Multigrid Methods applied to Fluid-Structure Interaction

Vom Fachbereich Maschinenbau
an der Technischen Universität Darmstadt
zur Erlangung des Grades eines

Doktor der Naturwissenschaften
(Dr. rer. nat.)

genehmigte

D i s s e r t a t i o n

vorgelegt von

Dipl.-Math. Stephen Sachs

aus Raleigh, NC (USA)

Berichterstatter:	Prof. Dr. rer. nat. Michael Schäfer
Mitberichterstatter:	Prof. Dr. rer. nat. Stefan Ulbrich
Tag der Einreichung:	05.12.2011
Tag der mündlichen Prüfung:	08.02.2012

Darmstadt 2012
D17

Erklärung

Hiermit versichere ich die vorliegende Arbeit selbstständig verfasst und nur die angegebenen Hilfsmittel verwendet zu haben.

Darmstadt, den 1. Dezember 2011

Stephen Sachs

Danksagung

Ich möchte diese Stelle nutzen um mich bei allen Personen zu bedanken, die am gelingen meiner Arbeit beteiligt waren. Die vorliegende Arbeit wäre ohne die zahlreiche Unterstützung nicht in dieser Form zustande gekommen. Auch wenn ich nicht alle die mich in den letzten Jahren unterstützt haben an dieser Stelle nennen kann, möchte ich hier stellvertretend den wichtigsten Personen für meine Arbeit danken.
Ich möchte Herr Schäfer und Herr Ulbrich für die Betreuung meiner Arbeit danken. Auch dass sie mir die Möglichkeit geboten haben jederzeit mit Fragen auf sie zuzukommen weiß ich sehr zu schätzen. Mein Dank gilt vor allem Dörte Sternel für all ihre Unterstützung und aufbauenden Worte in den schwierigen Zeiten meiner Dissertation. In diesem Sinne möchte ich auch meiner Familie und meinen Freunden sowie meinen Mitbewohnern für ihre Unterstützung und ihre Geduld in den letzten Jahren danken. Meinen Bürokollegen Yu Du, Stefan Kneißl und Hannes Lück möchte ich für die Zeiten die wir innerhalb und ausserhalb des Büros zusammen verbracht haben danken. Ich hatte eine sehr schöne Zeit mit Euch und habe viel von Euch gelernt. Zu besonderem Dank bin ich meinen Korrekturlesern Astrid Walle, Yu Du und Ekkehard Sachs verpflichtet. Sie haben in Stunden mühevoller Arbeit meine Dissertation von Fehlern befreit. Ich weiß Euren Aufwand zu schätzen. Desweiteren möchte ich mich bei allen Kollegen am Fachgebiet sowie an der Graduiertenschule für die offene und freundliche Atmosphäre und das freundschaftliche Verhältnis über die Arbeit hinaus sowie die fruchtbare Zusammenarbeit bedanken. Für ihren Einsatz einen reibungslosen Ablauf meiner Arbeit zu gewährleisten sowie als Ansprechpartner in wissenschaftlichen und nichtwissenschaftlichen Belangen möchte ich mich bei Monika Müller, Heike Hoffmann, Carina Schuster, Friederike Gauß, Markus Lazanowski und Florian van de Loo bedanken. Danke, dass Ihr mir immer den Rücken frei gehalten habt. Desweiteren möchte ich Michael Fladerer und Christian Schmitt für ihren besonderen Einsatz danken und dafür dass sie sich auch spät abends und am Wochenende einsetzen eine einwandfreie Infrastruktur zu gewährleisten. Zu guter Letzt möchte ich mich noch einmal bei Allen nicht persönlich genannten bedanken. Ich weiss Eure Hilfe zu würdigen.

Diese Arbeit wurde von der Exzellenzinitiative des Bundes und der Länder und der Graduate School of Computational Engineering an der Technischen Universität Darmstadt unterstützt.

Contents

Chapter 1

Introduction

In the last decades the numerical simulation of physical phenomena has permanently advanced into the development process in engineering. It is used to give an a priori insight into the behaviour of components, thus assisting the prototyping process. Furthermore an optimization for certain goals can be conducted in advance of manufacturing. For various of today's engineering applications the knowledge of the behaviour of one independent physical field is insufficient. This is especially valid in the examination of unsteady processes where a dynamic response of two or more components is present. In the case of fluids interacting with structures this includes components in wind energy plants as e.g. in Bazilevs et al. [3], aeronautics as e.g. Liu et al. [42], or turbo machinery as e.g. in Du [17]. Another recent field of application is medicine. The modelling of aneurysms as e.g. in Razzaq et al. [48] or Bazilevs et al. [4] is rendered possible through numerical simulation. For these kinds of applications the numerical field of Fluid-Structure Interaction (FSI) has been established. Individual solvers for fluid and structure problems, or monolithic approaches containing both, have been developed that are capable of simulating the respective domain with unsteady boundary conditions as a result of their counterpart. The numerical simulation of FSI is an ongoing challenge and several research groups are working on this topic.

1.1 Motivation

One of the main challenges in the field of FSI arises from the vast amount of computational resources needed to carry out a coupled simulation. Conducting a physically relevant simulation of a structure or fluid domain alone is already a demanding task given the computing capacity available today. In FSI the required resources grow even higher, thus more effective solution methods are necessary. Using sophisticated solvers is one way to approach this, which has been and still is used. The sparsity of the resulting system

matrices can be greatly exploited and fast nonlinear solvers additionally accelerate the computation. Another idea is to exploit the structure of the FSI problem itself. This can be achieved by different approaches. On the one hand the FSI problem can be regarded as two individual problems on two disjoint domains. These problems are coupled by their boundary conditions but solved by individual solvers. On the other hand the FSI problem can be stated as one continuous problem over the entire FSI domain and discretized by one grid covering the different physical fields. Thereby efficient pre-conditioners can be applied.

1.2 State of the Art

The latter approach described in the last section, which employs the continuous problem, is called the monolithic approach. The two physical fields for fluid and structure are virtually extended to the entire FSI domain and a step function is introduced to determine which field is active in which area. Due to the common discretization and globally applied solvers this approach is numerically very robust. But, as a drawback, because of the common discretization and analytical description low flexibility in the choice of solution algorithms and grids is given. The monolithic approach is the basis to the implementations in the works of e.g. Dunne [18], Hron and Turek [34], Walhorn et al. [73], Heil [30], and Matthies and Steindorf [43].

The former approach described in the last section is called the partitioned approach. The physical fields are restricted to their respective domains and discretization as well as their solution is carried out individually. Only the boundary conditions at the FSI interface are exchanged at every time step. This approach remains highly adaptable as the fluid and the structure part can be solved by established solvers and the individual grids enable discretizations fitted to the physical field. Due to its iterative nature this approach is numerically challenging. Under-relaxation of the values that cross the FSI boundary is necessary to ensure convergence. As the partitioned approach suffers from stability issues and severe time step restrictions, a more sophisticated approach, named the implicit partitioned approach, has been developed. Built from the partitioned approach it inherits its flexibility but gains more stability by several implicit coupling steps within a time step. The implicit partitioned approach is the basis to the implementations in the works of e.g. Sternel et al. [59], Schäfer et al. [55], van Brummelen et al. [70], Sieber [58], Vierendeels et al. [71], and Gerstenberger and Wall [26]. As both approaches bring along their individual advantages and disadvantages (see e.g. Degroote et al. [14] or Michler et al. [44]) the goal of this work is specified by combining the advantages, while avoiding the drawbacks.

1.3 Goal of this Work

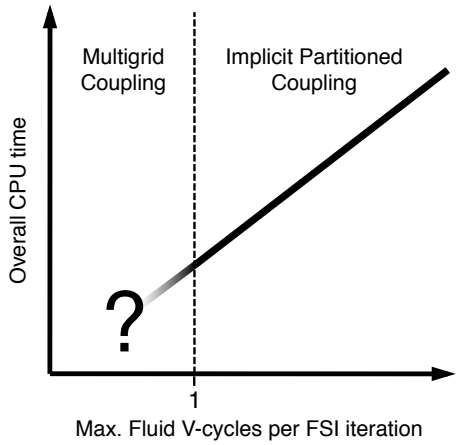


Figure 1.1: Effect of max. number of fluid V-cycles in each FSI iteration on the overall CPU time of the coupled simulation

The main subject of this work is to create an approach, which is located in between the monolithic and the implicit partitioned one. One method to improve the convergence speed of iterative solvers is the use of a multigrid method. Therefore the goal of this thesis is the implementation of a global multigrid method in an FSI approach.
As low-frequency errors are reduced substantially faster on coarser grids than on fine ones and solver iterations on coarser grids are per se faster, multigrid methods have a noticeable influence on computing time. The multigrid methods as introduced by Brandt in the early 1970's [7] are up to the present a popular choice in the solution of large sparse system. As they imply linear run-time, they remain even by today's standards a competitive solver.
Using the implicit partitioned approach as the basis for the implementation, the multigrid coupling algorithm is introduced, which is an enhancement towards a monolithic approach while preserving the flexibility of two independent codes. The geometric multigrid method is applied to the coupled problem, and the solution procedure from the implicit partitioned approach is used as a smoother, i.e. is executed on every grid level. By this close coupling on multiple grids not only the error of the individual fields is reduced by the multigrid method, but also the error that arises from the coupling itself is directly affected in the multigrid computation. In other words, a great part of it is eliminated during coarse grid iterations, thus reducing computation time.
This multigrid coupling approach introduces a possibility to exploit the advantages of multigrid computation to a FSI coupling framework. Further-

more, due to the comparatively high number of exchanges within a time-step, it combines the robustness of the monolithic approach with the flexibility of the partitioned approach. Figure 1.1 shows a motivation for the use of multigrid coupling. The average CPU time for one time step plotted over the number of V-cycles in the fluid solver until coupling. The diagram represents the computations of an implicit partitioned coupling approach with a multigrid fluid solver. As the maximum number of V-cycles per coupling iteration decreases, the CPU time also decreases, reaching its minimum value at 1 V-cycle. The next member in the sequence of decreasing number of V-cycles per FSI iteration is the multigrid coupling approach with multiple FSI iterations per V-cycle.

As a second strategy for the acceleration of partitioned coupling approaches the extrapolation of force is presented. Extrapolation techniques have shown their influence on computing time in all kinds of numerical applications. In this work the extrapolation is applied in between time steps in order to create a more promising initial state for the following coupled simulation step. The use of temporal interpolation to predict an initial condition for the computations in a new time step in FSI include the works of Breuer and Münsch [9, 10], Schäfer et al. [56], or Piperno [46]. All these approaches display their distinct properties in the computation of the unknown, but have the variable they are applied to in common. All of the above works use the displacement as variable for the extrapolation. In this work the extrapolation is applied to the velocity and pressure variables in the fluid domain or the resulting force at the FSI interface, respectively. This is compared with applying the same extrapolation functions to the displacement. A numerical test case is performed to show the advantages that follow from the extrapolation of the fluid variables.

The chapters below are organized as follows. The second chapter gives the analytical description of the FSI coupling. The partial differential equations for the fluid and structure parts are defined on a moving domain while incorporating the coupling. In the third chapter the numerical methods used in this work are introduced. Special attention is given to the discretization on the moving fluid domain and the nonlinear multigrid method. In the fourth chapter the multigrid coupling approach for FSI problems is introduced. The theoretical considerations are described and the implementation is validated by a series of test cases. In the fifth chapter the displacement and force extrapolations are introduced to accelerate FSI computations. In the sixth chapter the resulting methods of this work are applied to a numerical benchmark in order to compare their performance. And finally, in the seventh chapter concluding remarks and an outlook for further work is given.

Chapter 2

Conservation Principles

In this chapter the coupled FSI system as used in this work is introduced. First, the Arbitrary Lagrangian-Eulerian (ALE) approach, which defines the framework used in this work, is introduced. Then, based on the framework the Navier Stokes equation for incompressible Newtonian flow on moving grids is defined. Next, the elasticity equation for structure computation with the St. Venant Kirchhoff material law is defined. Finally, the coupled problem containing the FSI boundary conditions as implemented in this work is stated.

2.1 Arbitrary Lagrangian Eulerian Formulation for Fluid-Structure Interaction

In this section the ALE formulation is introduced. First the Lagrangian, Eulerian and ALE frameworks are presented, then the conservation equations in the ALE framework are derived directly from the Reynolds Transport Theorem.
One of the first records on the ALE formulation was by Hirt, Amsden and Cook [33], which was embedded in the finite difference context. The theoretical framework for an ALE formulation for incompressible viscous flow in a weighted residual context was first introduced by Hughes et al. [36].

The *Lagrangian framework* or material framework is described through a coordinate system, that is attached to the underlying material. By its nature this excludes continuous description of fluid problems of being examined in a Lagrangian framework, as very large displacements of mass and swirls cannot be mapped.
The *Eulerian Framework* or spatial framework is defined by a spatially constant coordinate system. Fluid problems are usually solved in a Eulerian framework, because large and nonlinear displacements can be treated easily. For a structure problem this framework is not the common choice, as

additional convection terms arise from the material time derivative.
The *ALE Framework* is an approach to combine the advantages of both of the aforementioned frameworks and give a unified description of coupled problems. Especially in the case of FSI such a description cannot be avoided. In the ALE formulation a third arbitrarily defined coordinate system is introduced which can either be attached to the structure or to the fluid mesh or any other description.
In FSI environments two main paths have been followed. On the one hand there is the *Interface Capturing* method as in Dunne [18] or Hron [34]. In this approach the ALE coordinate system coincides with the Eulerian coordinate system. On the other hand there is the *Interface Tracking* method, see Donea [16] or Sternel [59]. In this approach the coordinate system follows the interface and for the structure domain coincides with the Lagrangian coordinate system.
In the following the variables with respect to the Lagrangian framework are denoted by X, Eulerian by x and ALE by χ.

2.2 Fluid

In the following, the ALE formulation of the Navier-Stokes equation is derived from the mass, momentum and rotational momentum conservation. The conservation equations are applied to a time dependent volume $V = V(t)$ with closed surface ∂V. The abbreviation $|_{(.)}$ indicates the framework in which the derivative is applied. For details refer to Appendix A.

2.2.1 Conservation of Mass

Mass conservation states that within the computational domain no mass is produced or lost. Consequentially, for density ρ and infinitesimal volume dV and Area dA this states

$$\left.\frac{d}{dt}\right|_X \int_V \rho \; dV = 0 \; . \tag{2.1}$$

Applying the Reynolds transport theorem to (2.1), we obtain the mass conservation in Eulerian description (2.2). The right hand side consists of the temporal variation of density integrated over the volume and the mass flux through the boundary:

$$\left.\frac{d}{dt}\right|_X \int_V \rho \; dV = \int_V \left.\frac{\partial \rho}{\partial t}\right|_x \; dV + \int_{\partial V} \rho v \; n \; dA \; . \tag{2.2}$$

The velocity in the second right hand side term is called the material velocity in Eulerian framework. Applying the same theorem to an arbitrary volume in the ALE framework yields

$$\left.\frac{d}{dt}\right|_\chi \int_V \rho \; dV = \int_V \left.\frac{\partial \rho}{\partial t}\right|_x \; dV + \int_{\partial V} \rho v^g \; n \; dA \tag{2.3}$$

in which v^g is the velocity of the ALE referential domain. Substituting the first term on the right hand side by the negative second term on the right hand side of (2.2) results in

$$\left.\frac{d}{dt}\right|_{\chi} \int_V \rho \; dV + \int_{\partial V} \rho(v - v^g) \; n \; dA = 0 \; . \tag{2.4}$$

Applying equation (A.9) results in the mass conservation in the ALE framework

$$\frac{\partial}{\partial t} \int_V \rho \; dV + \int_{\partial V} \rho(v - v^g) \; n \; dA = 0 \; . \tag{2.5}$$

This is a generalization of the Lagrangian and the Eulerian case. If χ is set to X, the ALE velocity is equal to the material velocity, thus the convective term vanishes. If χ is set to x, the ALE velocity vanishes, and the Eulerian formulation is produced.

2.2.2 Space Conservation Law

Equation (2.5) always has to be fulfilled for a non-moving fluid in every time step. Demirdzic [15] showed, that this is a nontrivial task. In order to check this property the so called Space Conservation Law (SCL) or Geometric Conservation Law as first stated by Thomas and Lombard [67] is introduced:

$$\frac{\partial}{\partial t} \int_V \rho \; dV = \int_{\partial V} \rho v^g \; n \; dA \tag{2.6}$$

which is (2.5) in the case of $v = 0$. If the SCL is fulfilled, (2.5) reduces to the same form as in the Eulerian case:

$$\int_{\partial V} \rho v \; n \; dA = 0. \tag{2.7}$$

From this follows that for the ALE description of the mass conservation it is sufficient to satisfy the mass conservation in Eulerian description (2.7) if the SCL is fulfilled.

2.2.3 Conservation of Momentum

The conservation of momentum originates directly from Newton's second law of motion. The equality of mass times acceleration and forces in Eulerian description yields:

$$\left.\frac{d}{dt}\right|_{X} \int_V \rho v \; dV = \mathcal{F} \tag{2.8}$$

in which the net force $\mathcal{F}$ is split up into forces acting on the surface, or stresses, and body forces

$$\mathcal{F} = \int_{\partial V} \sigma n \; dA + \int_V \rho f \; dV \tag{2.9}$$

with σ the so called Cauchy stress tensor and f the vector of body forces. Applying the same transformation as in section 2.2.1 to the left hand side of 2.8 and substituting F as in equation (2.9) results in the ALE description of momentum conservation

$$\frac{\partial}{\partial t}\int_V \rho v \; dV + \int_{\partial V} \rho(v - v^g)vn \; dA = \int_{\partial V} \sigma n \; dA + \int_V \rho f \; dV \tag{2.10}$$

2.2.4 Cauchy Stress Tensor

The conservation of angular momentum states, that the time derivative of angular momentum is equal to the applied angular momentum arising from surface and body forces. In Eulerian description this reads

$$\left.\frac{d}{dt}\right|_X \int_V x \times \rho v \; dV = \int_V x \times \rho f \; dV + \int_{\partial V} x \times \sigma n \; dA \tag{2.11}$$

Using this and the Eulerian description of momentum conservation one can conlude, that the stress tensor has to be symmetric $\sigma = \sigma^T$. Furthermore, by the definition of viscosity (see Bird et al. [6]), it must be assumed that σ is linear dependent on the velocity gradient. Adding the absence of divergence, which follows from equation (2.7), the Cauchy stress tensor for incompressible Newtonian fluids results in

$$\sigma = \mu_f \left(\nabla v + \nabla v^T\right) - pI \tag{2.12}$$

in which μ_f represents the dynamic viscosity and I the unity matrix.

2.3 Structure

In the following the ALE formulation of the elasticity equation used in this work is derived. Recalling, that the interface tracking method is used in this work, the ALE coordinate system coincides with the Lagrangian on the structure domain.

Since the ALE velocity equals the material velocity, the convection term in equation (2.10) vanishes. Furthermore, Gauß's divergence theorem is applied to the remaining surface integral and the volume integral is omitted. Then, the momentum conservation in ALE formulation for the structure states

$$\rho\ddot{\chi} = \nabla \cdot \sigma + \rho f \tag{2.13}$$

with f being the outer forces on the structure and ρ the structure density. The second time derivative of χ is denoted by $\ddot{\chi}$. In order to comply with

common notation, the stress tensor is given in terms of the second Piola-Kirchhoff stress S

$$\sigma = FS^T \tag{2.14}$$

with $F = \frac{\partial \chi}{\partial X}$ being the deformation gradient. For the test cases in this work the St. Venant Kirchhoff material model for isotropic, hyper-elastic materials is used, which is a simple extension to the linear elasticity theory. It describes the stress as

$$S = \lambda_s \mathrm{tr}(E) I + 2\mu_s E \tag{2.15}$$

with λ_s and μ_s as the Lamé constants and

$$E = \frac{1}{2}\left(F^T F - I\right) \tag{2.16}$$

the Green-Lagrange strain for finite strains (nonlinear kinematics).

2.4 Coupled Problem in ALE Framework

In order to assemble the FSI system, the following notation needs to be introduced. A subscript f refers to the fluid and a subscript s to the structure. Let Ω be our computational FSI domain described in the ALE framework, which has a disjoint partition into the structure domain Ω_s and the fluid domain Ω_f with FSI boundary $\Gamma = \overline{\Omega}_s \cap \overline{\Omega}_f$. Let the velocity and pressure be well defined as well as the boundary conditions on the respective remaining boundaries $\partial\Omega_f \backslash \Gamma$ and $\partial\Omega_s \backslash \Gamma$. Then, for every control volume $V \subset \Omega_f$ the fluid part of the coupled problem can be stated as the Navier Stokes equation for incompressible Newtonian fluids in the ALE framework

$$\frac{\partial}{\partial t}\int_V \rho_f v \; dV + \int_{\partial V} \rho_f (v - v^g) v n \; dA = \int_{\partial V} \sigma n \; dA + \int_V \rho_f f_f \; dV \quad \text{in } \Omega_f \tag{2.17}$$

$$\int_{\partial V} \rho_f v \; n \; dA = 0 \quad \text{in } \Omega_f \tag{2.18}$$

with the SCL

$$\frac{\partial}{\partial t}\int_V \rho_f \; dV = \int_{\partial V} \rho_f v^g \; n \; dA \quad \text{in } \Omega_f \tag{2.19}$$

and ρ_f the fluid density, n the outer normal vector, f_f outer forces acting on the fluid,

$$\sigma = \mu_f(\nabla v + \nabla v^T) - pI \tag{2.20}$$

and μ_f the dynamic viscosity.
As for the structure part, the elasticity equation with the St. Venant Kirchhoff material can be denoted as

$$\nabla \cdot \left(FS^T\right) + \rho_s f_s = \rho_s \ddot{\chi} \quad \text{in } \Omega_s \tag{2.21}$$

with f_s the outer forces acting on the structure, ρ_s the structure density,

$$S = \lambda_s \ \mathrm{tr}(E) \ I + 2\mu_s E \quad \text{in } \Omega_s \tag{2.22}$$

λ_s, and μ_s the Lamé constants,

$$E = \frac{1}{2}\left(F^T F - I\right) \quad \text{in } \Omega_s \tag{2.23}$$

the Green-Lagrange strain, and

$$v = \dot{\chi} \ , \quad \text{in } \Omega_s \tag{2.24}$$

the structure velocity. The Dirichlet-Neumann coupling at the FSI boundary Γ is realized by

$$v|_{\Omega_f} = v|_{\Omega_s} \quad \text{and} \quad \frac{1}{\det(F)} F S F^T n|_{\Omega_s} = \sigma n|_{\Omega_f} \ . \tag{2.25}$$

Chapter 3

Numerical Foundations

This chapter gives a short overview over the numerical methods applied in this work. Furthermore, the notation for the subsequent chapters is introduced. For the sake of simplicity all numerical methods are shown for the 2D case, although the 3D case is implemented.
First the time discretization methods used in this work are introduced and applied to the respective semidiscrete system. Next, the space discretization for the structure and fluid part is introduced. The focus lies on the fluid discretization on moving grids with regard to the ALE framework. Then the grid movement techniques used in this work are introduced. Thereafter, the solvers used in this work are introduced, with focus on the linear and nonlinear multigrid method. Finally the extrapolation functions used in this work are introduced.

3.1 Time Discretization

In this section a short overview of the employed time discretization schemes is given. In order to preserve a coherent system, second order implicit schemes are used on the fluid and structure domain, and the quantities arising in the ALE description are discretized accordingly.

3.1.1 Newmark beta method

As the elasticity equation contains a second order time derivative, a suitable method for this has to be found. The Newmark beta method is a widely used time integration scheme in computational solid dynamics. It is first or second order accurate depending on its parameters. Let

$$\bar{K}(\chi) - \bar{M}\ddot{\chi} = \bar{f} \tag{3.1}$$

be the semi-discrete elasticity equation as resulting from section 3.2.1. The Newmark beta scheme can be derived directly from the Taylor expansion of

the displacement and velocity around time step $n+1$ ($\chi_{n+1} := \chi(x,(n+1)\Delta t)$), in which the linearized third time derivative $\dddot{\chi}_n \approx \frac{\ddot{\chi}_{n+1}-\ddot{\chi}_n}{\Delta t}$ is multiplied with the parameters 6β and 2γ for displacement and velocity respectively. The resulting scheme then states

$$\chi_{n+1} = \chi_n + \Delta t\dot{\chi}_n + \frac{\Delta t^2}{2}\ddot{\chi}_n + \beta\Delta t^2\left(\ddot{\chi}_{n+1} - \ddot{\chi}_n\right) \tag{3.2}$$

and

$$\dot{\chi}_{n+1} = \dot{\chi}_n + \Delta t\ddot{\chi}_n + \gamma\Delta t\left(\ddot{\chi}_{n+1} - \ddot{\chi}_n\right) \tag{3.3}$$

in which β and γ are the Newmark parameters and Δt is the time step. Rearranging equations (3.2) and (3.3) and adding it to equation (3.1) yields the first order system

$$\begin{pmatrix} 0 \\ 0 \\ \bar{K}(\chi_{n+1}) \end{pmatrix} + \begin{pmatrix} 1 & 0 & -\frac{1}{\beta\Delta t^2} \\ 0 & 1 & -\frac{\gamma}{\beta\Delta t} \\ -\bar{M} & 0 & 0 \end{pmatrix} \begin{pmatrix} \ddot{\chi}_{n+1} \\ \dot{\chi}_{n+1} \\ \chi_{n+1} \end{pmatrix} \tag{3.4}$$

$$= \begin{pmatrix} -\frac{1}{\beta\Delta t^2}\chi_n - \frac{1}{\beta\Delta t}\dot{\chi}_n + \left(1-\frac{1}{2\beta}\right)\ddot{\chi}_n \\ -\frac{\gamma}{\beta\Delta t}\chi_n + \left(1-\frac{\gamma}{\beta}\right)\dot{\chi}_n + \left(1-\frac{\gamma}{2\beta}\right)\Delta t\ddot{\chi}_n \\ \bar{f} \end{pmatrix} \tag{3.5}$$

which can be rewritten as:

$$K(\chi) + C\dot{\chi} + M\ddot{\chi} = f \tag{3.6}$$

According to Hilber, Hughes and Taylor [32], a connection of β and γ can be established. In order to best possible reduce the coefficients to the higher order terms in the local truncation error while preserving stability, the Newmark parameters have to satisfy

$$4\beta = (\gamma + 0.5)^2 \tag{3.7}$$

This is better known as the *Hilber-Hughes-Taylor-α-method* (HHT). Thus, β is the only remaining degree of freedom. For more information on this topic refer to the original paper or Wood [77].

3.1.2 Backward Differentiation Formula

The Backward Differentiation Formula (BDF) belongs to the family of linear multistep methods. It is suitable for first order ordinary differential equations and popular in solving stiff differential equations. It is derived from linear approximations of the gradient at unknown time Δtn. Let

$$\dot{v} = f(v,p) \tag{3.8}$$

be the semi-discrete momentum conservation from section 3.2.2. The second order approximation of the derivative, as used in this work by the fluid solver, is constructed by

$$\dot{v}_n \approx \frac{v_n - v_{n-1}}{\Delta t} + \frac{v_n - 2v_{n-1} + v_{n-2}}{2\Delta t} \tag{3.9}$$

Thus, the resulting time integration scheme, the second order BDF (BDF 2), is

$$v_n = 4/3v_{n-1} - 1/3v_{n-2} + 2/3\Delta t f(v_n, p_n) \tag{3.10}$$

This implicit integration scheme is second order accurate and A-stable. The BDF 2 time stepping scheme is used for the time integration in the fluid domain in this work.

3.2 Space Discretization

3.2.1 Finite Element Method

The Finite Element Method (FEM) is a widely used solution approach for partial differential equations (pdes). Its wide range of applicable orders of discretization, as well as the possibility to use highly customized elements has made it a widely used approach for structure problems.
Using the elasticity equation (2.21) with (2.22) the stress tensor and (2.23) the strain tensor the FEM is used to describe the semi-discrete system as implemented in the structure solver FEAP version 8.2 [65].
As there is no analytical solution to the elasticity equation, it is solved with respect to so called test functions. To further constrain the solution, it is set to consist of a linear combination of ansatz functions. With these simplifications, the equation can be rewritten as a discrete algebraic equation. For a detailed view of the discretization see the FEAP theory manual [64].

Weak Formulation Assuming certain differentiability properties of χ, the first step is to formulate the elasticity equation in a weak sense. Introducing test functions ϕ_j, $j \in \{1, \dots, N\}$ with $\phi_j|_{\Gamma_D} = 0$ on the Dirichlet boundary.
Equation (2.21) is multiplied by ϕ_j and integrated over the structure domain. Integration by parts is accomplished using Gauß's divergence theorem. As the test functions are set to be 0 at the Dirichlet boundary, the Dirichlet boundary integral resulting from partial integration vanishes. Inserting the von Neumann boundary condition into the equation results in

$$\int_{\Omega_s} FS^T \nabla\phi_j \, dV - \int_{\Gamma} \sigma\phi_j n \, dS + \int_{\Omega_s} \rho_s f_s \phi_j \, dV = \int_{\Omega_s} \rho_s \ddot{\chi}\phi_j \, dV \, , \forall j \in \{1, \dots, N\} \tag{3.11}$$

Discretization The Galerkin method states, that the desired function (in this case the displacement χ) is discretized into a weighted sum of so called ansatz functions

$$\chi(X,t) \approx \phi_0(X,t) + \sum_{i=1}^{N} \chi_i(t)\phi_i(X) \tag{3.12}$$

while the ansatz function ϕ_0 fulfills the Dirichlet boundary condition and the other ϕ_i are the same as the test functions in equation (3.11). In order that this discretization is a valid representation of the displacement, the test functions have to be a partition of unity. In this work first order polynomials with finite support are used for test functions. Furthermore, the structure domain is split into quadrilateral grid cells.

Integration The integrals are computed by Gaußian quadrature. It is exact for polynomials up to an order of $2n-1$, in which n is the number quadrature points used. The integration is carried out on the unit square, thus for all elements a unique mapping to the unit square has to be defined.

The Discrete Elasticity Equation Applying this mapping and integrating results in the nonlinear semi-discrete system, which can now formally be written as

$$\bar{K}(\chi) - \bar{f} = \bar{M}\ddot{\chi}. \tag{3.13}$$

$\bar{K}$ is called the stiffness, f the load vector, $\bar{M}$ the mass matrix, and χ the vector of displacements.
Note: For static and quasi-static problems, the acceleration term is omitted, and therefore the former equation reduces to:

$$\bar{K}(\chi) = \bar{f} \tag{3.14}$$

For further description of this method refer to textbooks on FEM, for example by Belytschko [5].

3.2.2 Finite Volume Method

Another member of the weighted residual family is the Finite Volume Method (FVM). As well as the FEM it is constructed by separating the computational domain into finite subspaces, referred to as volumes. The FVM is widely used in fluid mechanics as its resulting system matrix is comparably sparse and (in contrast to e.g. FEM) and it is natively conservative. The former attribute also contributes to a high parallel efficiency and the latter eliminates the need for special attention when conservation of mass is crucial. This is the case for the incompressible Navier Stokes equation.

Based on the Navier Stokes equation in the ALE framework (2.17), (2.18) with Cauchy stress tensor (2.20) and the space conservation law (2.19), the discretization as currently implemented in the fluid code FASTEST is described by the FVM. For a more detailed view see Appendix B.

Integration As a first step in the discretization, a quadrature formula is applied to the integrals in the momentum equation. The approximation of the cell centered and cell face centered terms is carried out according to the second order accurate midpoint rule.
The combined mass and grid flux $\dot{m}^g$ in the convective term is assumed to be known at the cell face centers. They arise from the pressure coupling in the solution of the discrete mass conservation equation.

Interpolation Cell face values such as the velocity in the convective term have to be interpolated as the values are computed and stored only on the cell centers. For the convective term different interpolation techniques are used (and mixed) as they involve different drawbacks and advantages.
The Upstream Differencing Scheme (UDS) is a straightforward approach for the interpolation and results in a numerically stable discretization, but due to its second order error term introduces numerical diffusion.
The Taylor Based Interpolation (TBI) scheme (MuLI in [39] or TSE in [40])is an extension of the second order accurate Central Differencing Scheme (CDS). The CDS suffers from order reduction on non-Cartesian grids. Thus, the TBI using more cell centers remains second order accurate on distorted grids.
Being an extension of the CDS, the TBI inherits the bad habit of possibly introducing oscillatory modes into the solution. Thus, Flux Blending is used. This includes the use of the Deferred Correction to accelerate the solution procedure.

Approximation of Gradients Gradients on the cell faces as arising in the diffusive term can be approximated easily on Cartesian grids. On distorted grids a cell face centered local coordinate system has to be introduced. The gradient is then calculated in the local coordinate system and transformed back into the global system. This is referred to as DABT in Lehnhäuser [39].
For cell centered gradients as arising in the pressure term another local, cell centered coordinate system has to be introduced. The pressure gradient is computed from the CDS approximation of pressure on the control volume faces and transformed to the global system.

The Discrete Momentum Conservation Equation Applying the time integration (3.10) to the discrete system concludes the discretization. All terms of the discrete momentum equation are dependent on the cell centered values of v and p as well as the geometry information. Thus, the system matrix can be assembled as:

- The implicit part of the convective mass and grid flux according to the deferred correction.
- The implicit part of the diffusive flux, which describes half of the main flow through the according cell face.

The right hand side is assembled by:

- The explicit part of the convective mass and grid flux, which is the TBI contribution without the CDS contribution under-relaxed by the flux blending parameter.
- The explicit part of the diffusive fluxes, including the remaining half of the main flow plus cross diffusion fluxes.
- The discretized pressure gradient multiplied with the corresponding cell volume.
- Other discrete source terms evaluated at the cell center multiplied with the corresponding cell volume that act as body forces.

The resulting system can now be written in a compact form as

$$a_P v_P + \sum_C a_C v_C = b_P \tag{3.15}$$

in which P indicates the cell center of the diagonal element, $C \in \{E, W, N, S\}$ is the index of neighbouring cell centers, and $a_C = a_C(v, p)$ as well as $b_C = b_C(v, p, f)$ are dependent on the velocity and pressure.

The Discrete Mass Conservation Equation The mass conservation equation is used to solve for the pressure as the momentum equation solves for the velocity. As the pressure term does not appear in the mass conservation equation an iterative pressure correction scheme has to be employed. Integrating equation (2.18) with the midpoint rule over a grid cell V_f in the fluid domain yields the sum of mass fluxes around every grid cell to be zero:

$$\int_{\partial V_f} \rho_f v \; n \; dA \approx \sum_c \dot{m}_c \tag{3.16}$$

The index $c \in \{e, w, n, s\}$ represents the cell face. The interpolation which leads to the mass flux is described in equation (3.24). If these mass fluxes

are to be computed using the velocity resulting from the last momentum equation solve v^k, with $m_c^k := m_c(v^k)$, their sum will not fulfill the mass conservation equation

$$\sum_c \dot{m}_c^k \neq 0 \ . \tag{3.17}$$

Thus an artificial mass source is added to the system. The iteration counter k is skipped in the discretization of the momentum equation, as all values are associated with the same iteration. The aim of the pressure correction scheme is to find a set of velocity and pressure vectors v^{k+1}, p^{k+1} which fulfill the mass conservation equation:

$$\sum_c \dot{m}_c^{k+1} = \sum_c \dot{m}_c' + \sum_c \dot{m}_c^k = 0 \tag{3.18}$$

with $\dot{m}_c'$ being the mass flux correction, v' and p' the velocity and pressure correction, respectively.
Recalling the discrete momentum equation (3.15). Denoting the pressure dependence of the right hand side explicitly yields:

$$a_P v_P^k + \sum_C a_C v_C^k = \nabla p_P^k \delta V_P + c(v^k, f) \tag{3.19}$$

with δV_P the discrete cell volume of cell P.From this the velocity at step k is obtained by:

$$v_P^k = \frac{1}{a_P} \left(- \sum_C a_C v_C^k + \nabla p_P^k \frac{\delta V_P}{a_P} + c(v^k, f) \right) \tag{3.20}$$

Subtracting equation (3.20) for step $k+1$ from (3.20) for step k results in a description of the velocity correction:

$$v_P' = \underbrace{-\frac{1}{a_P} \sum_C a_C v_C'}_{\approx 0} + \nabla p_P' \frac{\delta V_P}{a_P}. \tag{3.21}$$

Note that, due to the fact that the grid displacement stays constant within the pressure correction cycle, the grid flux in a_P and a_C cancels out at this point. Thus the pressure correction for moving grids acts analogously to its non-moving counterpart. Furthermore, $c(.)$ is an explicit contribution to the momentum equation, thus stays constant when switching from k to $k+1$.
The under-braced part is set to zero following the Semi Implicit Method for Pressure Linked Equations (SIMPLE) rule. This is a feasible simplification, as it will tend to zero at convergence.
In order to describe the velocity at the cell face c, the "selective interpolation" is used by which only the geometry variables are interpolated by TBI. As the pressure is given on the grid cell centers and therefore their gradient

at the cell face c can be derived directly from the pressure at P and C, the velocity at the cell face is

$$v'_c = \nabla p'_c \left(\frac{\delta V_c}{a_P}\right)^{\mathrm{TBI}} \tag{3.22}$$

Inserting the velocity correction in terms of pressure correction from the latter equation into the mass conservation equation (3.18) results in the pressure correction equation

$$\sum_c \underbrace{\rho_f \nabla p'_c \left(\frac{\delta V_c}{a_P}\right)^{\mathrm{TBI}} \delta A_c}_{\dot{m}'_c} + \sum_c \underbrace{\rho_f v_c^k \delta A_c}_{\dot{m}_c^k} = 0 \ . \tag{3.23}$$

Due to the non-staggered grid approach used in this work, special care has to be taken when discretizing the velocity v^k in the mass conservation equation. Due to the interpolation of cell centered gradients the pressure field can decouple from the velocity field and decompose into 4 separate fields (checkerboard distribution). According to Rhie and Chow [49] this is a result from the $2\Delta x$ discretization of the pressure field. To solve this problem they proposed to correct the old mass flux by an alternative pressure discretization involving a $1\Delta x$ difference quotient, which results in:

$$\dot{m}_c^k = \rho_f \delta A_c \left(v_c^{k,\mathrm{TBI}} + \left(\frac{\delta V_c}{a_P} \nabla p_c^k\right)^{\mathrm{TBI}} - \left(\frac{\delta V_c}{a_P}\right)^{\mathrm{TBI}} \nabla p_c^k \right) \tag{3.24}$$

By this remedy the $1\Delta x$ pressure oscillation can be resolved and an alternating pressure field can be avoided.
In order to reduce the number of bands in the resulting matrix (thus gaining the possibility to use the same SIP solver as for the momentum equation), the pressure correction gradient is separated from its non-orthogonal terms. For a detailed description see Lehnhäuser [39], Rhie [49] , or Waclawczyk [72].

$$\widehat{\nabla p'_e} = (F_e)^{-1} \begin{pmatrix} \frac{p_E - p_P}{l_{E,P}} \\ 0 \end{pmatrix} \tag{3.25}$$

Introducing this reduced gradient into the mass flux correction term of equation (3.18), the resulting discrete pressure correction equation then states:

$$\begin{aligned} & \sum_c \rho_f \widehat{\nabla p'_c} \left(\frac{\delta V_c}{a_P}\right)^{\mathrm{TBI}} \delta A_c \qquad (3.26) \\ = \ & -\sum_c \rho_f \delta A_c \left(v_c^{k,\mathrm{TBI}} + \left(\frac{\delta V_c}{a_P} \nabla p_c^k\right)^{\mathrm{TBI}} - \left(\frac{\delta V_c}{a_P}\right)^{\mathrm{TBI}} \nabla p_c^k \right) \end{aligned}$$

After solving the above equation for p', the mass flux, velocity and pressure field is updated:

$$\dot{m}_c^{k+1} = \dot{m}_c^k + \rho_f \widehat{\nabla p'_c} \left(\frac{\delta V_c}{a_P}\right)^{\mathrm{TBI}} \delta A_c \tag{3.27}$$

$$v_P^{k+1} = v_P^k + \nabla p'_P \frac{\delta V_P}{a_P} \tag{3.28}$$

$$p_P^{k+1} = p_P^k + \pi p'_P \tag{3.29}$$

with π the underrelaxation factor for the pressure coupling.

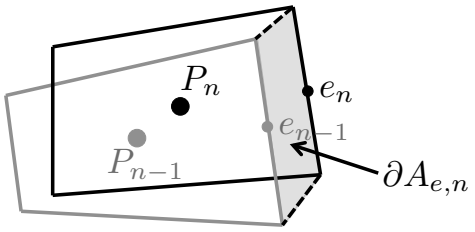


Figure 3.1: Volume swept over by cell face e from time step $n-1$ to n

The Discrete Grid Flux The mass flux for the momentum equation is assembled within the pressure correction equation. As shown above, the grid flux does not influence the pressure correction, but arises in the convective term of the momentum equation. To obtain the grid flux one must measure the volume of a virtual cell bounded by a cell face at time step $n-1$, the same cell face at the current iteration and the linear interpolation of its boundaries. The virtual cell for the east side shown as shaded region in figure 3.1. The virtual cell is then triangulated and its volume is computed and stored in ∂A_c. The grid flux in time step n then denotes:

$$\dot{g}_c = \rho_f v_c^g \delta A_c = \rho_f \left(\frac{3\partial A_{c,*} - \partial A_{c,n-1}}{2\Delta t} \right) \tag{3.30}$$

in which the asterisk denotes the current deformation as in the current FSI iteration. With the mass flux from equation (3.24) and its correction from equation (3.27) the resulting flux for the momentum equation is

$$(\dot{m}_c^g)^{k+1} = \dot{m}_c^{k+1} - \dot{g}_c \tag{3.31}$$

The Discrete Space Conservation Law According to Farhat [21] the discrete Space Conservation Law (SCL) or discrete Geometric Conservation Law is a necessary and sufficient condition to preserve the stability of a time integration scheme on moving grids. The discrete SCL associated with the BDF 2 discretized equation (2.19) as follows:

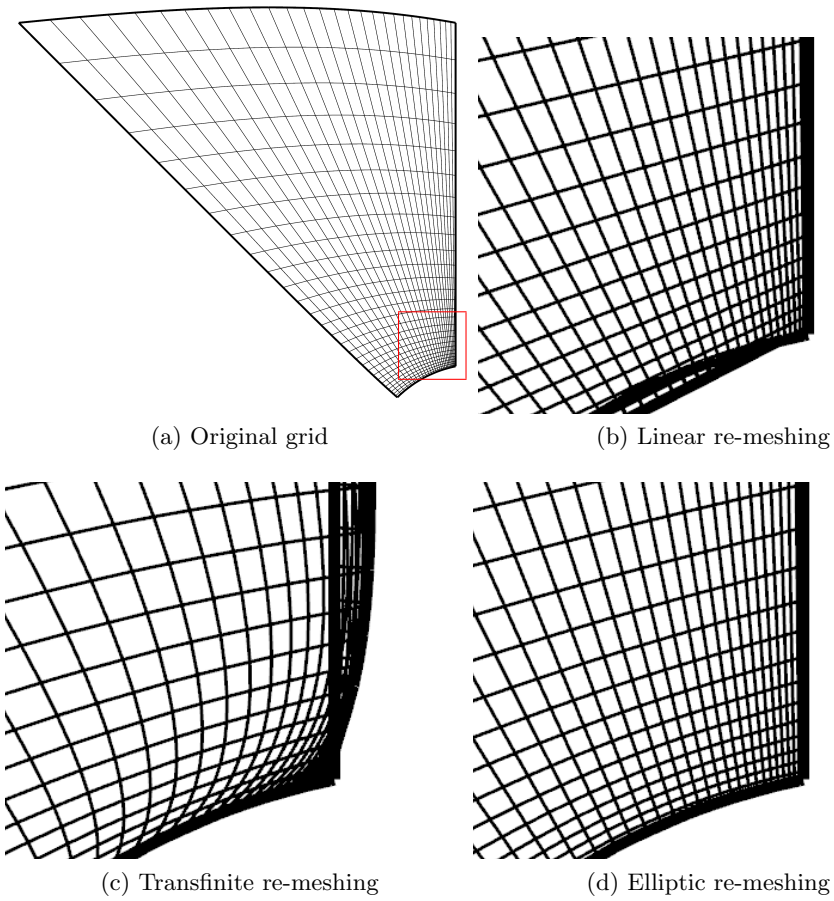
$$3\delta V_{P,n} - 4\delta V_{P,n-1} + \delta V_{P,n-2} = 3\sum_c \partial A_{c,n} - \sum_c \partial A_{c,n-1} \tag{3.32}$$

in which the discrete cell volumes $\delta V_{P,n}$ are triangulated according to Davies [13] and the right hand side is discretized according to Guillard and Farhat [27] with ∂A_c as in figure 3.1.

3.2.3 Grid Movement

As a block-structured fluid solver is used, the grid movement is split up into three stages. First the block edges are computed, second the block faces are moved accordingly, and third the inner grid points of the blocks are moved. There are two possibilities for the edge movement. Either by linear interpolation of the two adjacent corners, or by a cubic spline approximation. To determine the additional degrees of freedom of a cubic approximation in between two points, the angles in between all edges intersecting at a corner are to remain unaltered. In this work for all blocks in the fluid domain, which are actually moved, the cubic approximation is used.
For the movement of block faces there are three possibilities. Linear interpolation, TransFinite Interpolation (TFI), or elliptic movement. In the following the principle operation of these methods is described. For details on the implementaion see Pironkov [47].
In the linear interpolation the grid points in between the two opposing edges are distributed along the convex combination of the respective edge points while preserving the relative spacing of the original grid. With regard to CPU time, this is the cheapest method, but restricted to very simple geometries. Also, due to the poor quality of the resulting grid, the overall CPU time for the coupled problem can be even higher than for the other methods.
The TFI is another algebraic approach. The grid points are distributed along the linear combination of all four respective edge points by a bi-linear interpolation. The relative spacing is again retained from the original grid. This method is more expensive than the linear interpolation, but far more robust and better grid quality can be expected. Thus, for most cases this is the preferred method.
In the elliptic grid movement the face in question is mapped to a reference domain and the points are distributed by solving the Poisson equation. The distribution along the bordering edges is used as boundary conditions for the resulting system. The discrete system is then solved by Gauß Seidel iterations. This is by far the most expensive movement method, but also the most robust. Furthermore, as the solution of the Poisson equation implies a higher order of differentiability than the above mentioned methods, the resulting distribution is smooth.

The need for the elliptic grid movement is easily shown with the help of figure 3.2. In figure 3.2a one block of a grid with its original node distribution is shown. Figure 3.2b shows the lower part of the grid after applying linear grid movement with zero displacement. Disregarding the poor grid quality in the upper, coarser part, the concave edge cannot be handled by this grid movement. It produces degenerate grid cells at the lower edge. Figure 3.2c shows the lower part of the grid after applying zero displacement TFI grid movement. The mesh quality throughout the block is widely



(a) Original grid (b) Linear re-meshing

(c) Transfinite re-meshing (d) Elliptic re-meshing

Figure 3.2: Different gird displacement techniques applied to a zero displacement case

preserved, but the accumulation of grid points at the right corner cannot be mapped correctly. At the right hand side the grid cells are degenerated. This is primarily due to the accumulation being weak at the upper part and becoming stronger towards the lower edge. The last figure 3.2d shows the lower part of the grid after zero displacement elliptic grid deformation was applied. For this test case, the elliptic grid movement is the only method able to re-mesh the grid without degenerating any grid cells.
The inner blocks are moved by the three dimensional extension of the face movement method. As all three methods have their advantages and drawbacks in terms of accuracy and computational cost, all three types of movement are used in this work.

3.3 Solvers

3.3.1 Applied Solvers

The discrete momentum equations are linearized by a fixed point iteration in the fluid code. Given a function f of x, if the assumptions of the Banach fixed point theorem are fulfilled the iteration $x_{n+1} = f(x_n)$ converges at least linearly to a unique point x_*. To solve the resulting linear system as well as the pressure correction equation, the Strongly Implicit Procedure (SIP) by Stone [61] is used. This employs an Incomplete Lower Upper (ILU) factorization in which there are no additional diagonals filled in in the system. This means the lower and upper matrix have a similar sparsity structure as the original matrix. The structure solver uses Newton's method for the nonlinear system. This again implies linear convergence if the assumptions from the Banach fixed point theorem are met and under certain circumstances even implies local quadratic convergence. The linear system is solved by the Conjugate Gradient (CG) method. In this case this can be seen as a direct solver, which at best and with ideal preconditioning (e.g. by a multigrid solver as in Herzog and Sachs [31]) requires $\mathcal{O}(n)$ time to solve a linear system of size n to machine accuracy. In the multigrid implementation the Gauß-Seidel method is used as smoother. Its convergence is assured if either the system matrix is symmetric and positive definite or strictly or irreducibly diagonal dominant. It implies rather poor convergence properties in comparison to today's linear solvers, but it employs good smoothing properties. On the coarsest grid a direct Lower Upper decomposition (LU) is used. This solver requires a run time of $\mathcal{O}(n^3)$ to solve a linear system of size n, which is rather poor performance. Due to fact, that this is only used on the coarsest grid, n is a small number (compared to the original system) and its run time is inferior to the overall multigrid run time.

For more information on any of the above methods refer to any standard numerical analysis textbook, Saad [51] or Stoer and Bulirsch [60].

3.3.2 Multigrid Method

Here, the basic idea of the geometric multigrid method, as used in this work is described. The multigrid method can be used in its linear and nonlinear form and implies a run time of $\mathcal{O}(n)$ for a (non)linear system of size n. According to Ferziger [23] the number of fine grid iterations in multigrid is independent on the number of grid points.
For basic iterative methods (Jacobi, Gauß Seidel, SOR, ILU, SIP, ...) the rate of convergence is dependent on the spectral radius of its iteration matrix, thus dependent on the eigenvalue with largest absolute value. Hackbusch [28] has shown that for elliptic problems and many basic iterative methods the largest eigenvalue is associated with a smooth eigenvector.

Thus, these methods yield a fast reduction of high frequency errors, and a poor performance for smooth errors.
The idea of multigrid method is to decompose the current system error e_n in iteration n into modes of different frequency as for the Fourier transform. Applying one of the above mentioned solvers, reduces the high frequency errors while leaving the low frequency errors essentially untouched. In the following this will be referred to as smoothing. Then the problem is formulated on a coarser grid, and the intermediate solution is represented on that grid. This process is called restriction. Due to the coarser resolution on this grid, wavelengths which were of low frequency on the fine grid are now of high frequency and can be reduced effectively by smoothing. A representation of the coarse grid solution, or its correction towards the intermediate solution, is found on the fine grid and set as the new solution iterate, or added to the current solution iterate, respectively. This is called prolongation.
The multigrid algorithm in this work is described for consecutive coarser grids, in which a grid with step size $2h$ consists of every second point (in every dimension) of a grid with step size h. This is not a necessary restriction, but coincides with the implementation in this work. The multigrid method is the recursive application of the two grid method to consecutive coarser grids. Thus, in the following only the two grid method is described.
In multigrid methods there are various ways to cycle through the different

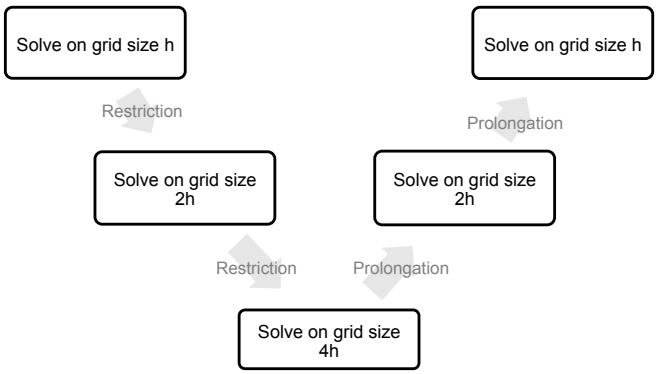


Figure 3.3: Schematic V-cycle

grid levels, the V-, F-, and W-cycle to name a few. This work focuses on the implementation of the V-cycle, which consists of first unidirectional restricting and smoothing steps through all grids until the coarsest grid is reached. Then one uses unidirectional prolongating and smoothing steps up to the finest to complete one cycle. Figure 3.3 shows the course of one V-cycle.

Please note, that when referring to the multigrid method within this work

the geometric multigrid method is meant. The algebraic multigrid method is not subject of this work. For more information on this see Ruge and Stüben [50], Schmid [57], or Trottenberg et al. [68].

Linear Multigrid

First, a few elements of the multigrid method and its course of events is described using a linear version of the geometric multigrid method or Multi-Level Adaptive Technique as introduced by Brandt [7]. For more information on this see e.g. Wesseling [75].

Smoothing Property The multigrid method uses linear system solvers for the smoothing iterations and the coarse grid solve. On the fine grids the ability to effectively smooth an error is more important than the absolute error reduction. The smoothing property states that the solver reduces the high frequency errors without amplifying the low frequency errors. Hackbusch [28] verified the smoothing properties for basic linear solvers such as Jacobi, Gauß Seidel, or SOR. He also stated that the smoothing property of the Gauß Seidel method is superior to that of the Successive Over-relaxation (SOR) although its error reduction is inferior. Wittum [76] showed the smoothing property for the ILU linear solver.

Restriction & Prolongation As the multigrid method operates on several grids an exchange routine must be defined. The restriction operator from grid step size h to $2h$ is called I_h^{2h} and its counterpart, the prolongation from $2h$ to h, I_{2h}^h. The order of interpolation is set according to the order of the underlying pde. According to Hackbusch [28] the sum of orders of restriction and prolongation has to be at least the order of the pde. As the Navier Stokes as well as the elasticity equation are of second order, a first order interpolation for restriction and prolongation is sufficient.

Correction Scheme The correction scheme or defect correction uses the residual equation to assemble the coarse grid system. The scheme is briefly introduced using a model problem. Let

$$A^h x^h = b^h \tag{3.33}$$

be a discrete representation of a set of linear pdes and x_*^h the exact solution to it. Smoothing on the fine grid results in an intermediate solution x_0^h. The fine grid residual is defined as

$$r^h = b^h - A^h x_0^h. \tag{3.34}$$

With $e^h = x_*^h - x_0^h$ being the current fine grid error, $b^h = A^h x_*^h$, and A^h a linear operator, (3.34) can be rewritten as

$$A^h e^h = r^h. \tag{3.35}$$

This system looks similar to (3.33) but, as a result of the smoothing property, the error e^h is smooth in contrast to the intermediate solution and therefore can be restricted more effectively. This resulting *defect equation* is now to be solved on a coarser grid with step size $2h$. Applying the restriction operator I_h^{2h} to the latter equation yields

$$A^{2h}e^{2h} = r^{2h}. \tag{3.36}$$

After solving (3.36) its solution e_*^{2h} is prolongated and added to the intermediate fine grid solution as a correction

$$x^h = x_0^h + I_{2h}^h e_*^{2h}. \tag{3.37}$$

In order to eliminate the high frequency errors due to the interpolation the fine grid system is smoothed once again.

Nonlinear Multigrid

Considering nonlinear problems the correction scheme cannot be used for nonlinear problems as the equivalence between (3.34) and (3.35) only holds for linear A^h. For these cases there are two main approaches to use the multigrid method. The first one consists of linearization via Newtons method (or one of its derivatives) and solving the linear problem with the correction scheme, the so called Newton Multigrid. The second one uses an extension to the correction scheme which covers nonlinear problems as well, namely the Full Approximation Scheme (FAS).

Newton Multigrid The Newton Multigrid is the straightforward approach to solve nonlinear problems with a linear solver. For definition and applications see Brandt [7]. A disadvantage of this scheme is the absence of the current iterate of the unknown on coarser meshes. Thus, an application to local refinement as in Gauß [25] is not possible. Furthermore, although Newton's method implies quadratic convergence, this can only be achieved using the optimal number of V-cycles. See Hackbusch [28] for details.

Full Approximation Scheme The FAS is the nonlinear system solver used in this work. It was first introduced by Brandt [8]. A brief outline of the approach is given here, for further details refer to Brandt [8], Briggs et al. [11], Hackbusch [28], or Schäfer [54].
A nonlinear model problem with a nonlinear dependent right hand side is used. This right hand side is chosen merely for simplification of the notation in chapter 4, as dependent right hand sides appear in the boundary conditions of a coupled FSI problem. Let

$$A^h(x^h) = b^h(x^h) \tag{3.38}$$

be a set of nonlinear pdes. Analog to the correction scheme, the system is solved a few times on the fine level by an appropriate nonlinear solver, resulting in x_0^h, which is the solution to

$$r^h = b^h(x_0^h) - A^h(x_0^h). \tag{3.39}$$

Due to the nonlinearity the full approximation of the current value x^h as opposed to the error in the linear case has to be restricted. This gives the Full Approximation Scheme its name. Subtracting (3.39) from (3.38) and applying the restriction operator I_h^{2h} yields

$$A^{2h}(x^{2h}) = b^{2h}(x^{2h}) - \left(b^{2h}(x_0^{2h}) - A^{2h}(x_0^{2h})\right) + I_h^{2h} r^h \tag{3.40}$$

in which $x_0^{2h} = I_h^{2h} x_0^h$ is the restricted intermediate fine grid value. Solving the latter equation yields x_*^{2h}. As for the correction scheme, the intermediate fine grid solution is corrected only by the prolongated coarse grid correction, not the prolongated value. Therefore the restricted intermediate solution is subtracted from the coarse grid solution and the resulting correction is prolongated and added to the intermediate fine grid solution.

$$x^h = x_0^h + I_{2h}^h \left(x_*^{2h} - x_0^{2h}\right) \tag{3.41}$$

Again, a few more fine grid solution steps are carried out to eliminate the error introduced from interpolation.

If the underlying problem is linear, the FAS is equivalent to the correction scheme apart from numerical deficiencies. Figure 3.4 illustrates one FAS V-cycle applied to a point load problem. Five Gauß-Seidel sweeps were performed in between every two pictures.

3.4 Extrapolation

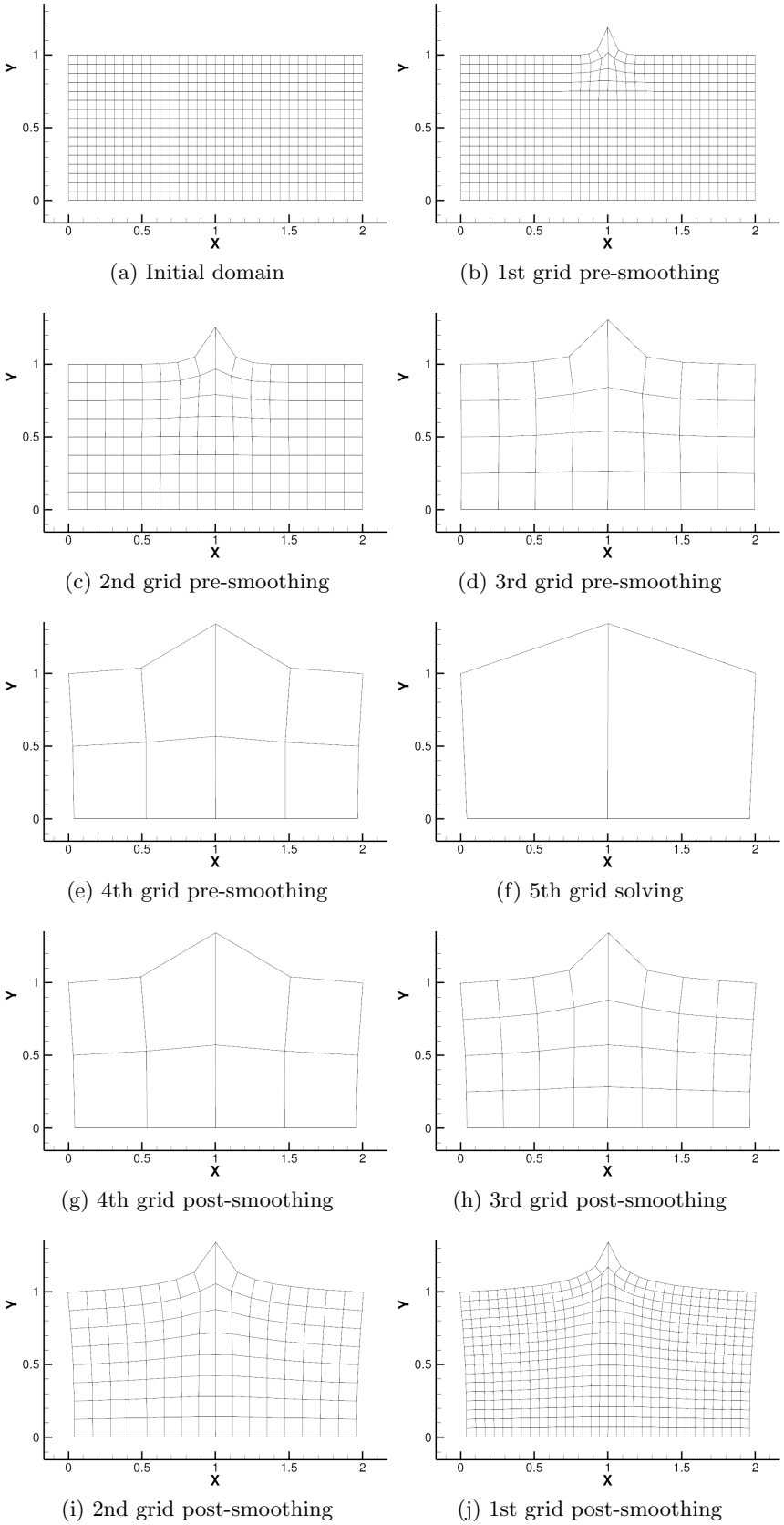
As polynomial extrapolation functions have been proven of value in this field, see [9, 66, 10, 74], they are also used in this work. To demonstrate the effect of different orders of extrapolation, the Lagrangian polynomial interpolation functions are implemented in this work. Given a discrete set of data points (x_i, y_i), the respective polynomial of order n reads:

$$L(x) = \sum_{i=0}^{n} y_i l_i(x) \tag{3.42}$$

with

$$l_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} \ . \tag{3.43}$$

These functions are used for both displacement and force extrapolation. Higher order extrapolation suggest a higher order of accuracy, thus less iterations to convergence, however, their implementation requires more memory and computing time. Furthermore, at a certain point higher order extrapolation begins to produce worse results, as the high order interpolation polynomials produce highly oscillatory results outside the known interval.
The extrapolation is used in this context to produce a beneficial starting point for the computations in every time step. Furthermore, by it's simple implementation it can be easily transferred to different variables.



(a) Initial domain

(b) 1st grid pre-smoothing

(c) 2nd grid pre-smoothing

(d) 3rd grid pre-smoothing

(e) 4th grid pre-smoothing

(f) 5th grid solving

(g) 4th grid post-smoothing

(h) 3rd grid post-smoothing

(i) 2nd grid post-smoothing

(j) 1st grid post-smoothing

Figure 3.4: Example of a 5 level FAS applied to a structure point load problem showing all smoothing steps in the first V-cycle. Figures show the domain with the corresponding discretization after application of the operation in the caption.

Chapter 4

Application of Multigrid Coupling to Fluid-Structure Interaction

In this chapter the multigrid solver is applied to the coupled FSI problem. Due to its nonlinear nature in the fluid as well as in the structure domain, the problem has to be solved via the FAS.
In the following a short overview over possible coupling approaches is given, in order to demonstrate that the multigrid coupling (MG CPL) can not only be perceived as a solver applied to an existing approach, but also as a coupling approach itself.
Next, the implicit partitioned coupling scheme is introduced, as this is the basis for the implementation of the MG CPL. In order to provide for a sensible comparison to the MG CPL, minor changes have been made to the existing approach.
Then, the MG CPL is introduced, based on the theoretical considerations for a nonlinear system solved on multiple grid levels. Even when using the FAS on both individual domains, the MG CPL is distinguished from the partitioned approach by the coarse grid treatment and especially the boundary condition treatment.

4.1 Fluid-Structure Interaction Approaches

In classifying fluid-structure interaction approaches, first the question of boundary realization has to be taken into account. There are two forms of realization. On the one hand the *immersed boundary* method, which uses a Eulerian framework for the fluid and a Lagrangian framework for the structure, linking the two domains at the interface by a Dirac delta function. Its main advantage is the ability to handle arbitrary interface shapes and topology changes of the interface. A drawback is that the interface has to

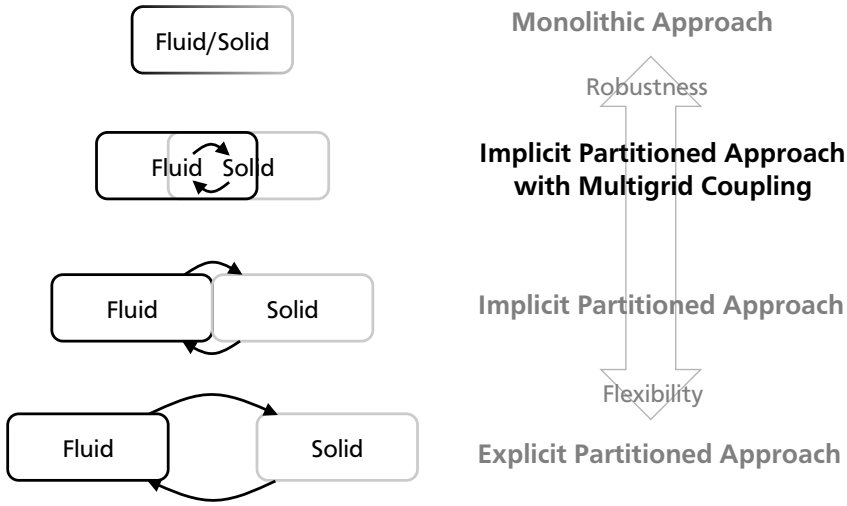


Figure 4.1: Classification of FSI approaches

be tracked and resolved in every time step. Thus this realization is widely used in combination with adaptive grid refinement.
On the other hand there is the *boundary fitted* method based on ALE. The fluid, the structure and the interface are treated as described in chapter 2. Due to the smooth and sharp boundary a high accuracy of the velocity and pressure field can be generated at the boundary. As no topology changes are expected and the applied fluid code can handle even more complex boundaries than necessary for this work, this realization is used.
Next, in order to subdivide the approaches, the closeness of the coupling has to be quantified. The loosest coupling is the *explicit partitioned* approach. Here, the fluid and structure parts are computed sequentially and boundary values are exchanged once per time step. This approach lacks stability and is usually subject to severe time step restrictions. As this approach induces a rather small computational complexity it has been widely used in the history of FSI, e.g. the staggered solution procedure in Felippa and Park [22]. This approach also makes the use of highly developed fluid and structure solvers possible, as only the boundary conditions have to be altered.
A direct generalization to the above approach is the *implicit partitioned* approach. Again, the fluid and structure parts are solved in sequence, but coupling multiple times per time step. The individual parts are sequentially solved until FSI convergence is reached. This results in an implicit dependence of the fluid unknowns on the structure solution and vice versa. As a result, larger time steps are applicable and this approach is numerically more stable, while preserving the interchangeability of the underlying solvers. The drawback is the additional computation time for solving the individual parts multiple times per time step. This approach is used by e.g.

Le Tallec and Mouro [63], Bathe et al. [2], Schäfer [55], and in commercial tools.
Another widely used approach is the *monolithic* approach. Here, the fluid and structure parts are discretized into one single system of equations, which is solved at once in every time step. This results in an even more stable approach, but a new coupled solver has to be developed and maintained. For applications of this approach see e.g. Hübner et al. [35] or Hron and Turek [34].
The *multigrid coupling* approach introduced in this work can be regarded as a mixture of the implicit partitioned and the monolithic approach. It exploits the structure of the multigrid solver, i.e. the split between smoother and inter-grid transfer, and results in one single multigrid solver for the coupled problem using the implicit partitioned coupling approach as smoother. Thus, the resulting approach is more robust than the implicit partitioned coupling approach, but still preserves the interchangeability of the underlying solvers, presuming the (non)linear system solvers are editable. Moreover, the error introduced into the numerical procedure by the coupling is also considered by the multigrid solver as the coupling is carried out on the fine and coarse grids at every smoothing step. This results in a faster FSI convergence compared to the implicit partitioned approach.
Figure 4.1 shows a graphical representation of the above approaches in the context of robustness, flexibility, and the closeness of the individual solvers.

4.1.1 Implicit Partitioned Coupling

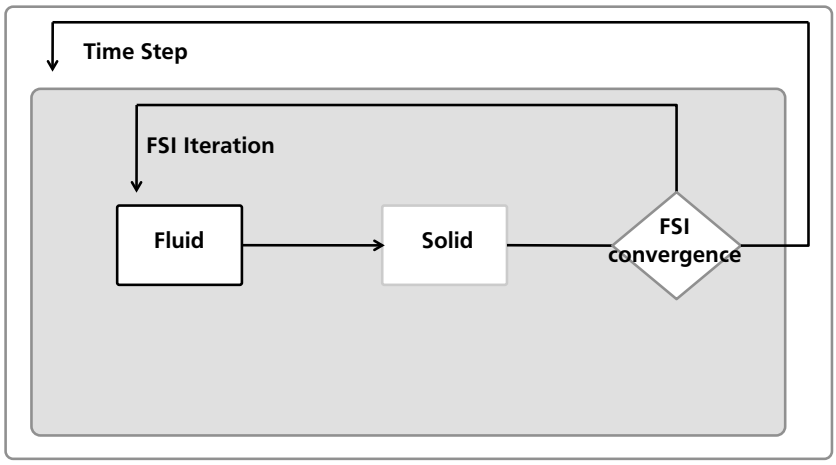


Figure 4.2: Course of events in an implicit partitioned coupling approach

The starting point for the implementation in this work is an implicit partitioned coupling approach with a geometric multigrid fluid solver and

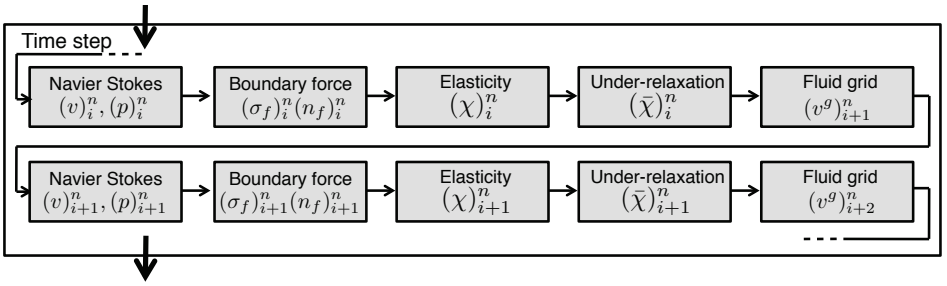


Figure 4.3: Detail of implicit partitioned coupling approach

a conjugate gradient structure solver. The Fachgebiet Numerische Berechnungsverfahren at TU Darmstadt uses the ALE implementation within the finite volume code FASTEST [20] as fluid solver, the finite element code FEAP [65] as structure solver and the code coupling software MpCCI [24] to handle the interface. Figure 4.2 shows a schematic view of the coupling approach.
For a more detailed view, one time step of the native partitioned implicit FSI implementation is depicted in figure 4.3. The ith iteration of the nth time step can be described as follows:

- First the Navier Stokes equation is solved for velocity and pressure $(v)_i^n, (p)_i^n$ in the fluid domain using FASTEST.
- Using this solution the force on the FSI boundary $(\sigma)_i^n (n_f)_i^n$ is derived and sent to the structure solver FEAP via MpCCI.
- Implementing the force as boundary condition, the elasticity equation is solved for the displacement $(\chi)_i^n$ on the structure domain via FEAP.
- Then the structure solution, i.e. the displacement at the interface, is send to FASTEST via MpCCI and is under-relaxed by a constant factor α to ensure convergence of the overall procedure.
- Finally, the grid in the fluid domain is recomputed based on the under-relaxed displacement $(\bar{\chi})_i^n$, and the new grid velocity $(v^g)_{i+1}^n$is computed.

An underrelaxation is introduced in order to stabilize the coupled procedure. At FSI iteration i the displacement at the interface, which was calculated by the structure solver is under-relaxed by a factor α:

$$\bar{\chi}_{i,f}^n = \alpha \chi_{i,f}^n + (1-\alpha)\bar{\chi}_{i-1,f}^n \tag{4.1}$$

This underrelaxation has to be considered when defining the FSI convergence. In order to compare this implementation to the multigrid coupling,

the FSI convergence criterion was improved. In earlier works of Heck [29] and Yigit [78] convergence was reached if

$$\frac{\sum_{k=1}^{N}\left|\left(\bar{\chi}_{i,f}^{n}\right)^{k}-\left(\bar{\chi}_{i-1,f}^{n}\right)^{k}\right|}{N}<\epsilon_{\mathrm{FSI}} \tag{4.2}$$

was valid for the displacement in every spatial direction. N describes the length of the vector of displacements. Substituting (4.1) into the latter equation results in

$$|\alpha|\frac{\sum_{k=1}^{N}\left|\left(\chi_{i,f}^{n}\right)^{k}-\left(\bar{\chi}_{i-1,f}^{n}\right)^{k}\right|}{N}<\epsilon_{\mathrm{FSI}} \tag{4.3}$$

which results in a direct dependence of the convergence criterion, i.e. the number of iterations needed to reach convergence, on the underrelaxation factor. Thus, equation (4.3) omitting the $|\alpha|$ is used to check for FSI convergence in this work. Furthermore, overall convergence in one time step is reached if the fluid and the structure solver have converged in addition to FSI convergence.
The fluid and structure grids at the FSI interface are non-matching. Thus, an interpolation method is employed. For the forces originating at the cell face centers of the fluid grid and destined to the cell corners of the structure grid the interpolation is based on overlapping area weighting. For details see *intersection algorithm* in MpCCI [24]. For the displacement originating at the cell corners of the structure grid and destined to the cell corners of the fluid grid a distance weighted interpolation is used. This is called *minimal distance algorithm* in MpCCI [24].
In every coupling step the fluid solver makes use of its multigrid solver, but the exchange of forces and displacements is restricted to the finest grid level. Also, the discrete space conservation law (3.32) is checked on the finest grid in every FSI iteration.
According to Yigit [78] it is sufficient for this setup to consider the discrete grid flux (3.30) in the momentum equation only on the finest grid level.
As the discrete momentum equation (3.15) accounts for the FAS framework, but the discrete mass equation (3.24) does not, special care has to be taken when restricting the mass and grid flux. The discrete grid flux (3.30) is calculated before restriction and added to the mass flux. Then the momentum equation can calculate its fine grid residual and carry on with the interpolation. The grid flux is then subtracted from the combined flux in order to build the residual for the mass conservation equation.

4.1.2 Multigrid Coupling

In the following the extensions and modifications to the implicit partitioned coupling model to multigrid coupling is described. Special attention has

been given to the boundary conditions in the setting of an FAS of a coupled problem.

Preliminaries in FASTEST As a first step in the implementation the MG CPL into FASTEST the structure of the main subroutines had to be adapted. The new structure is as follows. The outer loop runs over the time steps. The next loop runs over the grid levels in a V-cycle manner. Every iteration of this loop starts and ends at the currently finest grid. The inner loop controls the number of coupling steps used for smoothing. In every loop the coupled problem is solved in the following order. First the problem on the fluid domain is solved with boundary conditions from the last structure solve or, on the coarse grid, from the fine grid. Then, after its boundary conditions (forces) are updated, the structure problem is solved. After that the underrelaxation is applied and the new fluid boundary conditions are constructed. Figure 4.4 schematically shows the nested loops in the new arrangement.

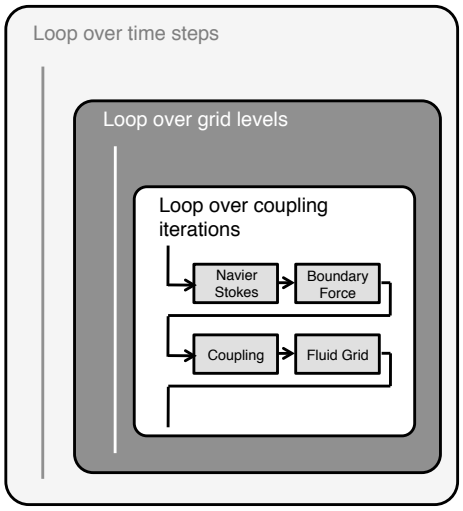


Figure 4.4: Adapted structure of the fluid code FASTEST in preparation of multigrid coupling

The Discrete Momentum Equation Keeping in mind that in equation (3.15) the factors a are dependant on v and χ and the right hand side b is dependant on v, p, and χ, full representation of the FAS is appropriate. Thus, the coarse grid ($2h$) representation of the discrete momentum equation yields:

$$a_P^{2h} v_P^{2h} + \sum_C a_C^{2h} v_C^{2h} = b_P^{2h} - b_{P,0}^{2h} + a_{P,0}^{2h} v_{P,0}^{2h} + \sum_C a_{C,0}^{2h} v_{C,0}^{2h} + I_h^{2h} r^h \qquad (4.4)$$

with

$$r^h = b_P^h - a_P^h v_P^h - \sum_C a_C^h v_C^h. \tag{4.5}$$

As the coefficients a_i^{2h} on the coarse grid are assembled analogously to their fine grid counter pieces, it becomes clear that the discrete grid flux has to be added to the mass flux for the coarse grid momentum equation. The boundary conditions to the momentum equation at the FSI interface are the first part of (2.25). Thus for the fluid velocity on coarse grids at the FSI interface holds

$$v_f^{2h} = v_s^{2h} - v_{s,0}^{2h} + I_h^{2h}\left(v_{f,0}^h\right). \tag{4.6}$$

Note, that v in the momentum equation is defined by equation (A.6) in Eulerian coordinates, thus in contrast to the ALE velocity existent at the FSI boundary.

The Discrete Mass Conservation Equation In order to solve the pressure correction equation, the mass flux must be taken into account. Mass fluxes are restricted conservatively, i.e. the initial coarse grid mass flux of a cell is the sum of all fine grid mass fluxes in the corresponding fine grid cells. The coarse grid mass flux is then updated by velocity corrections, which means equation (3.24) yields omitting the iteration number:

$$\begin{aligned} \dot{m}_c &= \sum_i \left(\dot{m}_i\right)^h \\ &+ \rho_f \delta A_c^{2h} \left(\left(v_c^{\mathrm{TBI}}\right)^{2h} - \left(v_{0,c}^{\mathrm{TBI}}\right)^{2h}\right. \\ &+ \left.\left(\frac{\delta V_c^{2h}}{a_P^{2h}} \nabla p_c^{2h}\right)^{\mathrm{TBI}} - \left(\frac{\delta V_c^{2h}}{a_P^{2h}}\right)^{\mathrm{TBI}} \nabla p_c^{2h}\right) . \end{aligned} \tag{4.7}$$

Again, the mass conservation is examined in Eulerian formulation, thus the boundary velocity is existent. Furthermore, the pressure gradient at the interface equals 0. Its boundary condition at the FSI boundary then only consists of the mass flux introduced by the moving structure:

$$\dot{m}_c^h = \rho_f \delta A_c v_f^h \ . \tag{4.8}$$

As the velocity at the interface changes at every coupling iteration, the resulting boundary conditions for the fluxes have to be recalculated every coupling iteration, too. Recalling the calculation of inner fluxes, equation (4.7), the boundary fluxes are calculated accordingly as sum over the fine grid fluxes added to a corrected coarse grid flux originating from the velocity correction:

$$\dot{m}_c^{2h} = \sum_i \dot{m}_i^h + \rho_f \delta A_c^{2h} (v_f^{2h} - v_{f,0}^{2h}) \ . \tag{4.9}$$

The Space Conservation Law The SCL is checked on every grid level separately, ensuring the stability of the time discretization scheme. The discrete SCL on the coarse grid is the straightforward coarse grid representation of (3.32) as the displacement on the coarse grid is defined by the coarse grid velocity as a correction to the fine displacement. The coarse grid geometry information of the previous time steps is taken from the coarse grid displacement of the according step. The coupled codes proceed to the next time step at FSI convergence. This ensures the coarse grid displacement to coincide with the fine grid displacements where applicable and thus the correct use of the SCL.

The Discrete Grid Flux As the grid moves in every FSI iteration, The change in the discrete grid flux has to be considered in every grid level. The discrete grid flux is computed as in (3.30) based on the corrected coarse grid velocities and added to the inner convective flux. At the FSI boundary the coarse grid flux is not computed, because a Dirichlet boundary condition for the velocity is present.

The Discrete Elasticity Equation The coarse grid system for the discrete elasticity equation is again the FAS applied to equation (3.6). This yields

$$\begin{aligned} & K^{2h}(\chi^{2h}) + C^{2h}\dot{\chi}^{2h} + M^{2h}\ddot{\chi}^{2h} \\ = \;& f^{2h} - \left[f_0^{2h} - K^{2h}(\chi_0^{2h}) - C^{2h}\dot{\chi}_0^{2h} - M^{2h}\ddot{\chi}_0^{2h}\right] + I_h^{2h} r^h \end{aligned} \tag{4.10}$$

with

$$r^h = f_0^h - K^h(\chi_0^h) - C^h\dot{\chi}_0^h - M^h\ddot{\chi}_0^h. \tag{4.11}$$

The outer forces on the right hand side are arguably independent of χ and thus a more compact form of the coarse grid equation can be found. With a view to section 4.2.2 this form is more suitable to compare the actual implementation to the underlying theory. The coarse grid von Neumann boundary condition at the FSI interface is defined to be corresponding to the coarse grid system:

$$\frac{1}{\det(F^{2h})} F^{2h} S^{2h} \left(F^{2h}\right)^T n^{2h} = \sigma^{2h} n^{2h} - \sigma_0^{2h} n_0^{2h} + I_h^{2h}\left(\sigma_0^h n_0^h\right) \; . \tag{4.12}$$

Convergence Criterion The choice for an appropriate convergence criterion is nontrivial as convergence may only appear on the finest grid, but due to the utilization of the coupled solution as a smoother, small changes in between two fine grid solves are expected even when far from convergence. Therefore, the summed change of displacements throughout an entire V-cycle is taken into account when checking for convergence. The convergence

criterion reads:

$$\frac{\sum_{k=1}^{N} \left| (\chi_i^n)^k - (\bar{\chi}_{i-1}^n)^k \right|}{N} < \epsilon_{\mathrm{FSI}} \tag{4.13}$$

in which i indicates the number of V-cycle.

Figure 4.5 shows an overview of the overall MG CPL process. This can also

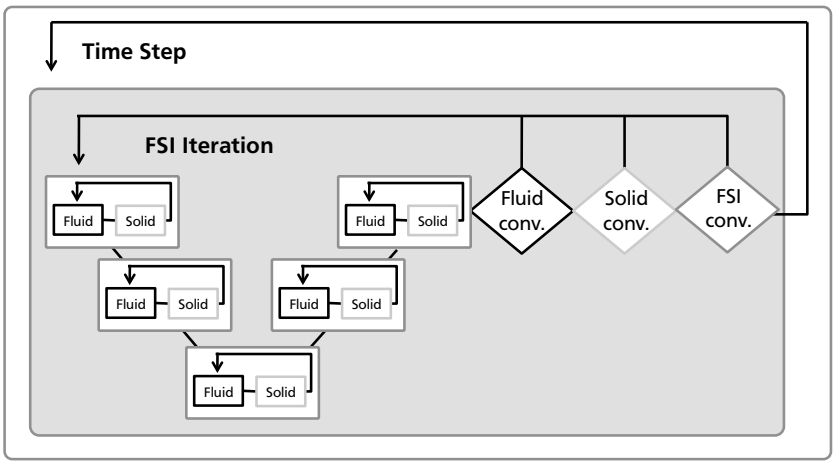


Figure 4.5: Course of events in a multigrid coupling approach

be seen as a single multigrid implementation in each code running in simultaneous V-cycles. On every grid level the coupling procedure as described in figure 4.3 is applied.

4.2 Multigrid Coupling Implementation

In this section the development of the implementation of the MG CPL algorithm is described and tested one by one. The implementation is based on the structure solver FEAP [65] version 8.2, the current version of the fluid solver FASTEST [20], and the code coupling interface MpCCI [24] version 3.1.1. All computations in this section were executed on an Intel Core i7 2.93GHz.

The implementation and tests for the multigrid solver are only shown for the structure part, as the multigrid implementation in FASTEST has been tested and validated before, see e.g. Durst and Schäfer [19]. The MG CPL implementation includes extensions to both FASTEST and FEAP. Thus, the implementation is shown for both sides.

All structural multigrid implementations use five sweeps Gauß Seidel as smoother. Successive overrelaxation and Jacobi were tested too, but worse performance of the overall multigrid solver was observed, which is in agreement to Hackbusch [28]. For the non-coupling test cases a LAPACK [1]

direct solver was used on the coarsest grid. Using more Gauß Seidel sweeps or more Newton iterations on the coarser levels that are not the coarsest can accelerate convergence. The same is true for more Newton iterations on the coarsest grid and especially valid for nonlinear test cases. For the sake of simplicity and reliable comparison only one smoothing step (or Newton iteration on the coarsest grid) per grid was used.

All test cases were computed with the MG CPL implementation as well as the standard solvers or partitioned implicit solver respectively. The results of the different solvers on the structure sides varied by less than 0.01% of the measured displacement, thus are not visible in the results given by FEAP.

The standard FEAP convergence criterion, reduce the energy norm by a factor of 10^{-16}, is used. The solution of all methods coincides with the solution of the standard FEAP conjugate gradient solver. The standard FASTEST convergence criterion, the summarized residual to be less than 10^{-7}, is applied. FSI convergence implies FEAP and FASTEST convergence, too. The convergence factor is defined as the average energy norm reduction in between two V-cycles.

4.2.1 Boundary Conditions

The first test cases were chosen to ensure the correct implementation of boundary conditions as well as investigating the correctness and performance of the geometric multigrid implementation in the structure code.

Consideration

In order to ensure the correct treatment of boundary conditions they have to be applied before the restriction on the fine grid and afterwards on the coarse grid. During prolongation attention has to be payed to von Neumann boundary conditions to prevent the introduction of artificial interpolation errors.

Implementation

In FEAP the successive coarser grids are created individually and a mapping in between them is constructed by the *TIE* command. It automatically deletes nodes defined twice as long as their location coincides. In this process the adjacencies of the coarse grid remain untouched, while the storage for the coarse grid variables is unallocated. This allows the use of the preexisting boundary condition implementation of FEAP. This is organized as follows. Dirichlet boundary conditions affect the size of the tangent matrix and residual vector. Thus, an additional mapping is needed in FEAP. For

every grid point on a Dirichlet boundary one row and column in the tangent and one entry in the residual is deleted. This compressed form has to be taken into account in the interpolation and the application of boundary conditions on coarse grids. As the interpolation is defined for the full system, a mapping in between the compressed and decompressed form has to be applied at all grid levels. Von Neumann boundary conditions are embedded into the tangent and residual without altering their size.
Before prolongation the Dirichlet boundary points on the coarse grid are overwritten with their interpolated fine grid values. If this step was skipped the coarse grid correction on these points were nonzero and by interpolation an additional error were introduced to the points in the vicinity.
Von Neumann boundary conditions on the fine grids are stored before restriction and restored during prolongation, as the same vector is used on coarser grids, and thus is overwritten.

Testing

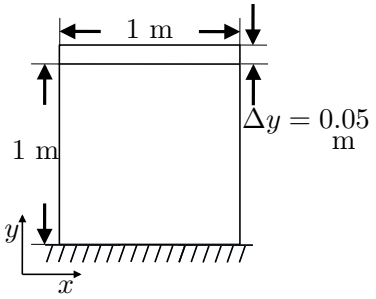


Figure 4.6: Initial conditions for Dirichlet boundary condition test case

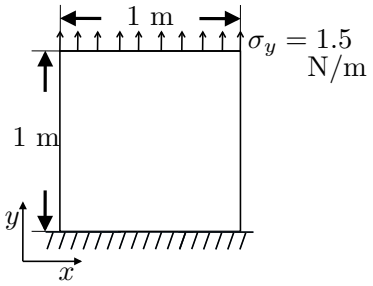


Figure 4.7: Initial conditions for von Neumann boundary condition test case

The first test cases are employed to ensure the correct implementation of the boundary conditions. For this purpose the results as well as the performance of the multigrid code is observed. A linear elastic material as well as linear strain-stress relationships are used in order to eliminate the possibility of additional sources of error, as these test cases focus on the correct implementation of boundary conditions.

Dirichlet Boundary Condition A first test case consists of a linear elastic material with Lamé constants $\lambda_s = 17.31$ kg/ms^2 and $\mu_s = 11.54$ kg/ms^2. The domain is discretized by first order quadrangles. The number of quadrangles reach from 64 to 262,144. The initial domain is the unit square ($[0,1]$ m$\times[0,1]$ m) with a homogeneous Dirichlet boundary condition at the bottom ($y = 0$) and a displacement of $\Delta y = 0.05$ m in y direction

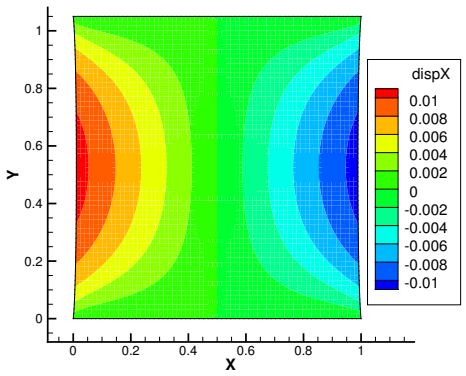


Figure 4.8: x deformation for Dirichlet boundary condition test case

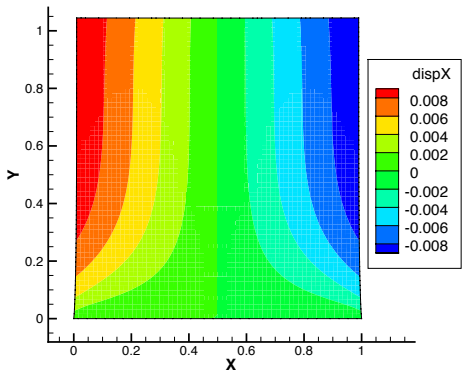


Figure 4.9: x deformation for von Neumann boundary condition test case

at the top ($y = 1$). Homogeneous von Neumann boundary conditions are applied at the remaining sides. Figure 4.6 shows a schematic sketch of the domain with its boundary conditions. As solvers a 2, 3, 4, and 5 level multigrid solver as well as the standard FEAP conjugate gradient solvers are used. The size of the displacement is 5% of the object size, therefore sufficiently small for a linear material model to produce reliable results.
The deformation of the domain, the convergence factors of the multigrid solvers, and CPU time are used as quantities for comparison. As a constant displacement in y direction is applied, the displacement in x direction of point A located at $(0.0, 0.5)$ is monitored to observe the deformation.

Von Neumann Boundary Condition The second test case is used to verify the correct implementation of the von Neumann boundary condition. Once again the unit square is used as the initial domain and the same strain-stress relationship as well as material model as for the former case are applied. Homogeneous Dirichlet boundary conditions at $y = 0$, a constant force per length of $\sigma_y := \sigma n_y = 1.5$ N/m in positive y direction along $y = 1$ and homogeneous von Neumann boundary conditions at the remaining sides are applied. The load has been chosen to excite only sufficiently small displacements. Therefore, the linear material model is suitable. Figure 4.7 shows a sketch of the initial domain and its applied loads. The domain is discretized by 64 through 262,144 quadrangles. Once again the multigrid solvers with up to 5 grids as well as the standard FEAP solver are used for comparison. The observed quantities are the CPU time, the convergence factor for the multigrid solvers, and the x displacement of point A located at $(0.0, 0.5)$.

Figures 4.8 and 4.9 show the resulting domain for 8,192 degrees of free-

# Degrees of Freedom	x-Displ of A [10^{-2} m]
128	1.0887
512	1.0864
2,048	1.0858
8,192	1.0857
32,768	1.0857
131,072	1.0857
524,288	1.0857

Table 4.1: Displacement of point $A = (0.0, 0.5)$ in the Dirichlet boundary condition test case

# Degrees of Freedom	x Displ of A [10^{-3} m]
128	9.5271
512	9.5360
2,048	9.5409
8,192	9.5434
32,768	9.5445
131,072	9.5449
524,288	9.5451

Table 4.2: Displacement of point $A = (0.0, 0.5)$ in the von Neumann boundary condition test case

dom. In tables 4.1 and 4.2 a detailed view of the deformation is given. The Dirichlet as well as the Neumann test case approach the grid independent solution. In figures 4.10 and 4.11 the CPU time of the different solvers is plotted against the number of degrees of freedom from the different resolutions in a log-log representation. The label iGrid represents the computation with a multigrid solver with i grid levels. CG stands for the standard FEAP solver. Figures 4.10 and 4.11 clearly indicate the exponential run-time of the standard solver. The multigrid implementation shows the expected linear performance if the maximum of available grid levels is used. Figures 4.12 and 4.13 show the convergence factors for the multigrid computations. The convergence factors stay quite constant throughout the different resolutions, which is an indication for the correct implementation of the multigrid solver. All multigrid implementations need 6 to 7 V-cycles to reach the convergence criterion independent of the resolutions. This is a necessary condition for the linear performance of multigrid solvers.
Comparing figures 4.10 and 4.12 shows, that for example the 2 grid multigrid has a better convergence factor, but needs more CPU time than the other implementations. This again is an indication that the 2 grid multigrid needs much more CPU time per V-cycle, which can be explained by the direct solve on the coarsest grid. By using only 2 grids the influence of the

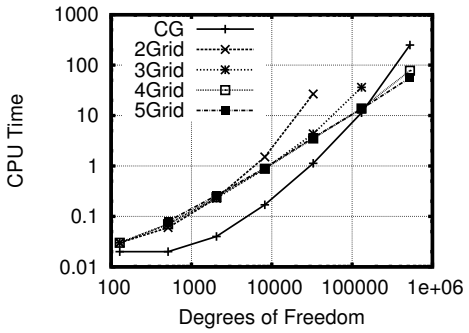


Figure 4.10: CPU time for multigrid and the standard solver on the Dirichlet boundary condition test case

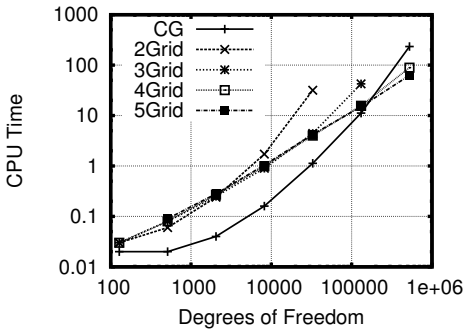


Figure 4.11: CPU time for the multigrid and the standard solver on the von Neumann boundary condition test case

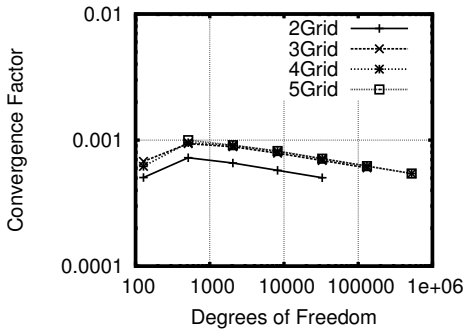


Figure 4.12: Convergence factors for the multigrid solver on the Dirichlet boundary condition test case

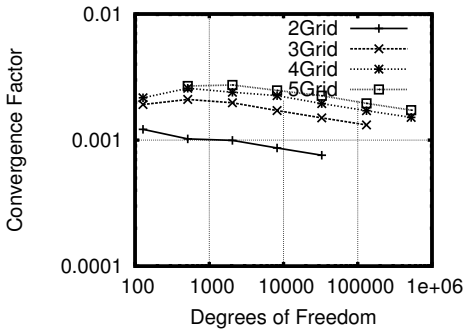


Figure 4.13: Convergence factors for the multigrid solver on the von Neumann boundary condition test case

direct solver on the computing time is visible.

Mixed Boundary Condition In order to test the implementations in a more challenging environment, the next test case uses mixed boundary conditions. This includes different boundary conditions in x and y direction as well as discontinuous boundary conditions along the edges. It is a adaption of the test case from Parsons and Hall [45]. The initial domain is a $[0.0, 2.0]$ m $\times$ $[0.0, 1.0]$ m rectangle. A linear elastic material with Lamé constants $\lambda_s = 17.31$ kg/ms^2 and $\mu_s = 11.54$ kg/ms^2, as well as a linear strain-stress relationship is used. At $y = 0$ a homogeneous Dirichlet boundary condition for the y displacement and a homogeneous von Neumann con-

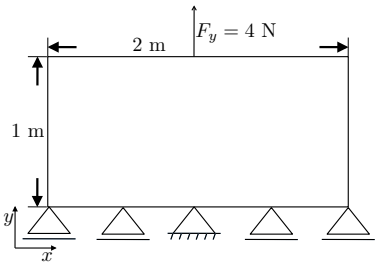


Figure 4.14: Initial conditions for the mixed boundary condition test case

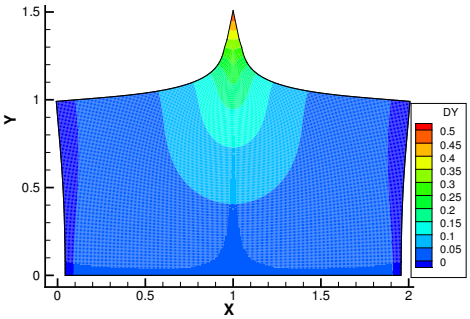


Figure 4.15: y deformation for the mixed boundary condition test case

dition for the x displacement is applied. At $(1.0, 0.0)$ the x displacement is fixed to zero. Furthermore, a point load of $F_y = 4$ N in y direction is applied at $(1.0, 1.0)$. Figure 4.14 shows a sketch of the initial domain and boundary conditions. The domain is discretized by 8×4 through 512 quadrangles.
For comparison the CPU time as well as the convergence factors for the multigrid solvers, and the displacement of point $B = (1.0, 1.0)$ are monitored. This test case is well suited to validate the correct boundary treatment because an edge with a Dirichlet boundary, an edge with a von Neumann boundary and an edge with both boundary conditions can be found.

# Degrees of Freedom	y Displ of B [10^{-1} m]
64	2.4195
256	2.9483
1,024	3.4820
4,096	4.0170
16,384	4.5523
65,536	5.0877
262,144	5.6231

Table 4.3: Displacement of point $B = (1.0, 1.0)$ in the mixed boundary condition test case

Figure 4.17 shows the convergence factors for the multigrid solvers. The convergence factor is higher than for the last test cases, especially for lower resolutions. As the resolution grows, the convergence factor falls below 0.01, which is in the regime of the previous test cases. This can be explained by the challenging boundary conditions.
Table 4.3 shows the displacement of the point $B = (1.0, 1.0)$ with the applied

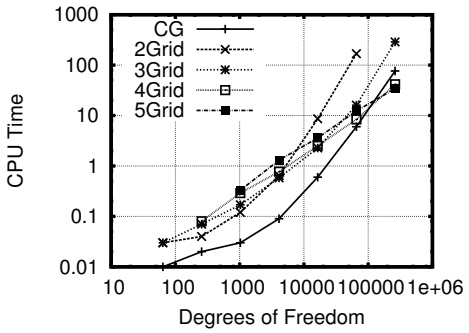


Figure 4.16: CPU time for the multigrid and the standard solver on the mixed boundary condition test case

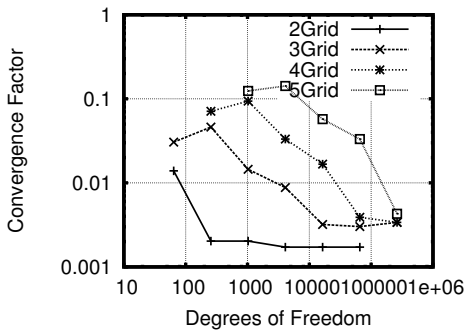


Figure 4.17: Convergence factors for the multigrid solver on the mixed boundary test case

point load for different resolutions. The displacement does not converge to a single solution as the actual solution would be non-physical. The good-natured solutions in this test are due to the limitation of the linear elements with a linear material model. Figure 4.15 shows the final domain for 16,384 degrees of freedom.
Figure 4.16 shows the CPU time of the different implementations. Again, using the maximum number of grids available in multigrid computation results in a linearly dependent computation time on the number of degrees of freedom. The standard solver shows the anticipated exponential behavior. In conclusion, the correct behaviour for the multigrid implementation can be shown for different kinds of boundary conditions. The performance of the multigrid shows the expected linear dependence, although for small and mid-size problems the standard solver is superior concerning CPU time.
Further investigations of this test case can be found in Sachs et al. [52].

4.2.2 Full Approximation Scheme

Consideration

In the linear test cases the FAS is equivalent to the correction scheme. In solving the nonlinear problems with FEAP, the implicit treatment of Newton's method has to be take into account. This means during the assembly of the system matrix from the individual elements, the elements produce directly the residual vector and the Jacobian. At no stage of the program the nonlinear problem as in (3.38) is existent. Thus the FAS cannot be applied directly to this problem.

Implementation

In order to make use of the design of FEAP the FAS scheme has to be built using FEAPs particular elements. On every grid the residual vector $R^h(\chi^h) = f^h - K^h(\chi^h) - C^h\dot{\chi}^h - M^h\ddot{\chi}^h$ and the according tangent matrix $\frac{\partial R^h(\chi^h)}{\partial \chi^h}$ are available. If the FAS approach as described in 3.3.2 would be applied directly to the Newton iteration system supplied by FEAP

$$\frac{\partial R(\chi)}{\partial \chi}\Delta\chi = -R(\chi), \tag{4.14}$$

then the coarse grid system would result in the following

$$\begin{aligned}\frac{\partial R^{2h}(\chi^{2h})}{\partial \chi^{2h}}\Delta\chi^{2h} &= -R^{2h}(\chi^{2h}) + R_0^{2h}(\chi_0^{2h}) + \underline{\frac{\partial R_0^{2h}(\chi^{2h})}{\partial \chi^{2h}}\Delta\chi_0^{2h}} \\ &+ I_h^{2h}\left(-R_0^h(\chi_0^h) - \underline{\frac{\partial R_0^h(\chi)}{\partial \chi^h}\Delta\chi_0^h}\right) .\end{aligned} \tag{4.15}$$

The underlined terms are explained below. The desired system is Newton's method applied to the coarse grid FAS equation of the discrete elasticity equation. One Newton step of equation (4.10) yields

$$\begin{aligned}&\frac{\partial\left(f^{2h} - K^{2h}(\chi^{2h}) - C^{2h}\dot{\chi}^{2h} - M^{2h}\ddot{\chi}^{2h}\right)}{\partial \chi^{2h}}\Delta\chi^{2h} \\ =\ & -f^{2h} + K^{2h}(\chi^{2h}) + C^{2h}\dot{\chi}^{2h} + M^{2h}\ddot{\chi}^{2h} \\ +\ & f_0^{2h} - K_0^{2h}(\chi_0^{2h}) - C^{2h}\dot{\chi}_0^{2h} - M^{2h}\ddot{\chi}_0^{2h} \\ -\ & I_h^{2h}\left[f_0^h - K_0^h(\chi_0^h) - C^h\dot{\chi}_0^h - M^h\ddot{\chi}_0^h\right] .\end{aligned} \tag{4.16}$$

Finally, when comparing equations (4.15) and (4.16), the equivalent parts can be identified. Simple insertion shows that the left hand side of both equations is equal and the right hand side is equal up to the underlined parts of (4.15). To account for this inequality the FAS implementation in FEAP is designed as follows.

First a few Gauß Seidel steps are computed as pre-smoothing. Then, the intermediate fine grid solution is stored and the fine grid residual is computed. Then the residual vector in Newton's method is set up using the actual *build residual*-subroutine from the FEAP call *FORM*. Note, that a flag that should prevent calling multiple times *FORM* without solving in between has to be reset. Then the residual vector, containing the boundary conditions, is decompressed according to section 4.2.1.

The next step is the restriction of displacement, velocity, acceleration, and the residual vector. The intermediate fine grid solutions are restricted by injection and stored in order to compute the coarse grid correction. The residual is restricted by linear interpolation. Here, the second underlined

part of (4.15) is skipped. Another call of FORM on the coarse grid produces the coarse grid residual vector dependent on the interpolated fine grid displacement, velocity, and acceleration. The interpolated fine grid residual is added to the actual coarse grid residual and the sum is stored. Here, the first underlined part of (4.15) is skipped.
To complete the coarse grid system, the Newton iteration is built on the coarse grid using the standard FEAP functions. Then the stored residuals are added to the right hand side.
Solving (or smoothing) the coarse grid system results in a displacement χ_*^{2h}, velocity $\dot{\chi}_*^{2h}$, and acceleration $\ddot{\chi}_*^{2h}$. The correction to the previously stored restricted fine grid solution is prolongated by linear interpolation and added to the intermediate fine grid solution

$$\chi^h = \chi_0^h + I_{2h}^h(\chi_*^{2h} - I_h^{2h}\chi_0^h). \tag{4.17}$$

The velocity and acceleration are constructed analogously. The vector storing the coarse grid displacement and residual is then set to zero and post-smoothing is conducted in the manner of pre-smoothing.

Testing

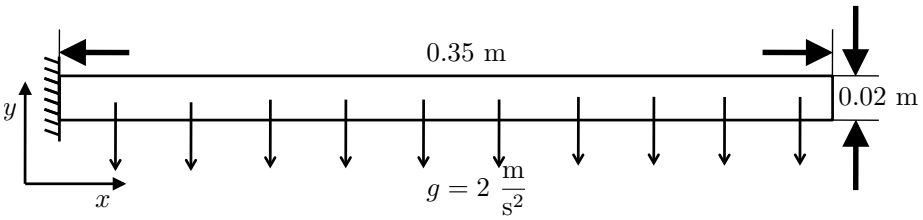


Figure 4.18: Initial conditions for the CSM 1 test case

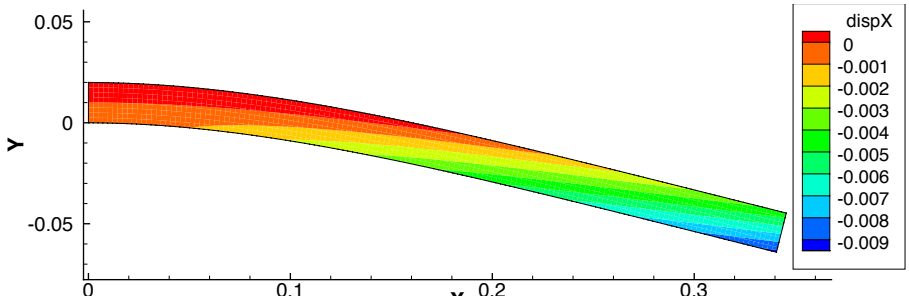


Figure 4.19: Deformed state of the CSM 1 test case (x displacement shown)

A nonlinear test case is used to validate the FAS implementation, since the FAS collapses to the regular correction scheme for linear cases. This test case is Computational Structure Mechanics (CSM) 1 taken from Turek and Hron [69]. The problem domain is a beam $[0.0, 0.02]$ m $\times$ $[0.0, 0.35]$ m with a body load of $2 \cdot 10^3$ N/m^2 applied in negative y direction. Figure 4.18 shows a sketch of the initial domain and applied loads. A homogeneous Dirichlet condition in x and y direction is applied at $x = 0$ and homogeneous von Neumann conditions are applied at the remaining sides. The structure consists of an elastic, compressible material and the St. Venant Kirchhoff material with Lamé constants $\lambda_s = 8 \cdot 10^6$ kg/ms^2 and $\mu_s = 2 \cdot 10^6$ kg/ms^2 as well as nonlinear kinematics are applied. The domain is discretized by 2×36 through $128 \times 2,304$ quadrangles. The test is calculated by the standard FEAP solver as well as multigrid implementations with up to 5 grid levels. Again, the CPU time for all implementations as well as the convergence factors for the multigrid implementations are compared. Also the deformation is monitored by the displacement of point $A = (0.01, 0.35)$ and compared with the mesh independent solution.

Figure 4.19 shows the deformed domain for 2,304 degrees of freedom. Figure 4.21 shows the convergence factors for the different multigrid approaches. These are much higher than for the previous test cases. This is in agreement with the fact that the correction scheme restricts the error, which is always smooth compared to the actual value and therefore can be reduced on coarse grids more effectively. The FAS restricts the actual variables χ, $\dot{\chi}$, and $\ddot{\chi}$. Due to the nonlinear problem, the FAS does not collapse to the correction scheme as for the previous test cases. This phenomenon can also be observed in the CPU time of the different solvers in figure 4.20. The runtime of the multigrid solvers is linear dependent on the resolution similar as for the linear test cases although its performance is inferior to the standard solver.

# Degrees of Freedom	x Displ of A [10^{-3} m]	y Displ of A [10^{-2} m]
144	-4.4881	-5.2327
576	-6.3168	-6.1996
2,304	-6.9691	-6.5103
9,216	-7.1550	-6.5967
36,864	-7.2056	-6.6202
147,456	-7.2195	-6.6267
589,824	-7.2235	-6.6286
independent	-7.2251	-6.6294
Turek	-7.187	-6.610
Heck	-7.011	-6.516

Table 4.4: Displacement of point $A = (0.01, 0.35)$ in the CSM 1 test case

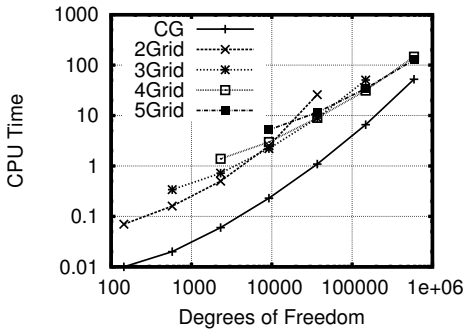


Figure 4.20: CPU time for the multigrid and the standard solver on the CSM 1 test case

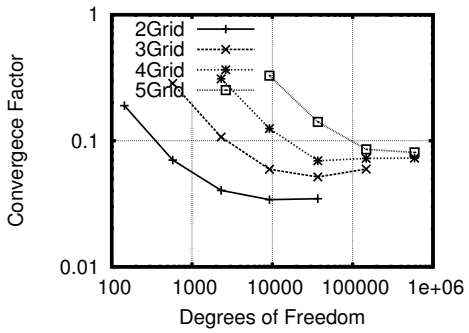


Figure 4.21: Convergence factor for the multigrid solver on the CSM 1 test case

In table 4.4 the displacement of point A is tabulated with the numbers of degrees of freedom. The test case has a unique solution, which is better approximated with increasing resolution. Thus, a grid independent solution is computed by Richardson extrapolation. Also the solutions computed by Turek and Hron [69] and Heck [29] are given for comparison. The grid independent solution is 0.53% off Tureks solution. The finest grid solution with about half a million degrees of freedom is 0.02% off the independent solution and the next coarser solution with about 150k degrees of freedom is 0.07% off. Even the next coarser solution with less than 37k degrees of freedom is still only 0.27% off the independent solution.
In conclusion, the FAS implementation in FEAP computes the correct displacement for nonlinear problems. It needs the expected CPU time in linear dependence on the degrees of freedom, although for the presented two dimensional problem it is outperformed by the standard solver. Also the employed grid shows good results, as the three finest discretizations show an error of less than 1%.

4.2.3 Restriction & Prolongation

Consideration

As stated by Hackbusch [28] first order interpolation for restriction and prolongation is sufficient since only second order pdes are present. In forethought of the coupling, interpolation is implemented analog to FASTEST. In the restriction of the displacements an injection is used. For the restriction of residuals and prolongation of displacements a linear interpolation in between neighboring points is applied.

Implementation

As the FEM code uses node centered values and the coarse grid nodes are a subset of the fine grid nodes, the displacement can be interpolated by simple injection. This is realized automatically when using the *TIE* command.
The residual associated with a fine grid node which is does not correspond to a coarse grid node must be split up between neighboring coarse grid nodes. This is accomplished by linear interpolation.
During the initialization of a multigrid computation an interpolation matrix is created. Since this is a sparse matrix, it is stored in the so called Compressed Row Storage (CRS) format. Thus the restriction as well as the prolongation achieves a run-time of $\mathcal{O}$(degrees of freedom).

Testing

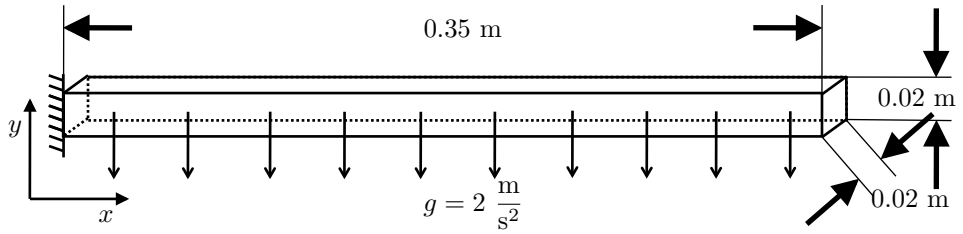


Figure 4.22: Initial conditions for the 3D CSM 1 and the 3D CSM 3 test case

In the next test case the implementation of the interpolation operators is tested for three dimensional problems. This test case is a 3D adaption of the CSM 1 test case used above. The computational domain is a 3D beam $[0.0, 0.35]\ \mathrm{m} \times [0.0, 0.02]\ \mathrm{m} \times [0.0, 0.02]\ \mathrm{m}$, which is discretized by first order brick elements with enhanced formulation. A body force of 2 $\mathrm{m/s}^2$ in negative y direction is applied to the entire structure. The same material, strain-stress relationship, and Lamé constants as in the latter test case are used. Figure 4.22 shows a sketch of the initial domain and applied loads of this test case. The domain is discretized into $2 \times 36 \times 2$ through $16 \times 288 \times 16$ linear brick elements. Once again, CPU time, displacement of point $A = (0.35, 0.01, 0.01)$, and the convergence factors are used for comparison. The test case is calculated by the 2 to 4 stage multigrid solvers and the standard FEAP solver.

In figure 4.23 the different solver implementations are compared regarding CPU time. The standard solver performs well for small discretizations, but complies with its exponential scaling property as the number of grid points rise. The 2 grid scheme performs better but still does not really show the linear dependence. Regarding a multigrid solver using all available grid

# Degrees of Freedom	x Displ A [10^{-3} m]	y Displ A [10^{-2} m]
432	-7.1504	-6.5890
3,456	-7.1860	-6.6086
27,648	-7.2079	-6.6203
221,184	-7.2178	-6.6256
independent	-7.2260	-6.6300

Table 4.5: Displacement of point $A = (0.35, 0.01, 0.01)$ in the 3D CSM 1 test case

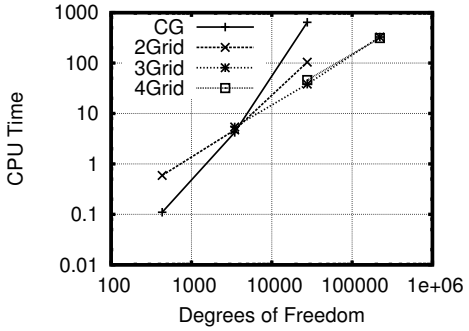


Figure 4.23: CPU time for the multigrid and the standard solver on the 3D CSM 1 test case

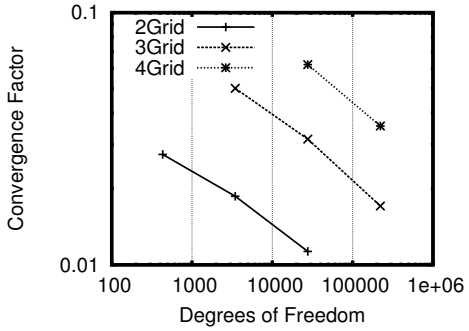


Figure 4.24: Convergence factors for the multigrid solver on the 3D CSM 1 test case

levels, i.e. the maximum of coarser grid levels available for the current resolution, the linear dependence can be observed. Thus, the performance of the multigrid implementation for a 3D test case is demonstrated.
Here, the great potential of multigrid solvers can be seen. As for 3 dimensional test cases the coarsening of a problem reduces the number of unknowns by a factor of around 8 instead of around 4 for 2D problems. Furthermore, comparing the convergence factors in figure 4.24 to the two dimensional case in table 4.21 improvement in the coarse grid correction can be seen.
Comparing the CPU times in figure 4.23 to the simple boundary condition test cases above, the multigrid already outperforms the standard solver for a discretization with less than 10,000 degrees of freedom. In figures 4.10 and 4.11 for the Dirichlet and von Neumann boundary condition test case this phenomenon occurs beyond 100,000 degrees of freedom. Table 4.5 shows the x and y displacement of point A for different discretizations and the mesh independent solution from the Richardson extrapolation. The finest grid solution is 0.11% off the independent one. The second finest solution is still 0.25% off. And the third finest solution implies an error of 0.55%.
In conclusion, the fast superior performance of the multigrid implementation and it's linear dependent run-time indicate the correct implementation

of 3 the restriction and prolongation operator in 3 dimensions.
Note that the multigrid solvers are able to solve finer discretizations, whereas the standard solver runs out of memory (4GB). Thus, the full potential of the multigrid algorithm in comparison to the standard solver cannot even be shown.

4.2.4 Transient Problems

Consideration

The unknown variable in the FAS now consists not only of χ, but also of $\dot{\chi}$ and $\ddot{\chi}$. The transient variables velocity and acceleration have to be restricted, prolongated and corrected according to the displacement. Also the mass and damping matrix have to be treated by the FAS according to the stiffness.

Implementation

Although the mass and damping part of equation (3.6) are linear, the FAS implementation is used. This is done for implementation reasons, as the mass, damping and stiffness are not stored explicitly, but the iteration matrix for Newton's method is computed directly. The implementation follows the findings of the previous sections, as the theory derived there included the transient case. The mass and damping matrix are restricted and prolongated by linear interpolation. The velocity and acceleration are restricted using injection and their correction is prolongated by linear interpolation.

Testing

The test case for transient solutions is the 3D adaption of CSM 3 from Turek and Hron [69]. The initial domain is $[0.0, 0.02]$ m $\times$ $[0.0, 0.35]$ m $\times$ $[0.0, 0.02]$ m, the Saint Venant Kirchhoff material law as well as a nonlinear kinematics is used. The material parameters are described by $\lambda_s = 8 \cdot 10^6$ kg/ms^2 and $\mu_s = 2 \cdot 10^6$ kg/ms^2. A body force of $2 \cdot 10^3$ N/m^3 in negative y direction is applied to the entire structure. The material density is $\rho_s = 1 \cdot 10^{-3}$ kg/m^3. The initial domain and loads is shown in figure 4.22. The domain is discretized by a successively increasing number of 8 node brick elements. For the test two seconds of physical time were calculated, the second order Newmark beta method with standard parameters $\beta = 0.25$ and $\gamma = 0.5$ is used for the time discretization.
Three different discretizations were used. A coarse grid with $2 \times 36 \times 2$ elements, i.e. 432 degrees of freedom, a middle one with $4 \times 72 \times 4$ elements, i.e. 3,456 degrees of freedom and a fine one with $8 \times 144 \times 4$ elements, i.e. 27,648 degrees of freedom. To be consistent with Turek and Hron's paper

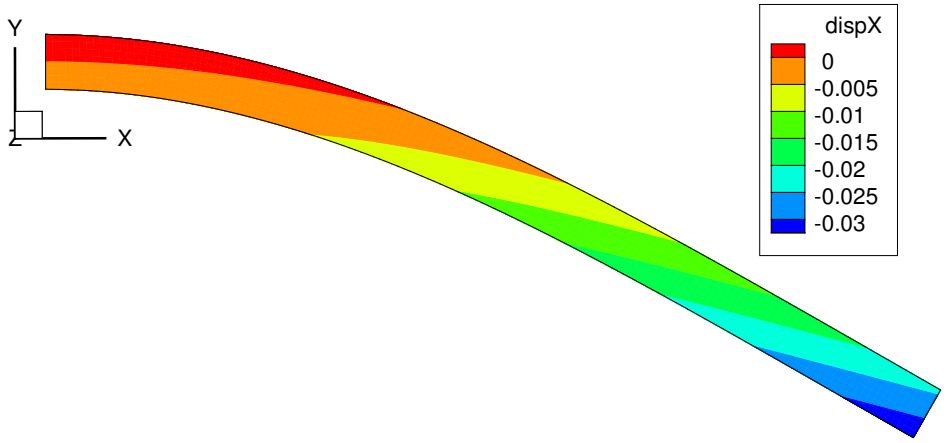


Figure 4.25: Deformed state of the 3D CSM 3 test case (x displacement shown)

three different time step sizes, $\Delta t = 0.02$, 0.01, and 0.005 are used for comparison. All test cases were run with the multigrid implementation with up to 4 levels as well as the standard FEAP solver. The mean as well as maximum displacement of point $A = (0.35, 0.01, 0.01)$ as well as the average CPU time per time step are monitored.

On the basis of the finest discretization a modal analysis was carried

Grid	x Disp. [10^{-2} m] (Freq. [1/s])	y Disp. [10^{-2} m] (Freq. [1/s])
coarse	-1.44361 ± 1.44359 [1.08696]	-6.36306 ± 6.51894 [1.11111]
middle	-1.45056 ± 1.45054 [1.08696]	-6.38658 ± 6.53242 [1.11111]
fine	-1.45482 ± 1.45488 [1.08696]	-6.40057 ± 6.54043 [1.11111]

Table 4.6: Displacements of point $A = (0.35, 0.01, 0.01)$ for the 3D CSM 3 test case with $\Delta t = 0.02$ s

Grid	x Disp. [10^{-2} m] (Freq. [1/s])	y Disp. [10^{-2} m] (Freq. [1/s])
coarse	-1.45558 ± 1.45572 [1.08696]	-6.44451 ± 6.48449 [1.08696]
middle	-1.46211 ± 1.46229 [1.0989]	-6.45029 ± 6.51271 [1.08696]
fine	-1.46676 ± 1.46694 [1.08696]	-6.45673 ± 6.52727 [1.0989]

Table 4.7: Displacement of point $A = (0.35, 0.01, 0.01)$ for the 3D CSM 3 test case with $\Delta t = 0.01$ s

out. The first 3 Eigenfrequencies are at 1.07 Hz, 6.61 Hz, and 18.0 Hz. This identifies the first three cycle lengths to be 0.93 s, 0.15 s, and $0.0\bar{5}$ s.
Figure 4.25 shows the 3D CSM three domain at its maximal deformed state.

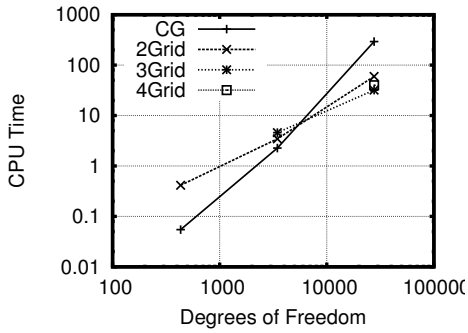


Figure 4.26: Average CPU time per time step for the multigrid and standard solver on the 3D CSM 3 test case for $\Delta t = 0.02$ s

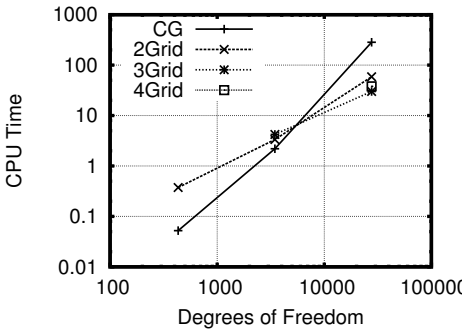


Figure 4.27: Average CPU time per time step for the multigrid and standard solver on the 3D CSM 3 test case for $\Delta t = 0.01$ s

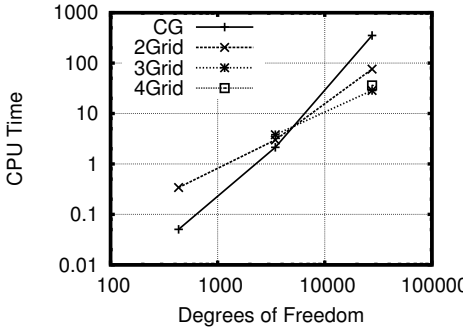


Figure 4.28: Average CPU time per time step for the multigrid and standard solver on the 3D CSM 3 test case for $\Delta t = 0.005$ s

Tables 4.6, 4.7, and 4.8 show the displacement of point A for varying time steps and resolutions. The grid independent solution was computed from the computations with the smallest time step. Table 4.9 shows in addition the solution of Turek and Hron [69] and Heck [29].
Looking at the displacement in x direction of the fine grid, one can see although the actual values for the maximal displacement is only 0.72% off the independent solution, the time step is not suitable to display the correct frequency. This results in a frequency which is 1.67% off the independent solution.
The fine grid solution for $\Delta t = 0.01$ s results in an amplitude error in x direction of 0.39%, an amplitude error of 0.52% in y direction, and a frequency error of 0.55%. For the second finest grid these errors are 0.55% in x, 0.83% in y, and 0.54% in frequency.
The finest time step of $\Delta t = 0.005$ s and the fine grid result in an amplitude error of 0.33% in x direction and 0.37% in y direction. The middle grid

Grid	x Disp. [10^{-2} m] (Freq. [1/s])	y Disp. [10^{-2} m] (Freq. [1/s])
coarse	-1.45677 ± 1.45683 [1.0989]	-6.46513 ± 6.46387 [1.0989]
middle	-1.46458 ± 1.46462 [1.0929]	-6.48513 ± 6.48387 [1.0929]
fine	-1.46893 ± 1.46897 [1.0929]	-6.49381 ± 6.49819 [1.0929]

Table 4.8: Displacement of point $A = (0.35, 0.01, 0.01)$ for the 3D CSM 3 test case with $\Delta t = 0.005$ s

Solution	x Disp. [10^{-2} m] (Freq. [1/s])	y Disp. [10^{-2} m] (Freq. [1/s])
independent	-1.4744 ± 1.4745 [1.0929]	-6.5005 ± 6.5343 [1.0929]
Turek	-1.4305 ± 1.4305 [1.0995]	-6.3607 ± 6.5160 [1.0995]
Heck	-1.421 ± 1.421 [1.098]	-6.309 ± 6.161 [1.098]

Table 4.9: Displacement of point $A = (0.35, 0.01, 0.01)$ for the 3D CSM 3 test case

results in an amplitude error of 0.51% in x direction and 0.67% in y. For both grids the frequency error is below 0.01% and thus not visible in the given results. The coarse grid results in an error of 1.20% in amplitude and 0.55% in frequency.
Figures 4.26, 4.27, and 4.28 show the average CPU time for one time step for the multigrid solvers and the standard solver. All three time step show similar results, and the multigrid implementations outruns the standard implementation for increasing resolution. The similar behaviour of the multigrid solver to the standard solver when altering the time step size indicates the correct implementation of the transient FAS.
As this test case serves as a basis for the FSI computations in chapter 6, note that all resolutions on the finest time step show an error of less than 1.2% in tip displacement.

4.2.5 Coupling

Consideration

The MG CPL implementation is constructed by applying the FAS to an implicit partitioned coupled problem as described in section 4.1.2. The pre- and post-smoothing in the MG CPL implementation consist of an analog to the implicit partitioned implementation. Solely the individual equation solvers are modified. After pre-smoothing, the FAS coarse grid representation of the coupled system is solved. Then, the coarse grid correction (3.41) is applied to velocity, pressure and displacement individually. Afterwards a few post-smoothing steps are calculated similar to the pre-smoothing.
The discrete fine grid coupled system as outlined in section 4.1.2 containing

all applied boundary conditions can be stated as:

$$A^h\left(v^h(\chi^h), p^h(\chi^h)\right) = S^h\left(v^h(\chi^h), p^h(\chi^h)\right) \tag{4.18}$$

$$R^h_\chi(\chi^h) = -R^h\left(\chi^h, v^h, p^h\right) \tag{4.19}$$

in which only the direct dependencies are denoted. Equation (4.18) consists of the discrete momentum and pressure correction equation as well as the grid deformation on the fluid domain and the fluid FSI boundary condition. Equation (4.19) consists of the residual equation arising from the Newton iteration applied to the discrete momentum equation in the structure domain as well as the structure FSI boundary condition. R_χ denotes the derivative of R with respect to χ.

The resulting FAS coarse grid equations for the coupling problem then read as follows:

$$\begin{aligned} & A^{2h}\left(v^{2h}(\chi^{2h}), p^{2h}(\chi^{2h})\right) \\ = \;& S^{2h}\left(v^{2h}(\chi^{2h}), p^{2h}(\chi^{2h})\right) \\ - \;& \left[S^{2h}\left(v_0^{2h}(\chi_0^{2h}), p_0^{2h}(\chi_0^{2h})\right) - A^{2h}\left(v_0^{2h}(\chi_0^{2h}), p_0^{2h}(\chi_0^{2h})\right)\right] \\ + \;& I_h^{2h}\left[S^h\left(v_0^h(\chi_0^h), p_0^h(\chi_0^h)\right) - A^h\left(v_0^h(\chi_0^h), p_0^h(\chi_0^h)\right)\right] \end{aligned} \tag{4.20}$$

$$\begin{aligned} R^{2h}_\chi(\chi^{2h}) \;=\; & -R^{2h}\left(\chi^{2h}, v^{2h}, p^{2h}\right) + R^{2h}\left(\chi_0^{2h}, v_0^{2h}, p_0^{2h}\right) \\ & -I_h^{2h} R^h\left(\chi_0^h, v_0^h, p_0^h\right) \end{aligned} \tag{4.21}$$

From the theory of multigrid it follows that all boundary conditions have to be applied to the fine grid problem before restriction and the coarse grid representation of these boundary conditions has to be applied to the coarse grid problem. In FSI problems the boundary conditions at the FSI interface for both the fluid and the structure code is permanently changing. Hence, special care has to be taken at the Dirichlet boundary, i.e. the velocity at the interface acting on the fluid, as the influence of this boundary spreads into the problem domain during prolongation.
One extra exchange iteration is introduced on the coarse grid, in order to assemble the above system. It is carried out directly after restriction, its purpose will become clear in the following.

Implementation in FEAP

The extension of the FEAP multigrid implementation to the MG CPL is explained using equation (4.21). The left hand side is simply the tangent matrix as for the non coupled case. The first item on the right hand side

is dependent on the solution of the most recent coarse grid coupled solve. Thus, in the first iteration it is set equal to the restricted fine grid solution. The second item on the right hand side of (4.21) needs special attention. The R^{2h} vector is dependent on the interpolated fine grid velocity as well as the interpolated pressure. In the fluid code the velocity is interpolated and the resulting shear force based on $v_0^{2h} = I_h^{2h} v_0^h$ is sent to the structure code. This is done during the one extra coupling step. The initial coarse grid pressure is set to zero, as the fluid solver only operates on the gradient and the correction of the pressure (for more information see [41]). This numerical simplification in the fluid code does not affect the course of interaction. As R^{2h} is linear in p and the starting point of p^{2h} in the first right hand side term of (4.21) is set to p_0^{2h}, it is possible to set $p_0^{2h} = I_h^{2h} p_0^h := 0$ and let p^{2h} be only the correction to the fine grid pressure. The displacement χ_0^{2h} is set to the interpolated fine grid displacement prior to any coarse grid exchange as for the non coupled case.
The remaining third residual on the right hand side is also calculated as for the non coupled case, originating from the last coupled fine grid solve.
Additionally, in the extra coupling step the interpolated fine grid displacement is send to the fluid code as well.
Considering the memory consumption, as the coarse grid points are actually stored as a subset of the fine grid points, extra memory has to be allocated to store the doubly allocated arrays holding fine grid values which are not supposed to be altered during the coarse grid solution.

Implementation in FASTEST

The extension of the implicit partitioned coupling to MG CPL of FASTEST is explained with the help of equation (4.20). The left hand side and the first item of the right hand side of (4.20) are dependent on v^{2h} and p^{2h} which have to be equal to $I_h^{2h} v^h\left(\chi^h\right)$ and $I_h^{2h} p^h\left(\chi^h\right)$ at convergence. Thus, the coarse grid displacement must be treated as a correction to the fine grid displacement. This is realized by setting the initial coarse grid displacement to the interpolated fine grid displacement $\chi_f^{2h} := I_h^{2h} \chi_{f,0}^h$ and moving the coarse grid accordingly prior to the first coarse grid FSI iteration. Simultaneously, using the extra exchange, the first coarse grid displacement $\chi_{s,0}^{2h}$, which is received from the structure, is stored in order that during all subsequent coarse grid FSI iterations the grid is only moved by a correction to the interpolated fine grid displacement. Thus, the actual displacement in the following coarse grid FSI iterations results in

$$\chi_f^{2h} := \chi_s^{2h} - \chi_{s,0}^{2h} + I_h^{2h} \chi_{f,0}^h. \tag{4.22}$$

The second item on the right hand side of (4.20) as well as the system matrix on the right hand side is built dependent on the interpolated fine

grid velocity and pressure, prior to any exchange.
The last two items are calculated on the fine grid based on the last coupling iteration prior to restriction.
The Dirichlet boundary conditions for the fluid velocity must be applied again on the coarse grid directly after restriction, and dependent on the interpolated fine grid displacement $I_h^{2h}\chi_{f,0}^h$, as v^{2h} is set to $I_h^{2h}v_0^h\left(\chi_{f,0}^h\right)$ as well in restriction. From this follows, that before prolongation the current Dirichlet boundary condition, depending on the displacement from equation (4.22), must be applied again in order to be consistent with the coarse grid velocity correction.
The implementation of the Dirichlet boundary condition at the FSI interface (2.25) is consistent with the underlying time integration scheme. Instead of the velocity, the displacement χ^h is interpolated and transferred from the structure and the boundary velocity in time step n is then computed according to the BDF 2 as:

$$v_n^h = \frac{3\chi_n^h - 4\chi_{n-1}^h + \chi_{n-2}^h}{2\Delta t} \tag{4.23}$$

in which χ is taken at the cell centers, not the cell corners. In order for its coarse grid counterpart (4.6) to be consistent with the time stepping scheme, the coarse grid displacement has to be calculated as in (4.22) and again, χ at the cell center. Note, that χ^{2h} from previous time steps is stored at FSI convergence, thus is consistent with the fine grid displacement. By injecting the latter equation into the $2h$ version of (4.23) it can be shown, that the choice of coarse grid displacement from corrected coarse grid velocity is valid.
The restriction of displacements is realized by simple injection, the residual and the boundary flux for every coarse grid cell is restricted by summation over the corresponding fine grid cells.
Note, that every iteration ends with a structure solve, thus the boundary conditions for the fluid are usually calculated at the beginning. After the last FSI iteration on the fine grid the velocity at the FSI interface has to be calculated once again in order to have a complete set of feasible boundary conditions.

Implementation in the MpCCI Interface

A unique grid to grid mapping has to be used when coupling on multiple non-matching grids with one coupling tool. Therefore, for every grid used in the computation a separate communicator has to be introduced into MpCCI.
The node and element numbering in MpCCI is different to the numbering in FASTEST and FEAP, as it has only knowledge about the interface nodes / elements and its internal elements are situated at the FSI boundary, therefore of lower dimension. Special care has to be taken in FEAP when using MG CPL, as after the *TIE* command (eliminating nodes defined twice)

coarse grid points are virtually non-existent, and thus a mapping between the fine and coarse grid point has to be applied.

Testing

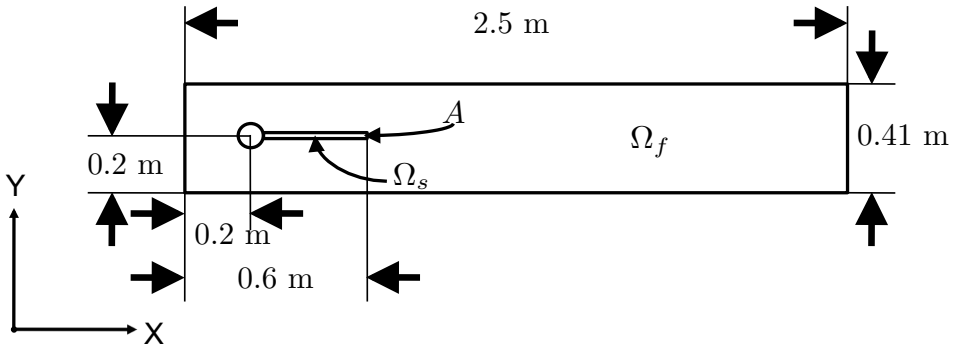


Figure 4.29: Schematic sketch of the fluid and structure domain of FSI 1 test case

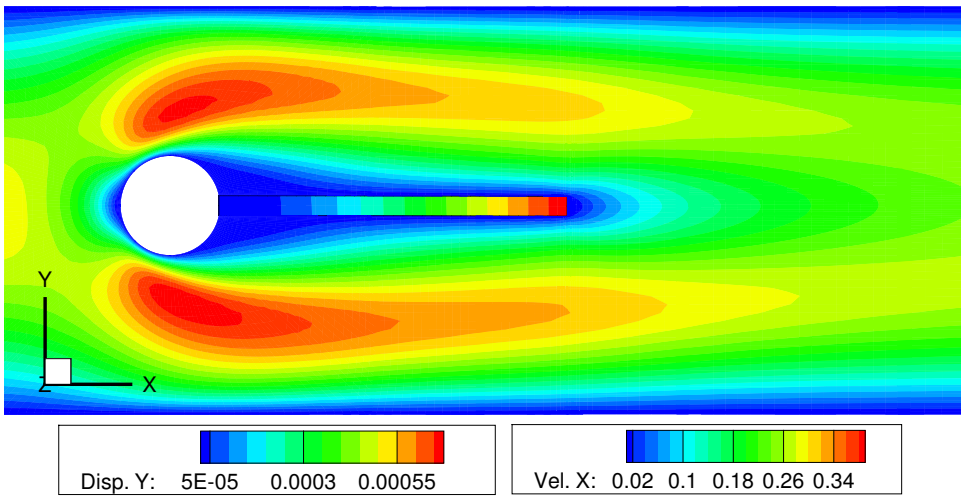


Figure 4.30: Final state of the FSI 1 test case (fluid x velocity and structure y displacement shown)

The next test case is employed to verify the coupled FAS implementation in the MG CPL. For this purpose the FSI 1 test case from Turek and Hron [69] is used. The test case consists of a solid object submerged in a channel flow. The fluid properties are described by an incompressible Newtonian fluid with a dynamic viscosity of $\mu_f = 1$ kg/ms and a parabolic inflow

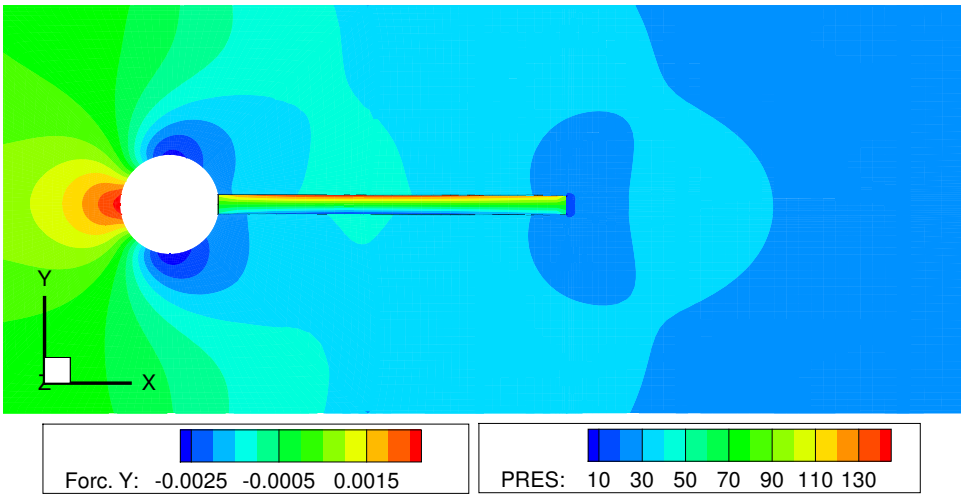


Figure 4.31: Final state of the FSI 1 test case (fluid pressure and structure y loads shown)

profile in positive x direction

$$v_f|_{\text{Inlet}} = 1.5\ \bar{v} \left(\frac{4.0}{0.1681\ \text{m}} \frac{y}{\text{m}} (0.41 - \frac{y}{\text{m}}), 0 \right)^T , \tag{4.24}$$

with average inflow velocity $\bar{v} = 0.2$ m/s. This results in a Reynolds number of Re $= 20$, thus a laminar fluid state is considered. At the lower wall ($y = 0$ m) and upper wall ($y = 0.41$ m) homogeneous Dirichlet boundary conditions for the fluid are active. The outflow ($x = 2.5$ m) is characterized by a homogeneous von Neumann boundary condition. The fluid domain is initialized with zero velocity and pressure. The structure part consists of the cylinder, which is rigid and the attached elastic bar. In the elastic bar the St. Venant Kirchhoff material law is used, with density $\rho = 1 \cdot 10^3$ kg/m^3, and Lamé constants $\lambda_s = 8 \cdot 10^6$ kg/ms^2 and $\mu_s = 2 \cdot 10^6$ kg/ms^2. Furthermore nonlinear kinematics are applied. The boundary conditions of the bar are homogeneous Dirichlet at the cylinder and von Neumann FSI coupling conditions on the remaining sides.
The FSI interface corresponds to the entire surface of the structure. As the cylinder is rigid, a homogeneous Dirichlet boundary condition is set there in the fluid solver. As this is a two dimensional test case but the fluid solver is written for three dimensions, the velocity in the direction of the third dimension is initialized with zero and skipped in the solution. In the structure domain homogeneous Dirichlet conditions in the third dimension are set at the boundaries.
In this test case the tip displacement of point A located initially at $(0.2, 0.6)$ as well as the drag and lift forces acting on the structure are observed. Additionally, for the comparison of the performance of the MG CPL, the fluid

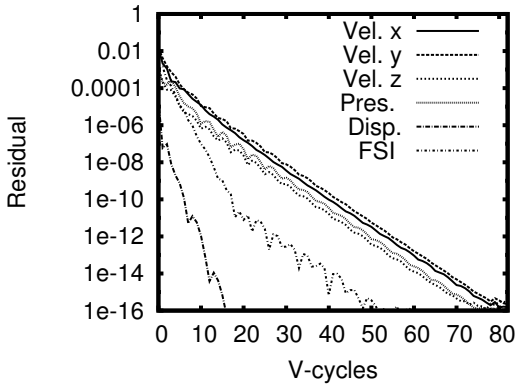


Figure 4.32: Development of the fluid, structure and FSI residual on the FSI 1 test case

fine grid iterations and the CPU time are monitored. The test case is computed using two implementations, namely the implicit partitioned coupling approach with a multigrid fluid solver and the standard FEAP solver and the MG CPL approach with two grid levels. The test case is discretized into three different grids for the fluid as well as the structure domain. The fluid domain is discretized into 60,800, 234,200, and 972,800 volumes and the structure domain into 144, 1,152, and 9,216 elements, respectively.

Solution	x Disp. [10^{-5} m]	y Disp. [10^{-4} m]
coarse	2.2994	6.9527
middle	2.2738	7.8346
fine	2.2691	8.0920
independent	2.2680	8.1981
Turek	2.27	8.29

Table 4.10: displacement of point $A = (0.6, 0.2)$ in the FSI 1 test case

Solution	Lift [N]	y Drag [N]
coarse	0.80976	14.2907
middle	0.77752	14.2917
fine	0.76755	14.2874
independent	0.76309	14.2910
Turek	0.7638	14.295

Table 4.11: Drag and lift forces acting on the structure of the FSI 1 test case

Figure 4.30 and 4.31 show the final state of the FSI 1 test case. Figure 4.30 shows the x component of the fluid velocity as well as the y component of

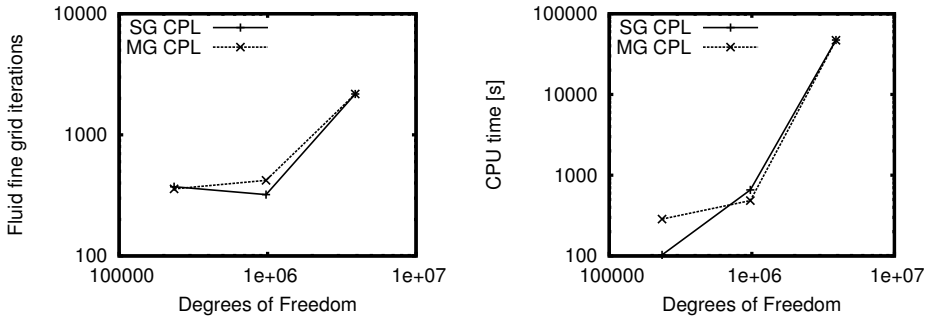


Figure 4.33: Fluid fine grid iterations and overall CPU time for the implicit partitioned coupling approach (SG) and MG CPL on the FSI 1 test case

the structure displacement. Figure 4.31 shows the fluid pressure distribution and the forces acting on the structure in y direction. Table 4.10 shows the displacement for the different resolutions as well as the grid independent solution of the Richardson extrapolation and the solution from Turek. Compared to the independent solution the fine grid introduces a maximum error of 1.29 % in the tip displacement and the middle grid 4.43 %. In table 4.11 the forces acting on the structure are summarized for the different discretizations. Compared to the independent solution, the fine discretization introduces a maximum error of 0.58% and the middle grid of 1.89%.

According to Ferziger and Peric [23] a common error that can arise in implementing a multigrid solver is the wrong treatment of the boundary conditions. This is reflected in the fine grid solution being unable to converge its residual below a certain point. Thus, a correct multigrid implementation must be able to run to machine accuracy. This is a crucial requirement for the MG CPL implementation. As the two separate solvers for fluid and structure are coupled by boundary conditions and the value of these conditions are changing on fine and coarse levels, a correct implementation has to be ensured. Figure 4.32 shows the lapse of the residuals for the fluid velocity, the pressure, the structure displacement and the coupled FSI problem. They all reach 10^{-16}, which is machine accuracy for double precision calculations. This shows the correct implementation of the FSI boundary conditions in the MG CPL.

Figure 4.33 shows the number of fluid fine grid iterations and the overall CPU time per time step for three different discretizations of the FSI 1 test case plotted over the number of degrees of freedom. The MG CPL shows almost a similar behaviour as the implicit partitioned approach concerning fluid fine grid iterations and overall CPU time. This is due to weak interacting nature of this test case. As the majority of computational effort is used

to solve the fluid problem, which is not accelerated by the MG CPL, the advantage of the new FSI implementation is not visible. Instead, in both cases the performance of the fluid solver is depicted.
Tests involving more interaction and thus showing the effect of the MG CPL are shown in chapter 6.

Chapter 5

Application of Extrapolation to Fluid-Structure Interaction

A second strategy to accelerate implicit partitioned FSI computations is the prediction of the values in question by extrapolation. Extrapolation is a well known technique to speed up many kinds of unsteady processes, as it can provide a sophisticated initial guess to the underlying solver, ideally predicting the current solution accurate to the truncation error of the underlying interpolation scheme. In implicit partitioned FSI computations it is mainly used in two areas. First, the prediction of a starting point for the current FSI iteration by extrapolating the results of the last few iterations within a time step or, second, the prediction of a starting point for the current time step by extrapolating the converged FSI solutions over the last time steps. The former include, among others, adaptive underrelaxation of Aitken type as in Vierendeels et al. [71], Küttler [38, 37], Wall [74], Yigit [79], or Schäfer et al. [56]. This also includes reduced order modeling as in Vierendeels et al. [71], steepest descent methods, or vector extrapolation as in Küttler [37, 38]. The second point of application includes polynomial extrapolation as in e.g. Breuer and Münsch [9, 10] or Schäfer et al. [56], or prediction based on numerical integration as in Piperno [46].
In the application to FSI of the above mentioned prediction methods attention has been limited only to the structure side, namely the extrapolation of displacements. In this work the extrapolation is applied to the fluid variables pressure and velocity and a comparison to the structure extrapolation is given.
In this chapter the first four Lagrangian interpolation functions as described in section 3.4 are used. For a constant time step size and an arbitrary time dependent variable ϕ, with ϕ_i^n is the value of ϕ at iteration i and time step

n, the considered polynomials are

$$\begin{aligned}
\text{0th order:} \quad & \phi_1^n = \phi_*^{n-1} && (5.1)\\
\text{1st order:} \quad & \phi_1^n = 2\phi_*^{n-1} - \phi_*^{n-2} && (5.2)\\
\text{2nd order:} \quad & \phi_1^n = 3\phi_*^{n-1} - 3\phi_*^{n-2} + \phi_*^{n-3} && (5.3)\\
\text{3rd order:} \quad & \phi_1^n = 4\phi_*^{n-1} - 6\phi_*^{n-2} + 4\phi_*^{n-3} - \phi_*^{n-4}. && (5.4)
\end{aligned}$$

with ϕ being either displacement, velocity, or pressure. The asterisk denotes the converged solution at the end of a time step. A comparison of the different orders of extrapolation is given in section 6.2.2. Note that for higher order extrapolation at the beginning of the coupled computation, the order of extrapolation is decreased until a sufficient number of old time steps is available.
In the following the application of polynomial extrapolation to FSI is described, while using the acceleration for both structure and fluid variables. First, the implementation of displacement extrapolation in the implicit partitioned FSI framework is described. Afterwards, the implementation of force extrapolation is described.
The findings in this chapter result from joint work with Streitenberger [62].

5.1 Implementation of Displacement Extrapolation

The displacement extrapolation affects the displacement exchange in the coupled computation, i.e. the first part of equation (2.25). The velocity boundary condition for the fluid sub-problem is then derived from the displacement of the FSI boundary as described in section 4.2.5.
The structural displacement χ on the FSI boundary is altered in the first FSI iteration of every time step. Since the deformation of the fluid grid is dependent on the FSI boundary deformation, the extrapolation is carried out prior to the fluid grid deformation. Recalling the implicit partitioned coupling in terms of figure 4.3, the course of this algorithm in time step n is altered at the beginning of the time step. The resulting algorithm is depicted in figure 5.1. In contrast to the implicit partitioned algorithm, the fluid solution, derivation of force on the boundary, and solution of the structure problem are skipped in first iteration of time step n. Instead, the first structure solution $(\chi)_1^n$ is extrapolated by one of the equations (5.1) to (5.4). In order to fully exploit the advantages of the extrapolation no underrelaxation is applied to the extrapolated displacements in the first iteration, as this would damp the introduced correction. The following iterations in time step n are carried out according to the implicit partitioned coupling scheme.

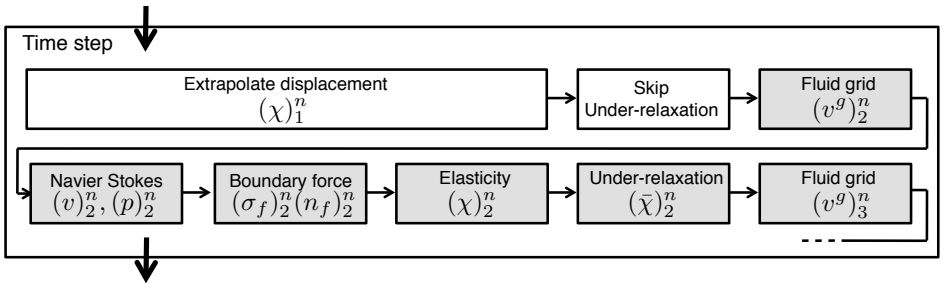


Figure 5.1: Detail of implicit partitoned approach with displacement extrapolation

5.2 Implementation of Force Extrapolation

The extrapolation of fluid variables is implemented to the better part similar to the latter approach. Here, the extrapolation operates at the force exchange of the FSI coupling.
When applying the extrapolation to the fluid variables, the choice falls on restricting the extrapolation domain to the FSI boundary. In the case that the fluid problem is solved to convergence in every FSI iteration, extrapolating the values in the entire fluid domain would improve the starting point for the next fluid iteration in addition to the FSI acceleration. However, it would only accelerate one single fluid iteration, as the solution of the coupling iteration would remain unchanged. This minor benefit is outweighed by the additional computation and storage cost of extrapolating on a higher dimensional domain. The case of running only for a constant number of iterations in the fluid solver before coupling is covered in the work of Streitenberger [62]. It is shown that this choice is inferior to extrapolating only on the FSI boundary.
As the extrapolation domain is restrcited to the FSI boundary, the extrapolated value has to be reconsidered. Instead of extrapolating - and storing - the three velocity components and the pressure, simply the force scting on the boundary can be used. No alterations of the extrapolation functions is needed, as the force is linear dependent on velocity components and the pressure.
The algorithm force extrapolation in FSI can be described as a alteration of the implicit partitioned approach described in figure 4.3. The first iteration in every time step is depicted in figure 5.2. Here, the first FSI iteration omits the fluid solution and the derivation of force on the FSI boundary. In this case, the force $(\sigma)_1^n (n_f)_1^n$ is extrapolated by one of the equations (5.1) to (5.4). Once again no underrelaxation is applied in the first FSI iteration. The resulting displacement from the structure solver is used unchanged in the grid displacement routine. Again, the following iterations are identical

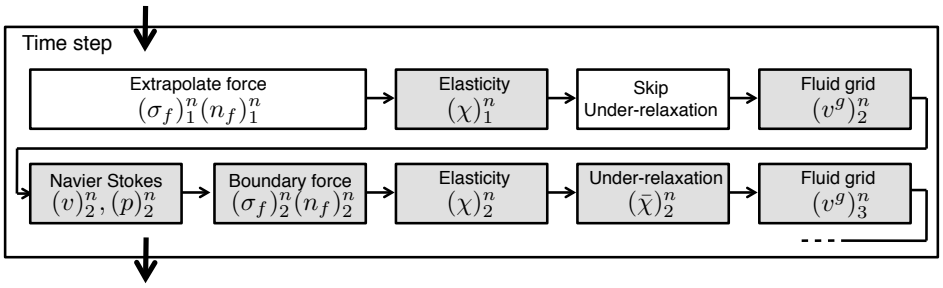


Figure 5.2: Detail of implicit partitioned approach with force extrapolation

to the non-extrapolated case. Compared to the displacement extrapolation this introduces an additional structure solve at the beginning of every time step. However, the gain in accuracy outweighs this additional solve in terms of CPU time needed for convergence.

Results and comparisons of the different extrapolation algorithms are shown in section 6.2.2 and in Sachs et al. [53].

Chapter 6

Test Cases

In this chapter the acceleration approaches for FSI coupling described in the last two chapters are applied to a set of test cases. Furthermore, a comparison of all presented methods among each other, their interaction, and the implicit partitioned coupling is conducted.
First the setup of the test cases is introduced, and an analysis for different grids is performed for the fluid and structure domain individually. This analysis is performed in order to assure that the grids used are the best compromise about accuracy and efficiency and that with even finer grids no notable deviations of the solutions occur. This ensures a valid basis for comparison. Then, a short study of the stability of the second order time stepping scheme is carried out. In this study the handling of non-physical pressure oscillation and sign switching around the structure domain in between time steps is analyzed. Then, the performance of the MG CPL approach as well as the extrapolation approaches in examined. And finally the MG CPL implementation is compared to the implicit partitioned coupling in terms of robustness against underrelaxation factors.

6.1 Preliminaries

6.1.1 Fluid-Structure Interaction Benchmark

A commonly used set of test cases for FSI is chosen to compare the different approaches described above. This set was developed within the Forschergruppe 493 of the Deutsche Forschungsgemeinschaft [69] and consists of a purely numerical benchmark. A great advantage of this benchmark is the modularity, as it consists of pure structure and fluid tests as well as coupled tests, and comparability with many other researchers (e.g. contributions to Bungartz and Schäfer [12]). As the investigation of the solvers on the individual domains without any coupled computation is part of the benchmark, the fluid and structure multigrid solvers as well as the used discretizations and settings can be tested prior to the coupled computation.

All parameters are set according to the paper of Turek and Hron [69]. The problem domain consists of a channel flow around a rigid cylinder with an elastic bar attached to it. The channel has the dimensions 2.5×0.41 m^2, with a slightly off symmetric positioned cylinder at $(0.2\text{ m}, 0.2\text{ m})$, radius 0.05 m and a bar attached to the cylinder of size 0.35×0.02 m^2.
The fluid and structure domain is equal to the domains used in section 4.2.5. Note, that the cylinder is rigid, thus not an active part of the structure domain. The boundary conditions for the fluid are: A parabolic inflow profile

$$v_f|_{\text{Inlet}} = 1.5\ \bar{v} \left(\frac{4.0}{0.1681} \frac{y}{\text{m}} (0.41 - \frac{y}{\text{m}}), 0 \right)^T, \tag{6.1}$$

with $\bar{v} = 2$ m/s at $x = 0$ m, a zero-gradient outlet condition at $x = 2.5$ m, and homogeneous Dirichlet boundary condition on all other surfaces. This includes the FSI boundary, which is the surface of the elastic beam. The elastic structure is clamped at the cylinder and the von Neumann coupling boundary condition is applied at the remaining sides. Initial conditions are a fully developed flow field with periodic separation, which is computed in the pure fluid test, and the undeformed structure without any additional loading at rest.
The fluid is set to be incompressible and Newtonian with density $\rho = 1 \cdot 10^3$ kg/m^3 and dynamic viscosity $\mu_f = 1$ kg/ms. This leads to a Reynolds number of Re = 200. The structure is characterized by the St. Venant Kirchhoff material model with density $\rho = 1 \cdot 10^3$ kg/m^3, and $\lambda_s = 8 \cdot 10^6$ kg/ms^2 and $\mu_s = 2 \cdot 10^6$ kg/ms^2 being the Lamé constants.
The expected behaviour of this setup is that the elastic bar starts oscillating with growing amplitude. After overshooting the average amplitude and reaching the maximum amplitude it will describe a periodic slope. For the comparison with other researchers the lift and drag force on the entire structure and the displacement of point A in figure 4.29 at the tip of the structure is monitored. The comparison starts when the oscillation reaches periodicity.

6.1.2 Fluid

In the first test case the fluid field is calculated with the structure remaining rigid. A survey including three different grids is conducted. The fluid domain is discretized by three different grids containing 60,800, 243,200, and 972,800 control volumes. As this test case is to be computed in two dimensions, but the employed fluid solver is written for three dimensional problems, a quasi two dimensional setup is achieved by setting periodic boundary conditions in the third dimension in the fluid and structure solver. Furthermore, the grid distortion in the fluid solver in the third dimension is set to 0. This only accounts for the break off error in the edge distortion of the individual blocks (see cubic spline interpolation in Schäfer et al. [56]), as

the received displacement is already zero in the third dimension. In order to compare the results with other investigations, the number of control volumes in two dimensions is crucial. These are 15,200, 60,800, and 243,200, respectively, as there are four layers of control volumes in the third direction. In preparation of the coupled computations, the quantity for comparison is the drag and lift acting on the structure. As a precaution to additional error the time step sizes for the fluid solver were chosen according to a Courant Friedrichs Lewy (CFL) number of strictly below 1. This results in a time step sizes of $3 \cdot 10^{-4}$ s, $1 \cdot 10^{-4}$ s, $5 \cdot 10^{-5}$ s, for the coarse, middle, and fine grid, respectively. Note, that the CFL number used here is a worst case approximation of the actual CFL number. This means the velocity vector entry with the largest absolute value as well as the smallest control volume elongation is used to compute the CFL number.
The simulations are carried out with the fluid code FASTEST using TBI discretization for the convective flux and the second order BDF for time integration. As convergence a criterion the summed up residual to be below $1 \cdot 10^{-7}$ is used.

Figure 6.1 shows the block structure of the discrete fluid domain. The

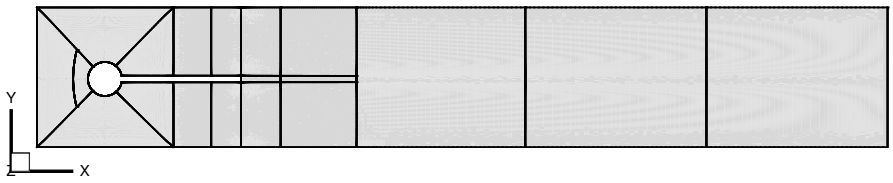


Figure 6.1: Block decomposition of the fluid domain for the FSI 3 test case

blocking has to satisfy different criteria. First, in order to provide for an efficient parallel implementation, all processors have to be equally loaded concerning the number of control volumes. In this blocking the average number of control volumes per processor divided by the maximum number of control volumes per processor is 98.96% for up to 8 processors. Second, a robust grid movement must be assured, as the grid is frequently moved in FSI iterations. Therefore, the additional cuts for the decomposition above and below the structure are perpendicular to the expected major grid movement. And third, the communication in between processors has to be minimal. As the fluid region is decomposed into only 19 blocks, this task can be easily done by hand.

Table 6.1 shows the calculated forces acting on the structure in amplitude and frequency for quantitative comparison and figure 6.2 shows the slope of the drag and lift forces over time for qualitative comparison. In table 6.1 the results of the three grid levels are shown as well as the solutions from Turek and Hron [69] and Heck [29]. Figure 6.2 shows, that the middle and the fine

Grid	Lift [N] (Freq. [1/s])	Drag [N] (Freq. [1/s])
coarse	-10.443 ± 50.52 (8.1900)	431.18 ± 0.2578 (8.2102)
middle	-16.448 ± 376.36 (4.3821)	437.41 ± 4.4215 (4.3802)
fine	-10.170 ± 421.81 (4.4053)	438.47 ± 5.4695 (4.4248)
Turek	-11.893 ± 437.81 (4.3956)	439.45 ± 5.6183 (4.3956)
Heck	-5.61 ± 436.1 (4.43)	440.99 ± 5.12 (4.43)

Table 6.1: Drag and lift forces acting on the structure of the unsteady pure fluid test case

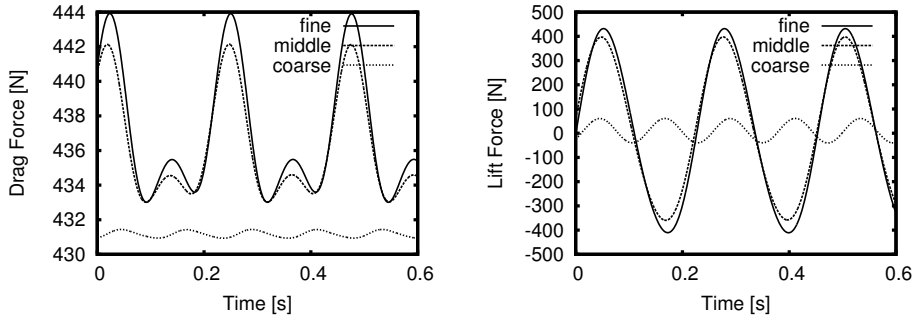


Figure 6.2: Drag and Lift of the coarse, middle, and fine grid for the unsteady pure fluid test case

grid display the same path of emerging force, only the coarsest setup cannot describe the forces correctly. Additionally the frequency of separation is predicted to be almost twice as high as for the fine solution. Comparing the absolute drag and lift values with the solution from Turek and Hron, the fine grid introduces an error of 3.94 % in amplitude and 0.22 % in frequency. The middle grid introduces an error of 12.65 % in amplitude and 0.31 % in frequency.

6.1.3 Structure

In the second test case only the structure is observed. The cylinder remains rigid and the elastic bar is modeled according to section 4.2.4. An analysis for the structure part has been performed in section 4.2.4. Figure 6.3 shows the displacements for the different grids.

6.1.4 Stability

One important aspect of the FSI test case is the stability of the time stepping scheme. The course of events prescribes to first compute a fully developed fluid flow with a rigid structure and then starting a FSI computation based on the developed flow. Particularly at the beginning of the FSI computation

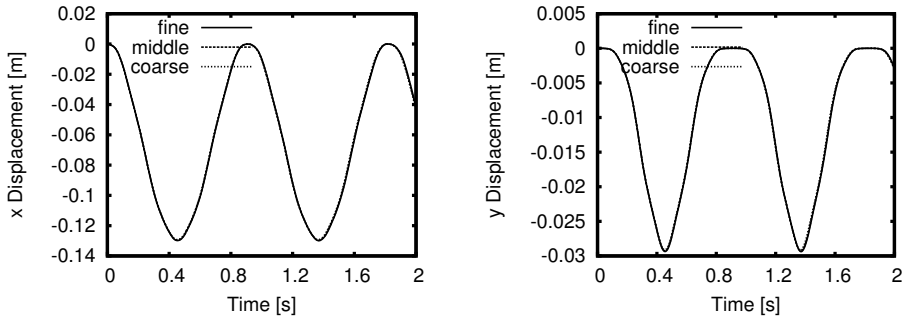


Figure 6.3: x and y displacement of the coarse, middle, and fine grid for the unsteady pure structure test case

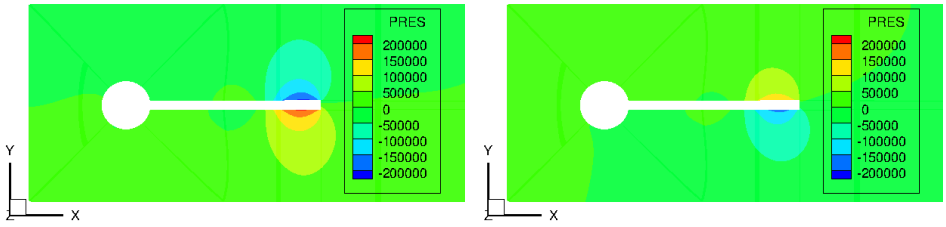


Figure 6.4: Pressure distribution in the fluid domain after 3 and 4 time steps respectively with Newmark $\beta = 0.25, \gamma = 0.5$ on the FSI 3 test case

this can lead to severe instabilities. Using second order BDF and Newmarks beta method with standard parameters for time discretization can result in an unstable time stepping scheme. This becomes apparent in an alternating pressure field above and below the moving structure. Figure 6.4 shows a detailed view of the pressure distribution of the fluid field after three and four time steps of coupled computation, starting from the fully developed fluid field and the structure at rest.

The maximum and minimum pressure differs by several orders and switch sign in between time steps. Also the displacement is showing oscillations which superimpose the physically reasonable displacement. This instability can be shown to be independent of the time step size and discretization.

A remedy to this situation is the introduction of numerical dissipation ($\beta > 0.25$) into the Newmark scheme, while giving up its second order accuracy. This stabilizes the system, hence reduces the initial error.

In order to compare the error reduction for different values of β and different time step sizes an artificial error is inserted into the coupled system. The error is introduced by starting the FSI computation with inconsistent time step sizes. The FSI computation starts from a fully developed fluid flow and structure at rest with time step sizes $\Delta t = 0.0003$ s, 0.0015 s, and 0.0075 s, while the second order BDF expects the same time step size for the last time

steps stored in the fluid restart file. The actual fluid restart file is created with a time step size of $\Delta t = 0.015$ s, such that for all three time step sizes the error in the time stepping scheme can be triggered.

To quantify the error, which is mainly visible in the non-physical pressure

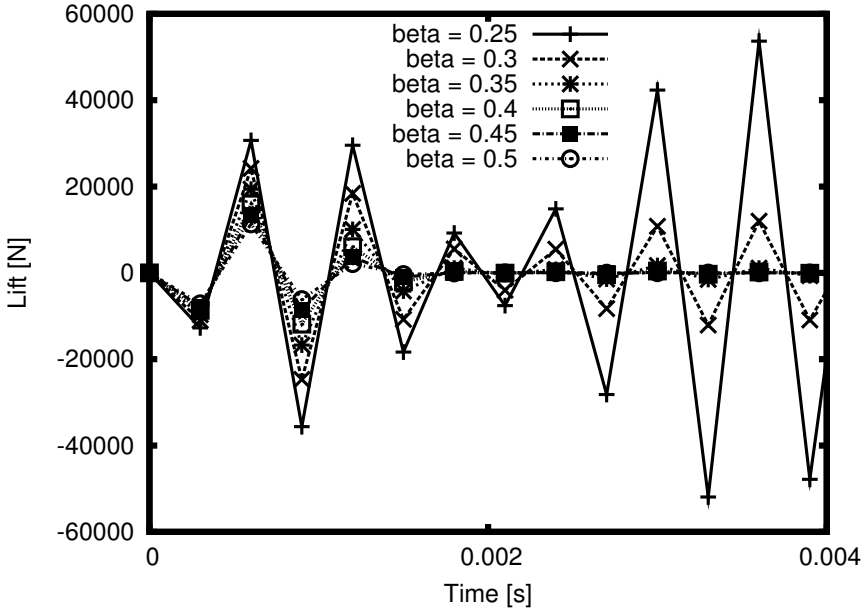


Figure 6.5: Detail of development of lift force for different Newmark parameters and $\Delta t = 0.0003$ s on the FSI 3 test case

distribution, the lift force is monitored over the first time steps of a coupled computation. The computation has been carried out on the coarsest of the above mentioned grids for fluid domain as well as for the structure domain. Furthermore the implicit partitioned coupling scheme has been used. The error occurs independently of the spatial discretization and the underlying coupling approach, thus the setup is valid for the following analysis.

Figures 6.5, 6.6, and 6.7 show the development of the lift force over time for the time steps 0.0003 s, 0.0015 s, and 0.0075 s respectively. The same physical time period of 0.015 s has been calculated with all time step sizes. Figure 6.5 shows a detailed view of only 0.004 s. The Newmark parameter β has been varied from 0.25, where no numerical dissipation is introduced, to 0.5, which inserts the maximum dissipation.

In order to find the optimal value for β, a scalar representation of the error has to be defined. The unphysical behaviour dominantly superimposes the lift change in between time steps. Thus the variation of the lift over a time period of 0.015 seconds minus the real variation over this time step is considered. Comparing figures 6.5, 6.6, and 6.7 yields, that the error amplitude significantly changes with different time step sizes. Also, the

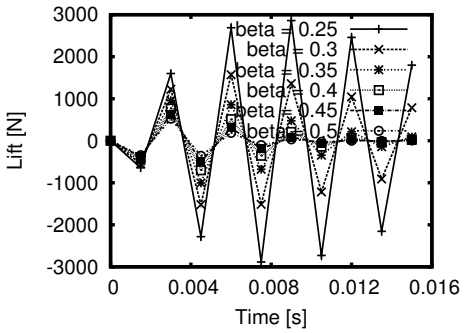


Figure 6.6: Development of lift force for different Newmark parameters and $\Delta t = 0.0015$ s on the FSI 3 test case

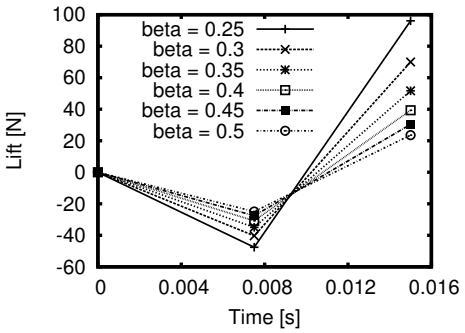


Figure 6.7: Development of lift force for different Newmark parameters and $\Delta t = 0.0075$ s on the FSI 3 test case

quantity in question is rather the error damping than the absolute value. Thus, the variation difference is scaled by the lift in the first time step. The resulting relative variation difference

$$V_\Delta = \frac{1}{N \text{lift}_1} \left(\sum_{n=0}^{N} |\text{lift}_{\Delta,n+1} - \text{lift}_{\Delta,n}| - |\text{lift}_N - \text{lift}_0| \right) \tag{6.2}$$

is plotted for different time step sizes in figure 6.8. The Δ above stands for the time step size and N is the number of time steps for this time step size. Figure 6.8 shows a strong variance of the results from different time steps

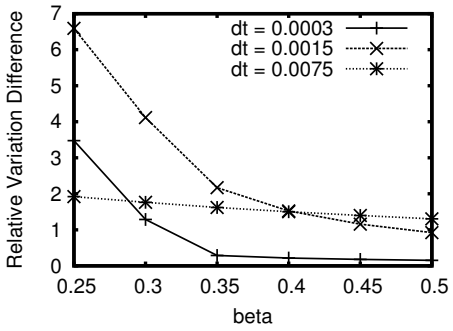


Figure 6.8: Relative variation of lift force for different time steps

and no trend for the time step size. A common behaviour throughout all time steps is the decrease of variance towards larger values of β. Generally it can be deduced from the results, that the largest improvement can be accomplished when shifting β from 0.25 to 0.3 or 0.35. Keeping in mind, that adding numerical dissipation can alter the solution of the coupled computation, the β value should be chosen as small as possible, though according to

these opposing requirements the computations in this work are carried out with the best comprising value $\beta = 0.3$.

6.2 Performance

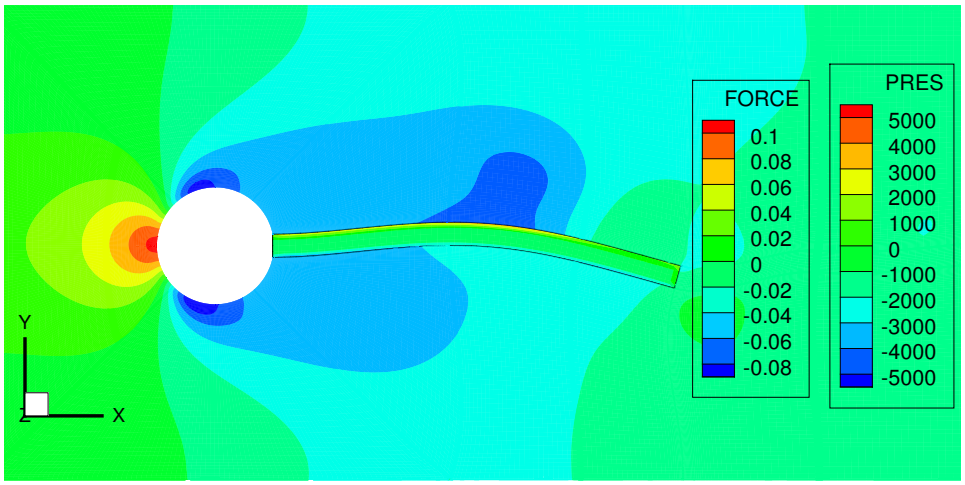


Figure 6.9: Snapshot of the FSI 3 test case (fluid pressure and forces acting on the structure shown)

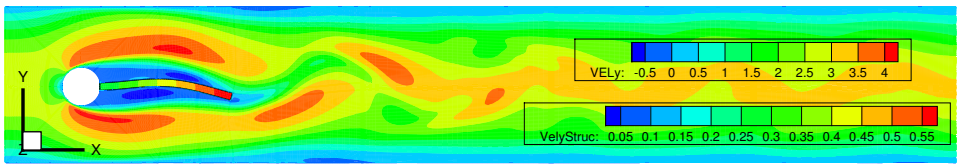


Figure 6.10: Snapshot of the FSI 3 test case (fluid x velocity and structure y velocity shown)

The third test case used in this work is the coupled FSI simulation called FSI 3. The FSI 3 test case is used to compare the performance of the MG CPL implementation as well as the extrapolation approach to the implicit partitioned coupling approach. To compare the performance of the different implementations, the average CPU time per time step is monitored. As the CPU time is always subject to minor variations due to the operating system, the most time consuming part of the coupling process, namely the number of fluid fine grid iterations, is monitored as well. This also allows for comparison in between iterations and CPU time, which in the case of

MG CPL shows the cost of the extensive coupling.
As for convergence criteria, a relative change in lift of less than $1 \cdot 10^{-5}$ is assured with a FSI convergence criterion of summarized displacement at the FSI boundary of less than $1 \cdot 10^{-5}$. Furthermore the fluid convergence criterion is the summed residual of less than $1 \cdot 10^{-7}$ and the structure criterion, a decay of $1 \cdot 10^{-16}$ of the energy norm, is set as before. These criteria allow the setup to calculate the values in question to a difference of less than 0.01% in between different solution approaches. Hence, the difference in solutions of different approaches is not monitored.
The computations were carried out on an Intel i7 CPU (2.93 MHz), while using its 4 cores for a parallel fluid computation. The fluid computation is parallelized by domain decomposition and the communication is handled by OpenMPI.
Figures 6.9 and 6.10 show a snapshot of one time step during the FSI computation. Figure 6.10 shows the stream-wise velocity in the fluid domain and the velocity component perpendicular to the stream, which is the main velocity component in the beam, in the structure. Figure 6.9 shows a detailed view of the computational domain. The fluid domain is colored with pressure distribution and the structure is colored with the loads acting on the boundary. Tables 6.2 and 6.3 show the drag, lift and displacement

Grid	Lift [N] (Freq. [1/s])	Drag [N] (Freq. [1/s])
coarse	-1.09 ± 180.38 (8.00)	451.06 ± 5.80 (15.39)
middle	1.91 ± 174.06 (5.44)	457.65 ± 27.99 (11.11)
fine	2.72 ± 175.66 (5.50)	459.00 ± 29.00 (10.75)
Turek	2.22 ± 149.78 (5.3)	457.3 ± 22.66 (10.9)

Table 6.2: Drag and Lift forces acting on the structure in the FSI 3 test case

Grid	x Disp. [10^{-3}m] (Freq. [1/s])	y Disp. [10^{-3}m] (Freq. [1/s])
coarse	1.67 ± 12.03 (8.00)	-0.38 ± 0.32 (16.67)
middle	1.47 ± 34.26 (5.38)	-2.85 ± 2.69 (10.87)
fine	1.45 ± 34.80 (5.47)	-2.91 ± 2.74 (10.99)
Turek	1.48 ± 34.38 (5.3)	-2.69 ± 2.53 (10.9)

Table 6.3: Displacement of point $A = (0.6, 0.2)$ in the FSI 3 test case

of the FSI 3 test case for the three different discretizations for the fluid and structure domain. Coarse, middle, and fine stand for the respective discretizations of the fluid domain in section 6.1.2 and the structure discretization in section 6.1.3. The tables also show the results from Turek and Hron for comparison.
The fine grid differs from Turek and Hrons solution by maximum 17.34% concerning force amplitude and 3.77% in frequency. Looking at the tip dis-

placement, the fine grid brings along an error of 8.24% in amplitude and 3.21% in frequency. The middle grid results in an error of 15.77% in force amplitude and 1.93% in frequency. For the displacement an error of 6.13% in amplitude and 1.51% in frequency can be found comparing to Turek and Hron. In comparison with the results of the pure fluid simulation in table 6.1, the fine and middle grid both show drastically less error in amplitude and more error in frequency. Figures 6.11 and 6.12 show the tip displace-

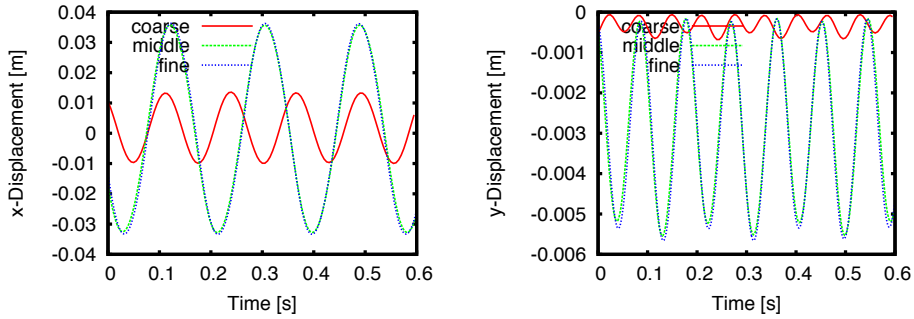


Figure 6.11: x and y tip displacement for the three different grids on the FSI 3 test case with $\Delta t = 0.005$ s, 0.002 s, and 0.001 s, respectively.

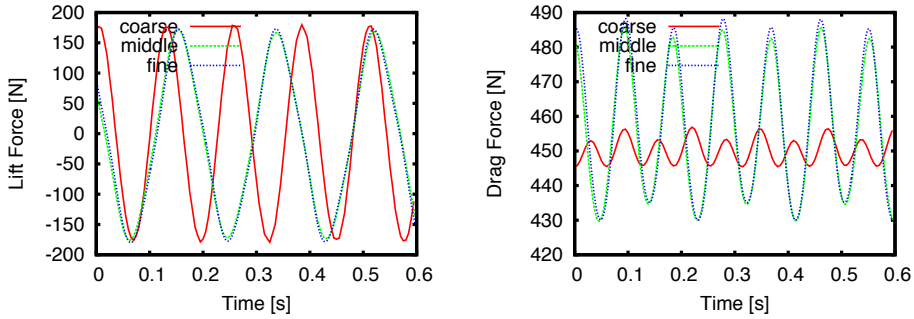


Figure 6.12: Drag and lift force for the three different grids on the FSI 3 test case with $\Delta t = 0.005$ s, 0.002 s, and 0.001 s, respectively.

ment and the forces acting on the structure over time. The computation on the coarsest mesh is not able to display the correct tip movement as expected, neither in frequency nor in amplitude. It does, however, display the correct amplitude of the lift.

6.2.1 Multigrid Coupling

One possibility to accelerate the implicit procedure is to incorporate the coupling into the fluid solving instead of waiting for the fluid to converge in every coupling step. When using this acceleration, coupling is applied

after a constant number of fluid solves or the fluid residual reaching its convergence criterion, depending on which is prior. This gradually changes the fluid system, and therefore the coupled problem is solved while the fluid is solved. This indeed results in more FSI iterations, but less overall CPU time. Heck [29] has shown, that less V-cycles on the fluid system per FSI iteration leads to faster convergence of the coupled problem. Thus, in order to provide a basis for fair testing, the comparison in this chapter involves only setups with at least one FSI iteration per fluid V-cycle.

Furthermore, the possibility of solving both the fluid and the structure

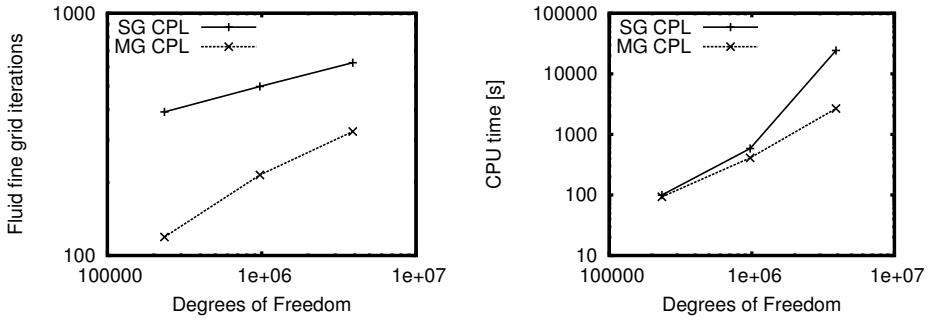


Figure 6.13: Fluid fine grid iterations and overall CPU time for the implicit partitioned coupling approach (SG) and MG CPL for the FSI 3 test case

problem on a single grid is not included in the comparison. To validate this limitation a test run on the coarse grid with a time step of $\Delta t = 0.002$ s has been carried out. The pure single grid computation needs an average of 3409 fluid fine grid iterations, whereas the implementation using V-cycles for the fluid and the standard FEAP conjugate gradient solver for the structure needs 391.6, and the MG CPL needs 119.2 fluid fine grid iterations. Thus, only the two latter competitive alternatives are included in the comparison. Figure 6.13 shows the number of fluid fine grid iterations and the overall CPU time per time step for three different discretizations of the FSI 3 test case plotted over the summarized number of degrees of freedom. The implicit partitioned (SG CPL) case implicates a two level multigrid fluid solver and the standard conjugate gradient structure solver. The MG CPL implicates a two level multigrid solver for the entire domain as described in chapter 4. The coarse, middle, and fine discretization from the last sections for the fluid and structure are used for comparison. As the time step size cannot be held constant over the different resolutions, the results for the smallest time step size of every resolution are compared. The MG CPL clearly outruns the implicit partitioned approach in terms of fluid fine grid iterations. This effect is due to the more frequent coupling. In terms of CPU time, the MG CPL also outruns the partitioned case and the speed up even grows with increasing degrees of freedom. Even more important, the MG CPL

shows linear behaviour over the number of degrees of freedom. Hence, the multigrid performance on the coupled FSI problem is shown.
A more detailed analysis and comparison of these solution approaches is given in section 6.3.

6.2.2 Extrapolation

In this section the different extrapolation methods described in chapter 5 are applied to the FSI 3 test case. The objective is to compare the different methods with respect to numerical efficiency. All computations were carried out with a fix FSI underrelaxation factor of 0.1 in order to ensure that the acceleration originates only from the employed extrapolation approach. From constant to third order polynomial extrapolation is used for the displacement as well as for the force extrapolation. Constant displacement extrapolation coincides with the partitioned implicit approach without any force or displacement extrapolation, as the grid displacement originating from the last time step is used as initial displacement. The comparison is carried out on the middle and coarse fluid and structure grid and the FSI 3 setup with boundary conditions as described in section 6.1.2 and 6.1.3 are used. In order to analyze the behaviour of the extrapolation when altering the time step size, both grids were computed with three different time step sizes. The coarse grid was computed with $\Delta t = 0.002$ s, 0.005 s, and 0.01 s and the middle grid with $\Delta t = 0.001$ s, 0.002 s, and 0.004 s, respectively.

Figures 6.14, 6.15, and 6.16 show the average number of fluid fine grid iter-

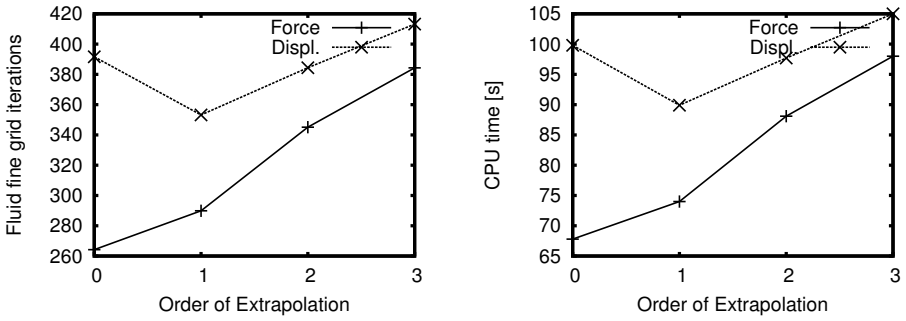


Figure 6.14: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.002$ s for force and displacement extrapolation on the coarse grid

ations and average CPU time for one time step in different implementations of extrapolation for the coarse fluid and structure grid. The performance of the extrapolation on the coarse grid is strongly dependent on the time step size. For a time step size of 0.002 s the first order displacement extrapolation accelerates the computation, but higher orders even slow the computation

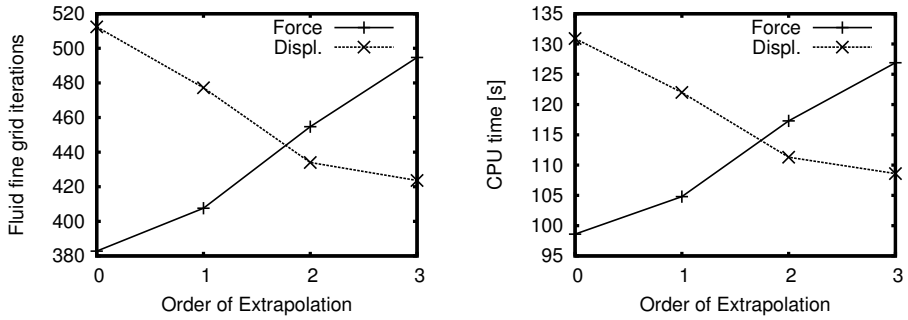


Figure 6.15: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.005$ s for force and displacement extrapolation on the coarse grid

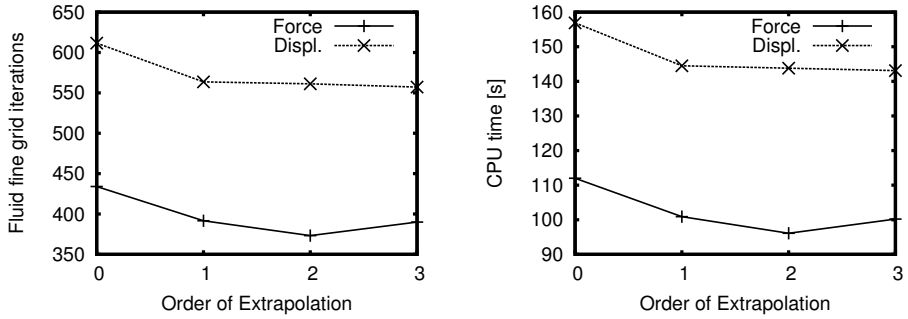


Figure 6.16: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.01$ s for force and displacement extrapolation on the coarse grid

down. In the case of force extrapolation, the highest acceleration of 1.47 in CPU time compared to no extrapolation can be found for the constant polynomial. Higher orders are inferior to the constant extrapolation, but still faster than the displacement counterpart and faster than no extrapolation at all, which coincides with constant displacement extrapolation.
For the coarse grid and the middle time step size of 0.005 s the displacement extrapolation improves as the order of extrapolation rises. The force case shows the same behaviour as for the smaller time step. Again the best improvement of 1.32 is achieved by constant force extrapolation. The higher order force extrapolation performs worse as the order rises and is even slower than the displacement extrapolation beyond second order. However, even the third order force extrapolation performs better than no extrapolation at at.
For the largest time step of 0.01 s the displacement extrapolation follows the expected slope, but extrapolation orders higher than one have no significant influence. The gain of the force extrapolation increases as the order

of extrapolation rises, resulting in the maximum acceleration of 1.63 for the second order force extrapolation.

Figures 6.17, 6.18, and 6.19 show the average number of fine grid itera-

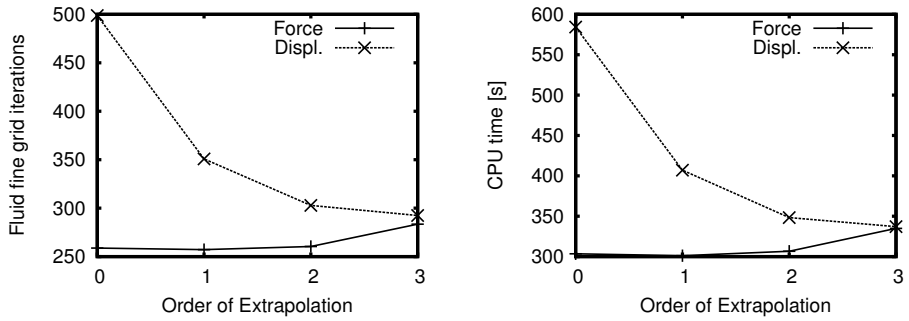


Figure 6.17: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.001$ s for force and displacement extrapolation on the middle grid

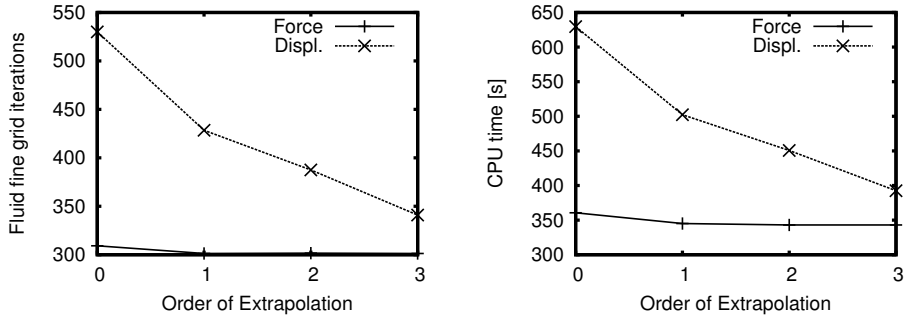


Figure 6.18: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.002$ s for force and displacement extrapolation on the middle grid

tions in the fluid solver as well as the average overall CPU time for different extrapolation implementations and the middle fluid and structure grid. For the middle grid and the smallest time step size of 0.001 s the acceleration from the displacement extrapolation increases as the order of extrapolation increases. But nevertheless, it is outperformed by the force extrapolation. The maximum acceleration of the computation of 1.94 can be found for first order force extrapolation.

The simulations with time step size 0.002 s show a similar behaviour as for the $\Delta t = 0.001$ case in displacement extrapolation, which means the computations decreases as the order of extrapolation increases. The force extrapolation shows increasing acceleration for rising orders and the maximum speed up of 2.09 is reached for third order force extrapolation. And

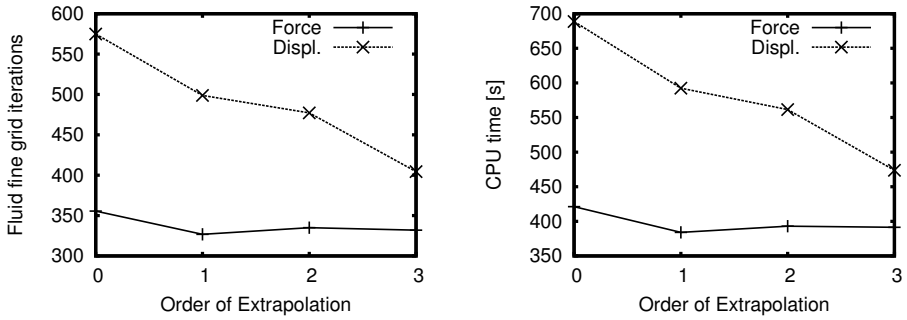


Figure 6.19: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.004$ s for force and displacement extrapolation on the middle grid

again the force case outperforms the displacement case.
For the time step $\Delta t = 0.004$ s and the middle grid the expected slope for the displacement extrapolation is continued. The force extrapolation in this case is superior as well and hardly dependent on the applied order and the maximum speed up of 1.74 is reached at first order.
In conclusion, the use of extrapolation in between time steps in FSI calculations has a significant effect on the required overall computing time for coupled simulations. This effects even rises for larger time step sizes. Furthermore, the force extrapolation has shown to be superior to the displacement extrapolation throughout all time steps and resolutions, and the higher order extrapolation is even more useful when applied to large time steps. By linear extrapolation a maximum acceleration factor of 2.55 for the fine grid can be predicted. Given the manageable mathematical complexity and the negligible memory usage, the force extrapolation is a powerful tool in speeding up partitioned coupling simulations.

6.2.3 Multigrid Coupling & Extrapolation

Since the extrapolation of forces and displacements has a significant influence on the CPU time of implicit partitioned coupling calculations, the influence on the MG CPL is also investigated. In terms of implementation, the coarse grid solution as well as all solution dependent parts of the coarse grid elasticity and Navier Stokes equation are recomputed at every restriction. Therefore, the implementation of force and displacement extrapolation runs analogously to the implicit partitioned case. In this section the same grids and time steps as in the last section are used.

Figures 6.20, 6.21, and 6.22 show the average number of fluid fine grid iterations as well as the average CPU time for the coarsest fluid and structure grid and different extrapolation implementations. For the smallest time step size of 0.002 s and the displacement extrapolation a qualitatively similar

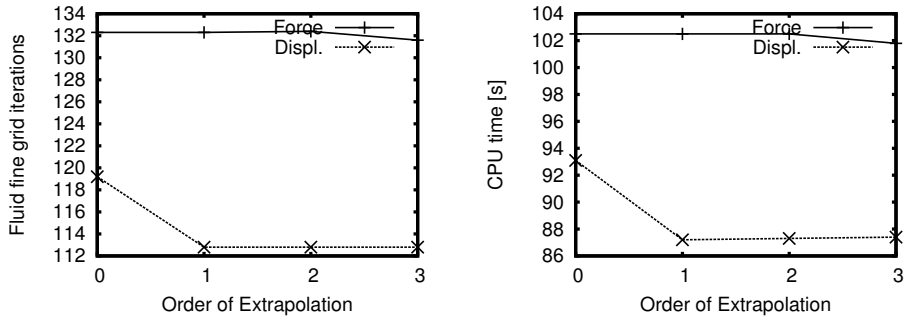


Figure 6.20: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.002$ s for force and displacement extrapolation with MG CPL on the coarse grid

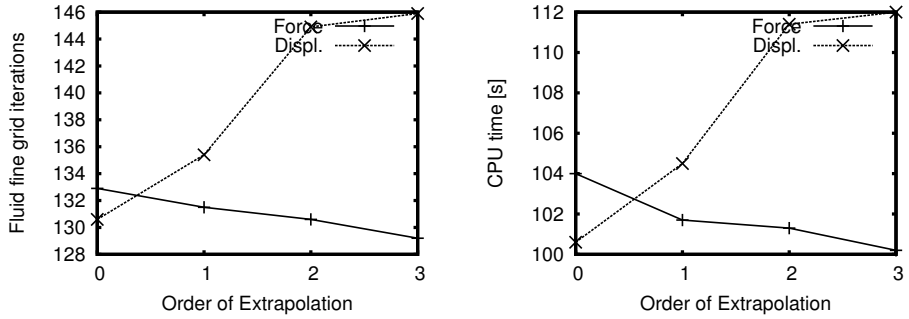


Figure 6.21: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.005$ s for force and displacement extrapolation with MG CPL on the coarse grid

characteristics as in section 6.2.2 can be found. Contrary to the findings in the last section the force extrapolation shows a worse performance than the non-extrapolated case independent of the order. The highest acceleration of 1.07 can be found for the first order displacement case.
The computations for time step size 0.005 s show a qualitatively similar behaviour as for the implicit partitioned case in figure 6.15. Again, the best performance is shown for the third order displacement extrapolation, but with no noticeable speed up it is not competitive to the implicit partitioned case.
Comparing the results for the coarsest time step 0.01 s and the coarse grid, the displacement extrapolation decelerates the computation with rising order. For the force extrapolation no noticeable speed up can be found. The different results in the CPU time are due to the variation of thread handling in the operating system and only visible due to the small scale.
Figures 6.25, 6.24, and 6.23 show a comparison of different orders of interpolation in the MG CPL approach applied the middle fluid and structure

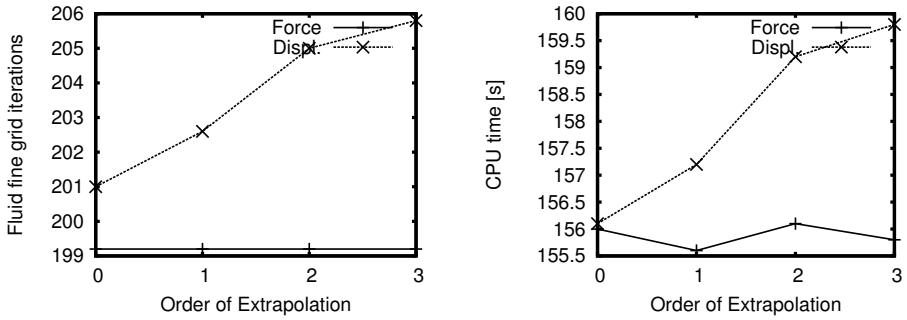


Figure 6.22: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.01$ s for force and displacement extrapolation with MG CPL on the coarse grid

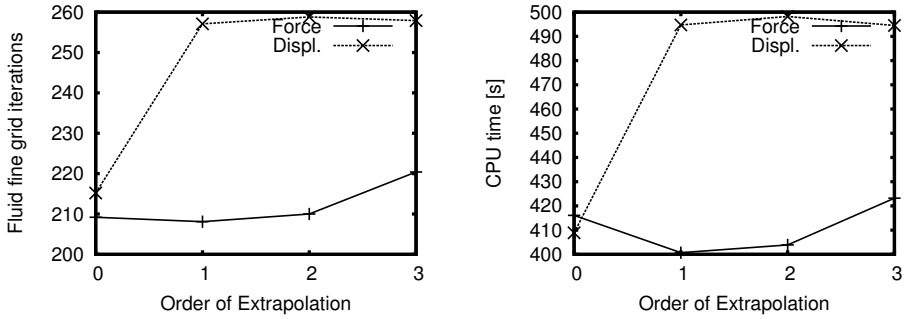


Figure 6.23: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.001$ s for force and displacement extrapolation with MG CPL on the middle grid

grid and time steps 0.004 s, 0.002 s, and 0.001 s, respectively. The average number of fine grid iterations and the average CPU time per time step are shown.

For the time step size 0.001 s it can be seen, that higher order displacement extrapolation slows the computations down, and even for the best performing case, which is first order force extrapolation, a speed up of only 1.02 can be achieved.

The force extrapolation for time step 0.002 s is superior to the displacement extrapolation ,which shows basically no effect. However, the advantage of the force extrapolation is still small enough to go down under the thread scheduling in terms of CPU time.

For the largest time step size of 0.004 s the number of fine grid iterations as well as the overall CPU time are so close to each other, that no conclusions from the effect of the extrapolation can be drawn.

Although the extrapolation achieves essential speed ups for the implicit partitioned case, this kind of behaviour cannot be observed for the MG CPL

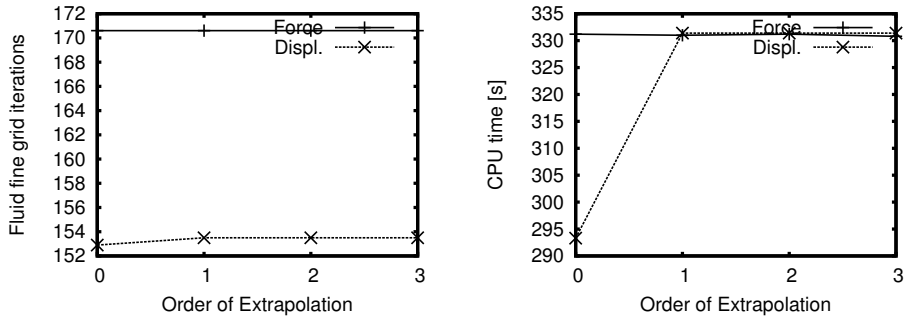


Figure 6.24: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.002$ s for force and displacement extrapolation with MG CPL on the middle grid

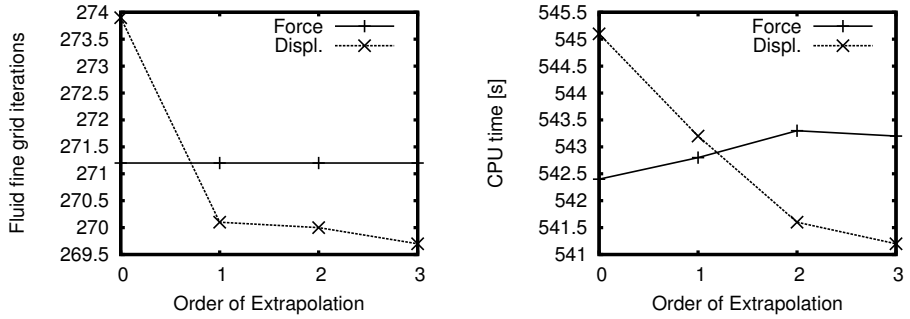


Figure 6.25: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.004$ s for force and displacement extrapolation with MG CPL on the middle grid

approach. For some cases the extrapolation even decelerates the computation. Also, the impact of the extrapolation on the CPU time of the MG CPL calculations is considerably smaller than the impact in the implicit partitioned case. The best result can be found for the coarse grid and the smallest time step $\Delta t = 0.002$ s with third order displacement extrapolation. But even here a speed up of only 1.06 compared to the non extrapolated case can be seen.

A possible explanation for this behaviour could be that the extrapolation reduces the absolute value of the error in the coupled system, but simultaneously renders it less smooth. In this the advantage of the MG CPL, effectively reducing smooth error components, would become useless. This supposition as well as the combination of MG CPL and extrapolation needs further investigation in the future.

6.3 Robustness

In this section the multigrid coupling implementation as described in chapter 4 as well as the implicit partitioned coupling is applied to the FSI 3 test case. The objective is to analyze the behaviour of the different approaches regarding different discretizations, time step sizes and underrelaxation factors. For this purpose the number of fine grid fluid iterations as well as the overall CPU time for one time step is compared. The computations were performed using an implicit partitioned coupling approach with a two level multigrid fluid solver and the standard conjugate gradient structure solver on the one hand and a two level MG CPL solver as described in chapter 4 on the other hand. All computations were carried out with the same number of solver iterations on each grid on an Intel Core i7 (2.93 MHz). The coarse grid was computed with the time step sizes $\Delta t = 0.002$, 0.005, and 0.01, and the middle and fine grid with $\Delta t = 0.001$, 0.002, and 0.004.

Figures 6.26, 6.27, and 6.28 show the number of fluid fine grid iterations

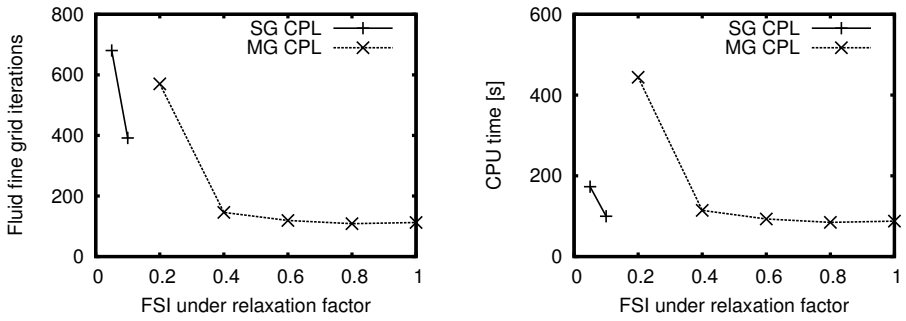


Figure 6.26: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.002$ s for implicit partitioned coupling (SG) and MG CPL on the coarse grid

and the overall CPU time for one time step using the implicit partitioned approach and the MG CPL approach on the coarse grid. The computations for different values of the FSI underrelaxation factor, which is multiplied to the boundary displacement when transferring it to the fluid domain, are shown. For the implicit partitioned approach underrelaxation of more than 0.1 leads to divergence and underrelaxation below 0.05 is always slower than the results shown. This is also true for the MG CPL case with underrelaxation below 0.2, thus these cases are not considered in the comparison. For a time step size of $\Delta t = 0.002$ and the coarse grid the MG CPL clearly outruns the implicit partitioned case in terms of fluid fine grid iterations for an underrelaxation of 0.4 and above. A speed up of 3.61 can be measured when comparing the implicit partitioned approach with an underrelaxation of 0.1 and the MG CPL with an underrelaxation of 0.8. In terms of CPU

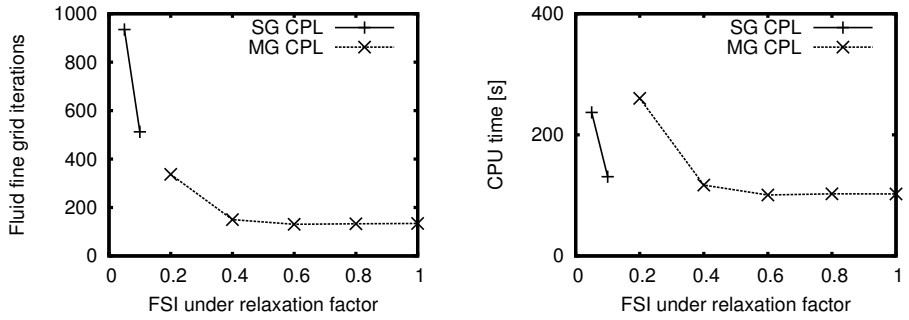


Figure 6.27: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.005$ s for implicit partitioned coupling (SG) and MG CPL on the coarse grid

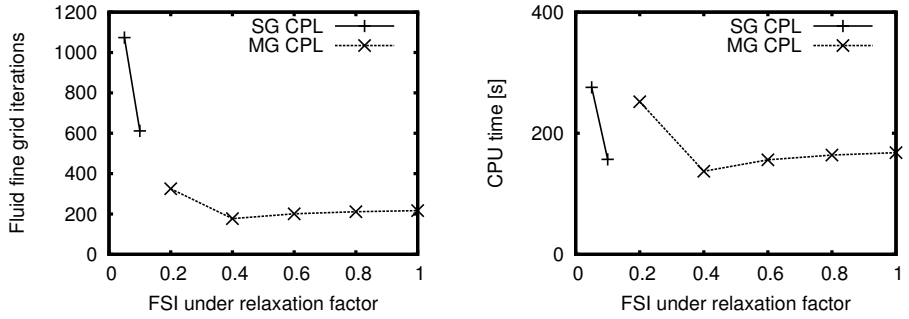


Figure 6.28: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.01$ s for implicit partitioned coupling (SG) and MG CPL on the coarse grid

time, this advantage shrinks to a speed up of 1.18, but the MG CPL still outperforms the partitioned approach.
When increasing the time step size the gap in the different approaches increases. For the time step size of $\Delta t = 0.005$ the MG CPL with an underrelaxation of 0.2 and above outruns the best performing implicit partitioned approach in terms of fluid fine grid iterations. The maximal speedup compared to the implicit partitioned approach with underrelaxation 0.1 is 3.92 in terms of iterations and 1.30 in terms of CPU time. These speedups are found at an underrelaxation of 0.6.
For a time step size of $\Delta t = 0.01$ and the coarse grid, the MG CPL again outperforms the partitioned approach by factors of 3.46 and 1.14 in fluid fine grid iterations and CPU time, respectively. Here, the best performing MG CPL underrelaxation factor is 0.4.

Figure 6.29 shows the average number of fluid fine grid iterations and the CPU time for one time step for the middle grid and a time step size of $\Delta t = 0.001$ s. Again, the MG CPL outruns the partitioned approach for

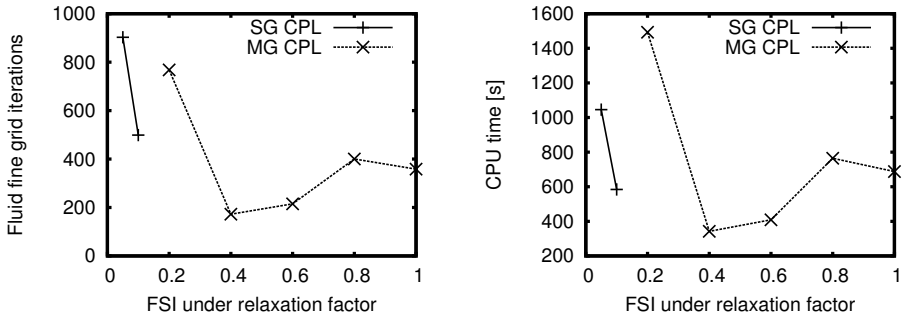


Figure 6.29: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.001$ s for implicit partitioned coupling (SG) and MG CPL on the middle grid

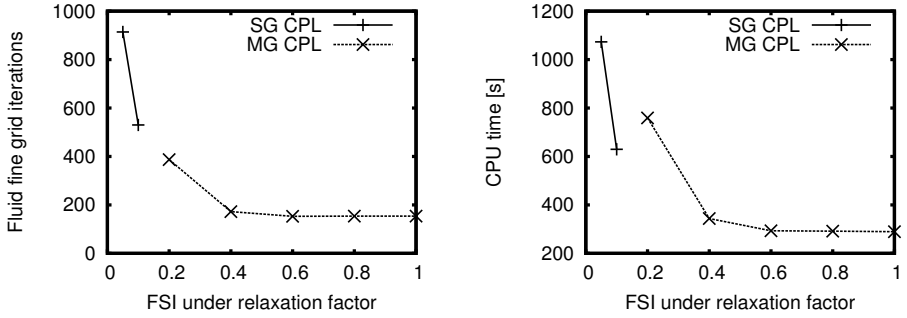


Figure 6.30: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.002$ s for implicit partitioned coupling (SG) and MG CPL on the middle grid

underrelaxation factors above 0.4. The performance of the MG CPL shows a dependence on the underrelaxation, but no distinct behaviour can be found. The maximum speed up in terms of fluid fine grid iterations and CPU time is 2.84 and 1.71, respectively and can be found at an underrelaxation factor of 0.4.

Figure 6.30 shows the average number of fluid fine grid iterations and CPU time for the time step size $\Delta t = 0.002$ s. As expected the implicit partitioned case needs slightly more iterations, thus more CPU time, as for $\Delta t = 0.001$ s. The MG CPL case needs even less iterations and is even less sensitive to the choice of the FSI underrelaxation factor compared to the 0.001 case. The maximum speed up is 3.47 and 2.15 for fluid fine grid iterations and CPU time respectively.

The gap in between fluid fine grid speed up and CPU time speed up narrows when further increasing the time step size. Figure 6.31 shows the fluid fine grid iterations and CPU times for the middle grid and the time step size of $\Delta t = 0.004$. Here the maximum speed up is 2.13 for fluid fine grid itera-

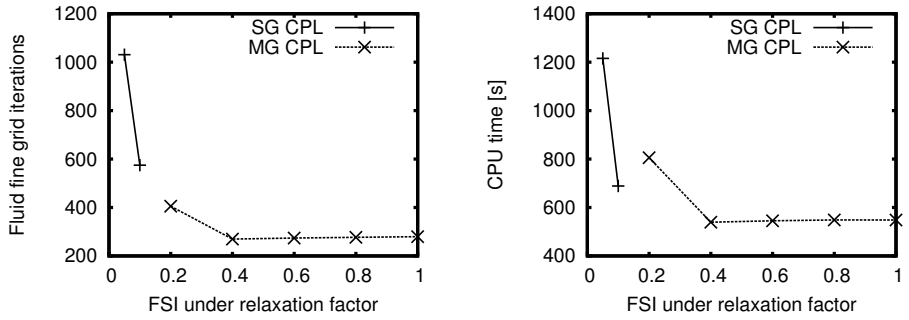


Figure 6.31: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.004$ s for implicit partitioned coupling (SG) and MG CPL on the middle grid

tions and 1.28 for CPU time. The maximum speed up can be found at an underrelaxation of 0.4, but the difference in comparing with higher values is negligible.

Figures 6.32, 6.33, and 6.34 show the fluid fine grid iterations and CPU

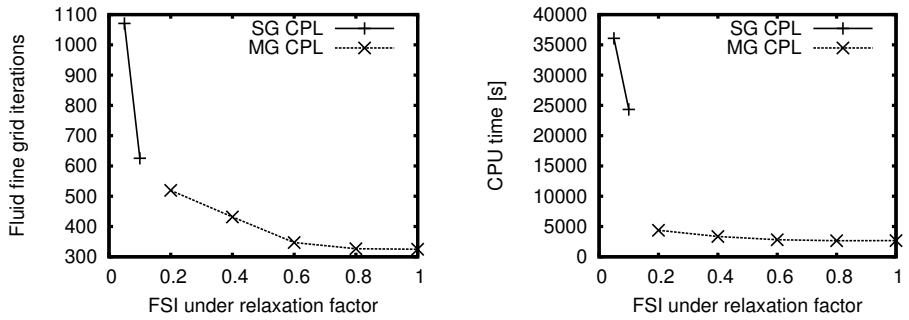


Figure 6.32: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.001$ s for implicit partitioned coupling (SG) and MG CPL on the fine grid

time for the FSI 3 calculation on the fine grid with the time step sizes 0.001, 0.002, and 0.004, respectively. Compared to the middle grid the gap between the partitioned approach and the MG CPL increases in terms of CPU time. For a time step size of 0.001 and the finest grid, the MG CPL outperforms the partitioned approach for any underrelaxation factor above 0.2. The maximum speed up of 1.92 and 9.10 in fluid fine grid iterations and CPU time, respectively, can be found at an underrelaxation factor of 1.0. Thus, in contrast to the coarser grids considered above, the speed up in CPU time is greater than the speed up in fine grid iterations.
Looking at the performance of the partitioned approach for the time step size $\Delta t = 0.002$ on the finest grid, the best achievable performance is no

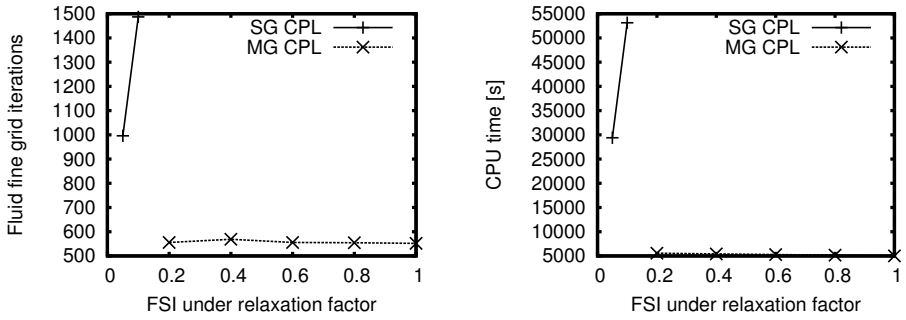


Figure 6.33: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.002$ s for implicit partitioned coupling (SG) and MG CPL on the fine grid

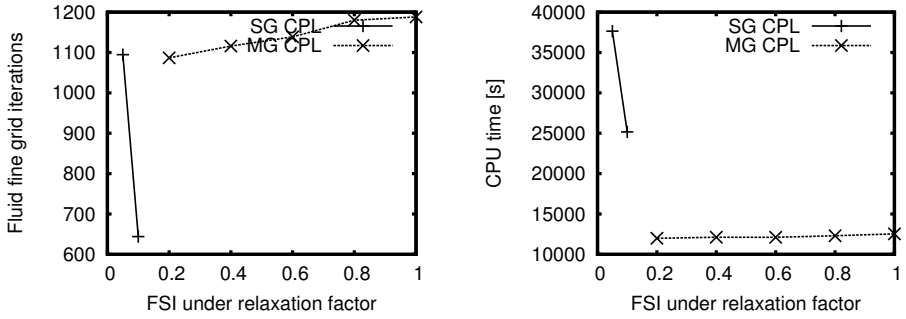


Figure 6.34: Number of fluid fine grid iterations and overall CPU time for one time step of $\Delta t = 0.004$ s for implicit partitioned coupling (SG) and MG CPL on the fine grid

longer at an underrelaxation of 0.1, but at 0.05. The MG CPL is hardly dependent on the large time step, showing better stability features and the best performance with an underrelaxation factor 1.0. The maximum speed up when comparing the best results of the two approaches are 1.80 in fluid fine grid iterations and 5.86 in CPU time. These are the factor by which the MG CPL is faster than the implicit partitioned approach.
When comparing the approaches for the time step size of $\Delta t = 0.004$, the MG CPL shows a weak dependence on the underrelaxation factor, as the best performance can be seen at an underrelaxation factor of 0.2. These differences are still negligible in comparison with the CPU time speed up. For this setup the partitioned approach is even faster in terms of fluid fine grid iterations, but looking at the overall CPU time maximum speed up of 2.10 for the MG CPL implementation can be found.

In conclusion, the MG CPL implementation shows a wider range of possible underrelaxation factors than the implicit partitioned approach and the

run time is less dependent on it. Comparing the number of iterations and the CPU time on the coarse and middle grid yields, that the MG CPL needs in average more CPU time per fluid fine grid iteration than the implicit partitioned approach. This originates from the high usage of the coupling interface. The time consuming MpCCI coupling and the grid movement are used extensively, therefore resulting in a higher CPU time. On the fine grid this effect is subordinate to the performance of the multigrid method.
Comparing the overall CPU time shows the great advantage of the MG CPL. A maximum acceleration of 9.10 in terms of CPU time can be found for the finest discretization and time step size 0.001, when comparing the implicit partitioned approach with underrelaxation 0.1 with the MG CPL approach with underrelaxation 1.0.
As the coupled solution procedure, which acts as smoothing, can be thought of as a block Gauß Seidel method, setting the FSI underrelaxation factor to values greater than 1.0 results in a block SOR method. Since the MG CPL has shown a constant behaviour over a wide range of underrelaxation factors and no sign of instability at no underrelaxation at all, the use of overrelaxation is possible.
And finally, as the resolution becomes finer, the speed up in CPU time catches up with the speed up in fluid fine grid iterations, and finally passes it. This is an intended result, as it shows once again the superiority of the multigrid method on large system.

Chapter 7

Conclusions and Outlook

In this work the multigrid coupling approach for FSI simulations has been introduced, which is situated in between the implicit partitioned and the monolithic one. Due to a close coupling as in the monolithic approach and the modularity of the implicit approach the multigrid coupling approach combines the advantages of the both.
The multigrid coupling approach is constructed by applying the nonlinear multigrid method to the coupled FSI problem. By this construction the implicit coupling algorithm of two independent codes is used as a smoother for the multigrid method. The assembly of the coupled systems has been carried out based on the coupled FSI system in ALE representation. A nonlinear geometric multigrid solver has been implemented into the structure solver FEAP. Furthermore the coupling interface in FEAP as well as in the fluid solver FASTEST has been enhanced to cope with multigrid coupling. Furthermore the multigrid implementation in FASTEST has been adapted to use the implicit partitioned coupling procedure as a smoother for the nonlinear multigrid.
To ensure a valid basis for comparison a set of test cases from a well known FSI benchmark were employed. Furthermore an analysis of the employed discretizations in the fluid and structure domain was carried out individually. Finally the multigrid coupling approach as well as the most promising implicit partitioned approach with only one V-cycle per FSI iteration were applied to a steady-state and an unsteady test case with different spatial discretizations and time step sizes.
It has been shown, that the multigrid coupling approach outperforms the implicit partitioned approach in the number of fluid fine grid iterations as well as CPU time for the overall coupled simulation. Speed ups up to 9.10 can be achieved, and even on the coarse grid, speed ups of up to 1.3 are possible.
In comparison to the implicit partitioned approach the multigrid coupling approach does not only solve the fluid and structure problem with a geomet-

ric multigrid solver, but also reduces the error arising from the coupling on the fine as well as coarse grids. Furthermore, as the multigrid coupling stabilizes itself, a more effective underrelaxation and even overrelaxation factors can be used. In conclusion, it has been shown, that the multigrid coupling outperforms the partitioned approach as expected and presents a promising addition in between the partitioned and monolithic coupling approach.

As a second subject, the force extrapolation to accelerate unsteady coupled FSI simulations has been introduced. In contrast to common extrapolation methods used in FSI, here the forces acting on the structure domain are extrapolated and not the displacements acting on the fluid domain. Therefore an additional structure solve is carried out with the extrapolated values. In the common extrapolation methods an entire coupling step is simulated by the extrapolation. When using the force extrapolation only half of the step is simulated and the other half is calculated, resulting in a more accurate prediction of the simulation in the current time step.
For a comparison the force and displacement extrapolation based on the same polynomial interpolation methods have been implemented into a partitioned coupling scheme. As interpolation polynomials the constant and first through third order Lagrange interpolation are used.
Application to different discretizations and time steps of an unsteady benchmark test case have shown the superiority of the force extrapolation. Especially for large time steps the calculations using force extrapolation are even for small orders considerably faster than the ones using any of the implemented orders of the displacement case. Speed ups of up to 2.09 are possible using the force extrapolation, whereas for the displacement case the highest speed up is 1.73. These result are very promising, as they were achieved by simple polynomial extrapolation and the force case is applicable to more sophisticated extrapolation methods.

As for future prospects, the full potential of overrelaxation of the coupled computation in multigrid coupling needs further investigation. Also, applications with a larger FSI interface and which require closer coupling will benefit from the multigrid coupling. This approach can be actually transferred to any multi-physical numerical simulation which requires strong coupling in its solution algorithm. And finally, the combination of force extrapolation with multigrid coupling would benefit from further investigation.

Appendix A

Arbitrary Lagrangian-Eulerian Framework

The definition of the mappings between the different frameworks are defined as figure A.1 indicates. The relationship of the frameworks as well as their derivatives is given in the following.

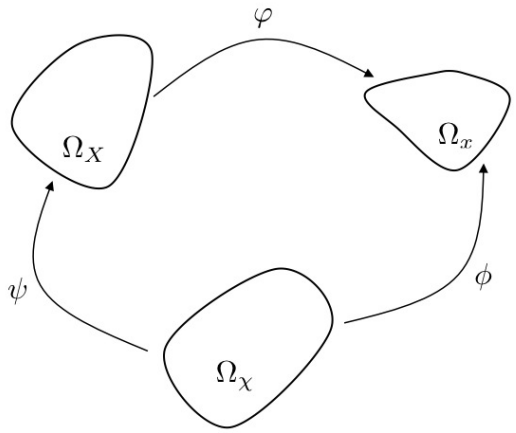


Figure A.1: Eulerian (Ω_x), Lagrangian (Ω_X) and ALE (Ω_χ) Framework with the corresponding mappings $\varphi : \Omega_X \to \Omega_x$, $\phi : \Omega_\chi \to \Omega_x$, and $\psi : \Omega_\chi \to \Omega_X$

A.0.1 Lagrangian Framework

Due to the material fixed coordinate system in a Lagrangian framework any time dependent control volume $\Omega_X(t)$ contains the same control mass throughout transient computation. The derivatives with respect to time t and space X state:

$$\left.\frac{d}{dX}X\right|_X = I \qquad \text{(A.1)} \qquad \left.\frac{d}{dt}X\right|_X = \frac{\partial X}{\partial t} \ . \qquad \text{(A.2)}$$

Note that the notation $(.)|_X$ means keeping X fixed. For a scalar valued function f defined in the Lagrangian framework the derivatives state:

$$\left.\frac{d}{dX}f(X,t)\right|_X = \frac{\partial f}{\partial X} \qquad \text{(A.3)} \qquad \left.\frac{d}{dt}f(X,t)\right|_X = \frac{\partial f}{\partial t} \ . \qquad \text{(A.4)}$$

A.0.2 Eulerian Framework

The derivatives with respect to space and time in the Eulerian or spatial framework state:

$$\left.\frac{d}{dX}x\right|_X = \frac{d\varphi(X,t)}{dX} = \nabla_X\varphi \quad \text{(A.5)} \qquad \left.\frac{d}{dt}x\right|_X = \frac{d\varphi(X,t)}{dt} = v \qquad \text{(A.6)}$$

with $\nabla_X = \frac{\partial}{\partial X}$ the gradient. For a scalar valued function defined in the Eulerian framework, the derivatives with respect to space and time are given as:

$$\left.\frac{d}{dX}f(x,t)\right|_X = \left.\frac{\partial f}{\partial x}\right|_x + \nabla_x f \ \frac{\partial x}{\partial X} = \left.\frac{\partial f}{\partial x}\right|_x + \nabla_x f \ \nabla_X\varphi \qquad \text{(A.7)}$$

$$\left.\frac{d}{dt}f(x,t)\right|_X = \left.\frac{d \ f(\varphi(X,t),t)}{dt}\right|_X = \left.\frac{\partial f}{\partial t}\right|_x + \nabla_x f \ v \ . \qquad \text{(A.8)}$$

The latter equation states the well known connection of Lagrangian and Eulerian frameworks. The material time derivative equals the spatial derivative plus a convection term.

A.0.3 Arbitrary Lagrangian Eulerian Framework

The crucial advantage of the ALE formulation is to be able to operate in non-attached coordinate systems, which can be adjusted to the individual needs. The drawback of this approach lies in more complicated expressions for derivatives:

$$\left.\frac{d}{dt}x\right|_\chi = \left.\frac{d}{dt}\phi(\chi,t)\right|_\chi = \frac{\partial x}{\partial t} \qquad \text{(A.9)} \qquad \left.\frac{d}{dt}\chi\right|_\chi = \left.\frac{d}{dt}\psi^{-1}(X,t)\right|_X = \frac{\partial \chi}{\partial t} \qquad \text{(A.10)}$$

$$\begin{aligned} \left.\frac{d}{dt}x\right|_X = \left.\frac{d}{dt}\varphi(X,t)\right|_X &= \left.\frac{d}{dt}\phi\left(\psi^{-1}(X,t),t\right)\right|_X \qquad \text{(A.11)} \\ &= \frac{\partial\phi}{\partial t} + \left.\frac{d\phi}{d\psi^{-1}}\right|_\chi \left.\frac{d\psi^{-1}}{dt}\right|_X \\ \Rightarrow \left.\frac{dx}{dt}(X,t)\right|_X &= \frac{\partial x}{\partial t}(\chi,t) + \left.\frac{dx}{d\chi}(\chi,t)\right|_\chi \left.\frac{d\chi}{dt}(X,t)\right|_X . \qquad \text{(A.12)} \end{aligned}$$

The left hand side of (A.12) represents the material velocity and the first term on the right hand side the grid velocity. Thus the remaining term constitutes the material velocity with respect to the ALE framework.

Appendix B

Discretization of the Momentum Equation in FASTEST

In the following the discretization and interpolation for the momentum equation as implemented in FASTEST is given. Although the actual implementation was created in 3D, for the sake of simplicity only the 2D descritption is given. Let V_f be a grid cell in the fluid domain.

The Discrete Momentum Equation The discretization of the cell centered and cell face centered terms is carried out according to the second order accurate midpoint rule. In the following the index c stands for one of $\{w, e, s, n\}$.

Convective Term For the convective term, the boundary integral results in

$$\int_{\partial V_f} \rho_f (v - v^g) v n \, dA \approx \sum_c \dot{m}_c^g v_c \tag{B.1}$$

with $\dot{m}_c^g$ the discrete mass and grid flux from equations 3.24 and 3.30 and v_c the velocity at the center of cell face c. The mass and grid fluxes are assumed to be known for the momentum equation, as they are computed at the end of the pressure coupling process.

Diffusive Term The diffusive term is discretized into the sum of gradients multiplied by the cell face area

$$\int_{\partial V_f} \mu_f (\nabla v + \nabla v^T) n \, dA \approx \sum_c \mu_f (\nabla v_c + \nabla v_c^T) \delta A_c \tag{B.2}$$

in which ∇v_c describes the discretized velocity gradient at the cell face c and δA_c the outward normal vector to the cell face c with $||\delta A_c||$ equals the area of the cell face.

Pressure Term As pressure is a scalar function, it can be posed as the pressure gradient integrated over V_f. Discretizing yields the gradient at the center of cell P times the cell volume δV_P.

$$\int_{\partial V_f} pI \; dS = \int_{V_f} \nabla p \; dV \approx \nabla p_P \; \delta V_P \tag{B.3}$$

in which the gradient is located at the cell face center, and δV_P is the volume of cell P.

External Forces External force terms are discretized as the force at the cell center multiplied with the cell volume.

$$\int_{V_f} \rho_f f_f \; dV \approx \rho_f f_{f,P} \delta V_P \tag{B.4}$$

Interpolation Cell face values such as the velocity in the convective term have to be interpolated as the values are computed and stored only on the cell centers. For the convective term different interpolation techniques can be used (and mixed) as they involve different drawbacks and advantages. In the following the interpolation is shown for the e face. The other faces are treated analogously.

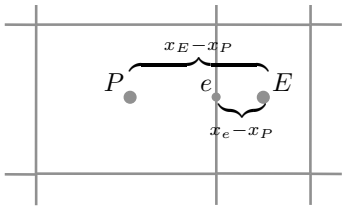


Figure B.1: Calculation of CDS interpolation factor

Central Differencing Scheme The straightforward attempt to obtain the cell face value is a linear interpolation between the two neighbouring cells. The interpolation parameter $\alpha_e = \frac{||x_e - x_P||_2}{||x_E - x_P||_2}$ is derived directly from the distance of points P and E from point e as shown in figure B.1.

$$v_e^{\text{CDS}} = \alpha_e v_E + (1 - \alpha_e) v_P. \tag{B.5}$$

This results in the second order accurate Central Differencing Scheme (CDS). The drawbacks of this scheme are possibly oscillatory solutions due to the second order and order reduction on non-Cartesian grids.

Upstream Differencing Scheme Another approach to obtain a cell face value is to use the cell center value in direction opposite to the mass and grid flux $\dot{m}_e^g$.

$$v_e^{\mathrm{UDS}} = \begin{cases} v_E & \text{, if } \dot{m}_e^g < 0 \\ v_P & \text{, else} \end{cases} \tag{B.6}$$

This results in the Upstream Differencing Scheme (UDS), which is first order accurate. Application of this scheme results in a numerically more stable system, but due to the second order derivative in its error term it introduces numerical diffusion into the system.

Taylor Based Interpolation To compensate for the order reduction of the CDS scheme on non-Cartesian grids, Taylor Based Interpolation (TBI) is introduced. This scheme is based on a first order Taylor series approximation of the velocity at point P. The derivatives are approximated via coordinate transformation from a local coordinate system:

$$v_e^{\mathrm{TBI}} = v_P + \tau_e(v_E - v_P) + \tau_n(v_N - v_S) \tag{B.7}$$

in which the right hand side is an approximation to the transformation from the local to the global coordinate system. For further details of this scheme see Lehnhäuser [39]. This scheme remains second order accurate even on non-Cartesian grids, but still can introduce oscillatory solutions.

Flux Blending In order to cope with the unphysical oscillations of the CDS and TBI and at the same time diminish the numerical diffusion introduced by the UDS, a linear interpolation of both schemes is used.

$$v_e = v_e^{\mathrm{UDS}} + \gamma(v_e^{\mathrm{CDS/TBI}} - v_e^{\mathrm{UDS}}). \tag{B.8}$$

This interpolation of fluxes is called flux blending and is controlled by the coefficient γ. Using the first part of equation B.8 implicit (as part of the entries in the resulting system matrix) and the rest explicit (in the right hand side) is referred to as *Deferred Correction*. This reduces the number of diagonals in the resulting system matrix, but achieves the accuracy of the higher order scheme on convergence.

Cell Face Gradient In order to compute cell face gradients on non-Cartesian grids, a local coordinate system with coordinates ξ has to be introduced. The global coordinates are stored as functions $x = x(\xi)$. The global derivatives can be obtained by the inverse of the transformation gradient $\frac{\partial x(\xi)}{\partial \xi}$. Let F_e be the discrete counterpart of $\frac{\partial x(\xi)}{\partial \xi}$, defined on a virtual cell around the center of face e (from P to E and from se to ne) as in Figure B.2. Its inverse can be computed as:

$$F_e^{-1} = \frac{1}{\det(F_e)} \operatorname{adj}(F_e)^T \tag{B.9}$$

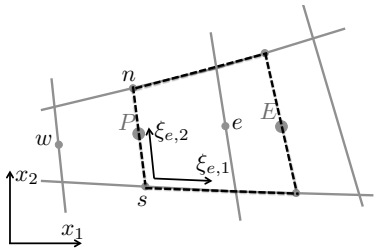


Figure B.2: Cell face centered local coordinate system

Then the diffusive term on the east cell face is interpolated as:

$$\begin{aligned}
&\mu_f(\nabla v_e + \nabla v_e^T)\delta A_e \qquad \text{(B.10)}\\
&= \underbrace{\mu_f \left[(F_e)^{-1} \begin{pmatrix} \left(\frac{v_E - v_P}{l_{P,E}}\right)^T \\ 0 \end{pmatrix} \right] \delta A_e}_{\text{implicit}}\\
&+ \underbrace{\mu_f \left[(F_e)^{-1} \begin{pmatrix} 0 \\ \left(\frac{v_{ne} - v_{se}}{l_{se,ne}}\right)^T \end{pmatrix} + \left(\frac{v_E - v_P}{l_{E,P}}, \frac{v_{ne} - v_{se}}{l_{se,ne}} \right) (F_e)^{-T} \right] \delta A_e}_{\text{explicit}}
\end{aligned}$$

In which v_{ne} is defined by $\frac{v_N + v_{NE}}{2}$. The face normal vector δA_e as shown in Figure B.3 is defined by the two adjacent grid points.$(F_e)^{-1}$ is computed using the cell centers of the neighbouring grid points N, NE, S, SE as well as P and E. The implicit part of the diffusive flux will contribute to the system matrix as the explicit will to the right hand side. For a detailed analysis of

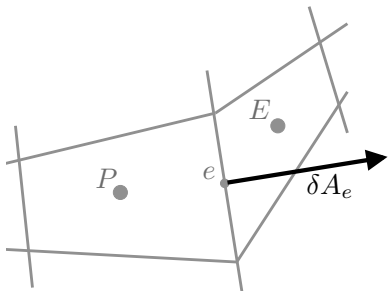


Figure B.3: Vector normal to east cell face

this discretization refer to Lehnhäuser [39]. In his work this discretization is called DABT.

Cell Center Gradient For the pressure gradient, which is located at the cell center, a cell centered local coordinate system ξ_P is introduced (see

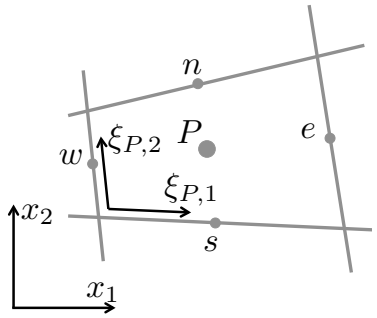


Figure B.4: Cell centered local coordinate system

Figure B.4). Let F_P be the discrete counterpart of $\frac{\partial x(\xi_P)}{\partial \xi_P}$ defined on the cell P. Then the pressure gradient can be defined as the transformation of the discrete gradient of CDS interpolated cell face values.

$$\nabla p_P \delta V_P = (F_P)^{-1} \begin{pmatrix} \frac{p_e^{\mathrm{CDS}} - p_w^{\mathrm{CDS}}}{l_{e,w}} \\ \frac{p_n^{\mathrm{CDS}} - p_s^{\mathrm{CDS}}}{l_{s,n}} \end{pmatrix} \delta V_P \tag{B.11}$$

Appendix C

Input Files

In this chapter exemplary input files for a pure structure simulation as well as a coupled FSI simulation are given. Irrelevant entries are skipped.

C.1 3D CSM 3

Below the FEAP input file for the 3D CSM 3 test case is given. In this case it is solved by a three grid multigrid solver.

```
feap * * icsm3-3level
  0 0 0 3 3 8

!3D multigrid coupling (FSI3 Benchmark)

PRINt

PARAmeters                      ! Geometry information
  x1 = 0.19
  x2 = 0.21
  y1 = 0.24899
  y2 = 0.6
  z1 = 0.0
  z2 = 0.02
  n = 2

MATErial,1
  SOLId
  FINIte                        ! Use nonlinear kinematics
  ENHAnced                      ! Use enhanced Elements
  ELAStic STVK  1.4d6 0.4       ! St. Venant-Kirchhoff E, Nu
  DENSity MASS        1d3       ! Rho
  BODY FORCe -2d3 0 0           ! Forces acting on structure
```

```
EBOUndary
  2 y1  1 1 1    ! Clamped at cylinder
  3 z1  0 0 1    ! No movement in z-direction
  3 z2  0 0 1    ! NO movement in z-direction

REGIon 1              ! Region for finest grid
BLOCk                 ! 8 node elements
   CARTesian 4*n 72*n 4*n
   BRICk 8
   1 x1 y1 z1
   2 x2 y1 z1
   3 x2 y2 z1
   4 x1 y2 z1
   5 x1 y1 z2
   6 x2 y1 z2
   7 x2 y2 z2
   8 x1 y2 z2

REGIon 2           ! Region for 2nd grid
BLOCk
   CARTesian 2*n 36*n 2*n
   BRICk 8
   1 x1 y1 z1
   ...

REGIon 3           ! Region for coarse grid
BLOCk
   CARTesian 1*n 18*n 1*n
   BRICk 8
   1 x1 y1 z1
   ...


end

TIE               ! Delete all twice and three times defined nodes

batch
 GAUSseidel INIT 5            ! Use 5 sweep Gauss Seidel solver
  DEACtivate,,2               ! Deactivate 2nd region
  DEACtivate,,1
  ACTIvate,,2                 ! Activate 2nd region
```

```
  MULTigrid BUILdmatrix 2 3   ! BUILdmatrix assmebles the matrix for
  DEACtivate,,3               ! restriction & prolongation. It defines
  ACTIvate,,1                 ! the matrix in between the 'old' active
  MULTigrid BUILdmatrix 1 2   ! region and the last activated region.
  DEACtivate,,2
  TRANsient NEWMark           ! Use Newmark beta scheme
  DT,,0.005                   ! Time step size
  LOOP TIME 400               ! Loop over number of time steps
  TIME                        ! Increment time
  LOOP VCYCLe 200
       PRINT
       TANG                   ! Build Jacobian
       FORM                   ! Build residual vector
       SOLVE                  ! Solve the system
       NOPRint
       FORM                   ! Build residual for coarse grid RHS
       MULTigrid RESTrict 2 1 ! Restrict RHS, disp, vel, acce
! When FEAP hits the convergence criterion it exits the most inner loop it
! is in. This is not supposed to happen on coarse grids.
            LOOP NEWTon 1     ! Auxiliary loop
            TANG              ! Build Jacobian
            FORM              ! Build residual vector
            MULTigrid SAVE 1  ! Put both of the above in the RHS
            FORM              ! Build residual vector
            MULTigrid MAKE 1  ! Add RHS from fine grid to residual vector
            SOLVe             ! Solve system
            NEXT NEWTon       ! End loop
            FORM              ! Build residual vector for RHS
            MULTigrid MAKE 1  ! Add RHS from fine grid to residual vector
            MULTigrid RESTrict 3 2    ! as above ...
                 GAUSseidel DIREct    ! Use LU decomp. solver
                 LOOP NEWTon 1
                 TANG
                 FORM
                 MULTigrid SAVE 2
                 FORM
                 MULTigird MAKE 2
                 SOLVe
                 NEXT NEWTon
                 GAUSseidel ITERative ! Use Gauss Seidel solver
            MULTigrid PROLongate 2 3  ! Prolongate disp, vel, acce
            LOOP NEWTon 1
            TANG
            FORM
```

```
            MULTigrid MAKE 1  ! Add the RHS from 'MULT SAVE 1' to residual
            SOLVe
            NEXT NEWTon
       MULT PROLongate 1 2
       TANG
       FORM
       SOLVe
  NEXT VCYCLe                 ! End loop V-cycle
  DISP NODE (x1+x2)/2 y2 z1   ! Write displacmenet into output.
NEXT TIME                     ! End loop time
end

stop
```

C.2 3D FSI 3

C.2.1 FASTEST input file

Below the FASTEST input file for the 3D FSI 3 test case is given. In this case it is solved by a two grid MG CPL implementation.

```
...
### lcalc
vel time         ;[vel][turb|visles][temp][time]
                       ;turb:   use a RANS turbulence model
                       ;visles: perform a Large Eddy Simulation
...
### grid levels
2 2                      ;no. of coarse / fine grid
### grid level for tecplot output
2
...
### convergence criterion
1                ;1 --> sum of all residuals, 2 --> max. residual
### residuum limits
1.e-7  1.e+10
### interpolation method
tbi                           ;[cds|tbi]
### flux blending
1.0 0.0 0.0 0.0 0.0 0.0 ;1. Grid vel k eps temp zeta f
1.0 0.0 0.0 0.0 0.0 0.0 ;2. Grid
1.0 0.0 0.0 0.0 0.0 0.0 ;3. Grid
### underrelaxation
0.5 0.5 0.5 0.5 0.9 0.9 0.5 0.3 0.9 0.5 0.5 ;1. Grid u v w p k eps vis den t ze
```

```
0.5 0.5 0.5 0.5 0.9 0.9 0.5 0.3 0.9 0.5 0.5 ;2. Grid
0.7 0.7 0.7 0.5 0.9 0.9 0.5 0.3 0.9 0.5 0.5 ;3. Grid
### sipsol
0.92                                ;alfa
0.5 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 ;sor: u v w p k eps t zeta f
2 2 2 10 1 1 1 1 1                  ;nsw: u v w p k eps t zeta f
### multigrid cycle definition
20
2 14 2
### number of multigrid cycles
100 100
### time discretization
sofi                    ;discretization method [fofi|sofi|crni]
10000 0.002                ;no of timesteps,size of timestep
...
### fluid regions
t                       ;for each flow region: t - fluid, f - solid
### moving grids
y                       ;lmvgr
### fluid structure interaction
y y y y                 ;lfsi  lfsiread  lfsiwrite lfsisend
-3                      ;extrapolation of forces: [0|1|2|3]-order, [-1] = none
2 12 2                  ;max. number of outer iterations
2                       ;coupling interface (0=none, 1=GRISSLi, 2=MpCCI)
1                       ;interpolation scheme (0=NNB, 1=CONS)
0.6 0.0                 ;underrel. parameter for structural distortions, aitken damping
1.e-5 1.e-5 1.e-5        ;fsi-residuum limit (x,y,z-direction)
-3                      ;extrpolation of displacements: [1|2|3]-order, [-1] = none
1.e-4                   ;convergence criterion for the TFSI
0.0 ;underrelaxation parameter for the TFSI
### grid distortions
y ;grid smoother [y/n]
12 1          ;number of blocks to be distorted; dist_norm
n ;use detailed grid distortion input matrix [y/n]
5  2  4 0 0 3 1 4  2 0.0   ;simple version [n]
6  2  4 1 0 3 0 4  2 0.0
7  2  0 1 4 4 3 1  2 0.0
8  2  0 1 4 4 3 1  2 0.0
9  2  1 1 4 4 3 0  2 0.0
10 2  1 1 4 4 3 0  2 0.0
13 2  3 1 4 4 8 3  2 0.5
11 2  0 1 4 4 8 1  2 0.5
15 2  1 1 4 4 8 0  2 0.5
14 2  3 1 4 4 0 3  2 0.0
```

```
12 2  0 1 4 4 0 1  2 0.0
16 2  1 1 4 4 0 0  2 0.0
### high mem fast calculation
1                       ;switch for fast calculation with higher mem usage
### mpcci input data
1                       ;mycode
2                       ;remoteCodeId
10 15                   ;meshId                  from *.cci file
2                       ;nQuantityIds
11                      ;QuantityIds(1),localMeshIds(1)
12                      ;QuantityIds(2),localMeshIds(2)
...
### draglift
y                       ;draglift calculation on/off
12                       ;number of faces where draglift is to be calc.
2 3 4 5 5 6 6 7 8 9 10 13   ;blocknumbers of blocks w. draglift faces
4 4 3 3 5 3 2 6 6 1  1  2   ;
0.004                    ;reference area for D,L coeff.
0.2                     ;reference velocity (u_inf) for D,L coeff.
1.0e3                    ;reference density for D,L coeff.
90                       ;rotation angle in degrees
0 0 1                   ;vector components nx,ny,nz that define rotation axis (
...
### pressure reference point
t 0.0 2.5 0.0 t 0d0 ;[lcpref][xpref][ypref][zpref][lpfix][fpref]
; move[t/f] move where?  fix[t/f] fix at?
; one line per fluid region(fr)
### static pressure exit boundary condition
f
```

C.2.2 FEAP input file

Below the corresponding FEAP input file to the 3D FSI 3 test case solved by an two grid MG CPL implementation is given.

```
feap * * ifsi3
  0 0 0 3 3 8

!3D multigrid coupling (3D FSI3 Testcase)

PRINt

PARAmeters                      !  Geometry information
  x1 = 0.19
```

```
  x2 = 0.21
  y1 = 0.24899
  y2 = 0.6
  z1 = 0.0
  z2 = 0.02

MATErial,1
  SOLId
  FINIte                          ! Use nonlinear kinematics
  ELAStic STVK  5.6e6 0.4         ! St. Venant-Kirchhoff E, Nu
  DENSity DATA        1e3         ! Rho


EBOUndary                         ! Unset boundary conditions are v. Neumann type
  2 y1  1 1 1                     ! Clamped at cylinder
  3 z1  0 0 1                     ! No movement in z-direction
  3 z2  0 0 1                     ! No movement in z-direction


REGIon 1                          ! Region for finest grid
BLOCk                             ! 8 node elements
   CARTesian 8 144 8
   BRICk 8
   1 x1 y1 z1
   2 x2 y1 z1
   3 x2 y2 z1
   4 x1 y2 z1
   5 x1 y1 z2
   6 x2 y1 z2
   7 x2 y2 z2
   8 x1 y2 z2

REGIon 2                          ! Region for coarse grid
BLOCk
   CARTesian 4 72 4
   BRICk 8
   1 x1 y1 z1
   ...


MPCCi                             ! MpCCI interface input
  GRIDlevel 1                     ! Finest grid first
  MESHid 20                       ! MpCCI mesh ID
  PARTid 25                       ! MpCCI partition ID
```

```
NODEs                          ! Nodes lying on the interface
1 0 0 0 0
9 0 0 0 0
...
11745 0 0 0 0

ELEMents                       ! Interface elements (4 node elements)
1 0 9 18 1323 1314
2 0 18 27 1332 1323
...
2368 0 10440 10439 11744 11745

  GRIDlevel 2                  ! MpCCI input for coarse grid
  MESHid 23                    ! MpCCI mesh ID
  PARTid 26                    ! MpCCI partition ID
  NODEs                        ! Nodes lying on the interface
11746 0 0 0 0                  ! Actual numbering without 'TIE' command
11750 0 0 0 0
...
13570 0 0 0 0

ELEMents                       ! Interface Elements
1 0 11750 11755 12120 12115 ! Element numbering starts at 1
2 0 11755 11760 12125 12120
...


end

TIE REGIon 1 2                 ! Delete nodes in region 2
                               ! rather than region 1.

batch
  MPCCi INIT                   ! Initialize MpCCI interface
  GAUSseidel INIT 5            ! Use 5 sweeps Gauss Seidel solver
  TRANsient NEWMark 0.3 0.6    ! Newmark time stepping beta=0.3, gamma=0.6
  RESTart restart              ! Read restart file
  DEACtivate,,1                ! Buildmatrix assembles the restrction &
  ACTIvate,,1                  ! prolongation matrix from the current
  MULTigrid BUILdmatrix 1 2    ! active region to the last activated region
  DEACtivate,,2                ! Starting computations on region 1
  DT,,0.002                    ! Time step size
  LOOP TIME,10000              ! Loop over time
  TIME                         ! Increment time step
```

```
    LOOP VCYCle 300             ! Loop over V-cycles
      LOOP FSIIteration 2       ! FSI iterations presmoothing
        RECV FORCE 1            ! Receive force from FASTEST on region 1
! When FEAP hits the convergence criterion it exits the most inner loop
! it is in. This is not supposed to happen, as convergence is handled
! by FASTEST. Thus the auxiliary loops.
        LOOP NEWTon 1           ! Auxiliary loop
        TANG                    ! Build Jacobian
        FORM                    ! Build residual vector
        SOLVe                   ! Solve the system
        NEXT NEWTon
        SEND DISPLacement 1     ! Send disp to FASTEST on region 1
      NEXT FSIiteration         ! End loop
      FORM                      ! Build residual for coarse grid RHS
      MULTigrid RESTrict  2 1   ! Restrict RHS, disp, vel, acce
!       GAUSseidel DIREct       ! Only used for steady simulations
        LOOP NEWTon 1           ! Auxiliary loop
        RECV FORCE 2            ! Receive force from FASTEST on region 2
        FORM                    ! Build residual vector
        MULTigrid SAVE 1        ! Put the residual in the RHS
        NEXT NEWTon
        SEND DISPLacement 2     ! Send disp to FASTEST on region 2
! The 6 lines above describe the additional exchange on coarse grids
! without solving
        LOOP FSIiteration 12    ! Smoothing iterations on coarse grid
          LOOP NEWTon 1         ! Auxiliary loop
          TANG                  ! Build Jacobian
          RECV FORCE 2          ! Receive force from FASTEST on region 2
          FORM                  ! Build residual vector
          MULTigrid MAKE 1      ! Add RHS from fine grid to residual
          SOLVe                 ! Solve the system
          NEXT NEWTon
          SEND DISPLacement 2   ! Send disp to FASTEST on region 2
     NEXT FSIiteration       ! End loop
!       GAUSseidel ITERative    ! Only used for steady calculations
      MULTigrid PROLongate 1 2  ! Prolongate disp, vel, acce
      LOOP FSIiteration 2       ! FSI iterations postsmoothing
        RECV FORCE 1            ! Receive force from FASTEST on region 1
        LOOP NEWTon 1           ! Auxiliary loop
        TANG                    ! Build Jacobian
FORM                    ! Build residual vector
SOLVe                   ! Solve system
NEXT NEWTon
        SEND DISPLacement 1     ! Send disp to FASTEST on region 2
```

```
      NEXT FSIiteration         ! End loop
    NEXT VCYCle
    DISPLacement NODE (x1+x2)/2 y2 (z1+z2)/2 ! Write displacement to output
    SAVE restart                ! Write restart file
  NEXT TIME                     ! Next time step
MpCCi END                       ! Finalize MpCCI
end

stop
```

C.2.3 MpCCI input file

Below the corresponding MpCCI input file for the 3D FSI 3 test case and the MG CPL implementation is given.

```
code fhp
    id_string = "fhp";
    mesh( no = 10);
    force1( no = 11, loc = elem, dim = vector, type = flux );
    displ1( no = 12, loc = node, dim = vector, type = field );
end


code feap
    id_string = "feap";
    mesh( no = 20);
    force( no = 21, loc = node, dim = vector, type = flux );
    displ( no = 22, loc = node, dim = vector, type = field );
end


quantities
    feap.force = fhp.force1;
    fhp.displ1 = feap.displ;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%
% CONTROL-BLOCK %
%%%%%%%%%%%%%%%%%
%
% Uncomment this for an input to the mpcci visualizer showing forces
% and displacements on the interface
%
```

```
%control
%  tracefile( name = "test.ccv",
%      close_after_writing = on,       % default: off
%      implicit_coupling_steps = on,   % default: on
%      trace_comm_values = on,         % default: on
%      trace_mesh_values = on,         % default: off
%      trace_mesh_change_values = off, % default: on
%      format = hdf5                  % default: hdf5
% );
%
%end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

contact
    default_alg(
type = EE,
matching_criterion = intersection(
           perform_tests = true,
           rejection     = 0.001,
           scale_result  = true,
           max_projection_angle    = 30,
           max_projection_distance = -1.0
        )
    );
    alg displacements(
     type = PE,
    matching_criterion = minimal_distance (
           theta1 = 0.0,
           theta2 = 1.0,
           theta3 = 0.0,
           rejection = 1.0e-3,
           inside_only = false
        )
    );
    alg forces(
     type = EE,
        matching_criterion = intersection (
           perform_tests = true,
           rejection     = 0.001,
           scale_result  = true,
           max_projection_angle    = 30,
           max_projection_distance = -1.0
        )
```

```
    );
    mesh_pair (FASTEST3D/50,FEAP/115) : displacements;
    mesh_pair (FASTEST3D/10,FEAP/23)  : forces;
    mesh_pair (FASTEST3D/75,FEAP/100) : displacements;
    mesh_pair (FASTEST3D/15,FEAP/20)  : forces;
end

switches
    output_level=0;
end


jobs
    FASTEST3D = fhp (
        pwd = "/path/to/fastest/project/directory",
        exec = "unused",
        nprocs = 4
    );

    FEAP = feap (
    pwd = "/path/to/feap/project/directory",
    exec = "unused",
    nprocs = 1
);
end
```

Bibliography

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[2] K.J. Bathe, H. Zhang, and S. Ji. Finite element analysis of fluid flows fully coupled with structural interactions. *Computers and Structures*, 72(1):1–16, 1999.

[3] Y. Bazilevs, M.-C. Hsu, J. Kiendl, R. Wchner, and K.-U. Bletzinger. 3d simulation of wind turbine rotors at full scale. Part II: Fluidstructure interaction modeling with composite blades. *International Journal for Numerical Methods in Fluids*, 65(1-3):236–253, 2011.

[4] Y. Bazilevs, M.-C. Hsu, Y. Zhang, W. Wang, X. Liang, T. Kvamsdal, B. Brekken, and J. G. Isaksen. A fully-coupled fluid-structure interaction simulation of cerebral aneurysms. *Computational Mechanics*, 2010.

[5] T. Belytschko, W. K. Liu, and B. Moran. *Nonlinear Finite Elements for Continua and Structures*. Wiley, 2000.

[6] R. B. Bird, W. E. Stewart, and E. N. Lightfoot. *Transport Phenomena*. Wiley, 2 edition, December 2006.

[7] A. Brandt. Multi-level adaptive technique (mlat) for fast numerical solution to boundary value problems. In H. Cabannes and R. Temam, editors, *Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics*, volume 18 of *Lecture Notes in Physics*, pages 82–89. Springer Berlin / Heidelberg, 1973. 10.1007/BFb0118663.

[8] A. Brandt. Multi-Level Adaptive Solutions to Boundary-Value Problems. *Mathematics of computation*, 31(138):333–390, 1977.

[9] M. Breuer and M. Münsch. FSI of the turbulent Flow around a Swiveling Flat Plate Using Large-Eddy Simulation. In S. Hartmann, A. Meister, M. Schäfer, and S. Turek, editors, *International Workshop on Fluid-*

Structure Interaction - Theory, Numerics and Applications, pages 31–43, 2008.

[10] M. Breuer and M. Münsch. LES meets FSI - Important Numerical and Modelling Aspects. In V. Armenio et al., editors, *Direct and Large Eddy Simulations VII*, ERCOFTAC. Springer Science+Business Media B.V., 2010.

[11] W.L. Briggs and S.F. McCormick. *A Multigrid Tutorial*, volume 72. Society for Industrial Mathematics, 2000.

[12] H.-J. Bungartz and M. Schäfer, editors. *Fluid-Structure Interaction*, volume 53 of *Lecture Notes in Computational Science and Engineering*. Springer Berlin Heidelberg, 2006.

[13] D. E. Davies and D. J. Salmond. Calculation of the volume of a general hexahedron for flow predictions. *AIAA Journal*, 23:954–956, June 1985.

[14] J. Degroote, K.-J. Bathe, and J. Vierendeels. Perfromance of a new partitioned procedure versus a monolithic procedure in fluid-structure interaction. *Computers and Structures*, 2009.

[15] I. Demirdzic and M. Peric. Space conservation law in finite volume calculations of fluid flow. *Int. Journal for Numerical Methods in Fluids*, 8, 1988.

[16] J. Donea. *Computational Methods for Transient Analysis, Computational Methods in Mechanics*, volume 1, chapter Arbitrary lagrangian Eulerian Methods. Elsevier, North-Holland, 1983.

[17] Y. Du. *Numerical Simulation of Mechanical and Thermal Fluid-Structure Interaction in Labyrinth Seals*. PhD thesis, Technische Universität Darmstadt, 2010.

[18] T. Dunne. *Adaptive Finite Element Approximation of Fluid-Struture Interaction Based on Eulerian and Arbitrary Lagrangian-Eulerian Variational Formulations*. PhD thesis, Ruprecht-Karls Universität Heidelberg, 2007.

[19] F. Durst and M. Schäfer. A Parallel Blockstructured Multigrid Method for the Prediction of Incompressible Flows. *Int. J. for Numerical Methods in Fluids*, 22:549–565, 1996.

[20] Technische Universität Darmstadt Fachgebiet Numerische Berechnungsverfahren im Maschinenbau. FASTEST User Manual, 2005.

[21] C. Farhat, P. Geuzaine, and C. Grandmont. The discrete geometric conservation law and the nonlinear stability of ale schemes for the solution of flow problems on moving grids. *Journal of Computational Physics*, 174(2):669 – 694, 2001.

[22] C. A. Felippa and K. C. Park. Staggered transient analysis procedures for coupled mechanical systems: formulation. *Computer Methods in Applied Mechanics and Engineering*, 24(1):61–111, 1980.

[23] J.H. Ferziger and M. Perić. *Computational Methods for Fluid Dynamics*, volume 2. Springer Berlin, 1999.

[24] Fraunhofer Institute - Algorithms and Scientific Computing. *MpCCI 3.0.6 Documentation*, 2007.

[25] F. Gauß. *Strategien zur adaptiven Gitterverfeinerung für Strömungsöser*. PhD thesis, Technische Universität Darmstadt, 2010.

[26] A. Gerstenberger and W. A. Wall. An extended finite element method/lagrange multiplier based approach for fluid-structure interaction. *Computer Methods in Applied Mechanics and Engineering*, 197(19-20):1699 – 1714, 2008.

[27] H. Guillard and C. Farhat. On the significance of the geometric conservation law for flow computations on moving meshes. *Computer Methods in Applied Mechanics and Engineering*, 190(11-12):1467 – 1482, 2000.

[28] W. Hackbusch. *Multi-Grid Methods and Applications*. Springer, 1985.

[29] M. Heck. *Mehgitterverfahren zur effizienten numerischen Simulation von Fluid-Struktur-Wechselwirkungen*. PhD thesis, Technische Universität Darmstadt, 2008.

[30] M. Heil. An efficient solver for the fully coupled solution of large-displacement fluid-structure interaction problems. *Computer Methods in Applied Mechanics and Engineering*, 193(1-2):1 – 23, 2004.

[31] R. Herzog and E. Sachs. Preconditioned Conjugate Gradient Method for Optimal Control Problems with Control and State Constraints. *SIAM Journal on Matrix Analysis and Applications*, 31(5), 2010.

[32] H. M. Hilber, T. J. R. Hughes, and R. L. Taylor. Improved numerical dissipation for time integration algorithms in structural dynamics. *Earthquake Engineering and Structural Dynamics*, 5:283.292, 1977.

[33] C. W. Hirt, A. A. Amsden, and J. L. Cook. An arbitrary lagrangian-eulerian computing method for all flow speeds. *Journal of Computational Physics*, 14(3):227 – 253, 1974.

[34] J. Hron and S. Turek. A Monolithic FEM/Multigrid Solver for an ALE Formulation of Fluid-Structure Interaction with Applications in Biomechanics. In H.-J. Bungartz and M. Schäfer, editors, *Fluid-Structure Interaction*, volume 53 of *Lecture Notes in Computational Science and Engineering*, pages 146–170. Sppringer, 2006.

[35] B. Hübner, E. Walhorn, and D. Dinkler. A monolithic approach to fluid-structure interaction using space-time finite elements. *Computer Methods in Applied Mechanics and Engineering*, 193(23-26):2087 – 2104, 2004.

[36] T. J. R. Hughes, W. K. Liu, and T. K. Zimmermann. Lagrangian-Eulerian finite element formulation for incompressible viscous flows. *Computer Methods in Applied Mechanics and Engineering*, 29(3):329 – 349, 1981.

[37] U. Küttler and W. Wall. Fixed-point fluid-structure interaction solvers with dynamic relaxation. *Computational Mechanics*, 43(1):62–72, 2008.

[38] U. Küttler and W. Wall. Vector Extrapolation for Strong Coupling Fluid-Structure Interaction Solvers. *Journal of Applied Mechanics*, 76(2):021205, 2009.

[39] T. Lehnhäuser. *Eine effiziente numerische Methode zur Gestaltsoptimierung von Strömungsgebieten*. PhD thesis, Techn. Univ. Darmstadt, 2003.

[40] T. Lehnhäuser and M. Schäfer. Improved linear interpolation practice for finite-volume schemes on complex grids. *Int. Journal for Numerical Methods in Fluids*, 38:625–645, 2002.

[41] Lehrstuhl für Strömungsmechanik Universität Erlangen. *FASTEST Parallel Multigrid Solver for Flows in Complex Geometries*.

[42] F. Liu, J. Cai, and Y. Zhu. Calculation of Wing Flutter by a Coupled Fluid-Structure Method. *Journal of Aircraft*, 38(2), 2001.

[43] H. G. Matthies and J. Steindorf. Partitioned strong coupling algorithms for fluid-structure interaction. *Computers & Structures*, 81(8-11):805 – 812, 2003.

[44] C. Michler, S.J. Hulshoff, E.H. van Brummelen, and R. de Borst. A monolithic approach to fluid-structure interaction. *Computers & Fluids*, 33(5-6):839 – 848, 2004.

[45] I. D. Parsons and J. F. Hall. The multigrid method in solid mechanics: Part ialgorithm description and behaviour. *International Journal for Numerical Methods in Engineering*, 29(4):719–737, 1990.

[46] S. Piperno. ExplicitImplicit fluidstructure staggered procedures with a structural predictor and fluid subcycling for 2d inviscid aeroelastic simulations. *International Journal for Numerical Methods in Fluids*, 25(10):1207–1226, 1997.

[47] P. Pironkov. *Numerical Simulation of Thermal Fluid-Structure Interaction*. PhD thesis, Technische Universität Darmstadt, 2010.

[48] M. Razzaq, H. Damanik, J. Hron, A. Ouazzi, and S. Turek. FEM multigrid techniques for fluid-structure interaction with application to hermodynamics. *Applied Numerical Mathematics*, 2011. accepted.

[49] C.M. Rhie and W.L.Chow. Numerical Study of the Turbulent Flow Past an Airfoil with Trailing Edge Seperation. *AIAA Journal*, 21(11):1525–1532, 1983.

[50] J. W. Ruge and K. Stueben. Algebraic multigrid. *Multigrid Methods*, 3(13):73–130, 1987.

[51] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, 2003.

[52] S. Sachs, D. Sternel, and M. Schäfer. Implicit Partitioned Coupling with Global Multigrid in Fluid-Structure Interaction. In J.C.F. Pereira and A. Sequeira, editors, *V European Conference on Computational Fluid Dynamics*. ECCOMAS CFD, 2010.

[53] S. Sachs, M. Streitenberger, D. Sternel, and M. Schäfer. Extrapolation Methods for Accelerating Unsteady Partitioned Fluid-Structure Interaction Simulations. *International Journal of Multiphysics*. accepted.

[54] M. Schäfer. *Numerik im Maschinenbau*. Springer, 1999.

[55] M. Schäfer, H. Lange, and M. Heck. Implicit Partitioned Fluid-Structure Interaction Coupling. In *Résumé pour le 1er colloque GDR interactions fluide-structure*, pages 31–38, 2005.

[56] M. Schäfer, D. C. Sternel, G. Becker, and P. Pironkov. Efficient Numerical Simulation and Optimization of Fluid-Structure Interaction. In *Fluid Structure Interaction II: Modelling, Simulation, Optimization*, volume 73, pages 131–158. Springer, 2010.

[57] F. Schmid. *Zum Einsatz algebraischer Mehrgitterverfahren bei der Simulation inkrementeller Umformverfahren*. PhD thesis, Technische Universität Darmstadt, 2008.

[58] G. Sieber. *Fluid-Structure Interaction Using Loose Coupling Methods*. PhD thesis, Technische Universität Darmstadt, 2002.

[59] D. Sternel, M. Schäfer, M. Heck, and S. Yigit. Efficiency and accuracy of fluid-structure interaction simulations using an implicit partitioned approach. *Computational Mechanics*, 2008.

[60] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer, 2002.

[61] H. L. Stone. Iterative Solution of Implicit Approximations of Multidimensional Partial Differential Equations. *SIAM Journal on Numerical Analysis*, 5(3), 1968.

[62] M. Streitenberger. Einfluss von Extrapolation auf die Konvergenz von Fluid-Strucktur-Interaktion. Bachelor's Thesis, 2010.

[63] P. Le Tallec and J. Mouro. Fluid structure interaction with large structural displacements. *Computer Methods in Applied Mechanics and Engineering*, 190(24-25):3039 – 3067, 2001.

[64] R. L. Taylor. *FEAP - A Finite Element Analysis Program - Version 8.2 Theory Manual*. Department of Civil and Environmental Engineering, University of Berkeley, 2008.

[65] R. L. Taylor. *FEAP - A Finite Element Analysis Program - Version 8.2 User Manual*. Department of Civil and Environmental Engineering, University of Berkeley, 2008.

[66] T. E. Tezduyar, M. Schwaab, and S. Sathe. Sequentially-Coupled Arterial Fluid-Structure Interaction (SCAFSI) technique. *Computer Methods in Applied Mechanics and Engineering*, 198(45-46):3524 – 3533, 2009.

[67] P. D. Thomas and C. K. Lombard. Geometric Conservation Law and Its Application to Flow Computations on Moving Grids. *AIAA Journal*, 17:1030–1037, 1979.

[68] U. Trottenberg, C. W. Oosterlee, and Anton Schüller. *Multigrid*. Academic Press, 2001.

[69] S. Turek and J. Hron. Proposal for Numerical Benchmarking of Fluid-Structure Interaction between an Elastic Object and Laminar Incompressible Flow. In H.-J. Bungartz and M. Schäfer, editors, *Fluid-Structure Interaction*, volume 53 of *Lecture Notes in Computational Science and Engineering*, pages 371–385. Springer Berlin Heidelberg, 2006.

[70] E. H. van Brummmelen, C. Michler, and R. de Borst. Interface-GMRES(R) Acceleration of Subiterations for Fluid-Structure-Interaction problems. Technical report, Delft University of Technology, Faculty of Aerospace Engineering, 2005.

[71] J. Vierendeels, L. Lanoye, J. Degroote, and P. Verdonck. Implicit coupling of partitioned fluid-structure interaction problems with reduced order models. *Computers & Structures*, 85(11-14):970–976, 2007.

[72] T. Waclawczyk. *Numerical Modelling of Free Surface Flows in Ship Dynamics*. PhD thesis, Polish academy of Sciences, 2007.

[73] E. Walhorn, A. Kölke, B. Hübner, and D. Dinkler. Fluid-structure coupling within a monolithic model involving free surface flows. *Computers & Structures*, 83(25-26):2100 – 2111, 2005.

[74] W. A. Wall, A. Gerstenberger, P. Gamnitzer, C. Förster, and E. Ramm. Large Deformation Fluid-Structure Interaction - Advances in ALE Methods and New Fixed Grid Approaches. In H.-J. Bungartz and M. Schäfer, editors, *Fluid-Structure Interaction*, volume 53 of *Lecture Notes in Computational Science and Engineering*, pages 195–232. Springer Berlin Heidelberg, 2006.

[75] P. Wesseling. *An Introduction to Multigrid Methods*. R. T. Edwards, 2004.

[76] G. Wittum. Linear iterations as smoothers in multigrid methods: Theory with applications to incomplete decompositions. *IMPACT of Computing in Science and Engineering*, 1(2):180 – 215, 1989.

[77] W. L. Wood. *Practical time-stepping schemes*. Clarendon Press, 1990. B0940.

[78] S. Yigit. *Phänomene der Fluid-Struktur-Wechselwirkung und deren numerische Berechnung*. PhD thesis, Technische Universität Darmstadt, 2008.

[79] S. Yigit, M. Heck, D. C. Sternel, and M. Schäfer. Efficiency of Fluid-Structure Interaction Simulations with Adaptive Underrelaxation and Multigrid Acceleration. *International Journal of Multiphysics*, 1(1):85–99, 2007.

Stephen Sachs

Persönliche Daten

Name	Sachs, Stephen Markus
Adresse	Frankfurter Strasse 29, 64293 Darmstadt, Germany
Telefon	+49 (0)178 4787 339
E-Mail	stephenmsachs@gmail.com
Staatsangehörigkeiten	Deutschland, USA, Österreich
Geburtstag	12.03.1984

Ausbildung

Mai 2008 - Feb. 2012
Dr. rer. nat. Maschinenbau, *Fachgebiet Numerische Berechnungsverfahren im Maschinenbau und Graduate School of Computational Engineering*, Technische Universität Darmstadt.
Thesis: "Multigrid Methods applied to Fluid-Structure Interaction"
Note: 1.0

Apr. 2003 - März 2008
Diplom Mathematik, *Nebenfach Informatik*, Universität Trier und Technische Universität Darmstadt.
Thesis: "Form Optimization of the Navier Stokes Equations using Free Form Deformation"
Note: 1.3

Juli 1994 - März 2003
Abitur, Max Planck Gymnasium, Trier.
Hauptfächer: Mathematik, Physik, Englisch
Note: 2.8

Juli 2000 - Dez. 2000
High School, Blacksburg, Virginia.
Hauptfächer: Mathematik, Englisch

Auszeichnung

Mai 2008 - Apr. 2011
Dr. rer. nat. Stipendium, *Graduate School of Computational Engineering*, Exzellenzinitiative des Bundes und der Länder.

Publikationen

Bücher

S. Sachs, *Multigrid Methods applied to Fluid-Structure Interaction*, Dissertation, Technische Universität Darmstadt, erscheint 2012.

S. Sachs, *Form Optimization of the Navier Stokes Equations using Free Form Deformation*, Diplomarbeit, Technische Universität Darmstadt, 2008.

Journals

E. Sachs, **S. Sachs**, *Nonmonotone Line Searches for Optimization Algorithms*, Control and Cybernetics, erscheint 2012.

S. Sachs, M. Streitenberger, D. Sternel, M. Schäfer, *Extrapolation Methods for Accelerating Unsteady Partitioned Fluid-Structure Interaction Simulations*, International Journal of Multiphysics, 5(4), pp.287-297, 2012.

Proceedings

S. Sachs, D. Sternel, M. Schäfer, *Implicit Partitioned Coupling with Global Multigrid in Fluid-Structure Interaction*, European Conference on Computational Fluid Dynamics (ECCOMAS CFD), 2010.

Präsentationen

Juni 2010

ECCOMAS CFD, *Implicit Partitioned Coupling with Global Multigrid in Fluid-Structure Interaction*, Lissabon, Portugal.

Apr. 2012

ICAM Special Lecture, *Multigrid Methods applied to Fluid-Structure Interaction*, Virginia Tech, USA.

Darmstadt, den 15. März, 2012