,, \tag{2.7}$$

where the proportionality coefficient k is the thermal conductivity of the fluid.

Introducing equations (2.6) and (2.7) into equation (2.5), the energy equation can be re-written in term of temperature as:

$$\frac{\partial}{\partial t} \int\limits_V \rho c_{\mathrm{p}} T \,\mathrm{d}V + \oint\limits_S \rho c_{\mathrm{p}} T \,\mathbf{v} \cdot \mathrm{d}\mathbf{s} = \oint\limits_S \mathbf{q} \cdot \mathrm{d}\mathbf{s} + \int\limits_V \left(\boldsymbol{\sigma} : \mathrm{grad}\, \mathbf{v} + q_T \right) \mathrm{d}V \,. \tag{2.8}$$

2.3 Generic transport equation

Conservation equations presented above describe the laminar flow of an incompressible Newtonian fluid. In some situations additional processes take place and besides the basic equations some additional transport equations have to be solved. An example is the Reynolds-averaged modeling of turbulent flows, where a number of additional transport equations for turbulent quantities have to be solved[3]. The turbulent quantities may even have vector or tensor character but can be expressed via a certain number of non-zero components. In addition, equations describing transport of chemical species may be introduced. In general, such equations for the transport of a generic scalar variable ϕ can be written in the following form:

$$\frac{\partial}{\partial t} \int\limits_V \rho\phi \,\mathrm{d}V + \oint\limits_S \rho\phi \,\mathbf{v} \cdot \mathrm{d}\mathbf{s} = \oint\limits_S \Gamma_\phi \mathrm{grad}\, \phi \cdot \mathrm{d}\mathbf{s} + \oint\limits_S \mathbf{q}_{\phi S} \cdot \mathrm{d}\mathbf{s} + \int\limits_V q_{\phi V} \,, \tag{2.9}$$

where ϕ stands for the transported variable, Γ_ϕ is the diffusion coefficient and $Q_{\phi S}$ and $Q_{\phi V}$ stand for the surface exchange terms and volume sources, respectively. It is important to note that the momentum and energy equations can also be written in the form of equation (2.9). This fact significantly facilitates the numerical procedure since, from the numerical point of view, only one equation has to be discretized[4]. Equation (2.9) is therefore used as the generic equation for deriving the numerical procedure described in the next chapter.

[3] Depending on the model used, additional terms appear in the Navier-Stokes equations.

[4] For a segregated solution algorithm, which is also adopted in this study.

2.4 Boundary and initial conditions

In order to obtain the solution of the governing equations, the initial and boundary conditions have to be specified. Boundary conditions in fluid flow problems are conveniently divided according to the physical meaning of boundaries like wall, inflow, outflow, symmetry etc. Whatever boundaries appear in a specified problem, the initial and boundary conditions have to be selected so that the problem to be solved is *well posed*, i.e. the solution exists and depends continuously upon them.

Due to the parabolic nature of the unsteady equations, one set of initial conditions has to be specified. An initial distribution of all dependent variables at the initial instant of time, $t = t_0$, have to be prescribed in the whole computational domain V:

$$\phi\left(\mathbf{r}, t_0\right) = \phi^0\left(\mathbf{r}\right), \quad \mathbf{r} \in V \ . \tag{2.10}$$

The governing equations have an elliptic character in space. Therefore, boundary conditions have to be specified at all times and at all domain boundaries $S_{\mathtt{B}}$. Different type of boundaries require different boundary conditions to be applied. All of them can be classified into two groups:

- **Dirichlet** boundary condition, when the value of a dependent variable at the portion of the boundary $S_{\mathtt{B}}^{\mathtt{D}}$ of the solution domain is specified, e.g.

$$\phi\left(\mathbf{r}_B, t\right) = f_{\mathtt{j}}(t), \qquad \mathbf{r}_B \in S_{\mathtt{B}}^{\mathtt{D}} \, . \tag{2.11}$$

- **Neumann** boundary condition, when the gradient of a dependent variable at the portion of the boundary of the solution domain $S_{\mathtt{B}}^{\mathtt{N}}$ is specified, e.g.

$$\mathrm{grad}\, \phi\left(\mathbf{r}_B, t\right) = \mathbf{f}_{\mathtt{j}}(t), \qquad \mathbf{r}_B \in S_{\mathtt{B}}^{\mathtt{N}} \ . \tag{2.12}$$

The implementation of boundary conditions within the numerical method adopted in this study is described in detail in the next chapter.

CHAPTER 3

Numerical method

In this chapter the *finite volume method* (FVM) adopted in the present study for solving the conservation equations presented in the previous chapter is described. It is based on a second-order accurate spatial discretization which accommodates unstructured meshes with cells of arbitrary shape. In addition a fully implicit integration in time is used for the solution of unsteady problems. Computational points are located in the cell center and a collocated variable arrangement is used. A segregated solution procedure is employed to solve the resulting set of non-linear algebraic equations. It leads to a decoupled system of linear algebraic equations for each dependent variable. The linearized equation systems are solved using a conjugate gradient solver. The SIMPLE algorithm, leading to an equation for the pressure correction, is used to establish the pressure-velocity coupling and calculate the pressure. Finally the implementation of various boundary conditions is discussed and solution algorithm is outlined.

3.1 Finite volume method

There exists a vast number of methods for the numerical solution of the governing equations. Most of them follow closely the path consisting of the following three steps:

- *Space discretization*, consisting of defining a numerical grid, which replaces the continuous space with a finite number of discrete elements with computational points at their centroids. At those points the solution of dependent variables are computed. This process is termed grid generation.

- *Time discretization*, which assumes the division of the entire time interval into a finite number of small subintervals, called time steps.

- *Equation discretization*, which is the replacement of the individual terms in the governing equations by algebraic expressions connecting the variable values at computational points in the grid.

In the present study *the finite volume method* of discretization is adopted. It utilizes directly the conservation laws, i.e. the governing equations are discretized starting from their integral

form. The solution domain is discretized by an unstructured mesh composed of a finite number of contiguous control volumes (CVs) or cells. Each control volume is bounded by a number of cell faces which compose the CV-surface and the computational points are placed at the center of each control volume. There is no restriction in the shape that the control volumes may have, i.e. an arbitrary polyhedral shape is allowed (see figure 3.1). This is achieved by a special data structure based on cell faces, which provides the data connectivity between the cells sharing the same cell face. It makes the flow solver capable to deal with meshes consisting of cells of different topology (e.g. hybrid meshes) since the number of cell faces per CV can vary arbitrarily from one CV to another. The same data structure supports also the local (cell-wise) grid refinement since only the list of cell faces needs to be updated. The interface between refined and non-refined cells can, thus, be treated fully conservatively. All this provides a great flexibility regarding the numerical mesh that can be used.

The collocated (non-staggered) variable arrangement, which is suitable for the arbitrary unstructured meshes, is used. It assumes that all dependent variables share the same control volume. Equations are solved in a Cartesian coordinate system, providing the strong conservation form of the momentum equations [18] and making the method insensitive to the grid non-smoothness [59]. The system of conservation equations is treated in the segregated way, meaning that they are solved one at a time, with the inter-equation coupling treated in the explicit manner. Before solving, each equation is linearized and the non-linear terms are lagged. An iteration procedure is employed to handle the non-linearity and inter-equation coupling.

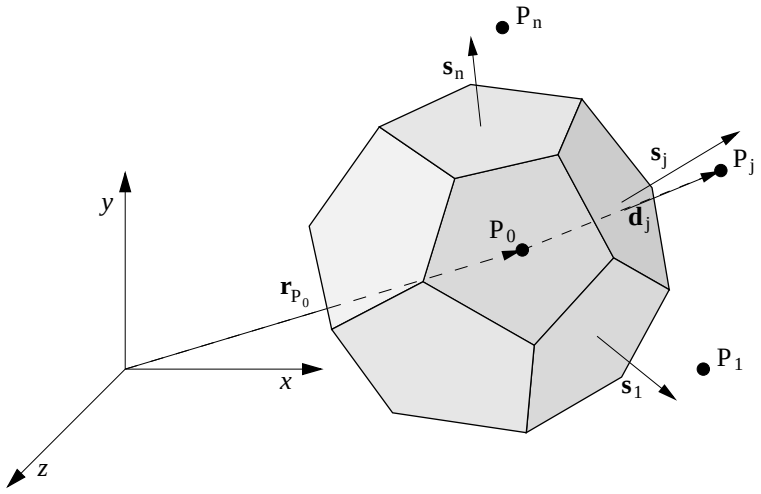


Figure 3.1: A general control volume of arbitrary shape.

3.2 Discretization procedure

The discretization procedure will be demonstrated for a generic transport equation (2.9). The same procedure is followed to obtain the discrete counterparts of all governing equations. Only

the discretization of the continuity equation is described separately. In the present method, the continuity equation is utilized to compute the pressure. The derivation of a pressure-correction equation, used to obtain the pressure, and its discretization is described in section 3.3.

When equation (2.9) is integrated over a CV shown in figure 3.1, it attains the following form:

$$\underbrace{\frac{\partial}{\partial t}\int_{\Delta V}\rho\phi\,\mathrm{d}V}_{\text{Rate of change}}+\underbrace{\sum_{j=1}^{N}\int_{S_j}\rho\phi\,\mathbf{v}\cdot\mathrm{d}\mathbf{s}}_{\text{Convection}}=\underbrace{\sum_{j=1}^{N}\int_{S_j}\Gamma_\phi\mathbf{grad}\,\phi\cdot\mathrm{d}\mathbf{s}}_{\text{Diffusion}}+\underbrace{\sum_{j=1}^{N}\int_{S_j}\mathbf{q}_{\phi S}\cdot\mathrm{d}\mathbf{s}}_{\text{Surface source}}+\underbrace{\int_{\Delta V}q_{\phi V}\,\mathrm{d}V}_{\text{Volume source}}\quad .(3.1)$$

The surface integrals in equation (2.9) are here represented as a sum of integrals over a number of cell faces defining the cell P_0. Equation (3.1) has four distinct parts: rate of change (transient or unsteady term), convection, diffusion and sources. The discretization of each particular term in equation (3.1) is described in the following subsections.

3.2.1 Approximation of volume integrals

The volume integrals in equation (3.1) are approximated by the mid-point rule, which is second-order accurate for any CV shape. The mean value of the integrated variable is approximated by the value of the function at the CV center P_0,

$$\Psi_{P_0}=\int_{\Delta V}\psi\,\mathrm{d}V=\overline{\psi}\,\Delta V\approx\psi_{P_0}\,\Delta V_{P_0}\,. \tag{3.2}$$

Since the variable value at the CV center, needed for the calculation of the integral, is readily available (variables are stored at CV centers), no additional approximations are required.

3.2.2 Approximation of surface integrals

The surface integrals are also approximated by mid-point rule, thus

$$F_j=\int_{S_j}\boldsymbol{f}\cdot\mathrm{d}\mathbf{s}\approx\boldsymbol{f}_j\cdot\mathbf{s}_j\,, \tag{3.3}$$

where F_j represents the flux of the transported variable across the cell face j and $\boldsymbol{f}$ is the flux vector, see equation (3.1). Unlike in volume integrals, the variable values at cell-face center required for the calculation of surface integrals are not directly available. They have to be evaluated using additional approximations. In order to retain second-order accuracy of mid-point rule approximation for integrals, the cell-face value has to be evaluated with at least a second-order accuracy.

3.2.3 Convection

Using the mid-point rule, the convective flux through the cell face j is approximated as

$$F_j^{\mathrm{c}} = \int_{S_j} \rho\phi\mathbf{v}\cdot\mathrm{d}\mathbf{s} \approx \phi_j \dot{m}_j \,, \tag{3.4}$$

where ϕ_j is the value of the variable ϕ at the cell-face center and $\dot{m}_j$ is the mass flux through the cell face j. The later is assumed known, it is computed using values from the previous iteration according to the simple Picard-linearization procedure [26]. The value of ϕ at the cell-face center has to be computed by interpolation from cell-center values. The methods used in this study are described below.

Central differencing scheme (CDS)

There are many possibilities to interpolate variables at cell-face center. One of the simplest is the linear interpolation, which provides the second-order accuracy. Using linear interpolation the value at point j' (figure 3.2), which lies at the line connecting two cell centers, is obtained as:

$$\phi_{j'} = \phi_{P_j}\lambda_j + \phi_{P_0}\left(1-\lambda_j\right) \,, \tag{3.5}$$

where λ_j is the interpolation factor calculated as

$$\lambda_j = \frac{(\mathbf{r}_j - \mathbf{r}_{P_0})\cdot\mathbf{d}_j}{\mathbf{d}_j\cdot\mathbf{d}_j} \,. \tag{3.6}$$

Here $\mathbf{d}_j = \mathbf{r}_{P_j} - \mathbf{r}_{P_0}$ is the vector connecting the point P_0 with its neighboring point P_j (see figure 3.2). This approximation is second-order accurate at the location j'. If $\phi_{j'}$ is used instead of ϕ_j in equation (3.4) a first order error term is introduced which depends on the distance between j and j'. If point j' lies close to the point j, the first order error term is small and the accuracy of the integral approximation will not be affected significantly. On the other hand, if j' is far from cell-face center j, the accuracy of the integral approximation may be impaired. In such a case, the second-order accuracy can be recovered by applying a correction as follows (see figure 3.2):

$$\phi_j = \phi'_j + (\mathrm{grad}\,\phi)_{j'}(\mathbf{r}_j - \mathbf{r}_{j'}). \tag{3.7}$$

The gradient at j' is obtained by interpolating the gradients from the two CV centers according to equation (3.5). The gradients at CV center have to be calculated anyway for the evaluation of diffusive fluxes through CV faces and also in some cases to compute source terms.

Linear interpolation described here is usually referred to as *central-differencing scheme* (CDS), since it leads to the same results as the use of central differences for the first derivatives in finite-difference methods.

Although the CDS is very simple and has the desired accuracy (second-order accurate) it may, in certain circumstances (e.g. high value of the local Peclet number or in computations of turbulent flows [28]), produce non-physical oscillatory solutions. Therefore, a method that provides necessary stability is required.

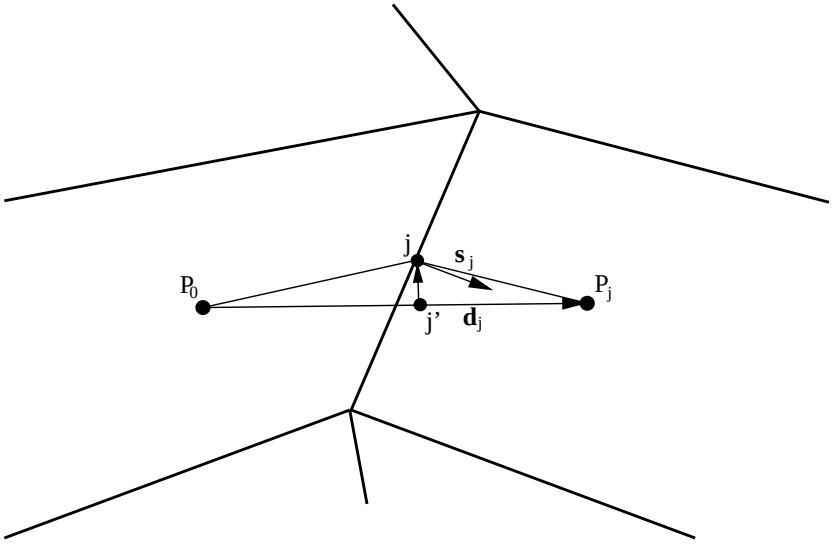


Figure 3.2: Linear interpolation at cell-face center.

Upwind differencing scheme (UDS)

A widely used scheme, which guarantees bounded (non-oscillatory) solutions is the *upwind differencing scheme* (UDS). It assumes zero order polynomial approximation between the two neighboring cells. The value of ϕ at cell-face center is approximated by the value at CV center on the upwind side of the face, i.e.

$$\phi_j = \begin{cases} \phi_{P_0} & \text{if} \quad \dot{m}_j \geq 0 \ , \\ \phi_{P_j} & \text{if} \quad \dot{m}_j < 0 \ . \end{cases} \tag{3.8}$$

UDS is bounded and unconditionally stable, i.e it will never produce oscillatory solutions. However, these properties have been achieved by sacrifying accuracy. Since it is only first-order accurate, UDS was found to be *numerically diffusive*, requiring a very fine grid resolution to achieve acceptable accuracy.

Blending of two schemes

To recover the lost accuracy and at the same time to maintain the boundedness, the second-order CDS approximation ϕ_j^{cds} can be blended with some amount of the first-order UDS approximation ϕ_j^{uds}. The value at the cell face is then calculated as:

$$\phi_j = \phi_j^{\text{uds}} + \gamma \left(\phi_j^{\text{cds}} - \phi_j^{\text{uds}} \right)^{\text{old}} \ , \tag{3.9}$$

where γ is the blending factor with a value between zero and unity. The amount of CDS scheme can be controlled by choosing an appropriate value of the blending factor. In some cases it may be necessary to use some lower values of γ in order to suppress oscillations near discontinuities or peaks in profiles when the grid is too coarse. On the other hand, more upwind, i.e. lower value

of blending factor may impair the accuracy [56]. It is, therefore, recommended to use the values of the blending factor as close to unity as possible. One can seek an optimum between stability and accuracy.

Equation (3.9) is implemented using the so called *deferred correction* approach [59]. Only the first term on the right hand side in equation (3.9), which comes from UDS, contributes to the coefficient matrix, ensuring that the contributions to the coefficients are unconditionally positive (see section 3.2.8). The second term which presents the difference between CDS and UDS approximation is treated explicitly. This term is calculated using the values from the previous iteration. Many other possibilities to compute ϕ_j exist; see reference [26] for further details.

3.2.4 Diffusion

Using the mid-point rule, the diffusive flux F_j^D through the cell face j can be calculated as:

$$F_j^D = \int_{S_j} \Gamma_\phi \mathrm{grad}\,\phi \cdot \mathrm{d}\mathbf{s} \approx \Gamma_{\phi j}\ (\mathrm{grad}\,\phi)_j^* \cdot \mathbf{s}_j = \Gamma_{\phi j} \left(\frac{\partial \phi}{\partial n}\right)_j S_j, \tag{3.10}$$

where Γ_ϕ is the diffusion coefficient and S_j is the area of cell face. In order to calculate the diffusion flux, an approximation of the derivative at the CV face in the direction normal to the cell face is required. This quantity could be computed by interpolation of cell-center gradients to the cell-face center. Cell-center gradients are calculated explicitly using one of the approximations described in section 3.2.5. Since the simple interpolation may lead to oscillations (see [26] for details), another approximation suggested by Muzaferija [54], which prevents oscillations and retains second-order accuracy, is used:

$$\left(\frac{\partial \phi}{\partial n}\right)_j \approx \frac{\phi_{P_j} - \phi_{P_0}}{|\mathbf{d}_j|} - (\mathrm{grad}\,\phi)_j^{\mathrm{old}} \cdot \left(\frac{\mathbf{d}_j}{|\mathbf{d}_j|} - \frac{\mathbf{s}_j}{|\mathbf{s}_j|}\right), \tag{3.11}$$

where the value of $(\mathrm{grad}\,\phi)_j^{\mathrm{old}}$ is calculated by interpolation using equation (3.5). The first term on the right-hand side represents a central-difference approximation of the derivative in the direction ξ of a straight line connecting nodes P_0 and P_j (see figure 3.2). This term is treated implicitly, i.e. it contributes to the matrix coefficients. The second term which corrects the error due to the fact that we need the derivative in the direction of cell-face normal n (figure 3.2) is calculated using previous values of the variables and treated explicitly, i.e. added to the source term.

3.2.5 Calculation of gradients at CV center

The gradient at CV-center can be easily calculated using Gauß' theorem and mid-point rule approximation as follows:

$$\int_V \mathrm{grad}\,\phi\,\mathrm{d}V = \oint_S \phi\,\mathrm{d}\mathbf{s} \quad \Rightarrow \quad (\mathrm{grad}\,\phi)_{P_0} \approx \frac{\sum_j \phi_j\,\mathbf{s}_j}{\Delta V_{P_0}}. \tag{3.12}$$

This approximation is second-order accurate and is applicable to cells of arbitrary shape, which makes it useful in conjunction with unstructured grids.

Another possibility to calculate gradients at the CV center with second-order accuracy is based on linear shape functions. Assumption of a linear variation of the dependent variable ϕ in the neighborhood of point P_0 (including points P_j) leads to the following set of equations:

$$\mathbf{d}_j \cdot (\text{grad}\,\phi)_{P_0} = \phi_{P_j} - \phi_{P_0} \quad (j = 1, ..., N_j), \tag{3.13}$$

where $\mathbf{d}_j = \mathbf{r}_{P_j} - \mathbf{r}_{P_0}$ is the vector connecting point P_0 with its neighbor P_j (see figure 3.2). The unknown gradient vector is obtained by solving this system of equations which is over-determined since the number of neighboring points P_j involved in equation (3.13)[1] is always grater then the number of the unknown vector components[2]. The system (3.13) is solved by means of a least-squares method and the unknown gradient vector is computed as:

$$(\text{grad}\,\phi)_{P_0} = D^{-1} \sum_j \mathbf{d}_j^T (\phi_{P_j} - \phi_{P_0})\,, \tag{3.14}$$

where matrix D is calculated as

$$D = \sum_j \mathbf{d}_j^T \mathbf{d}_j\,. \tag{3.15}$$

Note that the matrix D is symmetric[3] and its coefficients depend solely on the grid geometrical properties and hence a single evaluation of D^{-1} is necessary for a given grid.

In this study both methods have been incorporated and tested. It was found that, on irregular grids with strong non orthogonality (like e.g. triangular grids), the second method based on shape functions produces more accurate results. If the method based on Gauß' theorem is used on such grids, cell-face values in equation (3.12) should be calculated using equation (3.7). This, however, requires iterations since the values of gradients needed for interpolation are still not available and have to be computed. Numerical tests have shown that a few iterations (typically 3-5) are enough to recover the accuracy. But the least-squares method still remains more accurate and, therefore, may be recommended to be used for calculation of gradients. One, however, needs to be careful, since the least square method does not perform well in computations with complex turbulence models (like Reynolds' stress models). In such cases it may even cause serious convergence problems [29]. Our experience has also shown that the least-square method may lead to divergence on grids with a high aspect ratio, which evolve e.g. in the near-wall region in computations with low-Re turbulence models. This may be put in touch with explanation in [53] that on strongly distorted grids matrix D in equation (3.15) can become singular and thus lead to divergence of the solution process.

[1] The number of points is equal to the number of cell faces enclosing the cell P_0. For boundary faces, points which lie at the boundary are included in equation (3.13).

[2] Minimal number of cell faces surrounding a CV is 3 in 2-D and 4 in 3-D.

[3] D is a 3×3 matrix in 3-D and 2×2 in 2-D problems. Due to symmetry only 6 coefficients in 3-D versus 3 in 2-D have to be stored.

3.2.6 Source terms

The volume source term is obtained by integrating the specific source $q_{\phi V}$ of variable ϕ over the control volume ΔV_{P_0}. Making use of equation (3.2), the volume source is approximated as:

$$Q_{\phi V} = \int_{V_P} q_{\phi V} \, \mathrm{d}V \approx \left(q_{\phi V}\right)_{P_0} \Delta V_{P_0} \, . \tag{3.16}$$

Since the value of the function $q_{\phi V}$ at point P_0 is known (either given or computed), no additional approximations are necessary.

Similarly, the surface source is obtained by integrating the specific source $\mathbf{q}_{\phi S}$ over the control volume surface S_{P_0}:

$$Q_{\phi S} = \oint_S \mathbf{q}_{\phi S} \cdot \mathrm{d}\mathbf{s} = \sum_{j=1}^{N_j} \int_{S_j} \mathbf{q}_{\phi S} \cdot \mathrm{d}\mathbf{s} \approx \sum_{j=1}^{N_j} \mathbf{q}_{\phi S_j} \cdot \mathbf{s}_j \, . \tag{3.17}$$

The value of function $\mathbf{q}_{\phi S}$ at cell-face center j is computed by interpolation using equation (3.7).

Treatment of pressure term in momentum equation

The term involving pressure in the momentum equation can be treated in two ways. The first way is to calculate the contribution from the pressure by calculating and summing the pressure forces over cell faces. Another possibility is to transform the surface integral into a volume integral and calculate the contribution of the pressure as a volume source. Making use of Gauß' theorem, the following volume source which involves the pressure gradient is obtained:

$$\oint_S p\mathbf{I} \cdot \mathrm{d}\mathbf{s} = \int_V \mathrm{grad}\, p \, \mathrm{d}V \, . \tag{3.18}$$

Although the first approach ensures the fully conservative treatment of the pressure term, the second approach is adopted in this study. It is also conservative if the pressure gradient at the CV center is calculated using equation (3.12) which is equivalent to the equation (3.18).

3.2.7 Integration in time

For unsteady flows the integration of equation (3.1) needs also to be performed in time. Before doing this, we rearrange the equation (3.1) into the following form:

$$\frac{\partial \Psi}{\partial t} = f\left(t, \phi\right) \, , \quad \text{where} \quad \Psi = \int_V \rho\phi \, \mathrm{d}V \approx \rho\phi_{P_0} \Delta V_{P_0} \, , \quad \text{and} \quad \phi = \phi\left(\mathbf{r}, t\right) \, . \tag{3.19}$$

Time-stepping schemes may be divided into explicit and implicit methods. Explicit methods are easy to implement (no linear equation systems have to be solved) and alow for an arbitrary order

of temporal accuracy. However, due to stability reasons, explicit schemes usually require very small time step (much smaller than it would be required for the sake of accuracy) making them inefficient for many practical computations. On the other hand, implicit schemes are formally unconditionally stable and allow arbitrary time step, which is governed only by the accuracy considerations. In present study two implicit methods were considered: first-order Euler method and second-order three-time-levels (TTL) scheme.

The Euler scheme is obtained by integrating the equation (3.19) over the time interval Δt placed between previous time step t_{n-1} and current time step t_n, leading to the following approximation:

$$\frac{1}{\Delta t}\int\limits_{t_{n-1}}^{t_n}\left(\frac{\partial \Psi}{\partial t}\right)\mathrm{d}t \approx \frac{\Psi_{P_0}^{n}-\Psi_{P_0}^{n-1}}{\Delta t} = f^n\,. \tag{3.20}$$

For the sake of convenience the equation (3.19) is divided by Δt before the integrating. The right-hand side of equation (3.19) is evaluated in terms of the unknown variable values at the current time step, providing the first-order accuracy. Euler scheme is unconditionally stable and very simple. However, due to numerical diffusion, this scheme may lead to strong damping of unsteady flow features (see [41]).

In order to improve the accuracy, in some situations a higher-order scheme is necessary (see [41]). In this work the second-order three-time-levels implicit scheme is used in such situations. This scheme involves variable values from three time levels and performs the integration of equation (3.19) over a time interval Δt centered around the current time level t_n, i.e. from $t_n - \Delta t/2$ to $t_n + \Delta t/2$. The right-hand side of equation (3.19) is evaluated at time level t_n, which is a centered approximation with respect to time. Multiplying it by Δt is a second-order mid-point rule approximation of the time-integral. A second-order approximation of the time derivative at time level t_n is achieved by fitting a parabola through the solution at three time levels, thus

$$\frac{1}{\Delta t}\int\limits_{t_{n-\frac{1}{2}}}^{t_{n+\frac{1}{2}}}\left(\frac{\partial \Psi}{\partial t}\right)dt \approx \frac{3\Psi_{P_0}^{n}-4\Psi_{P_0}^{n-1}+\Psi_{P_0}^{n-2}}{2\Delta t} = f^n\,. \tag{3.21}$$

Note that the three-time-levels scheme is marginally more complex than the Euler scheme (only solutions of one more time level have to be stored) but is second-order accurate.

Both schemes presented above are the so called fully-implicit schemes, since the fluxes and the source terms on the right hand side of the equation (3.19) are approximated at the current time level.

An important parameter in unsteady simulations is the so called Courant-number defined as:

$$\mathrm{Co} = \frac{u\Delta t}{\Delta x}, \tag{3.22}$$

where u is the local velocity in the flow direction, Δt is the time step and δx is the local grid spacing in the flow direction. This number describes how far a fluid particle travels during one time step relative to mesh size. Courant-number is usually used as a parameter to assess the accuracy and in many cases it is required to be of the order of unity or smaller.

3.2.8 Final form of algebraic equations

By summing the approximations of all terms in equation (3.1), an algebraic equation for each CV is obtained which relates the value of dependent variable ϕ at the CV center to the values at neighboring CVs. This equation can be written in the following form:

$$a_{P_0\phi}\phi_{P_0} - \sum_{j=1}^{N_j} a_j\phi_{P_j} = b_{P_0}\,, \tag{3.23}$$

where the index j runs over the range of neighbor nodes involved in the implicit approximation of integrals[4], and b_{P_0} contains source terms, contributions from the transient term, parts of convection and diffusion fluxes which are treated explicitly in a deferred correction manner and contributions from boundary cell faces. The values of the coefficients $a_{P_0\phi}$ and a_j and source term b_{P_0} follow directly from the approximations involved for each term of equation (3.1)[5]:

$$\begin{aligned}
a_j &= \Gamma_\phi \frac{|\mathbf{s}_j|}{|\mathbf{d}_j|} - \min\left(\dot{m}_j, 0\right)\,, \\
a_{P_0\phi} &= a^t_{P_0} + \sum_{j=1}^{N_j} a_j\,, \\
b_{P_0} &= Q_{\phi V} + Q_{\phi S} + b^t_{P_0} \\
&- \gamma\left\{\dot{m}_j\phi_j^{\mathrm{CDS}} - \left[\max\left(\dot{m}_j, 0\right)\phi_{P_0} - \min\left(\dot{m}_j, 0\right)\phi_{P_j}\right]\right\} \\
&+ \Gamma_\phi\left(\mathrm{grad}\,\phi\right)_j \cdot \left(\mathbf{s}_j - \frac{|\mathbf{s}_j|}{|\mathbf{d}_j|}\mathbf{d}_j\right) \\
&+ \sum_B a_B\phi_B\,,
\end{aligned} \tag{3.24}$$

where $a^t_{P_0}$ and $b^t_{P_0}$ stand for the contributions from the transient term which depend on the scheme used for time integration (see section 3.2.7) and their values are given in table 3.1. Coefficient a_B multiplies the variable value ϕ_B at the boundary and index B runs over the range of boundary cell faces surrounding the cell P_0 if it lies next to the boundary. Coefficients and source term in equation (3.24) are evaluated using the values of dependent variables from the previous iteration which is in accordance with the Picard iteration scheme.

[4] For the adopted approximation it is equal to the number of inner cell faces surrounding the cell P_0.

[5] UDS contribution is extracted from equation (3.8) which for this purpose was rearranged into: $\dot{m}_j\phi_j^{\mathrm{uds}} = \max(\dot{m}_j, 0)\phi_{P_0} + \min(\dot{m}_j, 0)\phi_{P_j}$

Table 3.1: Meaning of the coefficients $a^t_{P_0}$ and $b^t_{P_0}$ in equation (3.24).

Coefficient	Euler implicit scheme	Three time levels scheme
$a^t_{P_0}$	$\frac{\rho\Delta V_{P_0}}{\Delta t}$	$\frac{3\rho\Delta V_{P_0}}{2\Delta t}$
$b^t_{P_0}$	$\frac{\rho\Delta V_{P_0}\phi^{n-1}_{P_0}}{\Delta t}$	$\frac{4\rho\Delta V_{P_0}\phi^{n-1}_{P_0} - \rho\Delta V_{P_0}\phi^{n-2}_{P_0}}{2\Delta t}$

3.3 Calculation of pressure

The discretization procedure considered up to now is applicable to the momentum and other transport equations for scalar quantities which can be represented in the form of the generic transport equation (2.9). Thus, by solving a given equation the field of the corresponding dependent variable is obtained. The problem that remains to be solved is the calculation of pressure, which via the source term in the momentum equation (2.2) influences the velocity field. The main difficulty is that there is no equation which contains the pressure as a dominant variable and as such could be solved to obtain the pressure field. For compressible flows the density may be considered as a dependent variable which can be computed from the continuity equation, while the pressure can be obtained from an equation of state. This approach is, however, not applicable to incompressible flows since in that case the density remains constant and the continuity equation can no longer be regarded as a dynamic equation. On the other hand, the pressure does not feature in the continuity equation which therefore cannot be directly used as an equation for pressure. The continuity equation is actually only an additional constraint on the velocity field which can be satisfied only by adjusting the pressure field. However, it is not obvious how this adjustment of pressure needs to be performed. A possible way around this problem is to solve the momentum and continuity equations simultaneously in a coupled manner. An explicit derivation of an equation for pressure can thus be avoided. However, even this approach requires special treatment since the continuity equation delivers zero elements on the diagonal of the coefficient matrix (since it does not contain the pressure). The problem can be solved by a special preconditioning, see Weiss *et al.*[91]. The approach requires four time more memory then the alternative used here and does not bring advantages when unsteady flows are computed. In the present study a sequential solution procedure, which has been successfully applied to a broad range of problems involving incompressible flow, is favored. The strategy adopted here to resolve the pressure-velocity coupling is based on the SIMPLE algorithm [58].

3.3.1 SIMPLE algorithm

This method is based on an iterative procedure in which the linearized momentum equations are solved first, using a current pressure estimate (usually from the previous non-linear iteration or previous time step). The obtained velocity field is then used to discretize the continuity equation. Since the current velocity field does not satisfy the continuity equation, a mass imbalance will be produced. This imbalance is then used to calculate the pressure-correction field such that the corrected velocities satisfy the continuity equation. An appropriate link between the velocity and pressure corrections is derived from the discretized momentum equation. After the pressure-correction is calculated, the velocities and pressure are corrected. The new estimated velocities do not satisfy momentum equation, so that the iterative procedure is continued until both the momentum and the continuity equations are satisfied. In what follows the procedure for the calculation of pressure on collocated grids is described.

Cell-face velocity

For the calculation of mass fluxes in the discretized continuity equation, the velocities at cell-face centers have to be calculated. When a collocated variable arrangement is used, a simple linear interpolation of velocity can lead to oscillations in the pressure field as explained in [58] and [59]. In order to avoid oscillations, a special interpolation practice which implies that the cell-face velocity depends not only on the velocity field but also on the pressure field was proposed by Rhie and Chaw [62] (see also [59]). This influence of pressure field on the velocity field can be expressed in the following manner [19, 26]:

$$\mathbf{v}_j^\star = \mathbf{v}_j - \overline{\left(\frac{\Delta V_{P_0}}{a_{P_0}^v}\right)}_j \left[\frac{p_{P_j} - p_{P_0}}{|\mathbf{d}_j|} - \frac{\overline{\operatorname{grad} p} \cdot \mathbf{d}_j}{|\mathbf{d}_j|}\right] \frac{|\mathbf{d}_j|\mathbf{s}_j}{\mathbf{d}_j \cdot \mathbf{s}_j}\,, \tag{3.25}$$

where $\mathbf{v}_j$ is the spatially interpolated velocity defined by equation (3.5) and overbar denotes the arithmetic average of the values at nodes P_0 and P_j. The term in square brackets acts as a correction of the interpolated velocity. This term is small for a smooth pressure distribution and tends to zero as the grid is refined; it is proportional to the third derivative of pressure in the direction of vector $\mathbf{d}_j$ and the square of distance between P_0 and P_j [26]. The role of this term is to smooth out oscillations if they develop during iterations.

Predictor stage

When the cell-face velocities are computed, the mass fluxes can be calculated as $\dot{m}_j^* = \rho \mathbf{v}_j^* \cdot \mathbf{s}_j$ and the discretized continuity equation, which is in general not satisfied, leads then to:

$$\sum_{j=1}^{N_j} \dot{m}_j^\star = \Delta \dot{m}\,, \tag{3.26}$$

where $\Delta \dot{m}$ is the mass imbalance which should be zero. The velocities have to be corrected so that mass conservation is satisfied in each CV. By employing the SIMPLE algorithm [26], the

correction of the mass flux through the cell face j can be expressed in term of the gradient of the pressure correction[6]; see equation (3.25):

$$\dot{m}'_j = \rho \mathbf{v}'_j \cdot \mathbf{s}_j = -\rho \left|\mathbf{s}_j\right| \overline{\left(\frac{\Delta V_{P_0}}{a^v_{P_0}}\right)}_j \left(\frac{\partial p'}{\partial n}\right)_j \approx -\rho \left|\mathbf{s}_j\right| \overline{\left(\frac{\Delta V_{P_0}}{a^v_{P_0}}\right)}_j \frac{p'_{P_j} - p'_{P_0}}{\mathbf{d}_j \cdot \mathbf{s}_j} \left|\mathbf{s}_j\right| . \tag{3.27}$$

The requirement that the corrected mass fluxes satisfy the continuity equation can be expressed as:

$$\sum_{j=1}^{N_j} \dot{m}'_j + \Delta \dot{m} = 0. \tag{3.28}$$

Substituting equation (3.27) into (3.28), the following pressure-correction equation is obtained:

$$\sum_{j=1}^{N_j} \left[-\rho \frac{\mathbf{s}_j \cdot \mathbf{s}_j}{\mathbf{d}_j \cdot \mathbf{s}_j} \overline{\left(\frac{\Delta V_{P_0}}{a^v_{P_0}}\right)}_j \left(p'_{P_j} - p'_{P_0}\right) \right] = -\Delta \dot{m} \,, \tag{3.29}$$

which can be rewritten in the following form:

$$a_{p'_P} p'_{P_0} - \sum_{j=1}^{N_j} a_{p'_j} p'_{P_j} = b_{p'}, \tag{3.30}$$

with coefficients:

$$a_{p'_j} = \rho \frac{\mathbf{s}_j \cdot \mathbf{s}_j}{\mathbf{d}_j \cdot \mathbf{s}_j} \overline{\left(\frac{\Delta V_{P_0}}{a^v_{P_0}}\right)}_j \quad , \quad a_{p'_P} = \sum_{j=1}^{N_j} a_{p'_j} \quad , \quad b_{p'} = -\Delta \dot{m} = -\sum_{j=1}^{N_j} \dot{m}^\star_j \,. \tag{3.31}$$

Corrector stage

After equation (3.30) is solved, the pressure-correction so obtained is used to correct pressure and velocity fields and mass fluxes. According to equations (3.28), (3.29) and (3.31), the mass fluxes can be corrected via:

$$\dot{m}_j = \dot{m}^\star_j + a_{p'_j} \left(p'_{P_j} - p'_{P_0}\right) \,. \tag{3.32}$$

Corrected mass fluxes now satisfy the continuity equation and are used to calculate convective fluxes in the next iteration. Velocities at CV center are corrected using the following equation, which follows from equation (3.27) when written for the CV center:

$$\mathbf{v}_{P_0} = \mathbf{v}^{\mathrm{pred}}_{P_0} + \mathbf{v}'_{P_0} = \mathbf{v}^{\mathrm{pred}}_{P_0} - \frac{\Delta V_{P_0}}{a^v_{P_0}} \left(\mathrm{grad}\, p'\right)_{P_0} \,, \tag{3.33}$$

[6] More details about the derivation of pressure correction equation according to the SIMPLE algorithm can be found in references [26] and [58].

where superscript pred means predicted value. Finally, the pressure is updated by adding the pressure-correction to the pressure:

$$p_{P_0} = p_{P_0}^{\mathrm{pred}} + \alpha_p p'_{P_0} \,, \tag{3.34}$$

where α_p is an under-relaxation factor, which is necessary because the calculated pressure-correction is usually overestimated due to simplifications made in deriving equation (3.27) – neglecting the effect of velocity corrections at neighbor nodes and non-alignment of vectors $\mathbf{d}_j$ and $\mathbf{n}_j$. Typical values of under-relaxation factor are $\alpha_p = 0.1 - 0.3$ for steady problems. In unsteady problems higher values for α_p (up to 1) can be chosen.

3.4 Implementation of boundary conditions

For the closure of the algebraic equation systems that arise from the discretization, the boundary conditions have to be implemented. The boundary values are specified at the boundary cell faces which are related to only one cell (see figure 3.3). Therefore, a special attention has to be paid to implementation of boundary conditions, which depend on their type. In what follows, the boundary conditions used in this study and their implementation into the discretization procedure are described.

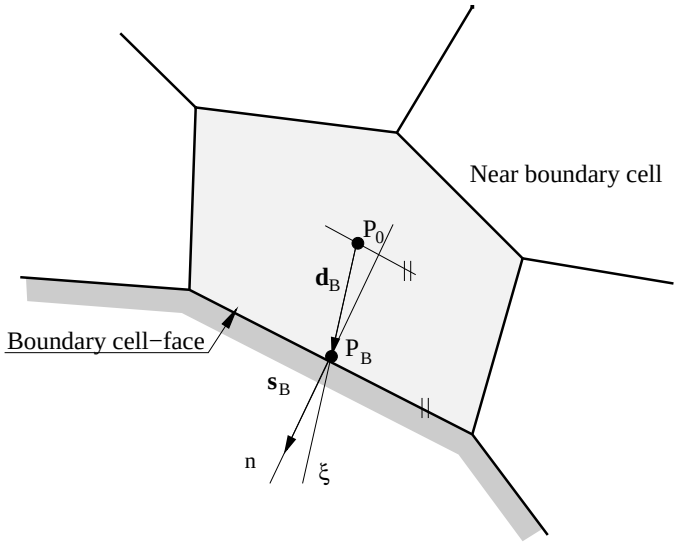


Figure 3.3: Near-boundary cell with definitions and notation.

3.4.1 Inlet boundaries

At inlet boundaries the values of all variables are usually known. This implies the Dirichlet boundary conditions and simplifies implementation. All convective fluxes can be calculated using given boundary values and diffusion fluxes can be approximated using given boundary values and one-sided finite differences in approximations of diffusive fluxes.

3.4.2 Outlet boundaries

Unlike at inlet, at outlet boundaries we do not know exact values of the dependent variables but need to approximate them. Therefore, outlet boundary should be placed as far downstream from the region of interest as possible. Furthermore, it should be placed at a location where the flow is everywhere directed outwards in order to avoid propagation of any error introduced by estimations of the outlet conditions. The variable values at the outlet boundary may be extrapolated from the flow domain. The simplest approximation is that of zero gradient extrapolation which, for simple backward approximation, reduces to $\phi_{P_b} = \phi_{P_0}$. Velocity components require a special treatment since it has to be ensured that the overall mass conservation is satisfied. The velocity components are first estimated by extrapolation from interior. These values are used to calculate outlet mass fluxes. The velocity components and mass fluxes are then corrected by multiplying them with the ratio $\dot{m}_\mathrm{I}/\dot{m}_\mathrm{O}$, where $\dot{m}_\mathrm{I}$ is the total mass inflow and $\dot{m}_\mathrm{O}$ is the total mass outflow. This ensures that the global mass conservation is fulfilled in each outer iteration, which is important for the solution of the pressure-correction equation [11] (see also section 5.3.3).

3.4.3 Symmetry boundaries

If the flow is to be symmetrical with respect to a line or a plane, the first condition which has to be fulfilled is that there is no flow across the boundary, i.e. the normal velocity at a symmetry boundary must be zero (and so are all convective fluxes). Using this condition, the velocity vector at the boundary can be obtained from the known velocity vector at the CV-center next to the boundary (see figure 3.4)

$$\mathbf{v}_B = \mathbf{v}_{P_0} - (\mathbf{v}_{P_0} \cdot \mathbf{n})\,\mathbf{n}\,. \tag{3.35}$$

This velocity can now be used to calculate diffusion fluxes in the usual way, but this approach can lead to poor convergence. Symmetry boundary conditions can be implemented in the mo-

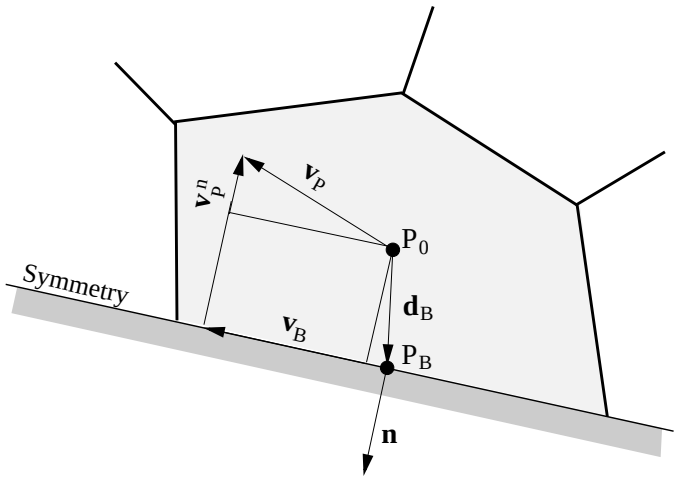


Figure 3.4: Implementation of symmetry boundary conditions.

mentum equations using the fact that only the gradient of the normal velocity component in the direction normal to the boundary is non-zero, i.e. that only the normal stress $\tau_{nn} = 2\mu\frac{\partial v_n}{\partial n}$ makes, contribution to the force coming from the viscous part of the stress tensor [26]:

$$\mathbf{f}_{\text{sym}} = \int_{S_B} \tau_{nn}\mathbf{n}\,\mathrm{d}S = \int_{S_B} 2\mu\left(\frac{\partial v_n}{\partial n}\right)\mathbf{n}\,\mathrm{d}S \approx \left[2\mu\left(\frac{\partial v_n}{\partial n}\right)S\mathbf{n}\right]_B \tag{3.36}$$

where $v_n = \mathbf{v}_{P_0}\cdot\mathbf{n}$ is the normal velocity component and the normal derivative is approximated using a one-side difference (see figure 3.4):

$$\frac{\partial v_n}{\partial n} \approx \frac{v_n}{\delta n} = \frac{\mathbf{v}_{P_0}\cdot\mathbf{n}}{\mathbf{d}_b\cdot\mathbf{n}}\,. \tag{3.37}$$

Contributions of the force calculated by equation (3.36) are distributed to corresponding momentum equations for each velocity component.

For all scalar quantities, normal derivatives at the symmetry boundary must be zero – a boundary condition of Neumann type is applied (zero diffusion fluxes).

3.4.4 Wall boundaries

Solid walls are considered as impermeable and a no-slip boundary condition is applied, i.e. the fluid velocity is equal to the wall velocity. Hence, the Dirichlet boundary condition is directly applicable for the momentum equation. Implementation is similar as for the inlet, except that, due to the wall impermeability, the convective fluxes are zero. For scalar quantities (e.g. temperature) either the Dirichlet or the Neumann boundary condition can be applied, depending on whether the variable value or its gradient is prescribed at the boundary. In the former case diffusion fluxes are usually approximated using one-sided differences and in the latter case, the fluxes can be computed directly (usually they are already given) and inserted into the conservation equation for the near-wall CV.

3.4.5 Boundary conditions for the pressure-correction equation

Since the pressure-correction equation is different from the other basic equations, the implementation of boundary conditions for this equation deserves some attention. When the mass flux through a boundary is prescribed, which is the case for all types of boundary conditions mentioned above, its correction is zero at the boundary. It implies, according to equation (3.27), that the normal derivative of pressure-correction is zero at the boundary, i.e. the Neumann boundary condition is applied.

Another possibility is that the pressure at the boundary is given. In that case the value of the pressure-correction at the boundary is zero, implying the Dirichlet boundary condition for pressure-correction. If the pressure is prescribed at the boundary, velocity cannot be specified there but has to be extrapolated from the interior. This is done in the same way as for inner cell

faces (equation (3.25)), except that here extrapolation is used instead of interpolation. Extrapolated boundary velocities have to be corrected to satisfy the continuity equation.

The pressure at the boundary which is needed for the calculation of forces exerted on the boundary surfaces and for the calculation of gradients at the CV-center next to the boundary is obtained by extrapolation from interior. In this study the following linear extrapolation is used:

$$p_b = p_{P_0} + (\text{grad}\, p)_{P_0} \cdot \mathbf{d}_b \,, \tag{3.38}$$

where $(\text{grad}\, p)_{P_0}$, which depends on the value p_b itself, can be obtained using the value of p_b from previous iteration.

3.5 Solution procedure

As a result of the discretization, algebraic equations of the form (3.23) are obtained for each cell and each variable. These equations are non-linear and coupled, i.e. the coefficients in equation (3.23) depend on the dependent variables and more than one dependent variable appears in each equation. In order to solve this coupled equation system, an iterative procedure based on the *segregated solution algorithm* is employed.

3.5.1 Segregated algorithm

The basic feature of the segregated solution algorithm is that the coupled system of equations is solved in sequence for one dependent variable at a time. Equations (3.23) are linearized by calculating their coefficients using the values of dependent variables from previous iteration. This results in a linear equation sub-system for each dependent variable, which can be written in the following matrix form:

$$\mathrm{A}_\phi \boldsymbol{\phi} = \mathbf{b}_\phi. \tag{3.39}$$

Here, A_ϕ is the coefficient matrix, consisting of $N \times N$ coefficients, with N being the number of cells, $\boldsymbol{\phi}$ is a vector containing the values of ϕ and $\mathbf{b}_\phi$ is the vector containing the source terms. The matrix A_ϕ has two very important features:

a) It is sparse, since each row contains only $N_j + 1$ non-zero coefficients, where N_j is the number of neighboring points P_j included in equation (3.23).

b) It is diagonally dominant, i.e. it satisfies the condition:

$$|a_{P_0\phi}| \geq \sum_j |a_{j\phi}|, \tag{3.40}$$

at all cells.

These properties make the system (3.39) suitable to be solved using an iterative linear equation solver which retains the matrix sparsity. This results in a significant saving in memory storage, since only the non-zero coefficients have to be stored. The linear equation system (3.39) is solved in turn for each dependent variable. Iterative solution need not to be taken to complete convergence, because the coefficients are only tentative. It is usually sufficient to iterate until the absolute sum of residuals reduces by one order of magnitude. Iterations performed while solving the linear equation system are called *inner iterations*.

After the linear equation system (3.39) is solved for all variables, the coefficients are updated and the entire procedure is repeated. The iterative procedure (*outer iterations*) is continued until the convergence criterion is satisfied. As convergence criterion one usually uses the sum of absolute residuals r_ϕ^k normalized by an appropriate normalizing factor. In the present study the following criterion is used as the test for convergence of outer iterations:

$$R_\phi^k = \frac{\sum_{i=1}^{N} |r_\phi^k|}{M_\phi} = \frac{\sum_{i=1}^{N} |b_{P_0} - a_{P_0\phi}\phi_{P_0} + \sum_{j=1}^{N_j} a_j \phi_{P_j}|}{\sum_{j=1}^{N_j} |a_{P_0\phi}\phi_{P_0}|} \leq \varepsilon, \tag{3.41}$$

where residual r_ϕ^k at node P_0 and iteration k is calculated as the difference between the right and the left hand side of equation (3.23) and ε is a fraction of R_ϕ^0. Usually it is enough to iterate until the residuals fall for three orders of magnitude, i.e until $R_\phi^k < 0.001\, R_\phi^0$. Convergence criterion is, however, problem-dependent and, in some cases, it may be required to reduce the residuals for more then three orders of magnitude to achieve a converged solution. One can also use additional convergence criteria like e.g. the values of some integral parameters, which should stop changing at the three or more most significant digits.

3.5.2 Under-relaxation

For an iterative method to converge, when solving a linear equation system (inner iterations), the sufficient condition is that the coefficient matrix is diagonal dominant [59]. This condition, however, may not be sufficient to guaranty the convergence of an iterative solution of a coupled equation system (outer iterations). Therefore, in order to promote convergence it is usually necessary to slow down the changes of dependent variables through the iterations. This is achieved via under-relaxation, which allows dependent variable to change only a fraction α_ϕ of the difference which normally would be achieved by the solution of the original equation system (3.39), i.e.:

$$\phi_{P_0}^m = \phi_{P_0}^{m-1} + \alpha_\phi \left(\phi_{P_0}^{\text{new}} - \phi_{P_0}^{m-1} \right), \tag{3.42}$$

where superscript m denotes outer iteration counter, α_ϕ is an under-relaxation factor which has the value between zero and unity and $\phi_{P_0}^{\text{new}}$ is the solution obtained from the equation (3.23) as

$$\phi_{P_0}^{\text{new}} = \frac{b_{P_0} - \sum_{j=1}^{N_j} a_j \phi_{P_j}^m}{a_{P_0}} \,. \tag{3.43}$$

By substituting (3.43) into equation (3.42), equation (3.23) is rearranged into its under-relaxed form:

$$\underbrace{\frac{a_{P_0}}{\alpha_\phi}}_{a_{P_0}^*} \phi_{P_0}^m + \sum_{j=1}^{N_j} a_j \phi_{P_j}^m = \underbrace{b_{P_0} + \frac{1-\alpha_\phi}{\alpha_\phi}\, a_{P_0} \phi_{P_0}^{m-1}}_{b_{P_0}^*} \,, \tag{3.44}$$

with $a_{P_0}^*$ and $b_{P_0}^*$ being the modified central coefficient and source term in equation (3.39).

When the converged solution is obtained, the terms involving α_ϕ cancel out and the obtained solution is also the solution of the original equation system.

3.5.3 Solution of linear equation systems

The methods for the solution of the linear equation systems can be divided into two groups: direct and iterative methods. Direct methods are seldom used to solve the linear equation systems resulting in CFD since iterations are inevitable due to non-linearity and inter-equation coupling. On the other hand, the iterative methods can exploit some features of the system matrix like its sparsity and diagonal dominance, which for most of iterative solvers promote the convergence. Therefore, only iterative methods are considered in this study.

There exist a number of efficient linear equation solvers suitable for the solution of sparse linear equation systems. Some of them are, however, specially designed for the solution of systems with regular diagonal structure of the coefficient matrix, which arises from discretization on structured grids. The coefficient matrix arising from discretization on unstructured grids has on the contrary an irregular structure requiring a linear equation solver capable to deal with such matrices. The solvers of conjugate gradient type can be adapted to irregular unstructured grids with varying number of neighbors. In the present study the *Bi-Conjugate Gradient Stabilized* (BiCGSTAB) solver of Van der Vorst [87] with incomplete Cholesky preconditioning was used to solve the resulting linear equation systems. More details about iterative methods for the solution of linear equation systems and algorithms for their implementation can be found in references [4] and [50].

3.5.4 Solution algorithm

The sequence of steps required to solve the set of equations governing fluid flow can be summarized as follows:

1. Initialize all field values by an initial guess;
2. Solve the linearized momentum equations to predict the velocity;
3. Solve the pressure-correction equation to obtain p';
4. Correct the velocities using equation (3.33), the mass fluxes using equation (3.32) and the pressure using equation (3.34);
5. Solve energy equation to obtain the temperature;
6. Solve equations for additional scalars (e.g. turbulent kinetic energy and dissipation, species, etc.);
7. If convergence criterion is not satisfied return to step 2.

In the case of unsteady problems, the time step size has to be selected. In principle it can vary, but in this study constant time steps have been used. When the flow is periodic, one usually selects the time step as a fraction of the period. Otherwise, a characteristic time scale has to be defined.

CHAPTER 4

Grid Movement

The conservation equations governing fluid flow presented in chapter 2, were given for a fixed CV and the numerical method, described in chapter 3, assumed that stationary grids were used. The subject of this chapter is the computation of fluid flow on moving grids. Two techniques for treatment of grid movement used in the present study are presented: a) arbitrary Lagrangian-Eulerian approach, which uses a moving reference frame formulation of the governing equations and b) a novel method, which solves the same equations as for stationary grids and uses interpolation to obtain the variable values from the previous time steps on the new grid locations. Modifications of the discretization method necessary to take into account the grid movement are discussed in this chapter. The performance and accuracy of both methods have been assessed on the flow in a pipe contraction driven by a moving piston. Finally, the new method for grid movement was tested in conjunction with re-meshing technique.

4.1 Introduction

In many practical situations the fluid interacts with rigid or flexible bodies. This interaction results in changes of the fluid-flow domain caused by the body motion. This can arise as a result of fluid forces exerted on the body, like for example motion of floating bodies or deformation of a membrane, or can be forced externally, for example the flow in cylinder driven by a piston. In such cases it is necessary to solve the flow equations on a moving grid. Our attention is especially focused on the flows around moving bodies, which fall into this class of problems. The solution of such a problem requires special treatment which takes the grid movement into account. In this study two techniques for the computation of incompressible flows on moving grids are incorporated into the numerical method adopted in this study. In what follows these two techniques and steps necessary to adapt them to the solution method are described in detail.

4.2 Arbitrary Lagrangian-Eulerian approach

The most common way to treat problems with moving grids is to rewrite the governing equations in a moving frame of reference, usually referred to as *Arbitrary Lagrangian-Eulerian (ALE)* for-

mulation [39, 40, 43]. This approach is used in all commercial CFD software. The conservation equations for mass, momentum and scalar quantities discussed in chapter 2, for a control volume V bounded by a moving surface S in their integral form read:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int\limits_V \rho\,\mathrm{d}V + \oint\limits_S \rho\,(\mathbf{v}-\mathbf{v}_s)\cdot\mathrm{d}\mathbf{s} = 0\,, \tag{4.1}$$

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int\limits_V \rho u_i\,\mathrm{d}V + \oint\limits_S \rho\,u_i\,(\mathbf{v}-\mathbf{v}_s)\cdot\mathrm{d}\mathbf{s} &= \oint\limits_S \mu\,\mathrm{grad}\,u_i\cdot\mathrm{d}\mathbf{s} + \oint\limits_S \mu\,\left[(\mathrm{grad}\,\mathbf{v})^T\cdot\mathbf{i}_i\right]\cdot\mathrm{d}\mathbf{s}\\ &- \oint\limits_S p\,\mathbf{i}_i\cdot\mathrm{d}\mathbf{s} + \int\limits_V \rho\,\mathrm{f}_{\mathrm{b}_i}\,\mathrm{d}V\,,\end{aligned} \tag{4.2}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int\limits_V \rho\phi\,\mathrm{d}V + \oint\limits_S \rho\phi\,(\mathbf{v}-\mathbf{v}_s)\cdot\mathrm{d}\mathbf{s} = \oint\limits_S \Gamma_\phi \mathrm{grad}\,\phi\cdot\mathrm{d}\mathbf{s} + \oint\limits_S Q_{\phi S}\cdot\mathrm{d}\mathbf{s} + \int\limits_V Q_{\phi V}\,\mathrm{d}V\,, \tag{4.3}$$

where $\mathbf{v}_s$ is the velocity of the CV-surface[1]. Observe that in the case when the CV does not move, i.e. the surface velocity $\mathbf{v}_s$ is zero, the Eulerian form of the equations, dealt with in chapter 2, is recovered. If, however, the surface velocity $\mathbf{v}_s$ is equal to the fluid velocity, there is no flux through the CV surface, i.e. the same fluid remains in the CV. Thus the control volume becomes the control mass and the Lagrangian description of fluid motion is obtained.

Conservation equations related to a moving control volume differ from their counterparts for a space-fixed CV in the definition of the unsteady term and the convective fluxes. In the case of a fixed control volume, the time derivative (denoted as $\partial/\partial t$) represents the local rate of change at a fixed point in space and the convective fluxes are computed using fluid velocity $\mathbf{v}$. On the other hand, for a moving CV the time derivative (denoted as $\mathrm{d}/\mathrm{d}t$) expresses the rate of change in a volume whose location changes and convective fluxes are computed using the relative velocity $\mathbf{v}-\mathbf{v}_s$.

4.2.1 Space conservation law

Having the surface velocities known, the conservation equations can be solved in the usual way by simply calculating the convective fluxes using the relative velocities at the cell-faces and modifying the unsteady term in order to take into account changes of the CV volume. However, the use of the numerically approximated surface velocity in the evaluation of fluxes based on known face positions at different times does not guarantee the conservation of mass and other conserved quantities. Artificial mass sources or sinks can be produced which can cause accumulation of solution errors or even divergence of the method. For example, in a uniform free stream spurious flows can be induced by grid motion. Influences of the grid motion onto the flow field can be avoided if the grid velocities are calculated so that the following equation is satisfied:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int\limits_V \mathrm{d}V - \oint\limits_S \mathbf{v}_s\cdot\mathrm{d}\mathbf{s} = 0\,. \tag{4.4}$$

[1] Also referred to as the grid velocity.

Equation (4.4) relates the time variation of the control volume to the velocity of its surface and hence is called the *space (geometric) conservation law (SCL)*. The cell surface velocity can be obtained by solving equation (4.4) [80]. This is however not necessary. Instead of solving the SCL equation, grid velocities can be calculated from the known grid positions in such a manner so that the space conservation law is always satisfied. This approach, proposed by Demirdžić and Perić [20], is adopted in this study.

Using an implicit Euler scheme for time discretization, the SCL equation (4.4) can be discretized in the following way:

$$\frac{\Delta V_{P_0}^n - \Delta V_{P_0}^{n-1}}{\Delta t} = \sum_{j=1}^{N_j} \int_{S_j} \mathbf{v}_s \cdot \mathrm{d}\mathbf{s} = \sum_{j=1}^{N_j} \frac{\delta V_j^n}{\Delta t} \,, \tag{4.5}$$

where $\Delta V_{P_0}^n$ and $\Delta V_{P_0}^{n-1}$ are the volumes of the cell P_0 at times t_n and t_{n-1} and δV_j^n represents the volume swept by CV face j when moving from the old to the new position (indicated by shaded area in figure 4.1). The above definition of volume fluxes due to mesh motion (right-hand side of equation (4.5)) follows from the fact that the change of CV volume is equal to the sum of volumes δV_j swept by each CV face during the movement between two successive positions, i.e.,

$$\Delta V_{P_0}^n - \Delta V_{P_0}^{n-1} = \sum_{j=1}^{N_j} \delta V_j^n \,. \tag{4.6}$$

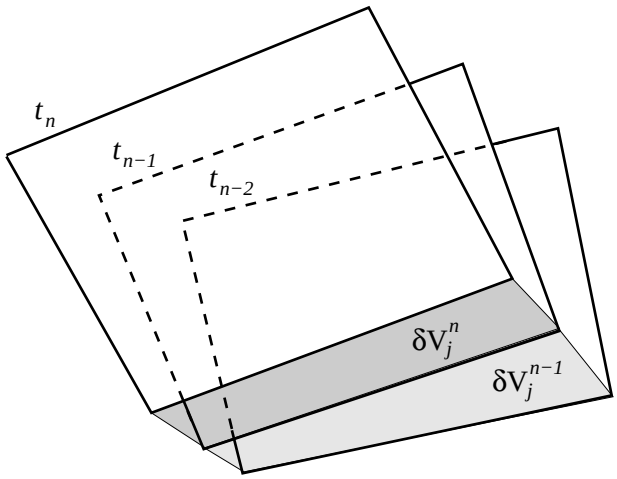


Figure 4.1: A control volume at three time steps and the volume swept by CV face j between each two consecutive time steps.

Accordingly, the following form of the discretized SCL equation is obtained when the implicit three-time-levels scheme is used:

$$\frac{3\Delta V_{P_0}^n - 4\Delta V_{P_0}^{n-1} + \Delta V_{P_0}^{n-2}}{2\Delta t} = \sum_{j=1}^{N_j} \int_{S_j} \mathbf{v}_s \cdot \mathrm{d}\mathbf{s} = \sum_{j=1}^{N_j} \frac{3\delta V_j^n - \delta V_j^{n-1}}{2\Delta t} \,. \tag{4.7}$$

The expressions on the right hand side of equation (4.5) and (4.7) represent the volume fluxes through CV faces due to their movement. Since they are used for the evaluation of convective fluxes in the conservation equations, there is no need to calculate the grid velocity explicitly.

4.2.2 Discretization and boundary conditions

The discretization of the momentum equation and conservation equations for scalar quantities is principally the same as for a stationary grid. The unsteady term is discretized in a more general way, which takes into account the changes of CV volumes which arise if the grid deforms. For the implicit Euler scheme one obtains

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho\phi \,\mathrm{d}V \approx \frac{(\rho\phi\Delta V_{P_0})^n - (\rho\phi\Delta V_{P_0})^{n-1}}{\Delta t} \tag{4.8}$$

and

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho\phi \,\mathrm{d}V \approx \frac{3\,(\rho\phi\Delta V_{P_0})^n - 4\,(\rho\phi\Delta V_{P_0})^{n-1} + (\rho\phi\Delta V_{P_0})^{n-2}}{2\;\Delta t} \tag{4.9}$$

for the implicit three-time-levels scheme. The approximations (4.8) and (4.9) reduce to those given in 3.1 if the grid does not move or moves rigidly without change of CV volume.

For convective fluxes no modifications are needed since the contribution from the grid movement is added in the mass fluxes, which are computed once while solving the continuity (pressure-correction) equation and treated as known in all other equations.

The continuity equation requires some attention. In order to explain how the continuity equation is discretized, we rewrite equation (4.1) in the following form:

$$\oint_S \rho\mathbf{v}\mathrm{d}\mathbf{s} = \oint_S \rho\mathbf{v}_s\mathrm{d}\mathbf{s} - \frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho \,\mathrm{d}V\,. \tag{4.10}$$

The term on left-hand side represents the total mass flux through the CV surface due to fluid, motion which corresponds to the continuity equation for a stationary grid. The right hand side of equation (4.10) involves the contribution from the grid movement. In order to obtain strict mass conservation, the mass fluxes due to grid movement has to cancel the unsteady term. This is fulfilled if the right-hand side is discretized according to the space conservation law as shown above (equations (4.5) and (4.7)).

The first term on the right hand side in equation (4.10) is added to the mass flux, which is now for a cell face j calculated as:

$$\dot{m}_j = \rho\,(\mathbf{v}_j - \mathbf{v}_s)\cdot\mathbf{s}_j = \rho\mathbf{v}_j\cdot\mathbf{s}_j - \rho\frac{\delta V_j}{\delta t}\,. \tag{4.11}$$

The second term on the right-hand side of equation (4.10) is added to the source term, which is modified as follows:

$$b_{p'} = b_{p'}^{\mathrm{stat}} + b_{p'}^{\mathrm{scl}}, \tag{4.12}$$

where superscript stat denotes the contribution from the stationary grid and scl denotes the contribution from the moving grid treated according to the space conservation law. Coefficient $b_{p'}^{\mathrm{scl}}$ is computed as follows:

$$b_{p'}^{\mathrm{scl}} = \begin{cases} -\rho \dfrac{\Delta V_{P_0}^{n} - \Delta V_{P_0}^{n-1}}{\Delta t} & \text{for IE} \\ -\rho \dfrac{3\Delta V_{P_0}^{n} - 4\Delta V_{P_0}^{n-1} + \Delta V_{P_0}^{n-2}}{\Delta t} & \text{for ITTL ,} \end{cases} \tag{4.13}$$

where IE stands for implicit Euler and ITTL for implicit three-time-levels scheme.

While imposing the boundary conditions for the momentum equation at solid walls, the velocity by which the wall surface moves needs to be taken into consideration. According to the no-slip condition the fluid velocity is equal to the wall velocity.

The continuity equation requires also special treatment near the wall. Since the walls are treated as impermeable, no flux through their surfaces is allowed. Therefore the volume fluxes due to wall motion have to cancel out the contribution of fluid velocity, which is equal to wall velocity (in accordance with no-slip boundary condition). Care needs also to be taken to enforce the global mass conservation. The outflow mass flux has to be adjusted not only to the total inlet fluxes, but also to the total mass source/sink produced by the wall movement.

4.3 Local-time-derivative based approach

Besides the ALE approach described above, which is the standard method also used in all commercial CFD codes, another novel method for treating moving grids was developed in this study. The idea of this method is to compute the flow on a moving grid by solving the same equations as for a stationary grid (Eulerian formulation of equations presented in chapter 2). Since the fully implicit method for integration in time is used, the fluxes and source terms in the governing equations are evaluated using solely the solution form the current time step. No data about the grid from previous time steps are needed, i.e. we do not have to know where the grid was and what shape the CVs had. This fact enables us to ignore the grid motion and to solve the governing equations in the usual way as for a stationary grid. The old solutions appear only in unsteady term which is represented by the local time derivative expressing the rate of change at a fixed point in space – the new position of CV center. In order to approximate the unsteady term it is necessary to use the variable values from one or two previous time steps (depending on the time-integration scheme) at the new grid position. For the mid-point rule approximation of the volume integrals, only the old values at the new position of CV center are needed. These values can be obtained by interpolation. In principle any kind of interpolation can be used. Generally the interpolation process consists of finding of sufficient number of donor points required for construction of interpolation functions and calculation of weighting factors. The main part of the interpolation process is the searching for donor points. If the grid topology does not change, the

interpolation procedure is significantly simplified, since the donor cells need to be sought in the immediate neighborhood of the corresponding cell, whose old position is known.

One possibility to obtain the old value at the new CV center is the following linear interpolation:

$$\phi_{P_0^{\mathrm{new}}}^{\mathrm{old}} = \phi_{P_0^{\mathrm{old}}}^{\mathrm{old}} + (\mathrm{grad}\,\phi)_{P_0^{\mathrm{old}}}^{\mathrm{old}} \cdot \left(\mathbf{r}_{P_0^{\mathrm{new}}} - \mathbf{r}_{P_0^{\mathrm{old}}}\right) \tag{4.14}$$

where, superscripts old and new denote the old and new solution or position respectively (see figure 4.2). The gradients of old variables are calculated at the old grid. Note that this interpola-

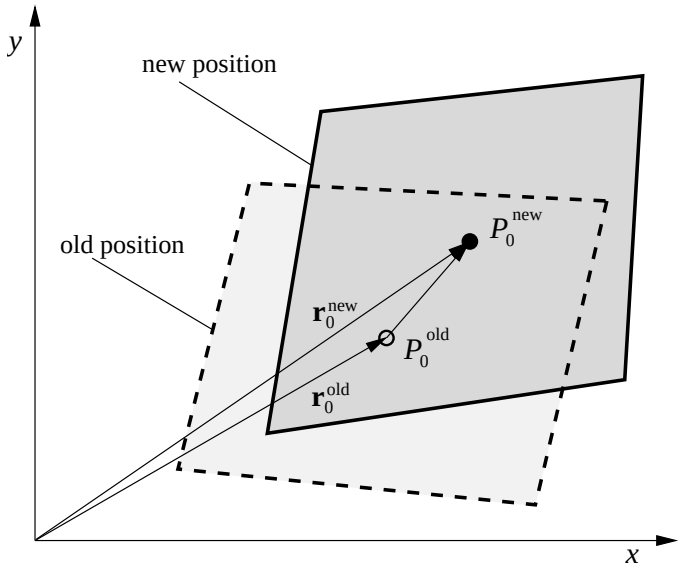


Figure 4.2: Old and new positions of a control volume between two consecutive time steps.

tion can be used only if the grid does not change its topology while moving, i.e. if every cell has its old counterpart. Furthermore it is desirable that the grid does not move significantly which is usually the case due to accuracy reasons. If the grid changes its topology, when for example re-meshing is made, equation (4.14) can still be used, but instead of point P_0^{old} one needs to find the nearest point at the old grid. Otherwise, interpolation shape functions can be used. This situation is illustrated in figure 4.3 where the old value at the point P_0^{new} is interpolated using four points from the old grid.

The major attractiveness of this approach is that it allows a change in grid topology at any time during simulation. Since the method described earlier needs to compute volumes swept by a cell face between two time steps, one cannot change the mesh topology so easily. Also, if one thinks of arbitrary 3D polyhedral cells, the computation of swept volumes can also be complicated. The above interpolation formula, however, is the same for any cell type. If the distance between cell center and the donor point is not larger than the half-cell width, the interpolation error is of the same size as in the evaluation of cell-face value for convective fluxes. It can thus be expected that the additional interpolation error will not substantially increase the overall discretization error.

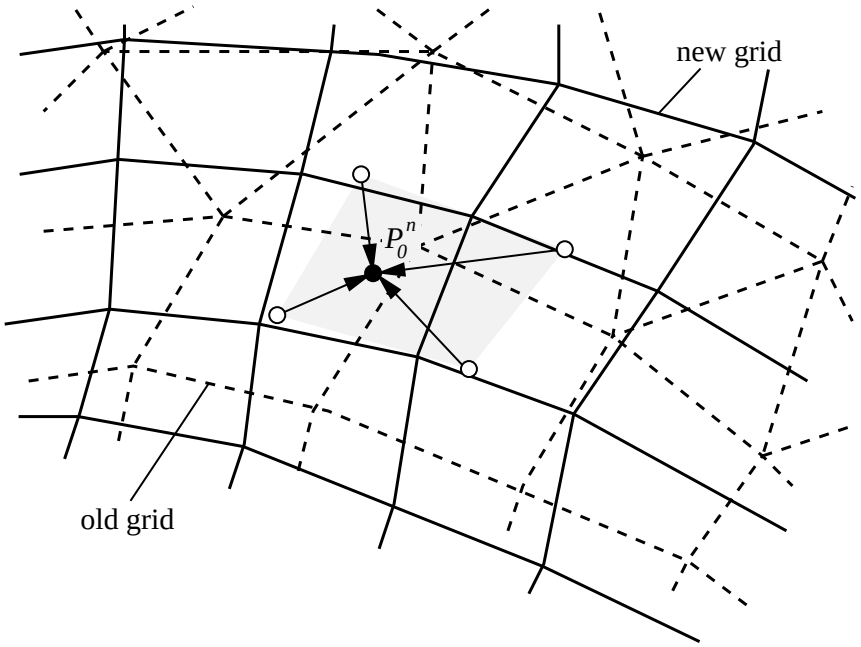


Figure 4.3: Interpolation of the old solution to the new grid when the grid topology has changed between two successive time steps.

4.3.1 Near-wall treatment

Since the moving walls have an important influence on the flow in the near-wall region, care needs to be taken to account for the wall movement properly. Due to wall movement the fluid is either displaced or the space is made to be filed by fluid. In order to take this effect into account, the mass flux, which corresponds to the amount of fluid displaced by the wall or fluid which fills the space made by the wall, is added to the source term in the pressure-correction equation. This additional mass source can be computed using the wall velocity and cell face area at the current time level (see figure 4.4). The source term in the pressure-correction equation is thus modified as:

$$b_{p'} = b_{p'}^{\mathrm{stat}} - (\rho \mathbf{v}_{\mathrm{w}} \cdot \mathbf{s}_{\mathrm{w}})^n \ , \tag{4.15}$$

where $\mathbf{v}_{\mathrm{w}}$ is the wall velocity and the term $b_{p'}^{\mathrm{stat}}$ represents the usual contribution when the grid does not move. Contribution from the wall movement given by equation (4.15) allows that boundary faces may change between two time steps (re-meshing).

In the momentum equation, as usual, fluid velocity at the wall should be given the value of the velocity by which the wall moves.

4.4 Assessment of moving grid methods

In this section we consider two examples which were selected as the test cases for assessment of the methods for grid movement presented in the previous sections. Our attention is focused on

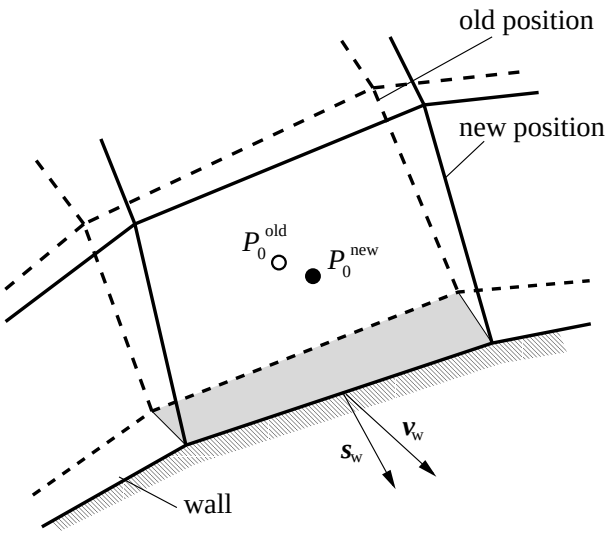


Figure 4.4: On the treatment of wall movement.

the estimation of the accuracy and efficiency of the two methods as well as on the applicability of the method proposed in section 4.3 in conjunction with re-meshing technique.

4.4.1 Piston-driven flow in a pipe contraction

As a suitable example for an assessment of the methods for moving grids, the unsteady flow in a pipe with sudden expansion driven by a piston was computed. Both numerical and experimental investigations of piston-driven flows have been carried out in the past [76, 78, 77, 25]. An extensive numerical study for various parameters, such as initial piston clearance, stroke length, piston velocity, etc., is provided in reference [76]. We refer here to the computations and experimental data reported in [78].

The geometry and dimensions of the computational domain for the test case studied here are shown in figure 4.5. The flow is induced by a piston which moves periodically between two limiting points ($x_{p_{\min}}$ and $x_{p_{\max}}$) inside of the larger pipe[2]. The variation of the piston velocity and position during one cycle is given in figure 4.6. The piston, initially positioned at $x_p = 21\,\mathrm{mm}$ (minimum piston displacement), accelerates strongly at the beginning of the intake stroke and within $0.05\,\mathrm{s}$ reaches the velocity of $6.22\,\mathrm{mm/s}$ which remains constant until $0.05\,\mathrm{s}$ before the end of the intake stroke. In the exhaust stroke the acceleration phase lasts $0.18\,\mathrm{s}$ within which time the piston reaches the maximum velocity of $-26.5\,\mathrm{mm/s}$, which remains constant until $0.18\,\mathrm{s}$ before the end of the exhaust stroke.

The flow is regarded as laminar and axisymmetric. The fluid properties used in the computation ($\rho = 1\,\mathrm{kg/m^3}$ and $\mu = 3.89 \cdot 10^{-6}$) are chosen so that the Reynolds number based on the diameter of the larger pipe (D) and the piston velocity (u_p) is 72 in the intake stroke and 306 in the exhaust stroke, as in experiments reported in [78]. Furthermore, the length of the smaller

[2] Also referred to as cylinder.

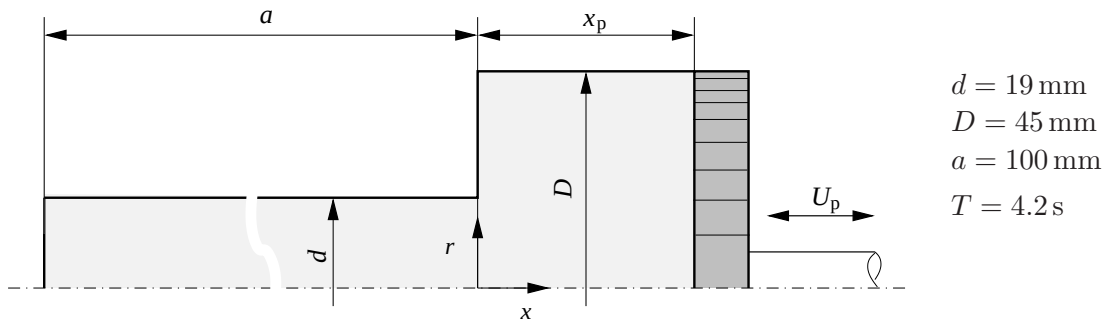


Figure 4.5: Geometry of the solution domain for the piston-driven flow.

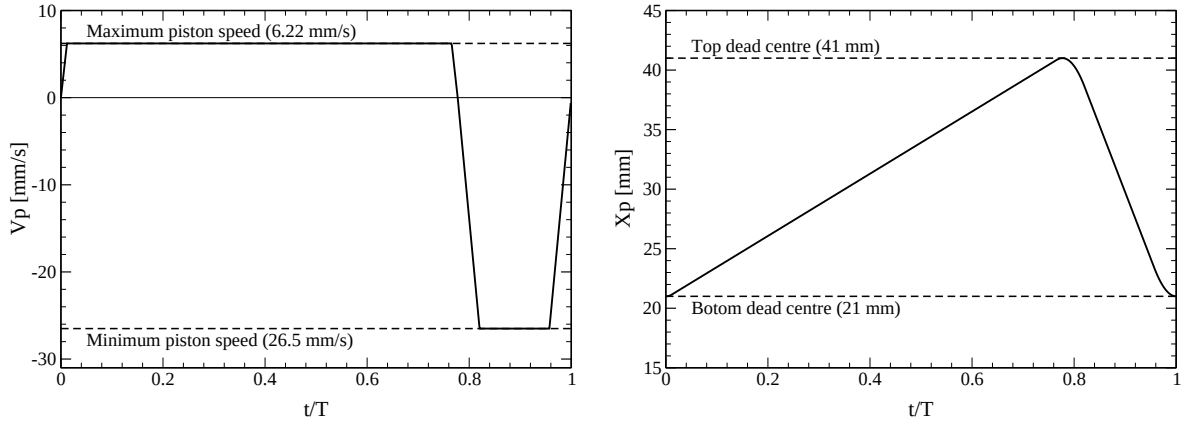


Figure 4.6: Piston velocity (left) and piston clearance (right) as functions of time.

pipe was $100\,\mathrm{mm}$ in the computation (as in the simulation reported in [78]). Along the pipe axis the symmetry boundary condition is applied and at the inlet to the smaller pipe we imposed the pressure boundary condition, which allows the change of the flow direction, rather then the standard outlet boundary condition, which in such cases may become instable. All other boundaries are considered as impermeable walls and no-slip boundary conditions are applied there. At the piston surface the fluid velocity is set to the piston velocity while at other walls it was set to zero.

To assess the accuracy, we performed the computations on three systematically refined grids which had 1630, 6520 and 26080 CVs, respectively. The coarse grid is shown in figure 4.7. An orthogonal nonuniform grid is employed as the best choice for the given geometry. The grid in the large pipe region is adjusted at each time step to conform to the domain boundaries defined by the new piston position. This is achieved by moving the vertices in axial direction proportional to the piston displacement. Thus the cells in the large pipe region are deformed but the grid topology remains unchanged. Convection fluxes are discretized using the CDS, providing a second-order discretization in space. For time integration we used the implicit Euler scheme, which was found to produce satisfactorily accurate results. The preliminary computation with the three time levels scheme, performed on the medium grid, did not bring any significant change in the results. A constant time step of $0.00125\,\mathrm{s}$ was used during the entire cycle and on all grids. This time step

size was chosen to be small enough to ensure that the Courant number stays close to unity on the finest grid.

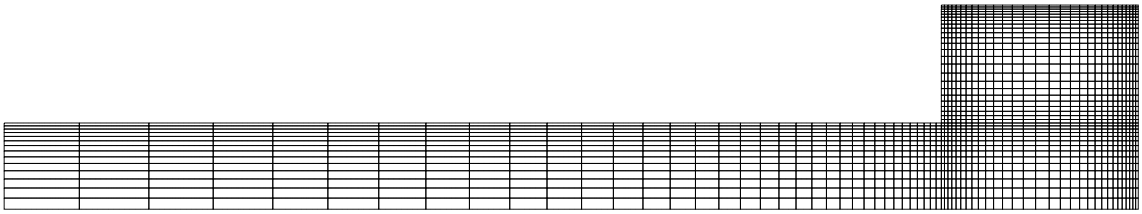


Figure 4.7: Initial coarse grid (1630 CV) for the piston-driven flow.

The computations on all three grids were performed using both techniques for grid movement, namely: arbitrary Lagrangeian-Eulerian approach with the space conservation law, denoted as SCL, and the new approach based on the approximation of the unsteady term by a local time derivative proposed in section 4.3, denoted as LTD. In addition, for further verification of our results, we also performed the computations using the commercial program Comet which employs the ALE approach and the space conservation law [37].

The computation on each grid was started from zero fields. The outer iterations were repeated within each time step until the normalized sum of absolute residuals has fallen by five orders of magnitude. This was done in order to ensure that iteration errors are negligible compared to discretization errors; normally a reduction of residuals by three orders of magnitude is sufficient. On average about 15 outer iterations per time step were needed to achieve the required convergence. Table 4.1 shows the average CPU time per time step for all three computations and for all three levels of grid refinement.

Table 4.1: CPU time in seconds per time step for the piston-driven flow computation.

	Grid level		
Method	1	2	3
SCL	0.227	1.732	13.943
LTD	0.215	1.551	11.803
Comet	0.229	1.944	15.501

The development of the pressure and velocity fields during the first cycle is shown in figures 4.10 and 4.11 (pages 47 and 48). The flow development in subsequent cycles is expected to be different. Continuing the computation on the medium grid over several cycles, the periodic behavior of the flow has been observed from the second cycle onwards.

For the estimation of discretization errors, we compared the results obtained on different grids. An average difference between the solutions on the two successive grids defined as:

$$\varepsilon_\phi = \frac{\sum_{i=1}^{N} |\phi_i^C - \phi_i^F|}{N} \tag{4.16}$$

was computed, where ϕ stands for a dependent variable and C and F stand for coarse and fine grid, respectively. Since the computational points are at different locations on the two grids, the data from finer grid were linearly interpolated to the cell centers on the corresponding coarser grid. The differences obtained for the u velocity at five piston positions during the first cycle are given in table 4.4.1. It can be observed that the differences are approximately in the same range for all three methods. They decrease with a factor between 2.5 and 4 as the grid is refined; asymptotically the factor of four should result for a second-order method. The LTD method has consistently slightly higher errors, but the difference is not significant.

Table 4.2: Average differences between two coarser grids ε_{12} and two finer grids ε_{23} obtained for the u velocity component at five piston positions during the first cycle and using three different methods as indicated.

			Intake stroke		Exhaust stroke	
Method	$x_p \rightarrow$	26.6680	32.4992	38.3305	40.1324	25.6044
SCL	ε_{12}	6.63686E-05	8.44496E-05	9.34186E-05	5.38343E-04	9.37905E-04
	ε_{23}	2.63986E-05	2.48267E-05	2.49655E-05	1.31143E-04	3.64998E-04
LTD	ε_{12}	6.96606E-05	8.88590E-05	1.03009E-04	5.54924E-04	1.03106E-03
	ε_{23}	2.75505E-05	2.56196E-05	2.67919E-05	1.38675E-04	3.82655E-04
Comet	ε_{12}	7.45466E-05	8.96748E-05	9.29392E-05	5.62832E-04	8.22030E-04
	ε_{23}	2.69811E-05	2.68979E-05	2.54343E-05	1.32231E-04	1.85277E-04

A qualitative comparison of the results obtained using different techniques with each other is given in figure 4.8, which presents the profiles of the axial velocity component at two characteristic locations for two piston positions in the intake and the exhaust stroke. The presented results obtained on the medium grid agree obviously very well – no appreciable difference can be visually observed.

Figure 4.12 on page 49 shows comparison between the computed and measured profiles of the axial velocity component at various locations for the piston position $x_p = 34.847\,\mathrm{mm}$ in the first intake stroke. The flow field for this position is shown in Figs. 4.10 c) and 4.11 c). The results presented in figure 4.12 refer to those obtained using the present ALE approach and the treatment of the grid movement according to the space conservation law (SCL), which were selected as the reference results. Having in mind figure 4.8, comparison of the profiles obtained using the other two techniques was not necessary. Except of under-predicted negative velocity in the region near the center of the primary vortex, a qualitatively good agreement between predicted and measured profiles can be observed. Some discrepancies are also visible in some profiles near the symmetry axis, but disturbances in measured profiles at $x = 11$, $x = 26$ and $x = 29\,\mathrm{mm}$ may suggest some uncertainties in experiments or the flow asymmetry as the reason for the disagreement. Nevertheless, the level of agreement obtained here is comparable to that from [78].

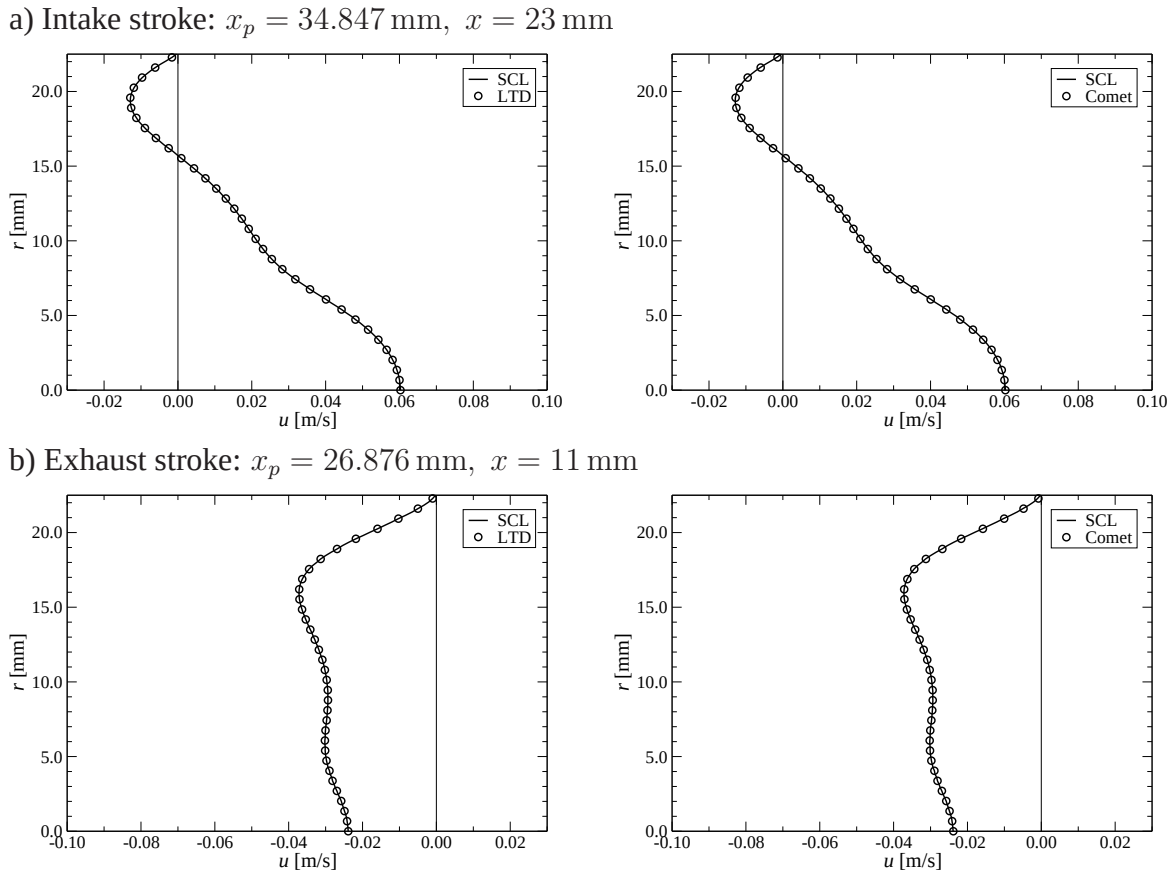


Figure 4.8: Comparison of results obtained using the SCL technique with: a) results obtained using Comet and b) results obtained using LTD technique. Presented are profiles of the axial velocity calculated at two piston positions.

4.4.2 Re-meshing

In the case considered in the previous section, the grid movement was achieved by deforming the grid in the large pipe region. Sometimes, the grid deformation caused by grid motion may become extremely large, resulting in severely poor grid quality. In such cases it may be necessary to replace the existing mesh with another one with better properties. This practice is called re-meshing. In order to continue the computation after re-meshing, the flow information needs to be transferred from the old to the new grid. The new method for moving grids, proposed here, is suitable for computations with re-meshing, since the data transfer between two successive grids can be accomplished relatively easily. Since a fully implicit integration in time is used, convection and diffusion fluxes are completely approximated using variables and geometrical data from the current time step. Furthermore, the LTD-treatment of the grid movement described in section 4.3 does not need any information about the grid from the previous time step. Thus, the only information that needs to be transferred from the old to the new grid are the values of the dependent variables at the cell centers.

In this section we investigate the possibility of using this new method for the grid movement

in combination with re-meshing technique for handling of problems with moving boundaries. The same example as in the previous section was considered, but we modified the piston movement in order to get a stronger compression in the large pipe region. The variation of the piston velocity, which is given as a sines function, and the corresponding piston position are given in figure 4.9. The Reynolds number based on the maximum piston velocity of $9.42\,\mathrm{mm/s}$ was 109,

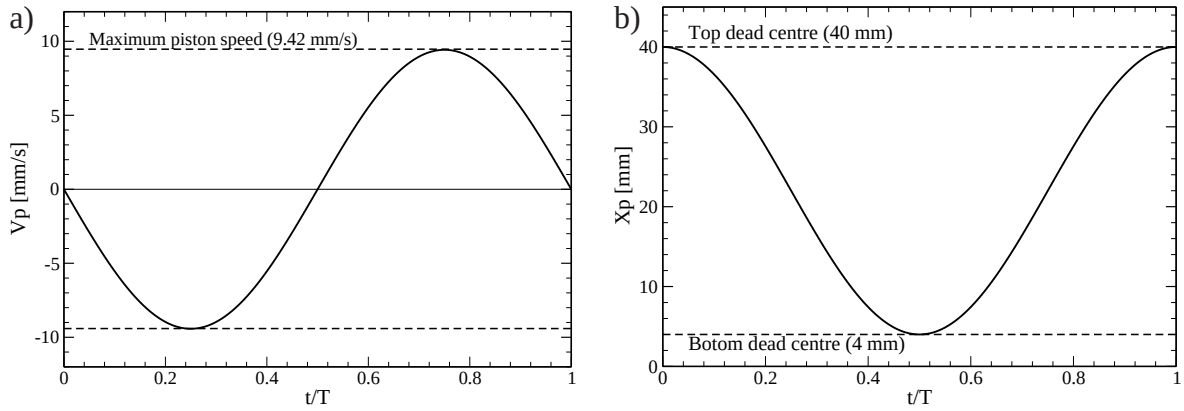


Figure 4.9: Piston clearance (left) and piston velocity (right) as functions of time.

yielding a laminar flow. The piston moves between the top and bottom dead center, which correspond to the piston position of $x_{\mathrm{p_{max}}} = 40\,\mathrm{mm}$ and $x_{\mathrm{p_{min}}} = 4\,\mathrm{mm}$, respectively. Thus, the volume occupied by the piston in the large pipe reduces at the end of exhaust stroke to $1/10$ of the initial volume defined by the piston position at the top dead center. A relatively high compression ratio leads to strong deformation of cells in the large pipe region when the initial grid topology is preserved.

We performed the computations on a single grid (without re-meshing) and using the re-meshing technique. The initial grid used in the computation had 3780 cells, therefrom 40×57 cells in the large pipe region. The re-meshing was made three times during a stroke, whereby the number of cells in axial direction in the large pipe was reducing/increasing while the number of cells in radial direction remained the same. The re-meshing relevant data that contain the information about time and position of the re-meshing as well as the number of cells in axial direction are given in table 4.3. Using the re-meshing the number of cells in x direction in the large pipe

Table 4.3: Data for re-meshing.

t/T	$x_{\mathrm{p}}\,[\mathrm{mm}]$	Reduction of cell numbers in x direction in large pipe region
$1/4$	22.0000	$40 \rightarrow 32$
$1/3$	13.0000	$32 \rightarrow 26$
$47/120$	8.0114	$26 \rightarrow 20$

was reducing from 40 at the beginning to 20 at the end of exhaust stroke. In the intake stroke the

number of cells was increasing in reverse order.

The computation was first carried out on a single grid over two cycles which were enough to get the periodic solution. These results were stored and used as initial field for the computations with both techniques over a further cycle. Discretization in space and time was the same as in the previous example, the time step used was $0.02\,\mathrm{s}$.

Several characteristic flow features were used for the assessment of results. Figure 4.13 shows the time history of the pressure and shear forces exerted on the piston. From the pressure force presented in figure 4.13 a) one can conclude that the pressure field near the piston surface remains almost unaffected by re-meshing. On the other hand, for shear force some discrepancies are visible immediately after each re-meshing. The differences are more pronounced during the intake stroke. However, although the differences look in a graphical presentation like significant, one has to take into account that shear force is about 200 times smaller that the pressure force, since the velocity tangential to piston surface is relatively small. The shear stress is rather sensitive on the grid resolution near the wall which changes significantly after each re-meshing. This might be considered as the main reason for these differences.

Comparison of profiles of the axial velocity component calculated at the entrance into the smaller pipe (see figure 4.14) shows very good agreement between the two results. Profiles were computed for six different piston positions during the exhaust and intake stroke. In this case only slight differences can be observed in profiles that were computed when the piston reaches the bottom dead center.

Figure 4.15 shows the numerical grid and the pressure field for six positions of the piston during a cycle. Pressure fields computed using the two approaches show also very good agreement. It is obvious from the pressure field that the data transfer during the re-meshing is done correctly. Small differences in the pressure contours that can be recognized may be expected due to differences in the numerical mesh, which become significant as the piston approaches the bottom dead center.

4.5 Concluding remarks

The results obtained for the two test simulations presented in previous sections show that the novel LTD-technique for handling grid motion produces solutions of the same accuracy as the standard method based on space-conservation law. The advantages of the new technique are:

- It does not require computation of volumes swept by all faces from one time step to another, which can be both complicated and time consuming if polyhedral cells with a large number of faces are used. The effort needed to interpolate the values from the previous time step to the new cell-center location in the new method is independent of the cell shape.
- It can be conveniently used in computations with re-meshing, where both the number of cells and possibly also their shape changes so swept volumes cannot be computed.

The new method requires search for nearest neighbors on previous mesh when re-meshing is used; this is also the case when overlapping meshes are used, so this topic will be covered in the next chapter.

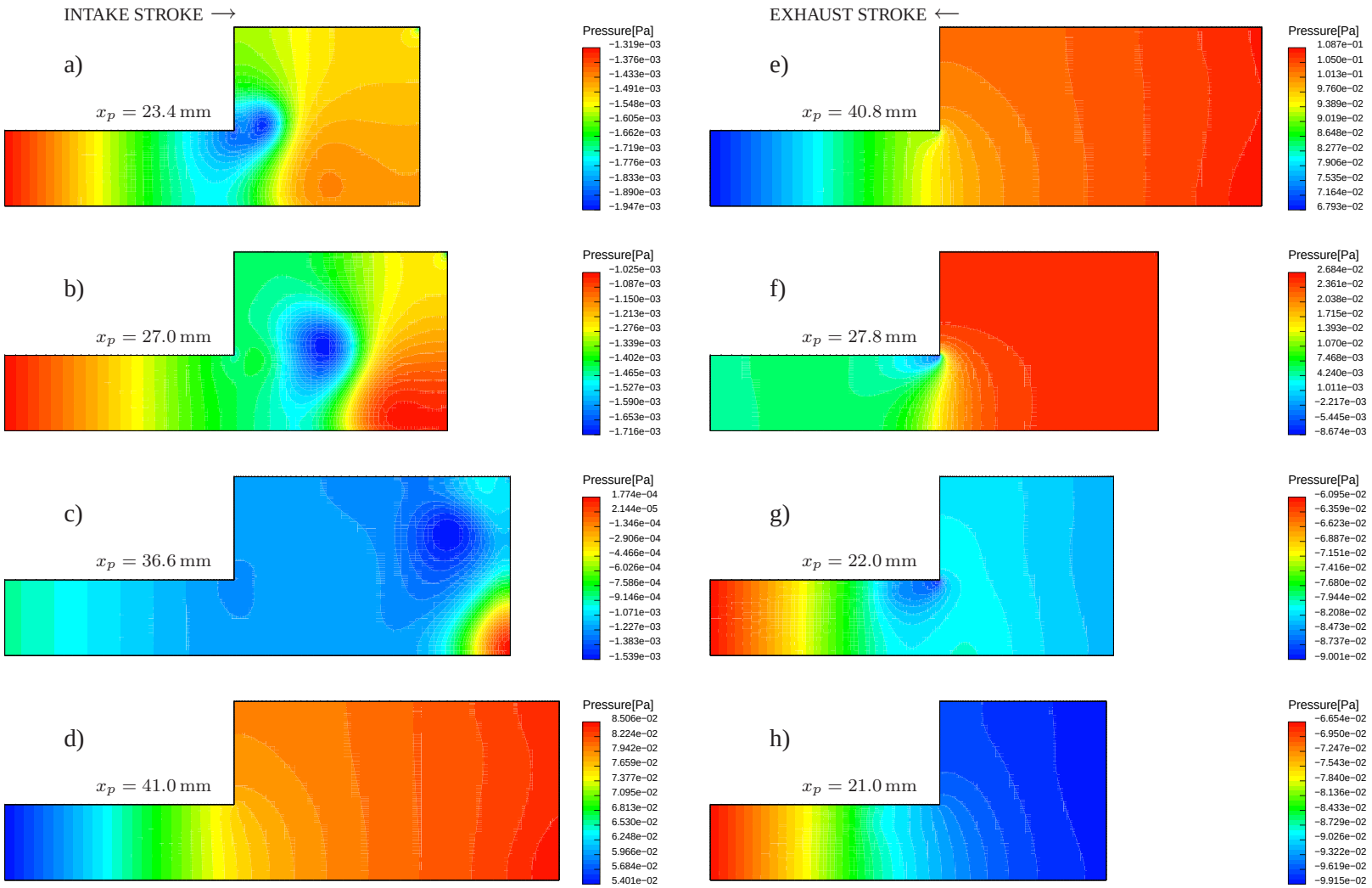


Figure 4.10: Pressure field for several positions of the piston during the intake stroke (left) and exhaust stroke (right).

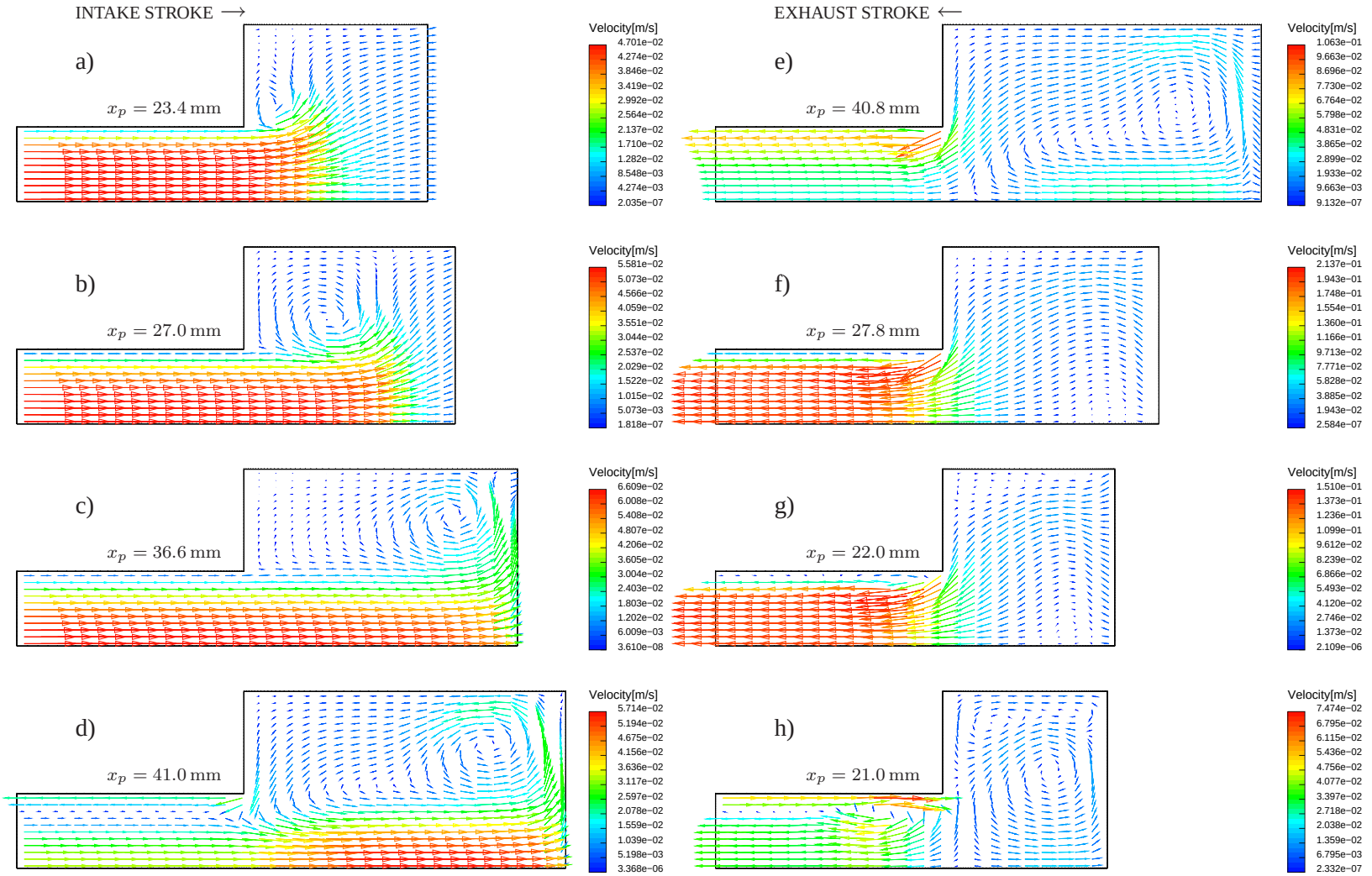


Figure 4.11: Velocity field for several positions of the piston during the intake stroke (left) and exhaust stroke (right).

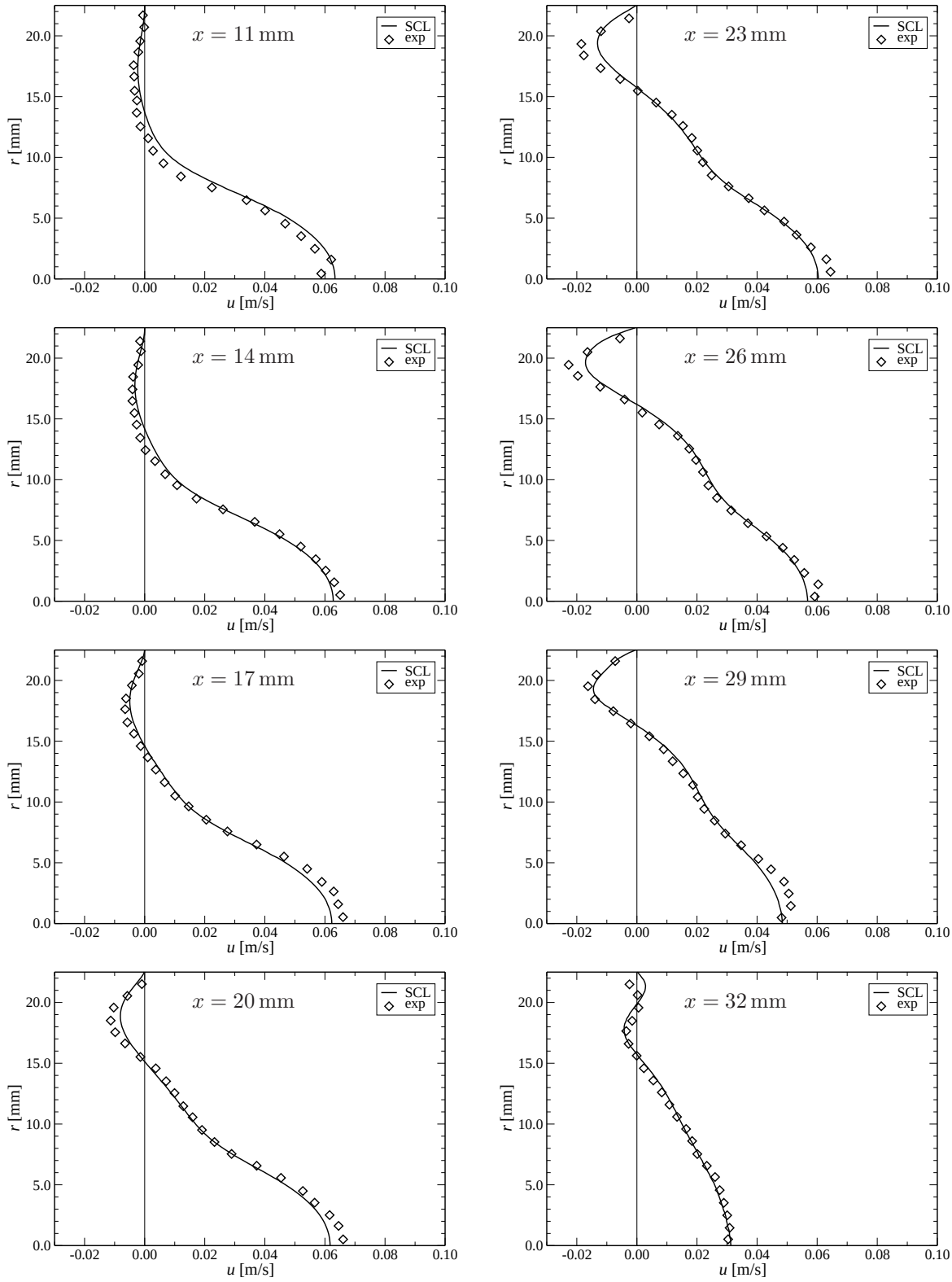


Figure 4.12: Comparison of computed and measured [78] profiles of the axial velocity component for the intake stroke of the first cycle and piston position $x_p = 34.847\,\mathrm{mm}$.

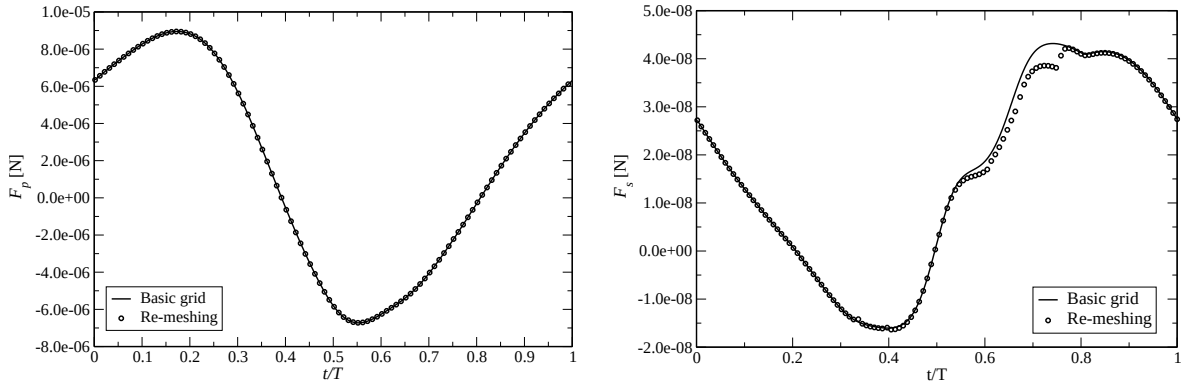


Figure 4.13: Pressure force (left) and shear force (right) at piston surface, computed using single grid and re-meshing.

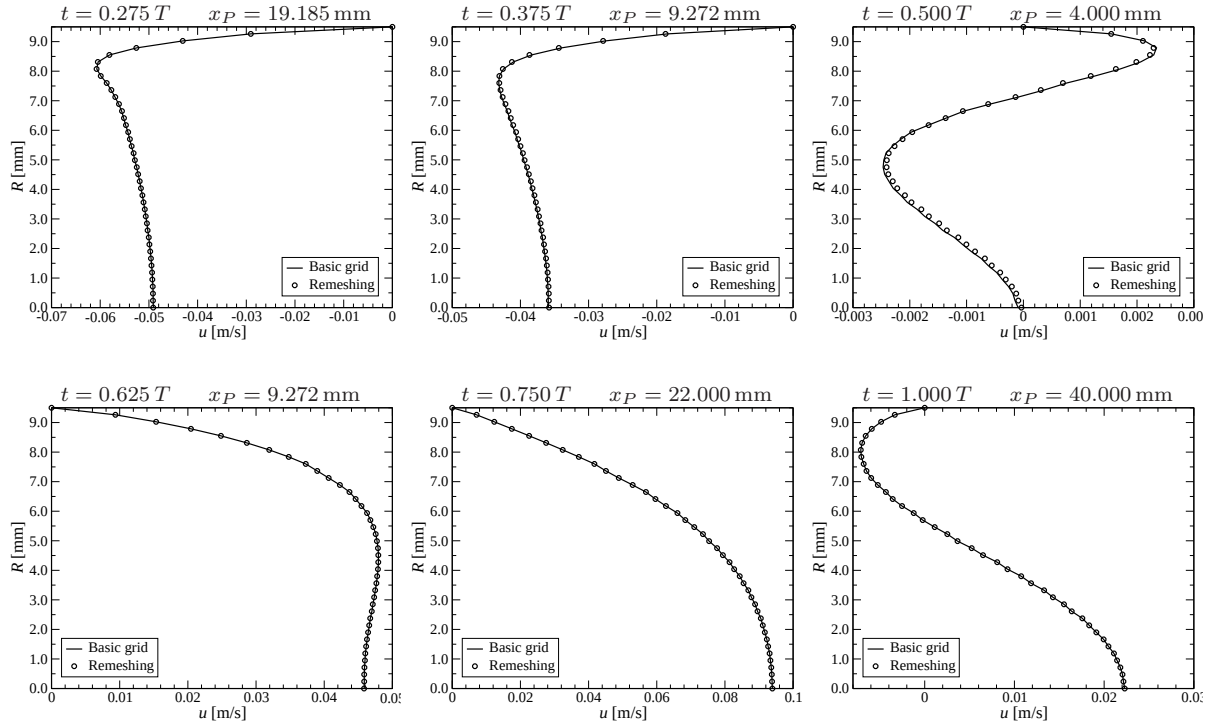


Figure 4.14: Comparison of profiles of the axial velocity component along the cross section at the entrance into smaller pipe for six different piston positions during a cycle. Results obtained on a single grid and using re-meshing technique are presented.

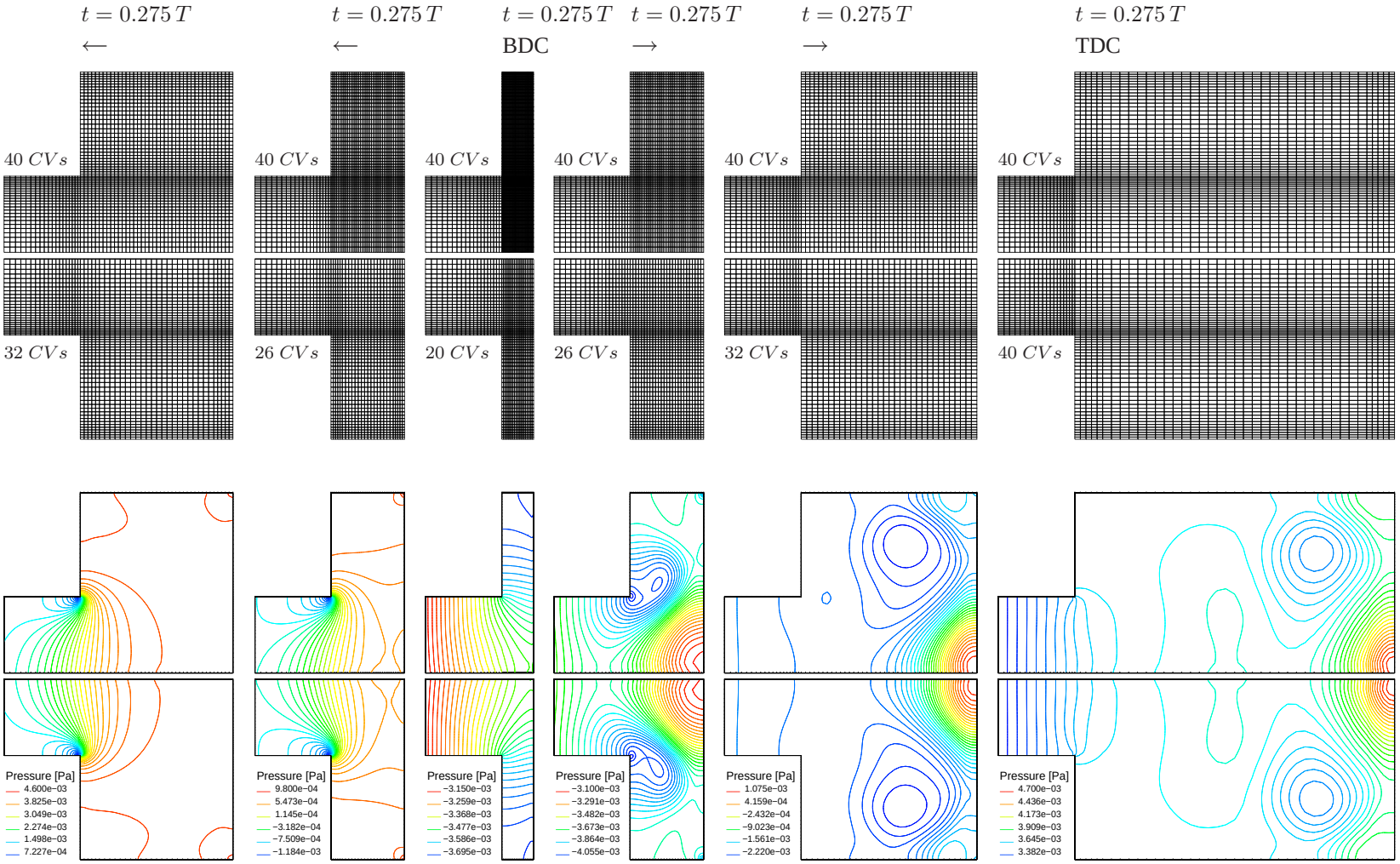


Figure 4.15: Numerical grid and the pressure field for several piston positions. Number of control volumes in x direction in the cylinder region and the current direction of piston movement are indicated. The upper part of each plot is for single grid computation and the lower part is for computation with re-meshing.

CHAPTER 5

Overlapping grid technique

In this chapter the *overlapping grid technique* (also called Chimera grids technique) developed in this study is described. The principals of the overlapping grid methodology are first outlined. The algorithms for hole cutting and donor searching used in the present study are then described. The main attention has been paid to the solution of the governing equations on overlapping grids. Special features concerning the integration of the inter-grid coupling in the global solution procedure and mass conservation at overlapping interfaces developed within this study have been discussed in detail.

5.1 Introduction

The overlapping grid techniques, also known as "Chimera" or overset grid techniques, have been developed by several authors in the past [7, 22, 31, 74, 15]. In these methods, the computational domain is covered by a number of grids which overlap with each other in an arbitrary manner. An advantage of such a grid arrangement is the possible employment of grids of high quality (e.g. orthogonal, cylindrical, spherical grids or other grids with good numerical properties) when handling complex geometries. Grid components can be added or altered to represent the arbitrary shape of real-life geometries, thus permitting a great deal of flexibility in the discretization of multi-connected domains. This approach has much of its advantages in computation of multiple-body and moving body problems as well as in optimization-study problems.

Nowadays, the unstructured grids are the rule rather then exception in CFD, due to their flexibility in treatment of complex geometry. Furthermore, some other features like adaptive local grid refinement capability and automatic grid generation make unstructured grids attractive for complex industrial applications. Thus, the leading commercial CFD flow solvers are based on unstructured grids. The solution algorithms for unstructured meshes are also improving [68, 45], thus continuously increasing the efficiency of unstructured solvers. Although the unstructured grids are capable to accommodate the domains of an arbitrary level of complexity, their use in situations with bodies in relative motion or optimization studies still involves some difficulties and limitations. In order to treat such problems properly using single domain-fitted grids, the grid deformation is necessary to accommodate the domain boundaries which are changing due

to body motion. If the motion of bodies is large, the grid deformation becomes severe and a part of the computational grid or, in some cases, the whole grid has to be newly generated.

Owing to the flexibility in matching of different grid components, the overlapping grid techniques, on the other hand, offer much more flexibility in handling the problems that involve multiple bodies in relative motion. In that case, as the component grids move relative to each other, only the location of boundary points at overlapping interfaces that are involved in interpolation changes. The grid points do not need to be regenerated and the grids retain their topology and geometrical properties. Similarly, in the optimization studies one needs often to modify the geometry (e.g. to exchange one part of a configuration with a new one) while searching for the optimal design. As a consequence, in most cases a new grid has to be generated for the new configuration. Overlapping grid techniques offer also in this case a great flexibility, since the grid, if at all, usually needs to be modified only locally (e.g. around a new part). This is the main advantage of the overlapping grid approach over the standard techniques that employ single domain-fitted grids, which require grid deformation and regeneration to account for body motion or modification of the geometry. Unstructured grids in combination with overlapping grid techniques still have advantages over structured overlapping grids. Thus, using the unstructured overlapping grids for moving bodies the number of component grids necessary to cover the computational domain may be reduced. This in turn reduces the complexity of the overlapping grid systems used for such computations, which positively reflects on the efficiency of the method.

Early examples of applications of the overlapping grids are shown by Atta [1], who computed the solutions of the full-potential equation on an underlying Cartesian grid with an overlapping curvilinear body-fitted grid around an airfoil. The overlapping grid method was extended to the solution of the full-potential equation about multi-component configurations [2]. Some solutions of the Euler equations for flows around multi-component configurations are presented by Benek *et al.*[7, 8] and Steger and Benek [74]. Chesshire and Henshaw [15] developed a general algorithm for the generation of overlapping grids. They describe some techniques for the solution of elliptic and time-dependent PDEs and present some applications to the solution of the compressible Navier-Stokes equations on such grids. Henshaw [32] describes a fourth-order accurate finite difference method for the solution of incompressible time-dependent Navier-Stokes equations on overlapping grids. He employs an explicit time scheme to update the velocity and the multigrid algorithm [33] to solve a Poison equation to obtain the pressure. The method used by Petersson [61] is essentially the same, but second-order accurate in space. Hinatsu and Ferziger [34] and Burton and Eaton [14] used a fractional-step method in time in combination with a second-order finite volume discretization in space to solve unsteady incompressible Navier-Stokes equations. A method for the solution of steady incompressible Navier-Stokes equations on overlapping grids, based on use of SIMPLE algorithm on staggered grid systems, is presented by Wright and Shyy [93]. Tu *et al.*[83, 85, 84] used an overlapping grid technique and multigrid method for computation of flows in internal combustion engines. Further applications of the overlapping grids to the flow computations are given in [6, 35, 13, 88] among others. Except to accommodate complex geometry, the overlapping grid techniques was used by some authors for the adaptive local grid refinement [10, 38].

Recent development of methods for unstructured grids has motivated researchers to apply the overlapping grid techniques to such grids as well. Several publications have recently appeared that present overlapping grid techniques for unstructured grids. Nakashi *et al.*[55] developed an overlapping grid method for unstructured grids and used it in combination with a finite volume cell-vertex scheme to solve compressible Euler equations around complex multi-component configurations [81]. Further applications of unstructured overlapping grids to the solution of Euler and Navier-Stokes equations are given in [44] and [47].

All of these methods use either block-structured or unstructured triangular or tetrahedral grids and were developed to handle a particular class of problems (incompressible or compressible, Euler or Navier-Stokes equations).

The present method is designed to:

- achieve a strong coupling of all grids for an efficient iterative solution up to the round-off level of residuals,
- provide smooth solutions in overlapping regions,
- achieve global mass conservation across all grids,
- be applicable to arbitrary polyhedral grids,
- allow an arbitrary motion of bodies relative to each other.

These properties ensure not only an efficient iterative solution procedure but also the accuracy of the solution in the neighborhood of the overlapping region which is consistent with the accuracy in the rest of the solution domain. In particular, the method guarantees that the solution in the overlapping region is the same on all overlapping grids, as will be demonstrated in applications presented in chapter 6.

5.2 Outline of overlapping grid methodology

Two principal elements of the present overlapping grid technique are:

- decomposition of computational domain into sub-domains, and generation of a suitable grid in each sub-domain,
- development of a coupling method for an accurate, efficient and unique solution of the governing equations on the overlapping grids.

The basic idea of this approach is to break the computational domain into a number of smaller regions (sub-domains), which are in general associated with components of a configuration. Each sub-domain is covered by a suitable grid, which is in general simpler and easier to generate than a single grid for the whole domain. The component grids associated with sub-domains are not required to match in any special way, but they have to overlap sufficiently to provide the means of

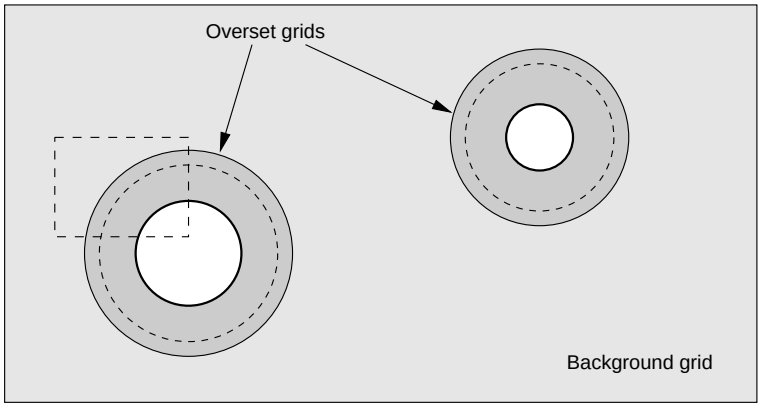


Figure 5.1: An overlapping grid system consisting of two minor grids attached to the bodies and embedded into major background grid stretched over the entire domain.

coupling the solutions between the grid components. The concept of overlapping grid approach is illustrated in figure 5.1. Two separate grids are generated around each body covering a small area close to bodies, while the remaining part of the domain is covered by another grid, which is usually called the background grid. In many situations the background grid can be a simple Cartesian grid. The grids attached to bodies are completely embedded in the background grid. On the other side, parts of background grid lie within bodies and are thus outside the flow field. These parts of the background grid have to be excluded from the computation and as such may be physically removed. In the case of moving bodies it might be necessary to keep all grid cells, since, due to body motion, different parts of the grid are covered by the bodies at different times. Therefore, the cells which are covered by the bodies are deactivated rather then removed. These cells are called inactive (hole) cells. In order to separate inactive cells from the rest of the domain, an artificial boundary within the background grid has to be created. Nodes in the centers of cells along such a boundary within a background grid are denoted with "$\circ$" in figure 5.2, which shows a detail of the overlapping region masked by the dashed rectangle in figure 5.1. The variable values at these cells and cells along the outer boundary of the overset body grid (denoted with "$\bullet$") are obtained by interpolation of the variable values from the other component grid (donor grid).

According to their role in the solution process of the governing equations, the cells in an overlapping grid system may be divided into three groups: *discretization (active), interpolation, and inactive (hole) cells*. Discretization cells are used to discretize governing equations, interpolation cells receive the solution information by interpolation and inactive cells are disregarded during the solution process. All three cell types are shown in figure 5.2.

5.2.1 Inter-grid communication

When all component grids necessary to appropriately cover the computational domain are generated, the next step is to establish the inter-grid communication. This is necessary in order to

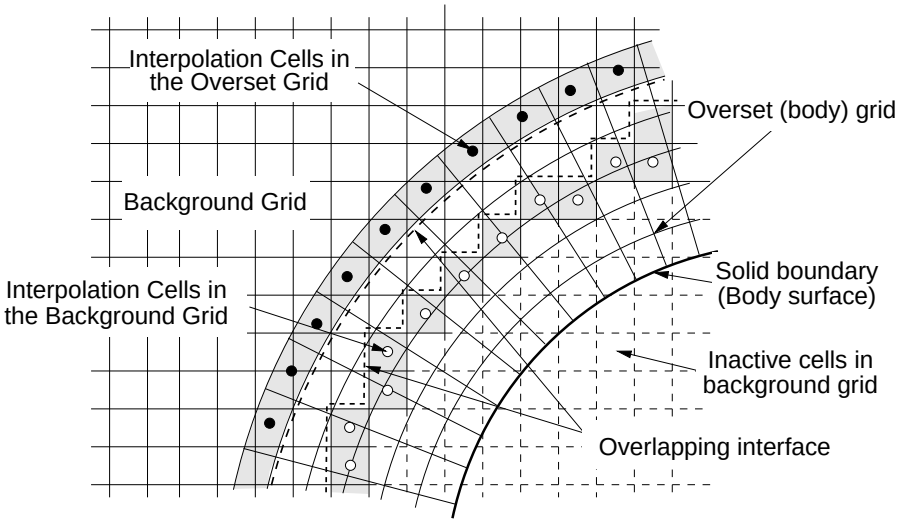


Figure 5.2: A detailed view of the overlap region with some definitions and notation.

be able to obtain a unique solution over the whole domain. As described above, some parts of the grid may be outside of the computational domain, requiring to be deactivated or removed. On the other hand, in order to obtain the solution on each part of the grid, the cells involved in the coupling of the component grids have to be identified. Thus, there are two major steps to establish the inter-grid communication:

1. Hole cutting, which involves the identification of the cells that are outside of the computational domain,

2. Identification of interpolation stencils which are used to construct the interpolation formulas for grid coupling.

In the first step all cells are divided into two groups: active and passive. After that the inter-grid boundary cells are identified as the active cells which have common faces with passive cells. For cells which lie along the outer boundary of a component grid (cells denoted by "•" in figure 5.2), the type of the boundary region is used as the criterion for the type classification of these cells. During the grid generation, the outer boundary regions are assigned a special boundary type (overlap boundary) and the cells that lie along such boundaries are recognized as interpolation cells.

Hole cutting

In this step, all grid cells which lie outside of the computational domain have to be identified in order to be treated appropriately during the computation. To explain the algorithm developed for purposes of this study, consider the overlapping grid system shown in figure 5.3, which consists of two component grids: a background grid and another grid around the body overset on the background grid. The cells in the background grid which are covered by the body have

to be excluded from the computational domain. In addition to these cells, a major part of the background grid covered by the body grid can also be marked as inactive. Only an overlapping region consisting of a number of cell layers necessary for inter-grid coupling needs to be left. For selecting the regions that need to be deactivated, the outer boundary C_0 of the body grid is used as a reference for the hole creation. All cells which are outside of the curve C_0 remain active. The cells inside of the curve C_0 are further tested to check if they are inside of the overlapping region or need to be deactivated. A mesh point is considered to be inside the curve C_0 if the dot product between the **r** (a vector from the closest boundary point on the curve C_0 to the mesh point in question, see figure 5.3), and **n** (the normal vector on the curve C_0 at the closest boundary point, directed outward from the hole region) is negative. For cells inside the curve C_0 the distance $\delta = |\mathbf{n} \cdot \mathbf{r}|$ is used to determine if the cell is inside the overlapping region (if $\delta < \delta_0$, see figure 5.3) when it should remain active, or outside the overlapping region, when it needs to be deactivated. The width of the overlapping zone δ_0 depends on the grid spacing in

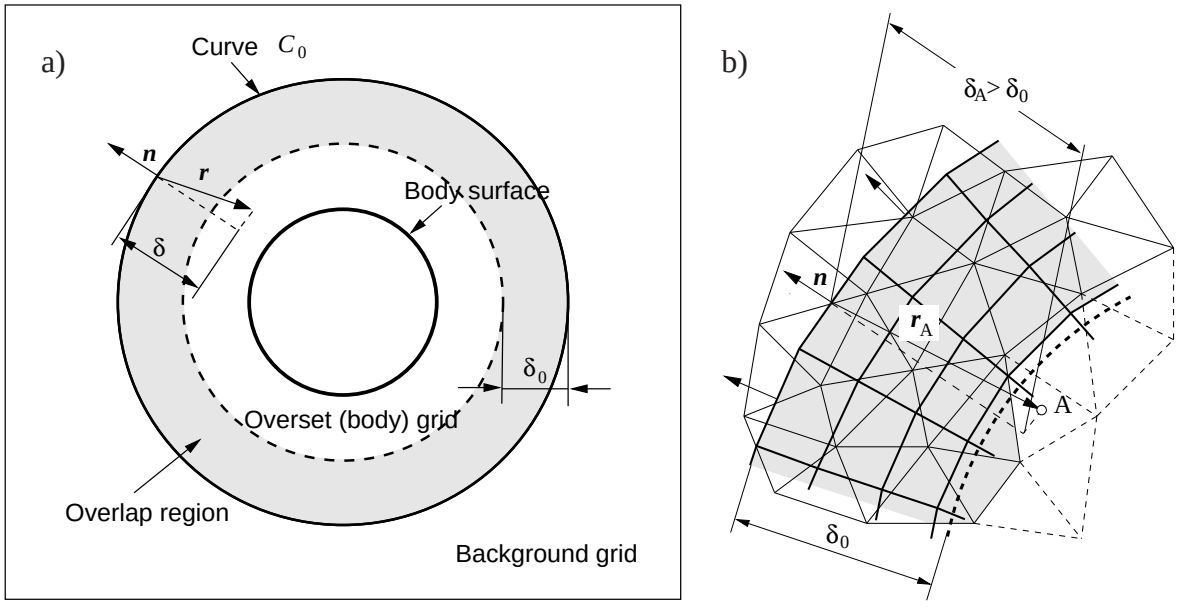


Figure 5.3: Hole cutting.

the overlapping region. It has to be large enough to provide sufficient overlap between the grids in order that the inter-grid coupling can be accomplished.

If there are more then one body grid (see figure 5.1), the procedure described above is applied to each body, resulting in as many hole regions in the background grid as there are bodies. If the overset body grids overlap each other (see figure 5.4), it is necessary to deactivate all regions on these grids which are outside the computational domain, i.e. covered by other bodies or lying outside the background grid. In addition, since the parts of each body grid may be covered by another overset grid, the holes may be created in these grids as well. Some criteria are necessary to decide in which grid the holes need to be created, and which grids are used to obtain the solution. To this end a grid hierarchy is introduced. The component grids are divided into

different levels. Grids on higher levels have higher priority then grids on lower levels. The background grid has usually the lowest level of priority, and in general finer grids have higher priority. Cells which need to be deactivated are from grids which lie underneath a grid with a higher priority. Figure 5.4 shows an overlapping grid system consisting of three component grids: a background grid and two grids created around bodies. Figure a) shows the situation when the grid 2 has a higher level of priority than the grid 3. Thus, in addition to the region outside the domain, also a part of the grid 3 covered by grid 2 is deactivated. Figure b) shows the opposite situation, the grid 3 is of higher priority than the grid 2. In this case an additional hole is created in grid 2. Grid levels are specified during the grid generation, i.e. before beginning of the computation.

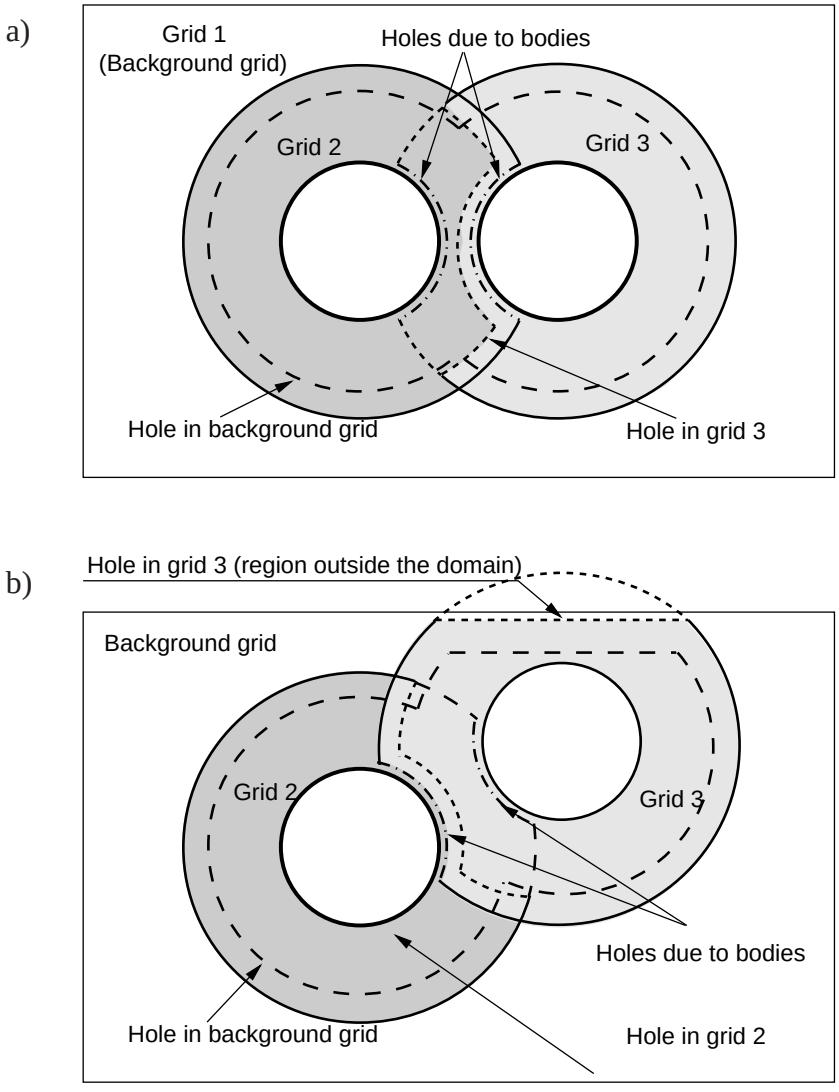


Figure 5.4: Multiple overlaps with different grid hierarchy: a) grid 2 has a higher level of priority then grid 3 and b) grid 3 has a higher level of priority then grid 2; some parts of grid 3 lie outside the background grid and are deactivated.

In cases that involve moving bodies[1] it may be difficult to define the grid hierarchy correctly. Since the relative position between body grids that overlap each other changes, the grid hierarchy defined for the starting position may not be valid all the time. In such cases only the parts of each overset body grid that lie outside the domain are deactivated, while the regions covered by other overset grids are retained. This results in more active cells then necessary, since in some regions the solution is computed on more then one grid. In most cases these regions may be expected to be small in comparison to the whole grid, thus the computational overhead should not be significant.

For computations on overlapping grid, especially on moving grids, it is important that the hole cutting can be performed automatically. For complex configurations this may become difficult and more general algorithms may be necessary to accomplish an automatic hole cutting. The research of the present study was basically focused on the solution of governing equations on overlapping grids and less attention has been paid to the development of algorithms for creation of overlapping grids. Some generalized algorithms for creation of hole cuttings on structured [15, 60, 79, 49] as well as unstructured [55] grids can be found in literature. With some modifications, these strategies may also be applied to general unstructured cell-centered grids.

Donor searching

For each interpolation cell, donor cells on the grid that overlaps it have to be identified. How many cells are needed and which relation between them is allowed, depends on the interpolation scheme. The simplest method requires that only one cell whose centroid is closest to the centroid of the interpolation cell should be identified. This cell is called here the host cell. Any additional node contributing to the interpolation formula must come from the immediate neighborhood of the host cell.

There are numerous ways to find host cells for a given point. The simplest one is the brute-force algorithm, which assumes a search over the entire grid until the host cell is found. For a considerable number of interpolation points, and for fine grids, this procedure becomes prohibitively expensive, since a large number of searches has to be performed.

In the present study the neighbor-to-neighbor searching algorithm [42], suitable for unstructured grids, is employed to accomplish this task. The method is schematically shown in figure 5.5. Starting from a given cell, one jumps to the neighboring cell that lies in the direction of the target. This procedure is repeated until the cell which contains the target point is reached. The selection of the next starting cell among the neighbor cells of the current starting cell is based on the scalar product of the vector $\mathbf{p}_j$, connecting the midpoint of each cell face and the target point, with the outward normal $\mathbf{n}_j$ on that cell face (see figure 5.5 b). The cell face whose normal encloses the smallest angle with vector $\mathbf{p}_j$ is selected and the neighboring cell that shares this face with the current starting cell is chosen as the new starting cell. If scalar products $\mathbf{p}_j \cdot \mathbf{n}_j$ are negative for all cell faces, the target point lies inside that cell, i.e. the host point is found.

This searching algorithm is very efficient, since the searching path is one-dimensional even on a three-dimensional grid made of arbitrary polyhedral cells. It performs very well inside the

[1] An example is presented in section 6.5.

domain but can have problems with boundaries. This is illustrated in figure 5.5 a). If the cell B is used as the starting cell, the algorithm fails since it cannot 'jump' over the boundary. In such cases, the next starting cell is the one which lies near the boundary cell face (cell D in figure 5.5 a) that intersects with the line connecting the last starting cell C and target point P. Since there can exist more than one intersection, the cell which is closest to the target point is selected as the new starting cell. Alternatively if boundary face crossed in the preceding step are excluded from the check described above, the algorithm will still work and force the search path to go around the obstacle.

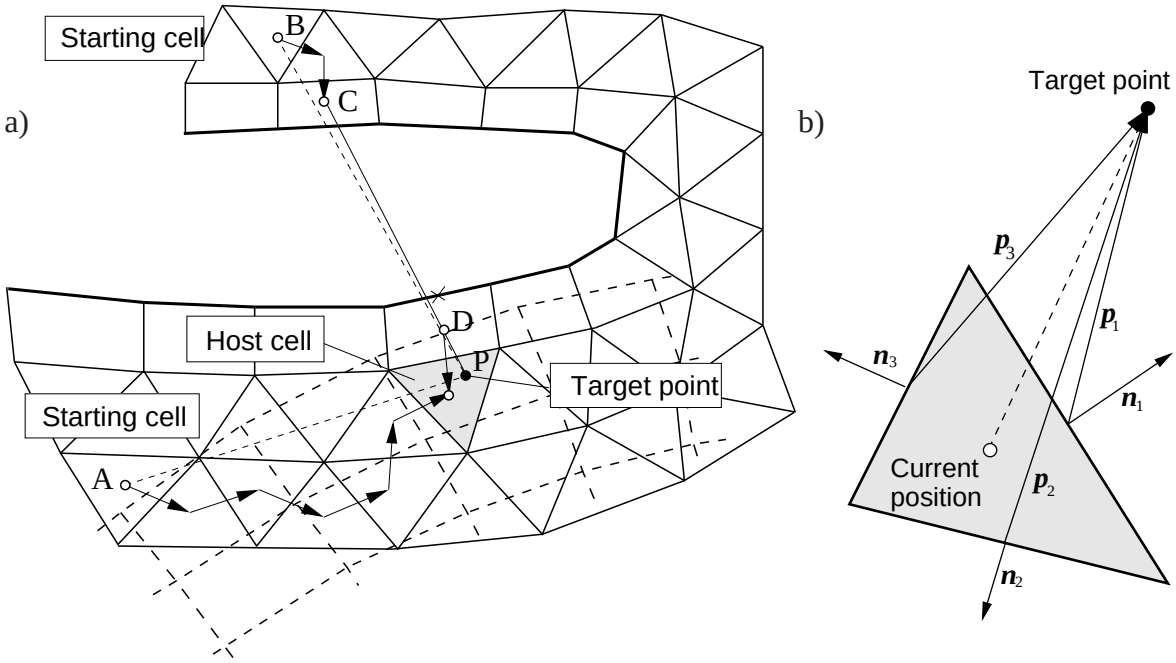


Figure 5.5: Neighbor-to-neighbor searching.

For the moving-body problems, the donor searching for new position is significantly simplified, since the existing overlapping definition can be used as the initial guess for the next search. Since the body movement is small within a time step, the new host cell for each interpolation cell lies in the immediate neighborhood of the previous donor cell. Similarly, any new interpolation cell comes from the neighborhood of an old interpolation cell, which is used to determine the starting guess for the new search.

Note that the technique described above, used to determine whether the target point lies within a cell, can be applied to three-dimensional grids made of arbitrary polyhedra as well. However, the algorithm works only for convex cells, which is assumed to be always the case in this study. Since this condition may not always be fulfilled for arbitrary unstructured grids, some other techniques might be necessary to localize the host cell. In such cases the Shimarat's algorithm [52] can be used, which works well for concave cells too.

Construction of interpolation stencils

Once the host cells for each interpolation point are found, the interpolation stencils are generated. The number of points included in interpolation stencils and their form depends on the interpolation functions used. In general, any kind of shape functions can be used to construct interpolation formulas, but the most often used are linear shape functions [38, 6, 81, 44, 65, 86]. Higher order shape functions can also be used, but one has to be careful, since approximations of higher order may introduce oscillations in the solution [57]. This especially may happen when the overlapping region is placed in the area with strong gradients in the flow field. Alternatively, higher order monotonicity-preserving interpolation methods can be used.

The quality of grid coupling may also depend on the quality of interpolation stencils. Therefore some attention needs to be paid when choosing the donor points contributing to interpolation stencils. For construction of interpolation stencils, the host cell is used as the starting point. Additional donors are sought in the neighborhood of the host cell. Some examples of possible interpolation stencils consisting of different number of donors and constructed on grids of different topology are shown in figure 5.6. Figure a) shows how the interpolation stencils consisting of three or four donors can be constructed when a regular quadrilateral grid is the donor grid. Figure b) shows an irregular triangular mesh as the donor grid. There are a few options to construct an interpolation stencil consisting of three donors. The first possibility would be to construct a stencil using the cell D_1 and its nearest neighbors D_2 and D_3. However, since the target point P_i is not enclosed in the triangle $D_1D_2D_3$, this stencil would produce some negative weight coefficients, i.e. an extrapolation would be obtained instead of an interpolation. Although, such a stencil would also work, it is better to avoid such a situation whenever possible, since negative weight coefficients can, in some circumstances, produce non-physical results [57]. A "regular" stencil can be created using, for example, cells D_1, D_3 and D_4. Another interpolation stencil consisting of four donors can be constructed using cells D_1, D_2, D_3 and D_4, while the cell D_5 can be added to construct a five-point stencil.

After all donors are determined, an interpolation formula for each interpolation point can be constructed which, regardless of the shape function used, has the following form:

$$\phi_{P_i} = \sum_{k=1}^{N_D} \alpha_{w_k} \, \phi_{D_k} \, , \tag{5.1}$$

expressing the interpolated function value at node P_i, ϕ_{P_i}, in terms of the function values at donor points D_k, ϕ_{D_k}, with α_{w_k} being the interpolation weights.

5.2.2 Accuracy and conservation properties of interpolation

Conditions at grid interfaces, by which solutions of different sub-domains are connected with each other, are crucial in calculations on overlapping grids. A good interface treatment should lead to a converging iterative scheme and a solution of acceptable accuracy. An important issue with regard to the interpolation is its effect on the overall accuracy of the solution and its conservation properties. The form of inter-grid interpolation described in the previous section, based on interpolation of dependent variables at overlapping interfaces, is not necessarily conservative,

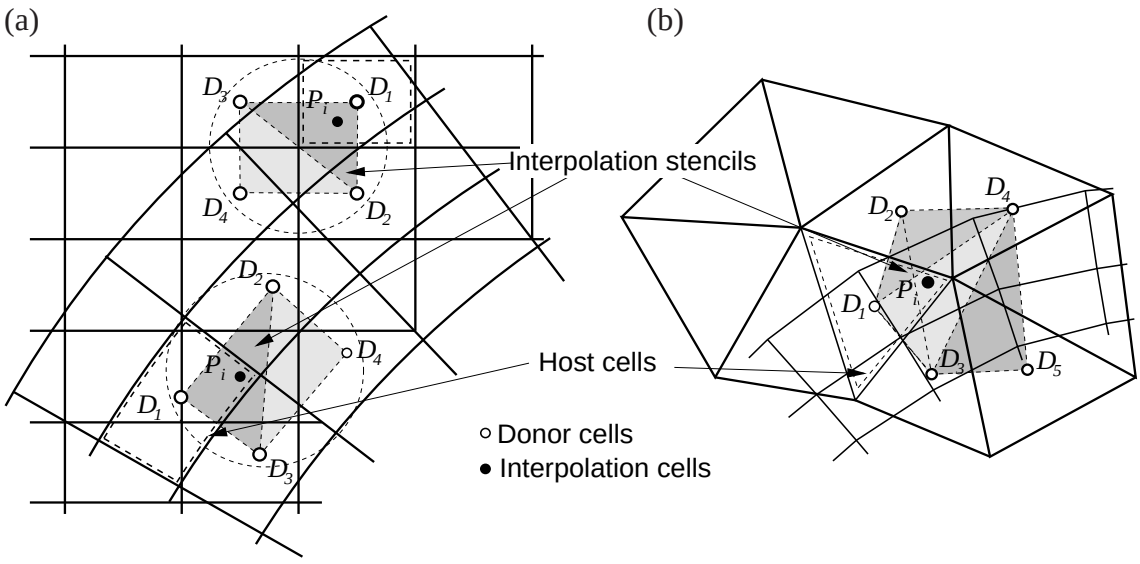


Figure 5.6: Some variants of interpolation stencils for different types of interpolation formulas and different types of grid.

since the fluxes between component grids are not balanced explicitly. Thus, although the numerical discretization in all sub-domains may be conservative, numerical results are not conservative globally. However, in many applications it was shown that this lack of conservation does not degrade the solutions when the overlapping interfaces are placed in regions of smooth solutions. Successful applications of the methods with non-conservative interpolation have been reported in literature [6, 63, 81, 12].

Meakin [48] investigated the spatial and temporal accuracy of the overlapping method for moving body problems and suggested that the issue with interface treatment is not necessarily one of conservative versus nonconservative interpolation, but one of grid resolution.

The nonconservative approach was adopted in the present study mainly due to its simplicity, which is especially advantageous for the dynamic overlapping grids. It could be demonstrated that the non-conservative errors are of the same order as other discretization errors, and that they tend to zero as the mesh is refined. In the present study the major concern is the conservation of mass, which may be essential for the solution of the pressure-correction equation. This issue is discussed later in section 5.3.3, where some options for handling the problem are proposed.

Although non-conservative interpolation gives satisfactory results in most cases, conservative treatment of grid interfaces might be necessary when the overlapping interface is placed in regions with strong changes in flow field or when discontinuities such as shocks pass through the interface [57]. The subject of conservative inter-grid interpolation is a very complex one and beyond the scope of this study. Detailed discussions on this topic and some options for handling of grid interfaces conservatively can be found in references [9, 16, 90].

Besides the type of interpolation at overlapping interfaces, the accuracy of the solution on overlapping grids can also be affected by the order of accuracy of interpolation formulas [15]. Linear interpolation used in the present study was found to be a good compromise between

simplicity ad accuracy, as the test computations presented in chapter 6 show. The accuracy of the computation can also be degraded when the size of grid cells on two grids that overlap width each other is significantly different. This is due to disparate abilities of a coarse grid to resolve the flow features in comparison to a finer grid. Therefore, it is advisable to avoid such a situation whenever possible and try to provide as close grid spacing in the overlapping zone on all grids overlapping each other as possible.

5.3 Solving the governing equations on overlapping grids

In order to solve the governing equations on overlapping grids, two basic changes in the solution procedure are necessary, which

- take into account the presence of holes in the grid,
- provide appropriate coupling between the grid components in an overlapping grid system.

5.3.1 Treatment of inactive cells

As described above, those cells in the overlapping grid system, which are not needed in the computation, are marked as inactive. For these cells a special treatment is required in order to prevent their influence on the overall results. Owing to the use of unstructured grids, the inactive cells can be removed from the grid already during the grid generation. This would somewhat complicate the grid generation, but on the other hand it would facilitate the numerical solution since no extra work would have to be done due to inactive cells. Furthermore, the total number of cells in the grid would also be reduced. Although this approach obviously has some advantages, it is applicable only to stationary grids and possibly to the dynamic grids with very small relative motion of the component grids, which can be carried out within the initial overlapping regions. In the present study we are primarily interested in problems which involve an arbitrary large-scale body motion. In order to handle such problems properly, it is necessary to allow the overlapping regions to change in time according to body motion. Therefore, all cells in the grid are retained and they are allowed to change their status during the computation. This means that some cells, which are for some time outside of the computational domain (e.g. covered by a body), are temporarily declared as inactive and thus excluded from the computation. Some time later these cells may need to be reactivated and used in the computation again as they re-enter the computational domain while some other cells, which in meantime have left the computational domain (e.g. have been covered by a body), need to be deactivated.

To prevent the governing equations from being solved at inactive cells the following modifications are made to the discrete equations (3.23) at these cells:

- the off-diagonal coefficients $a_{j\phi}$ are set to zero,
- the central coefficients $a_{P_0\phi}$ are set to unity,

- the source terms b_{P_0} are assigned the values of the dependent variables at the cell-center.

This provides that the dependent variables do not change their values at these cells as long as they do not change their status, i.e. as long as they remain passive. The information about cell status is stored in a single array i_b which has a value for each cell on every component grid and is defined as:

$$i_b = \begin{cases} 1, & \text{for discretization cells} \\ 0, & \text{for inactive and interpolation cells} \end{cases} \tag{5.2}$$

The above-mentioned modifications of the matrix coefficients and source terms are achieved by making use of the array i_b as follows:

$$\begin{aligned} a_j &= a_j \cdot i_b, \\ a_{P_0} &= a_{P_0} \cdot i_b + (1 - i_b), \\ b_{P_0} &= b_{P_0} \cdot i_b + (1 - i_b) \cdot \phi_{P_0}. \end{aligned} \tag{5.3}$$

To avoid any inconsistencies in estimation of the convergence rate, the residual norm defined by equation (3.41) is also appropriately modified:

$$R_\phi^k = \frac{\sum_{i=1}^{N} |b_{P_0} - a_{P_0\phi}\phi_{P_0}\, i_b + \sum_{j=1}^{N_j} a_j \phi_{P_j}\, i_b|}{\sum_{j=1}^{N_j} |a_{P_0\phi}\phi_{P_0}\, i_b|}. \tag{5.4}$$

For the linear equation solver no further changes are required.

In the computer code created during the present study, the discretization is first completed for all cells in the usual way as for single grids. Although this in principle does not have to be done for inactive cells, introducing additional checking for which parts of the computational domain the discretization needs to be made and which cells could be omitted would lead to additional complications of the programming. Since the number of hole cells is in most cases small in comparison to the total number of cells, the computational overhead is not significant. Thus, the main part of the program remains the same for single and overlapping grids. Changes necessary to accommodate the overlapping grids are introduced as extra modules and are used only if required by calling additional routines. As an alternative to this practice, one could renumber the cells and move the passive ones to the end of the list so that loops in the code run only over active cells. This can be complicated for moving grids and has not been tried in this study. Similar effect can be achieved by introducing an additional pointer array, which contains the indices of active cells. In such an approach, the loops would run over this array, providing that only active cells are included in the discretization.

5.3.2 Coupling of the solution on overlapping grids

Inter-grid communication plays a key role in the numerical solution on overlapping grids. The accuracy and efficiency of the numerical solution depends strongly on the way the inter-grid communication is integrated into the solution procedure.

Usually, the overlapping interfaces are considered as additional non-physical boundaries at which the boundary conditions are imposed by interpolating the solutions from another grid [82, 24, 64]. The equations may then be solved for each grid separately, and the boundary conditions are updated in each iteration. This procedure, however, couples the grids only weakly. Since the information exchange between the grids is lagged by an outer iteration, more iterations and stronger under-relaxation may be required to achieve a converged solution. Another difficulty of such an approach is to appropriately couple the velocity and the pressure fields on all grids. The strategy to impose boundary conditions at interfaces by interpolation of dependent variables can be applied directly to the momentum equation (Dirichlet boundary condition). Since the velocity is known, the mass fluxes through the cell faces along interfaces can be computed and taken as the boundary condition for the pressure-correction equation. Hence, the boundary conditions in the pressure-correction equation are of the Neumann type. Therefore, one has to ensure global mass conservation for each grid in order to be able to solve the pressure-correction equation with Neuman conditions on all boundaries when the flow is incompressible. This leads to the pressure fields on different grids independent of one another. To obtain a consistent pressure field in the entire flow domain, the reference pressure on each component grid needs to be corrected in such a way that the pressure levels on all grids are compatible with each other. This correction may be determined from the pressure difference on different grids at some part of the interface that is common for two or more adjacent grids overlapping with each other at that place. This, however, provides in general only local compatibility between the pressure fields in the regions where the corrections have been determined. In the rest of the overlapping interface, the discontinuities in the pressure field may still appear. This is illustrated in figure 5.7 which shows the resulting pressure field obtained on stationary overlapping grids, using the technique for a weak inter-grid coupling as described above. In the regions where the pressure level was corrected, the agreement between the pressure fields on different grids is good, while for the rest of the interface the discontinuities still can be observed. The problems concerning the inter-grid coupling, when the solution is sought on each grid separately, may be more pronounced on dynamic grids, which may lead to inaccurate results, or cause the divergence of the computation.

Having all this in mind, it is important to consider the solution on all grids simultaneously, i.e. to achieve a strong coupling. The focus of the present study has been the development of such a method, which besides the strong inter-grid coupling also provides the unique and smooth solution over the whole computational domain. A simultaneous solution for all component grids is necessary to achieve this goal. Thereby, the integration of the interpolation procedure for inter-grid coupling into the global solution algorithm is of primary importance for the efficiency and accuracy of the numerical method.

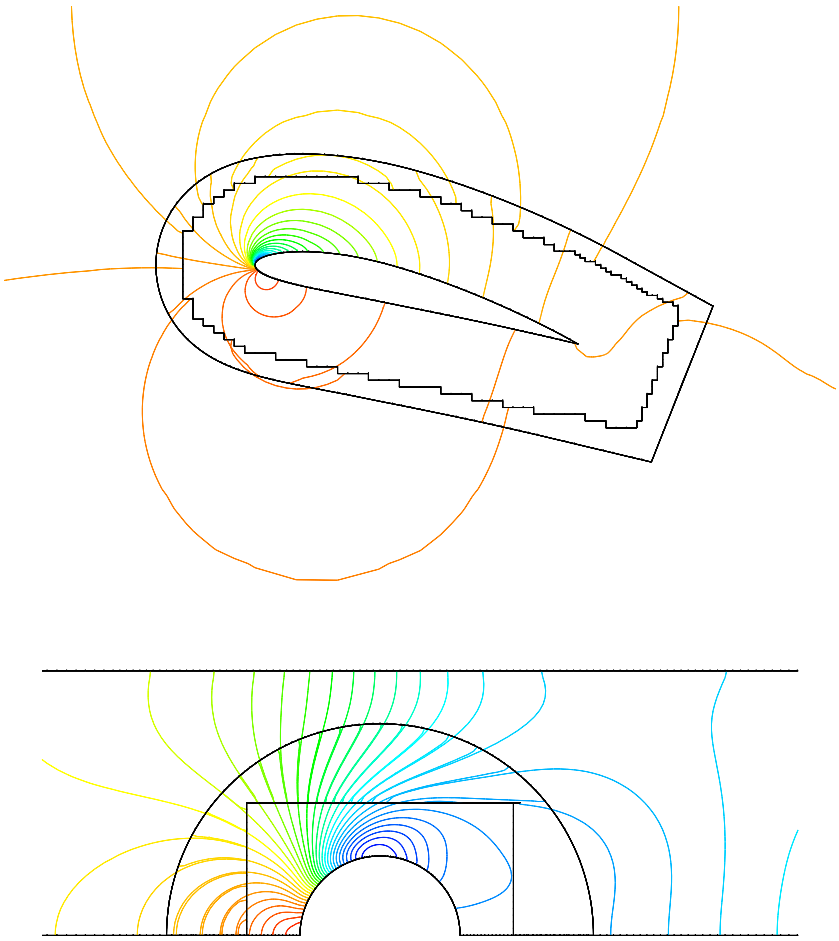


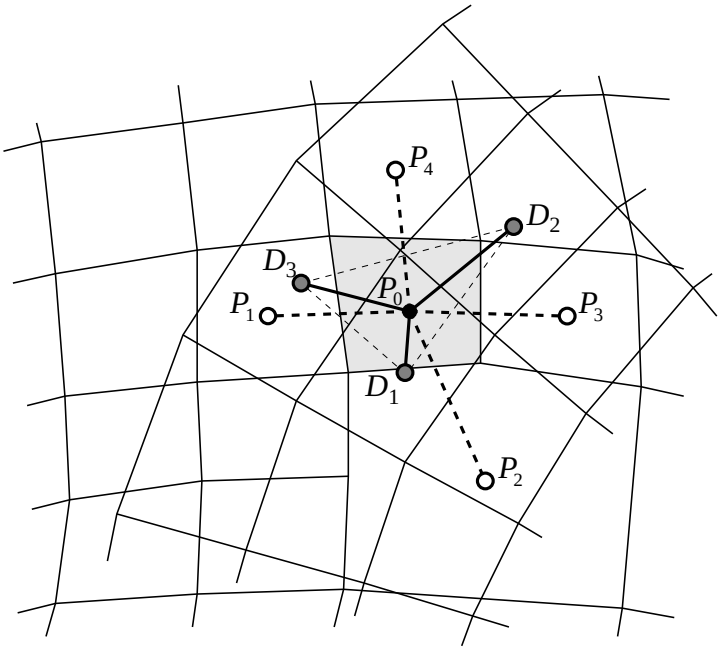
Figure 5.7: Pressure field around an airfoil (upper) and a cylinder (lower) obtained using a weak inter-grid coupling.

In the present study, a method for an implicit treatment of overlapping interfaces has been developed. The basic idea of the method is to solve a single linear equation system for all grid components simultaneously. The interpolation equations that express the variable values at interpolation cells in terms of a number of donor cells[2] from another grid are also involved into the linear equation system (3.39) and solved simultaneously with linearized equations (3.23) that arise from discretization. That means that a corresponding discrete equation for each interpolation cell P_0 (see figure 5.8) is replaced by an interpolation equation (5.1), which is for this purpose rearranged into:

$$\phi_{P_0} - \sum_{k=1}^{N_k} \alpha_k \, \phi_{D_k} = 0, \tag{5.5}$$

[2]Depending on the interpolation formulas.

with α_k being the interpolation weighting factors and ϕ_{D_k} the values of the dependent variable ϕ at donor cells D_k. This is illustrated in figure 5.8, which shows computational molecules for the cell around P_0 when it is one of discretization cells (dashed lines) and when it is an interpolation cell (full lines), with corresponding algebraic equations.



$$\underbrace{a_{P_0}\phi_{P_0} - \sum_{j=1}^{N_j} a_j \phi_{P_j} = b_{P_0}}_{\Downarrow}$$

$$\phi_{P_0} - \sum_{k=1}^{N_k} \alpha_k \, \phi_{D_k} = 0$$

Figure 5.8: Computational molecule of a cell for the case when it is a discretization cell (dashed lines) and for the case when it is an interpolation cell (full lines) with corresponding algebraic equations.

Figure 5.9 shows a one-dimensional overlapping grid with resulting (modified) coefficient matrix representing the linear equation system to be solved. The overlapping grid system consists of two grids. Grid 1 is in the system matrix represented by the upper left block, while the lower right block of the matrix represents the grid 2. The communication between the grids is achieved via the interpolation cells 7 and 11 whose values are obtained from interpolation equations. In the system matrix, this communication is achieved via the off-diagonal coefficients α_{ij}, which multiply the variable values from the cells on another grid. For inactive cells 4 and 15 the off-diagonal coefficients are zero and as such they are decoupled from the rest of cells.

Involvement of the interpolation equations in the linear equation system for the whole grid, ensures that the solution is simultaneously updated on all grids at the same time without any prolongation. This provides a strong inter-grid coupling and promotes the convergence, which, according to test computations presented in chapter 6, remains in the same range as on comparable single grids. Furthermore, a unique solution (for all variables) over the whole domain is achieved. This is illustrated in figure 5.10, which shows the pressure field around a NACA airfoil and a cylinder. In contrast to the results obtained using a weak coupling shown in figure 5.7, the pressure field is now smooth in the complete overlapping region.

Further assessment of the method concerning the overall accuracy, conservation errors and

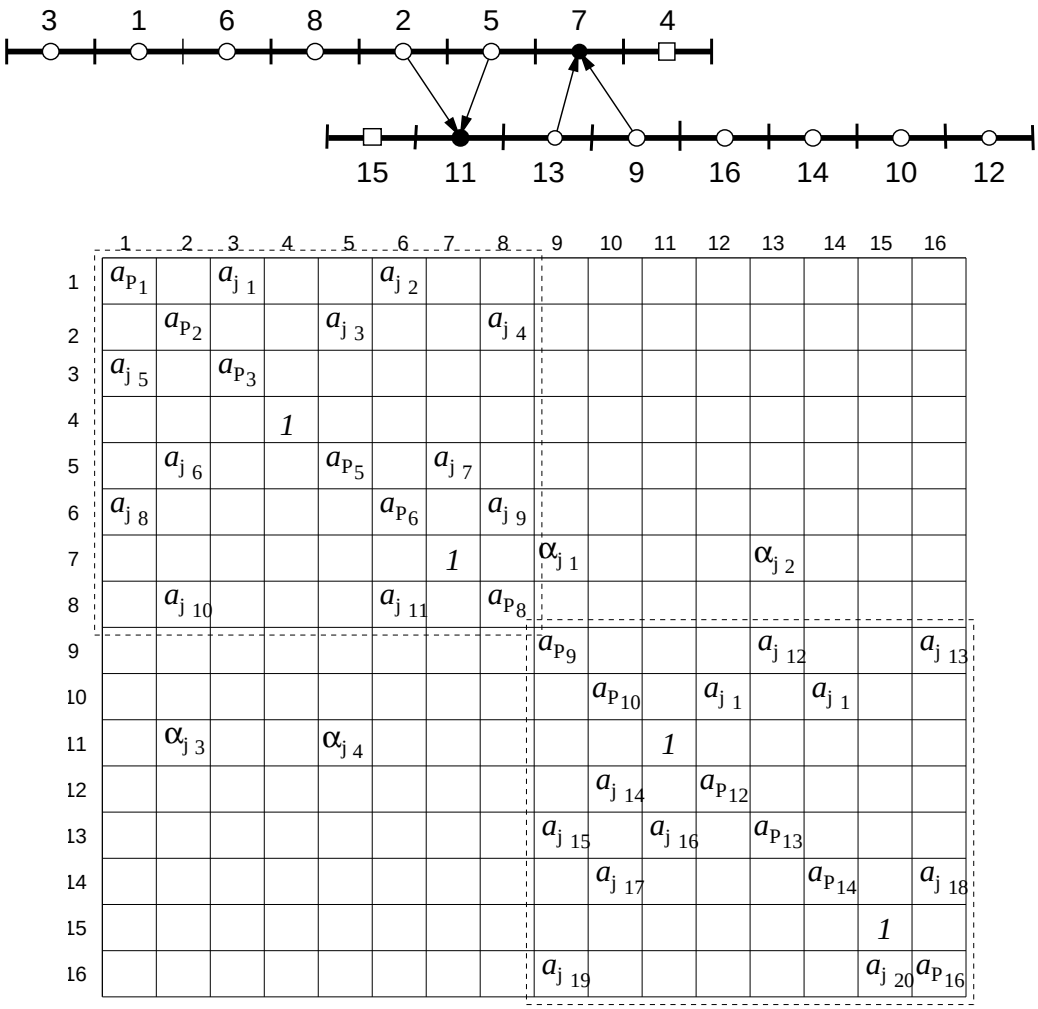


Figure 5.9: Structure of the linear equation system matrix for a one dimensional overlapping grid composed of two component grids.

applicability of the method to complex problems involving multiple moving bodies is provided in chapter 6.

5.3.3 Mass conservation

The interpolation of dependent variables at interpolation nodes is satisfactory as the interface treatment for the momentum and other transport equations for scalar quantities. Experience with computations has shown that the lack of conservation does not degrade the solution nor causes other computational problems when solving these equations.

On the other hand, the pressure-correction equation deserves a special attention, since the strict conservation of mass may be essential for the solution of this equation. When the pressure is prescribed at some parts of the boundary, i.e. the pressure boundary condition is applied, the pressure-correction equation can be solved irrespective of the interpolation formula used, since

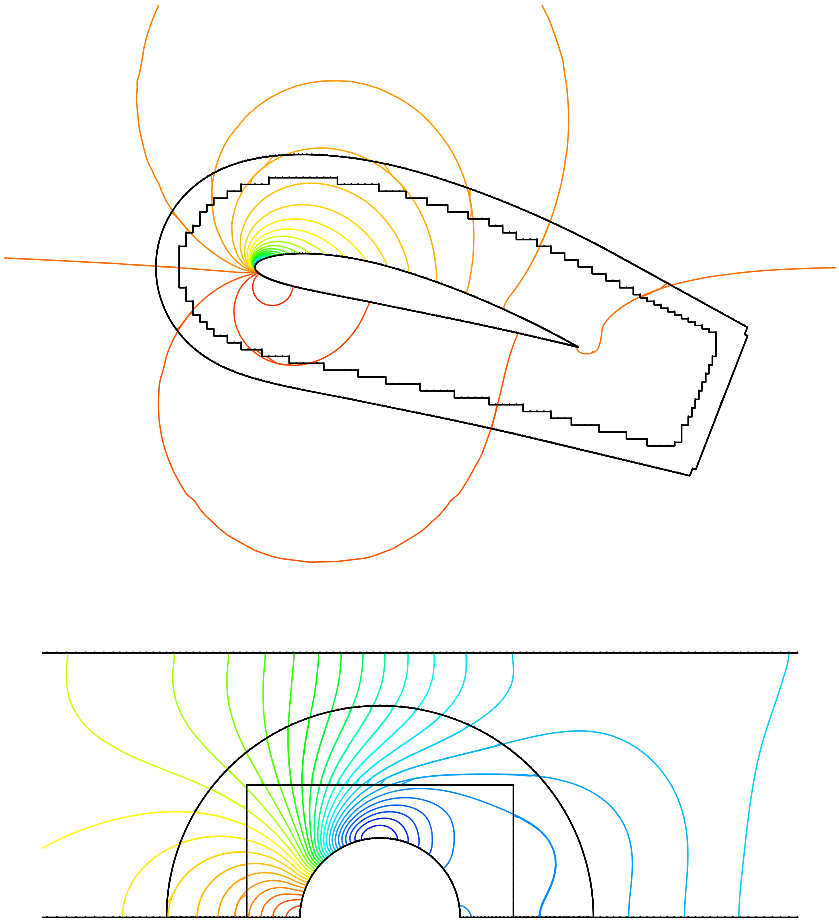


Figure 5.10: Pressure field around an airfoil upper and a cylinder lower obtained using a strong inter-grid coupling.

any interpolation error will be adjusted by the pressure-correction equation when computing mass fluxes through boundary faces where pressure is prescribed. On the other hand, when the velocities are prescribed at all boundaries, the Neumann boundary condition (zero gradient) applies for the pressure-correction equation at all boundaries. In such a case, the resulting linear equation system for the pressure-correction is singular. The equations are linearly dependent and the pressure-correction can be determined only up to a constant. To remove the singularity one of the equations is replaced by an equation that sets the level of the pressure, i.e. the pressure level is specified at one cell. In addition, in order for such a system to be solved and meaningful results to be obtained, the system of equations has to be consistent. This condition is satisfied for incompressible flows if the right-hand sides of all equations sum to zero. The situation can be linked to a steady-state heat conduction problem with heat sources and adiabatic boundaries. For such a problem, a steady-state solution exists obviously only if the sum of the source terms is zero. If there is a net heat source, then the temperature inside the domain will simply rise

without bound if an iterative solution scheme is used. In the pressure-correction equation (see equation (3.29)), the source term for a given cell represents the sum of mass fluxes across all cell faces of that cell. Since the discretization method is conservative, the mass fluxes through inner cell-faces cancel out when all source terms are summed up and to enforce the mass conservation, the sum of mass fluxes over all boundary faces has to be equal to zero. In section 3.4.2 it has been explained how the mass conservation is enforced for problems with open boundaries, when the standard outlet boundary condition based on extrapolation of dependent variables is applied at outflow regions. For the computations on overlapping grids, additional treatment is necessary at overlapping interfaces to enforce the global mass conservation. The interface is not the outer boundary of the overlapping grid; it is made by faces separating interpolation cells from discretization cells and is defined in the same way for both overlapping and background grid. The mass fluxes through cell faces along interfaces are computed as (see figure 5.11):

$$\dot{m}_i = \rho \mathbf{v}_i^* \cdot \mathbf{s}_i \,, \tag{5.6}$$

where the velocity $\mathbf{v}_i^*$ is obtained from equation (3.25) as for internal cell faces except that in this case the values from an active and an interpolation cell are involved in the computation of $\mathbf{v}_i^*$. Like the boundary fluxes, the interface fluxes contribute only to one cell and to ensure that the mass conservation is satisfied, their sum must be zero, i.e. the total mass flux across all interfaces must be equal to zero. However, due to non-conservative nature of the inter-grid interpolation, this is in general not the case. As result, a mass imbalance is obtained:

$$\Delta \dot{M}_{\text{olg}} = \sum \dot{m}_i, \tag{5.7}$$

which appears as a net mass source in the pressure-correction equation and has to be eliminated.

There is no unique way to achieve this constraint. Actually it is not necessary to strictly enforce the mass conservation in the usual sense. According to the solvability condition mentioned above, the obvious procedure to provide for the discrete system of equations (3.30) to be consistent is to add a fraction of the mass source $\Delta \dot{M}_{\text{olg}}$ to the right hand side of each equation such that the sum of the new right hand sides is zero. Thus, the right hand side of every equation in (3.30) may be re-computed as:

$$b_{p'}^{\text{cor}} = b_{p'} + \beta_s \, \Delta \dot{M}_{\text{olg}}, \tag{5.8}$$

where β_s is the corresponding factor defining the fraction of the mass source that needs to be added to each cell. In order to distribute the mass source continuously over the domain on non-uniform grids, the factor β_s can be computed for each cell proportional to cell volume.

This adjustment of the right hand sides provides conditions for appropriate solution of the pressure correction equation. However, it introduces an error into each equation characterized by a local mass source, which leads to a non-zero divergence in each cell[3]. This error is usually

[3]The continuity equation is not satisfied locally due to presence of a mass source in each cell.

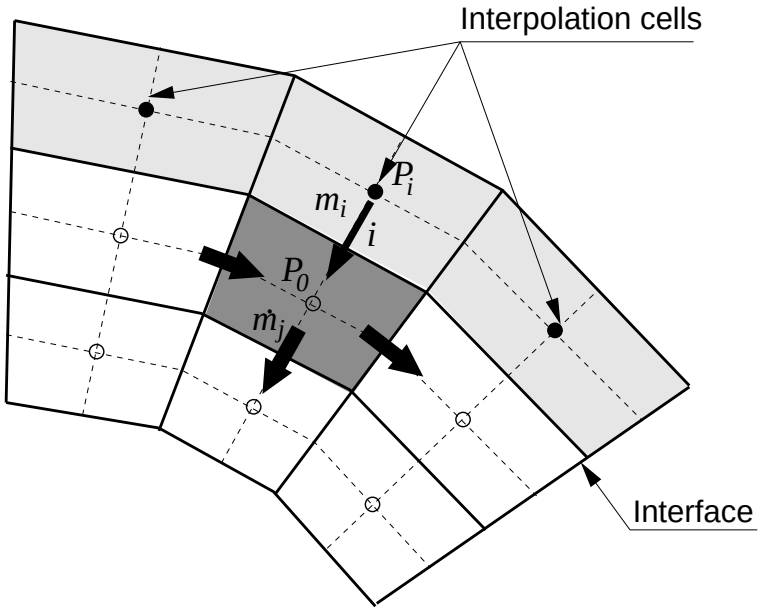


Figure 5.11: Mass balance for a near-interface cell.

very small and does not affect the solution.

Alternatively, the mass conservation can be enforced by an adjustment of interface fluxes (5.6) such that their sum (5.7) is zero. To explain how it can be done consider an overlapping grid system comprised of two grid components shown in figure 5.12. For the sake of clarity, the grid components are shown separately. In addition, the interface fluxes on grid 1 flow out of the domain (outflow fluxes) and on grid 2 they flow into the domain (inflow fluxes)[4]. Note that in the summation (5.6) outflow fluxes are considered as positive and inflow fluxes as negative. The adjustment of interface fluxes is achieved by subtracting a part of mass source $\Delta\dot{M}_{\text{olg}}$ from each flux $\dot{m}_i$ as follows:

$$\dot{m}_i^{\text{cor}} = \dot{m}_i - \beta_i \Delta\dot{M}_{\text{olg}} \,, \tag{5.9}$$

where the factor β_i is proportional to the share of the mass flux $\dot{m}_i$ in the total mass flux through the overlapping interface $\dot{M}_{\text{olg}} = \sum |\dot{m}_i|$. The proportionality factor is thus computed as:

$$\beta_i = \frac{|\dot{m}_i|}{\dot{M}_{\text{olg}}} \,. \tag{5.10}$$

The correction defined by equation (5.9) ensures that the sum (5.7) is zero and, since there is no contribution from the interfaces, no adjustment is required for the right hand sides in equation system (5.7). Hence, there are no artificial mass sources and the continuity equation is satisfied

[4]This is a special case, in general on each grid inflow as well as outflow fluxes may appear.

in each cell, i.e a divergence-free flow field is obtained. Note that the overlapping interfaces are treated for the entire domain rather then for each zone separately. As a consequence of such a treatment, the mass conservation is enforced only globally, for the whole domain. It is not ensured that the mass conservation is also fulfilled for each grid. This, however, has no consequence for the solution, since the equations are solved for all grids simultaneously and not for each grid separately. Therefore, it is not necessarily to strictly enforce the mass conservation for each grid. The test computations have shown that the mass conservation is nearly achieved also on each grid as a converged solution is approached (see section 6.2).

Since the conservative corrections described above have to be made in each outer iteration, the indices of the cell-faces which compose the overlapping interface are stored to speed up the access to them. Unlike the boundary faces, the cell faces along interfaces are not necessarily oriented outwards, so their orientation is also checked and appropriately stored in an array, which has a value of 1 if the cell face is oriented outward and -1 if the cell face is oriented inward.

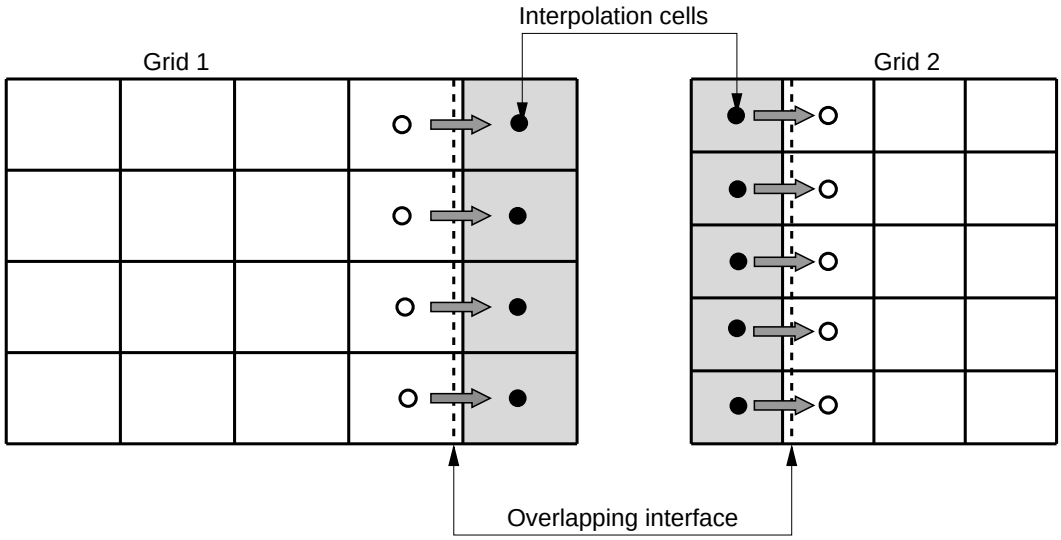


Figure 5.12: On the mass conservation at overlapping interfaces.

Both methodologies for conservative treatment of overlapping interfaces described above have been implemented and tested. However, since the first method based on the adjustment of the source terms in the pressure-correction equation introduces an error, the second method based on the adjustment of mass fluxes through the overlapping interfaces is preferred in the present study. In the computations presented in chapter 6, if it has not been explicitly emphasized, the second method was used wherever it was required. The effects of the conservative treatment of overlapping interfaces is assessed in section 6.2.

5.3.4 Cell-face values at interfaces

One layer of interpolation cells separating the active and passive part of the grid, provides the values of dependent variables at these cells, which are involved in the computational molecules of the adjacent active cells and as such treated implicitly. However, the values of dependent variables alone at interpolation cells are not enough to provide all information necessary for appropriate approximation of fluxes through the cell faces along interfaces.

As an example consider the diffusion flux, which is approximated by equation (3.10). This approximation involves the normal derivative of the dependent variable at the cell face. For the situation presented in figure 5.13, the normal derivative at the cell face i between an active cell P_0 and an interpolation cell P_i, is approximated as:

$$\left(\frac{\partial\phi}{\partial n}\right)_i \approx \frac{\phi_{P_i} - \phi_{P_0}}{|\mathbf{d}_i|} - (\text{grad}\,\phi)_i^{\text{old}} \cdot \left(\frac{\mathbf{d}_j}{|\mathbf{d}_j|} - \frac{\mathbf{s}_j}{|\mathbf{s}_j|}\right). \tag{5.11}$$

The second term on the right hand side in equation (5.11) contains the term $(\text{grad}\,\phi)_i^{\text{old}}$, which is obtained by interpolation of gradients of the dependent variable at adjacent cells P_0 and P_i. These gradients are computed explicitly using either equation (3.12) based on Gauss' theorem or equation (3.14) based on least square method. Whichever approximation is used, for the computation of gradients at the cell-center, the values of the dependent variable at neighboring cells are required. Thus, in the computation of gradient at cell P_i in figure 5.13, two inactive cells are involved. Since the values at inactive cells are, in general, not available, the computed gradients may be inaccurate and an error may be introduced in the solution in the area around interfaces. One possibility to remedy the problem would be to introduce a second layer of interpolation cells [71], which would provide that the neighbors of the first layer of interpolation cells are either active or interpolation cells. Thus, the variable values required for the computations of gradients would be available. This is, however, impractical since unnecessarily many more interpolation cells would have to be used[5], requiring more computational effort for searching of donor cells as well as more memory for data storage. Instead of this, in the present study the values of the gradients at interpolation cells are supplied by interpolation from another grid, using the same interpolation formula as for the variable values. Since the donor points and interpolation coefficients are available, and the interpolation needs to be performed only once within an outer iteration, the additional computational effort is minor. Alternatively, the gradients could also be extrapolated from the interior from the same grid. However, the former method has shown better performance and was adopted in the present study.

The same strategy is applied to convection fluxes - which depending on the approximations used - may require a similar treatment (see equation (3.7)). In general, any term whose approximation requires more information than available from the solution of the equations is supplied by interpolation from another grid.

[5] Approximately twice as many cells as required with one layer.

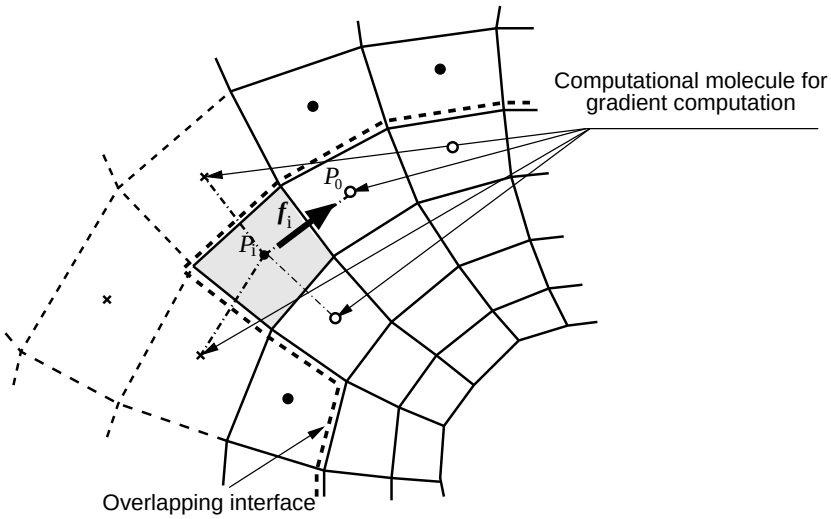


Figure 5.13: Computation of fluxes through interface cell faces.

5.3.5 Treatment of moving overlapping grids

When the moving overlapping grids are used, some additional issues need to be taken into consideration in order to solve the equations properly. A special attention has to be paid to the unsteady term in the governing equations. Since this term involves the values of dependent variables from one or two previous time steps, it is has to be ensured that these values are also available at all currently active cells. Since the cells change their status during the computation, this may not be always ensured. Care has to be taken to provide that the unsteady term is approximated properly also at those cells, which switch from inactive to active within a time step. Since the values of dependent variables at inactive cells are not updated during the solution, the ”old” values at the cells, which switch from inactive to active are not available. Without further treatment this may lead to an error. Therefore, such a situation has either to be avoided by choosing a smaller time step, or the required values need to be provided in another way (e.g. by interpolation). For cells which switch from interpolation to active this is not a problem, since the values at interpolation cells are also updated during the solution.

Some special features of overlapping grids associated with moving bodies can be exploited in the computation on moving grids. Since only the grids attached to bodies move, modifications to the governing equations due to grid movement need to be applied only to this part of the grid, which is in this case easy to identify. Furthermore, the grids which are moving usually do not deform. This also can be exploited in re-computation of geometrical properties of the grid.

5.4 Solution procedure

The sequence of steps required to simulate flows around moving bodies using overlapping grid technique is shown in figure 5.14. The initial data read in at the beginning of the computation,

contains the information about initial hole cutting prepared before starting the solution. This initial hole is used to compute the interpolation data for inter-grid coupling, which is for steady or unsteady problems without grid (body) motion, computed only once at the beginning of the computation. For the solution of governing equations the same steps are made as on single grids (see section 3.5.4), including additional modifications introduced for inactive and interpolation cells. For moving bodies the new body position is computed at each time step. The trajectory of body motion can be either predefined[6] or computed as a result of the interaction between the body and fluid flow. Based on the new body position, the overlapping definition is recomputed, i.e the new holes are created. The cells which are being covered by bodies are deactivated, and inactive cells which re-enter the computational domain are re-activated.

[6] In all cases computed in the present study the trajectory of body motion was predefined.

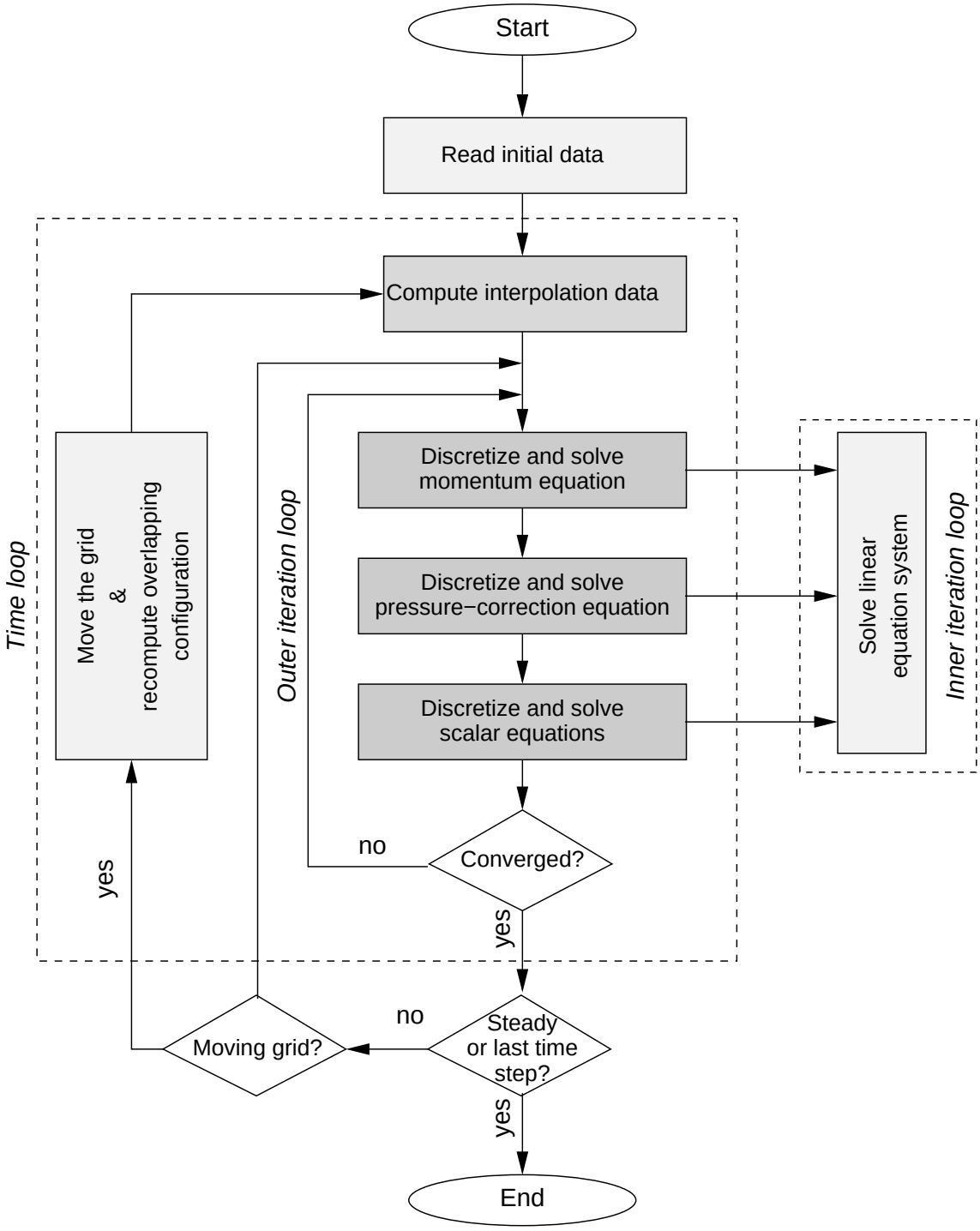


Figure 5.14: Solution procedure for computation of flows on overlapping moving grids.

CHAPTER 6

Verification and application of the method

The numerical method described in previous chapters has been embodied in a computer program for two-dimensional problems allowing the use of arbitrary unstructured overlapping and moving meshes. In this chapter the accuracy and efficiency of the method is assessed on a number of test computations. The test cases selected are ones for which either well established numerical solutions exist, experimental data is available, or the solution may be obtained by another program that employs the same or similar methodology[1]. On a number of test cases the present method was proven to produce correct results on single grids before it was extended to overlapping grids. Therefore, the attention here is focused mainly on the assessment of the method performance when overlapping grids are used. However, wherever it was possible, the computations were performed also on single grids. These results served mainly as a reference for testing of the overlapping grid method, but also could be used for further assessment of the single grid results. The great flexibility of the overlapping grid technique in the computation of flows around moving bodies is demonstrated on three examples which involve complex motion of bodies relative to each other.

6.1 Flow around a circular cylinder in a channel

In this section the steady laminar flow around a circular cylinder placed in a channel was considered. This case was used earlier as a benchmark test case under the DFG Priority Research Program "Flow Simulation on High Performance Computers". The case has been studied by several research groups and the results obtained using different numerical techniques were reported in [67]. All previous computations were performed on body-fitted, single grids. We compare here the results obtained using the overlapping-grid technique with reported results from the literature and our single-grid results.

The flow domain and the boundary conditions are schematically shown in figure 6.1. The parabolic velocity profile which corresponds to a fully developed laminar flow in a channel is

[1] In this study the commercial code *Comet* is used for an additional verification of the results obtained by the code developed within this study, were applicable.

prescribed at the inlet boundary, which is located $2D$ upstream from the cylinder center:

$$u = \frac{6U}{H^2}\left[(y+2D)\,H - (y+2D)^2\right], \qquad v = 0, \tag{6.1}$$

were U is the mean velocity, D is the cylinder diameter and $H = 4.1\ D$ is the channel height. At the outlet, which is located $20D$ downstream from the cylinder-center, the constant pressure is prescribed and at the cylinder surface and channel walls the no-slip wall boundary conditions were applied (the fluid velocity at wall $\mathbf{u}_b = 0$).

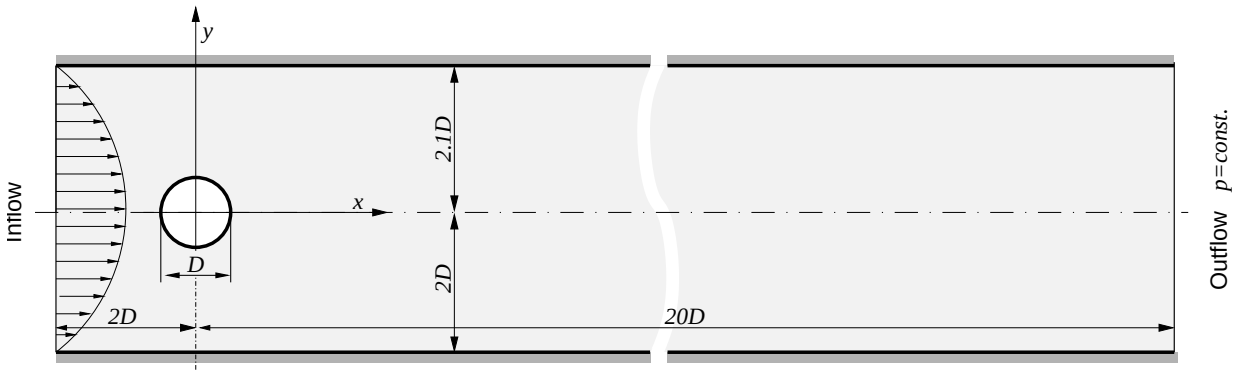


Figure 6.1: Geometry and boundary conditions for laminar flow around a cylinder in a channel.

The flow domain dimensions and the fluid properties used in the computation are as follows: $D = 0.1$ m, $U = 0.2\,\mathrm{m/s}$, $\rho = 1\ \mathrm{kg/m^3}$ and $\mu = 0.001$ Pas. The Reynolds number based on the mean inlet velocity U and cylinder diameter D is $Re = \rho U D/\mu = 20$. The characteristic flow pattern with two attached vortices formed behind the cylinder is shown in figure 6.2. The flow is slightly asymmetric since the cylinder center is not on the horizontal symmetry plane of the channel. Due to asymmetry, different flow rates and different pressures appear above and below the cylinder, resulting in a small lift force.

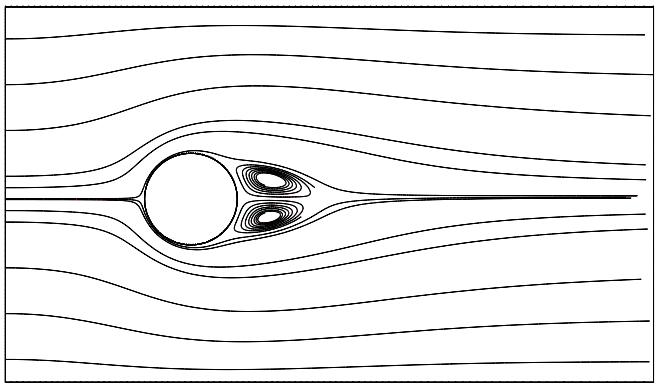
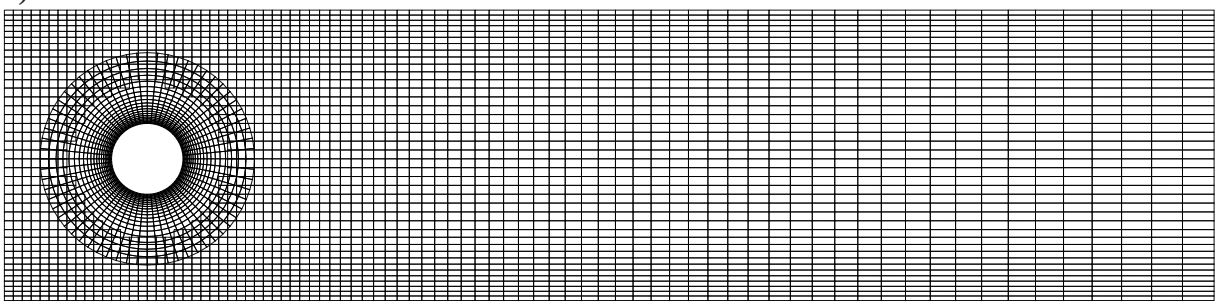


Figure 6.2: Steady flow pattern with two non-symmetrical vortices behind the cylinder at $Re = 20$.

For the analysis of spatial discretization errors, the computation has been performed on four systematically refined grids, using both the overlapping and single grids. The computations on the single grids were also performed using the commercial program Comet. The second level of grid-refinement for both grid configurations is shown in figure 6.3.

a) 4960 CVs



b) 4448 CVs

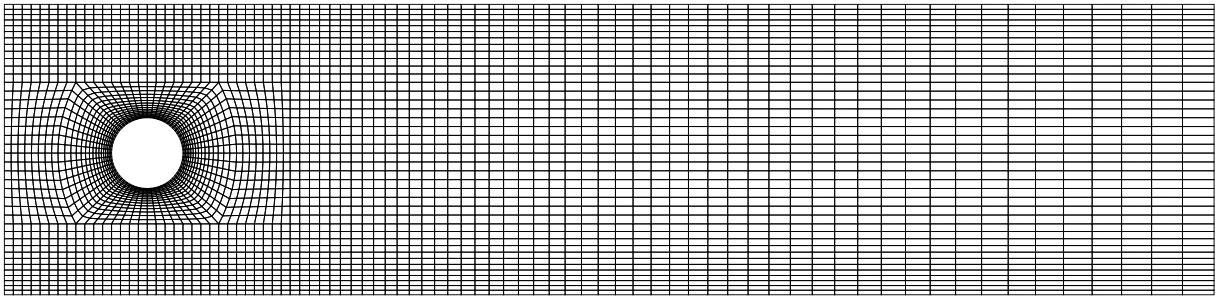


Figure 6.3: Second grid level for a) overlapping grid configuration with 4960 CVs and b) single grid configuration with 4448 CVs. Only the part of the computational domain for $x \leq 15D$ is shown.

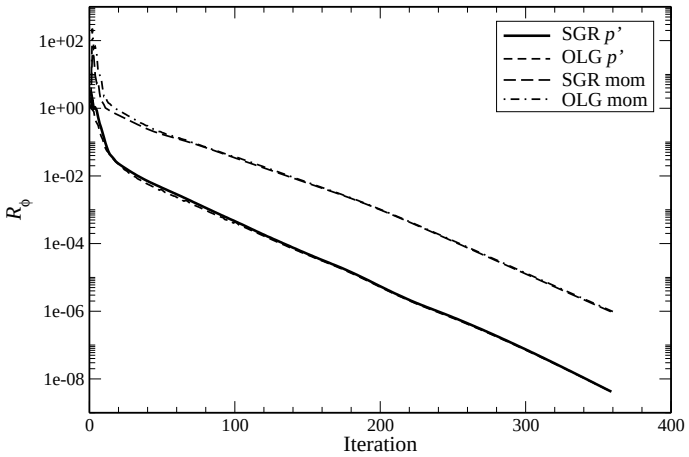
For the overlapping grid configuration, two grids are used to cover the computational domain: a cylindrical grid in the vicinity of the cylinder and an orthogonal nonuniform grid in the background which covers the whole channel. The cylindrical grid extends one diameter from the cylinder and is stretched to resolve the flow close to the cylinder wall. The first grid level had 32 uniformly distributed CVs around the cylinder and 10 CVs in the radial direction. The grid was expanded in the radial direction with an expansion factor of 1.25. Finer grids are obtained by doubling the number of cells in each direction. In order to keep the same grid-line distribution in the radial direction, expansion factor on each refined grid was calculated as the square-root of expansion factors on the next coarser grid. The thickness of the cell next to the wall, in the direction normal to the cylinder surface, on four grids was 0.03, 0.0142, 0.0069 and 0.0034 diameters, respectively. The first level background grid had 20 CVs in y direction and 46 CVs in x direction. The grid was stretched in the y direction to get better resolution near the channel walls. In the x direction, the grid was uniform in front of the cylinder and up to $2D$ behind the cylinder. Thereafter the grid was coarsened towards the outlet boundary.

For the single grid computation, the grid was modified in the region around cylinder. In order to fit the cylinder boundaries, a block structured grid is employed. An O-type grid block around

the cylinder and two H-type blocks in front and behind the cylinder were used to cover the part of the domain near the cylinder. The remaining part of the grid was the same as in the overlapping grid configuration. The grid block around the cylinder had the same number of cells in both directions as the cylindrical grid in the overlapping grid configuration. The cell distribution in the radial direction is similar as on the cylindrical grid, providing approximately the same cell thickness near the cylinder surface.

Overlapping grids had about 10% more cells than the corresponding single grids. Some cells in the background grids, which were not used in the computation, could be removed in this case since the grids does not move. However, these cells were deactivated rather then removed in order to investigate if the treatment of passive cells works correctly. Note that, although both overlapping and single grids have a regular structure, they are considered as unstructured during the computation.

In all computations the under-relaxation factors were set to 0.7 for the momentum equations and 0.3 for the pressure-correction equation. The inner iterations (in the linear equation solver) were stopped after the residuals have reduced one order of magnitude, while the convergence criterion for the outer iterations was set to 10^{-6}. The convective fluxes were computed using the central differencing scheme (CDS). All computations were started from the zero initial fields. Figure 6.4 shows the convergence history obtained on the second grid level for overlapping and single grids. The convergence rate is very close to each other on both grid configurations and almost the same number of iterations was required to achieve the convergence criterion. This convergence rate on overlapping grids is achieved owing to strong inter-grid coupling. The CPU time is higher for overlapping grids, which is partially due to additional computational effort associated with overlapping grids and partially can be attributed to the number of cells (the loops run also over the passive cells).



	Number of iterations		CPU time	
Gr.	SGR	OLG	SGR	OLG
1	122	122	0.95	1.03
2	359	360	27.27	30.31
3	1241	1247	1368	1451
4	4633	4618	32650	38351

Figure 6.4: Convergence history on the second grid level (left) and CPU time (right) for single (SGR) and overlapping (OLG) grid.

To investigate the accuracy of the method, several parameters of interest were computed. The

flow features we first look at are the drag and lift coefficients, defined as:

$$C_{\mathrm{D}} = \frac{F_x}{\frac{1}{2}\rho U^2 D}, \quad C_{\mathrm{L}} = \frac{F_y}{\frac{1}{2}\rho U^2 D}, \tag{6.2}$$

where F_x and F_y are the total forces in x and y directions exerted by fluid on the cylinder. Values obtained on four systematically refined grids and for both overlapping and single grid configuration, are summarized in table 6.1. Results on single grids obtained using the commercial code Comet are also presented. All results are very close to each other and they are in a very good agreement with the benchmark data. The difference between solutions on consecutive grids was reducing by about a factor of four, which is in accordance with expectations of a second-order scheme. From the difference between solutions obtained on two finest grids, the solutions at the finest grids are found to be accurate within 0.15 %. Table 6.1 also shows the "grid-independent" solutions obtained using Richardson extrapolation. Although the results on the coarsest level differ between the overlapping grid and Comet by $2.6\,\%$ for the drag force and $5\,\%$ for the lift force, the extrapolated "grid-independent" solutions agree within 0.006 % (drag) and $0.019\,\%$ (lift).

Table 6.1: Drag coefficient C_{D} and lift coefficient C_{L} as a function of the grid fineness.

	Overlapping grid			Single grid			Comet (single grid)		
Gr.	NCV	C_{D}	C_{L}	NCV	C_{D}	C_{L}	NCV	C_{D}	C_{L}
1	1240	5.40077	0.01183	1112	5.46882	0.01207	1112	5.53952	0.01245
2	4960	5.53412	0.01138	4448	5.55076	0.01088	4448	5.57310	0.01093
3	19840	5.56939	0.01072	17792	5.57288	0.01070	17792	5.58029	0.01071
4	79360	5.57784	0.01067	71168	5.57810	0.01066	71168	5.58082	0.01065
Extrapolated		5.58065	0.01065	——-	5.57984	0.01065	——-	5.5810	0.01063

Another accuracy test is presented in figure 6.5, which shows the pressure distribution along the cylinder surface versus angle φ, measured from the rear stagnation point counterclockwise towards the front stagnation point, as indicated in figure 6.5. The results obtained on single and overlapping grids are presented. Both results show approximately second-order convergence towards a grid-independent solution. Very small difference between results on the two finest grids suggests that the discretization errors on the finest grid are very small. The comparison of the results obtained on overlapping grids with those obtained on single grids using the present method and the program Comet is given in figure 6.6. Results obtained on the second level grid are presented, showing a very good agreement. The present method provides on the single grids results very close to Comet and therefore hereafter we shell compare only our single grid results with results obtained using overlapping grid technique.

The pressure difference between the front and rear stagnation points can be extracted from figure 6.5. Position $\phi = 180^{\mathrm{o}}$ corresponds to the front stagnation point, while the rear stagnation

point lies at the position $\phi = 0^{\circ}$ and $\phi = 360^{\circ}$. The pressure drop was found to vary between 0.109 on the grid 1 and 0.1172 on the finest grid, which agrees very well with solutions from literature [67] ($0.1172 - 0.1176$). Differences between results obtained on the overlapping grids and those obtained on single grids were within 1.5% at the coarsest grid and about 0.5% at the finest grid.

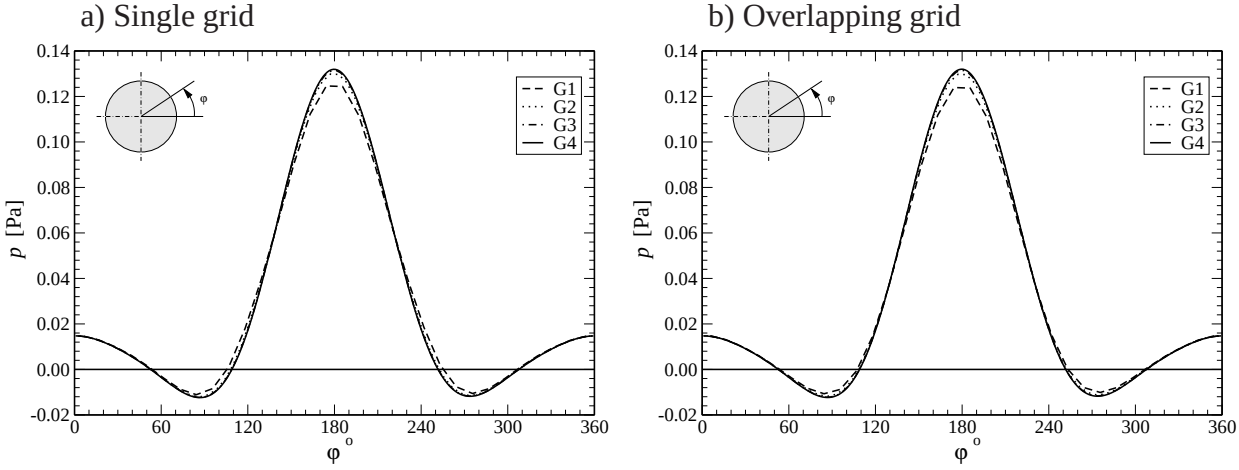


Figure 6.5: Distribution of pressure along cylinder surface, obtained on four systematically refined grids for: a) single grid, b) overlapping grid.

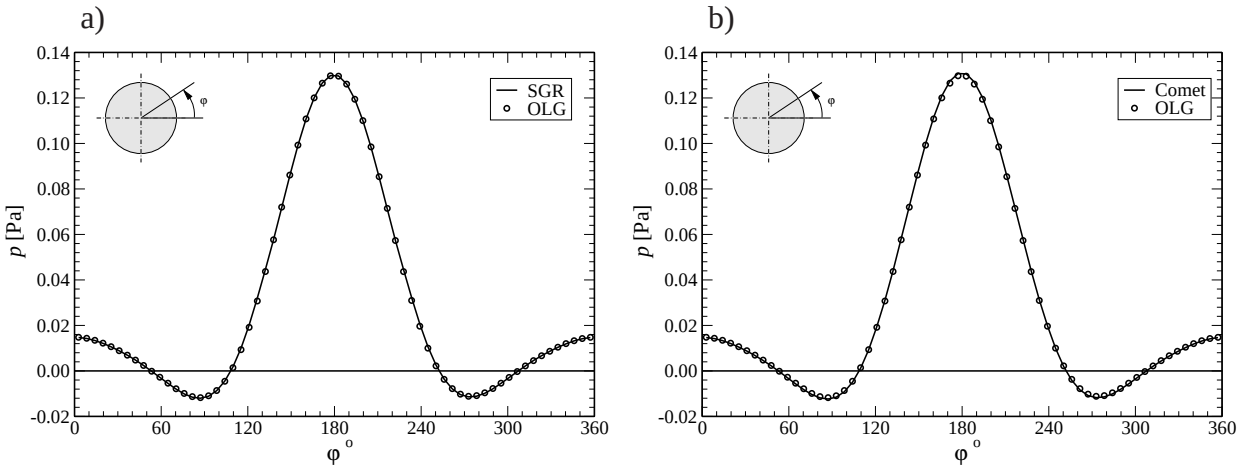


Figure 6.6: Comparison of the results obtained on the grid 2; SGR denotes single grid method, OLG denotes overlapping grid method.

The length of the recirculation zone behind the cylinder was also recorded and used as a grid-dependence indicator. Figure 6.7 shows the profiles of the streamwise velocity component along the line $y = 0$. Results for the single and overlapping grid computations obtained on four grids are presented. The profiles are given for only a part of the domain up to $x = 5D$. Convergence of approximately second-order can be observed also in this case. The results obtained on two finest grids are almost identical and thus nearly grid-independent. From figure 6.7 it is visible that the length of the recirculation zone was increasing with the grid refinement. On the finest grid it was

found to be about 0.0837, which agrees within 1% with the benchmark results from [67].

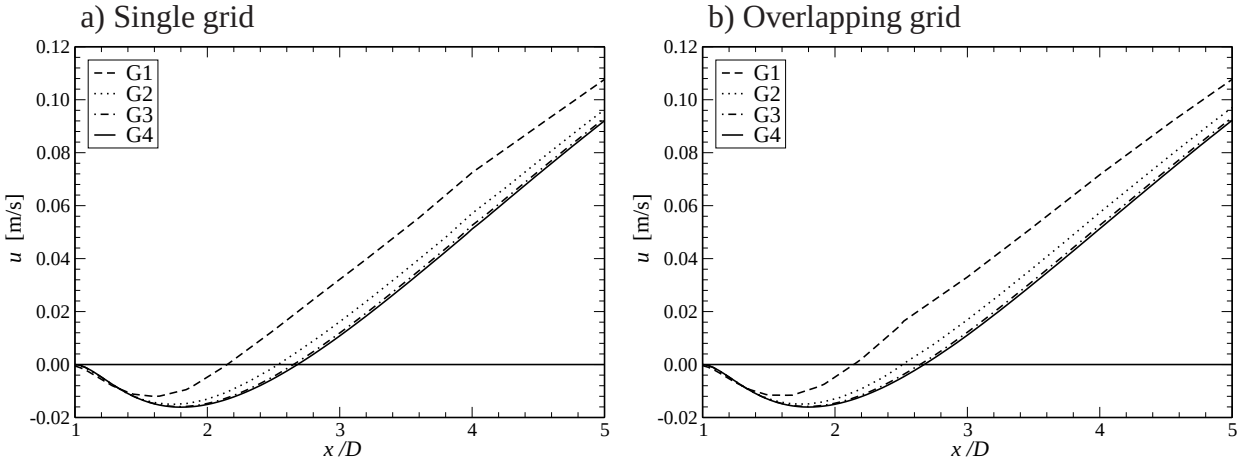


Figure 6.7: Profiles of the u velocity along the line $y = 0$ behind the cylinder. Results obtained on four systematically refined grids for: a) single grid and b) overlapping grid are presented.

A more detailed comparison of results obtained on single and overlapping grids is given in figure 6.8, which shows the contours of the pressure and the u velocity obtained on all four grid levels. The single grid solution is given by black lines and the overlapping grids results are given by color lines. For overlapping grids, the results on both component grids are shown in the overlapping region. The overlapping region is indicated by the outer boundary of the cylinder grid and the interface which separates the interpolation and inactive cells in the background grid[2]. The agreement between the two methods is obviously very good. As the grid gets finer, the differences become smaller, and on the finest grid it is hardly visible. The discrepancy visible on the coarsest grid is expected due to large differences between single and overlapping grids. Although in the overlapping region two different grids are used, there is almost no difference between the contours on the two component grids. Slight difference that appears near the overlap boundaries is due to the lack of the flow information at the boundary points during the postprocessing (presentation of results). Smooth representation of the flow field in the overlapping region confirms that the interpolation procedure used here provides a correct coupling between the component grids and leads to a unique solution over the whole domain.

[2]No information is plotted for the inactive cells.

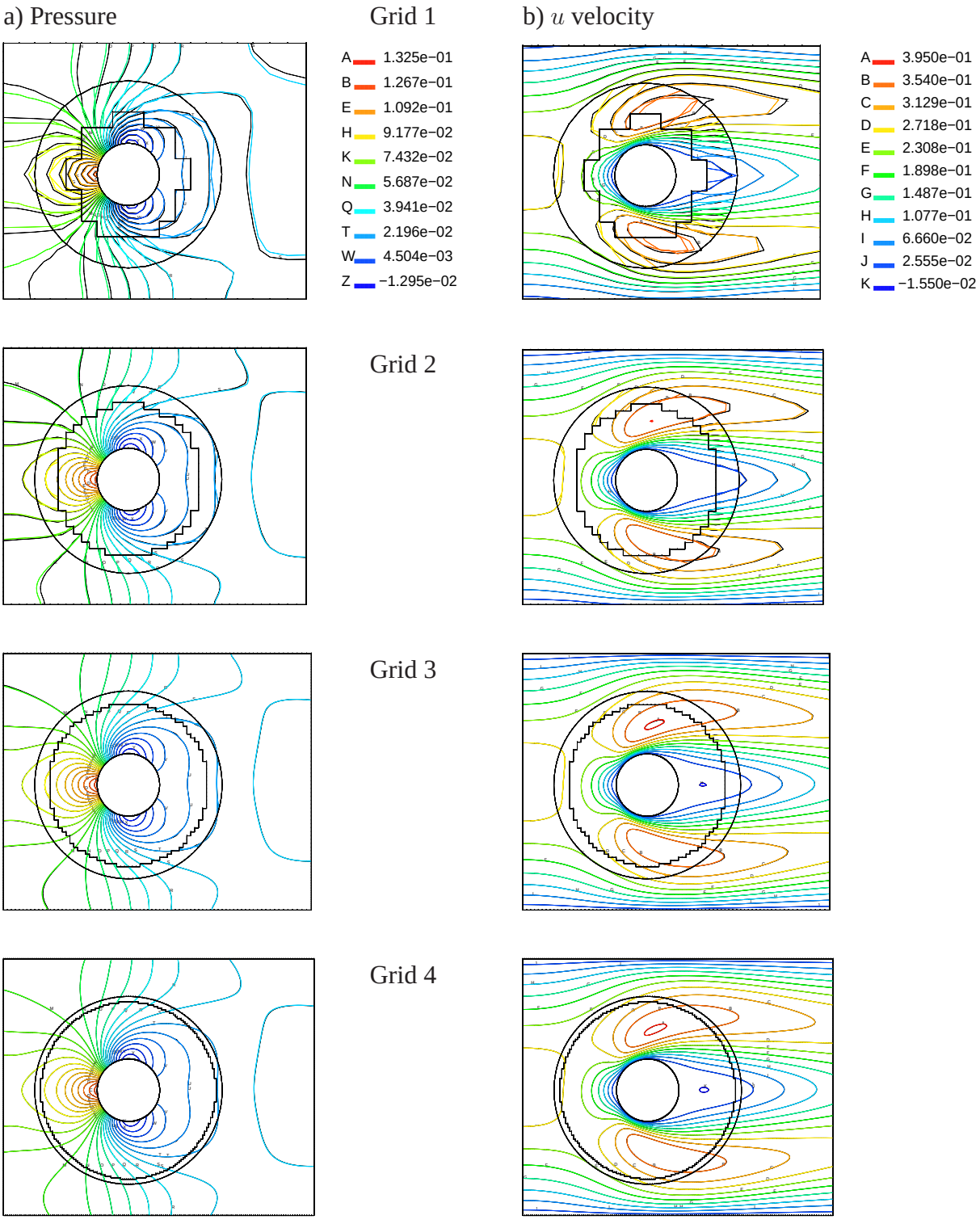


Figure 6.8: Pressure field a) and u velocity field b) obtained on for grids. Black lines are single grid results and color lines are overlapping grid results.

6.1.1 Influence of the cylinder grid

One of the factors which might influence the results obtained with the overlapping grid method is the location and the width of the overlapping region. This issue was also investigated here. To this end four cases were computed, with different grids around the cylinder so that the area around the cylinder covered by the cylindrical grid varied from $\delta = 0.5D$ to $\delta = 1.25D$ as shown in figure 6.9. In all four cases the same background grid was used. As the size of the cells at the outer boundary of the cylindrical grid is different for these four cases, the width of the overlapping region was chosen to be minimal. For each configuration, the computation was performed on three systematically refined grids. The results are summarized in table 6.2.

Table 6.2: Drag coefficient obtained on four grids given in figure 6.9.

Grid	$\delta = 0.5D$	$\delta = 0.75D$	$\delta = 1.0D$	$\delta = 1.25D$
2	5.5449	5.5333	5.5341	5.5361
3	5.5723	5.5696	5.5693	5.5710
4	5.5790	5.5804	5.5778	5.5791
Extrapolated	5.5812	5.5836	5.5806	5.5818

Except for the coarsest grid, the difference between solutions obtained on the same grid level with different location of interface are very small. On the grid level 4, the difference between the largest and the smallest value is about $0.05\,\%$; in all four cases, Richardson extrapolation produces values, which differ by less then $0.05\,\%$, which can be considered as very accurate.

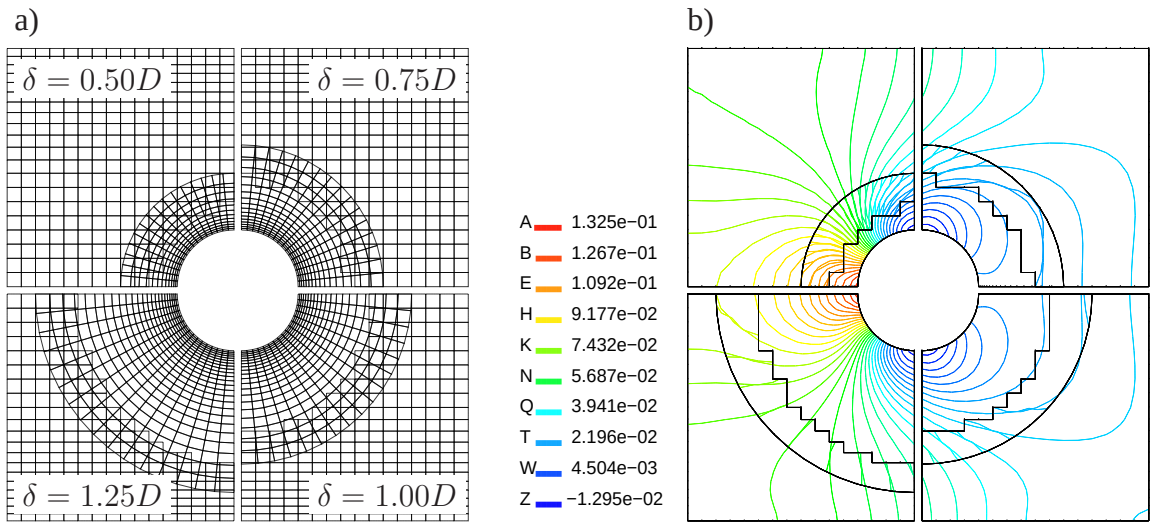


Figure 6.9: Resulting overlapping grids for different positions of the outer boundary on the cylinder grid a) and the resulting pressure field b).

6.1.2 Conservation errors

The computations presented in preceding sections have shown that accurate results can be obtained on overlapping grids although the conservation along grid interfaces is not strictly satisfied. However, since the mass conservation is in some circumstances essential for the numerical solution (e.g. flows in closed domains), it is important to estimate the conservation errors and especially to investigate their behavior as the grid is refined. Our attention is focused on the conservation of mass, since the conservation error for other quantities is of lesser concern. In the present case, the pressure boundary condition was applied at the outlet. Therefore it was not necessary to enforce the mass conservation along overlapping interfaces explicitly. We have utilized this fact to estimate the conservation error, which in this case can be represented as the difference between the total inflow and outflow mass fluxes. The outflow mass flux converges on a single grid to the value defined by the total inflow flux. Due to nonconservative interpolation, the sum of mass fluxes through the overlapping interfaces is not equal to zero. An artificial mass source is generated, which also contributes to the outflow mass flux. The resulting inflow and outflow fluxes as well as the difference between them obtained on four systematically refined grids is given in table 6.1.2[3]. It is obvious that the mass imbalance caused by the interpolation is small in comparison to the total inflow mass flux and below the level of discretization errors on the same grid. The error is below 1% at the coarsest grid and decreases with approximately second-order with grid refinement. On the finest grid the conservation error is an order of magnitude lower than the estimated discretization error for the drag, see previous section.

Table 6.3: Mass conservation errors estimated on four systematically refined overlapping grids.

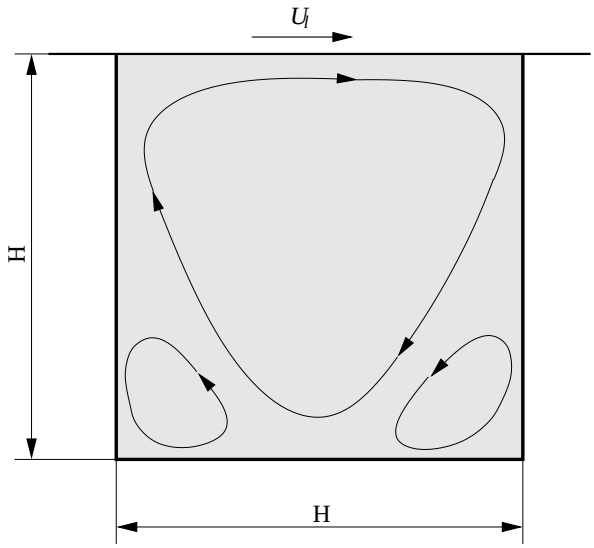
Grid	NCV	$\dot{m}_{\text{in}}$	$\dot{m}_{\text{out}}$	$\Delta\dot{m}$	$\varepsilon(\%)$
1	1240	0.08212	0.08164	$4.772 \cdot 10^{-4}$	0.5811
2	4960	0.08203	0.08186	$1.656 \cdot 10^{-4}$	0.2018
3	19840	0.08201	0.08197	$3.551 \cdot 10^{-5}$	0.0433
4	79360	0.08200	0.08110	$4.444 \cdot 10^{-6}$	0.0054

6.2 Lid-driven cavity flow

In the previous example it was shown that the mass conservation may not be strictly satisfied if the pressure boundary condition is used on a part of the solution domain. The flow rate at the pressure boundary is an unknown quantity and it depends on the inflow and/or any mass sources or sinks. The mass imbalance at interfaces of overlapping grids is compensated by adjusting outflow through the pressure boundary. For the other combinations of boundary conditions, when the flow rate at domain boundaries is fixed, the strict mass conservation is essential for

[3]Since the parabolic velocity profile is prescribed at inlet, the values for total inflow flux are slightly different on each grid, i.e. they depend on the grid resolution.

the solution of the pressure-correction equation. Otherwise, the convergence problems or even divergence of the iterative solution procedure may occur. In the present study a conservative correction is introduced, which ensures that the mass conservation is satisfied within each outer iteration (see section 5.3.3). A representative case of this situation is the flow in a closed domain with impermeable walls. For simplicity, we consider the well known lid-driven cavity flow. This test case has been studied by many authors and accurate results are available in literature, see e.g. [27]. The computational domain is a two-dimensional square cavity with characteristic dimension $H = 1\ \mathrm{m}$ as shown in figure 6.10. The fluid used in the computation has the following properties: density $\rho = 1\ \mathrm{kg/m^3}$ and dynamic viscosity $\mu = 0.001\ \mathrm{Pas}$. The no-slip boundary conditions are applied at all boundaries and the flow is driven by the top wall (lid) which moves with the velocity $U_l = 1\ \mathrm{m/s}$. The Reynolds number based on the height of the cavity and the lid velocity is $\mathrm{Re} = \rho H U_l/\mu = 1000$, yielding steady laminar flow. The benchmark solution for this case is given in [27].



$H = 1\ \mathrm{m}$
$U_l = 1\ \mathrm{m/s}$
$\rho = 1\ \mathrm{kg/m^3}$
$\mu = 0.001\ \mathrm{Pas}$
$\mathrm{Re} = 1000$

Figure 6.10: Geometry and boundary conditions for the lid-driven cavity flow.

For the solution of this problem a single Cartesian grid would be the most suitable. However, we are mainly interested in the solution on overlapping grids. Special attention was paid here to the investigation of conservation errors and their influence on the solution. To this end an overlapping grid system is used which consists of two components: a background grid, which covers the entire domain, and another grid embedded within the background grid, which covers a part of the computational domain in the central region of the cavity. A number of cells in the background grid were deactivated, thus making the solution on both grids dependent on each other. Figure 6.11 shows an overlapping grid system consisting of two uniform Cartesian grids: the background grid with 40×40 cells and an overset grid with 24×24 cells. The spacing on both grids was the same. Computations were performed for four different positions of the overset grid, as indicated in figure 6.11. For the position $\alpha = 0^o$ the "perfect" overlapping

between the background and the overset grid is obtained. In this case only one donor point is needed to interpolate the variable values at interpolation cells on both grids. These values are simply obtained as the values at corresponding donor cells. For other three positions of the overset grid, an arbitrary overlapping region is obtained.

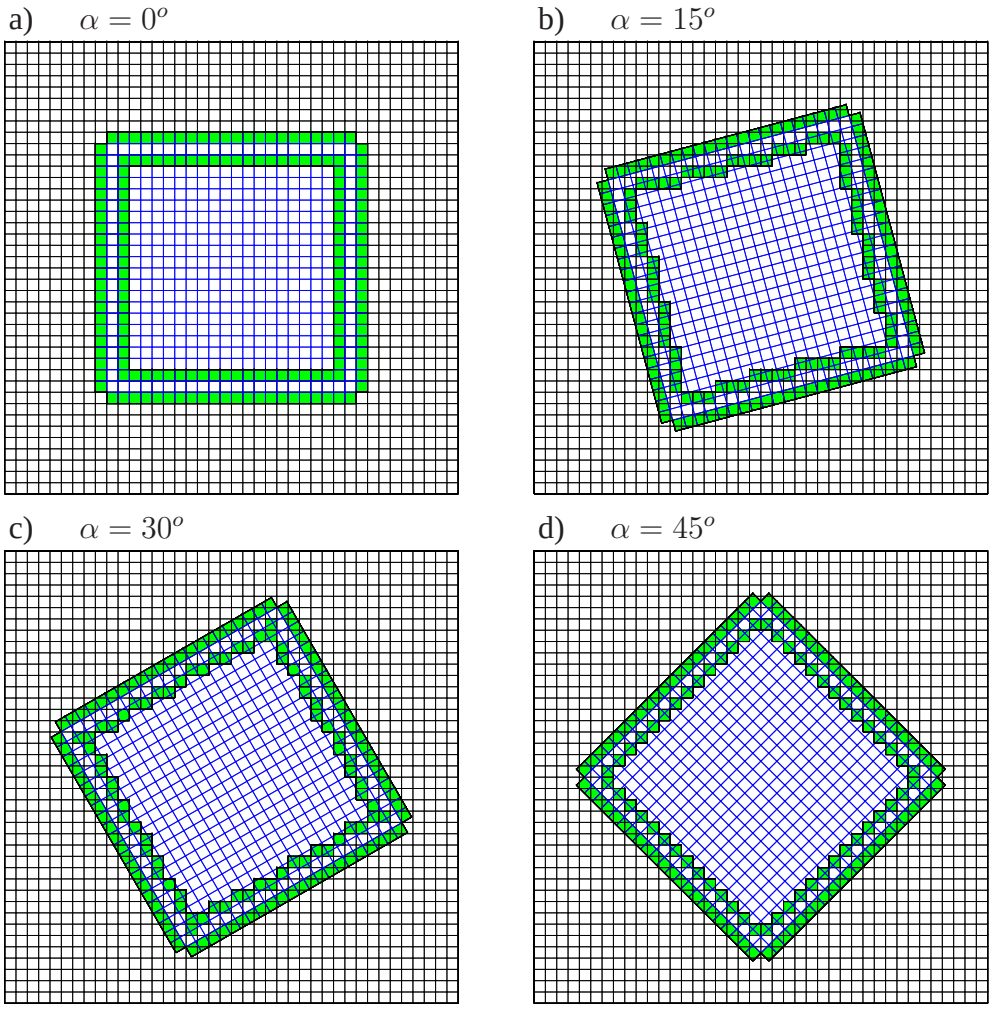
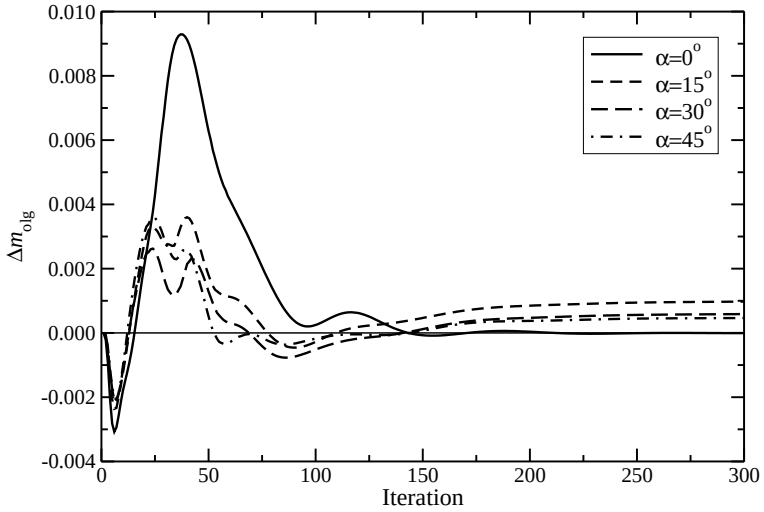


Figure 6.11: Overlapping Cartesian grids used for the computation of the lid-driven cavity flow. Number of cells: background grid $40 \times 40 = 1600$ cells; overset grid $24 \times 24 = 576$ cells.

In figure 6.12 the variation of the net mass flux through the overlapping interfaces during the course of iterations is shown. At the beginning of the computation (first 100 iterations), stronger changes are evident. Thereafter the changes are small and already after about 200 iterations, converged values given in the table (see figure 6.12) are reached. As expected, the net mass flux for $\alpha = 0^o$ converges to zero[4]. For the other three cases, it converges to a finite value, which is, according to data given in the table in figure 6.12, very small (below 1%) in comparison to the flow rate between the wall and the center of the large vortex defined by the maximum value of

[4]The final value is of the order of 10^{-15} which is within the round-off error of the computer.

stream function. Although the conservation errors are obviously small, they still may have an influence on the solution and may cause convergence problems. Therefore, these errors have to be eliminated, i.e. the strict mass conservation has to be enforced.



α	$\Delta\dot{m}$	$\psi_{\max}$
0^o	$-6.181 \cdot 10^{-15}$	0.1057
15^o	$9.925 \cdot 10^{-04}$	0.1042
30^o	$6.030 \cdot 10^{-04}$	0.1043
45^o	$4.757 \cdot 10^{-04}$	0.1043

Figure 6.12: Variation of the net mass flux through overlapping interfaces during iterations obtained for four different positions of the overset grid. The net mass flux is computed before the conservative correction defined by equation (5.9) is applied to the interface fluxes.

The effect of the mass conservation can be observed on the convergence of the iterative solution procedure. To this end the computations have also been performed without applying the conservative correction. In figure 6.13 the convergence history for cases $\alpha = 0^o$ and $\alpha = 45^o$ computed with and without conservative correction is shown. For $\alpha = 0^o$ (figure 6.13 a) there is no visible difference in the convergence rate: deep convergence (up to the accuracy of the computer 10^{-15}) could be reached also without correction. This is understandable since the meshes overlap perfectly and the solution is equivalent to that obtained on a single grid. The mass is conserved as the converged solution is approached. For $\alpha = 45^o$ (figure 6.13 b), the influence of the mass conservation is obvious. In the computation without conservative correction, the residuals could be reduced only by about five orders of magnitude, which is far from the accuracy of the computer. Although the convergence in this case was smooth and correct results were obtained, one needs to bear in mind that such situation is an exception rather then the rule. The level to which the residuals can be reduced, and thus the accuracy of the solution, depend on the mass imbalance at overlapping interfaces, which is both problem and grid dependent. With the correction residuals can be reduced to machine round-off level, and this takes the same number of iterations as in the case of a single grid.

In order to assess the validity of the proposed method for achieving mass balance and to investigate the accuracy, computations were performed on three levels of grid refinement, using both single and overlapping grids. Although a Cartesian grid can best accommodate the problem geometry, a triangular grid was also used in order to demonstrate the generality of the method.

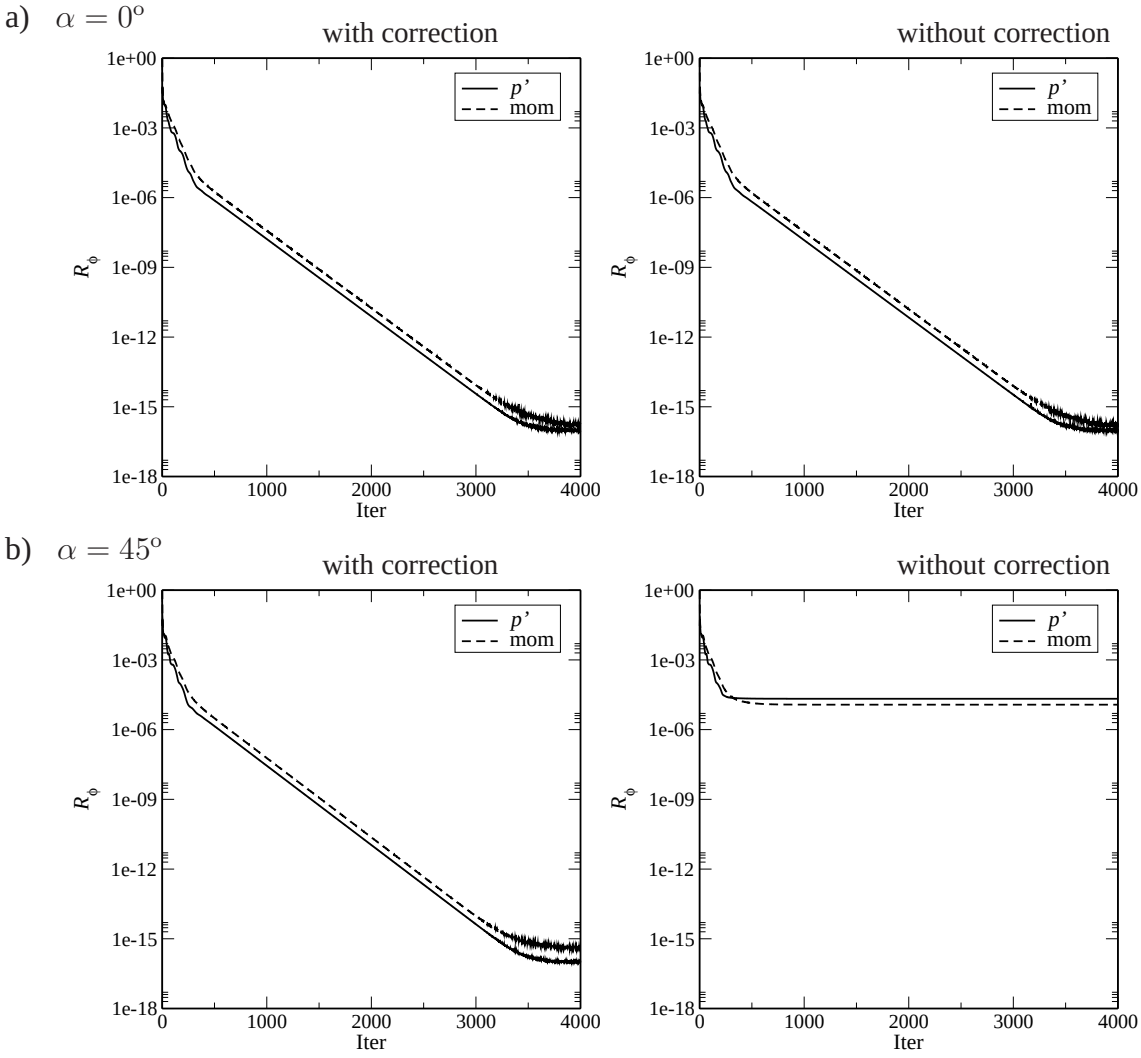


Figure 6.13: Convergence history for: a) $\alpha = 0^o$ and b) $\alpha = 45^o$.

Figure 6.14 shows two combinations of overlapping grids: Cartesian-Cartesian and Triangular-Cartesian for all three levels of refinement. For the single grid computations only the background grids, which cover the whole computational domain, were used. The number of cells on each grid given in figure 6.14 includes all cells on each grid component. Note that some cells in background grid were deactivated during the computation, but were not removed. Triangular grids were generated by trying to keep the number of cells close to the number of cells on corresponding Cartesian grids. Under-relaxation factors of 0.7 for the momentum equation and 0.3 for the pressure-correction equation were used. The same values were used for single and overlapping grids. The convergence criterion was set to 10^{-8}, which is considered to be enough to eliminate any influence of the iteration errors on the solution. The convective fluxes were approximated by CDS scheme, providing a second order spatial discretization. All computations were started from the zero initial field. The overlap region between the component grids was shrinking as the

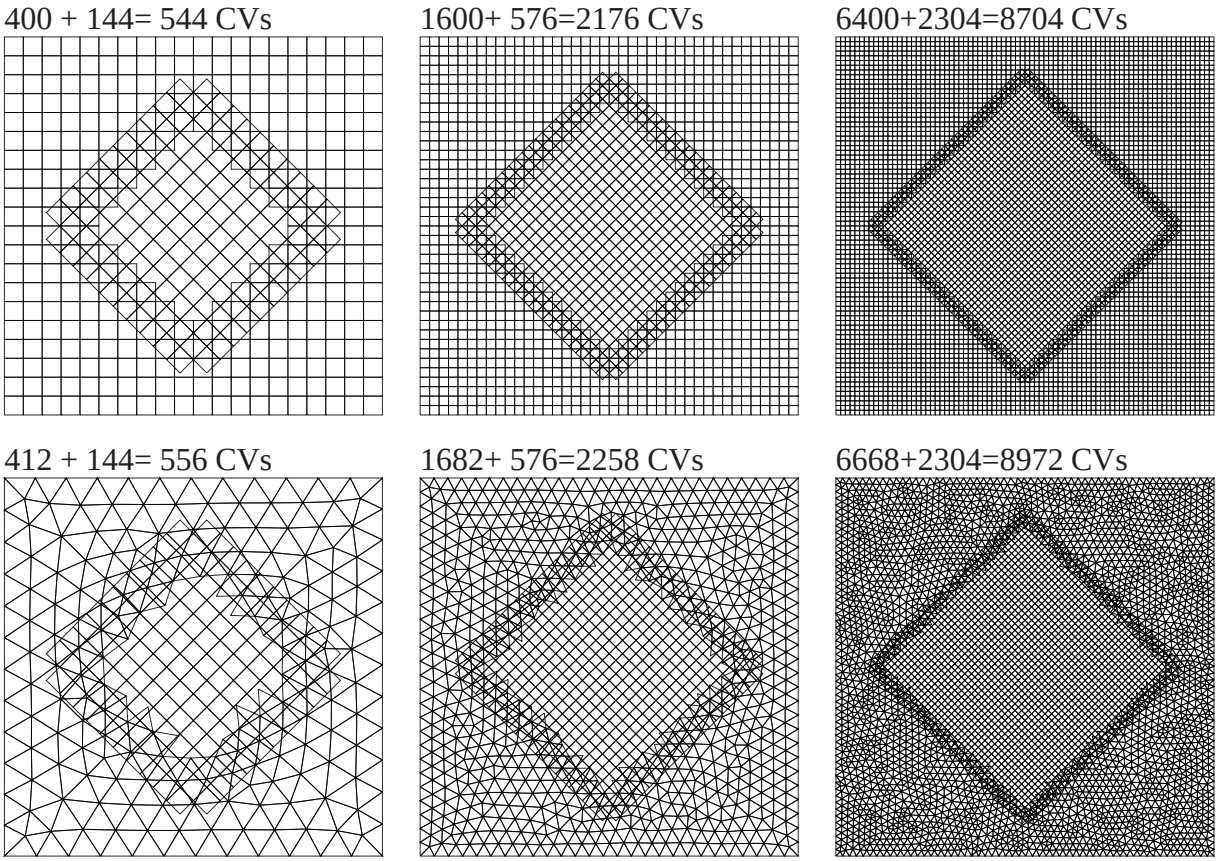


Figure 6.14: The overlapping grids used for the computation of the lid-driven cavity flow: a) overlapping grid Cartesian-Cartesian, b) overlapping grid triangular-Cartesian.

grids were refined, as shown in figure 6.14.

Figures 6.15 and 6.16 show the profiles of u and v velocity components taken along symmetry lines of the cavity on single and overlapping grids and for all three levels of grid refinement. The profiles are compared with the benchmark results from [27], showing a very good agreement on the finest grid. Slight discrepancies appear because the finest grid level is still not fine enough to produce a nearly grid-independent solution. Nevertheless, a monotonic convergence of approximately second-order to a grid-independent solution can be observed for single as well as for overlapping grids. Obviously, the results on overlapping grids are not degraded and approximately the same accuracy is achieved on both single and overlapping grids.

For the evaluation of conservation errors, the net mass flux through the overlapping interfaces denoted as $\Delta\dot{m}_{\mathrm{olg}}$ was calculated. This flux was compared with the stream function maximum, which represents the flow rate between the wall and the large vortex center and indicates how much mass is in motion. Thus, the mass conservation error is defined as: $\varepsilon_c = \Delta\dot{m}_{\mathrm{olg}}/\psi_{\max}$. The values obtained for each level of grid refinement and for both overlapping grids configurations

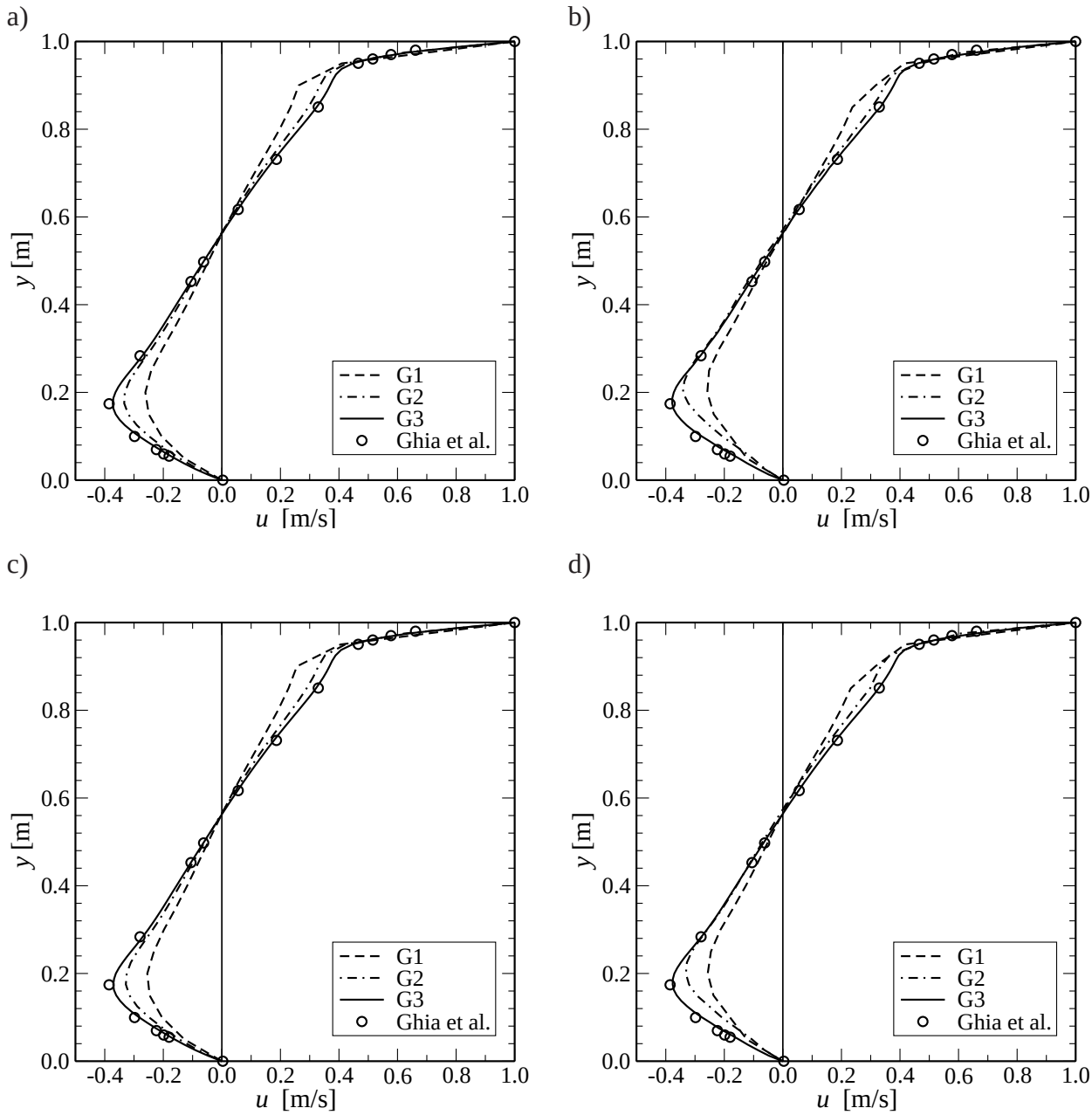


Figure 6.15: Profiles of u velocity component at vertical centerline $x = 0$ calculated on three levels of refinement and following grid setups: a) single Cartesian, b) single triangular, c) overlapping Cartesian-Cartesian and d) overlapping triangular-Cartesian grids.

are given in table 6.4. The data show that the conservation error, like the discretization error, is larger on irregular triangular grid. But the conservation error is still very small, about 1% on coarsest grid and it decreases monotonically with grid refinement.

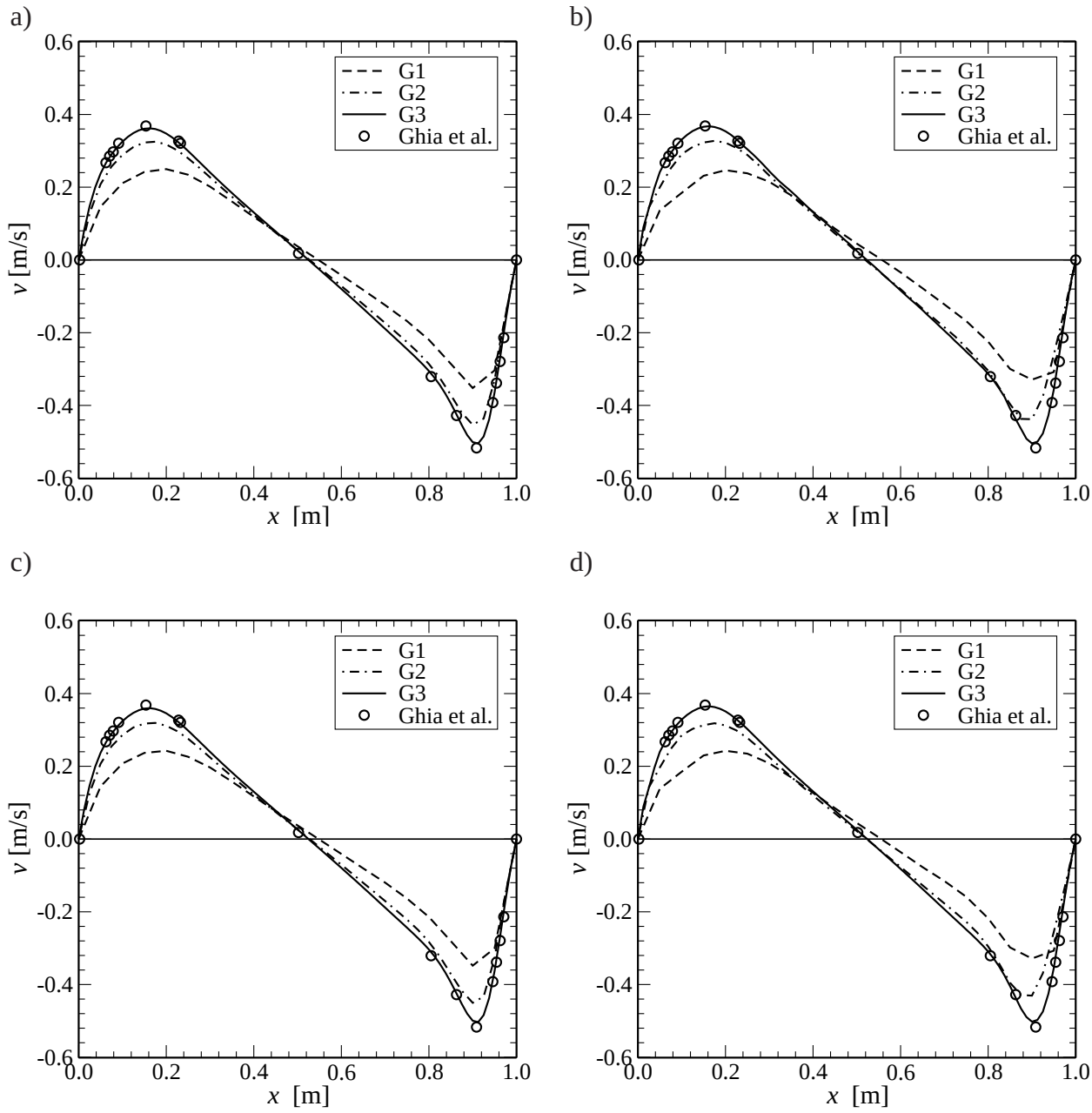


Figure 6.16: Profiles of v velocity component at horizontal centerline $y = 0$ calculated on three levels of refinement and following grid setups: a) single Cartesian, b) single triangular, c) overlapping Cartesian-Cartesian and d) overlapping triangular-Cartesian grids.

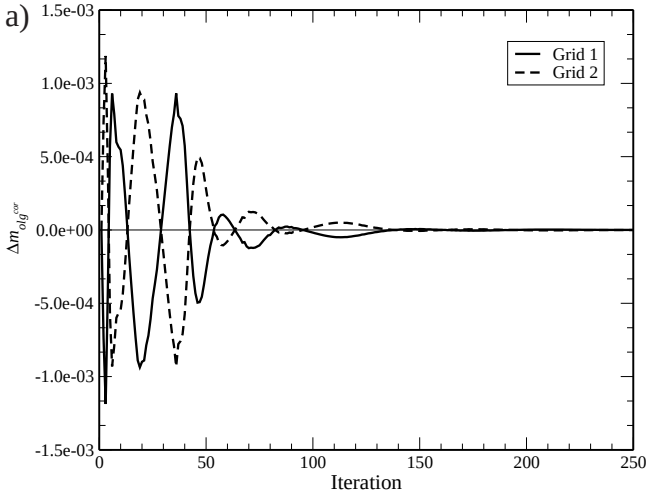
Another issue concerning the mass conservation is the level of conservation that is achieved on each grid. Since the conservative correction of interface mass fluxes defined by equation (5.9) is applied globally for the whole domain, the mass conservation is not enforced explicitly on each grid. To investigate how this reflects on the mass conservation on each grid, the net mass fluxes

Table 6.4: Net mass flux through the overlapping interfaces as a function of the grid refinement.

Grid level	Cartesian-Cartesian			Triangular-Cartesian		
	$\psi_{\max}$	$\Delta\dot{m}_{\mathrm{olg}}$	ε_c [%]	$\psi_{\max}$	$\Delta\dot{m}_{\mathrm{olg}}$	ε_c [%]
1	0.0967	$9.277 \cdot 10^{-4}$	0.3741	0.1012	$1.479 \cdot 10^{-3}$	0.3608
2	0.1071	$6.846 \cdot 10^{-5}$	0.0639	0.1080	$1.317 \cdot 10^{-4}$	0.0212
3	0.1148	$6.343 \cdot 10^{-6}$	0.0055	0.1167	$5.074 \cdot 10^{-5}$	0.0074

through the parts of the overlapping interface on each grid were computed after applying the correction (5.9). Figure 6.17 a) shows the variation of these fluxes during the computation obtained on the coarse Triangular-Cartesian grid configuration. Since there are only two component grids, the fluxes on each grid have the same[5] value but opposite sign. The final values obtained on each level of grid refinement and for both overlapping grid configurations are given in the table in figure 6.17 b). It is obvious that the interface fluxes tend to zero, indicating that the mass conservation is nearly satisfied also on each grid as the converged solution is approached. To check if this is valid also for more general cases, the computations have been performed for two more overlapping grids configurations (see figure 6.18). The results presented in figure 6.18 show that the interface fluxes on each grid converge to zero irrespective of the number of component grids.

a)



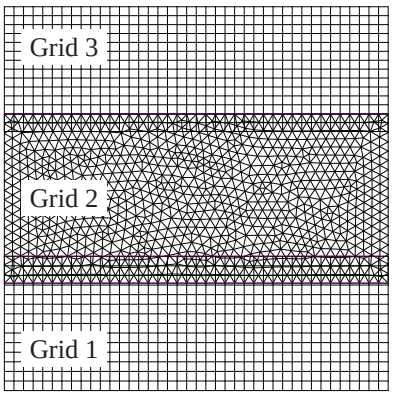
b)

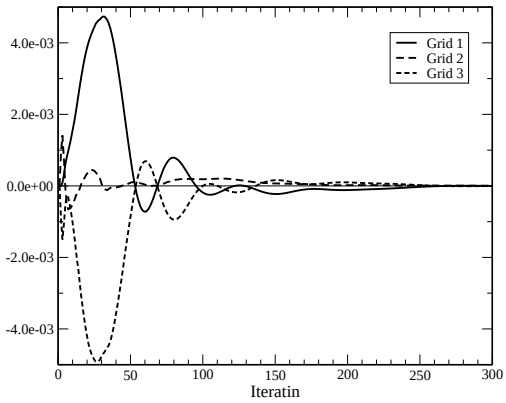
$\Delta\dot{m}^{\mathrm{cor}}_{\mathrm{olg_i}}$ - Cartesian-Cartesian		
Level	Grid 1	Grid 2
1	$-2.540 \cdot 10^{-09}$	$2.540 \cdot 10^{-09}$
2	$-8.057 \cdot 10^{-10}$	$8.057 \cdot 10^{-09}$
3	$-4.179 \cdot 10^{-10}$	$4.179 \cdot 10^{-09}$

$\Delta\dot{m}^{\mathrm{cor}}_{\mathrm{olg_i}}$ - Triangular-Cartesian		
Level	Grid 1	Grid 2
1	$4.139 \cdot 10^{-10}$	$-4.139 \cdot 10^{-10}$
2	$-1.683 \cdot 10^{-09}$	$1.683 \cdot 10^{-09}$
3	$-1.254 \cdot 10^{-09}$	$1.254 \cdot 10^{-09}$

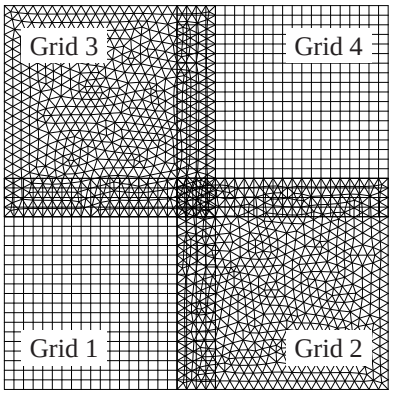
Figure 6.17: a) Variation of the net mass fluxes through the overlapping interfaces on each grid obtained on the first level of grid refinement for Triangular-Cartesian overlapping grid; b) Final values of interface fluxes on each grid obtained on different levels of grid refinement.

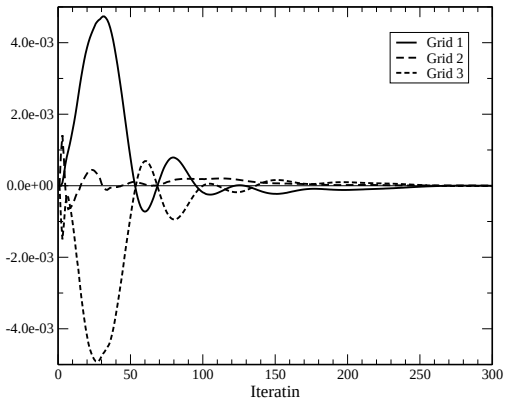
[5]Within the round-off error of the computer.





Grid	$\Delta\dot{m}$
1	$2.067 \cdot 10^{-9}$
2	$3.598 \cdot 10^{-10}$
3	$-2.427 \cdot 10^{-9}$





Grid	$\Delta\dot{m}$
1	$3.709 \cdot 10^{-9}$
2	$1.103 \cdot 10^{-9}$
3	$-2.238 \cdot 10^{-9}$
4	$-2.574 \cdot 10^{-9}$

Figure 6.18: Convergence of the net mass flux through overlapping interfaces on each grid for different overlapping grids configurations.

6.3 Flow around a NACA 4412 airfoil

In order to examine the reliability of the overlapping grid algorithm developed in the course of the present work for computation of turbulent flows, we performed a simulation of flow around the NACA 4412 airfoil as an example. The k-ω SST turbulence model of Menter [51] is used to compute the separated flow over the airfoil, and the results are compared with the experiments of Coles and Wadcock [17]. The computation was performed for the maximum lift conditions with the angle of attack $\alpha = 13.87^{\text{o}}$ and the Reynolds number $Re = c\,U_0/\nu = 1.52 \cdot 10^6$ based on chord length c and free-stream velocity U_0. The flow separates on the rear part of the airfoil upper surface at these conditions. It is known that the k-ε models fail to predict the separation in this case and it was demonstrated in numerous publications ([3]) that the $k-\omega$ SST model predicts such flows reasonably well.

A part of the overlapping grid system used for this computation is shown in figure 6.19. The grid consists of two blocks. The free-stream and the wake region far from airfoil is covered by an orthogonal background grid, while the area around the airfoil are covered by a body-fitted grid of C-type. In order to achieve a reasonable grid spacing and to keep the grid coarse in the region of lesser importance, the background grid is locally refined in the area close to the airfoil.

Several levels of refinement are applied to cover the wake region with sufficient grid resolution. The grid attached to the airfoil extends $0.3\,c$ from the airfoil-surface and $0.2\,c$ behind the airfoil and it is optimized in relation to the airfoil geometry. The total number of cells in the whole grid was 39503 whereof 13103 in the background grid and 26398 in the airfoil grid. Note that in this stationary case the inactive cells in the background grid were removed. The $k-\omega$ SST model used here is a low-Re-number model, meaning that the model equations are integrated up to the wall requiring very fine grid near the wall (typically for the cell near the wall $y_1^+ = \dfrac{y_1 u_\tau}{\nu} < 1$). Therefore the computational grid near the airfoil is very fine in the direction normal to wall.

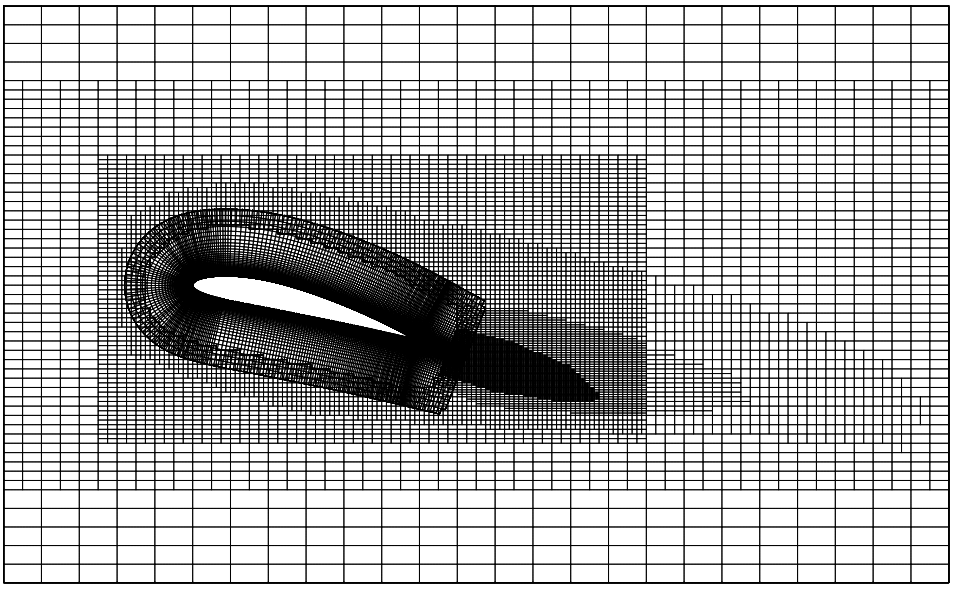


Figure 6.19: A part of the overlapping grids used for the computation of flow around NACA 4412 airfoil.

The inlet boundary was located at $5\,c$ in front of the airfoil with the following values specified: free-stream turbulence of $5\,\%$ ($\sqrt{\overline{u^2}}/U_o = 0.05$) and $Re_t = k^2/(\nu\beta^* k\omega) = 1000$ ($\beta^* = 0.09$ is a model constant, see appendix B). The outflow boundary with standard outlet boundary condition was located at $15\,c$ downstream from the airfoil. The top and the bottom boundaries of the computational domain, which were placed about $5\,c$ away from the airfoil, were considered as symmetry planes. The flow field was initialized with the free-stream conditions and the computation was started using the upwind scheme (blending factor $\gamma = 0$). Later, as the flow has developed, the blending factor was increased to $\gamma = 0.9$ ($90\,\%$ of CDS).

The computed distribution of the pressure coefficient $C_p = (p - p_o)/(0.5\,\rho\,U_o^2)$, ($p_o$ is the pressure at inlet boundary), shows very good agreement with experimental data. Figure 6.20 b) shows the streamlines around the airfoil. Contrary to most eddy-viscosity models, these results clearly show the trailing edge separation. Although the predicted recirculation is a bit smaller than observed in the experiments, the results obtained here are visually identical to results obtained by Huurdeman [36] with the same turbulence model but with a rather different finite-volume method.

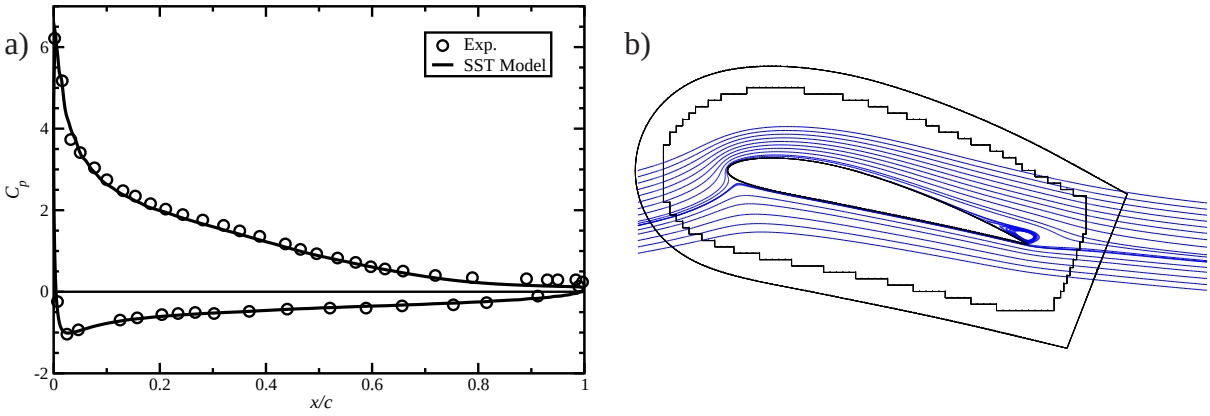


Figure 6.20: Distribution of pressure coefficient $C_p = \dfrac{p - p_o}{0.5\,\rho\,U_o^2}$ around airfoil.

Figure 6.21 shows the computed contours of pressure and turbulent kinetic energy fields around the airfoil. In figure a) it is clearly visible that the continuity of the pressure field is maintained across the whole overlapping interface. The contours of the turbulent kinetic energy are concentrated in the trailing edge area, as a consequence of having recirculation region, where the production of turbulent kinetic energy is the most intensive. In this area the grid spacing on the two grids is considerably different as shown in figure 6.22 a). This is mainly the consequence of the use of a C-type grid around the airfoil. The requirement for having a very fine grid resolution near the wall results in a very fine grid resolution also behind the airfoil. Especially unfavorable is the fact that the cell distribution is fine only in one direction resulting in cells with a very high aspect ratio. Despite that, no discontinuities of the flow variables can be observed in this area (see figure 6.22 b) and c). Slight differences visible in the boundary cells are due to plotting of results.

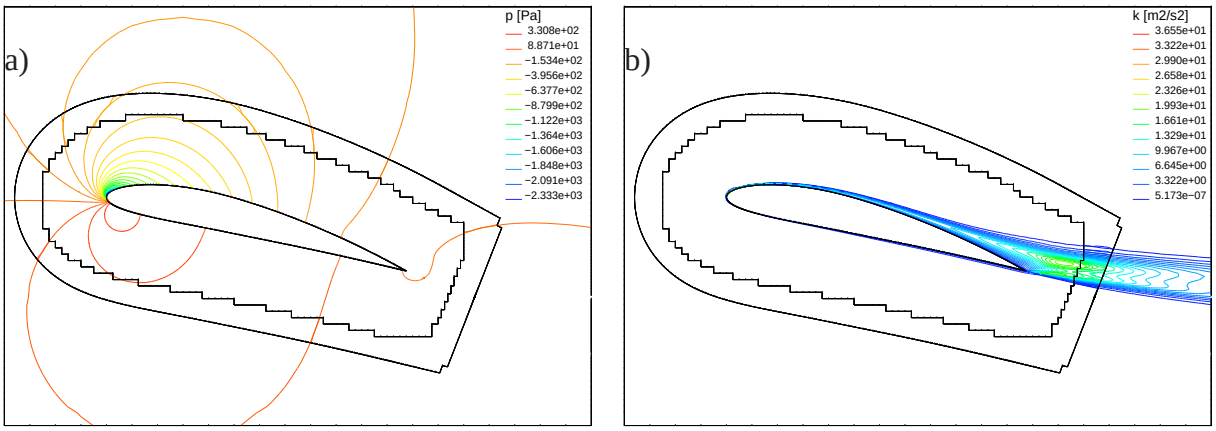


Figure 6.21: Pressure field a) and turbulent kinetic energy field b) around the foil.

Although this case can be readily computed using conventional single grid method, the overlapping grid approach still offers some advantages concerning the grid generation and grid qual-

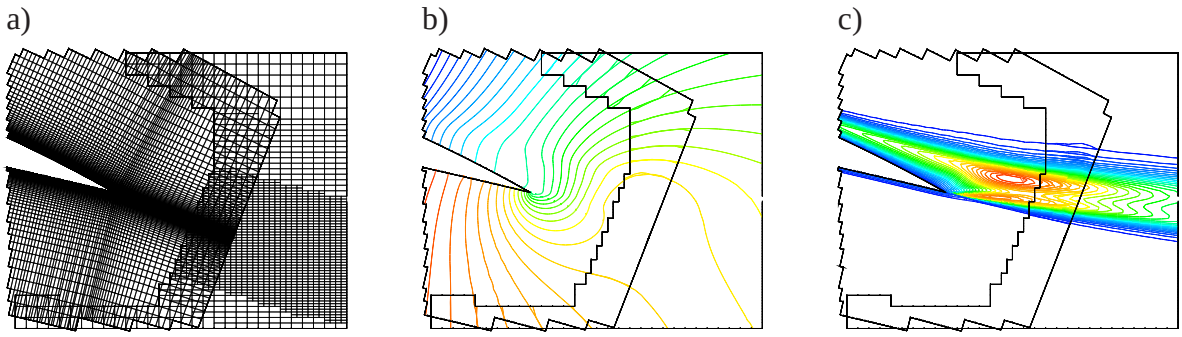


Figure 6.22: Numerical mesh a), pressure field b) and turbulent kinetic energy field c) in the area behind the airfoil.

ity. The advantage of the overlapping grid approach could be especially exploited in the case when for example the computations for different angles of attack would need to be performed in an optimization study, or in the case of a moving airfoil.

This computation has shown that the extension of the method to additional physical complexity is straightforward.

6.4 Flow around rotating plate in a channel

In this section we consider the unsteady flow around a rotating plate in a channel. This example was used as a test case for moving overlapping grids. Problem definition is given in figure 6.23. At inlet, the parabolic velocity profile of a fully developed channel flow was prescribed that corresponds to a mean velocity of $U = 1\,\mathrm{m/s}$. At outlet, constant pressure was prescribed, while at top and bottom channel walls and plate surface, the no-slip wall boundary conditions were applied. The fluid properties used in computation were as follows: fluid density $\rho = 1\,\mathrm{kg/m^3}$ and dynamic viscosity $\mu = 0.01\,\mathrm{Pas}$. The Reynolds number based on the mean inlet velocity U and the channel width H was $\mathrm{Re} = 100$ yielding laminar flow. The plate rotated in clock-wise direction (see figure 6.23) with a constant angular speed $\omega = 2\pi/T$ with period of rotation[6] of $2\,\mathrm{s}$. For this oscillating flow the Strouhal number can be defined as $\mathrm{St} = \omega H/U$ whose value for given parameters is $\mathrm{St} = \pi \approx 3.14$.

This case – although rather simple – was complex enough to test the most important features of the moving overlapping grids which include dynamic change of overlapping region and deactivation and reactivation of cells during the computation. For verification of our results we also performed computations using the commercial program Comet, which employs the sliding interface technique for the treatment of plate motion. The possibility to perform the computation using another available technique was the main motivation to select this example as a test case for the present method.

[6]The time in which the plate rotates by 360^{o}.

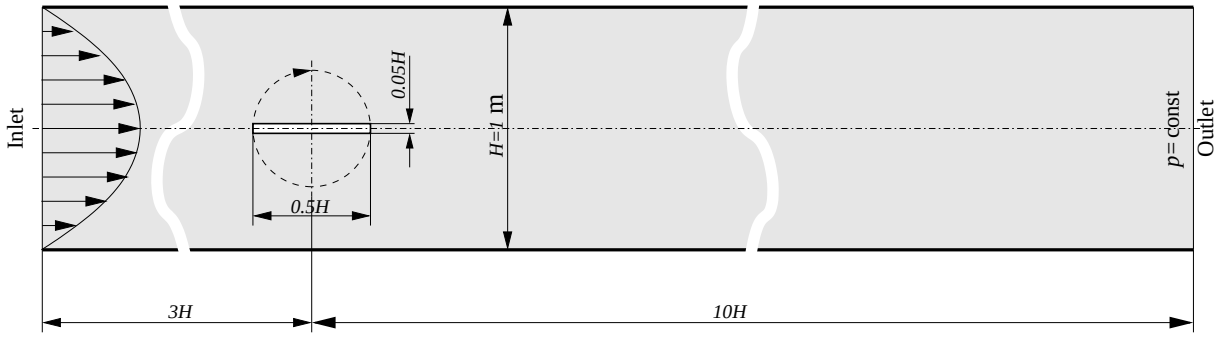


Figure 6.23: Computational domain and boundary conditions for the flow around a rotating plate.

Numerical meshes used for both configurations are shown in figure 6.24. In order to minimize the influence of the grid on results, a part of the grid next to the body and in the region far from it was the same in both configurations. For sliding grid computation, it was necessary to modify the grid around plate to incorporate properly the sliding interface (see figure 6.24 b). Thus, despite a very simple geometry a relatively complex grid that involves severe nonorthogonality had to be used. On the other hand, in the overlapping grid computation it was possible to use two Cartesian grids over the whole computational domain. This demonstrates the high flexibility of the overlapping grid technique regarding grid generation.

Computations were first performed for both configurations using the Euler implicit scheme. Each computation was started from a fully developed laminar channel flow. The periodic solution was obtained after the first cycle and thereafter flow repeats with a period of 180^{o} of the plate rotation. Convective fluxes were discretized using the CDS scheme and the ALE approach based on space conservation low, which is employed in Comet, was used for the grid movement. The under-relaxation factors were higher then in previous steady computations: 0.9 for the momentum and 0.5 for the pressure-correction equation. Convergence tolerance was set to 10^{-4}, requiring on average $14 - 18$ outer iterations per time step. The time step $\delta t = T/400 = 0.005$ s was used, which provided that the local values of Courant number were close to unity and the grid did not move more than one cell per time step.

Several characteristic parameters of the flow were recorded and used for the comparison purposes. Figure 6.25 shows the time variation of moments of pressure and shear forces on the rotating plate, as computed with the present method on overlapping grids and with the commercial code Comet using sliding grid technique. Only for the moments of shear forces, which are an order of magnitude smaller then the moments of pressure forces, are small differences between the two solutions noticeable. These are probably due to the fact that the grid is not identical in the plate vicinity, since the sliding grid technique requires a circular block around the plate.

In figure 6.26 a comparison of velocity profiles at $x = 0.5\,H$ behind the plate at four time instants (corresponding to plate angles of 45^{o}, 90^{o}, 135^{o}, and 180^{o}) is shown. The differences between the present solution and Comet solution are again small and of the order of discretiza-

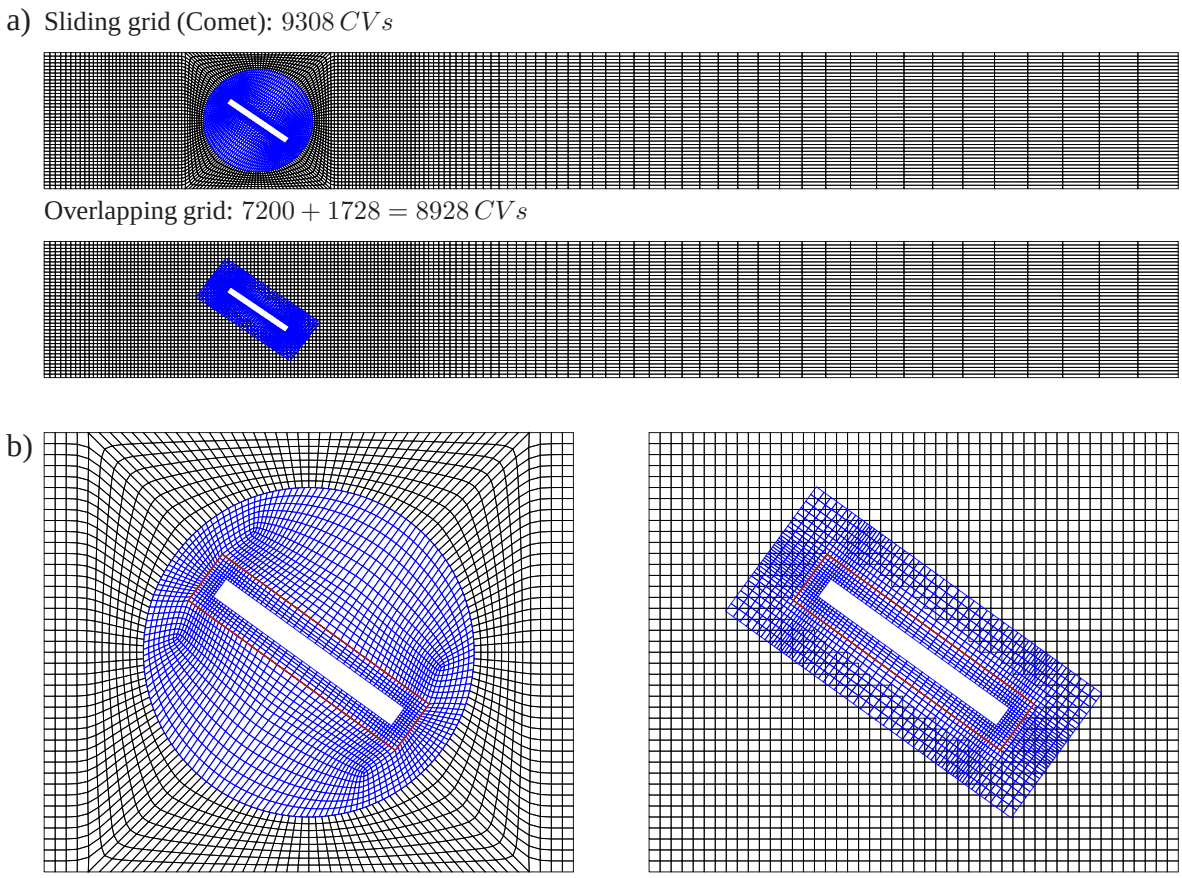


Figure 6.24: Sliding and overlapping grid used for the computation of flow around a rotating plate: a) numerical mesh over a major part of the computational domain, b) a detailed view of the mesh in the plate vicinity.

tion errors.

In order to asses the discretization errors, simulations with the present method using overlapping grids have been performed on two more grids (one coarser - g1 and one finer - g3). The grids were refined systematically, increasing the number of cells on each finer grid four times in comparison to the next coarser grid. Resulting number of cells on grids g1, g2 and g3 was 2232, 8928 and 35712 respectively. Figure 6.27 shows time history of shear force on plate in x and y direction as computed on all three grids. One can see from these diagrams that the difference between results obtained on the two finer grids, g2 and g3, is very small - most of the time the two curves cannot be distinguished from one another. The maximum difference at profile peaks is of the order of 2%, but only for a short period of time. Also, one can clearly see that the difference between g2 and g3 is about four times smaller than between g1 and g2, indicating a second-order convergence toward a grid-independent solution, as expected of a solution method based on second-order approximations.

To asses the influence of the time discretization scheme, the computations have also been

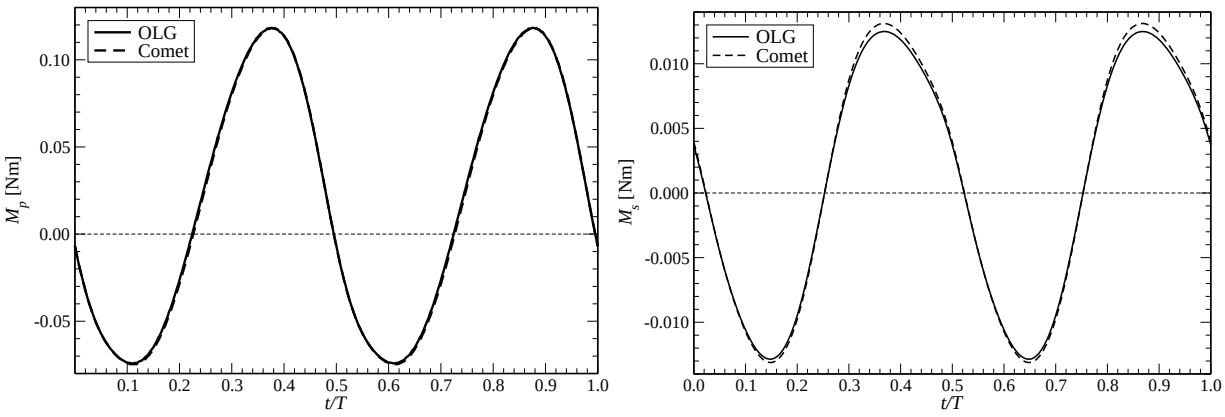


Figure 6.25: Variation of the moments of pressure forces (left) and shear forces (right) acting on the rotating plate with time, calculated using overlapping (OLG) and sliding grid technique (Comet).

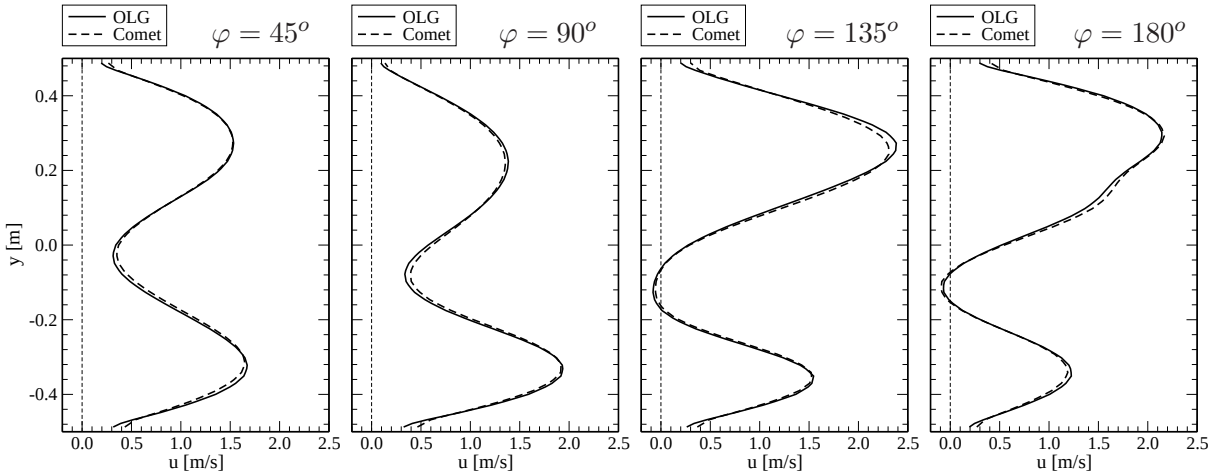


Figure 6.26: Profiles of the u velocity at the position $x = 0.5\,H$ behind the plate at four time instants $0.125\,T$, $0.25\,T$, $0.375\,T$ and $0.5\,T$, from left to right, respectively; comparison of results obtained using present method with overlapping grids (OLG) and sliding grid technique (Comet).

performed on the medium grid using the implicit three time levels scheme (ITTL) with the same time step. Figure 6.28 shows the pressure and shear forces in x direction exerted on the plate, obtained using the Euler and the ITTL scheme on the medium grid. Slight difference can be observed in the shear force, while the pressure force is almost identical. This suggests that the influence of the time discretization scheme for the given conditions (numerical mesh and time step) is small. This can be explained by the fact that the flow is driven by the plate motion, which does not depend on the flow field. Since the plate position is exactly defined at each time step, the errors cannot accumulate during the time. When the body motion is caused by the flow field (e.g. floating bodies, or flow-induced vibrations, etc.), the choice of the discretization scheme in time may be important. In such cases a higher-order scheme in time might be necessary to achieve the desired accuracy; otherwise, the error can accumulate during the time and inaccurate results may be obtained. This issue, however, was not investigated in detail in the present study.

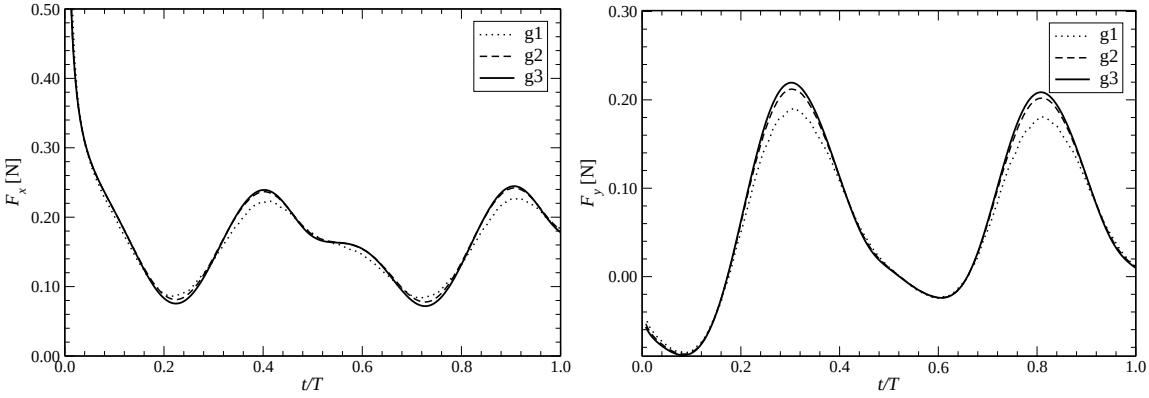


Figure 6.27: Time history of shear forces exerted on the plate, computed on three systematically refined overlapping grids. Results from the first cycle are shown, which are still not periodic.

The accuracy in time can be assessed in the same way as in space by a systematic refinement of time steps. Further computations with two smaller time steps have been performed on the medium grid using both schemes. These results, however, did not show significant changes. Differences between results obtained with different time steps were smaller then differences obtained using different schemes and are not shown here. The time discretization errors were found to be much smaller then the space discretization errors on the same grid. Further investigation on a finer grid were not performed because it would require much more computational effort.

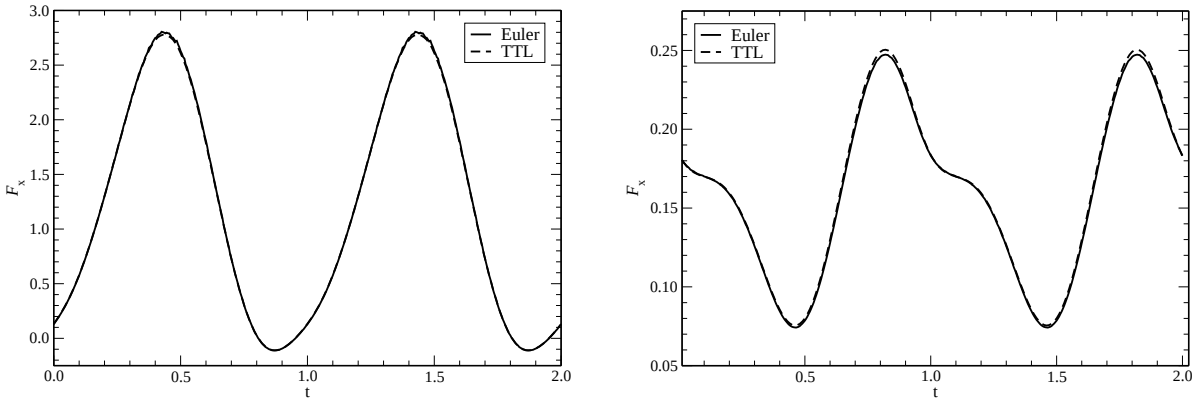


Figure 6.28: Pressure force (left) and shear force (right) in x direction exerted on the plate, obtained using the Euler and the three time levels scheme on the medium grid and with time step $\delta t = T/200$.

6.5 Flow around two rotating plates in a channel

This case was selected to demonstrate the capability of the overlapping grid method to solve problems with complex multiple body motion, which cannot be handled with sliding interfaces available in commercial software. In this case two plates are rotating in the opposite directions, see figure 6.29. The flow conditions and the grids attached to the plates were the same as in

the first example ($Re = 100$, $T = 2$ s). The background grid was modified in order to cover properly the region around the second plate, which was placed at $0.35\,H$ behind the first one. The computation has been performed only on the medium grid, which had 10656 cells. Other computational parameters (differencing schemes, convergence tolerances, under-relaxations and time step) were the same as in the previous example.

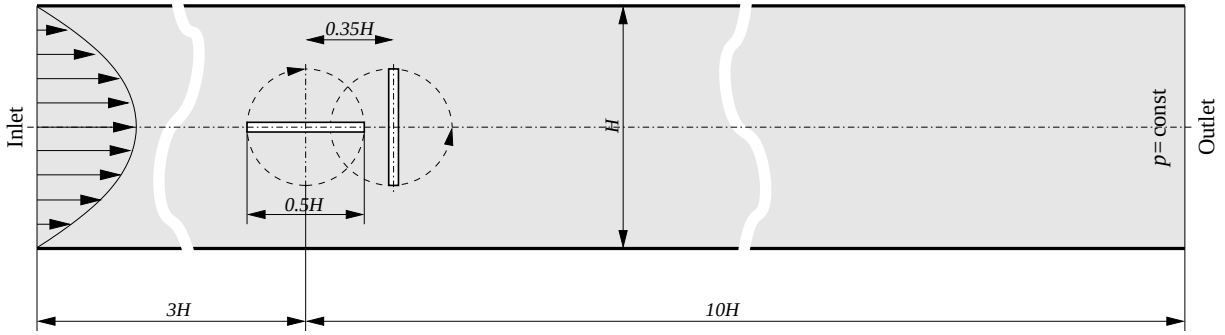


Figure 6.29: Computational domain and boundary conditions for the flow around two plates rotating in a channel.

The plates are relatively close to each other so that their paths intersect during the rotation. The grids attached to the plates overlap not only with the background grid but also with each other. This results in complex multiple overlapping regions involving more then two grids overlapping with each other (see figure 6.30). The major difference to the previous example is that in this case some parts of each body grid come also during the rotation outside the physical domain (i.e. they penetrate the other plate). Hence, the cells in moving parts of the grid have to be de-activated, see figure 6.31. Such a computation would be very difficult to perform using any technique based on body-fitted and deforming grids. Alternatively, re-meshing technique could be used, which introduces extra costs due to grid re-generation and data transfer from the old to the new grid. An additional problem is that automatic grid generation for complex geometries is still prone to errors and often requires manual intervention; also generation of grids consumes a substantial amount of time and is not so easily automated.

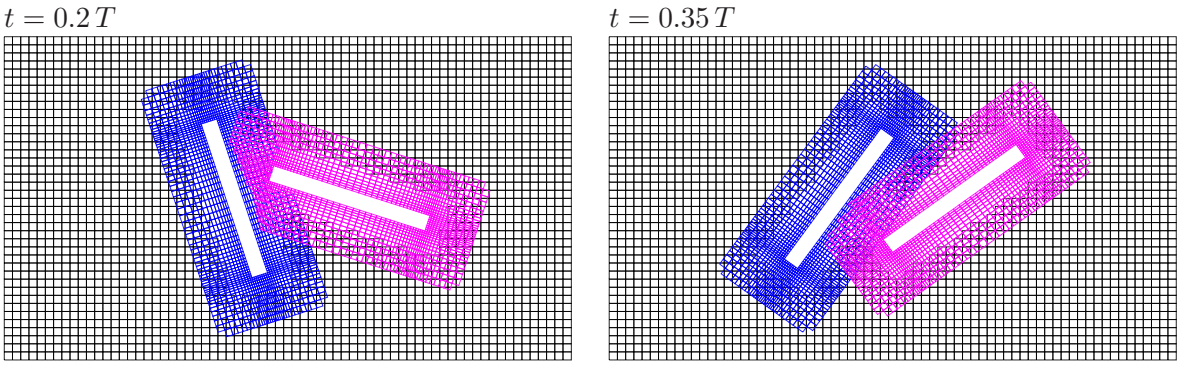


Figure 6.30: Overlapping grid for the flow around two rotating plates at two different positions.

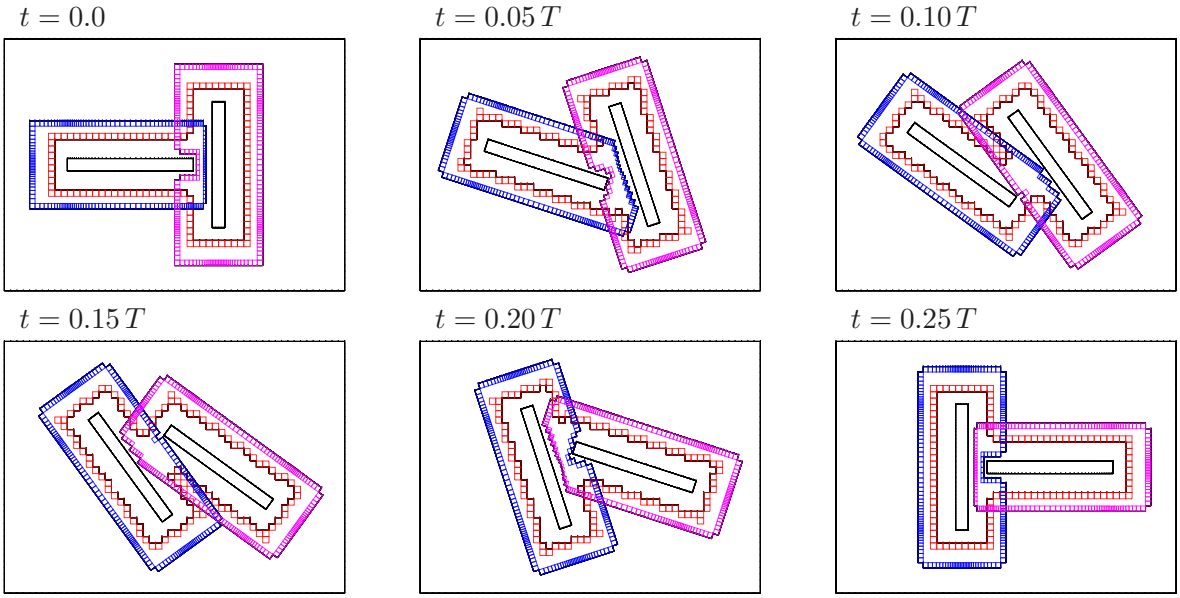


Figure 6.31: Changes in interface topology for the problem with two plates rotating in the opposite direction at six positions.

Figure 6.32 shows the contours of the pressure and the u velocity component in the vicinity of plates at three different times. Despite the very complex overlapping configuration, with multiple overlapping and dynamic changes of the overlapping region, which require de-activation and re-activation of cells on moving parts of the grid, the continuity of the flow field across overlapping interfaces is obviously maintained. The continuity of contour lines across the overlapping interfaces indicates that the coupling between grids is correct and accurate. Animations of mesh motion, pressure and velocity vectors during one rotation are available on the attached CD-ROM.

This example demonstrated the versatility of the method in handling very complicated relative motion of bodies, using at the same time, high-quality meshes and producing accurate solutions.

In the computations presented in section 4.4.1 it was demonstrated that the new method for handling of grid movement produces almost the same results as the standard method based on the arbitrary Lagrangian-Eulerian approach and use of the space conservation law. In these computations, however, the grid movement was only in one direction, which may be considered as a special case rather then a general one. In order to asses the performances of the new method when the grid motion is more general, the present case has also been computed using the LTD method for grid movement. The comparison of the moments of shear forces exerted on the two plates, for which the largest differences between two methods have been obtained, is shown in figure 6.33. Although slight discrepancy may be observed around extrema, the agreement may be considered as very good. Since the SCL method has in the previous example been proven to produce correct results, herewith it was confirmed that the LTD method also produces correct results for a general grid motion.

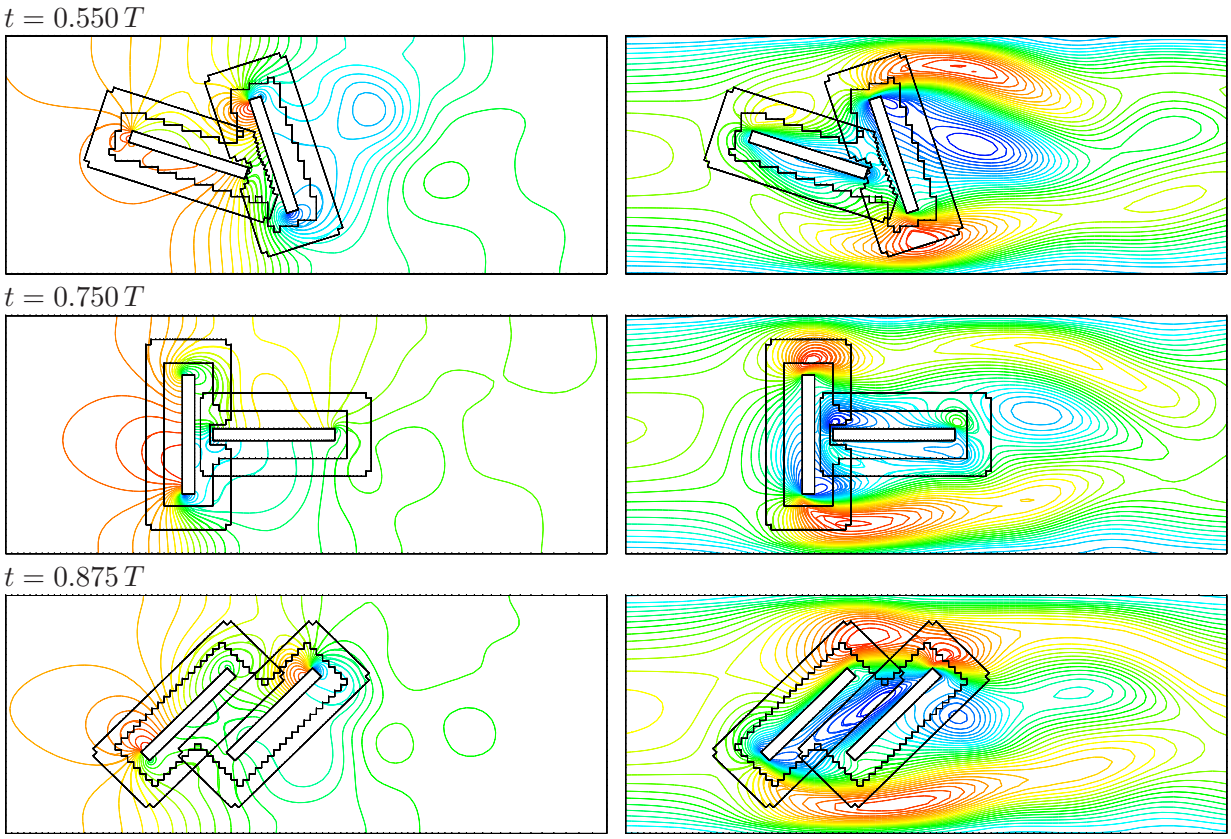


Figure 6.32: Contours of pressure (left) and u velocity (right) at three characteristic positions.

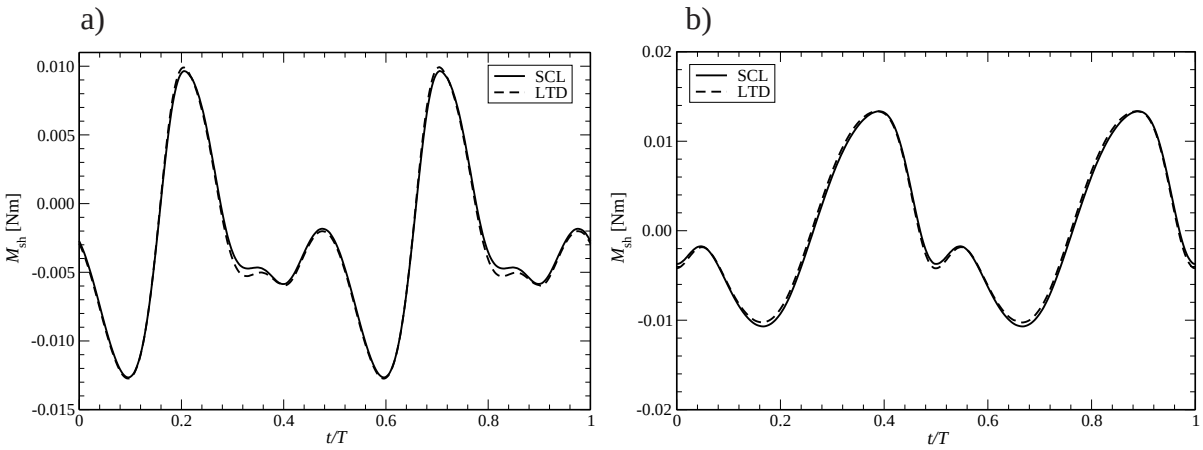


Figure 6.33: Time history of the moments of shear forces exerted on the plate one (a) and plate 2 (b) computed using SCL and LTD techniques for grid movement.

6.6 Flow in a mixer

As the next example we consider the flow in a closed box driven by two moving plates. Due to similarity to a mixing device we call it mixer. The problem definition for this case is given in figure 6.34 a). Two plates whose dimensions are 100×10 mm are placed in a square box whose side length is 340 mm. The box is filed with a viscous fluid with $\rho = 1000$ and $\mu = 0.01$. Initially the fluid and both plates are at rest and the plates are in the horizontal position as shown in figure 6.34 a). Then, they start to rotate with angular speed ω around their own axes in opposite directions: plate 1 (left in figure) in counter clock-wise and plate 2 (right in figure) in clock-wise direction. At the same time they start to rotate in clock-wise direction around the global center placed in the middle of the box with the angular speed of $\omega_0 = \omega/2$. That means, while the center of each plate makes a full cycle, each plate rotates twice around its own axis. The periods of rotations were $T = 1\,\mathrm{s}$ and $T_0 = 2\,\mathrm{s}$. Corresponding angular speeds can be computed as $\omega = 2\pi/T$.

The overlapping grid system used in this computation is shown in figure 6.34 b). Two Cartesian grids,which cover the area around plates, are attached to each plate and move together with them, while a triangular grid that covers the whole box is used in background. Triangular grid is used mainly to demonstrate the capability of the method to deal with meshes of arbitrary topology. This feature of the method has already been demonstrated in section 6.2; here we show that the method works correctly also with moving meshes.

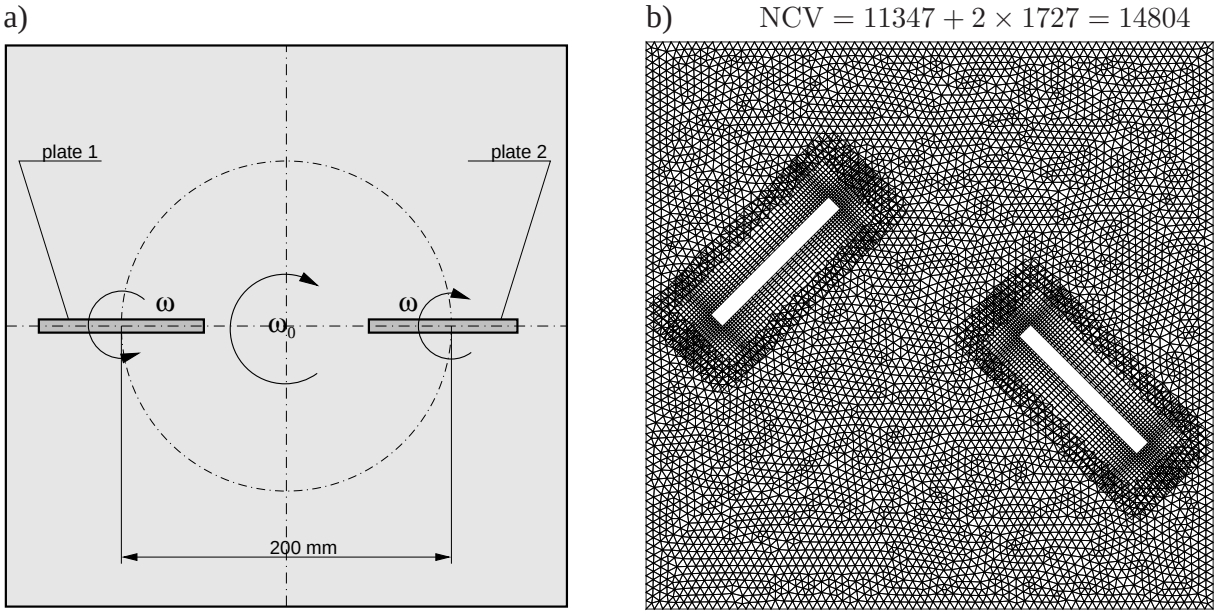


Figure 6.34: Problem definition a) and the overlapping grids b).

In comparison to the previous two examples, this case involves two important new features. First, the body motion is in this case much more complicated involving two motions combined with each other. The resulting motion is rather complex and large in comparison to the size of

the bodies. During their motion the plates sweep a relatively large area in the background grid. Thus, a large number of grid cells is involved in the inter-grid communication requiring either to be temporarily deactivated or to be used for inter-grid interpolation. Grids attached to plates are always far enough from each other so that they do not overlap with each other. However, since the plates at some positions come close to box walls, some parts of the overset grids come outside of the background grid, i.e. outside the computational domain, and thus have to be temporarily deactivated.

Second important feature of this case is that the flow takes place in a closed domain. Therefore strict mass conservation is essential for the solution. This is achieved by applying the conservative correction for mass fluxes across interfaces, as described in section 5.3.3. Since the grid movement is already treated conservatively (irrespective of whether single or overlapping grids are used), additional correction applied to the interface mass fluxes provides the required level of mass conservation on overlapping grids.

The computation was performed using the implicit Euler scheme with a time step of $\delta t = 0.0025\,\mathrm{s}$ which corresponds to $1/800\,T_0$ and two cycles[7] were computed. The grid movement was treated using the ALE approach. Figure 6.35 shows instantaneous flow fields in terms of velocity vectors and pressure contours at three times. Due to motion of the plates, complex unsteady flow structures characterized by many vortices are generated. The fluid is pushed away from each plate in the direction of their motion, resulting in high pressure and velocity vectors pointing out from the plate surface. Behind each plate the fluid is sucked to fill the space previously occupied by the body. This results in low pressure and velocity vectors pointing towards the wall. As a result of low pressure, vortices are generated behind each corner. Although strong pressure gradients are present in the area close to the plates, a very good agreement in the pressure field on the background and overset grids can be observed.

For the purpose of flow visualization, an equation of the form (2.9) for the transport of a passive scalar was solved in addition to momentum and continuity (pressure-correction) equations. Initially the scalar concentration was set to $\phi = 1$ in the lower half of the box and $\phi = 0$ in the upper half of the box. An animation showing the scalar concentration during two cycles and thus visualizing the mixing process is available on the attached CD-ROM.

[7] One cycle corresponds to period of rotation T_0 around the box center.

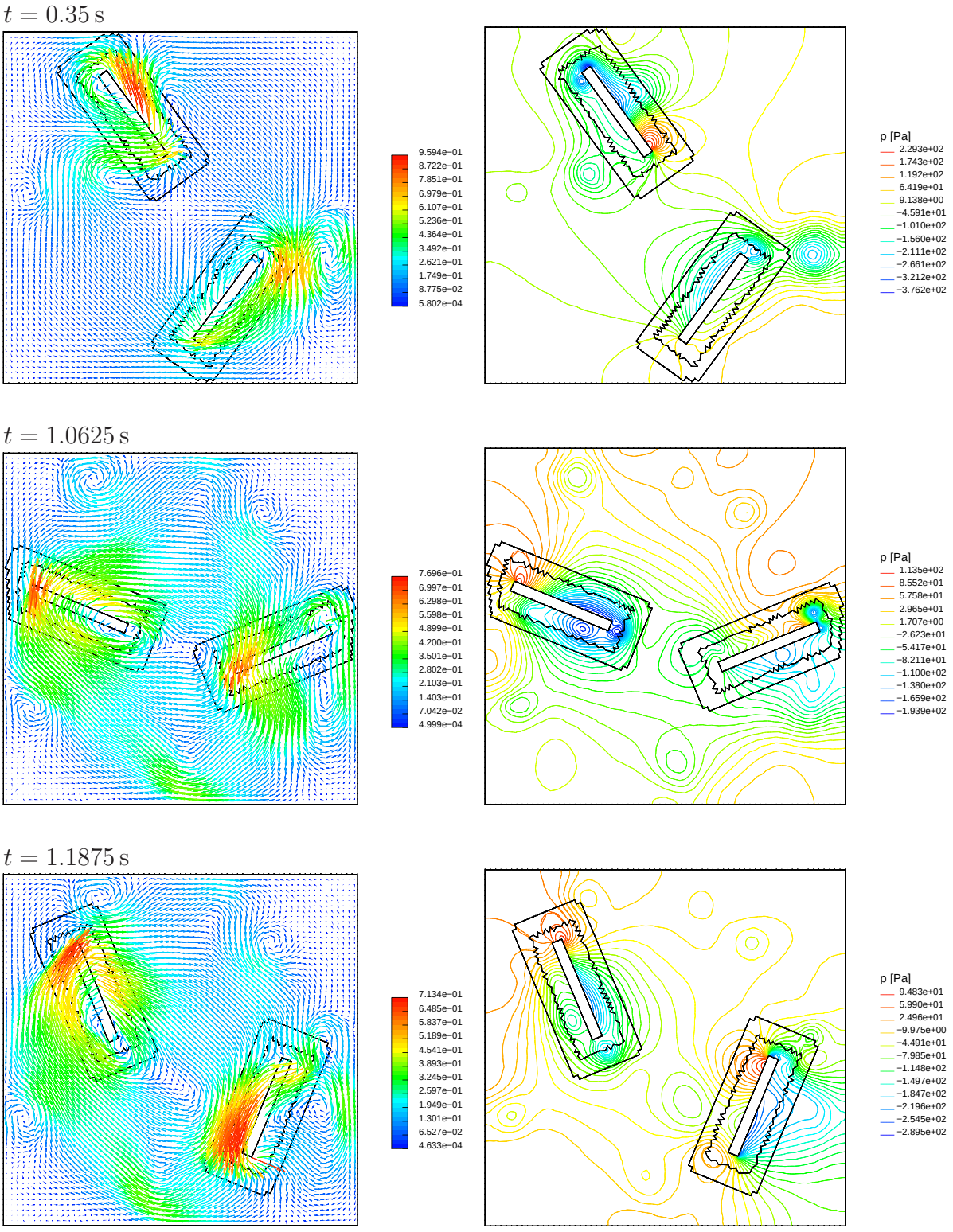


Figure 6.35: Instantaneous velocity (left) and pressure (right) fields at three times.

6.7 Flow around the Voith-Cycloidal-Rudder

In this section we consider the flow around a two-dimensional model of the Voith-Cycloidal-Rudder. This case was selected as an illustrative practical example that involves multiple moving bodies, where the flexibility of the overlapping grid approach in handling of such problems can be best demonstrated.

The Voith-Cycloidal-Rudder (VCR) has been recently developed by the Voith company as a new technique for ship maneuvering. The VCR is based on the same principle as the Voith Schneider Propeller (VSP) but unlike the VSP, which has five or more blades, the VCR has only two blades. With two working modes, passive and active, the VCR provides a high maneuvering performance of the ship. In the passive mode, while the ship operates at a sufficient speed, the blades of the VCR operate like standard rudders. The thrust necessary for the navigation of the ship is produced by changing the position of the whole rudder relative to the main flow direction. In the active mode, at very low speeds or stationary ship, the VCR operates like a cycloidal rudder providing the thrust in any direction. Thus, a very high manoeuvring performance of the ship is achieved also at low speed. Moreover, the VCR can be used as an alternative power unit in the case of failure of the main power unit.

6.7.1 Problem definition

The problem definition is given in figure 6.36, which shows the rudder in the starting position defined by $\theta = 0^o$. The rudder, which rotates with $250\,\mathrm{rpm}$ ($\omega = 26.18\,\mathrm{rad/s}$) in counter-clockwise direction, is placed in a free water stream of $1.83\,\mathrm{m/s}$ coming from left. While the

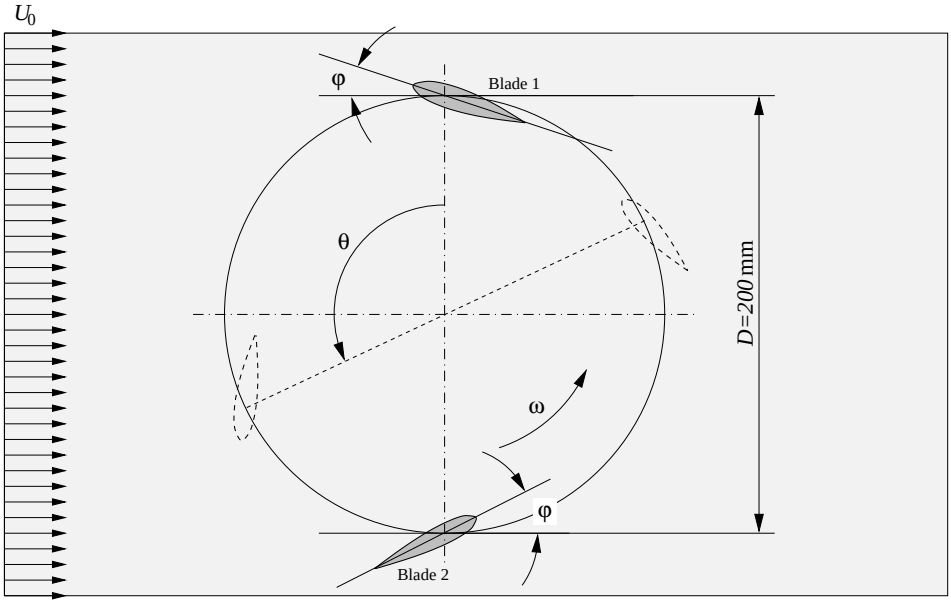


Figure 6.36: Problem definition for the flow around the Voith-Cycloidal-Rudder.

entire rudder rotates at a constant angular speed around its own axis, the rudder blades make at the same time an additional oscillatory motion around their own axes. This motion is characterized by the angle φ, which defines the position of the blade in relation to the tangent on the circular path along which the blades are moving (see figure 6.36). The rotation center of each blade is placed at one third of the chord length measured from the front side of the blade profile. The blades are represented by standard NACA0012 airfoil. The variation of the angle φ and the blade position during a cycle are given in figure 6.37. Positive values of the angle φ are in the area where the front side of the blade profile is outside and the negative values are in the area where the front side of the blade profile is inside the circle of the blade motion (see figure 6.37).

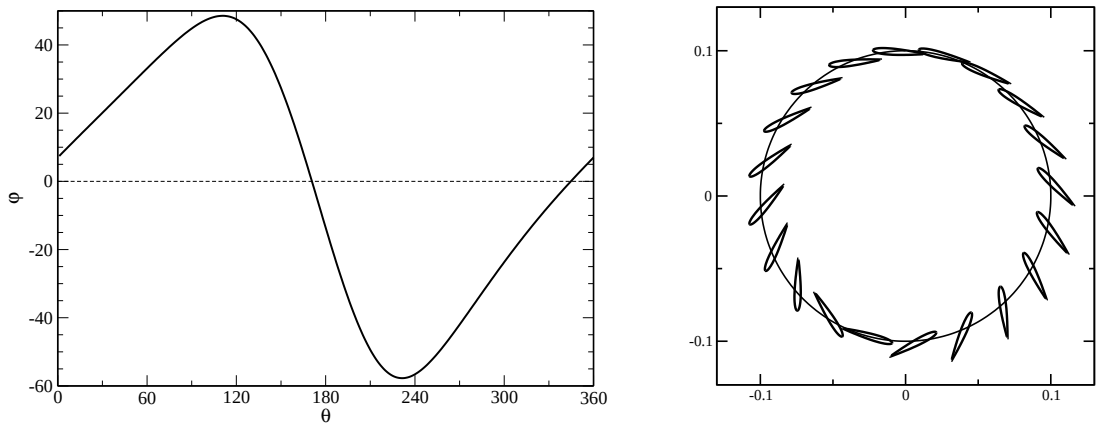


Figure 6.37: Variation of angle φ (left) and position of a blade (right) during a cycle.

6.7.2 Numerical grid and boundary conditions

The overlapping grid system used in this computation consists of three component grids: the background grid stretched over the whole domain and two grids generated around each blade covering the area close to the blades (figure 6.38).

As in the example with NACA 4412 airfoil, an orthogonal and locally refined background grid was used also in this case. Starting from a very coarse initial grid with $16 \times 24\,CVs$, several levels of local grid refinement were applied until the required grid resolution in area in the ruder vicinity was achieved. Final refinement encompasses an annular zone close to the circle path of the blade movement. Thus, the area in which the rudder blades move and where the changes in the flow field are strongest is covered by a reasonably fine grid. The grid resolution in this area is comparable with the grid resolution on the edge of the blade grid, as shown in figure 6.39. The final number of cells in the background grid was 31161. A C-type grid, which is commonly used for such a geometry, was generated around each blade. This grid had $84 \times 15\,CVs$ whereof $76\,CVs$ were distributed along the blade surface. The grid extends 0.3 chord length in radial direction and behind the profile and is stretched towards the blade surface. Note that, this configuration can be easily extended to the one with more blades by simply adding as

many blades as required and putting them to a right position, which may reasonably speed up the designing process.

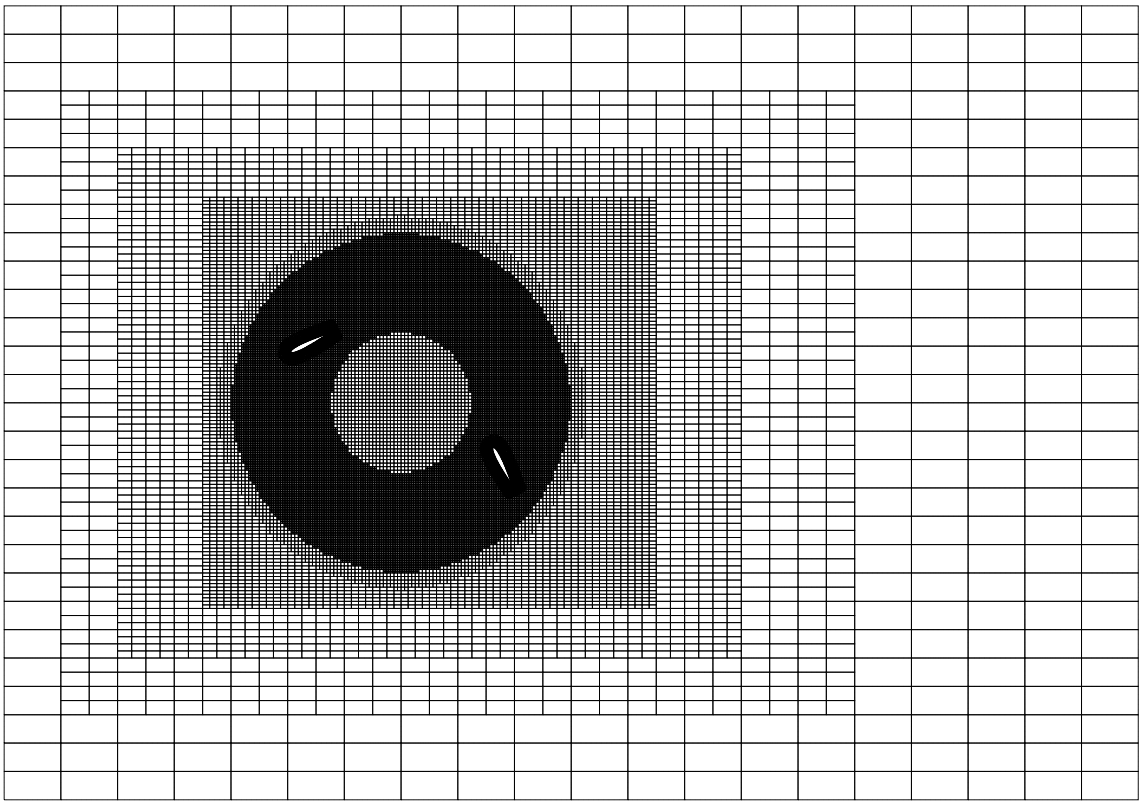


Figure 6.38: Overlapping grid system consisting of three grid components used for the computation of flow around the VCR (total number of cells: $31812 + 2 \times 1320 = 34452$). Only a part of the background grid for $0.00 < x < 0.000$ and $0.00 < y < 0.000$ is shown.

Because the location of the two foils changes in time, the background mesh was refined in the whole area swept by the overlapping grids. Further saving in memory and computing time could be achieved if the refinement was dynamic, because the mesh needs to be so fine only in the vicinity of the rudder profile. This, however, is a general issue of the desired dynamic mesh refinement and coarsening, which is not specific to overlapping meshes and due to lack of time could not be implemented in this study. Another possibility may be to have an additional orthogonal grid for each blade, which might be seen as a carrier of the blade grid. This grid would serve as bridge between the blade grid and the background grid and would move according to the blade movement.

The inlet boundary with prescribed constant free-stream velocity $u = 1.83\,\mathrm{m/s}$ was placed $3D$ far upstream from the rudder center. Outlet boundary was moved $5D$ far downstream and standard outlet boundary condition was applied there. The upper and lower boundaries, which were considered as symmetry planes, were placed $3D$ far above and below from the rudder center.

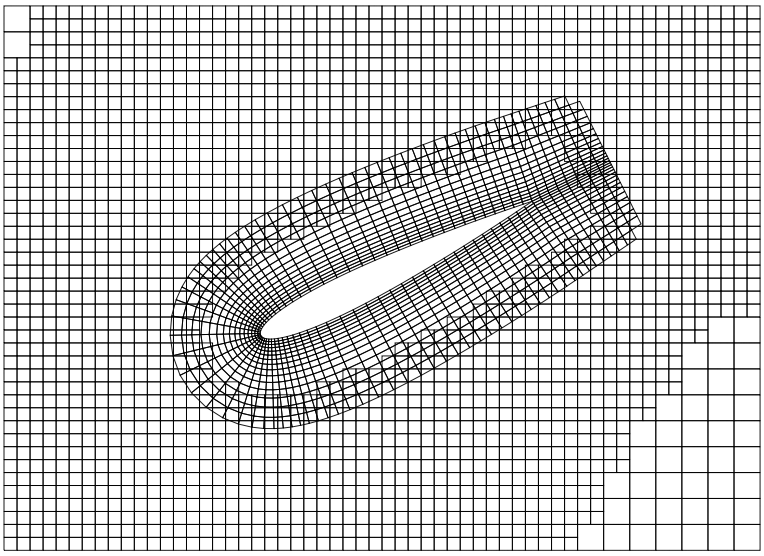


Figure 6.39: A detailed view of the numerical mesh around the blade profile and the overlapping region between the profile grid and the background grid at position $\theta = 60^o$.

At the blade surfaces the velocity resulting from the complex motion of blades was prescribed. This velocity is obtained as the sum of two components (see figure 6.40): velocity $\mathbf{v}_0$ due to rudder's rotation and velocity $\mathbf{v}_{P_1}$ due to oscillatory motion of the blade.

6.7.3 Results

The working fluid in this case was water with the density $\rho = 1000 \ \ \mathrm{kg/m^3}$ and the dynamic viscosity $\mu = 0.001 \ \ \mathrm{Pas}$. The computation was performed using the implicit Euler method for time discretization with a time step of $0.004 \ \mathrm{s}$ which is equal to $1/600\,T$, where T is the period of one revolution. For spatial discretization the UDS scheme was used. The flow field was initialized with free-stream conditions and the computation was performed for five revolutions. The solution showed periodic behavior from the third revolution onward. Results from the last revolution computed are used for assessment.

The attention here was mainly focused on the numerical issues related to the use of overlapping grids, especially on the assessment of the inter-grid coupling and maintenance of the flow continuity in such a realistic situation. Also, since no experimental or other reference data is available for comparisons, no attempts were made to obtain grid-independent solutions. The mesh and discretization schemes used are considered appropriate for demonstration purposes. The turbulence effects on the flow have not been taken into account. The use of a low Re model, available in the present code, in this case would require very fine grid close to the blade surfaces, which in turn would require a very fine time step in order to capture properly the blade movement. This would enormously increase the computational effort, which due to limited time and

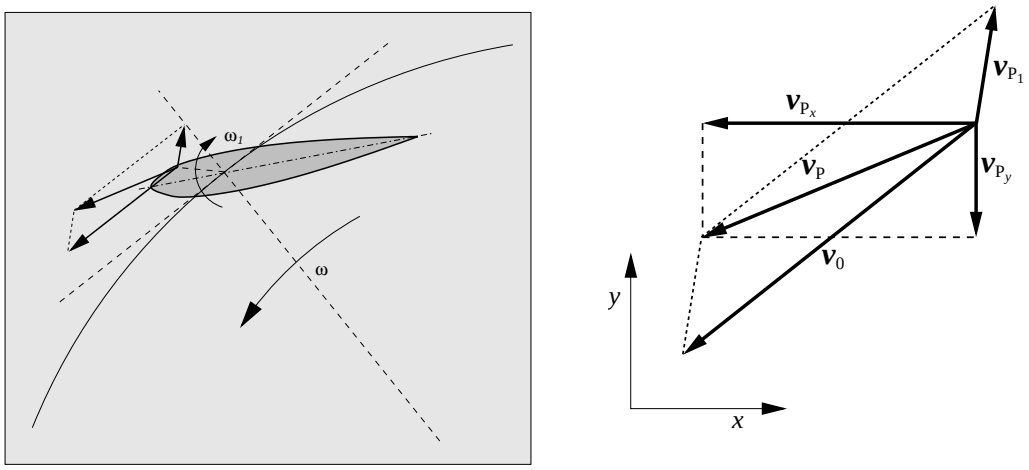


Figure 6.40: Boundary conditions for velocity components at the blade surface.

computer resources was not acceptable [8] in this study. The flow is mainly driven by the pressure, so the turbulence is not expected to be decisive in this case. Despite these simplifications, we expect that the most important features of the flow, relevant for the ruder performance, can still be captured.

Similarly as in the previous example the body motion is in this case also rather arbitrary and, which is especially important, large in comparison to the body size. This is a great challenge for the present method, since we consider the capability for handling of problems with large-scale body motion as one of its most important features.

Figure 6.41 shows instantaneous overall flow field in term of the velocity component in the free stream direction for three ruder positions: $\theta = 60^o$, $\theta = 90^o$ and $\theta = 120^o$. Strong unsteady features of the flow caused by the motion of ruder blades, which change their relative position to the free stream and to each other, can be observed. At certain periods one blade is in the wake of the other one, and the flow field around the blade in the wake is more complex than around the other blade. Therefore this problem has to be considered as a whole.

Figure 6.42 shows instantaneous pressure field for the same three ruder positions as in the previous figure : $\theta = 60^o$, $\theta = 90^o$ and $\theta = 120^o$. Both blades are presented in this figure for these three positions, as indicated. At the same time, due to the periodicity of the flow, these three positions correspond to $\theta = 240^o$, $\theta = 270^o$ and $\theta = 300^o$ of the blade # 1. In this way results for one blade at 6 positions are presented. The close-up of the pressure field in the vicinity of the blade is shown. The perfect continuity of the results over the overlapping regions demonstrates the correctness of the method.

The corresponding fields of the relative velocity as would be observed by an observer who

[8] In order to take the turbulence effects into account, a high Re model with wall functions would be appropriate in this case. However, since the turbulence model has been implemented after the computations of the flow around the VCR have been almost finished, due to lack of time, the implementation of wall functions in the present model has been abandoned.

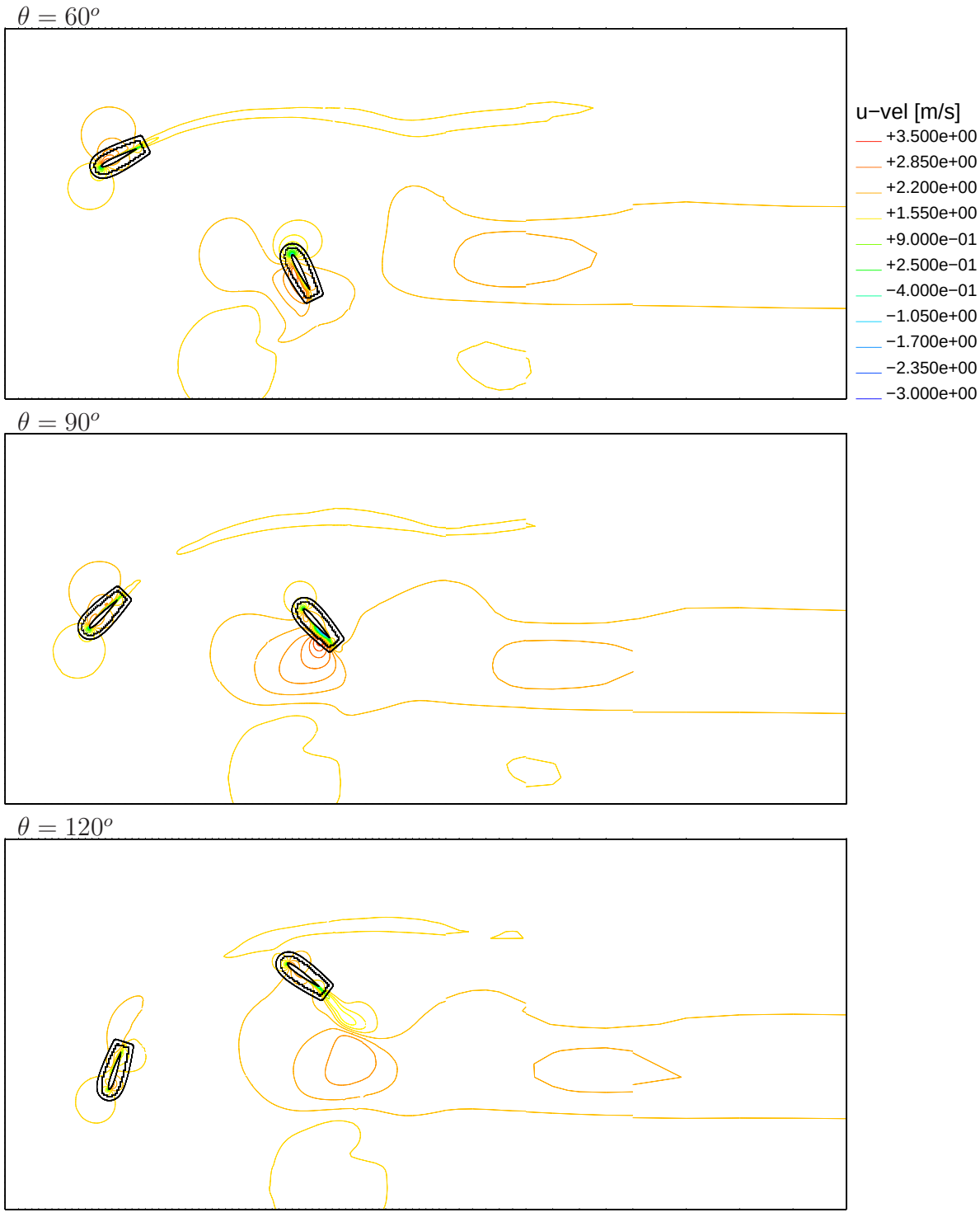


Figure 6.41: Instantaneous velocity field in terms of the u velocity component at three ruder positions.

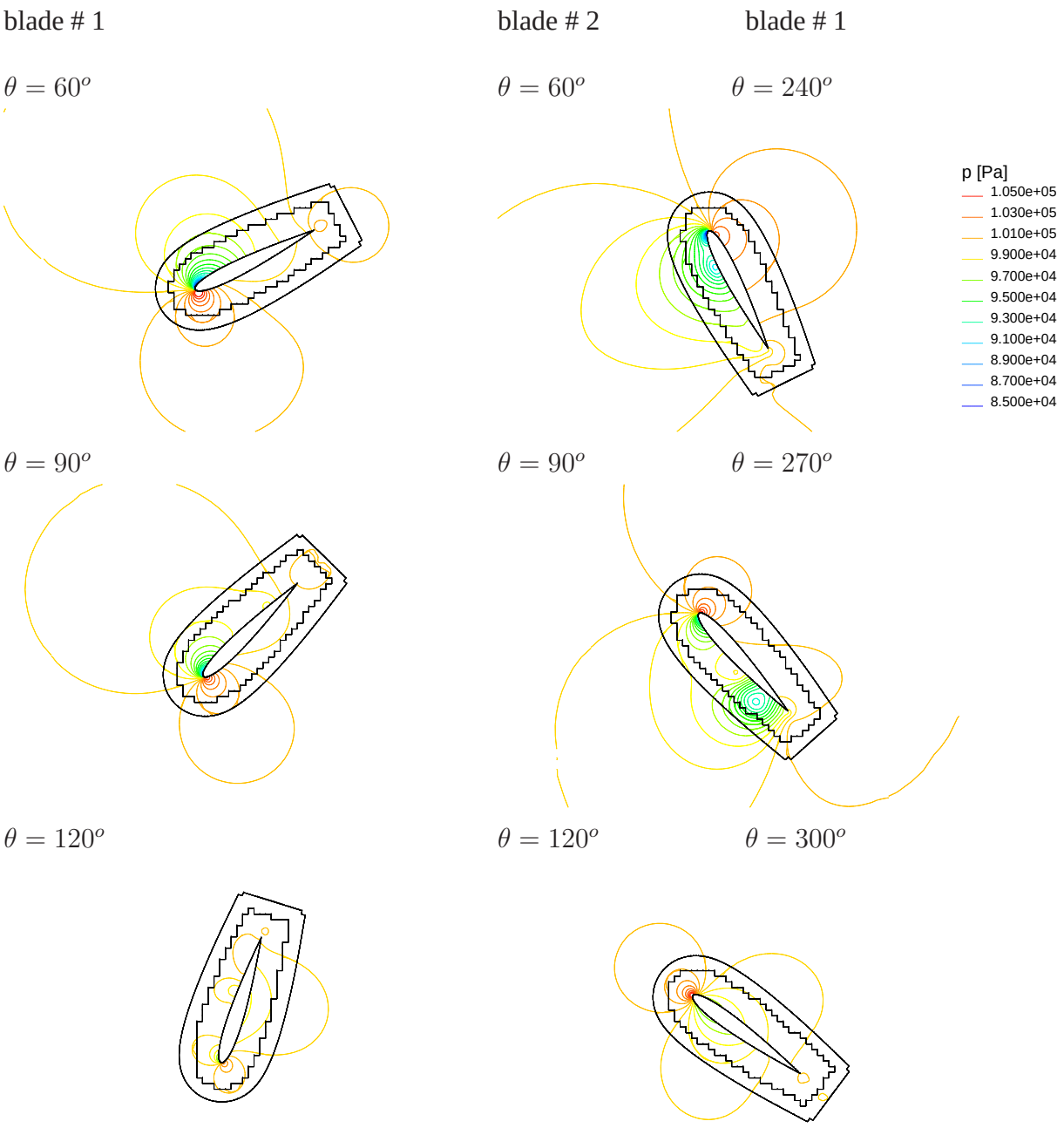


Figure 6.42: Pressure field around the blade at positions $\theta = 60^o$, $\theta = 90^o$ and $\theta = 120^o$.

travels on the blade are shown in figure 6.43. A bigger recirculation zone is visible at $\theta = 270^o$. This recirculation is on the corresponding figure for the pressure represented by the low pressure region in this area. Such recirculation zones may indicate a non-optimal flow over the blade,

which is related to hydrodynamical loses and therefore should be avoided if possible.

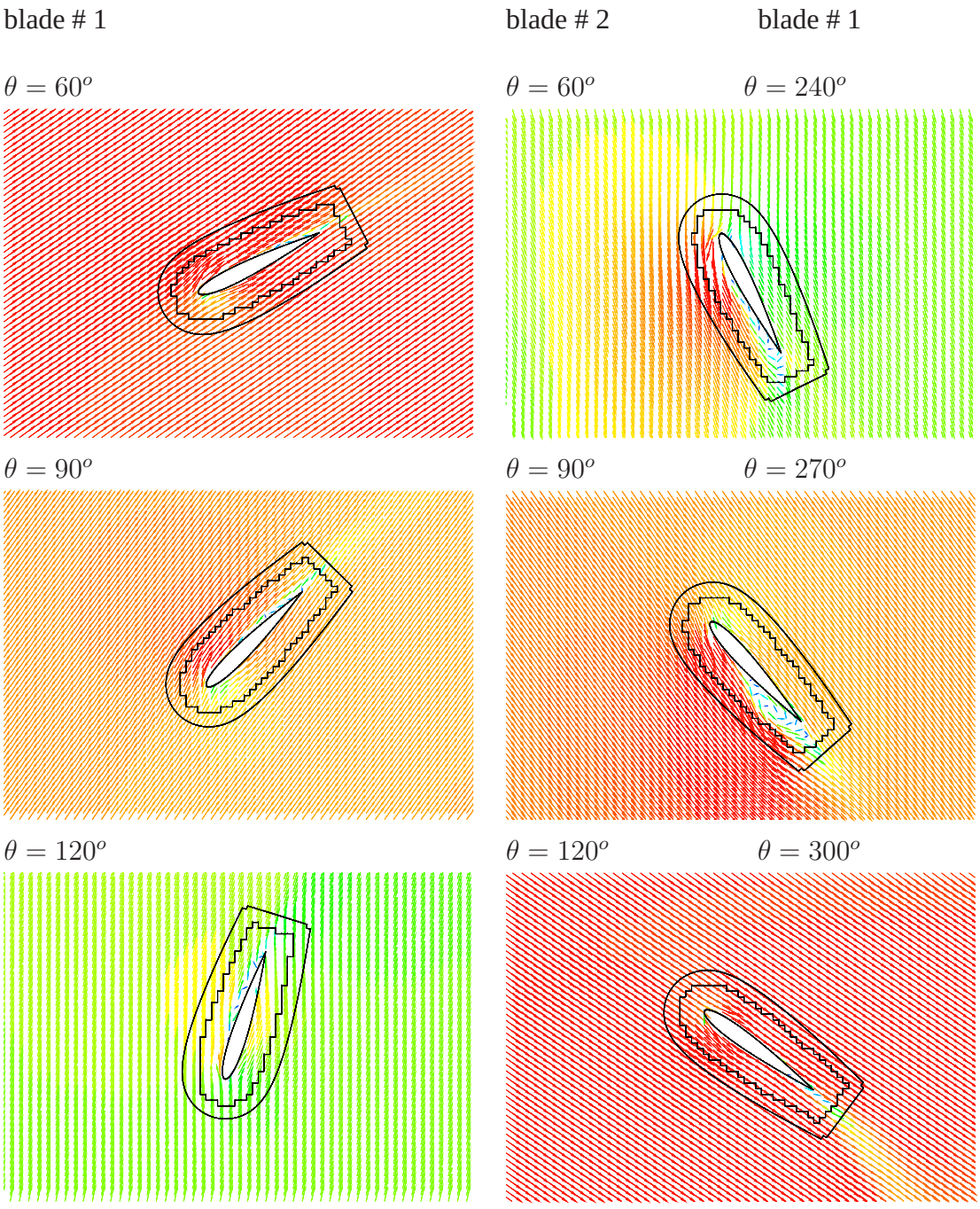


Figure 6.43: Fluid velocity relative to the speed of the blade center at positions $\theta = 60^o$, $\theta = 90^o$ and $\theta = 120^o$.

Figure 6.44 shows the time history of the hydrodynamic loads on a single blade during one

revolution. Figure 6.44 a) shows the integral forces in x and y direction of the Cartesian coordinate system during one period. The integral moment of the forces acting on the blade 6.44 b). These results can be used for assessment of the ruder performance and for the optimization of the ruder design.

We need to note that these integral results show fluctuations in the interval $\theta \approx 90^o$ to $\theta \approx 230^o$. This part of the cycle is critical for the ruder performance. Similar oscillations have been registered in computations of a VSP made by Seibel [66]. He applied the commercial CFD software Comet which uses essentially the same finite volume method and the ALE approach for the grid movement as in the present work. Seibel showed that these fluctuations are stronger as the time step is smaller and the numerical grid is finer. It was beyond the scope of the present work to investigate the cause of these fluctuations or to eliminate them.

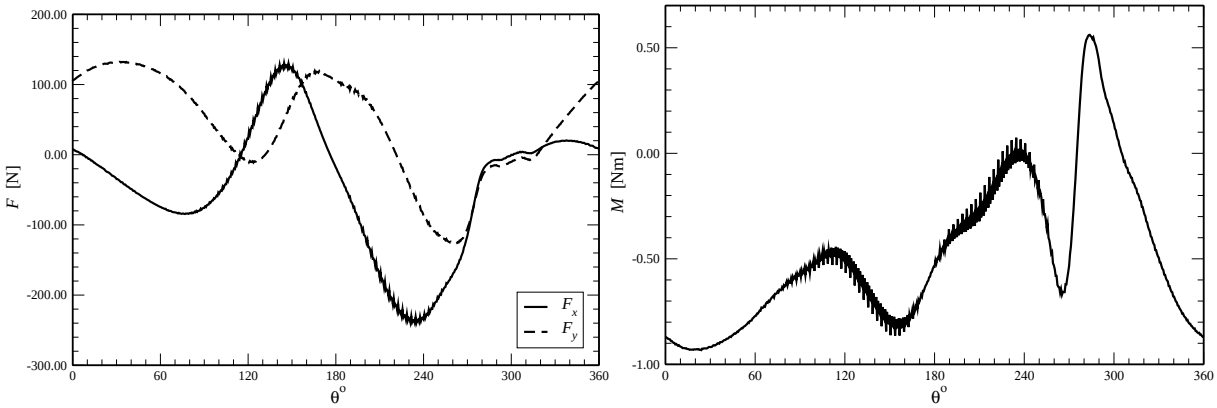


Figure 6.44: a) Integral forces in x and y directions (F_x and F_y respectively) exerted on the single blade; b) integral moment of forces acting on the single blade.

CHAPTER 7

Conclusions and outlook

In the course of the work presented in this thesis a numerical method for the computation of fluid flow around moving bodies using overlapping grid technique and unstructured grids composed of cells of arbitrary topology has been developed. Here, the most important findings and achievements of this work will be summarized.

The transport equations describing the viscous fluid flow were discretized in their integral form using a second-order finite volume method suitable for arbitrary unstructured grids which is nowadays the state of the art in industrial CFD. A computer code for solving these equations on two-dimensional unstructured, overlapping and moving grids has been developed. Appropriate data structures are employed which enable that cells with any number of cell faces can be utilized. The method is general and can be easily extended to arbitrary polyhedral three-dimensional grids.

For the grid movement a novel method based on approximation of the local time derivative (LTD) in the Eulerian formulation of the governing equations has been developed and used besides the standard arbitrary Lagrangian-Eulerian (ALE) approach based on the space conservation law. The new LTD method was found to perform very well in combination with a fully implicit solution strategy adopted in the present study. In LTD method all terms in the governing equations except the unsteady term are approximated using the variable values and the grid data from the current time step. The variable values from the previous time steps but at the new cell-center locations, which are required for the approximation of unsteady terms in the governing equations, are computed by interpolation. The computations done in the present study have shown that the new LTD technique produces solutions of the same accuracy as the ALE approach and with approximately the same computational effort.

In the ALE approach the volumes swept by each cell face from one time step to another need to be computed in an appropriate way to ensure that the space conservation law is satisfied. This, however, may be complicated and time consuming for arbitrary polyhedral cells with a large number of cell faces. On the other hand, the effort required to interpolate values from the previous time steps to the new location of cell center is independent of the grid topology. This makes the LTD method attractive for use with arbitrary unstructured grids. Another advantage of the LTD technique over the ALE approach is that it can be conveniently used in combination with re-meshing. In that case the grid topology (number of cells and their shape) changes so the

volumes swept by cell faces cannot be computed. The interpolation, on the other hand, can still be performed in essentially the same way as when the grid does not change. The computations presented in chapter 4 have shown a good performance of the LTD method in combination with a re-meshing technique.

Among a number of techniques that can be applied to simulations with moving bodies, the overlapping grid technique was selected to be used in the present study as the most suitable one for such kind of problems. In the overlapping grid approach the solution domain is covered by a number of grids which partially overlap with each other. There is no requirement for component grids to match in any way except that they have to overlap sufficiently to provide the means of coupling the solution between them. The overlapping grids allow that independent grids can be created around each body and move with bodies in an arbitrary fashion. No grid adjustments and grid re-generation are required.

The most important task in computations on overlapping grids is to accomplish the inter-grid coupling in an appropriate way. In an iterative solution procedure it is important to consider the solution on all grids simultaneously. Trying to obtain the solution on each grid separately with imposing boundary conditions at grid interfaces leads to a weak coupling between the grids. This tends to result in slower convergence and may lead to a decoupled pressure field in an incompressible flow. In the present study, a special implicit methodology for the treatment of grid interfaces has been developed. The interpolation equations used to compute the solution at interpolation cells are involved in the system of linearized equations arising from discretization. This modified system of linear equations is then solved for the whole domain ensuring that the solution is obtained on all grids simultaneously. In this way a strong inter-grid coupling is achieved, which is characterized by both smooth and unique solutions in the overlapping region and by a good convergence rate which is in the same range as on single grids. This is essential for computations of flows involving moving bodies.

A disadvantage of the overlapping grid technique is the violation of the conservation through the data interpolation across grid interfaces. For the present method it was shown that the strict conservation of mass is only necessary for the solution of the pressure-correction equation when the mass flow rate over the domain boundaries is prescribed, i.e. the Neumenn boundary conditions for the pressure correction are imposed on all boundaries. Such examples are flows in closed domains. The conservation of other quantities is of lesser importance. To enforce the mass conservation on overlapping grids, the interface mass fluxes are adjusted in such a manner that their sum is zero. Although this adjustment is applied globally for the whole domain, the test computations have shown that the conservation is achieved also on each grid as the converged solution is approached.

For a proper handling of body motion a strategy based on deactivation of grid cells which are temporarily outside the computational domain, i.e. covered by a body, and their reactivation when they re-enter the computational domain has been implemented. In this way the body motion of arbitrary scales can be handled successfully.

The present method was extensively assessed by applying it to some flows for which either numerical or experimental data exist or solutions could be obtained using another numerical

technique available in the commercial software. The emphases was placed on exploring the effects of the inter-grid coupling algorithm on the overall accuracy and efficiency of the method and to investigate the behavior of the conservation errors on overlapping grids. The inter-grid coupling strategy developed in the present study was proven to provide unique solutions over the whole computational domain. In the test computations performed here it was demonstrated that the overall accuracy on the overlapping grids is in the same range as on single grids. The linear interpolation used to compute the values at interpolation cells was found to be consistent with the second-order discretization scheme. Using a systematic grid refinement it was shown that the method features a second-order accuracy in space. The conservation errors were found to be about an order of magnitude smaller than the discretization errors and they decrease monotonically with approximately second-order as the grid is refined.

The capability of of the present overlapping grid technique to handle the problems with moving bodies has been demonstrated on three examples with complex relative motion of multiple bodies. These examples also demonstrated a great flexibility of the overlapping grid technique which is characterized by the possibility to generate separate grids of high quality around each body and to move these grids independently of each other. The method was proven to produce continuous flow fields irrespective of the kind and scale of body motion and complexity of the overlapping region (complex multiple overlapping and moving parts of the grid coming temporarily outside the solution domain).

Future development of this work should concentrate on the extension of the method to three dimensions, which is conceptually straightforward but requires some programming and testing effort. Another possible improvement would be the introduction of dynamic local mesh refinement in the overlapping region and body wakes. Since the background mesh can in most cases be a simple Cartesian grid (or another type of a structured grid), introducing an automatic refinement and coarsening is also conceptually not difficult, but time-consuming. This would, however, greatly improve the efficiency of the method when applied to three-dimensional problems with arbitrary motion of bodies.

Parallelization of the method involving overlapping grids and dynamic cell-wise grid refinement is another challenge, since the neighborhood relations and grid partitioning would have to change during simulation. This is also a time-consuming task, but a necessary one if large three-dimensional problems are to be solved efficiently on parallel computers or computer clusters.

Finally, body proximity, contact and detaching also requires a special attention. If two bodies, each with its own overlapping grid move over the background grid and touch each other, the overlap region goes to zero at the point or line of contact. In practice, a finite but small distance will have to be maintained, and either a dynamic local refinement of the overlapping grid, or its compression towards the contact location, would have to be implemented.

With these extensions, the overlapping grid method would become an excellent platform for a general-purpose computer code applicable to a wide range of engineering problems involving complex geometries and moving boundaries.

APPENDIX A

Interpolation functions

This appendix presents the interpolation functions used in the present study to compute the values at interpolation cells. The value of a variable ϕ in an interpolation cells P_i is obtained as a linear combination of the values in a number of donor cells D_k, i.e.

$$\phi_{P_i} = \sum \alpha_k \phi_{D_k} \,, \tag{A.1}$$

where α_k are the interpolation weighting factors and the number of the donor cells depends on the form of interpolation functions used.

The technique used here is illustrated in figure A.1. For given three points D_k surrounding the point P_i the weighting factors α_k can be obtained by solving the following linear equation system

$$\begin{aligned} \alpha_1 x_{D_1} + \alpha_2 x_{D_2} + \alpha_3 x_{D_3} &= x_{P_i} \\ \alpha_1 x_{D_1} + \alpha_2 x_{D_2} + \alpha_3 x_{D_3} &= x_{P_i} \\ \alpha_1 + \alpha_2 + \alpha_3 &= 1 \,. \end{aligned} \tag{A.2}$$

The values of weighting factors are then obtained in explicit form as:

$$\begin{aligned} \alpha_1 &= \frac{(x_{D_2}\, y_{D_3} - x_{D_3}\, y_{D_2}) + (y_{D_2} - y_{D_3})\; x_{P_i} + (x_{D_3} - x_{D_2})\; y_{P_i}}{(x_{D_2}\, y_{D_3} + x_{D_1}\, y_{D_2} + x_{D_3}\, y_{D_1}) - (x_{D_2}\, y_{D_1} + x_{D_3}\, y_{D_2} + x_{D_1}\, y_{D_3})} \\ \alpha_2 &= \frac{(x_{D_3}\, y_{D_1} - x_{D_1}\, y_{D_3}) + (y_{D_3} - y_{D_1})\; x_{P_i} + (x_{D_1} - x_{D_3})\; y_{P_i}}{(x_{D_2}\, y_{D_3} + x_{D_1}\, y_{D_2} + x_{D_3}\, y_{D_1}) - (x_{D_2}\, y_{D_1} + x_{D_3}\, y_{D_2} + x_{D_1}\, y_{D_3})} \\ \alpha_3 &= \frac{(x_{D_1}\, y_{D_2} - x_{D_2}\, y_{D_1}) + (y_{D_1} - y_{D_2})\; x_{P_i} + (x_{D_2} - x_{D_1})\; y_{P_i}}{(x_{D_2}\, y_{D_3} + x_{D_1}\, y_{D_2} + x_{D_3}\, y_{D_1}) - (x_{D_2}\, y_{D_1} + x_{D_3}\, y_{D_2} + x_{D_1}\, y_{D_3})} \end{aligned} \tag{A.3}$$

If the point P_i is enclosed in the triangle $D_1 D_2 D_3$, the weighting factors α_k are all positive and their some is equal to one.

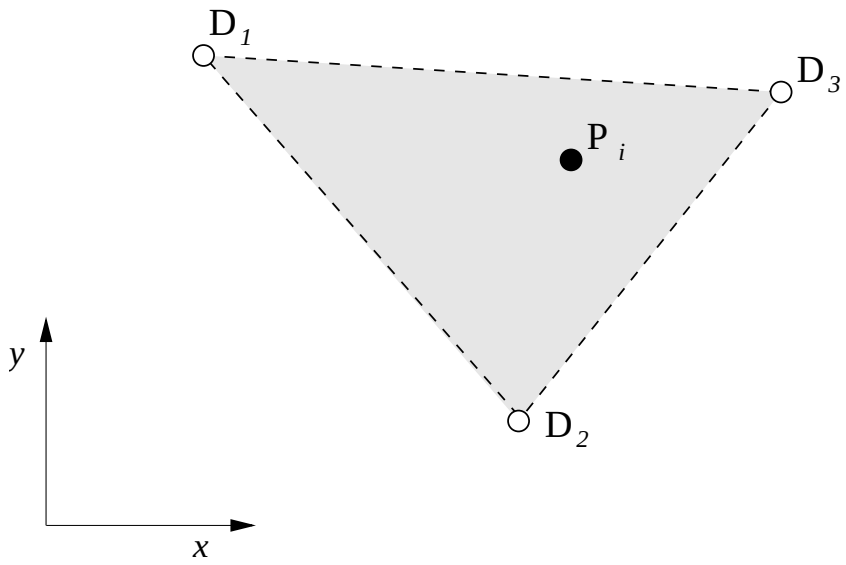


Figure A.1: Construction of a linear interpolation function based on three donor cells.

APPENDIX B

SST turbulence model

The k - ω SST turbulence closure model solves transport equations for k and ω [89] which may be written in the form of the generic transport equation:

$$\frac{\partial}{\partial t}\int\limits_V \rho\phi \,\mathrm{d}V + \oint\limits_S \rho\phi\,\mathbf{v}\cdot\mathrm{d}\mathbf{s} = \oint\limits_S \Gamma_\phi \mathrm{grad}\,\phi\cdot\mathrm{d}\mathbf{s} + \oint\limits_S Q_{\phi S}\cdot\mathrm{d}\mathbf{s} + \int\limits_V Q_{\phi V}\,\mathrm{d}V\,. \tag{B.1}$$

The meaning of each term in equation (B.1) is given in table below:

ϕ	k	ω
Γ_ϕ	$\mu + \dfrac{\mu_t}{\sigma_k}$	$\mu + \dfrac{\mu_t}{\sigma_\omega}$
$Q_{\phi S}$	0	0
$Q_{\phi V}$	$\rho\tilde{P}_k - \rho\beta^*\omega k$	$\rho\dfrac{\gamma}{\nu_t}P_k - \rho\beta\omega^2 + \rho(1-F_1)2\sigma_{\omega_2}\dfrac{1}{\omega}\dfrac{\partial k}{\partial x_j}\dfrac{\partial k}{\partial x_j}$

Production of turbulent kinetic energy P_k and its limitation for k equation

$$P_k = \tau_{ij}\frac{\partial U_i}{\partial x_j};\quad \tilde{P}_k = \min\left(P_k, c_l\varepsilon\right) \tag{B.2}$$

Model coefficients

$$\varphi = F_1\varphi_1 + (1-F_1)\varphi_2$$

φ	σ_k	σ_ω	κ	γ	β	β^*	c_l
φ_1	2.000	2.000	0.41	0.5532	0.0750	0.09	10
φ_2	1.000	1.168	0.41	0.4403	0.0828	0.09	10

$$F_1 = \tanh\left(arg_1^4\right);\quad arg_1 = \min\left(arg_2, \frac{4\rho\sigma_{\omega_2}k}{CD_{k\omega}n^2}\right);\quad arg_2 = \max\left(\frac{\sqrt{k}}{\beta^*\omega n}, \frac{500\nu}{\omega n^2}\right)$$

$$CD_{k\omega} = \max\left(2\rho\sigma_{\omega_2}\frac{1}{\omega}\frac{\partial k}{\partial x_k}\frac{\partial\omega}{\partial x_k}, 1.0\cdot 10^{-10}\right)$$

$$F_2 = \tanh\left(arg_3^2\right); \quad arg_3 = \max\left(2\frac{\sqrt{k}}{\beta^* \omega n}, \frac{500\nu}{\omega n^2}\right)$$

Turbulent viscosity

$$\mu_t = \rho \frac{a_1 k}{\max\left(a_1 \omega, \sqrt{2}\, S\, F_2\right)} \tag{B.3}$$

where $S = \sqrt{S_{ij} S_{ij}}$ and $a_1 = 0.31$.

Boundary conditions at wall

At first cell near the wall $\omega_1 = \frac{6\nu}{0.075 y_1^2}$, where y_1 is the distance to the wall, and at wall $k_w = 0$.

Bibliography

[1] ATTA, E. H. Component adaptive grid interfacing. *AIAA Paper 81–0382* (1981).

[2] ATTA, E. H., AND VADYAK, J. A grid overlapping scheme for flowfield computations about multi-component configurations. *AIAA Journal 21* (1983), 1271–1277.

[3] BARDINA, J. E., HUANG, P. G., AND COAKLEY, T. Turbulence modeling validation. *AIAA Paper 97–2121* (1997).

[4] BARRETT, R., BERRY, M., CHAN, T. F., DEMMEL, J., DONATO, J., DONGARA, J., EJKHOUT, V., POZO, R., ROMINE, C., AND DER VORST, H. V. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, Philadelphia, PA, 1994.

[5] BATINA, J. T. Unsteady Euler algorithm with unstructured dynamic mesh for complex aircraft aerodynamic analysis. *AIAA Journal 29* (1991), 327–333.

[6] BAYSAL, O., FOULADI, K., AND LESSARD, V. R. Multigrid and upwind viscous flow solver on three dimensional overlapped and embedded grids. *AIAA Journal 29* (1991), 903–910.

[7] BENEK, J. A., STEGER, J. L., AND DOUGHERTY, F. C. A flexible grid embedding technique with application to the Euler equations. *AIAA Paper 83–1944* (1983).

[8] BENEK, J. A., AND STEGER, P. G. B. J. L. A 3-D Chimera grid embedding technique. *AIAA Paper 85–1523* (1985).

[9] BERGER, M. On conservation at grid interfaces. *SIAM Journal on Numerical Analaysis 24* (1987), 967–984.

[10] BERGER, M. J., AND OLIGER, J. Adaptive mesh refinement for hypberbolic partial differential equations. *Journal of Computational Physics 53* (1984), 561–568.

[11] BLOSCH, E., SHYY, W., AND SMITH, R. W. The role of mass conservation in pressure-based algorithms. *Numer. Heat Transfer, Part B 24* (1993), 415–430.

[12] BROWN, D. L. An unsplit Godunov method for systems of conservation laws on curvilinear overlapping grids, Los Alamos Unclassified Report LA-UR-93-2868, 1993.

[13] BUNING, P. G., WONG, T.-C., DILLEY, A. D., AND PAO, J. L. Prediction of hyper-x stage separation aerodynamics using CFD. *AIAA Paper 2000–4009* (2000).

[14] BURTON, T. M., AND EATON, J. K. Analysis of a fractional-step method on overset grids. *Journal of Computational Physics 177* (2002), 336–364.

[15] CHESSHIRE, G., AND HESHAW, W. D. Composite overlapping meshes for the solution of partial differential equations. *Journal of Computational Physics 90* (1990), 1–64.

[16] CHESSHIRE, G., AND HESHAW, W. D. A scheme for conservative interpolation on overlapping grids. *SIAM Journal on Scientific Computation 15* (1994), 819–845.

[17] COLES, D., AND WADCOCK, A. J. Flying–hot–wire study of flow past a NACA 4412 airfoil at maximum lift. *AIAA Journal 17* (1979), 321–329.

[18] DEMIRDŽIĆ, I. *A Finite Volume Method for Computation of Fluid Flow in Complex Geometries*. PhD thesis, University of London, Imperial College, 1982.

[19] DEMIRDŽIĆ, I., AND MUZAFERIJA, S. Numerical method for coupled fluid flow, heat transfer and stress analysis using unstructured moving meshes with cells of arbitrary topology. *Comput. Methods Appl. Mech. Engrg. 125* (1995), 235–255.

[20] DEMIRDŽIĆ, I., AND PERIĆ, M. Space conservation law in finite volume calculations of fluid flow. *Int. J. Numer. Methods Fluids 8* (1988), 1037–1050.

[21] DEMIRDŽIĆ, I., AND PERIĆ, M. Finite volume method for prediction of fluid flow in arbitrarily shaped domains with moving boundaries. *Int. J. Numer. Methods Fluids 10* (1990), 771–790.

[22] DOUGHERTY, F. C. *Development of Chimera Grid Scheme with Applications to Unsteady Problems*. PhD thesis, Stanford University, 1985.

[23] DOUGHERTY, F. C., AND KUAN, J. Transonic store separation using a three-dimensional Chimera grid scheme. *AIAA Paper 89–0637* (1989).

[24] DRIKAKIS, D., MAJEWSKI, J., ROKICKI, J., AND ZOLTAK, J. Investigation of blending-function-based overlapping-grid technique for compressible flows. *Comput. Methods Appl. Mech. Engrg. 190* (2001), 5173–5195.

[25] DURST, F., MAXWORTHY, T., AND PEREIRA, J. C. F. Piston-driven unsteady separation at a sudden expansion in a tube: flow visualization and LDA measurements. *Comput. Methods Appl. Mech. Engrg. 190* (2001), 5173–5195.

[26] FERZIGER, J. H., AND PERIĆ, M. *Computational Methods for Fluid Dynamics*. Springer Verlag Berlin Heidelberg, 1996.

[27] GHIA, U., GHIA, K. N., AND SHIN, C. T. High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method. *Journal of Computational Physics 48* (1982), 387–411.

[28] HADŽIĆ, I. *Second-Moment Closure Modelling of Transitional and Unsteady Turbulent Flows*. PhD thesis, Delft University of Technology, 1999.

[29] HADŽIĆ, I. Private comunication. Unpublished, 2001.

[30] HADŽIĆ, I., XING, Y., MUZAFERIJA, S., AND PERIĆ, M. Numerical simulation of interaction of a floating body and a free-surface flow with waves. In *WCCM V: Fifth World Congress on Computational Mechanics* (July 2002), J. E. H. A. Mang, F. G. Rammerstorfer, Ed.

[31] HENSHAW, W. D. *Part II: Composite Overlapping Grid Techniques*. PhD thesis, Department of Applied Mathematics, California Institute of Technology, 1985.

[32] HENSHAW, W. D. A fourth-order accurate method for the incompressible Navier - Stokes equations on overlapping grids. *Journal of Computational Physics 113* (1994), 13–25.

[33] HENSHAW, W. D., AND CHESSHIRE, G. Multigrid on composite meshes. *SIAMSSC 8* (1987), 914–923.

[34] HINATSU, M., AND FERZIGER, J. H. Numerical computation of unsteady incompressible flow in complex geometry using a composite multigrid technique. *Int. J. Numer. Methods Fluids 13* (1991), 971–997.

[35] HUBBARD, B. J., AND CHEN, H. C. Calculation of unsteady flows around bodies with relative motion. In *Enbineering mechanichs: proceedings of the 10th conference* (New York, NY, 1995), A. S. of Civil Engineers, Ed., vol. 2.

[36] HUURDEMAN, B. *Numerische Simulation inkompressibler turbulenter Strömungen mit Mehrgitterverfahren auf unstrukturierten Gittern*. Dissertation, Universität Stuttgart, 1999.

[37] ICCM INSTITUTE OF COMPUTATIONAL CONTINUUM MECHANICS GMBH. *User Manual, Comet Version 2.00*, 2001.

[38] KAO, K.-H., LIOU, M.-S., AND CHOW, C.-Y. Grid adaptation using chimera composite overlapping meshes. *AIAA Journal 32* (1994), 942–949.

[39] KOOBUS, B., AND FARHAT, C. Second-order time-accurate and geometrically conservative implicit schemes for flow computations on unstructured dynamic meshes. *Comput. Methods Appl. Mech. Engrg. 170* (1999), 103–129.

[40] LESOINNE, M., AND FARHAT, C. Geometric conservation laws for flow problems with moving boundaries and deformable meshes and their impact on aeroelastic computations. *Comput. Methods Appl. Mech. Engrg. 134* (1996), 71–90.

[41] LILEK, Z., MUZAFERIJA, S., PERIĆ, M., AND SEIDL, V. Computation of unsteady flows using nonmatching blocks of structured grid. *Numer. Heat Transfer, Part B 32* (1997), 403–418.

[42] LÖHNER, R. Robust, vectorized search algorithms for interpolation on unstructured grids. *Journal of Computational Physics 118* (1995), 380–387.

[43] LÖHNER, R. *Applied Computational Fluid Dynamics Techniques; An Introduction Based on Finite Element Methods*. John Wiley & Sons, Ltd, 2001.

[44] LÖHNER, R., SHAROV, D., LUO, H., AND RAMAMURTI, R. Overlapping unstructured grids. *AIAA Paper 01–0439* (2001).

[45] LUO, H., BAUM, J. D., AND LÖHNER, R. A fast matrix-free implicit method for compressible flows on unstructured grids. *Journal of Computational Physics 146* (1998), 664–690.

[46] LUO, H., BAUM, J. D., AND LÖHNER, R. An accurate, fast, matrix-free implicit method for computing unsteady flows on unstructured grids. *Computers and Fluids 30* (2001), 137–159.

[47] MADRANE, A., HEINRICH, R., AND GERHOLD, T. Implementation of the chimera method in the unstructured hybrid DLR finite volume Tau-Code. In *Proceeding of 6th Overset Composite Grid and Solution Technology Symposium* (2002). Fort Walton Beach, Florida.

[48] MEAKIN, R. L. On the spatial and temporal accuracy of overset grid methods for moving body problems. *AIAA Paper 94–1925* (1994).

[49] MEAKIN, R. L. Object X-rays cutting holes in composite overset structured grids. *AIAA Paper 2001–2537* (2001).

[50] MEISTER, A. *Numerik linearer Gleichungssysteme*. Friedrich Vieweg & Sohn Verlaggesellschaft Braunschweig/Wiesbaden, 1999.

[51] MENTER, F. R. Two–equation eddy–viscosity turbulence models for engineering applications. *AIAA Journal 32* (1994), 269–289.

[52] MILGRAM, M. S. Does a point lie inside a poligon? *Journal of Computational Physics 84* (1989), 134–144.

[53] MOCTAR, O. A. M. E. *Numerische Berechnung von Strömungskräften beim Manövrieren von Schiffen*. Dissertation, Technische Universität Hamburg-Harburg, 2001.

[54] MUZAFERIJA, S. *Adaptive Finite Volume Method for Flow Predictions Using Unstructured Meshes and Multigrid Approach*. PhD thesis, University of London, 1994.

[55] NAKAHASHI, K., TOGASHI, F., AND SHAROV, D. An intergrid-boundary definition method for overset unstructured grid approach. *AIAA Journal 38*, 11 (2000), 2077–2084.

[56] NIČENO, B. *An Unstructured Parallel Algorithm for Large and Conjugate Heat Transfer Simulations*. PhD thesis, Delft University of Technology, 2001.

[57] PÄRT-ENANDER, E. *Overlapping Grids and Applications in Gas Dynamics*. PhD thesis, Uppsala University, Department of Scientific Computing, 1995.

[58] PATANKAR, S. V. *Numerical Heat Transfer and Fluid Flow*. McGraw-Hill, New York, 1980.

[59] PERIĆ, M. *A Finite Volume Method for the Prediction of Three-Dimensional Fluid Flow in Complex Ducts*. PhD thesis, University of London, Imperial College, 1985.

[60] PETERSSON, N. A. An algorithm for assembling overlapping grid systems. *SIAM Journal on Scientific Computation 20* (1999), 1995–2022.

[61] PETERSSON, N. A. Stability of pressure boundary conditions for Stokes and Navier-Stokes equations. *Journal of Computational Physics 172* (2001), 40–70.

[62] RHIE, C. M., AND CHOW, W. L. Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA Journal 21*, 11 (1983), 1525–1532.

[63] ROGERS, S. E., MENTER, F., AND DURBIN, P. A. A comparison of turbulence models in computing multi-element airfoil flows. *AIAA Paper 94–0291* (1994).

[64] ROKICKI, J., DRIKAKIS, D., MAJEWSKI, J., AND ZÓLTAK, J. Overlapping mesh technique for compressible flows - parallel implementation. In *Parallel Numerical Computations with Applications* (1999), T. Yang, Ed., Department of Computer Science, Linköping University, Sweden, pp. 159–176.

[65] SCHAHCHERAGHI, N., AND DWYER, H. A. Moving and rotating sphere in the thermal entrance region of a heated pipe. *ASME Journal of Heat Transfer. 122* (2000), 336–344.

[66] SEIBEL, A. *CFD–Berechnung eines Voith–Schneider–Propellers*. Diplomarbeit, Technische Universität Hamburg-Harburg, 2000.

[67] SHÄFER, M., AND TUREK, S. Benchmark computations of laminar flow around cylinder. In *Flow simulation with high-performance computers: DFG priority research programme results 1993–1995* (1996), E. H. Hirschel, Ed., vol. 52, Vieweg, Braunschweig, pp. 547–566.

[68] SHAROV, D., LUO, H., AND BAUM, J. D. Implementation of unstructured grid GMRS+LU-SGS method on shared-memory, cache-based parallel computers. *AIAA Paper 36*, 3 (1998), 484–486.

[69] SHYY, W. *Computational Fluid Dynamics with Moving Boundaries*. Taylor & Francis, 1996.

[70] SLATTERY, J. C. *Momentum, Energy and Mass Transfer in Continua*. Robert E. Krieger Publishing Company, 1981.

[71] STANGL, R., AND WAGNER, S. Euler simulation of a helicopter configuration in forward flight using a chimera technique. In *Proceedings of the 52th American Helicopter Society Annual Forum and Technology Display* (1996). Washington, D.C.

[72] STARIUS, G. Composite mesh difference methods for elliptic and boundary value problems. *Numer. Math. 28* (1977), 243–258.

[73] STARIUS, G. On composite mesh difference methods for hyperbolic differential equations. *Numer. Math. 35* (1980), 241–255.

[74] STEGER, J. L., AND BENEK, J. A. On the use of composite grid schemes in computational aerodynamics. *Comput. Methods Appl. Mech. Engrg. 64* (1987), 301–320.

[75] STEGER, J. L., DOUGHERTY, F. C., AND BENEK, J. A. A Chimera grid scheme. In *Advances in grid generation* (1983), K. N. Ghia and U. Ghia, Eds., ASME FED, pp. 59–69.

[76] STRÖLL, H. *Entwicklung eines effizienten Verfahrens zur Berechnung instationärer laminarer Kolben-Zylinder-Strömungen*. Dissertation, Universität Erlangen-Nürnberg, 1989.

[77] STRÖLL, H., DURST, F., PERIĆ, M., PEREIRA, J. C. F., AND SCHEUERER, G. Study of laminar, unsteady piston-cylinder flows. *Int. J. Numer. Methods Fluids 21* (1995), 237–251.

[78] STRÖLL, H., DURST, F., PERIĆ, M., AND SCHEUERER, G. Numerical study of a piston-driven unsteady flow in a pipe with sudden expansion. *Int. J. Numer. Methods Fluids 21* (1995), 237–251.

[79] SUHS, N. E., ROGERS, S. E., AND DIETZ, W. E. PEGASUS: An automated pre-processor for overset-grid CFD. *AIAA Paper 2002–3186* (2002).

[80] THOMAS, P. D., AND LOMBARD, C. K. Geometric conservation law and its application to flow computations on moving grids. *AIAA Journal 17* (1979), 1030–1037.

[81] TOGASHI, F., NAKAHASHI, K., ITO, Y., IWAMIYA, T., AND SHIMBO, Y. Flow simulation of NAL experimental supersonic airplane/booster separation using overset unstructured grids. *Computers and Fluids 30* (2001), 673–688.

[82] TU, J. *Three-Dimensional Overlapping Grids and Multigrid Methods for Flow Calculations in Complex IC Engine Geometries*. PhD thesis, Department of Mechanics/Applied Computational Fluid Dynamics, Royal Institute of Technology, Stockholm, 1992.

[83] TU, J. Y., AND FUCHS, L. Overlapping grids and multigrid methods for three-dimensional unsteady flow calculations in IC engines. *Int. J. Numer. Methods Fluids 15* (1992), 693.

[84] TU, J. Y., AND FUCHS, L. Three–dimensional unsteady flow calculations in both intake parts and combustion chamber. *AIAA Paper 92–0833* (1992).

[85] TU, J. Y., AND FUCHS, L. Three-dimensional overlapping grids for internal combustion engine geometries. In *Computational Fluid Dynamics '92* (North-Holland, 1992), C. Hirsch, Ed., Elsevier Science Publishers B. V.

[86] TUNCER, I. H. A 2-D unsteady navier-stokes solution method with overlapping/overset moving grids. *AIAA Paper 96–0822* (1996).

[87] VAN DER VORST, H. Bi-CGSTAB: A fast and smooth converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific Computation 13* (1992), 631–644.

[88] VASSBERG, J. C., BUNING, P. G., AND RUMSEY, C. L. Drag prediction for the DLR-F4 wing/body using OVERFLOW and CFL3D on an overset mesh. *AIAA Paper 2002–0840* (2002).

[89] VIESER, W., ESCH, T., AND MENTER, F. Heat transfer predictions using advanced two–equations turbulence models. Technical memorandum CFX–VAL 10/0602, CFX, 2002.

[90] WANG, Z. J. A fully conservative interface algorithm for overlapped grids. *Journal of Computational Physics 122* (1995), 96–106.

[91] WEISS, J. M., AND SMITH, W. A. Preconditioning applied to variable and constant density time–accurate flows on unstructured meshes. *AIAA Paper 94–2209* (1994).

[92] WHITE, F. M. *Fluid Mechanics*. McGraw-Hill, 1999.

[93] WRIGHT, J. A., AND SHYY, W. A pressure-based composite grid method for the Navier-Stokes equations. *Journal of Computational Physics 107* (1993), 225–238.

[94] ZHANG, L. P., AND WANG, Z. J. A block LU-SGS implicit dual time-stepping algorithm for hybrid dynamic meshes. *AIAA Paper 2003-3886* (2003).

About the author

27 January 1976	Born in Dubrave Donje, district Živinice, Bosnia-Herzegovina (BH)
1990 - 1994	Technical High School of Mechanical Engineering in Tuzla, BH
1994 - 1999	Studied mechanical engineering at the Faculty of Electrical and Mechanical Engineering of the University of Tuzla, BH
2000 - 2004	Research assistant in the Fluid Dynamics and Ship Theory Department of the Technical University Hamburg-Harburg
2004 -	Project engineer at ”Schuck Ingenieurbüro für Strömungsmechanik”, Munich