The 8th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2022
5–9 June 2022, Oslo, Norway

FULLY INTEGRATED MESH GENERATION IN FLUID-STRUCTURE INTERACTION

Thomas-Peter Fries, Domagoj Bošnjak, Richard Schussnig

Institute of Structural Analysis
Graz University of Technology
Lessingstr. 25, 8010 Graz, Austria
e-mail: fries@tugraz.at, www.ifb.tugraz.at

Key words: Fluid-structure interaction, FSI, multiphysics, mesh generation, transfinite maps, moving domain

Abstract. In fluid-structure interaction (FSI), the fluid and solid domains are permantently changing, coupled along a time-dependent, moving fluid-structure interface. The update of the fluid domain, i.e., in the numerical context, the mesh update is critical for robust and efficient simulations. Herein, we propose to inherently embed the mesh generation into the simulation. The FSI domain is defined based on structured building blocks that imply all the relevant information needed for the automatic mesh generation: Topology, geometry, and grading information. Transfinite maps play a crucial role for the definition of sub-meshes with any desired order and resolution in each building block. In every time step, a new mesh is generated, taking into account the deforming FSI interface. This generation is fast compared to the overall work load in each time step which is still dominated by the (iterative) solutions of the systems of equations. It is also very robust and removes any mesh entanglement by construction provided that suitable building blocks are selected once initially. Numerical results confirm the success of the proposed FSI strategy with integrated mesh generation.

1 Introduction

Mesh generation may be distinguished into structured and unstructured meshing [10]. Unstructured mesh generators only require boundary data of some domain of interest without the need to further specify topological information (e.g., the number of holes in the domain etc.). Therefore, they are very versatile and most modern mesh generators fall into this class. However, there is usually no direct control over the number of resulting elements and nodes and most algorithms only support linear (or other low-order) meshes.

On the other hand, structured mesh generators either require simpler geometries (e.g., block-shaped domains) or, for more general domains, require a set of building blocks which are topologically equivalent to the domain of interest [1, 23]. The topological representation of the domain based on coarse-scale building blocks is often semi-automatic and usually requires more user information than the unstructured generators. Nevertheless, provided that the full geometry information (in addition to only the topology) and further information concerning the grading of elements towards singularities or to resolve boundary layers near walls is given, structured mesh generators enable a fast and robust generation of higher-order and high-quality meshes.

For the same level of accuracy, they often result in far less elements than unstructured mesh generators.

In structured meshing, the number of resulting elements and nodes is directly controlled. This is of particular importance in the present paper where some parts of the boundary, i.e., the fluid-structure interface, are moving. Then, the resulting number of elements and nodes should remain fixed, so that mesh velocities as needed in the ALE framework are computed naturally without any need for search and projection schemes. It is noted that unstructured mesh generators result in different numbers of elements and nodes even for small changes of the domain of interest. Then, search and projection schemes are required in applications with moving domains which spoil the accuracy, rendering higher-order simulations virtually impossible.

In our structured mesh generator, (i) the topology is described via building blocks which may be thought of as coarse-scale, linear elements in a first step. Next, (ii) the (exact) geometry information of the domain of interest is associated to the edges or faces of the building blocks in two or three dimensions, respectively. Finally, (iii) element grading may be prescribed via the edges of the building blocks and is then carried over to the faces and block interiors thereafter. We allow for any shape of the building blocks such as triangles and quadrilaterals in two dimensions and tetrahedra, hexahedra, prisms, and pyramids in three dimensions. This naturally results into mixed higher-order elements which, in turn, may be transformed to single-element type meshes. For the geometry information, we enable the *explicit* definition via higher-order surface elements, B-splines, NURBS [13, 17] and generic function evaluations on the one hand and *implicit* definitions via level-set functions and root-search algorithms [12, 11, 18] on the other hand.

The key ingredient for the success of the structured mesh generation are so-called transfinite maps. In the first step, the geometry and grading information is taken to obtain higher-order elements on the exterior of each building block. Then, in the second step, the desired mesh is generated in the interiors of the blocks. The number of elements per building block and the desired order is prescribed by the user. In this way, meshes with millions of (higher-order) elements are generated within seconds; however, usually much less elements are required.

In applications where a fluid interacts with a structure, frequent mesh updates are required to consider for the moving fluid-structure interface. In the usual definition of the coupled boundary value problem modelling FSI, the fluid equations are formulated in an ALE framework and the solid in a total Lagrangean framework [3, 14, 15]. Therefore, mesh updates are required in the fluid domain only and mesh velocities have to be computed in this part of the overall domain. The topic of mesh updates has always played an important role in FSI simulations. A large number of approaches may be distinguished, see, e.g., [9, 19, 20, 22]. These approaches involve different complexities and computational efforts and feature some characteristic pros and cons. However, a shared feature is that they tend to be less successful for increasing element orders and that FSI simulations are still more likely to fail due to unsuccessful mesh updates rather than, say, convergence issues of the coupled iteration loop or the individual (non-linear) models for the fluid and the solid. Herein, we propose to replace any of these mesh update techniques and rather fully remesh the fluid domain in every time step of the simulation. We find that this is not only more robust and applicable to any order of the elements but also more efficient than many of the other mesh update techniques, in particular those which require the solution of large systems of equations.

The paper is organized as follows: In Section 2, the governing equations of FSI are described. Section 3 focuses on the mesh generation and the inherent coupling into the FSI framework. Numerical results are presented in Section 4 and confirm the benefits over classical mesh update techniques. Conclusions and an outlook are given in Section 5.

2 Fluid-structure interaction (FSI)

Fluid-structure interaction is a coupled problem, where some d-dimensional flow field in $\Omega_\mathrm{F} \subset \mathbb{R}^d$ exerts a load on a structure in $\Omega_\mathrm{S} \subset \mathbb{R}^d$ which, in turn, deforms and thereby changes the geometry of the flow domain, hence the flow field itself. We approach this coupled problem by a classical partitioned approach which, for the sake of brevity, is only shortly summarized here. We assume an incompressible flow modeled by the incompressible Navier-Stokes equations in Arbitrary Langrangean-Eulerian (ALE) formulation allowing for moving fluid domains. For the solid, we use a St. Venant model in total Langrangean description always referring to the initial domain. The interaction of the fluid and solid fields are sketched in Fig. 1, highlighting the iterative procedure carried out in every time step: We first compute the flow field in the fluid domain of the current time step, extract the loading on the fluid-structure interface Γ_FSI, and obtain the deformation, velocity and acceleration of the solid. From this, we extract the displacement of Γ_FSI which implies a change in the geometry of the fluid domain which classically requires a mesh update of the corresponding mesh, which, however, is replaced by a full remeshing step herein.

The instationary, incompressible Navier-Stokes equations are considered in velocity-pressure formulation. These equations are in an arbitrary Lagrangian-Eulerian (ALE) framework [8]

$$\begin{aligned} \varrho_\mathrm{F} \left(\boldsymbol{u}_{,t} + \bar{\boldsymbol{u}} \cdot \nabla_{\boldsymbol{x}} \boldsymbol{u} \right) - \nabla_{\boldsymbol{x}} \cdot \boldsymbol{\sigma} - \boldsymbol{f} &= \mathbf{0}, \quad \text{on } \Omega_\mathrm{F} \times (0,T)\,, && (1) \\ \nabla_{\boldsymbol{x}} \cdot \boldsymbol{u} &= 0, \quad \text{on } \Omega_\mathrm{F} \times (0,T)\,, && (2) \end{aligned}$$

where (1) is the momentum equation, (2) the continuity equation, $\boldsymbol{u}$ are the velocities, and ϱ_F is the fluid density. The advective velocity $\bar{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_\mathrm{M}$ takes movements of the reference ALE-coordinate system into account and $\boldsymbol{u}_\mathrm{M}$ is the mesh velocity. The stress tensor $\boldsymbol{\sigma}$ of a Newtonian fluid is defined as

$$\boldsymbol{\sigma}\left(\boldsymbol{u}, p\right) = -p\mathbf{I} + 2\mu\boldsymbol{\varepsilon}\left(\boldsymbol{u}\right), \text{ with } \boldsymbol{\varepsilon}\left(\boldsymbol{u}\right) = \frac{1}{2}\left(\nabla_{\boldsymbol{x}}\boldsymbol{u} + \left(\nabla_{\boldsymbol{x}}\boldsymbol{u}\right)^\mathrm{T}\right), \tag{3}$$

where p is the pressure, and μ the dynamic viscosity; for non-Newtonian fluids, see, e.g., [14, 15].

In the solid domain $\Omega_\mathrm{S} \subset \mathbb{R}^d$, the St. Venant model is chosen, resulting in the governing equations [2, 24]

$$\varrho_\mathrm{S} \ddot{\boldsymbol{d}} - \nabla_{\boldsymbol{X}} \cdot (\mathbf{FS}) - \boldsymbol{f} = \mathbf{0}, \quad \text{on } \Omega_\mathrm{S} \times (0,T)\,, \tag{4}$$

where ϱ_S is the density of the structure at the initial time step and $\boldsymbol{f}$ describes volume forces. The deformation gradient is $\mathbf{F} = \nabla_{\boldsymbol{X}} \boldsymbol{d} + \mathbf{I}$, the second Piola-Kirchhoff stress tensor is given as $\mathbf{S} = \lambda\left(\mathrm{tr}\mathbf{E}\right)\mathbf{I} + 2\eta\mathbf{E}$ and the Green-Lagrange strain tensor is $\mathbf{E} = \frac{1}{2}\left(\mathbf{F}^\mathrm{T}\mathbf{F} - \mathbf{I}\right)$.

A complete description of boundary conditions is omitted here for brevity, however, it is noted that fluid and solid domain share a common boundary, called the fluid-structure interface Γ_FSI. Geometrical consistency and mechanical equilibrium at the moving, hence, time-dependent $\Gamma_\mathrm{FSI}(t)$ must be ensured in the numerical treatment of the FSI problem. We use a standard

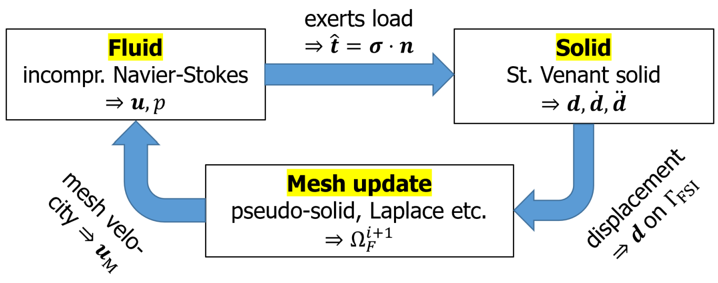

exerts load
$\Rightarrow \hat{\boldsymbol{t}} = \boldsymbol{\sigma} \cdot \boldsymbol{n}$
Fluid
incompr. Navier-Stokes
$\Rightarrow \boldsymbol{u}, p$
Solid
St. Venant solid
$\Rightarrow \boldsymbol{d}, \dot{\boldsymbol{d}}, \ddot{\boldsymbol{d}}$
Mesh update
pseudo-solid, Laplace etc.
$\Rightarrow \Omega_F^{i+1}$
mesh velocity $\Rightarrow \boldsymbol{u}_M$
displacement $\Rightarrow \boldsymbol{d}$ on Γ_{FSI}

Figure 1: FSI as a coupled problem. Herein, we propose to replace the mesh update by a full remeshing of the fluid domain.

stabilized, yet higher-order finite element framework for the spatial descrization and the finite difference method (Crank-Nicolson scheme) for the temporal descrization. The weak statement of the FSI model as required for the FEM approximation is not shown here because the focus is rather on the integrated mesh generation.

3 Mesh generation

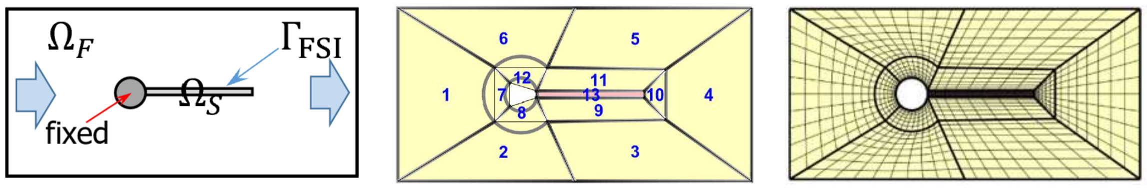
In FSI-applications, the fluid and solid domains are changing throughout the simulation due to the movement of the fluid-structure interface $\Gamma_{\mathrm{FSI}}(t)$. As the fluid model is based on the ALE framework, the corresponding fluid mesh has to be frequently updated to account for the moving interface. Herein, we propose to fully remesh the fluid domain in every time and iteration step which requires to inherently couple the mesh generator into the simulation. It is seen in Eq. 1 that the mesh velocity $\boldsymbol{u}_{\mathrm{M}}$ is required to compute the advective velocity $\bar{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_{\mathrm{M}}$. This is only simple when the number of elements and nodes remains fixed during the whole simulation; otherwise, cumbersome projections between different meshes would result and spoil the accuracy. Therefore, it is crucial to use a structured mesh generator rather than an unstructured one.

3.1 Structured mesh generation

The fluid and solid domains are represented by building blocks which may be seen as coarse-scale linear elements, see Fig. 2. Next, the (exact) geometry information is assigned to the edges or faces of the two- or three-dimensional building blocks, respectively. The geometry may either be defined explicitly via higher-order elements or NURBS, see Fig. 3(a) and (b), or implicitly via level-set functions, see Fig. 3(c) and (d). Note that the same two-dimensional geometry in Fig. 3(a) and (c) may result from both approaches. In Fig. 3(d), the geometry of a human aorta is defined implicitly based on convolution surfaces [5]. Next, grading information is associated to the edges of the building blocks in order to resolve boundary layers. Based on these three ingredients—topology, geometry and grading—higher-order and high-quality meshes are generated in each building block and combined to one final mesh, see, e.g., Fig. 2(c).

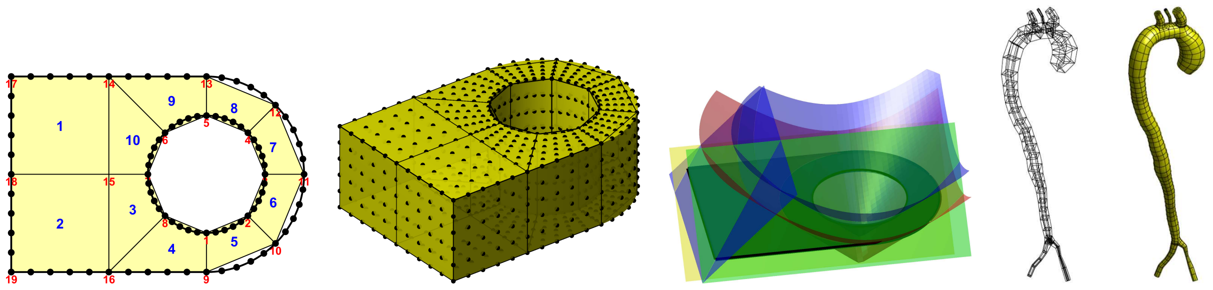
3.2 Structured mesh generation in FSI

In the present context of FSI, it is important to note that only in the initial mesh generation (at $t = 0$), the position of the fluid-structure interface $\Gamma_{\mathrm{FSI}}(0)$ is prescribed purely by given geometry information. The user may then generate this initial mesh with some desired number



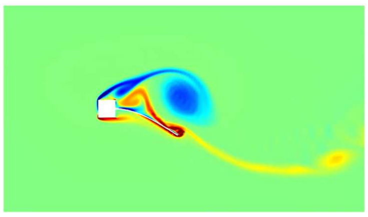
(a) FSI domain with Ω_{F} and Ω_{S} (b) building blocks (c) generated mesh

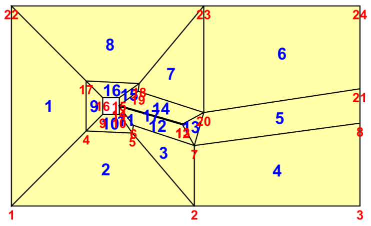
Figure 2: (a) Schematic sketch of the FSI domain (similar to [16]), (b) topology representation via building blocks including the assignment of fluid and solid blocks, (c) the generated mesh.

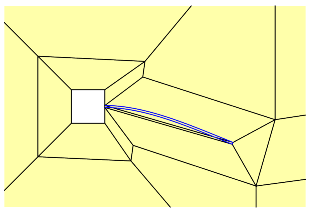


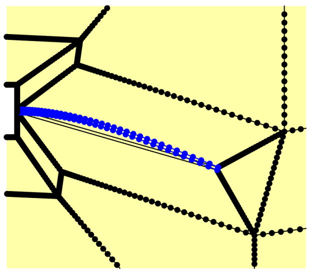
(a) 2D domain, explicit (b) 3D domain, explicit (c) 2D domain, implicit (d) 3D domain, implicit

Figure 3: Assignment of geometry data to the edges/faces of the building blocks: (a) and (b) *explicit* definition via high-order line or surface elements, resp., (c) and (d) *implicit* via level-set data.


(a) FSI snap shot


(b) building blocks


(c) FSI-interface $\Gamma_{\mathrm{FSI}}(t)$


(d) edge definition

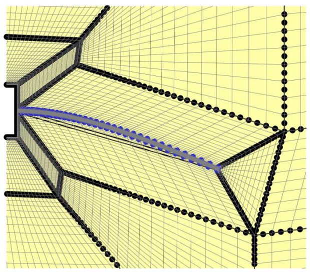

(e) generated mesh

Figure 4: The proposed integrated mesh generation in FSI: (a) Snap shot of a classical FSI benchmark with deformed flap in a flow field, (b) the corresponding coarse-scale building blocks, (c) the actual position of the moving flap, i.e., $\Gamma_{\mathrm{FSI}}(t)$, (d) representation of $\Gamma_{\mathrm{FSI}}(t)$ by the blue higher-order line elements resulting from the solid simulation, (e) the generated fluid mesh.

and order of elements per building block. In all future time and iteration steps, the interface $\Gamma_{\mathrm{FSI}}(t > 0)$ results from the displacement field of the solid, see Section 2, i.e., it is part of the solution. Then, for the geometry information of the neighboring building blocks of Γ_{FSI}, the initial geometry information must be replaced by the set of higher-order elements resulting from the solid deformation.

This is illustrated in Fig. 4 for the example of a classical FSI benchmark where a flap behind a square block is moving in a flow field [21, 16]: Fig. 4(a) shows some snap shot of the FSI-simulation with some flap deformation. The task is now to generate a fluid mesh representing this geometry. Fig. 4(b) shows the building blocks where only the end points of the flap are considered by the position of the block corners. A zoom of the situation around the flap is seen in Fig. 4(c) where the actual position of $\Gamma_{\mathrm{FSI}}(t)$ is shown in blue. Fig. 4(d) shows the nodes on the edges of the building blocks, the black ones are prescribed geometry data from the user, whereas the blue ones are rather resulting from the solid deformation. The final generated mesh near the flap is seen in Fig. 4(e).

Hence, the central aspect is that in order to fully integrate the mesh generation into the FSI simulation, some parts of the geometry definition are based on simulated data (the solid deformation at Γ_{FSI}) rather than on prescribed user data. It is noted that this generation is very efficient and robust: Structured meshes with several million of elements are generated in a few seconds. Hence, this remeshing is typically much faster than mesh update techniques which require the global solution(s) of system(s) of equations.

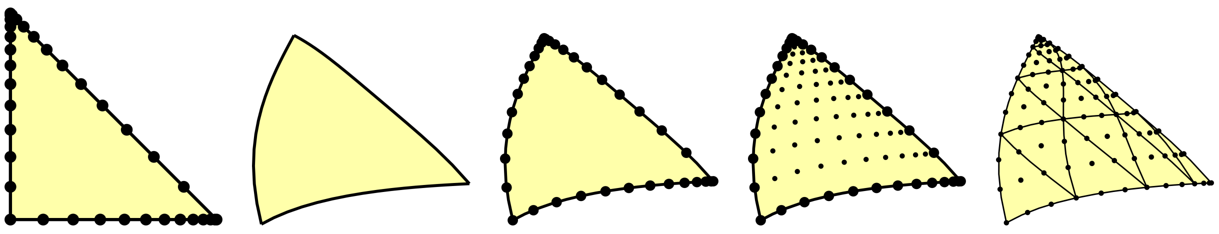


Figure 5: Procedure for the generation of sub-meshes in a triangular building block: (a) edge grading assigned to the edges of the *reference* building block, (b) prescribed geometry data at the edges, (c) edge grading in the *physical* building block, (d) generation of inner nodes using transfinite maps, (e) resulting higher-order mesh.

3.3 Transfinite maps

It remains to outline how the inner sub-meshes inside the building blocks are generated based on the geometry and grading data assigned to their exterior (boundaries). The procedure is sketched in Fig. 5 for the example of a triangular building block in which a user-defined number and order of higher-order elements is generated. The procedure falls into the following steps: Generation of *outer nodes* on the edges of the reference building block, see Fig. 5(a), according to the desired grading; the number of nodes on each edge is already related to the desired number and order of elements. Based on the geometry data assigned to the edges of the building block, see Fig. 5(b), these nodes are mapped to the physical domain, see Fig. 5(c). It is recalled that this geometry data may either refer to a prescribed fixed part of the boundary or—as in FSI—to some time-dependent interface $\Gamma_{\text{FSI}}(t)$. The next task is to generate *inner nodes* inside the building block according to the position of the outer nodes, see Fig. 5(d) for which transfinite maps are required, see below. Finally, the nodes have to be associated with a connectivity matrix relating them to higher-order elements, see Fig. 5(e).

A crucial step in this procedure is the generation of inner nodes based on the outer nodes using transfinite maps. These maps are quite trivial for tensor-product elements (here: building blocks) and are closely related to Coons patches [4] and the blending function method of Gordon [6, 7]. For other types of building blocks such as triangles and tetrahedra, transfinite maps are found, e.g., in [25]. Herein, we restrict ourselves to the definition for tensor-product elements, see Fig. 6, which is shortly summarized as follows.

Let there be four edge curves implying the outer contour of some arbitrary shaped quadrilateral with four corner nodes $\boldsymbol{x}_i^{\text{c}}$, $i = 1, 2, 3, 4$, as shown in Fig. 6(b). They are defined via parametrizations $\boldsymbol{x}_i(c_i)$ where the local coordinate $c_1 \in [-1, 1]$ is defined along the individual edges as seen in Fig. 6(a); there is an obvious relation between c_i and the reference coordinate system (r, s). The start and end points of the edges must coincide with the corner nodes $\boldsymbol{x}_i^{\text{c}}$ mentioned before. Associated with each edge, linear, one-dimensional shape functions $N_{i,1}^{\text{1D}}(c_i) = \frac{1}{2}(1-c_i)$ and $N_{i,2}^{\text{1D}}(c_i) = \frac{1}{2}(1+c_i)$ are defined. One may now generate bubble functions of the edge parametrizations as

$$\boldsymbol{x}_i^{\text{b}}(c_i) = \boldsymbol{x}_i(c_i) - \boldsymbol{x}_i(-1) \cdot N_{i,1}^{\text{1D}}(c_i) - \boldsymbol{x}_i(+1) \cdot N_{i,2}^{\text{1D}}(c_i),\ i = 1, 2, 3, 4.$$

Setting $c_1 = r$, $c_2 = s$, $c_3 = -r$, and $c_4 = -s$ results in four individual bubble functions $\boldsymbol{x}_i^{\text{b}}(r, s)$

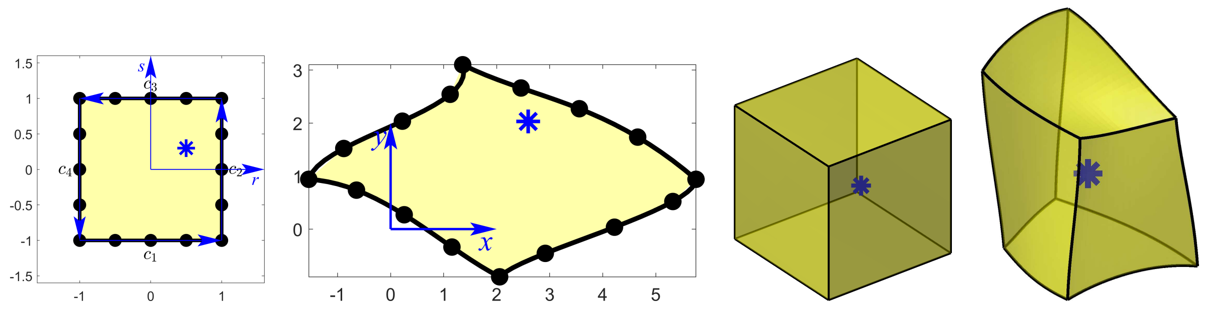


Figure 6: Transfinite maps are used to map points inside reference elements to physical elements of which only the boundary data (i.e., the exterior or hull) is given.

over the whole quadrilateral reference element as in Fig. 6(a). Introducing the bilinear shape functions in the reference quadrilateral,

$$\begin{aligned} N_1(\boldsymbol{r}) &= \frac{1}{4}(1-r)(1-s), \qquad & N_2(\boldsymbol{r}) &= \frac{1}{4}(1+r)(1-s), \\ N_3(\boldsymbol{r}) &= \frac{1}{4}(1+r)(1+s), \qquad & N_4(\boldsymbol{r}) &= \frac{1}{4}(1-r)(1+s), \end{aligned}$$

one may now define ramp functions associated with the four edges as

$$R_1 = N_1 + N_2,\ R_2 = N_2 + N_3,\ R_3 = N_3 + N_4,\ R_4 = N_4 + N_1.$$

Finally, the transfinite map for quadrilaterals is defined as

$$\boldsymbol{x}(\boldsymbol{r}) = \sum_{i=1}^{4} N_i(\boldsymbol{r}) \cdot \boldsymbol{x}_i^{\mathrm{c}} + \sum_{i=1}^{4} R_i(\boldsymbol{r}) \cdot \boldsymbol{x}_i^{\mathrm{b}}(\boldsymbol{r}).$$

Similar constructions lead to transfinite maps for hexahedral elements.

4 Numerical results

For the FSI benchmark with the flap behind a square block [21], we compare the mesh quality between two different approaches: The first approach realizes the fluid mesh update by considering the deformation of some virtual pseudo-solid in Ω_{F} [3, 19]. Zero-displacements are prescribed on all boundaries of Ω_{F} except for the fluid-structure interface Γ_{FSI}, where the displacement is extracted from the solid deformation in Ω_{S}. This approach is notorious for mesh entanglements, (locally) resulting into invalid elements featuring negative Jacobians. A crucial aspect is the assignment of element-specific stiffness parameters (Young's moduli) depending on the size and/or location of the elements. The resulting stiffness distribution used herein is seen in Fig. 7. The second approach for the mesh update is, in fact, the proposed full remeshing as outlined before.

Both results were able to generate valid meshes, however, it required significant fine-tuning in the assignment of the element stiffnesses to obtain valid meshes using the pseudo-solid approach.

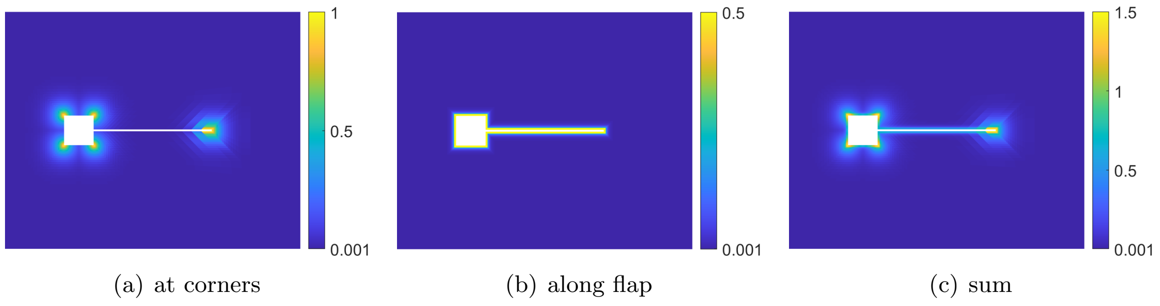


Figure 7: Distribution of Young's modulus for the pseudo-solid approach of the mesh update: (a) considering corners, (b) considering the flap, (c) resulting sum.

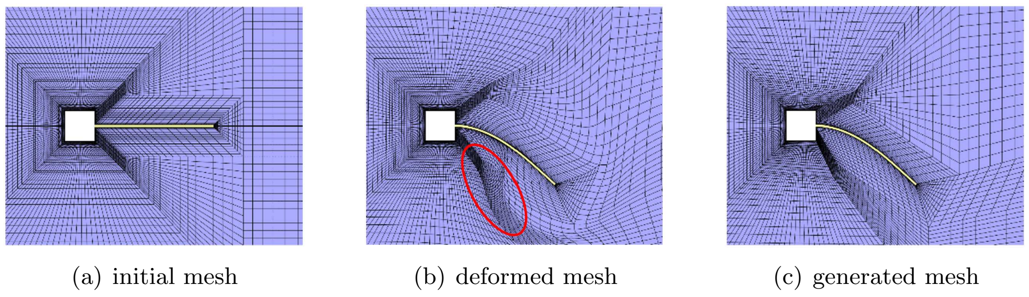


Figure 8: (a) Initial mesh, (b) deformed mesh according to the pseudo-solid approach, (c) resulting mesh for the integrated mesh generation as proposed herein.

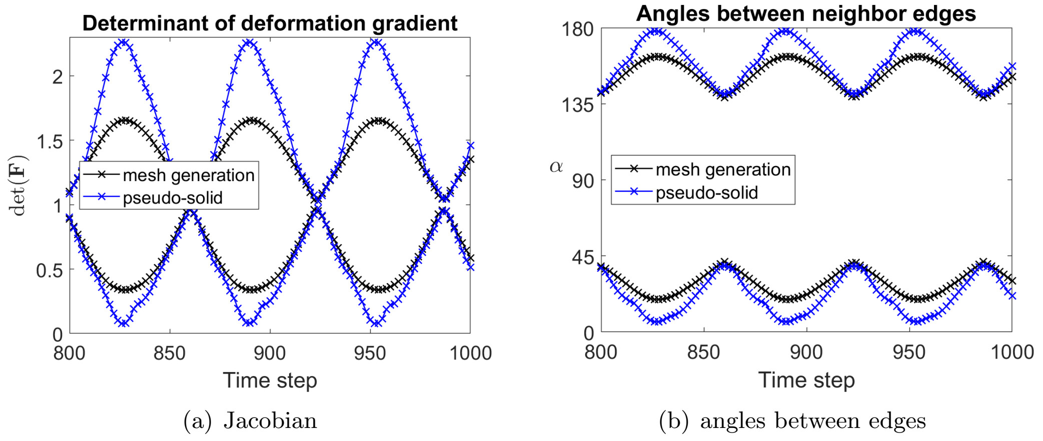


Figure 9: Time-dependent quality measures of the meshes resulting from the pseudo-solid approach (blue) and the full remeshing (black): (a) Jacobian of the deformation gradient, (b) maximum and minimum of the angles between neighbouring edges.

In contrast, the full mesh generation turned out to be much more robust without any abortions during the simulations. Some meshes near the flap are shown in Fig. 8 for one snap shot of the simulation. When comparing Figs. 8(b) and (c), it is seen that some elements in the fluid mesh are highly squeezed in the pseudo-solid approach which is not the case for the full remeshing. This is also confirmed when computing typical parameters for the assessment of the mesh quality (over time) such as the Jacobian of the deformation gradient or the maximum and minimum of the angles between neighbouring edges within all elements, see Fig. 9. As seen, the pseudo-solid approach squeezes some of the elements to almost zero-area and leads to maximum angles close to 180° and minimum angles close to 0°. In contrast, these quality measures are much better for the newly generated meshes. A number of other FSI simulations have confirmed these results but are omitted here for brevity.

5 Conclusions

We propose to integrate mesh generation into the simulation of FSI. Instead of classical strategies for the mesh update, we show that it is often faster and more robust to fully remesh the fluid domain in every time and iteration step. This requires advanced structured meshing strategies which are outlined herein: The topology is implied by building blocks, the geometry is assigned to the boundaries of these blocks and grading information is defined at the edges. Then, higher-order and high-quality meshes may be generated efficiently.

Uniting the mesh generator and FSI simulator comes at the cost of implementing the mesh generator and increasing the software complexity. The benefit are efficient, robust and higher-order accurate FSI simulations with large displacements.

REFERENCES

[1] Ali, Z.; Tyacke, J.; Tucker, P.G.; Shahpar, S.: Block Topology Generation for Structured Multi-block Meshing with Hierarchical Geometry Handling. *Procedia Engineering*, **163**,

212–224, 2016.

[2] Bathe, K.J.: *Finite Element Procedures.* Prentice-Hall, Englewood Cliffs, NJ, 1996.

[3] Bazilevs, Y.; Takizawa, K.; Tezduyar, T.E.: *Computational Fluid–Structure Interaction: Methods and Applications.* John Wiley & Sons, Chichester, 2013.

[4] Coons, S.A.: Surfaces for computer-aided design of space forms. MAC-TR-41, MIT, Cambridge, USA, 1967.

[5] Fuentes Suárez, A.J.; Hubert, E.; Zanni, C.: Anisotropic convolution surfaces. *Computers & Graphics*, **82**, 106–116, 2019.

[6] Gordon, W.J.; Hall, C.A.: Construction of curvi-linear co-ordinate systems and applications to mesh generation. *Internat. J. Numer. Methods Engrg.*, **7**, 461–477, 1973.

[7] Gordon, W.J.; Hall, C.A.: Transfinite element methods: blending function interpolation over arbitrary curved element domains. *Numer. Math.*, **21**, 109–129, 1973.

[8] Hughes, T.J.R.; Liu, W.K.; Zimmermann, T.K.: Lagrangian-Eulerian Finite Element Formulation for Incompressible Viscous Flows. *Comp. Methods Appl. Mech. Engrg.*, **29**, 329–349, 1981.

[9] Jendoubi, A.; Deteix, J.; Fortin, A.: A simple mesh-update procedure for fluid-structure interaction problems. *Computers & Structures*, **169**, 13–23, 2016.

[10] Lo, D.: *Finite Element Mesh Generation.* CRC Press, Boca Raton, 2015.

[11] Osher, S.; Fedkiw, R.P.: Level set methods: an overview and some recent results. *J. Comput. Phys.*, **169**, 463–502, 2001.

[12] Osher, S.; Fedkiw, R.P.: *Level Set Methods and Dynamic Implicit Surfaces.* Springer, Berlin, 2003.

[13] Piegl, L.; Tiller, W.: *The NURBS Book (Monographs in Visual Communication)*. Springer, Berlin, 2 edition, 1997.

[14] Schussnig, R.; Pacheco, D.R.Q.; Fries, T.P.: Robust stabilised finite element solvers for generalised Newtonian fluid flows. *J. Comput. Phys.*, **442**, 110436, 2021.

[15] Schussnig, R.; Pacheco, D.R.Q.; Fries, T.P.: Efficient split-step schemes for fluid-structure interaction involving incompressible generalised Newtonian flows. *Computers & Structures*, **260**, 106718, 2022.

[16] Schäfer, M.; Turek, S.; Durst, F.; Krause, E.; Rannacher, R.: Benchmark Computations of Laminar Flow around a Cylinder. In *Flow Simulation with High-Performance Computers II.* (Hirschel, E.H., Ed.), Vol. 48, *Notes on Numerical Fluid Mechanics (NNFM)*, Vieweg Verlag, Braunschweig, 1996.

[17] Sederberg, T.W.; Finnigan, G.T.; Li, X.; Lin, H.; Ipson, H.: Watertight trimmed NURBS. *ACM Transactions on Graphics*, **27**, 1–8, 2008.

[18] Sethian, J.A.: *Level Set Methods and Fast Marching Methods.* Cambridge University Press, Cambridge, 2 edition, 1999.

[19] Shamanskiy, A.; Simeon, B.: Mesh moving techniques in fluid-structure interaction: robustness, accumulated distortion and computational efficiency. *Comput. Mech.*, **67**, 583–600, 2021.

[20] Stein, K.; Tezduyar, T.; Benney, R.: Mesh moving techniques for fluid-structure interactions with large displacements. *J. Appl. Mech., ASME*, **70**, 58–63, 2003.

[21] Wall, W.A.; Ramm, E.: Fluid-Structure Interaction Based upon a Stabilized (ALE) Finite Element Method. *4th World Congress on Computational Mechanics – New Trends and Applications*, CIMNE, Barcelona, Spain, 1998.

[22] Wick, T.: Fluid-structure interactions using different mesh motion techniques. *Computers & Structures*, **89**, 1456–1467, 2011.

[23] Zhang, Y.; Bazilevs, Y.; Goswami, S.; Bajaj, C.; Hughes, T.J.R.: Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow. *Comp. Methods Appl. Mech. Engrg.*, **196**, 2943–2959, 2007.

[24] Zienkiewicz, O.C.; Taylor, R.L.: *The Finite Element Method*, Vol. 1-3. Butterworth-Heinemann, Oxford, 2000.

[25] Šolín, P.; Segeth, K.; Doležel, I.: *Higher-order finite element methods.* CRC Press, Boca Raton, FL, 2003.