Partitioned Fluid–Structure–Acoustics Interaction on Distributed Data: Numerical Results and Visualization

David Blom, Thomas Ertl, Oliver Fernandes, Steffen Frey, Harald Klimach, Verena Krupp, Miriam Mehl, Sabine Roller, Dörte C. Sternel, Benjamin Uekermann, Tilo Winter, and Alexander van Zuijlen

Abstract We present a coupled simulation approach for fluid–structure–acoustic interactions (FSAI) as an example for strongly surface coupled multi-physics problems. In addition to the multi-physics character, FSAI feature multi-scale properties as a further challenge. In our partitioned approach, the problem is split into spatially separated subdomains interacting via coupling surfaces. Within each subdomain, scalable, single-physics solvers are used to solve the respective equation systems. The surface coupling between them is realized with the scalable

D. Blom • A. van Zuijlen
Aerospace Engineering, Delft University of Technology, Delft, The Netherlands
e-mail: d.s.blom@tudelft.nl; A.H.vanZuijlen@tudelft.nl

B. Uekermann
Scientific Computing in Computer Science, Technical University of Munich, München, Germany
e-mail: uekerman@in.tum.de

H. Klimach • V. Krupp (✉) • S. Roller
Simulation Techniques and Scientific Computing, University of Siegen, Siegen, Germany
e-mail: harald.klimach@uni-siegen.de; verena.krupp@uni-siegen.de; sabine.roller@uni-siegen.de

M. Mehl
Institute for Parallel and Distributed Systems, University of Stuttgart, Stuttgart, Germany
e-mail: miriam.mehl@ipvs.uni-stuttgart.de

T. Ertl • O. Fernandes • S. Frey
VISUS, University of Stuttgart, Stuttgart, Germany
e-mail: thomas.ertl@visus.uni-stuttgart.de; oliver.fernandes@visus.uni-stuttgart.de; steffen.frey@visus.uni-stuttgart.de

D.C. Sternel
Institut for Scientific Computing, TU Darmstadt, Darmstadt, Germany
e-mail: doerte.sternel@hpc-Hessen.de

T. Winter
Institute of Numerical Methods in Mechanical Engineering, TU Darmstadt, Darmstadt, Germany,
e-mail: winter@fnb.tu-darmstadt.de

H.-J. Bungartz et al. (eds.), *Software for Exascale Computing – SPPEXA 2013-2015*, Lecture Notes in Computational Science and Engineering 113,
DOI 10.1007/978-3-319-40528-5_12

open-source coupling tool preCICE described in the “Partitioned Fluid–Structure–Acoustics Interaction on Distributed Data: Coupling via preCICE”. We show how this approach enables the use of existing solvers and present the overall scaling behavior for a three-dimensional test case with a bending tower generating acoustic waves. We run this simulation with different solvers demonstrating the performance of various solvers and the flexibility of the partitioned approach with the coupling tool preCICE. An efficient and scalable in-situ visualization reducing the amount of data in place at the simulation processors before sending them over the network or to a file system completes the simulation environment.

1 Introduction

The handling of fluid–structure–acoustics interactions (FSAI) in detailed, direct simulations is challenging and became feasible only recently on large scale computing systems. FSAI induces not only multiple physical domains but, at the same time, also different length and time scales. The possibility to consider the interaction of all those aspects in a single simulation enables us to gain a better understanding of the physical system and, thereby, to make more accurate predictions in engineering design processes. This can, for example, help to reduce the noise emission of technical devices such as aircraft, fans or wind turbines. Wind turbines are of increasing importance. While generating electricity, noise is emitted (due to complex rotor–wind–interaction). With their increasing size and growing number, limiting their noise emission becomes necessary. The multi-scale nature of the FSA interactions can be nicely seen in this example. Noise is generated in the boundary layers and resulting vortices at the moving geometry at a length scale in the order of centimeters. This whole turbine at the scale of meters while the noise emission is of relevance in a range of hundreds of meters up to a few kilometers. Simulating the entire domain while resolving the smallest turbulent scales and resolving the boundary layer adequately would require approximately 10^{18} degrees if freedom. This is out of reach for current systems.

Though tremendous computational power is available nowadays, such a simulation is still too demanding. In addition, acceptable software development times are possible only if we re-use existing scalable software based on decades of experience in each single-physics discipline. Therefore, we employ a partitioned coupling approach, i.e., we split the physical space into smaller domains, each covering a so-called single-physics subdomain. Within each of these subdomains, the specific physics are solved with a locally adapted resolution. The interaction between the subdomains is realized by exchanging data and applying suitable iterative solvers at the common surface. This coupled approach allows for adapted numerical methods and resolutions in each of the domains according to the physical requirements. By the adaptation of the numerical approximation in the individual domains, the computation of the complete interaction between fluid mechanics, structural mechanics, and acoustic wave propagation becomes feasible.

As the coupling mechanism needs to deal with various types of solvers and does not know or need to know anything about their internal numerical and technical details, this approach is also referred to as black-box coupling. A suitable tool to realize such a coupling in a parallel and scalable way is the open-source coupling library preCICE.[1] It handles without the data exchange, interpolations between non-matching meshes of adjacent domains, and iterative solvers for the coupling surface equations. As solvers, we use FEAP [17] or OpenFOAM[2] for structural mechanics, FASTEST [6] or OpenFOAM for the simulation of acoustic fluids, and Ateles[3] for the acoustic far field [1].

Naturally, this approach raises new numerical challenges such as stability issues due to the partitioned coupling and data interpolations between non-matching meshes. In addition, also the performance and scalability of the coupled black-box setup is a non-trivial task. The coupling library preCICE and the contributions to highly parallel partitioned coupling are described in “Partitioned Fluid–Structure–Acoustics Interaction on Distributed Data: Coupling via preCICE”. In this chapter, we describe a real world application including not only the two-dimensional surface coupling, but the efficiency of the overall three-dimensional simulation. Section 2 presents the individual solvers and the solved equation systems. The two coupling strategies for fluid–structure interaction and fluid–acoustic interaction are described in Sect. 3. Section 4 deals with the visualization of the application with a focus on in-situ visualization and the developed data handling strategy. This is followed by a discussion of numerical results and scaling data for a three-dimensional testcase in Sect. 5.

2 Description of the Individual Solvers

In this section, we briefly present the physical background including the governing equations for each single-physics discipline. We then continue with a short description of the solvers for each part including the deployed numerical methods. The physical regimes we simulate are in the field of classical mechanics. We are interested in the motion of fluids and elastic structures. Acoustic waves are a special case of fluid motion, where small disturbances of the pressure in a compressible medium are considered.

The collection of solvers we use is a combination of inhouse solvers and open-source software. The focus is on testing the methodologies of the inhouse solvers in a complex coupled simulation environment on the one hand and to check the black-box parallel coupling concept of preCICE on the other hand. The solvers use sophisticated techniques that make them particularly suited for high performance simulations. For example, the flow and acoustic solver Ateles uses octree grids and high order Discontinuous Galerkin (DG) discretizations that are both known to perform well on massively parallel computers and model version of DG is

[1] http://www.precice.org

[2] http://www.openfoam.org/

[3] `University of Siegen, STS`

predestined for solving linear problem efficiently. FASTEST implements an internal volume coupling between an incompressible flow and an acoustic perturbation in order to be able to efficiently use space and time adaptivity corresponding to the different spatial and temporal scale of flow and acoustics. OpenFOAM and Feap are widely used open-source solvers, an important class of solvers that preCICE should be able to handle. In addition, they are used extensively in real-world applications in two of the authors' groups.

Whereas the splitting into an elastic structure subdomain and a fluid subdomain in an application scenario is obvious, the division between flow and acoustics is more involved: because acoustics is the phenomenon of travelling pressure waves in fluids, it is ultimately governed by the compressible fluid dynamics equations. However, only small perturbations in an otherwise constant fluid state are considered. Moreover, the resolution of all relevant scales for turbulent flows in a low Mach number regime with the speed of sound would be prohibitively expensive. Therefore, we suggest a domain splitting into acoustic near field and acoustic far field. The near field is the domain where acoustic pressure waves are generated by the fluid flow and, thus, coincides with the flow domain. The far field neglects the flow and takes into account only the acoustic pressure waves.

For the near field, we use two different approaches: the first approach uses the fully compressible fluid dynamics equations (using OpenFOAM), the second one is based on a splitting as, for example, proposed by Hardin and Pope [9]. They suggest (for comparably small flow velocities) to use an incompressible method for the calculation of the flow field, from which the acoustic sources can be deduced by the time derivative of the pressure. These sources can then be fed into an acoustic solver, which calculates the propagation of the acoustic waves. We have realized both approaches with the following setups:

(a) compressible flow (OpenFOAM) $\rightarrow$ acoustic far field (Ateles)
(b) incompressible flow (FASTEST) $\rightarrow$ acoustic near field (FASTEST) $\rightarrow$ acoustic far field (Ateles)

The arrows define the direction of interaction. In the following, we summarize first the governing fluid dynamics equations and the two solvers OpenFOAM (compressible) and FASTEST (incompressible), followed by a section on the acoustic equations both in the near field (used for approach (a)) and in the far field (used in approach (a) and (b)). Afterwards, the mechanics of elastic structures and the respective solvers OpenFOAM and FEAP are recapitulated.

2.1 Fluid Dynamics in the Acoustic Near Field

Fluid flow is governed by the compressible Navier-Stokes equations. These equations are the conservation of mass

$$\frac{\partial \rho^f}{\partial t} + \nabla \cdot \left(\rho^f \boldsymbol{v}\right) = 0 \,, \tag{1}$$

the balance of momentum

$$\rho^f \left(\frac{\partial \boldsymbol{v}^f}{\partial t} + \left(\nabla \cdot \boldsymbol{v}^f \right) \boldsymbol{v}^f \right) = -\nabla p^f + \rho^f \nu \left(\Delta \boldsymbol{v}^f + \frac{1}{3} \nabla (\nabla \cdot \boldsymbol{v}^f) \right) + \boldsymbol{B} \,, \tag{2}$$

and the balance of energy

$$\frac{\partial \rho^f e}{\partial t} + \nabla \cdot \left(\boldsymbol{v}^f \left(\rho^f e + p^f \right) \right) = 0 \,. \tag{3}$$

In these equations, $\boldsymbol{v}^f$ denotes the velocity field, p^f the pressure field, ρ^f the density, ν the kinematic viscosity, B an external force, and e the energy. The assumption of an ideal gas yields a relation between the pressure p^f and the energy e:

$$p^f = \rho^f R T = (\gamma - 1) \left(e - \frac{\rho^f \boldsymbol{v}^f \cdot \boldsymbol{v}^f}{2} \right) \,.$$

R is the ideal gas constant, T the temperature, and γ the isentropic coefficient. For an incompressible flow, the density is assumed to be constant and the conservation of mass in Eq. (1) is reduced to

$$\nabla \cdot \boldsymbol{v}^f = 0 \,. \tag{4}$$

Equation (3) become redundant in this case and (2) is reduced to

$$\frac{\partial \boldsymbol{v}^f}{\partial t} + (\nabla \cdot \boldsymbol{v}^f) \boldsymbol{v}^f = -\nabla p^f + \nu \Delta \boldsymbol{v}^f + \boldsymbol{B} \,. \tag{5}$$

2.1.1 OpenFOAM: Compressible Flow Solver

We use the foam-extend-3.1 package[4] for compressible flow simulations. It is a fork of the well known OpenFOAM package.[5]

foam-extend-3.1 uses second order finite volume discretization in space and a second order backward differencing time integration scheme. To solve the Navier-Stokes equations at all speeds, a coupled pressure-based algorithm [4, 5] is employed. Hence, the continuity and momentum equations are solved in a fully coupled implicit manner and, thereafter, the energy equation is solved in a segregated manner.

[4] http://www.extend-project.de/

[5] http://www.openfoam.org/

2.1.2 FASTEST: Incompressible Flow Solver

FASTEST [6] is a fully conservative finite volume code which is second order accurate in space and time. The block structured, boundary adjusted grid with free topology enables the use of an efficient algorithm and the application to moderately complex geometries.

2.2 Acoustic Wave Propagation

Acoustics can be considered as small perturbations of the flow field. We split the state variables of the flow into a background state denoted by the subscript 0 and the acoustic perturbations denoted by the superscript a, accordingly. For the density, we get $\rho^f = \rho^0 + \rho^a$, for the velocity $\boldsymbol{v}^f = \boldsymbol{v}^0 + \boldsymbol{v}^a$ and for the pressure $p = p^0 + p^a$. Non-linear effects can be neglected for the acoustic perturbations ρ^a, $\boldsymbol{v}^a$, and $\boldsymbol{p}^a$. This leads to the linearized Euler equations given by the linearized equation for the conservation of mass

$$\frac{\partial \rho^a}{\partial t} + \nabla \cdot (\boldsymbol{v}_0\, \rho^a + \rho_0\, \boldsymbol{v}^a) = 0\ , \tag{6}$$

the conservation of the velocity perturbation (neglecting friction)

$$\frac{\partial \boldsymbol{v}^a}{\partial t} + \nabla \cdot \left(\boldsymbol{v}_0 \boldsymbol{v}^a + \frac{1}{\rho_0} p^a \right) = 0 \tag{7}$$

and the conservation of the pressure perturbation

$$\frac{\partial p^a}{\partial t} + \nabla \cdot (\boldsymbol{v}_0 p^a + \gamma\, p_0\, \boldsymbol{v}^a) = q\ . \tag{8}$$

The background variables ρ_0,$\boldsymbol{v}_0$, p_0 are constant variables which define the flow properties as the speed of sound $c = \sqrt{\frac{\gamma p_0}{\rho_0}}$, with which the acoustic perturbations are then transported through the far field. q denotes a source term, which generates the acoustic perturbations.

2.2.1 FASTEST: Acoustic Near Field

In FASTEST, the acoustic solver calculates the wave propagation in the acoustics near field in an integrated way (volume coupling) based on the incompressible background flow equations [11]. This integrated solver is based on the approach of Hardin and Pope [9] with the modifications of Shen and Sørensen [15, 16], which lead to a set of linearized Euler Eqs. (6), (7), and (8) with source term $q = -\frac{\partial p^0}{\partial t}$

(in this case p^0 is the incompressible pressure which is time variant) generating the acoustic perturbation in the near field. Considering this method, the state variables ρ^0, $\boldsymbol{v}^0$, and $\boldsymbol{p}^0$ are the incompressible flow variables.

To solve the linearized Euler equations, a finite volume scheme together with a dimensional splitting approach [18] is employed. For time stepping, an explicit Euler time discretization scheme is used. The flux computation is realized with a high resolution scheme that combines an exact Riemann solver and a Lax-Wendroff scheme with a suitable flux limiter. The details of the method are described in [11]. The internal coupling between the flow field and the acoustic perturbations is described in Sect. 3.2.

2.2.2 Ateles: Acoustic Far Field

The far field acoustic solver Ateles propagates acoustic waves solving (6), (7), and (8) based on boundary information provided at the coupling surface between acoustic near field and acoustic far field. For the far field, the source term $q = 0$, since acoustic waves are only propagation and not generated. The flow properties p_0,v_0,$\boldsymbol{v}_0$ need to be given similar to the acoustic near field.

Ateles uses a modal high order Discontinuous Galerkin discretization as part of the APES framework [14]. The Discontinuous Galerkin (DG) method is based on a polynomial representation within each element and the flux calculation between elements [10]. Ateles allows for an arbitrary choice of the polynomial degree in the spatial representation and thus, an arbitrarily high order in space. This allows for a perfect tailoring to the physics and the computer architecture.

A high order approximation provides a low numerical dissipation and dispersion and a high accuracy with only few degrees of freedom. For nonlinear systems, high order implies increased computational costs, but for our linear system a modal scheme keeps the computational effort per degree of freedom constant over increased spatial orders and solves them efficiently. Such a discretization is, therefore, ideally suited for the simulation of the acoustic domain. The APES [14] framework, in which the solver is included, is designed such to take advantage of massively parallel systems.

2.3 Structural Dynamics

The configuration of the structure domain is described by the displacement $\boldsymbol{u}^s$. An elastic and compressible structure is assumed. The governing equation in a Lagrangian description, i.e. with respect to the initial reference state Γ^s, is given by the balance of momentum

$$\rho^s \frac{\partial^2 \boldsymbol{u}^s}{\partial t^2} = \nabla \cdot \left(J \boldsymbol{\sigma}^s \boldsymbol{F}^{-T} \right) + \rho^s \boldsymbol{g} \qquad \text{in } \Omega^s \,, \tag{9}$$

where the deformation gradient tensor $\boldsymbol{F}$ is defined as $\boldsymbol{F} = \boldsymbol{I} + \nabla \boldsymbol{u}^s$, and the Jacobian J is the determinant of the deformation gradient tensor $\boldsymbol{F}$. Applying the constitutive law for the St. Venant-Kirchhoff material, the Cauchy stress tensor $\boldsymbol{\sigma}^s$ becomes

$$\boldsymbol{\sigma}^s = \frac{1}{J} \boldsymbol{F} \left(\lambda^s \left(\operatorname{tr} \boldsymbol{E} \right) \boldsymbol{I} + 2\mu^s \boldsymbol{E} \right) \boldsymbol{F}^T \tag{10}$$

with the Young's modulus $\boldsymbol{E} = \frac{1}{2}\left(\boldsymbol{F}^T \boldsymbol{F} - \boldsymbol{I}\right)$, the relation of the shear modulus μ^s, and the Poisson's ratio ν given in [3].

2.3.1 OpenFOAM: Finite Volume Structure Solver

To facilitate implementation, the structure solver which is implemented within the foam-extend-3.1 framework, is used in combination with the compressible flow solver. Here, a finite volume discretization is used. The reader is referred to Cardiff et al. [3] for further details on the solid mechanics solver.

2.3.2 FEAP: Finite Element Structure Solver

In FSA simulations with FASTEST, the finite element code FEAP [17] is used. The spatial discretization is based on bilinear enhanced solid elements, and the time discretization scheme is the Newmark-Beta scheme [13].

3 Coupling

In this section, we describe the coupling between our three involved fields—fluid flow, elastic structure deformation, and acoustic wave propagation. As already mentioned above, this includes three coupling interfaces:

1. a surface coupling between structure and acoustic fluid (near field),
2. a surface coupling between acoustic fluid (near field) and acoustic wave propagation (far field),
3. a volume coupling between background flow and acoustic perturbations (near field).

The latter is realized in a monolithic way in our setup using the compressible flow solver in OpenFOAM and a software-internal volume coupling in the setup with FASTEST, respectively. For the two surface couplings, in the first step we assume that the interaction between near field and far field is unidirectional which is a feasible assumption also used in traditional approaches like [12] or [9]. An advantage in using a coupling library is that the interaction can be easily expanded to be bidirectional. Accordingly, a strong implicit coupling is used only at the

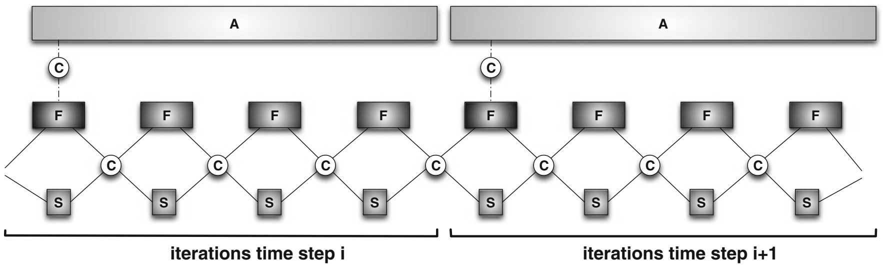

Fig. 1 Overview of the execution of the fluid–structure–acoustics simulations. Multiple calls to the fluid and solid solvers are performed, since an implicit coupling is applied for the fluid–structure interaction problem. A good load balancing can be achieved with the proper number of CPU cores for the acoustic domain. *F*: fluid, *S*: solid, *A*: acoustics, *C*: coupling

surface between the elastic structure and the near field whereas the acoustic far field simulation is executed only once per time step based on the current time step's data at the surface between near and far field. See Fig. 1 for an illustration. Thus, the fluid and structure solvers are executed multiple times per time step, while the acoustic far field solver is called only once per time step. An optimal load balancing can be obtained by choosing an appropriate number of CPU cores for each partition such that solving the acoustic domain takes approximately the same computational time as all fluid–structure iterations together. Each solver is considered to be black-box. Hence, only input and output information is accessible for the coupling. In the following, we describe the realization of the three coupling connections.

3.1 Coupling the Elastic Structure with the Acoustic Fluid

At the surface between the elastic structure and the acoustic fluid, structural displacements/velocities and forces exerted by the fluid on the structure are exchanged between the two involved solvers. The displacements/velocities are the input of the flow solver and the output of the structure solver, whereas the forces are the output of the flow solver and the input for the structure solver (Dirichlet-Neumann coupling). Therefore, we can shortly write the actions of the fluid solver F_f and the structure solver F_s at the coupling surface as at each time step the response of the fluid solver F_f is defined as

$$\boldsymbol{y} = F_f(\boldsymbol{x}) \quad \text{and} \quad \boldsymbol{x} = F_s(\boldsymbol{y}) \ , \tag{11}$$

where $\boldsymbol{x}$ denotes the displacement of the fluid–structure interface, and $\boldsymbol{y}$ denotes the force acting on the fluid–structure interface.

With this given, we can enforce the balance of stresses and the kinematic boundary conditions

$$\boldsymbol{\sigma}^f \boldsymbol{n} = \boldsymbol{\sigma}^s \boldsymbol{n} \quad \text{and} \quad \boldsymbol{v}^f = \boldsymbol{v}^s$$

for the stress tensors $\boldsymbol{\sigma}^f$ and $\boldsymbol{\sigma}^s$ at the surface between structure and fluid Γ^{fs} with the unit normal vector $\boldsymbol{n}$ by solving the staggered or the parallel fixed-point equation

$$\boldsymbol{x} = F_s \circ F_f(\boldsymbol{x}) \quad \text{or} \quad \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{pmatrix} = \begin{pmatrix} 0 & F_s \\ F_f & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{pmatrix}. \tag{12}$$

The motivation for these fixed-point equations, a further alternative and more details are described in [19]. The coupling tool preCICE provides several robust and efficient acceleration methods for the respective fixed-point iterations. Details are given in "Partitioned Fluid–Structure–Acoustics Interaction on Distributed Data–Coupling via preCICE".

3.2 Coupling the Acoustic Near Field with the Far Field

As already introduced, the primitive variables (pressure, density, velocity) at the interface between near field and far field are transferred (via the coupling tool preCICE) from near to far field once per time step. This yields a unidirectional coupling as shown above in Fig. 1.

To avoid non-physical oscillations induced by the coupling, the data mapping at the coupling interface is crucial. OpenFOAM uses an unstructured mesh in the fluid domain allowing for a geometry-adapted mesh even for complex geometries. A mesh deformation technique is used to interpolate the displacement of the fluid–structure interface into the complete flow field. In the acoustic far field, a structured octree mesh is used with a high order DG discretization. Thus, the exchange points at the coupling interface are non-equidistant quadrature points. Hence, coupling the finite volume OpenFOAM solver with this discontinuous Galerkin solver leads to non-matching meshes at the interface and an interpolation method is required to transfer the density, velocities and pressure from one solver to the other. We use a radial basis function interpolation or nearest neighbor projection (described in the chapter "Partitioned Fluid–Structure–Acoustics Interaction on Distributed Data–Coupling via preCICE") in order to reduce the introduced numerical errors due to the partitioning as far as possible.

Coupling the implicit second order backward difference time integration scheme used by OpenFOAM for the compressible flow with an explicit second or fourth order Runge Kutta scheme used by Ateles for the acoustic far field reduces the overall accuracy to first order in time. Thus, better combinations of time stepping schemes and more sophisticated coupling patterns in time are work in progress. In addition, applying explicit coupling poses time step restrictions for both solvers in order to achieve a stable integration in the acoustic field.

3.3 Coupling the Incompressible Flow with Acoustic Perturbations

As the kinetic energy of the acoustic perturbations is much smaller than the kinetic energy of the flow, there is only a coupling from the flow field to the acoustics field. Thus not from the acoustic field to the flow field. The acoustic sources $s(p^a)$ are computed as the time derivative $\frac{\partial p_0}{\partial t}$ of the incompressible flow pressure and transferred to the acoustic mesh.

Crucial aspects for the splitting approach are the coupling in time and the choice of time step sizes and mesh resolutions. Acoustics with an underlying "slow" flow velocity in a low Mach range, is a multi-scale problem in time as the flow velocity of an incompressible flow–with a maximum of $Ma = 0.3$–is much smaller than the speed of sound c. Using an explicit time discretization scheme for the acoustic equation, its *CFL* condition limits the time step size Δt by

$$\Delta t < \frac{\Delta x}{c} \tag{13}$$

depending on the spatial mesh resolution Δx. The flow solver with its slower velocity allows for larger time steps. Because the computation of the flow field is by far more expensive than the calculation of the acoustic perturbations, the numbers of flow solver time steps should be as small as possible. Two methods decoupling the time steps for the incompressible flow and the acoustic perturbations are available for the integrated flow–acoustics solver:

1. Sub-cycling: $N = \Delta t^a / \Delta t^f$ acoustic time steps are carried out within one flow time step and the acoustic sources are only updated after every N^{th} acoustic time step.
2. Adapting spatial mesh resolutions: the infrastructure of the geometric multigrid can be easily used for the restriction to a hierarchical coarsened spatial mesh for the acoustic perturbations leading correspondingly to a larger time step according to the CFL condition.

A combination of both leads to a substantial saving in computational time [11].

4 Visualization

In this section, we briefly discuss the visualization developed for the large scale simulation of fluid–structure–acoustic interactions. To reduce the communication and IO bottlenecks typical for massively parallel simulations, we employ an algorithm running *in-situ*: during the runtime of the simulation, a solver code calls visualization routines to process the data currently available. It resumes simulation calculations when the visualization routine call returns presented in Sect. 4.1. This concept is realized by a customized in-situ visualization architecture (Sect. 4.2). It employs an intermediate volume representation that reduces the amount of data but still maintains possibilities for user interaction (Sect. 4.3). We finally discuss in Sect. 4.4 how this representation is generated, stored and utilized for interactive exploration in the context of our in-situ architecture.

4.1 In-Situ Visualization

In our approach, the pressure data of the flow field determined after a full time step are used to generate an intermediate, view-dependent representation of the scalar field. Re-arranging and appropriately quantizing the new representation with the goal to raise statistical redundancy in the data, allows for an additional efficient compression of the pressure volume data by a lossless encoding scheme. Since no internode communication is necessary for this (all visualization steps are performed utilizing data available locally to a process), the network load is significantly reduced by only sending the efficiently compressed representation to a front-end node (instead of the full volume data). This enables an interactive volume visualization. The reduced representation additionally facilitates storing the amounts of data produced by an exascale simulation. A secondary goal is minimization of the impact of in-situ visualization calculations, as not to interfere with the ongoing simulation. Note that the acoustic domain is not yet considered in the integrated fluid–structure visualization.

4.2 Simulation–Visualization Setup

Figure 2 gives an overview on the integration of the visualization calculations into the solver. While the *simulation* step continuously generates new pressure data, *visualization transform* converts these volume data into an intermediate visualization representation of reduced size without or with only minimally compromising output quality. The reduced representation generated on each node is then sent to a front-end node which merges all the received data, from which *visualization render* finally generates an image. In the following, we shortly describe the basic ingredients of all these steps.

Simulation The simulation setup used for testing purposes is the bending tower case as described in Sect. 5.1, running a fluid–structure simulation with OpenFOAM components for the fluid and the structure solver (omitting the far field acoustics). In each time step, the simulation yields a three-dimensional scalar field for the pressure, representing a time-dependent unstructured mesh. At this point, the visualization algorithm is executed on these data.

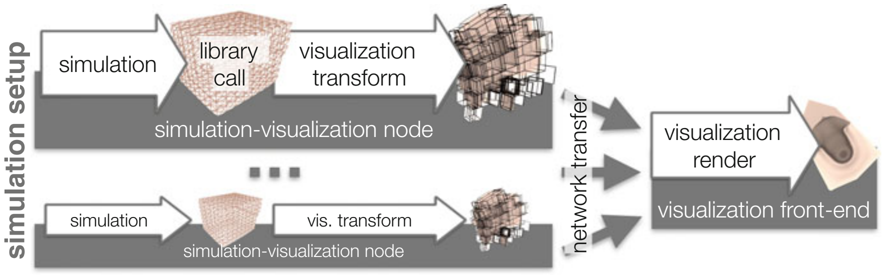

Fig. 2 The figure shows the embedding of the visualization into the simulation. The data gets transformed into an intermediate representation before being compressed by the TVDI algorithm on all simulation nodes. After being transferred and gathered on the front-end, the data from each node then gets reassembled and can be interactively explored

Visualization transform In *transform*, we generate an intermediate visualization representation with a direct interface to the solver. In particular, simulation results are not duplicated in memory, but the solver's own data object and access routines are employed for maximum performance and minimal resource strain. The core idea of visualization transform is to reduce the data set by partially already pre-processing the data as required by the chosen visualization method, yet not fully aggregating data to the final image in order to still maintain some flexibility. In the variant of the algorithm presented here, we employ a view-dependent representation that basically consists of volume rendered images in which the rays of a raycasting algorithm have not been fully composited to yield one resulting pixel color. Instead, while traversing the volume to gather samples, so-called segments are generated along the ray. For a detailed explanation on this intermediate representation, see Sect. 4.3.

Visualization render The visualization representation generated on each node is transferred to the front-end node where images are rendered, whereas certain parameters can be varied interactively. In the case of the algorithm presented here, the user can modify the selected time steps and arbitrary camera parameters. Only the transfer function used to determine a color from the scalar value is fixed.

4.3 Intermediate Representation: Volumetric Depth Images

Volumetric depth images (VDIs) are a condensed representation for classified volume data, providing high quality and reducing both render time and data size considerably. Instead of only saving one color value for each view ray as in standard images, VDIs store a set of so-called segments, each consisting of a geometry (depth range) with composited color and opacity. This compact representation is independent from the representation of the original data and can quickly be

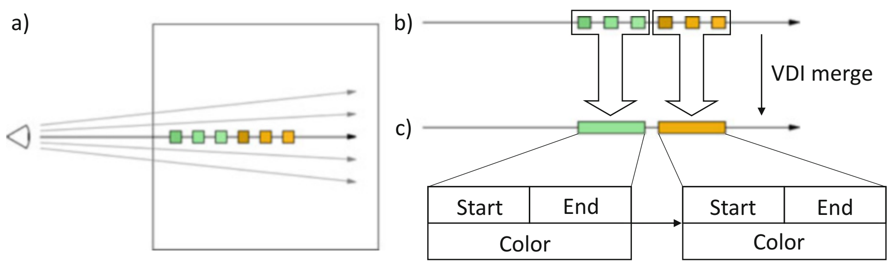

Fig. 3 VDI generation. (**a**) Raycasting is performed on pressure data. (**b**) Similar colors are merged to segments. (**c**) The list data structure is constructed

generated by a slight modification of existing raycaster codes. VDIs can be rendered efficiently at high quality with arbitrary camera configurations [8].

VDIs are generated during volumetric raycasting from a certain camera configuration by partitioning the samples along view rays according to similarity of the composited color, providing the additional possibility to skip 'empty' regions (see Fig. 3). These partitions are then stored as lists of so-called segments containing the bounding depth pair and respective accumulated color and opacity value. This reduces the amount of data in several ways: similar colors get merged and need to be stored only once per segment (as opposed to in every cell sampled). Domain knowledge can already be employed to limit data acquisition to an intuitively choosable camera frustum. Note that the raycasting is only performed on the data available to the process; no internode communication is performed.

4.4 Visualization Transform and Render

TVDI generation During the visualization transform stage, a VDI is generated by sampling the pressure data along viewing rays, which in turn are determined by a camera position, a view target and a size for the resulting "image" (i.e., the number of rays) chosen by the user. For each of these rays, the algorithm needs to traverse the time-dependent unstructured mesh of the simulation, generating the VDI as explained in Sect. 4.3.

Once the raycaster has completed its calculations, the newly obtained VDI data from the current time step are compared to the last step for all segments in a given ray. Considering that flow data typically vary rather smoothly over time, the segments stored in the current and the previous time steps will show a high degree of similarity, which can be exploited for compression. The so-called TVDI (time-clustered VDI) algorithm clusters the segments of a given ray over time by calculating the changes of VDI segments from the previous time step to the current. If similarity in color (a user-defined color space metric is employed) is determined, the segments get sorted into a so called region, in which all segments have the same color determined by the initial segment.

Exploiting the coherency between time steps mentioned before, the change in the segment geometry (starting/ending value) will be small between consecutive time steps and even constant over a few time steps, since flow pressure data typically vary approximately linear over a short time.

The changes in the geometry can be easily stored with minimal overhead (neighbor counts, for details see [7]) and are sufficient to reconstruct all segments at all time steps. Once a user-defined time step interval is reached, the current regions constructed in a given simulation node are sent over the network to the front-end node. An obvious gain is that the color for a time region only needs to be sent and stored once for all associated segments. Assuming that the geometry changes are constant (over a short time) and now have an optimal memory layout, we enable a lossless entropy encoder[6] to achieve a better compression, minimizing network load during data transfer.

Rendering (visualization render) On the front-end node, the regions constructed in the previous step are gathered, and the individual VDI representations for each time step are reconstructed. This is done by incrementally adding the changes stored for each segment to the previously constructed ones, with the first one having been stored as an absolute value. Each segment also gets assigned the regions color. The regions gathered from different processes are all incorporated into the same VDI data structure, giving a complete representation of each of the time steps within an output interval. Using the VDIs segment data and the origin and direction of the ray the segment was constructed from, a small frustum is constructed. Viewed from the origin of the constructing ray, the front and back square of this frustum exactly cover the pixel associated with the ray. The position of these frustum caps along the ray is just the front/end position stored in the segment as is the color. This is done for all segments in all rays, filling the space with these proxy geometry frustums. A second raycast, now taken from an arbitrary viewpoint, can then be run on this proxy geometry, exploiting the hardware acceleration associated with geometry than can be triangulated. Note that VDIs provide renderings identical to those of the original data when rendered from the view used for their generation, but provide high-quality approximations also for deviating views. The error in color stems from merging several similar segments across time, averaging their color. This error can be easily controlled by a user-defined threshold, trading quality versus compression rate.

5 The Three-Dimensional Bending Tower Testcase

In this section we present the physical and numerical results of our approach applied to a scenario involving all three discussed phenomena and their interaction. We have a look at the scalability of the coupled setup and the resulting solutions.

[6]http://bzip.org/

5.1 Testcase Description

To investigate the suitability of our approach with the coupling by preCICE and the different solvers for each domain, we use a bending tower as an example. The setup resembles for example the tower of a wind turbine, is fairly simple to understand, and includes three-dimensional effects. The tower is modeled with an elastic material and deforms according to the pressure forces by the surrounding fluid flow. Our computational setup is shown in Fig. 4. In the acoustic far field domain, only the propagation of the acoustic values is simulated, in the fluid domain the flow together with the acoustic perturbation as described in the previous sections. An overview to the physical parameters for each domain is given in Table 1 along with the initial conditions.

As already described shortly in Sect. 2, we apply two different sets of black-box solvers, the comparison of them is work in progress. In **set a**, OpenFOAM is used for the structure and for the acoustic fluid, which is considered compressible. For both regions, a finite volume discretization (2nd order) with a backward differencing time discretization (2nd order) is chosen. The structure is discretized with 4, 500 control volumes (CV), and the flow region with 3,095,500 CVs. For **set b**, the structural deflection is simulated with the finite element code FEAP, and the flow region with the finite volume flow solver FASTEST including the integrated finite volume approach for the acoustics perturbations (see Sect. 2). All discretization schemes are 2nd order accurate. For a first comparison, the numbers of elements and control

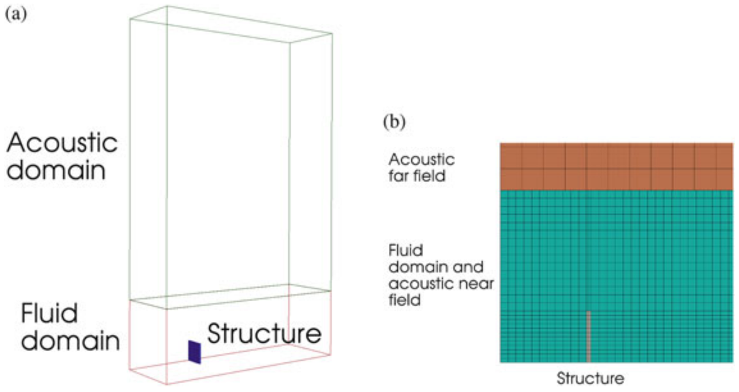

Fig. 4 Setup of our testcase scenario. (**a**) Domain description for a small near field acoustic fluid domain including the bending tower (*blue*) and a 4-fold larger acoustic far field domain on top. (**b**) Sketch of the mesh resolution in each domain, showing a matching (fluid–structure) and a non-matching (near field–far field) coupling interface. For production run, the structure is discretized with 3,600, the acoustic fluid domain in the near field with approximately 12.5 million control volumes and the acoustic domain in the far field with approximately 6 million elements of seventh order

Table 1 Overview of numerical set up for each domain in the three-dimensional bending tower testcase

Domain	Structure	Fluid	Acoustics
Material parameters	$E = 1.4 \cdot 10^6 \left[\frac{N}{m^2}\right]$	$\gamma = 1.4$	$\gamma = 1.4$
	$\nu = 0.4$	$\nu = 0.01 \left[\frac{m^2}{s}\right]$	$c = 11.8\left[\frac{m}{s}\right]$
	$\rho = 1000 \left[\frac{kg}{m^3}\right]$	$R = 1\left[\frac{m^2}{v^2 K}\right]$	$R = 1\left[\frac{m^2}{v^2 K}\right]$
			$T = 100\ [K]$
			$p_0 = 100 \left[\frac{N}{m^2}\right]$
			$\rho_0 = 1 \left[\frac{kg}{m^3}\right]$
			$v_0 = [2.3, 0, 0] \left[\frac{m}{s}\right]$
Initial condition	$u^s = [0, 0, 0]\ [m]$	$p^f = 100 \left[\frac{N}{m^2}\right]$	$p^a = 0 \left[\frac{N}{m}\right]$
		$\rho^f = 1 \left[\frac{kg}{m^3}\right]$	$\rho^a = 0 \left[\frac{kg}{m^3}\right]$
		$v^f = [2.3, 0, 0] \left[\frac{m}{s}\right]$	$v^a = [0, 0, 0] \left[\frac{m}{s}\right]$
		$B = 0\ [N]$	
		$T = 100\ [K]$	

volumes are chosen similar to those in **set a**. The pure acoustic domain in the far field is simulated with the acoustic solver Ateles in the APES framework.

As described in Sect. 3, a bi-directional implicit coupling is used for the fluid–structure interaction based on a Dirichlet-Neumann coupling. For the near field–far field interaction, we do one-way explicit coupling and the primitive flow variables density ρ, velocity v and pressure p are sent from OpenFOAM to Ateles via preCICE at the surface of the near field domain.

The derivative terms of the governing equations for the fluid domain are discretized using standard Gaussian finite volume integration, with linear interpolation from the cell centres to the cell faces. The same approach is used for the solid domain.

A relative tolerance of $1.0 \cdot 10^{-5}$ is set as a convergence criterion for the fluid–structure interface fixed-point equation including deviations both in displacement of the fluid–structure interface and the stresses acting on the interface. When convergence is reached for both variables, the data of the fluid–acoustics interface are communicated via preCICE to the acoustic domain and the next time step starts.

All simulations for the scaling results are done with **set a** on the IBM DataPlex machine SuperMUC at the Leibniz Computing Centre (LRZ) in Munich.

5.2 Numerical Results

A snapshot of a complete fluid–structure–acoustic interaction simulation, calculated with **set a**, is given in Fig. 5. The displacement of the tower structure is monitored at the center point at the top of the tower, as shown in Fig. 6. The displacement in x-direction is shown with respect to the initial condition. A periodic motion is observed for the simulated time span. Further damping of the motion of the console is expected until a steady state situation is reached. The differences between **set a** and **set b** are caused by the use of periodic boundary conditions in main flow direction in **set a** and the use of inflow velocity and zero gradient boundary conditions in **set b**. Figure 7 and 8 show the pressure, density, acoustics pressure, and acoustics density at the monitoring points A ($[3.2, 0.7, 0.0]$), B ($[3.2, 0.0, 0.8]$) and C ($[3.2, 0.0, 1.0]$) for both sets. The results of **set a** and **set b** are quantitatively in the same range. However, the pressure of the incompressible FASTEST solution is shifted, as for an incompressible flow solver, the absolute pressure level is not uniquely defined. Here, the acoustic pressure p^a has to be added to the pressure to get the same values as for the compressible OpenFOAM solution.

5.3 Scaling Results

Figure 9 shows the results for a strong scaling study of the fluid–structure–acoustics interaction testcase with **set a**. For this measurement, the fluid comprises 1.5 million cells, the solid domain consists of 2304 cells, and the acoustics mesh has 6144

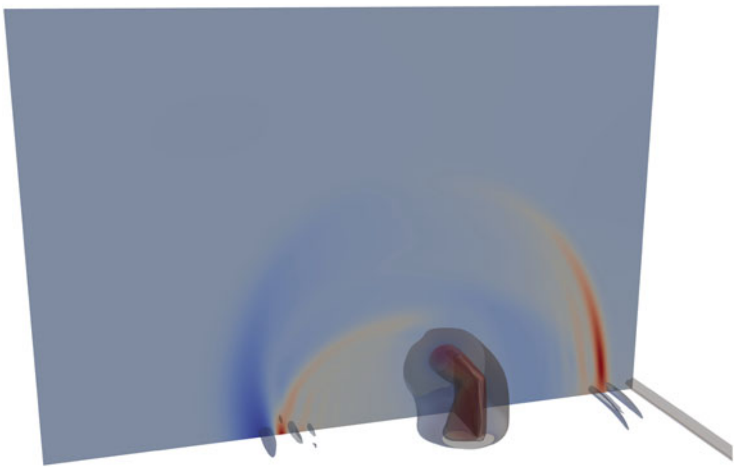

Fig. 5 Visualization of the pressure contours and Q-criterion of the velocity for the fluid flow of the three-dimensional bending tower testcase

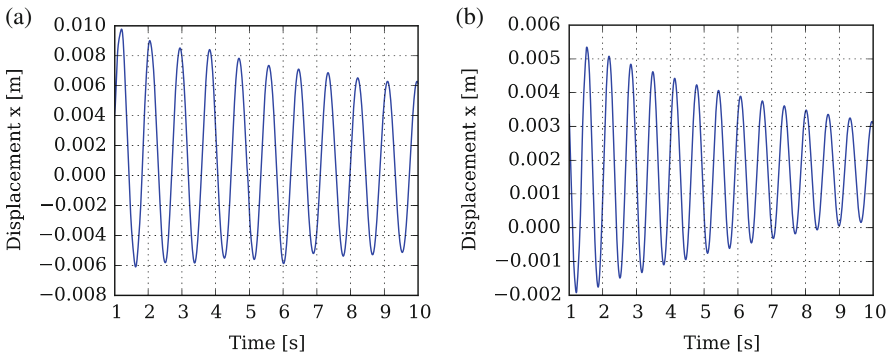

Fig. 6 Displacement of the center point of bending tower in x-direction—comparison of the results obtained for **set a** (**a**) and **set b** (**b**). (**a**) OpenFOAM setup. (**b**) FASTEST setup

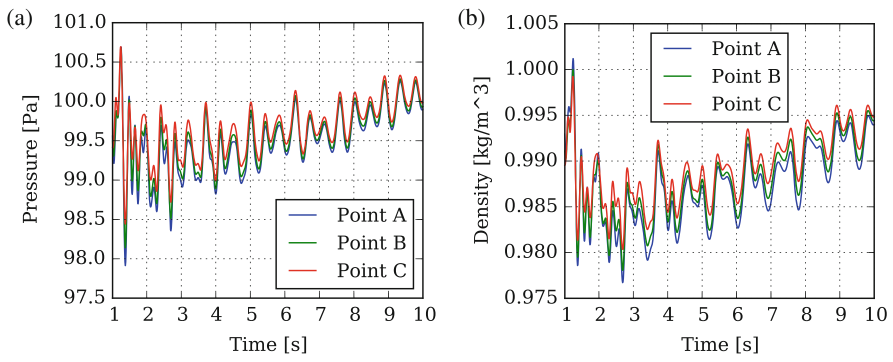

Fig. 7 Pressure and density at the monitoring points A, B and C for **set a** with OpenFOAM. (**a**) Pressure. (**b**) Density

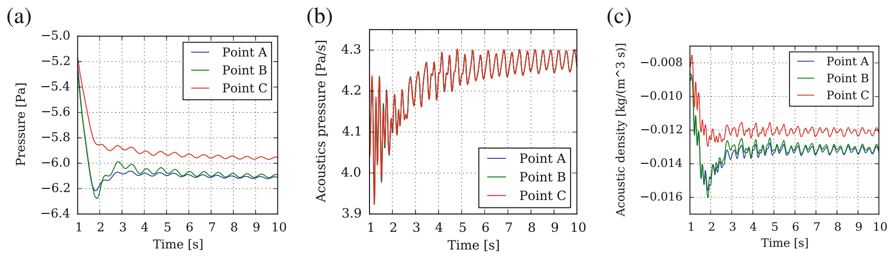

Fig. 8 Pressure, acoustic pressure and acoustic density at the monitoring points A, B and C for **set b** with FASTEST. (**a**) Pressure. (**b**) Acoustic pressure. (**c**) Acoustic density

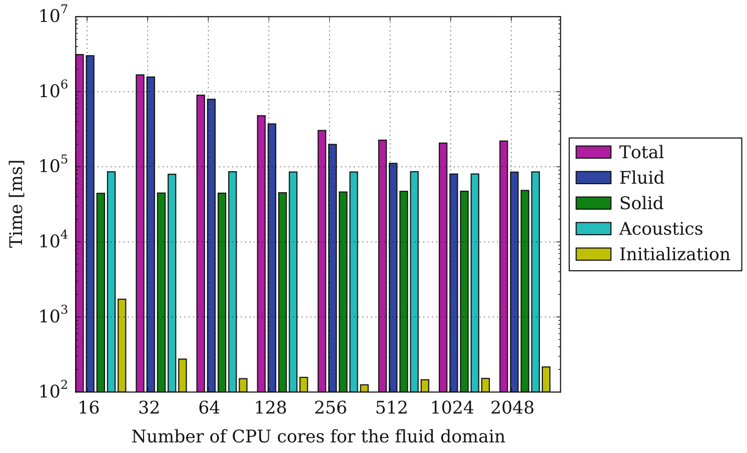

Fig. 9 Strong scaling for the fluid–structure–acoustics interaction **set a**. The number of CPU cores for the solid and acoustics domain is kept constant at two. The number of CPU cores for the flow simulation in the near field is increased

elements. The timings of the separate participants, of the initialization and the total timing are shown in the bar plot. While the number of CPU cores for the solid and acoustics domain is kept constant, the number of CPU cores for the fluid domain is increased from 16 to 2048. The fluid solver shows good performance for up to 512 MPI ranks. The initialization step of preCICE scales well for up to 64 cores. It includes the initialization of interpolation matrices between the different participants and the initialization of the socket connections which becomes more expensive when the number of ranks increases. However, the total runtime of the preCICE initialization is negligible compared to the total runtime of the simulation such that the bottleneck in this case is the scalability of the OpenFOAM compressible flow solver.

Figure 10 shows the results for a weak scaling study for **set a**. All domains are uniformly refined in each direction resulting in a factor eight increase in the number of degrees of freedom with each refinement step. Each CPU core used for the fluid domain holds approximately 25,000 control volumes and each CPU core used for the solid domain has 300 control volumes. The acoustics mesh is decomposed into 770 cells per core. It is important to note the poor performance of the initialization step of preCICE. The initialization does not scale linearly with the number of degrees of freedom on the fluid–structure and fluid–acoustics interface, and results in a bottleneck when increasing the domain size even further. The reduction of this complexity is work in progress. First ideas how to do this are presented in "Partitioned Fluid–Structure–Acoustics Interaction on Distributed Data: Coupling via preCICE".

Figures 9 and 10 show not optimal scaling behaviour. It is a three field coupling which includes different components. Each individual component has its advantages and disadvantages. Thus with OpenFOAM it is straightforward to implement a

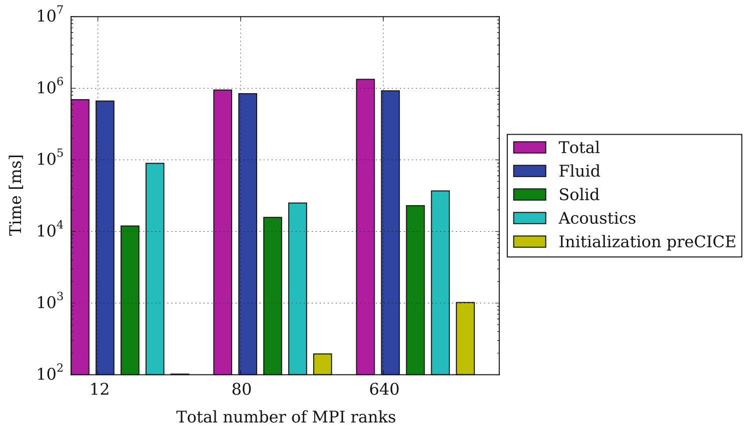

Fig. 10 Weak scaling for the fluid–structure–acoustics interaction **set a**. The number of degrees of freedom of each domain is increased proportional to the number of CPU cores

fluid–structure–acoustics interaction setup, but scalability is an issue. Other fluid solvers like Ateles, which is also capable of solving the governing flow equations, show better scaling results when coupled with preCICE [2], but does not yet include FSI. In such a complex setup, the disadvantages of one participant carry over to the complete fluid–structure–acoustics interaction simulation.

5.4 Visualization

To examine the performance of the visualization algorithm, several timing and data size measurements were taken. To judge the scalability, both a weak and a strong scaling test were performed for the bending tower testcase. For all tested configurations the compression ratio of the original data with lossless compressed TVDI was around 10:1. During the strong scaling test, high processor counts yielded a reduced compression of around 7:1. This is due to the extremely homogeneous data as can be seen in Fig. 11. The algorithm produces some minor overhead data for each process. If the data within a process are very similar (e.g., if there is only one large region), the overhead exceeds the gain by compression, and is noticeable in the reduced compression ratio.

The second measurement series took the total runtime of the visualization library call during the simulation. For all configurations, the relative runtime of the visualization relative to the simulation runtime was smaller than 1 %, and therefore achieved the goal of not impacting simulation calculations noticeably. The results presented in Table 2 show a strong scaling test (time_s) performed with a problem size of approximately 1.5 million cells. The weak scaling performance is shown in the last column denoted by time_w, using an appropriately resolved domain, ranging from 200,000 to 1.5 million cells.

Fig. 11 Volume rendering of the pressure field in the near field flow regime of the pressure field in the bending tower test case. The *left image* shows an early time step of the pressure field from the generating view, with a transfer function covering the whole pressure range. The *right image* shows the same scenario from a rotated perspective, which adequately represents the data

Table 2 Total time and compression ratio of the complete visualization algorithm per process and time step, averaged across all processes. The index s refers to strong scaling, w to the weak scaling test

# procs	time$_s$ (ms)	time$_w$ (ms)	Ratio$_s$
128	1291	271	10.2
256	691	280	10.4
512	402	312	10.5
1024	291	291	7.3

Judging from the measurements in Table 2, the calculation speed scales very well with the process count given a fixed problem size. However, for high process counts, a lot of unnecessary precomputations are done to traverse the data quickly during the raycast. If the raycasted domain contains too few cells, establishing the accelerating data structures costs more time than it saves and scaling diminishes.

As is to be expected, the timing stays nearly constant for the weak scaling test. This is not surprising considering the algorithm works exclusively on locally available data and does not employ any kind of internode communication. The variance still visible in the calculations time can be attributed to rather large fluctuations in storage access times. Note that averaging times across processes is admissible since the domain decomposition done by OpenFOAM yields evenly sized subdomains.

6 Conclusion and Outlook

We have shown the setup and numerical as well as scaling results for a complex fluid–structure–acoustics interaction testcase in a three-dimensional domain. Our partitioned simulation approach in combination with an efficient in-situ visualization has been proven to be highly flexible, efficient and scalable on parallel computer architectures.

Using a scalable coupling tool working on distributed data, presented in "Partitioned Fluid–Structure–Acoustics Interaction on Distributed Data: Coupling via preCICE", gives the benefits to efficiently reuse existing software which is adapted to the different physics (here OpenFOAM for structure and compressible flow, FEAP for structure, FASTEST for incompressible flow and acoustic near field, Ateles for acoustic far field). Using a monolithic approach is not feasible as resolving the small scales in a large domain would be highly demanding in terms of wall clock time, memory and efficiency. Exploiting appropriate methods for each physical discipline, e.g. higher order methods for the acoustic far field, facilitates such large scale simulations nowadays on supercomputers. Applying a static load-balancing based on heuristics, computational resources are used efficiently and the total time to solution is decreased. Certainly, in such a complex setup which consists of several participants, a bottleneck of one participant overshadows the whole simulation. Hence, e.g. point-to-point communication between parallel solver processes in the coupling tool described is indispensable. Further reductions of the complexity of numerical and initialization steps are work in progress such that also complex fixed-point acceleration methods and sophisticated data mapping become scalable on massively parallel architectures.

The comparison of results for simulations using different black-box solvers in terms of physical results, as well as in terms of compute time and scalability of the framework, has to be done next. For validation of the results, also experimental data shall be used.

The algorithm for the presented in-situ visualization features an easily adjustable and intuitive trade-off between image quality and runtime on a node level basis. The amount of rays used to perform the raycasting can be adjusted for each process individually. In addition, time intervals for off-loading a set of time regions can be of different length. This behavior can be exploited to further optimize the load balancing of the overall simulation: nodes known to have a high computational load from the simulation, as for example nodes at the coupling boundary, can run a low quality version of the algorithm, while internal nodes run the full quality variant. This way, idle processes can be avoided in favor of a higher accuracy and/or better compression. The current implementation would also allow for an adaptive quality adjustment, e.g., based on the idle time remaining during the last step.

Although we showed only the very simple example of a bending tower, the whole simulation framework comprising the solvers, the coupling tool and the visualization is prepared for more complex simulations such as wind turbines or fans. To actually realize such scenarios with realistic results, however, further numerical tests for easier cases have to be done. Applications with large geometry changes such as a rotating wind turbine in addition require either a different grid approach in the flow solver (Eulerian instead of ALE) or the introduction of a second mesh for the acoustic fluid domain that rotates with the turbine and is coupled to a surrounding fixed grid. A qualitative phenomenon that can be important in real-world scenarios where noise propagation in the acoustic far field includes the sound reflection at obstacles is the coupling of the acoustic far field back to the acoustic flow domain. We currently only consider a one-way coupling here. For the two-way coupling, preCICE provides a suite of coupling methods (as used for the fluid–structure interface). However, the correct modeling of coupling conditions for a two-way coupling still needs to be done. Summarizing, we can state that our simulation environment provides the technical components for even very complex simulations. The exact setup, however, has not yet been defined and modeled for all cases. The advantage over monolithic approaches that obviously offer more possibilities to taylor the methods for a specific application, is the high flexibility and generality of our approach.

Acknowledgements The financial support of the priority program 1648 Software for Exascale Computing (www.sppexa.de) of the German Research Foundation and of the Institute for Advanced Study (www.tum-ias.de) of the Technical University of Munich is thankfully acknowledged.

References

1. Blom, D.S., Krupp, V., van Zuijlen, A.H., Klimach, H., Roller, S., Bijl, H.: On parallel scalability aspects of strongly coupled partitioned fluid-structure-acoustics interaction. In: VI International Conference on Computational Methods for Coupled Problems in Science and Engineering – COUPLED PROBLEMS 2015 (2015)
2. Bungartz, H.J., Klimach, H., Krupp, V., Lindner, F., Mehl, M., Roller, S., Uekermann, B.: Fluid-acoustics interaction on massively parallel systems. In: Mehl, M., Bischoff, M., Schäfer, M. (eds.) International Workshop on Computational Engineering CE 2014. Lecture Notes in Computational Science and Engineering, pp. 151–165. Springer, Heidelberg/Berlin (2015)
3. Cardiff, P., Karač, A., Ivanković, A.: A large strain finite volume method for orthotropic bodies with general material orientations. Comput. Method. Appl. Mech. Eng. **268**, 318–335 (2014)
4. Darwish, M., Moukalled, F.: A fully coupled Navier-Stokes solver for fluid flow at all speeds. Numer. Heat Tr A.-Appl. **65**(5), 410–444 (2014)
5. Darwish, M., Sraj, I., Moukalled, F.: A coupled finite volume solver for the solution of incompressible flows on unstructured grids. J. Comput. Phys. **228**(1), 180–201 (2009)
6. FASTEST-Manual: Fachgebiet für Numerische Berechnungsverfahren im Maschinenbau, Technische Universität Darmstadt, 1st edn. (2005)
7. Fernandes, O., Frey, S., Sadlo, F., Ertl, T.: Space-time volumetric depth images for in-situ visualization. In: Proceedings of IEEE 4th Symposium on Large Data Analysis and Visualization (LDAV), pp. 59–65 (2014)
8. Frey, S., Sadlo, F., Ertl, T.: Explorable volumetric depth images from raycasting. In: Proceedings of the Conference on Graphics, Patterns and Images, pp. 123–130 (2013)
9. Hardin, J., Pope, D.: An acoustic/viscous splitting technique for computational aeroacoustics. Theor. Comput. Fluid Dyn. **6**, 323–340 (1994)
10. Hesthaven, J.S., Warburton, T.: Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications, 1 edn. Springer, New York (2007)
11. Kornhaas, M., Schäfer, M., Sternel, D.: Efficient numerical simulation of aeroacoustics for low mach number flows interacting with structures. Comput. Mech. **55**(6), 1143–1154 (2015). http://dx.doi.org/10.1007/s00466-014-1114-1
12. Lighthill, M.: On sound generated aerodynamically. I. General theory. Proc. R. Soc. A **211**, 564–587 (1952)
13. Newmark, N.M.: A method of computation for structural dynamics. J. Eng. Mech. Div.-Asce. **85**(7), 67–94 (1959)

14. Roller, S., Bernsdorf, J., Klimach, H., Hasert, M., Harlacher, D., Cakircali, M., Zimny, S., Masilamani, K., Didinger, L., Zudrop, J.: An adaptable simulation framework based on a linearized octree. In: Resch, M., Wang, X., Bez, W., Focht, E., Kobayashi, H., Roller, S. (eds.) High Performance Computing on Vector Systems 2011, pp. 93–105. Springer, Berlin/Heidelberg (2012)
15. Shen, W.Z., Sørensen, J.N.: Aeroacoustic modelling of low-speed flows. Theor. Comput. Fluid Dyn. **13**, 271–289 (1999)
16. Shen, W., Sørensen, J.: Comment on the aeroacoustic formulation of Hardin and Pope. AIAA J. **37**(1), 141–143 (1999)
17. Taylor, R.L.: FEAP – A Finite Element Analysis Program – Version 7.5 User Manual. University of California (2003). citeseer.ist.psu.edu/taylor03feap.html
18. Toro, E.F.: Riemann Solvers and Numerical Methods for Fluid Dynamics, 2 edn. Springer, Berlin/Heidelberg (1999)
19. Uekermann, B., Bungartz, H.J., Gatzhammer, B., Mehl, M.: A parallel, black-box coupling for fluid-structure interaction. In: Idelsohn, S., Papadrakakis, M., Schrefler, B. (eds.) Computational Methods for Coupled Problems in Science and Engineering, COUPLED PROBLEMS 2013. Stanta Eulalia, Ibiza (2013). http://congress.cimne.com/coupled2013/proceedings/full/p559.pdf