A Fluid-Structure-Interaction tool by coupling of existing codes

Pablo Mosquera Michaelsen
Institute of Fluid Machinery
Karlsruhe Institute of Technology
Kaiserstrasse 12
Karlsruhe, Germany
mosquera@kit.edu

Balázs Pritz
Institute of Fluid Machinery
Karlsruhe Institute of Technology
Kaiserstrasse 12
Karlsruhe, Germany
pritz@kit.edu

Martin Gabi
Institute of Fluid Machinery
Karlsruhe Institute of Technology
Kaiserstrasse 12
Karlsruhe, Germany
gabi@kit.edu

ABSTRACT

Fluid-Structure-Interaction (FSI), as a sub-discipline of computational mechanics, has been gaining relevance since the growth in clusters capacity has made it possible to simulate high resolution models. Although some commercial tools already present certain capabilities for coupled simulations, the lack of efficiency of these general purpose programs is still an issue. Moreover, the high cost of commercial licenses - on a per processor basis - hampers the computation of high resolution models for academic research. Instead, a coupled software solution that resorts to well established existent programs proves a good alternative to preserve the value of decades-long development and associated know-how. However, since such codes were mostly conceived for standalone run, an elegant software implementation is not easily achieved.

In this work our CFD code *SPARC* has been coupled with the open-source structural solver *CalculiX* by means of the in-house developed software packet *FSiM*. *FSiM* stands for *Fluid-Structure-Interaction Simulation Manager* and is in charge of the communication between the fluid and structure solver using the Dynamic Process Management of the MPI-2 standard. This approach facilitates that the parallelization strategies of both part-solvers be used with minimal modifications and no risk of mutual interference. Results of a simple two-dimensional test case of the panel flutter problem are presented to show the capabilities of the new coupling tool.

Keywords

Fluid-Structure-Interaction, Dynamic Process Management, partitioned approach

1. INTRODUCTION

The analysis of Fluid-Structure-Interaction (FSI) with numerical methods comprises a very broad area of research which addresses physical phenomena of different fields. Reliable FSI solvers are demanded in areas as diverse as aeroelasticity [3], civil engineering [7], fabric-like structures, such as windsocks, parachutes and sails [12], turbomachinery [11], hemodynamics [16] and noise control in flow ducts [2].

At the Institute of Fluid Machinery (FSM) of the Karlsruhe Institute of Technology the development of methods to analyse vibrations and stability of turbomachinery is of great interest. This has been typically done by means of linear formulations. More accurate analysis, however, requires to include non-linear effects such as large deformation of the blades/vanes or damping of the surrounding fluid. Such phenomena can be adequately described by means of Computational Fluid-Structure-Interaction in time-domain [6].

There are at the present time a number of commercial tools available offering capabilities for Fluid-Structure-Interaction. While these commercial programs are a reasonable choice for the industry (where robust, closed packages are required) they present a number of drawbacks for the academic research: no improvements or tests can be made in the core routines; the cost of licenses effectively limits the number of cores and therefore the size or resolution of the examined models; stabilization procedures are often applied to the basic algorithms to achieve convergence at all costs making it difficult to separate numerical from physical effects. Instead, an in-house development was judged to be more convenient in the long run.

The Institute of Fluid Machinery has been developing the code *SPARC* (Structured PArallel Research Code) for more than 15 years and an extension to solve coupled problems eventually became a natural need. We feel our case represents that of many scientific groups where decades-long code development is carried out by different generations of engineers or physicists whose expertise lies on the physics and numerical methods rather than on computational sciences. Therefore, although the source code of such simulation tools usually possesses a rigid and non-modular structure they are deemed very valuable and must be kept at the cost of a not so elegant implementation. In this context the use of the Dynamic Process Management features of MPI-2 provide the best available solution to couple such existent codes. Although similar developments have been performed in the past [15], [8] the implementation details are scarcely documented. We believe this work provides a good example of the issues that any scientific group would face when seeking to extend the functionality of its code.

This paper is organized as follows: In section 2 the mathematical model governing the Fluid-Structure-Interaction is exposed. Section 3 describes the general strategy devised to achieve the fluid-structure coupling while Section 4 deals with the MPI implementation along with current issues and shortcomings. Finally Section 5 presents the test case used to verify the development.

2. GOVERNING EQUATIONS

2.1 Fluid Model

The main *SPARC* module solves the full-compressible Navier-Stokes equations through the Finite Volume Method in body-fitted, block-structured meshes. Development is also being carried out in the areas of incompressible flows and immersed boundary method. Implicit time integration is performed by means of backward second order finite differences and dual-time stepping whereas the 4-stage Runge-Kutta method is applied for explicit integration. Acceleration methods like local time stepping and full-multigrid methods are employed on a standard basis. Since body-fitted meshes are used the Navier-Stokes equations are written in the Arbitrary Lagrangian Eulerian (ALE) formulation to deal with mesh deformation. The ALE integral Navier-Stokes equations in conservative variables can be written as:

$$\begin{aligned}
&\frac{\partial}{\partial t}\int_{V_t}\rho_f\,dV+\int_{S_t}\rho_f\,\mathbf{c}\cdot\mathbf{n}\,dS=0\\
&\frac{\partial}{\partial t}\int_{V_t}\rho_f\mathbf{u}\,dV+\int_{S_t}\rho_f\mathbf{u}\,\mathbf{c}\cdot\mathbf{n}\,dS=\\
&\qquad -\int_{S_t}p\,\mathbf{n}\,dS+\int_{S_t}\tau\cdot\mathbf{n}\,dS+\int_{V_t}\rho_f\mathbf{b_f}dV \qquad (1)\\
&\frac{\partial}{\partial t}\int_{V_t}\rho_f E dV+\int_{S_t}\rho_f E\,\mathbf{c}\cdot\mathbf{n}dS=\\
&\qquad -\int_{S_t}p\,\mathbf{u}\cdot\mathbf{n}+\int_{S_t}\mathbf{u}\,\tau\cdot\mathbf{n}dS-\int_{S_t}\mathbf{q}\cdot\mathbf{n}\,dS
\end{aligned}$$

where $\mathbf{u}$ is the fluid velocity, $\mathbf{c}$ is the convective velocity, ρ_f is the fluid density, p is the pressure, τ is the deviatoric stress tensor, $\mathbf{b}$ are the body forces, E is the energy and $\mathbf{q}$ is the heat flux, and V_t and S_t are the time-varying volume and surface of the control space. The swept volumes approach by Lai et al.[10] has been used to ensure the fulfillment Geometry Conservation Law at the calculation of the mesh velocity. Additionally, transfinite mapping algorithm has been adopted to deal with the adaption of the fluid mesh to the new position of the structure.

2.2 Structural Model

For nonlinear dynamics the equations ruling the structural deformation correspond to Newton's second law of motion, which are naturally posed on a lagrangian frame:

$$\frac{D}{Dt}\Big(\rho_s\frac{D\mathbf{d}}{Dt}\Big)-\nabla\cdot\sigma_\mathbf{s}=\rho_s\mathbf{b_s} \qquad (2)$$

where $\mathbf{d}$ are the displacements, ρ_s is the structural density, $\sigma_\mathbf{s}$ is the stress tensor and b_s are the body forces.

This equations are discretized in *CalculiX* according to the Finite Element Method over unstructured meshes and integrated in time with the Hilber-Hughes-Taylor method. A large displacements kinematic description introduces an additional degree of non-linearity.

2.3 Fluid-structure coupling conditions

Across the fluid-structure interface the continuity of surface tractions:

$$\sigma_\mathbf{s}\cdot\mathbf{n}=\tau\cdot\mathbf{n}-p\,\mathbf{n} \qquad (3)$$

and of displacements:

$$\mathbf{d_s}=\mathbf{d_f} \qquad (4)$$

is required for viscous flows (no-slip condition) [4], where $\mathbf{d_f}$ are the displacements of the fluid mesh at the interface.

These kinematic conditions also imply the respective time derivatives.

If the meshes at both sides of the interface match and both spatial discretizations have the same order of accuracy, then the transfer of variables from one field to the other is a relatively trivial task. In most realistic applications however, the fluid and structure meshes are incompatible along the fluid-structure interface, either because they have been designed by different analysts, or because the fluid and structure problems have different resolution requirements [4].

Therefore, when non-matching grids are used, some sort of mechanism for the exchange of data between meshes is to be devised. All existing methods include in some way two steps: a next-neighbour search and an interpolation stage. These, in turn, take place in two directions: from fluid mesh to structure mesh for the load transfer and from structure mesh to fluid mesh for the displacement transfer. The two interpolation procedures, however, cannot be independent if energy is to be conserved [6]. In our development the consistent interpolation [4] is used for displacement exchange and an own algorithm was developed for the transfer of loads.

3. GENERAL STRATEGY

The various options and issues that should be considered for the development of an efficient and modular fluid-structure coupling are discussed in this section.

Monolithic and partitioned approaches

The first choice for the solution of the coupled fluid-structure problem is between the monolithic and partitioned approaches. In the former the coupled algebraic system of equations is solved as a whole in a singe program. While the monolithic approach offers the advantages of higher stability and no need for data exchange procedures, it also has a number of drawbacks like loss of software modularity, limitations to the mathematical formulation of the different fields and challenges with respect to the problem size and conditioning of the overall system matrix [17]. Furthermore, at FSM we do intend to develop Structural Mechanics as the monolithic approach would demand, since it is not our research field.

Partitioned approaches, on the contrary, exhibit the following advantages [5]:

- different time integration schemes and time steps (subcycling) can be used for each of the fields, leading to a possibly more optimal solution

- ease computational load balancing (work load distribution among cores)
- enables the use of specialized and well tested pre-existent software packages for the individual fields

The partitioned approach shows clear advantages for the Institute's research strategy and was chosen for this work. The next step was to find an appropriate structural solver to match with *SPARC*. The structural solver Z88 which had been used in an earlier development of a steady FSI [9] is not capable of structural dynamics. After careful assessment the open-source program *CalculiX* [1] was adopted for the structural calculations. Beside a rich element library *CalculiX* offers the features needed for unsteady FSI modelling, namely, non-linear geometric support and time integration. Other important analysis types such as modal analysis for the determination of resonant frequencies are also available.

Software solution

From the software standpoint, several alternatives were then examined to implement the partitioned approach in an efficient way:

- Embed the *CalculiX* code into *SPARC* code. This strategy had been followed at FSM for the development of the early steady FSI and for another work in spray modelling. While this approach is simple it has a series of drawbacks: works only for strictly sequential workflow, makes the handling of dissimilar data types cumbersome, complicates the task of updating the third party source code after each new release and does not facilitate a modular handling
- Use scripting to call the *SPARC* and *CalculiX* executables making all communication through text files. This approach also has the disadvantage of a strictly sequential workflow and the problem of slow communication.
- Use a coupling manager based upon the Message Passing Interface to account for the communication between programs. This alternative provides a great degree of modularity. The de-facto-standard commercial software package MpCCI, developed by the Fraunhofer Institute SCAI has proven well suited for the task but introduces the already addressed inconveniences of commercial tools. To avoid them some research groups have developed their own tools, such as CoMA [15] and preCICE [8].

While the first two options are simpler, the last clearly offers the most advantages and was consequently chosen, leading to the creation of a Fluid-Structure-Interaction Simulation Manager, or *FSiM*.

This development has been focused on conventional FSI problems exchanging forces and displacements. An additional thermal coupling, comprising the solution of the thermal problem in the structure and the exchange of temperature or heat flow, can be nonetheless easily accommodated in the created structure.

Time-stepping

For the partitioned approach, there are few options available to combine the fluid and structure solutions in time.

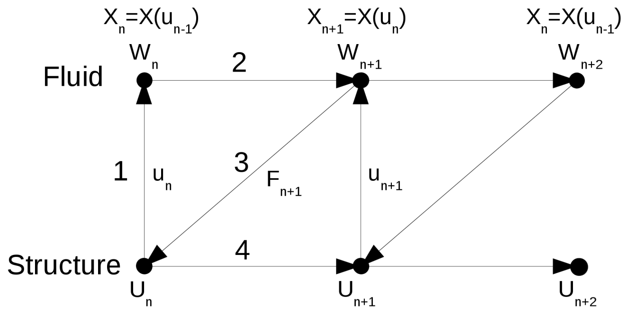

Figure 1: The Conventional Sequential Staggered procedure

A classical workhorse is the Conventional Sequential Staggered procedure of Farhat et al. [3], outlined in Figure 1, which has merely first order of accuracy in time and does not strictly enforce displacement continuity. However, if time steps are small as is the case for time-resolved turbulent (LES), compressible flows, the error is very small. This algorithm was chosen due to its simplicity. An alternative is the Improved Parallel Staggered procedure [3], which is beneficial only if the computational cost of both part-solvers is comparable. In fact the fluid solver resources demand is usually far greater than that of its structural counterpart.

Both procedures can be classified as loose coupling, since data exchange takes place only once per time step. The costly strong coupling (requiring extra iterations) is not needed for the considered applications, since the density ratios $\frac{\rho_f}{\rho_s}$ are rather small producing a quite stable interaction. Depending on how modularly the source code of the part-solver is structured, the modification of some core routines might be unavoidable. The most relevant example in our case is about the realisation of the time-stepping. Initially both part-solvers were spawned once per time step as depicted in Figure 2 and commanded to compute only one time step, using their restart features to advance in time. This was easier to implement at first, since less intervention in the *SPARC* source code was needed but it is clear that this approach produces recurrent reallocation of memory and reading of input files. For the structure this was found acceptable, due to the much smaller mesh size but an improvement was carried out to make *SPARC* run continously, as shown in Figure 3. This required, however, a deeper intervention which might be very time consuming if the code is new to the developer. What is more, if a strong coupling is desired, an extra loop would have to be added to *SPARC* to account for the coupling iterations.

Mesh matching

The neighbour search and interpolation stage are most conveniently placed in the coupling manager since data from both solvers is needed for that purpose. Additionally, on energy conservation considerations both directions of the data exchange are not independent and must be dealt with centrally. There is a parallelization potential in these interpolation algorithms which is however, only of secondary importance since the computational cost is relatively small. The neighbour search needs to be done only once at the beginning of the computation because there is no relative motion between meshes.

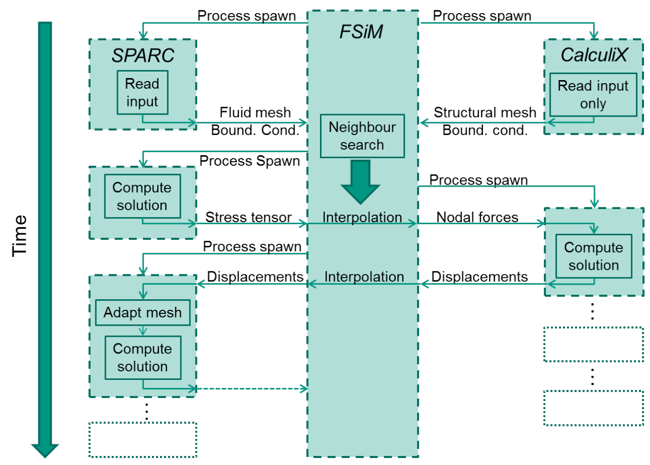

Figure 2: Basic flow diagram

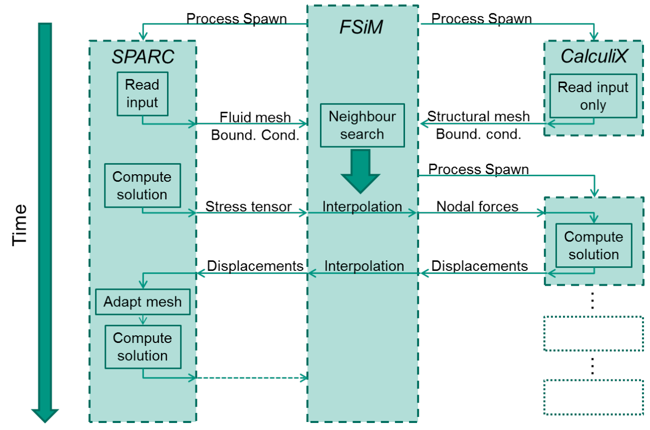

Figure 3: Improved flow diagram

Input files

The modular approach should be exploited also as regards the input files. It would be senseless to have the coupling manager read in the input files again. Rather, the arrays where the required data (mainly mesh and boundary conditions) is stored were identified in the part-solvers and transferred to the manager.

4. IMPLEMENTATION IN MPI

This section illustrates some details of the implementation in MPI. A somewhat heterogeneous framework resulted: on the one hand *SPARC* and *FSiM* are programmed in FORTRAN whereas *CalculiX* has been developed in C. On the other hand *SPARC* has a pure MPI parallelization while *CalculiX* takes advantage of the multithreaded version of the SPOOLES solver.

Once *FSiM* is running, *SPARC* is spawned first with the desired number of processes, 4 in our example, returning the new communicator. For the use in network clusters, an info object needs to be created to account for the machine files:

```
CALL MPI_INFO_CREATE(info, ierror)
CALL MPI_INFO_SET(info,'hostfile','./sparc_mf',ierror)
CALL MPI_COMM_SPAWN(sparc.bin,MPI_ARGV_NULL,4,  &
info,0,MPI_COMM_SELF,intercomm,array_of_errcodes,ierror)
```

Within *SPARC* a pure MPI parallelization has been developed throughout the years, using a collection of blocking, non-blocking and collective communication. A master-worker approach had been followed, i.e., the master process is in charge of all input and output and therefore distributes and collects data from all other processes. It was then natural to have only the master communicate with *FSiM*. In the *SPARC* code, right after the `MPI_INIT` line `MPI_COMM_GET_PARENT` is executed to retrieve the communicator between *SPARC* and *FSiM*. Since *SPARC* might also work independent from *FSiM* it is practical to inquiry the value of the communicator. A value `MPI_COMM_NULL` marks standalone operation.

The blocking synchronous send was adopted for all communication between *SPARC* and *FSiM*, for safety and simplicity reasons. For a parallel staggered algorithm non-blocking communication might be of convenience.

Finally, before `MPI_FINALIZE` the communicator must be freed from both sides using `MPI_COMM_DISCONNECT`.

In *CalculiX* the same strategy is followed but an additional control variable is passed from *FSiM* to let *CalculiX* know if mesh and boundary conditions or the displacement of the current time are to be transferred, as shown in Figure 3. There is no need for the special `MPI_INIT_THREAD` because no MPI commands take place during the execution of the multithreaded SPOOLES.

Error handling

One major difficulty with the use of existing codes is the error handling. A clean handling is quite painstaking since it requires intervention in the error handling implementation of the part-solvers and the proper cancellation of the pending communications. In *SPARC* the execution stop due to an error is centralized and that allowed a simple application of `MPI_ABORT`. This was not possible for *CalculiX*

Text output

Text output is scattered throughout both part-solver's code, leading to non-deterministic output. The only known solution is to centralize it in the manager, a very laborious option. A stream redirection to separate the three outputs would be very handy in this case. This can be done with an option for `mpiexec` but, to the author's knowledge, there is no info key that allows this for `MPI_COMM_SPAWN`.

Platform issues

FSiM is started differently depending on the platform. For use in supercomputers with a batch system the `job_sumit` command provided by the administrator must be used to request computing time, memory and number of CPUs and indicate the executable, for example:

```
job_submit -t 60 -m 512 -p 8 -J jobscript
In jobscript: mpirun -n 1 fsim.bin
```

where 8 CPUs and 512 MB are requested during 60 minutes. It is worth noting that although all desired CPUs must be requested from the beginning only one process is launched at first.

For the use in network connected clusters *mpirun* is directly executed but additional options are needed:

```
mpirun --mca plm_rsh_agent rsh -mca rmaps seq
 -n 1 -machinefile fsim_mf fsim.bin
```

FSiM	SPARC	CalculiX
NODE 1	NODE 2	NODE 1
NODE 2	NODE 2	
NODE 2	NODE 2	
NODE 2	NODE 2	
NODE 2		
NODE 1		

Table 1: Machine files

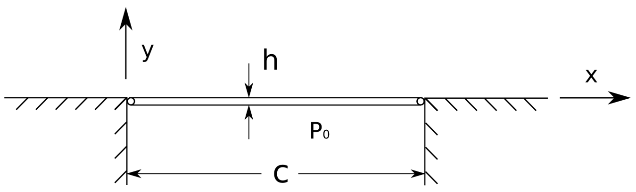

Figure 4: Geometry of the test case.

The first is how to execute the connection. In this case the widely extended *rsh* was used to easily manage permissions. The other two options regard the distribution of processes by means of the *machine files*. This is best illustrated with an example. Table 1 shows the content of the three machine files needed. In OpenMPI all nodes must be contained in the machine file given to *mpiexec* (first column), although only the first entry is used to spawn *FSiM*. The other files act as a filter on the entries of the first. In OpenMPI the *rmaps seq* option must be used to ensure the given sequence of nodes is strictly followed. Otherwise it cannot be controlled where the different programs are spawned. In this example *FSiM* and *CalculiX* are required to run on one node taking 1 and 3 CPUs respectively while *SPARC* takes the 4 CPUs of the other node. Note that only 1 multithreaded *CalculiX* process is created. Since *CalculiX* is spawned last, the required CPU on node 1 appears last.

Other issues

As presented in Figure 3 *CalculiX* is spawned and finishes several times during a simulation. With the MPICH-2 implementation zombie processes of the *hydra* manager accumulated during the simulation. This was reported and accepted as a bug and added to the MPICH-3.1 milestone.

An important shortcoming is the lack of CPU idling while waiting. Due to this, despite the strictly sequential character of the present solution, extra CPUs have to be requested for *FSiM* and *CalculiX*.

5. TEST CASE

The experimental validation of Fluid-Structure-Interaction tool is still a challenging issue. From the numerical standpoint 2D test cases are desired but 2D measurements are generally very difficult. Side-wall effects in experiments such as those conducted by Schäfer et al. [14] are hardly numerically reproducible. On these grounds we opted for a numerical verification with the help of the panel flutter problem, a known case from the aeronautic industry. Figure 4 shows the geometry of the problem, resembling a fuselage side panel.

The flexible aluminum panel has a chord length c of 1m, a thickness h of 2mm and is pivoted in its ends. One side of the panel is surrounded by air at aeronautic normal conditions at sea level. A constant pressure acts on the other side such that the forces are initially balanced. A transonic, non-viscous flow of Ma = 0.95 exists parallel to the panel, which is initially at rest. If a small perturbation is applied to the panel a travelling wave appears after a short transient. The panel deflection at three points along the chord are compared with the reference work of Massjung et al. [13] in Figure 5. Qualitatively both simulations deliver the same results with period deviation of 3%.

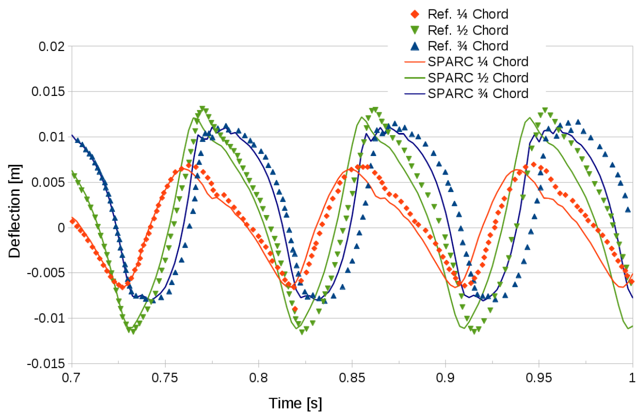

Figure 5: Comparison of results against the reference data.

The computation of the approximately 40,000 fluid cells and 40 solid structural elements for 30,000 time steps of 2.5e-5 s took around 30 hours on 4 Intel *i5* @ 3.4 GHz CPUs, 2 for *SPARC*, 1 for *FSiM* and 1 for *CalculiX*. Of the total computation time *SPARC* needed 82%, *CalculiX* 9%, the MPI communication 7% and *FSiM* 2%.

6. CONCLUSION

The implementation of an efficient tool for the simulation of the Fluid-Structure Interaction on the basis of existing part-solvers has been presented. The alternatives considered along with their advantages and disadvantages have been discussed. All relevant aspects that should be taken into account to accomplish the software solution were exposed. The main issues found along the development are related to the lack of CPU idling while waiting and a clean handling of text output and errors. Finally, a test case demonstrating the capabilities of the new tool is provided, showing very good agreement against reference results.

7. ACKNOWLEDGMENTS

This work has been financed in its greatest part by the German Academic Exchange Service (DAAD).

8. REFERENCES

[1] Copyright (C) 1998 Guido Dhondt and Klaus Wittig www.calculix.de

[2] P. Destuynder and J. Vétillard. Modelling and control of noise in a flexible flow duct. *Computer Methods in Applied Mechanics and Engineering*, 197(19-20):1801 – 1812, 2008. Computational Methods in Fluid-Structure Interaction.

[3] C. Farhat and M. Lesoinne. Two efficient staggered algorithms for the serial and parallel solution of three-dimensional nonlinear transient aeroelastic problems. *Computer Methods in Applied Mechanics and Engineering*, 182(3-4):499 – 515, 2000.

[4] C. Farhat, M. Lesoinne, and P. L. Tallec. Load and motion transfer algorithms for fluid/structure interaction problems with non-matching discrete interfaces: Momentum and energy conservation, optimal discretization and application to aeroelasticity. *Computer Methods in Applied Mechanics and Engineering*, 157(1-2):95 – 114, 1998.

[5] G. P. Farhat C, van der Zee K. Provably second-order time-accurate loosely-coupled solution algorithms for transient nonlinear computational aeroelasticity. *Comput Methods Appl Mech Eng*, 195:1973–2001, 2006.

[6] C. A. Felippa, K. C. Park, and M. R. Ross. A Classification of Interface Treatments for FSI. In H.-J. Bungartz, M. Mehl, and M. Schäfer, editors, *Fluid Structure Interaction II*, volume 73 of *Lecture Notes in Computational Science and Engineering*, pages 27–51. Springer Berlin Heidelberg, 2010.

[7] M. Glück. *Ein Beitrag zur numerischen Simulation von Fluid-Struktur Interaktionen - Grundlagenuntersuchungen und Anwendung auf Membrantragwerke*. Dissertation, Technische Fakultät der Universität Erlangen-Nürnberg, 2002.

[8] B. G. M. M. H.-J. Bungartz, J. Benk and T. Neckel. Partitioned Simulation of Fluid-Structure Interaction on Cartesian Grids. In *Fluid Structure Interaction II*. Springer Berlin Heidelberg, 2010.

[9] M. G. Iris Pantle, G. Bárdossy. A Hybrid Approach for Investigating Fluid-Structure Interactions Combining Classical CFD with FEM Methods. In *Proceedings 9th International Symposium on Experimental and Computational Aerothermodynamics of Internal Flows/9th ISAIF, GyeongJu, Korea*, 2009.

[10] P. A. J. Lai, Y. G. A Finite-Volume Method For Fluid Flow Simulations With Moving Boundaries. *International Journal of Computational Fluid Dynamics*, 2:1061–8562, 1994.

[11] F. Lippold. *Zur Simulation von Fluid-Struktur-Wechselwirkungen mit flexiblen Kopplungsverfahren*. PhD thesis, Institut für Strömungsmechanik und Hydraulische Strömungsmaschinen. Universität Stuttgart, 2009.

[12] T. G. U. I. R. W. M. Hojjat, E. Stavropoulou and K.-U. Bletzinger. Fluid-Structure Interaction in the Context of Shape Optimization and Computational Wind Engineering. In *Fluid Structure Interaction II*. Springer Berlin Heidelberg, 2010.

[13] W. D. J. B. Ralf Massjung, Jörg Hurka. On well-Posedness and Modelling for Nonlinear Aeroelasticity. In J. Ballmann, editor, *Flow Modulation and Fluid-Structure Interaction at Airplane Wings*, volume 84 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, chapter 4. Springer, 2003.

[14] M. Schäfer, M. Heck, and S. Yigit. An Implicit Partitioned Method for the Numerical Simulation of Fluid-Structure Interaction. 53:171–194, 2006.

[15] U. I. K.-U. B. T. Gallinger, A. Kupzok and R. Wüchner. A computational environment for Membrane-Wind Interaction. *Fluid-Structure Interaction. Theory, Numerics and Applications.*, Herrsching am Ammersee. 29.9-1.10, 2008.

[16] S. Turek, J. Hron, M. Madlik, M. Razzaq, H. Wobker, and J. Acker. Numerical simulation and benchmarking of a monolithic multigrid solver for fluid–structure interaction problems with application to hemodynamics. In H. Bungartz, M. Mehl, and M. Schäfer, editors, *Fluid-Structure Interaction II: Modelling, Simulation, Optimisation*. Springer, 2010. doi 10.1007/978-3-642-14206-2.

[17] W. 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. 10.1007/3-540-34596-5-9.