A fast and robust hybrid method for block-structured mesh deformation with emphasis on FSI-LES applications

Shuvam Sen, Guillaume de Nayer, Michael Breuer

International Journal for Numerical Methods in Engineering, vol. 111 (3), pp. 273–300, (2017).

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2017; **111 (3)**:273–300
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nme.5465

A fast and robust hybrid method for block-structured mesh deformation with emphasis on FSI-LES applications

Shuvam Sen[1], Guillaume De Nayer[2] and Michael Breuer[2,*]

[1] *Department of Mathematical Sciences, Tezpur University, Tezpur 784028, India*
[2] *Professur für Strömungsmechanik, Helmut-Schmidt Universität Hamburg, D-22043 Hamburg, Germany*

SUMMARY

The present work introduces an efficient technique for the deformation of block-structured grids occurring in simulations of fluid-structure interaction (FSI) problems relying on large-eddy simulations (LES). The proposed hybrid approach combines the advantages of the inverse distance weighting (IDW) interpolation with the simplicity and low computational effort of transfinite interpolations (TFI), while preserving the mesh quality in boundary layers. It is an improvement over the state-of-the-art currently in use.
To reach this objective, in a first step three elementary mesh deformation methods (TFI, IDW and radial basis functions) are investigated based on several test cases of different complexities analyzing their capabilities but also their computational costs. That allows to point out the advantages of each method but also demonstrates their drawbacks. Based on these specific properties of the different methods, a hybrid methodology is suggested which splits the entire grid deformation into two steps: First the movement of the block-boundaries of the block-structured grid and second the deformation of each block of the grid. Both steps rely on different methodologies, which allows to work out the most appropriate method for each step leading to a reasonable compromise between the grid quality achieved and the computational effort required. Finally, a hybrid IDW-TFI methodology is suggested which best fits to the specific requirements of coupled FSI-LES applications. This hybrid procedure is then applied to a real-life FSI-LES case. Copyright © 2017 John Wiley & Sons, Ltd.

Received . . .

KEY WORDS: Mesh deformation; Radial basis function; Inverse distance weighting function; Greedy algorithm; Transfinite interpolation; Fluid-structure interaction; Large-eddy simulation

1. INTRODUCTION

To compile an optimal methodology for the simulation of dynamically coupled fluid-structure interactions (FSI) of lightweight flexible structures in turbulent flow, a robust and efficient method combining eddy-resolving methods such as large-eddy simulation (LES) with flexible structures was developed and validated [1]. It relies on a partitioned Arbitrary Lagrangian-Eulerian (ALE) method applying block-structured grids. The FSI-LES applications tested were three-dimensional, however, with nominally two-dimensional geometries [2, 3]. In these FSI-LES cases a simple mesh deformation method based on transfinite interpolation (TFI) was sufficient to obtain a reasonable quality of the deformed mesh. However, for practically relevant FSI applications with fully three-dimensional geometries and large deformations, this mesh deformation method is not usable anymore. Moreover, improvements of the FSI-LES method have been achieved concerning

*Correspondence to: M. Breuer, Professur für Strömungsmechanik, Helmut-Schmidt-Universität Hamburg, D-22043 Hamburg, Germany, E-Mail: breuer@hsu-hh.de

International Journal for Numerical Methods in Engineering, vol. 111 (3), pp. 273–300, (2017).

the quality of the description of the interface geometry and the kinematics by applying the Iso-Geometric Analysis (IGA) and the Exact Coupling Layer (ECL) [4, 5]. Therefore, an appropriate grid deformation algorithm has to be developed, which allows to transfer the newly achieved high grid quality of the surface mesh at the FSI interface (fluid/structure) to the CFD volume grid. Another crucial postulate is that this mesh deformation algorithm should preserve the essential quality standard of the original mesh as far as possible. For LES the orthogonality of the grid is beside the smoothness one of the most important properties, especially in the vicinity of solid boundaries, where the flow gradients are large. Finally, for the developed FSI-LES methodology it is essential to find the best compromise between the attainable mesh quality and the required CPU-time.

In the literature various approaches to generate high-quality mesh deformations can be found. Historically for structured meshes efficient algebraic interpolation functions have been proposed to move the volume mesh points solely based on the surface motion. Such methods, known as transfinite interpolation (TFI) [6, 7], are very fast since the computational effort is proportional to the number of volume grid points N_v. TFI was used with great success even for some cases with large deformations [8, 9]. However, for more complex geometries applying block-structured grids, TFI can not be used directly. The block faces and edges have to be appropriately adapted first before TFI can be applied in each block separately. It is also important to notice that these TFI methods are not suitable for unstructured meshes.

Another classical mesh deformation algorithm applicable for all kinds of grids was originally proposed by Batina [10]. This method is based on torsional springs between all neighboring mesh points. The mesh deformation is controlled by the stiffness coefficient of the fictitious springs. A substantial improvement of this method was achieved by Farhat et al. [11, 12]. The authors introduced a safeguard against invalid cross-overs preventing neighboring triangles from inter-penetrating during large deformations. Furthermore, a mixture of normal and torsional springs was used to preserve the high orthogonality of the grid in the near-wall region. Since this method is based on an analogy with torsional springs, it is often termed as an analogy method. Another analogy-based method was proposed by Lynch and O'Neill [13]. The computational domain was seen as an elastic structure undergoing stress deformation with time. Here, the governing equations of elastostatics are solved with prescribed displacements on the boundary. This so-called *pseudo-structural* method was improved by Stein et al. [14]. The authors made a systematic choice of the stiffening parameters and thus were able to substantially improve the deformed mesh quality near solid boundaries even for cases involving large boundary displacements.

Considering the problem of mesh deformation for ALE simulations as a process, in which all grid points are typically assigned to some varying velocity, Löhner and Yang [15] used Laplace equations with variable diffusivity to deform unstructured grids. This so-called process of Laplacian smoothing of the mesh velocities leads to smooth meshes but the orthogonality is not preserved. Helenbrook [16] pointed out limitations of such methods relying on second-order partial differential equations (PDE) and proposed a biharmonic operator-based method for deforming the mesh. His algorithm, which is a generalization of Laplacian-based methods, allows to control both position and wall-normal spacing of the grid points along the boundaries. These analogy-based methods have the main disadvantage that they eventually lead to a system of equations of the order $(N_v \times N_v)$. Thus, these methods are too expensive for a recurring grid deformation within three-dimensional unsteady flow simulations using a large number of mesh points.

Liu et al. [17] proposed an interesting method using *Delaunay triangulations*. In this method boundary displacements are interpolated into the whole domain using a barycentric interpolation algorithm. This method gives good results and is fast. However, for cases involving rotations the orthogonality is adversely affected.

The interpolation of data using *radial basis functions* (RBFs) is an established procedure. De Boer et al. [18] used various RBFs to interpolate known displacements at the boundary nodes to interior volume mesh points and showed that the method can handle large deformations for both two-dimensional and three-dimensional grids. A comprehensive comparison by the authors reveals that five RBFs out of fourteen investigated are capable of generating high-quality meshes

DOI: 10.1002/nme.5465

International Journal for Numerical Methods in Engineering, vol. 111 (3), pp. 273–300, (2017).

consistent with the boundary deformation. According to the authors Wendland's C^2 function with compact support is the best choice and is closely followed by the thin plate spline (TPS) function having global support. Subsequently, Zuijlen et al. [19] were successful in using a RBF-based mesh deformation for three-dimensional FSI simulations. Rendall and Allen [20] demonstrated that the mesh deformation strategy based on RBF is appropriate to preserve the global grid quality. In particular they observed that the orthogonality remains undisturbed at or near moving surfaces. Using Wendland's C^2 function they were also able to adopt this method for coupled FSI problems. The real computational costs in using RBFs arise due to the inversion of a system of equations having the dimension $(N_s \times N_s)$, where N_s is the total number of surface grid points [18, 20]. In addition, the update of the volume mesh requires a further effort of the order $(N_v \times N_s)$. Nevertheless, these computational costs are a fraction of the effort required for analogy-based methods, but it is still prohibitive for simulating three-dimensional flows in complex and realistic geometries using the FSI-LES methodology.

An algorithmic error-driven method to reduce the number of boundary data points was proposed by Rendall and Allen [21]. This so-called *greedy* algorithm was used to identify a reduced set of sample surface points in a manner that minimizes the error of the interpolated data. Sheng and Allen [22] investigated two different *greedy* algorithms in the context of RBFs on unstructured meshes, one using the exact instantaneous surface motion at each time step and the other applying a constant displacement function. For the latter the surface points are selected once before the calculation, which makes the method very efficient. Recently, the *greedy* algorithm has been further improved and a double-edged strategy was proposed by Wang et al. [23]. Significant contributions and subsequent applications of RBFs can be found in the studies of Rendall and Allen [24, 25, 26] and Bos et al. [27].

An explicit and robust alternative to the implicit interpolation procedures involved in RBFs was presented by Witteveen and Bijl [28] and Witteveen [29]. This technique denoted *inverse distance weighting* (IDW) method utilizes reciprocal distance weighted sums of the boundary node displacements to the volume vertices. Witteveen demonstrated that the IDW mesh deformation applied to FSI leads to a reduction of the computational costs compared to RBF, but reaching a comparable mesh quality and accuracy. The method does not depend on the nature of the grid, but requires a weighting function composed of several parameters which have to be estimated. Significant improvements of the IDW method were achieved by Luke et al. [30] by assuming that displacements of the boundary surface nodes can be approximated by a combination of a rigid-body rotation and a translation vector. The rigid-body motion at each boundary point is estimated by fitting a quaternion and calculating the relative displacements of the neighboring surface points. Ultimately, the displacements of the volume grid points are described by a weighted average of all boundary nodes. The weighting function is carefully designed based on a combination of parameters, which need to be estimated only once at the beginning of the interpolation process. By including the rotations in the IDW method, Luke et al. [30] significantly improved the orthogonality of the deformed mesh near boundaries. Contrary to RBF the IDW technique does not require any matrix inversion, however, the computational costs still scale with $(N_v \times N_s)$.

The idea of using quaternions for the mesh deformation dates back to the work of Samareh [31]. Maruyama et al. [32] were able to extend this idea and used it for estimating the rotation of the bounding surface. They applied quaternions for orthogonality preserving mesh deformations. Each surface node was associated with a quaternion and a translation vector and two methods were applied to obtain the deformed mesh, viz., a spherical linear interpolation and a Lie-algebra linear interpolation.

Recently, a few other novel point-by-point mesh deformation schemes have been proposed. Among those the works of Stadler et al. [33] basing on neural networks and Zhou and Li [34] relying on the disk relaxation idea deserve special attention. The main advantage of the method proposed by Stadler et al. [33] is that the volume point interpolation does not require a summation over all boundary nodes and hence is computationally efficient. The mesh deformation procedure outlined by Zhou and Li [34] starts with an initially rough distribution of the mesh points, which is

DOI: 10.1002/nme.5465

International Journal for Numerical Methods in Engineering, vol. 111 (3), pp. 273–300, (2017).

updated by the disk relaxation algorithm using repulsive and attractive interactions to give rise to a high-quality mesh.

Apart from the above-mentioned methods, hybrid procedures combining different techniques are also available in the literature. The so-called *moving sub-mesh approach* (MSA) is a mixture between a physical analogy-based pseudo-structure method carried out on a coarser grid and an interpolation from this coarse grid onto the original grid [35]. A similar idea based on a *spring* method was successfully applied by Xuan et al. [36]. Another hybrid algorithm proposed by Landry et al. [37] moves the coarse mesh by the IDW method and then interpolates it to the fine mesh using a linear interpolation. In general, hybrid methods have the advantage that an optimal compromise between computational speed and quality of the grid can be found.

In the present work a new hybrid method is introduced with special emphasis on FSI-LES applications. The aim is to arrive at an appropriate grid deformation algorithm, which is the best compromise between high mesh quality and low CPU-time for block-structured grids with orthogonal cells in the vicinity of walls. To reach this goal, three elementary mesh deformation methods are selected, evaluated and then combined. TFI is chosen for its fast speed and its high quality of the deformed mesh obtained in a single block. RBF is taken into account, because of its reputation of robustness and the very high grid quality, which can be achieved. Finally, IDW is also selected for its simplicity, speed and good control of the mesh deformation in the near-wall region preserving the grid orthogonality.

The paper is organized as follows: Section 2 mathematically discusses the TFI, RBF and IDW methods. Section 3 deals with the numerical evaluation and comparison of these elementary methods. In Section 4, the hybrid algorithm is proposed. Section 5 is concerned with the numerical validation of the proposed hybrid algorithm based on a real-life FSI-LES application.

2. MATHEMATICAL DESCRIPTION AND IMPLEMENTATION

2.1. *Transfinite interpolation (TFI)*

The transfinite interpolation was introduced by Gordon and Hall [38]. They used the term "transfinite" to describe the general class of interpolation methods, which match the original primitive function to be interpolated at a non-denumerable, i.e., transfinite number of points. Later Eriksson [6] used TFI for grid generation in CFD. When used for this purpose, TFI has the advantage to provide complete conformity to surface boundaries of the domain to be meshed. Mathematically TFI is the Boolean sum of the tensor products formed using univariate interpolations. In general, these univariate interpolation functions are restricted to the linear Lagrangian combination of surface displacements based on the arc-length from the boundary points to the node.

For a known surface displacement, the mesh deformation $\boldsymbol{d}_{i,j,k} = (d^x_{i,j,k}, d^y_{i,j,k}, d^z_{i,j,k})$ at an arbitrary volume point (i, j, k), $1 \le i \le I$, $1 \le j \le J$, $1 \le k \le K$ in three dimensions is given by

$$\boldsymbol{d}_{i,j,k} = \boldsymbol{F_1} \oplus \boldsymbol{F_2} \oplus \boldsymbol{F_3} = \boldsymbol{F_1} + \boldsymbol{F_2} + \boldsymbol{F_3} - \boldsymbol{F_1F_2} - \boldsymbol{F_2F_3} - \boldsymbol{F_3F_1} + \boldsymbol{F_1F_2F_3}, \tag{1}$$

where $\boldsymbol{F}_i$ are univariate projectors. For two-dimensional cases the interpolation is only done along two directions and thus $\boldsymbol{d}_{i,j} = (d^x_{i,j}, d^y_{i,j})$ reads

$$\boldsymbol{d}_{i,j} = \boldsymbol{F_1} \oplus \boldsymbol{F_2} = \boldsymbol{F_1} + \boldsymbol{F_2} - \boldsymbol{F_1F_2}. \tag{2}$$

The linear interpolation and the tensor products of univariates are given by [39, 40]:

$$\begin{aligned}
\boldsymbol{F}_1 &= (1-\xi_{i,j,k})\,\boldsymbol{d}_{1,j,k} + \xi_{i,j,k}\,\boldsymbol{d}_{I,j,k}, && (3)\\
\boldsymbol{F}_2 &= (1-\eta_{i,j,k})\,\boldsymbol{d}_{i,1,k} + \eta_{i,j,k}\,\boldsymbol{d}_{i,J,k}, && (4)\\
\boldsymbol{F}_3 &= (1-\zeta_{i,j,k})\,\boldsymbol{d}_{i,j,1} + \zeta_{i,j,k}\,\boldsymbol{d}_{i,j,K}, && (5)\\
\boldsymbol{F_1F_2} &= (1-\xi_{i,j,k})(1-\eta_{i,j,k})\,\boldsymbol{d}_{1,1,k} + \xi_{i,j,k}(1-\eta_{i,j,k})\,\boldsymbol{d}_{I,1,k} \\
&\quad +(1-\xi_{i,j,k})\eta_{i,j,k}\,\boldsymbol{d}_{1,J,k} + \xi_{i,j,k}\eta_{i,j,k}\,\boldsymbol{d}_{I,J,k}, && (6)
\end{aligned}$$

DOI: 10.1002/nme.5465

$$\begin{aligned}
\boldsymbol{F}_2\boldsymbol{F}_3 &= (1-\eta_{i,j,k})(1-\zeta_{i,j,k})\ \boldsymbol{d}_{i,1,1} + \eta_{i,j,k}(1-\zeta_{i,j,k})\ \boldsymbol{d}_{i,J,1} \\
&\quad +(1-\eta_{i,j,k})\zeta_{i,j,k}\ \boldsymbol{d}_{i,1,K} + \eta_{i,j,k}\zeta_{i,j,k}\ \boldsymbol{d}_{i,J,K}, \qquad (7)\\
\boldsymbol{F}_3\boldsymbol{F}_1 &= (1-\zeta_{i,j,k})(1-\xi_{i,j,k})\ \boldsymbol{d}_{1,j,1} + \zeta_{i,j,k}(1-\xi_{i,j,k})\ \boldsymbol{d}_{1,j,K} \\
&\quad +(1-\zeta_{i,j,k})\xi_{i,j,k}\ \boldsymbol{d}_{I,j,1} + \zeta_{i,j,k}\xi_{i,j,k}\ \boldsymbol{d}_{I,j,K}, \qquad (8)\\
\boldsymbol{F}_1\boldsymbol{F}_2\boldsymbol{F}_3 &= (1-\xi_{i,j,k})(1-\eta_{i,j,k})(1-\zeta_{i,j,k})\ \boldsymbol{d}_{1,1,1} + \xi_{i,j,k}(1-\eta_{i,j,k})(1-\zeta_{i,j,k})\ \boldsymbol{d}_{I,1,1} \\
&\quad +(1-\xi_{i,j,k})\eta_{i,j,k}(1-\zeta_{i,j,k})\ \boldsymbol{d}_{1,J,1} + \xi_{i,j,k}\eta_{i,j,k}(1-\zeta_{i,j,k})\ \boldsymbol{d}_{I,J,1} \\
&\quad +(1-\xi_{i,j,k})(1-\eta_{i,j,k})\zeta_{i,j,k}\ \boldsymbol{d}_{1,1,K} + \xi_{i,j,k}(1-\eta_{i,j,k})\zeta_{i,j,k}\ \boldsymbol{d}_{I,1,K} \\
&\quad +(1-\xi_{i,j,k})\eta_{i,j,k}\zeta_{i,j,k}\ \boldsymbol{d}_{1,J,K} + \xi_{i,j,k}\eta_{i,j,k}\zeta_{i,j,k}\ \boldsymbol{d}_{I,J,K}. \qquad (9)
\end{aligned}$$

The arc-length based computational coordinates (ξ, η, ζ) used to maintain the grid distribution laws of the initial grid are

$$\begin{aligned}
&\xi_{1,j,k}=0, && \eta_{i,1,k}=0, && \zeta_{i,j,1}=0,\\
&\xi_{i,j,k}=\frac{\sum_{n=2}^{i} \parallel \boldsymbol{r}_{n,j,k}-\boldsymbol{r}_{n-1,j,k} \parallel}{\sum_{n=2}^{I} \parallel \boldsymbol{r}_{n,j,k}-\boldsymbol{r}_{n-1,j,k} \parallel}, && \eta_{i,j,k}=\frac{\sum_{n=2}^{j} \parallel \boldsymbol{r}_{i,n,k}-\boldsymbol{r}_{i,n-1,k} \parallel}{\sum_{n=2}^{J} \parallel \boldsymbol{r}_{i,n,k}-\boldsymbol{r}_{i,n-1,k} \parallel}, && \zeta_{i,j,k}=\frac{\sum_{n=2}^{k} \parallel \boldsymbol{r}_{i,j,n}-\boldsymbol{r}_{i,j,n-1} \parallel}{\sum_{n=2}^{K} \parallel \boldsymbol{r}_{i,j,n}-\boldsymbol{r}_{i,j,n-1} \parallel},
\end{aligned} \qquad (10)$$

where $\boldsymbol{r}_{i,j,k}$ denotes the coordinate at (i, j, k) and $\parallel . \parallel$ describes the Euclidian norm.

In order to reduce the CPU-time consumption of a CFD simulation, the coefficients $(\xi_{i,j,k}, \eta_{i,j,k}, \zeta_{i,j,k})$ can be computed and stored at the beginning. Indeed, they are only based on the initially undeformed grid and do not vary during the time marching. TFI is a very fast method since its computational complexity linearly scales with the number of volume grid points N_v. However, it requires a certain amount of memory for storing all coefficients.

2.2. *Radial basis function (RBF) interpolation*

Applying RBFs displacements at the coordinate points denoted by $\boldsymbol{d}(\boldsymbol{r})$ are interpolated using the following function

$$\boldsymbol{d}(\boldsymbol{r}) = \sum_{n=1}^{N_s} \boldsymbol{\alpha}_n \phi(\parallel \boldsymbol{r} - \boldsymbol{r}_{s_n} \parallel) + \boldsymbol{P}(\boldsymbol{r}). \qquad (11)$$

Here, $\boldsymbol{r}_{s_n}$ are the surface nodes at which the displacements are known, ϕ is the form of the RBF being used and $\boldsymbol{P}(\boldsymbol{r})$ is a set of three polynomials. Following the work of De Boer et al. [18] $\boldsymbol{P}(\boldsymbol{r})$ is restricted to linear polynomials so that rigid-body displacements can be recovered. Given a surface displacement vector $\boldsymbol{d}_{s_n}$, $n \in \{1, 2, ..., N_s\}$ for the three-dimensional case, three different sets of coefficients $\boldsymbol{\alpha}_n$ appearing in Eq. (11) and three linear polynomials of the form $\boldsymbol{P} = \boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 x + \boldsymbol{\beta}_2 y + \boldsymbol{\beta}_3 z$ can be obtained by solving the linear symmetric indefinite system

$$\begin{bmatrix} \boldsymbol{d}_{s_1} \\ \boldsymbol{d}_{s_2} \\ \vdots \\ \boldsymbol{d}_{s_{N_s}} \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \phi_{s_1 s_1} & \phi_{s_1 s_2} & \dots & \phi_{s_1 s_{N_s}} & 1 & x_{s_1} & y_{s_1} & z_{s_1} \\ \phi_{s_2 s_1} & \phi_{s_2 s_2} & \dots & \phi_{s_2 s_{N_s}} & 1 & x_{s_2} & y_{s_2} & z_{s_2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \phi_{s_{N_s} s_1} & \phi_{s_{N_s} s_2} & \dots & \phi_{s_{N_s} s_{N_s}} & 1 & x_{s_{N_s}} & y_{s_{N_s}} & z_{s_{N_s}} \\ 1 & 1 & \dots & 1 & 0 & 0 & 0 & 0 \\ x_{s_1} & x_{s_2} & \dots & x_{s_{N_s}} & 0 & 0 & 0 & 0 \\ y_{s_1} & y_{s_2} & \dots & y_{s_{N_s}} & 0 & 0 & 0 & 0 \\ z_{s_1} & z_{s_2} & \dots & z_{s_{N_s}} & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_1 \\ \boldsymbol{\alpha}_2 \\ \vdots \\ \boldsymbol{\alpha}_{N_s} \\ \boldsymbol{\beta}_0 \\ \boldsymbol{\beta}_1 \\ \boldsymbol{\beta}_2 \\ \boldsymbol{\beta}_3 \end{bmatrix}. \qquad (12)$$

The first N_s vector equations of the above system correspond to N_s surface nodes whose displacements are known. The additional four vector equations originate from the compatibility

DOI: 10.1002/nme.5465

International Journal for Numerical Methods in Engineering, vol. 111 (3), pp. 273–300, (2017).

requirement of a zero displacement vector at the far field given by

$$\sum_{n=1}^{N_s} \boldsymbol{\alpha}_n f(\boldsymbol{r}_{s_n}) = \mathbf{0} \tag{13}$$

with $f(\boldsymbol{r}) = \{1, x, y, z\}$. Hence, the system finally is of the order $(N_s + 4) \times (N_s + 4)$ with $\phi_{s_i s_j} = \phi(\parallel \boldsymbol{r}_{s_i} - \boldsymbol{r}_{s_j} \parallel)$. Once the values of $\boldsymbol{\alpha}_n$ and $\boldsymbol{P}$ are obtained, Eq. (11) can be used to estimate displacements at the interior nodal points.

In the literature various RBFs have been used to deform a mesh [18, 20, 23, 26, 27]. These basis functions can be divided into two categories: (a) RBFs with compact support and (b) RBFs with global support. It has been noticed that the approximation properties of RBFs with global support are generally better than those with compact support [18, 23]. On the other hand, RBFs with compact support lead to a well-conditioned sparse system and are computationally more efficient [20, 23, 26]. Some popular RBFs are listed in Table I for reference. Note that RBFs with compact support are scaled with a support radius R and are set to zero outside R.

Table I. Radial basis functions.

Name	Abbreviation	$\phi(\boldsymbol{r})$	Support
Thin plate spline	TPS	$\parallel \boldsymbol{r} \parallel^2 \ln(\parallel \boldsymbol{r} \parallel)$	Global
Wendland's C^0	W C^0	$(1- \parallel \boldsymbol{r} \parallel)^2$	Compact
Wendland's C^2	W C^2	$(1- \parallel \boldsymbol{r} \parallel)^4(4 \parallel \boldsymbol{r} \parallel +1)$	Compact
Continuous TPS C^1	CTPS C^1	$1 + \frac{80}{3} \parallel \boldsymbol{r} \parallel^2 -40 \parallel \boldsymbol{r} \parallel^3 +15 \parallel \boldsymbol{r} \parallel^4 -\frac{8}{3} \parallel \boldsymbol{r} \parallel^5 +20 \parallel \boldsymbol{r} \parallel^2 \ln(\parallel \boldsymbol{r} \parallel)$	Compact
Continuous TPS C^2_a	CTPS C^2_a	$1 - 30 \parallel \boldsymbol{r} \parallel^2 -10 \parallel \boldsymbol{r} \parallel^3 +45 \parallel \boldsymbol{r} \parallel^4 -6 \parallel \boldsymbol{r} \parallel^5 -60 \parallel \boldsymbol{r} \parallel^3 \ln(\parallel \boldsymbol{r} \parallel)$	Compact
Continuous TPS C^2_b	CTPS C^2_b	$1 - 20 \parallel \boldsymbol{r} \parallel^2 +80 \parallel \boldsymbol{r} \parallel^3 -45 \parallel \boldsymbol{r} \parallel^4 -16 \parallel \boldsymbol{r} \parallel^5 +60 \parallel \boldsymbol{r} \parallel^4 \ln(\parallel \boldsymbol{r} \parallel)$	Compact

For solving the resulting system of equations an iterative Krylov subspace based solver called *minimum residual* (MINRES) [41] is preferred. MINRES is particularly suitable since the associated system of equations is symmetric. The system is assumed to be converged if the residual falls below a predefined tolerance ε_c. Note that for small problems (see Section 3.1) a direct solver such as LU decomposition is advantageous in terms of required CPU-time. Furthermore, no dependency on the matrix characteristics of different RBFs is observed. However, for practical applications involving large systems of equations, direct solvers become unfeasible.

The solution of the system of equations (12) is computationally intensive. To reduce the computational effort Rendall and Allen [21] introduced a data reduction strategy called the *greedy* algorithm. They discussed three different implementation perspectives for this algorithm and were able to significantly reduce the computational cost. The single point *greedy* algorithm [21] starts with a single boundary point randomly chosen and calculating the solution of this system based on prescribed displacements. It then calculates the largest error between the exact and the approximate system. Based on this largest error and a predefined tolerance ε_g the algorithm updates the coefficients of the reduced system, which may lead to the addition of further points in the already chosen list of so-called *greedy* points. Note that a *greedy* point is added when the corresponding coefficient is updated for the first time. However, the main deficiency of the *greedy* algorithm is that it fails to take the consequence of correcting a single equation of the system on the remaining equations of this system into account.

The *greedy* algorithm was speed up by Wang et al. [23] using a double-edged strategy, where a second search for the largest error is required in an approximately conjugate direction. In the present work a *hybrid greedy* algorithm [21] is implemented. This algorithm starts by carrying out a *greedy* point selection procedure for m times, where m is some predefined number. As noted by Rendall and Allen [21] the *greedy* algorithm visits a point many times and hence the number of executions is

DOI: 10.1002/nme.5465

not equal to the total number of points chosen. Next the system consisting of so-called active *greedy* points is inverted. The process continues iteratively until the maximum absolute error reduces to the predefined tolerance ε_g.

Here it should be noted that the *greedy* algorithm does not work with some global RBFs, in particular the thin plate spline (TPS). A point reduction algorithm for TPS, if it exists, is not known.

2.3. *Inverse distance weighting (IDW) interpolation*

IDW interpolation, first introduced by Shepard [42], was adopted for mesh deformation by Witteveen [29]. In its generality this method interpolates a function at a given point $\boldsymbol{r}$ based on the weighted average of known values of scattered data points. Witteveen proposed that for a given boundary displacement vector $\boldsymbol{d}_{s_n}$, $n \in \{1, 2, ..., N_s\}$. The displacements at the interior nodes can then be calculated as

$$\boldsymbol{d}(\boldsymbol{r}) = \sum_{n=1}^{N_s} w_n(\boldsymbol{r})\, \boldsymbol{d}_{s_n} \Big/ \sum_{n=1}^{N_s} w_n(\boldsymbol{r}) \;, \tag{14}$$

where $w_n(\boldsymbol{r})$ is the weighting function often expressed as

$$w_n(\boldsymbol{r}) = \| \boldsymbol{r} - \boldsymbol{r}_{s_n} \|^{-p} \;. \tag{15}$$

Witteveen used different values of $p \in \{1, 2, 3, 4\}$ depending on the nature of the boundary node, i.e., whether the boundary node is dynamic or static as well as translating or rotating.

Luke et al. [30] modified the IDW method by applying the improved weighting function

$$w_n(\boldsymbol{r}) = A_n \left[\left(\frac{L_{\text{def}}}{\| \boldsymbol{r} - \boldsymbol{r}_{s_n} \|} \right)^a + \left(\frac{\alpha L_{\text{def}}}{\| \boldsymbol{r} - \boldsymbol{r}_{s_n} \|} \right)^b \right], \tag{16}$$

where A_n is the area weight of the boundary node n. They also suggested the best values for the parameters a and b as 3 and 5, respectively. L_{def} and α are defined as

$$L_{\text{def}} = \max_{n=1}^{N_s} \| \boldsymbol{r}_{s_n} - \boldsymbol{r}_{\text{centroid}} \| \qquad \text{where} \qquad \boldsymbol{r}_{\text{centroid}} = \frac{1}{N_s + N_v} \sum_{m=1}^{N_s+N_v} \boldsymbol{r}_m \;, \tag{17}$$

$$\alpha = \frac{\eta}{L_{\text{def}}} \max_{n=1}^{N_s} \| \boldsymbol{d}_{s_n}(\boldsymbol{r}) - \boldsymbol{d}_{\text{mean}} \| \quad \text{with} \quad \boldsymbol{d}_{\text{mean}} = \sum_{n=1}^{N_s} a_n\, \boldsymbol{d}_{s_n}(\boldsymbol{r}) \quad \text{and} \quad a_n = \frac{A_n}{\sum_{m=1}^{N_s} A_m}. \tag{18}$$

Following Luke et al. [30] the value of the parameter is set to $\eta = 5$. L_{def} denotes the maximum distance of any mesh node from the mesh centroid and α is a parameter, which provides the relative weight of the nearby nodes in comparison to the distant ones. Equation (18) is only one option and the user is free to set a fixed value of α based on his need.

Furthermore, Luke et al. [30] improves the quality of their deformed mesh by interpolating not the displacements, but the rotations and the translations resulting from the splitting of the displacements. However, it is worthwhile to point out that the IDW method loses its *extreme conserving* property once rotations are included. Denoting the introduced rotation matrix $\boldsymbol{M}_s$ and the associated translation vector as $\boldsymbol{t}_s$ for a surface boundary point having the coordinate $\boldsymbol{r}$ in the original mesh, the displacement field $\boldsymbol{d}_s(\boldsymbol{r})$ is defined as

$$\boldsymbol{r} + \boldsymbol{d}_s(\boldsymbol{r}) = \boldsymbol{M}_s \boldsymbol{r} + \boldsymbol{t}_s \;. \tag{19}$$

Following the work of Maruyama et al. [32] an ideal choice to estimate the rotation matrix is to define quaternions at each boundary mesh point. In terms of a quaternion Q defined at a boundary node, Eq. (19) can be rewritten as

$$R_d = Q\, R_u\, Q^* + T \;, \tag{20}$$

DOI: 10.1002/nme.5465

where $R_u = [0, \boldsymbol{r}]$, $R_d = [0, \boldsymbol{r} + \boldsymbol{d}]$ and $T = [0, \boldsymbol{t}]$ are the quaternions associated with the undeformed position vector, the deformed position vector and the translation vector, respectively. The subscripts u and d are used to indicate the undeformed and deformed states, respectively. The subscript s used to denote surface boundary points is dropped for the sake of brevity.

Q and T at each boundary node can be determined by using the steps outlined by Maruyama et al. [32] and briefly reproduced in Appendix A. Note that T is coordinate dependent since the rotation is calculated with respect to a local coordinate system (see example in Appendix B). To eliminate this coordinate dependence, a strategy called *transformation division* algorithm was proposed by Maruyama et al. [32] (see Appendix A). For large deformations this *transformation division* process is an absolute necessity as seen in Appendix B. However, if the IDW procedure relies on the last deformed mesh and the new displacements at the boundary nodes are small, this procedure can be omitted (see Appendix B). For example, in case of unsteady FSI-LES computations, very small time steps are typically used and thus marginal deformations appear at the boundaries of the structure between two time steps, so that the *transformation division steps* can be skipped.

Once the rotation quaternions and the translation vectors are known at the surface nodes, they can be interpolated to the volume nodes by using the IDW interpolation methodology given by Eq. (14). Note that the components of the rotation quaternions at the volume grid points have to be normalized before finally estimating the displacement vectors.

In the context of block-structured grids and a parallelization strategy based on domain decomposition, the application of IDW is done in the following steps: At the beginning of the simulation all fixed and moving boundary points are marked and saved in two different arrays for optimization purpose. These arrays have to be known by all processors. Then, at each time step the quaternions and translations of all boundary points N_s are evaluated as explained above and shared by all processors. Finally, each processor executes a loop over all volume grid points of its own block evaluating the displacements based on Eq. (14). This explains the fact that the computational effort of IDW scales according to $N_s \times N_v$, since the sum of all volume grid points contained in all blocks is N_v.

3. EVALUATION OF THE ELEMENTARY METHODS BASED ON TEST CASES

In this section the elementary mesh deformation methods described above are evaluated on a few test cases of rising complexity. To provide a global measure of the mesh quality, the harmonic mean value

$$\langle f_{\text{skew}} \rangle = \begin{cases} \left(\frac{1}{N_{cv}} \sum_{k=0}^{N_{cv}-1} \frac{1}{f_{\text{skew}_k}} \right)^{-1} & \text{if } 0 < f_{\text{skew}_k} \leq 1 \text{ for all } k = 0, 1, ..., N_{cv} \\ 0 & \text{if } f_{\text{skew}_k} = 0 \text{ for any } k = 0, 1, ..., N_{cv} \end{cases}$$

is defined following the work of Knupp [43] where f_{skew_k} is the *skew quality metric* of the k-th control volume of the mesh having N_{cv} control volumes. $\langle f_{\text{skew}} \rangle = 1$ will denote perfect orthogonality of the mesh.

For comparison of the computational effort of the different methods the floating-point operations and CPU-times are measured on the same empty machine with identical compiler options. The floating-point operations are evaluated by the open-source tool likwid[†] [44] and given in Giga-FLOP (GFLOP). The CPU-times reported below are scaled by the CPU-time of the fastest successful method leading to relative computational costs nearly independent of the machine.

3.1. Problem 1: Deformation of a two-dimensional square region

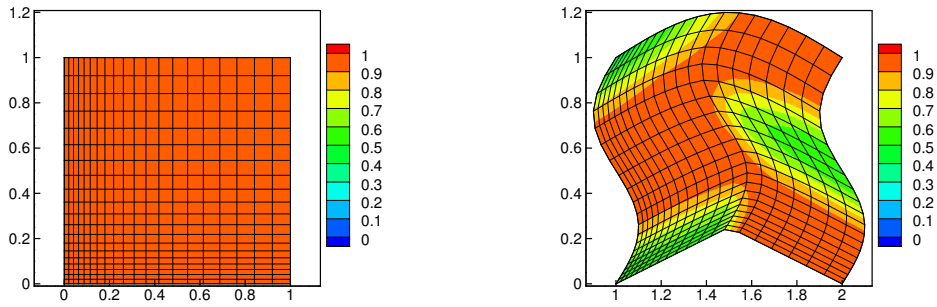
3.1.1. Description of the problem As depicted in Fig. 1 a square with the center at (0.5, 0.5) and a prescribed deformation on all sides of the domain is considered. Obviously, a sharp angle is generated at the bottom. On the other three sides the deformations are according to trigonometric

[†]https://github.com/RRZE-HPC/likwid/

DOI: 10.1002/nme.5465

functions in order to vary the angles and the overall shape. On the left and right boundaries identical displacements given by $0.1\,\sin(2\pi y)$ are applied, whereas on the top a deformation according to $0.2\,\sin(\pi x)$ is used. Moreover, on all boundary points a unit translation along the x-direction is prescribed. The original grid consisting of 21×21 grid points is clustered at the bottom and left wall. The undeformed grid is shown in Fig. 1(a). Computations are carried out for this coarse grid and a finer one with 301×301 grid points. The main objective is to evaluate the performance of TFI, RBF and IDW in a situation, which involves both contraction and relaxation. Also of interest is to see the effect on grid clustering. Furthermore, the orthogonality preserving nature of the schemes and its effect on the interior nodes is analyzed.

3.1.2. TFI mesh deformation For this type of deformation involving a convex undeformed domain, TFI is expected to perform well. The deformed grid obtained by TFI is depicted in Fig. 1(b). The color contours show the *skew quality metric* of the grid f_{skew}. Values of unity define the optimum and decreasing values represent a deterioration of the orthogonality. Apparently, the translation is recovered with ease. However, the orthogonality is deteriorated, which is particularly evident in the lower left portion. Although high *skew metric* values are preserved at a large portion of the interior nodes, overall it is clear that TFI does not maintain the high level of orthogonality of the original grid. Nevertheless, the clustering of the undeformed mesh is preserved. A benefit of the extreme preserving property of TFI is that all grid points are retained inside the boundary without any cross-over. The main advantage of TFI is its very low computational effort (see Table II).



(a) Undeformed mesh. (b) Deformed mesh and f_{skew} contour.

Figure 1. *Problem 1:* Undeformed mesh and TFI deformation.

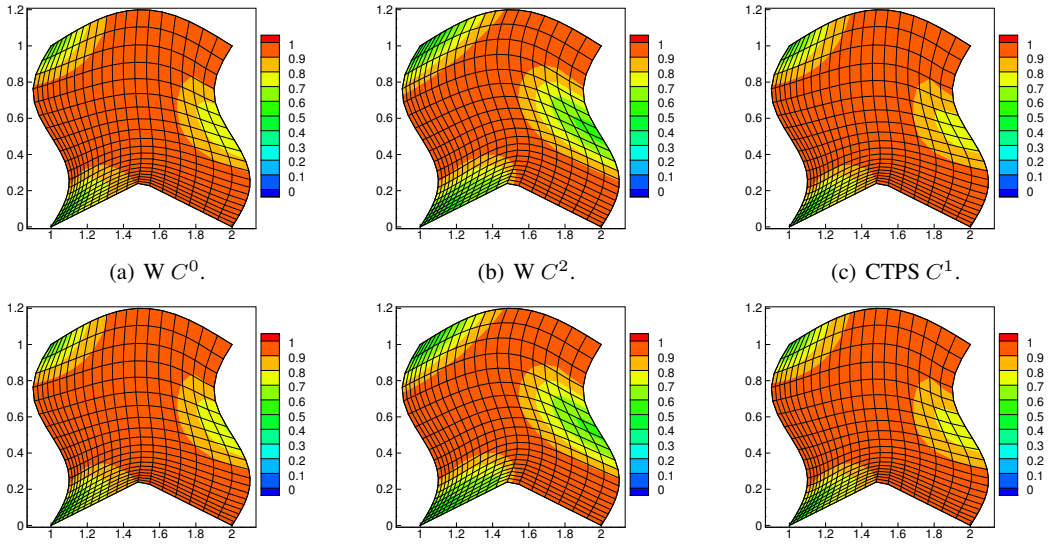
Table II. *Problem 1:* Comparison of the floating-point operations (GFLOP), the relative computational costs and global measure of the mesh quality $\langle f_{\text{skew}} \rangle$. All computations done using the fine grid with 301×301 grid points.

Method		GFLOP	Rel. comp. costs	Remark	$\langle f_{\text{skew}} \rangle$
TFI	TFI-2D	0.014	1.00		0.76
RBF	W C^0	10.42	20.04		0.86
	W C^0 with *greedy* (484)	1.54	5.52		0.86
	W C^2	1207.18	1598.56		0.80
	W C^2 with *greedy* (268)	10.22	15.39		0.80
	CTPS C^1	88.95	121.45		0.86
	CTPS C^1 with *greedy* (329)	4.51	9.02		0.86
	TPS	108.92	144.61		0.82
IDW	Without rotation computation	2.42	4.45		0.89
($\alpha = 0.1$)	With 25 *transf. div.* steps	59.34	88.97	Ortho. preserved	0.84
	With 500 *transf. div.* steps	1182.79	1759.77	near boundaries	0.86

DOI: 10.1002/nme.5465

3.1.3. RBF mesh deformation Since the main interest is to compare the number of floating-point operations and the computational costs of various RBFs having both global and local support, a uniform radius of influence of R = 1.5 is chosen for all RBFs with local support. This choice covers the entire domain. For all cases considered, convergence is assumed when the residual falls below $\varepsilon_c = 10^{-12}$ within the MINRES solver.

The grid deformed by Wendland's C^0 (W C^0) function is depicted in Fig. 2(a), while Fig. 2(d) uses the same basis function but with the *hybrid greedy* algorithm. Similarly, Figs. 2(b) and 2(e) show the results of Wendland's C^2 (W C^2) function without and with the *greedy* algorithm, respectively. The grids deformed by the continuous thin plate spline (CTPS) C^1 function are displayed in Figs. 2(c) and 2(f), where the *greedy* algorithm is applied in the second case. All RBFs used here fully recover the translation. Concerning the resulting skewness f_{skew}, Fig. 2 shows that the W C^2 function leads to slightly worse values than for the two other RBFs. That is also visible in Table II based on the global measure of the mesh quality $\langle f_{\text{skew}} \rangle$. This observation is in contradiction to De Boer et al. [18] who identified W C^2 as the best choice for RBFs with local support.



(a) W C^0. (b) W C^2. (c) CTPS C^1.

(d) W C^0 with *hybrid greedy* algorithm. (e) W C^2 with *hybrid greedy* algorithm. (f) CTPS C^1 with *hybrid greedy* algorithm.

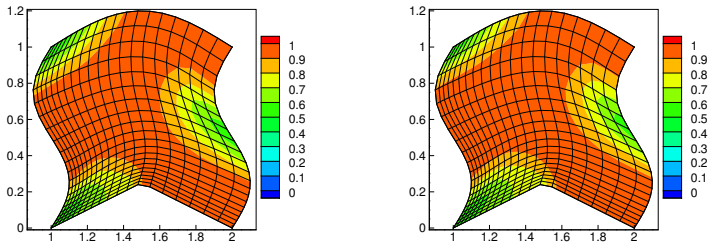
Figure 2. *Problem 1:* RBF mesh deformation with compact support (21×21 grid) and f_{skew} contour.

For the *hybrid greedy* algorithm the value of m is set to the number of boundary points and the tolerance limit ε_g defining the maximum absolute error to 10^{-2}. The *hybrid greedy* method works well for all three RBFs with local support. A strong reduction of the floating-point operations and of the CPU-time is observed in Table II. Furthermore, based on Fig. 2 and the global quality measure $\langle f_{\text{skew}} \rangle$ listed in Table II it is also evident that the quality of the mesh deformation only marginally differs between the cases with or without *greedy* selection. To give an idea about the efficiency of this algorithm, the number of points selected by the *greedy* procedure is listed in parentheses in Table II. This has to be compared to the entire number of boundary points $N_s = 1200$ for the fine grid. Among the three local RBFs the hybrid node selection algorithm works best with W C^2.

Figure 3 represents the grid deformed using the global TPS basis function providing a comparison of the coarse (21×21) and the fine grid (301×301). Note that in Fig. 3(b) only every 15^{th} grid point in both directions is shown. The quality of mesh deformations based on RBFs is theoretically independent of the number of nodes and their clustering. Thus, the deformed meshes are nearly identical. The same is true for the other RBFs not shown here. Figs. 2–3 and Table II reveal that all four RBFs are overall leading to a reasonable mesh deformation but none is able to maintain the orthogonality particularly at the challenging bottom side. The grid clustering is best preserved

DOI: 10.1002/nme.5465

by W C^2 and TPS. W C^0 and CTPS C^1 tend to shrink the region of clustered points under severe deformation.

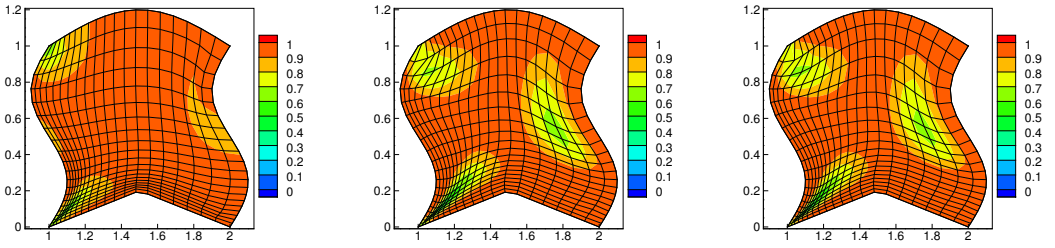


(a) 21×21 grid. (b) 301×301 grid.

Figure 3. *Problem 1:* RBF mesh deformation with global support (TPS) and f_{skew} contour. (For the fine grid only every 15^{th} grid point in both directions is shown).

Another interesting issue is the computational effort required by different RBFs. W C^2 takes much more time, whereas W C^0 is computationally the fastest. It is also seen that the effort required by CTPS C^1 and TPS is comparable. The matrices associated with different RBFs have a higher condition number along with more clustering of the eigenvalues near zero for the matrix corresponding to W C^2. This explains its higher computational effort using the MINRES solver. Replacing the iterative solver by a direct LU decomposition the CPU-time required for the solution of the system does not depend on the chosen RBF anymore. The relative computational costs (compare values in Table II) are 12.96, 13.08, 21.07 and 16.98, for W C^0, W C^2, CTPS C^1 and TPS, respectively. Thus, for the present two-dimensional problem the direct solver leads to a drastic gain in CPU-time for the same grid quality.

3.1.4. IDW mesh deformation Finally, the IDW method is applied. Three different cases are considered to understand various effects of IDW. The first case depicted in Fig. 4(a) is obtained by directly interpolating the displacements with the weighting function of Luke et al. [30]. The two other cases (Figs. 4(b) and 4(c)) are based on the interpolation of the translations and rotations (represented by quaternions) with the same weighting function. Here the parameter α is kept fixed and set to 0.1 which is especially suited for producing grid deformations of high quality according to Luke et al. [37]. Similar results are obtained with α dynamically predicted based on Eq. (18).



(a) IDW without rotation. (b) IDW using 25 *transf. div.* steps. (c) IDW using 500 *transf. div.* steps.

Figure 4. *Problem 1:* IDW mesh deformations ($\alpha = 0.1$) and f_{skew} contours.

Obviously, IDW is able to recover the translation without any difficulty. Although the clustered nature of the original grid is apparently preserved in Fig. 4(a), it has to be noticed that the orthogonality near the boundaries is not conserved. Computation leading to Figs. 4(b) uses 25 *transformation division* steps. Figure 4(c) is obtained using even 500 *transformation division* steps in order to check the stability of the *transformation division* algorithm. It is found that the algorithm performs in a sound manner with minor improvements of the mesh rendering the algorithm reliable. Both grids make an effort to preserve the orthogonality near all boundaries. That is an advantage

DOI: 10.1002/nme.5465

International Journal for Numerical Methods in Engineering, vol. 111 (3), pp. 273–300, (2017).

in comparison with the case without rotation. Although at some interior points, for example at the lower left side, the orthogonality is not preserved, such a situation is inevitable due to the singular character of the deformation. Thus, the fundamental property of the IDW procedure with rotation is the preservation of the orthogonality and the clustered nature of the grid near boundaries even under strained conditions.

As expected, a substantial increase of the computational effort is noted for the application of a large number of *transformation division* steps in Table II. It is found that the number of floating-point operations for IDW is directly proportional to the number of *transformation division* steps employed.

3.1.5. Summary All methods applied recover the translation perfectly. A comparison of the global skewness $\langle f_{\text{skew}} \rangle$ obtained by TFI, RBF and IDW reveals that all four RBFs deform the mesh slightly more appropriate than TFI. IDW possesses a good orthogonality preserving property near boundaries, better than TFI and RBF. TFI is faster than IDW, which is faster than pure RBFs. However, it is worthwhile to point out that by usage of the *greedy* algorithm the interpolations using compact RBFs become computationally much more efficient and reach a similar level as IDW. For the present small problem size a direct solver speeds up the RBF method significantly, but practical applications necessitate an iterative solver. The IDW computation based on rotation requires many steps of the *transformation division* technique, which decelerates the mesh deformation significantly. However, as discussed in Section 2.3 and shown in Appendix B, this drawback is eliminated for time-resolved predictions relevant for the present area of interest, i.e., LES of turbulent flows.

A remark concerning the two measures in Table II to evaluate the computational effort has to be added. The number of floating-point operations has the advantage that it is independent of the machine architecture. A clear disadvantage is that it is fully unclear what the compiler makes out of the code and whether some operations may be vectorized or running in parallel. The relative CPU-time, however, comprises the floating-point operations but also other important issues such as the memory access leading to the overall performance of the algorithm. For example, comparing TFI and W C^0 yields a ratio of floating-point operations of about 720. The relative computational costs, however, increase only by a factor of 20, since the RBF algorithm runs more efficiently with a significantly increased number of floating-point operations per second. Therefore for practical purposes, it represents a more reliable measure of the computational effort and thus is solely used below.

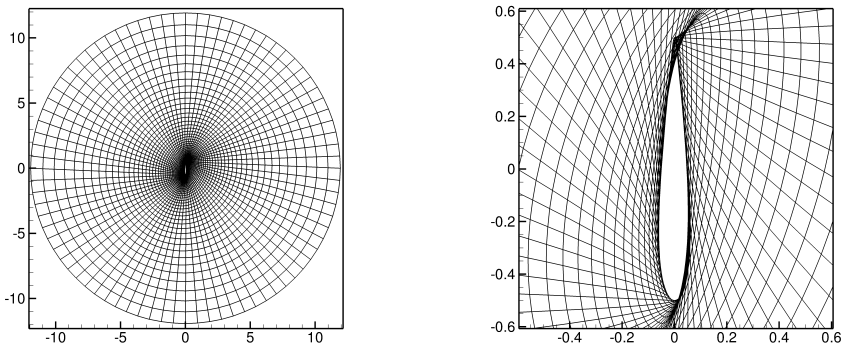
3.2. Problem 2: Rotation and deformation of an airfoil

3.2.1. Description of the problem To evaluate the three elementary mesh deformation strategies in situations involving large rotation of a body with a sharp angle, a 90° rotation of an elliptic airfoil around the center of the profile is considered. The major and minor axes of the profile have a length of 1 and 0.1, respectively. When this airfoil rotates about the origin of the coordinate system in a unit step, it deforms into a NACA 0012 airfoil of the same chord length c. The initial undeformed O-grid and a reference final grid are both generated using conformal transformations [45]. The grid points at the outer radius ($r = 12\,c$) remain fixed during the deformation process. The grid consists of 401 points equally distributed around the airfoil and 241 points in the wall-normal direction clustered towards the body leading to 802 boundary points.

3.2.2. TFI mesh deformation From a theoretical point of view this type of rotational grid deformation is a major challenge for the TFI interpolation method. Although the grid depicted in Fig. 5(a) looks promising, a closer view in Fig. 5(b) reveals the failure of TFI regarding almost all aspects of this specific test case. Although TFI possesses the extreme preserving property, the grid deformation yields degenerated cells with cross-overs at the two extremities of the airfoil. The orthogonality is destroyed also at boundary points, where no degenerated cells appear. Furthermore, an artificial clustering of grid points on the surface of the airfoil appears, which is unacceptable for

DOI: 10.1002/nme.5465

any numerical simulation. Hence as expected, TFI is not suitable in case of a large rotation of a structure with sharp edges.

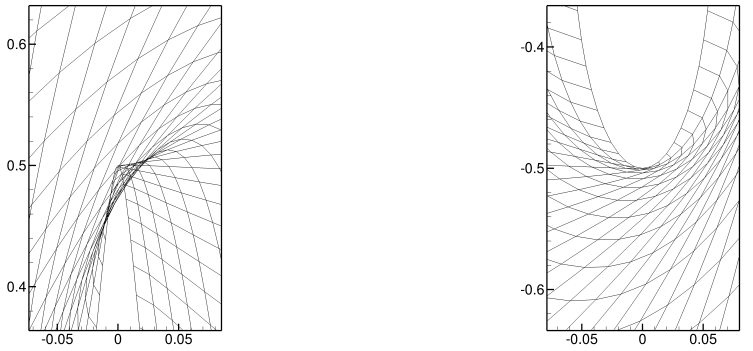


(a) Full view. (b) Close view.

Figure 5. *Problem 2:* TFI grid deformation (every 5^{th} grid point in each direction is displayed).

3.2.3. RBF mesh deformation The computations are carried out using six different RBFs. The five compact RBFs have a radius of support R = 15 in order to cover a major part of the domain under investigation. This value offers the best compromise between grid quality and computational effort. For four of these six RBFs further computations using the *hybrid greedy* algorithm are performed.

Table III provides an overview of all cases considered including the computational effort required and the global measure of the mesh quality $\langle f_{\text{skew}} \rangle$. The costs are provided as relative values to the corresponding time required for the mesh deformation using IDW[‡]. Obviously, Wendland's C^0 function fails to deform the mesh successfully leading to degenerated cells with cross-overs. A variation of the support radius R between 10 and 20 does not solve this problem. In a further test documented in Table III the convergence criterion for MINRES is decreased from $\varepsilon_c = 10^{-5}$ to 10^{-10}. Obviously, that has no effect and the deformation algorithm continues to produce degenerated cells. To the best of the authors' knowledge such a failure of W C^0 has not been reported in the literature. As visible in Fig. 6 the degenerated cells mainly appear in the vicinity of the trailing edge. This may be attributed to the combination of the rotation and the sudden change of the shape of the trailing edge.



(a) View of trailing edge. (b) View of leading edge.

Figure 6. *Problem 2:* Mesh deformation using W C^0.

[‡]IDW is taken as reference in this problem, because TFI fails to generate an appropriate grid.

DOI: 10.1002/nme.5465

Table III. *Problem 2:* Comparison of relative computational costs (IDW as reference), global measure of the mesh quality $\langle f_{\text{skew}} \rangle$ and remarks concerning the grid quality using RBFs (support radius set to 15) and IDW (*Degenerated cells* correspond to cross-overs of the cells and *bad first cell* to a highly and unacceptably deformed first cell on the airfoil).

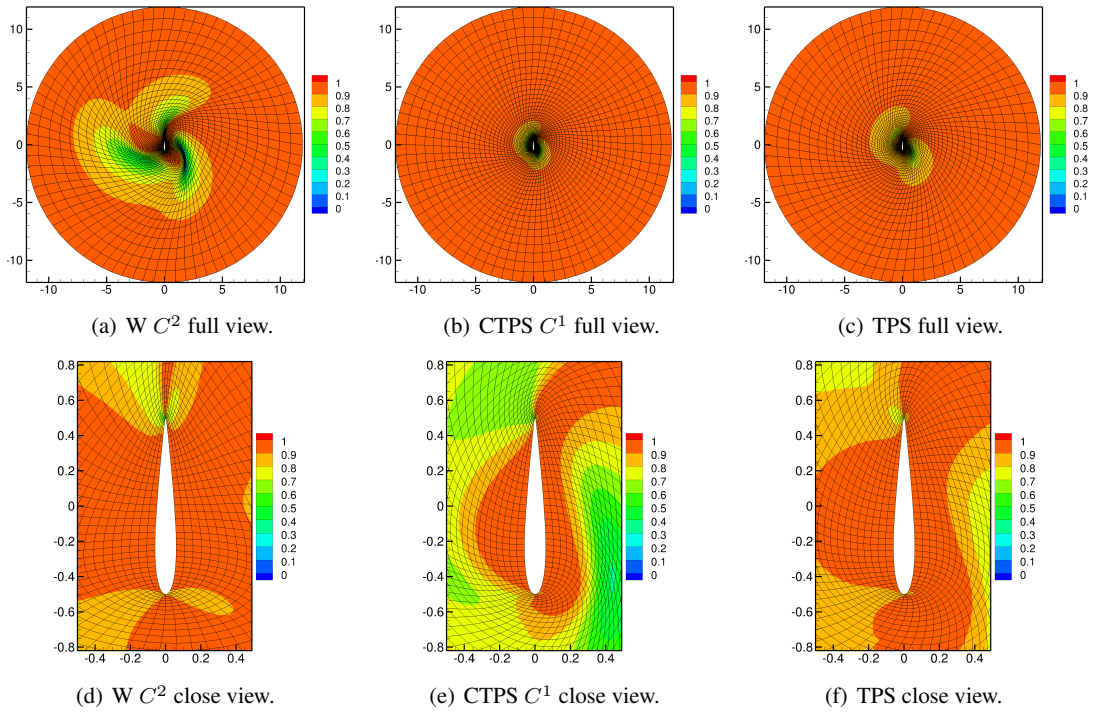
Method	ε_g	*Greedy points*	ε_c	Rel. comp. costs Total	*Greedy* part	Remark	$\langle f_{\text{skew}} \rangle$
RBF (W C^0)	-	-	10^{-5}	0.97	-	Degen. cells	0
	-	-	10^{-7}	1.24	-	Degen. cells	0
	-	-	10^{-10}	1.94	-	Degen. cells	0
RBF (W C^2)	-	-	10^{-5}	8.34	-		0.78
	-	-	10^{-7}	100.59	-		0.78
	-	-	10^{-10}	992.50	-		0.78
	10^{-2}	236	10^{-5}	1.70	79.2%	Bad first cell	0
		262	10^{-7}	438.29	99.9%	Degen. cells	0
		273	10^{-10}	4081.59	99.9%	Degen. cells	0
	10^{-3}	252	10^{-5}	6.53	94.2%		0.79
		271	10^{-7}	530.18	99.9%	Degen. cells	0
		291	10^{-10}	6718.52	99.9%	Bad first cell	0
RBF (CTPS C^1)	-	-	10^{-5}	3.84	-		0.86
	-	-	10^{-7}	8.89	-		0.86
	-	-	10^{-10}	28.91	-		0.86
	10^{-2}	430	10^{-5}	4.54	73.1%	Degen. cells	0
		430	10^{-7}	13.20	90.7%	Degen. cells	0
		430	10^{-10}	25.31	95.2%	Degen. cells	0
	10^{-3}	446	10^{-5}	11.42	88.9%		0.86
		447	10^{-7}	46.95	97.3%		0.86
		447	10^{-10}	99.86	98.7%		0.86
RBF (CTPS C^2_a)	-	-	10^{-5}	7.29	-		0.85
	-	-	10^{-7}	45.35	-		0.85
	-	-	10^{-10}	333.39	-		0.85
	10^{-2}	284	10^{-5}	2.79	70.3%	Bad first cell	0
		286	10^{-7}	20.62	95.9%	Degen. cells	0
		275	10^{-10}	36.20	97.8%	Bad first cell	0
	10^{-3}	326	10^{-5}	11.33	91.6%		0.85
		325	10^{-7}	128.03	99.2%		0.85
		318	10^{-10}	508.33	99.8%		0.85
RBF (CTPS C^2_b)	-	-	10^{-5}	8.98	-		0.82
	-	-	10^{-7}	73.96	-		0.82
	-	-	10^{-10}	679.70	-		0.82
	10^{-2}	275	10^{-5}	2.82	71.2%	Bad first cell	0
		307	10^{-7}	113.49	99.2%	Degen. cells	0
		305	10^{-10}	505.83	99.8%	Degen. cells	0
	10^{-3}	301	10^{-5}	15.03	94.0%		0.82
		313	10^{-7}	349.66	99.7%	Degen. cells	0
		328	10^{-10}	4968.41	99.9%	Bad first cell	0
RBF (TPS)	-	-	10^{-5}	4.45	-		0.89
	-	-	10^{-7}	13.29	-		0.89
	-	-	10^{-10}	52.78	-		0.89
IDW (no transf. div.)	-	-	-	1.01	-	α set by Eq. (18)	0.81
IDW (no transf. div.)	-	-	-	1	-	$\alpha = 0.1$	0.81

W C^2 is in general able to deform the mesh in a correct manner. The deformed mesh possesses the best quality in the vicinity of the airfoil in comparison with the other RBFs depicted in Fig. 7. It also closely resembles the grid around the NACA 0012 generated by the conformal transformation. However, in regions away from the airfoil the function W C^2 does not reach the same level of grid quality as the other RBFs. As clearly visible in Figs. 7, 8 and 9, the function W C^2 produces more skewed cells compared with the other basis functions investigated. That explains the lowest values of $\langle f_{\text{skew}} \rangle$ found for W C^2.

CTPS C^1, CTPS C^2_a, CTPS C^2_b and TPS perform almost in a similar manner and the outcome perceptibly differs from the deformations prescribed by W C^2. For the latter the effect of rotation on the grid has been successfully kept away from the vicinity of the airfoil, whereas in case of the

DOI: 10.1002/nme.5465

various types of thin plate spline functions the effect of rotation is somehow distributed starting at the wall-nearest grid points and resulting in a skewed grid in the vicinity of the airfoil. It is noteworthy that all basis functions are able to cope with the near-singularity situation close to the trailing edge. In the regions away from the airfoil the CTPS C^1 and TPS basis functions are found to perform best as visible based on the *skew quality metric* contours in Figs. 7 to 9 and the values of the global skewness $\langle f_{\text{skew}} \rangle$ in Table III.



(a) W C^2 full view. (b) CTPS C^1 full view. (c) TPS full view.

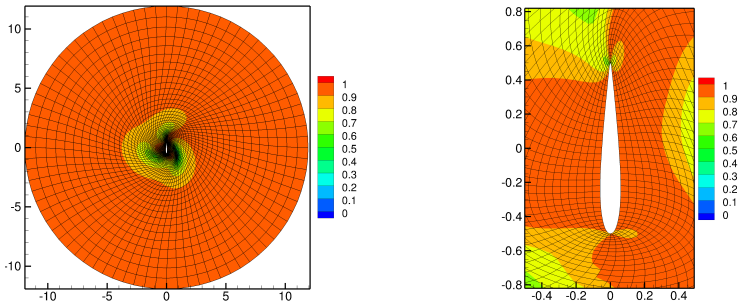
(d) W C^2 close view. (e) CTPS C^1 close view. (f) TPS close view.

Figure 7. *Problem 2:* Mesh deformation using RBFs (only every 5^{th} grid point along each direction shown).

To summarize, W C^0 fails to appropriately deform the grid in case of a sharp-edged rotating structure. W C^2 leads to a quasi rigid-body movement of the grid points near the rotating airfoil, whereas the major grid deformation is shifted in the direction of the fixed boundaries. Thus, the quality of the deformed grid suffers in this region leading to more skewed cells. The four RBFs (CTPS C^1, CTPS C^2_a, CTPS C^2_b and TPS) offer the best grid quality. The computational costs of CTPS C^1 and TPS are much lower than for CTPS C^2_a and CTPS C^2_b. W C^2 is the most expensive one. This trend closely resembles the relative costs reported for *problem 1* in Table II and is explained by the ill-conditioned W C^2 matrix. Note that the computational effort with compact RBFs obviously depends on the support radius. Indeed, for RBFs without any data reduction technique the computational effort can be reduced by decreasing the support radius. Since this also deteriorates the grid quality, a compromise on R has to be found.

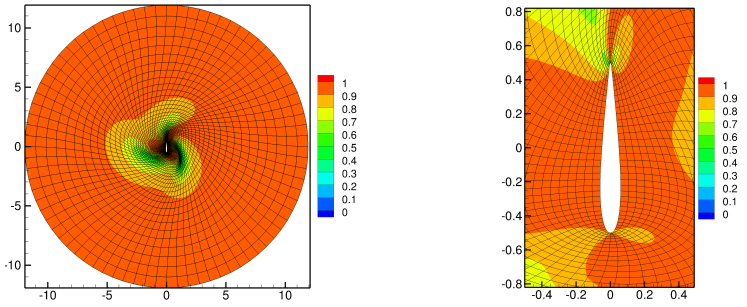
Next, it is of interest to investigate the performance of the *hybrid greedy* algorithm concerning data reduction and computational effort. If ε_g is set to a high value (here $\varepsilon_g = 10^{-2}$), the *greedy* algorithm generates an useless grid with degenerated cells for all RBFs tested. Applying a lower value of $\varepsilon_g = 10^{-3}$, CTPS C^1 and CTPS C^2_a yield an appropriate mesh. However, for W C^2 and CTPS C^2_b only a carefully chosen value of ε_c leads to appropriate deformations without degenerated cells. In most cases of failure the deformed mesh is degenerated at the trailing edge. From a geometrical point of view it is mandatory for an efficient data reduction strategy to choose *greedy* boundary points in the vicinity of this trailing edge. In all computations the initial guess needed for the *greedy* algorithm is on purpose taken at a node of the leading edge. This particularly harsh choice is made in order to see the response of the *greedy* algorithm to situations involving FSI-LES predictions, where the appearance of a sudden singularity can not be ruled out. Nevertheless, if the

DOI: 10.1002/nme.5465



(a) Full view. (b) Close view.

Figure 8. *Problem 2:* Mesh deformation using CTPS C_a^2 without and with the *hybrid greedy* algorithm (Since the results in both cases are nearly identical, only one set is shown here. Note that only every 5^{th} grid point along each direction is depicted).



(a) Full view. (b) Close view.

Figure 9. *Problem 2:* Mesh deformation using CTPS C_b^2 without and with the *hybrid greedy* algorithm (only every 5^{th} grid point along each direction shown).

mesh is not degenerated, it is almost indistinguishable from the deformed mesh obtained without the data reduction technique. Therefore, the corresponding counterparts to Figs. 8 and 9 using the RBFs CTPS C_a^2 and CTPS C_b^2 are omitted here for the sake of brevity.

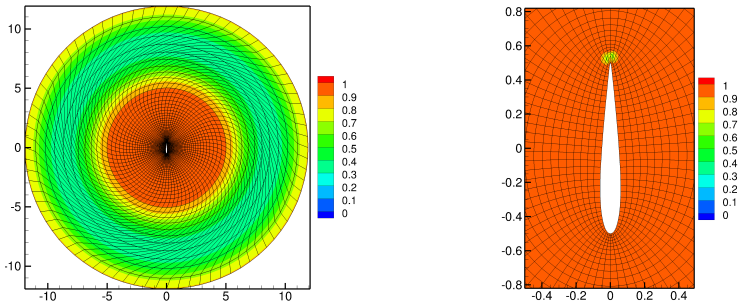
As can be seen in Table III the *hybrid greedy* algorithm is very time consuming. The procedure iterates many times and is more expensive than the algorithm without any data reduction in most cases. Therefore, the use of a *greedy* procedure often makes sense only for problems with several successive mesh deformations of a nearly identical grid such as encountered in time-dependent FSI problems. In this case the *greedy* algorithm is applied for a certain time step. The chosen *greedy* boundary points are stored and used for many successive deformation cycles without renewal of the chosen boundary points, so that the CPU-time is globally reduced. However, if the grid is strongly deformed, it might be necessary to renew the choice of the boundary points after a while.

3.2.4. IDW mesh deformation Finally, *problem 2* is evaluated using the IDW mesh deformation technique with the value of α calculated by Eq. (18). The design of this interpolation strategy by Luke et al. [30] is specifically targeted towards a method, which can interpolate the rotation with high accuracy. Therefore, IDW theoretically should be able to provide a high-quality grid for this problem. Since the origin is located at the center of the airfoil and the rotation is carried out with respect to this origin, no transformation division technique is employed.

The results for the same initial grid as used before are depicted in Fig. 10. Note that a nearly identical distribution of the *skew quality metric* is found on a much coarser grid evidencing the grid independence of the algorithm. Based on the work of Luke et al. [30] the computations are repeated using a fixed value of $\alpha = 0.1$. Again, no significant variation of the grid properties can be noticed.

DOI: 10.1002/nme.5465

Furthermore, the computational effort for a fixed or a predicted value of α are almost identical (see Table III). In comparison to the different RBFs without *greedy* algorithm, IDW is at least a factor of 3 faster even in the case of the worst tolerance $\varepsilon_c = 10^{-5}$. For stricter tolerance values ε_c the CPU-time ratio turns out to be even more clearly in favor of IDW. Here it is worthwhile to mention that the computational complexity of both RBF and IDW scales with $N_v \times N_s$ as noted earlier and hence for sufficiently large systems the CPU-times for both RBF and IDW may be isomorphic.



(a) Full view. (b) Close view.

Figure 10. *Problem 2:* Mesh deformation using IDW and α calculated by Eq. (18) (only every 5^{th} grid point along each direction shown).

The mesh deformation using IDW is nearly perfect, at least in the vicinity of the airfoil. A comparison with the grid generated by the conformal transformation reveals the effectiveness and significance of IDW for grid deformations involving significant rotation. The deformed grid is nearly orthogonal at all points close to the boundary of the airfoil, including the trailing edge. In regions away from the airfoil the skewness level increases. Thus, it is obvious that the IDW grid deformation is well-suited for pushing skewed grid cells away from the rotating body even with sharp edges. This is an important advantage for FSI-LES computations.

3.2.5. Summary Based on the performance of the three methods (TFI, RBF and IDW) it can be concluded that RBF and IDW perform significantly better than TFI. IDW delivers a nearly perfect mesh in the vicinity of the airfoil, whereas all RBFs perform better at a certain distance from the body. Among all RBFs the performance of W C^2 is the best as far as the orthogonality preservation in the vicinity of the body is concerned. Thus, for this test case the outcome is in close agreement with the observations of De Boer et al. [18]. It is worthwhile to mention that the IDW technique without the transformation division procedure takes much less time than any RBF. However, in case of very large systems the computational costs for both may be comparable. Globally, the *greedy* algorithm does not perform very well for all RBFs applied to *problem 2* and shows its limits. This is due to the presence of a sharp edge on the moving body combined with the severe rotation. The use of a limited number of *greedy* boundary nodes implies that the imposed deformation is only correctly represented at these selected boundary points. In order to get an appropriate grid deformation, a sufficient number of boundary points describing the sharp geometry has to be selected. In the original *greedy* algorithm that is not assured. Hence, a possible improvement of the *greedy* algorithm might be to take the first cell height in the error analysis into account, so that more boundary points are taken in highly resolved near-wall regions. Moreover, the *greedy* algorithm is more expensive than the mesh deformation without any data reduction technique and thus only makes sense if the choice of the boundary points can be used several times.

4. HYBRID MESH DEFORMATION METHODS

In order to find a reasonable compromise between the quality of the deformed grid and the computational effort required for the deformation, block-structured grids offer the opportunity to

DOI: 10.1002/nme.5465

combine the favorable properties of different methods while avoiding their drawbacks as far as possible. The hybrid mesh deformation methodology relies on this block structure and consists of a combination of two elementary mesh deformation algorithms: One for the deformation of the block interfaces and one for the inner points. In the following, the applied hybrid methods are denoted at first by the name of the technique applied for the block interface and in the second place by the name of the method used for the inner points. Since the number of grid points on the block interfaces only scales with $N_v^{2/3}$, major savings can be achieved if expensive techniques such as RBF or IDW are solely applied to the block interfaces and not to the entire grid. These hybrid grid deformation strategies are investigated based on the experience gained by the evaluation of the three elementary strategies. Furthermore, they are designed in such a manner that important advantages, i.e., low computational effort and high *skew quality metric* preservation are combined.

Since in FSI-LES predictions the CPU-time consumption is of paramount importance, it is anticipated that grid deformations based on RBFs applied to the entire domain are not feasible. However, taking the favorable mesh quality preservation property of RBF into account, it can be concluded that RBF may be of utmost importance when applied in a sub-domain. In such a situation a much smaller system of equations has to be inverted and therefore a huge amount of CPU-time can be saved. Furthermore, it has been noticed that for convex domains without substantial rotation TFI performs well and is very fast. The strength of IDW is its capacity to accurately capture the rotation and to preserve the orthogonality in the vicinity of the body, whereas it is found to be fall behind the others for a clustered grid. Based on these observations the following two hybrid methodologies are proposed:

- **IDW-RBF:** IDW is applied at the block interfaces to preserve more or less the orthogonality. For the inner points of each block it is exploited that RBF achieves a high grid quality. Furthermore, since the sub-domains are processed in parallel by different processors, the system of equations related to the RBF implementation of the inner points may not turn out to be very costly.
- **IDW-TFI:** Again IDW is applied at the block interfaces to retain the advantages of IDW, whereas TFI is used for the displacements of the inner points. Once the orthogonal character of the mesh is maintained at the interfaces, it is exploited that TFI produces high-quality grid deformations of the inner nodes very fast.

Both hybrid variants bypass the disadvantages of IDW for clustered meshes (see *problem 1*) to a large extent, since it is only used at the block interfaces and not for the nodes in the vicinity of the interface.

Additionally, a hybrid RBF-TFI methodology is used for comparison purposes. Since the solution of a huge system of equations involving all boundary nodes is required, this method is certainly not very interesting due to the required CPU-time.

For the sake of brevity and based on the experiences on the previous test cases considered, the application of the RBF mesh deformation technique is restricted to the TPS basis function with global support not taking the *greedy* algorithm into account. The fact that TPS leads to reasonable mesh deformations for *problem 1* and *2* which are among the best motivates this choice. The predictions based on IDW are carried out using quaternions but without transformation division steps. These are not required here since the origin of the coordinate system lies at the center of rotation and all boundary points of the structure undergo the same rotation.

4.1. Description of the problem

The proposed hybrid algorithms and their elementary counterparts are tested in a situation with substantial rotation and deformation. Furthermore, the analysis is extended to the three-dimensional case and for deformations, which resemble the most important features of a standard test case for FSI-LES simulations [1, 2, 3]. In analogy to most FSI-LES cases the integration domain is divided into several sub-domains allowing to parallelize the computation based on domain decomposition with explicit message passing.

DOI: 10.1002/nme.5465

International Journal for Numerical Methods in Engineering, vol. 111 (3), pp. 273–300, (2017).

The undeformed block-structured grid along with the partitioning into ten blocks is depicted in Fig. 11. The geometrical setup is as follows: A square cylinder with a unit edge is placed in a plane rectangular channel with a height of 13 units, a length of 12 units and a width of 0.5 units. A splitter plate with a length of 5 units and a thickness of 0.25 units is attached to the square cylinder. The center of the cylinder coincides with the origin of the coordinate system. Note that the associated flow problem is a three-dimensional generalization of the nominally two-dimensional geometric configuration.

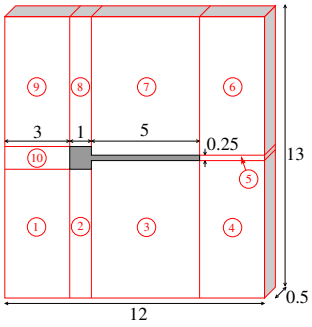


Figure 11. Sketch of the topology of the grid structure with the splitting into ten blocks.

In this study the cylinder is exposed to a rotation of 30° around the center. Furthermore, the splitter plate is deformed by a trigonometric function according to $0.2\sin(2\pi x)$, where x is the distance from the point of attachment. Hence, the splitter plate is subjected to two different kinds of deformations. The grid possesses 8 cells in the spanwise direction. Each of the spanwise planes is further sub-divided into square cells of dimension $1/16$ units leading to 363,033 nodes (= $193 \times 209 \times 9$ without subtracting the nodes inside the structure) in the entire domain under consideration.

4.2. Results of the elementary and hybrid methods

The results obtained by the elementary and hybrid methods are depicted in Fig. 12. For the grid deformed by TFI-TFI a C^1-discontinuity is observed at the interfaces between the blocks (see Fig. 12(a)). This grid discontinuity is not taken into account by the *skew quality metric*. A jump in the f_{skew} level appears but its value remains high. Therefore, the harmonic average $\langle f_{\text{skew}} \rangle$ reported in Table IV is found to be high. That shows that $\langle f_{\text{skew}} \rangle$ is not sufficient for the evaluation of the overall grid quality. This sudden change at the block interface is attributed to the differences in the interpolating polynomials used in distinct sub-domains. Thus as noted for *problem 1*, the pure TFI algorithm is unable to maintain the orthogonality and is inappropriate for FSI-LES predictions.

Table IV. Comparison of relative computational costs and global measure of the mesh quality $\langle f_{\text{skew}} \rangle$ for elementary and hybrid mesh deformation strategies applied to a block-structured grid.

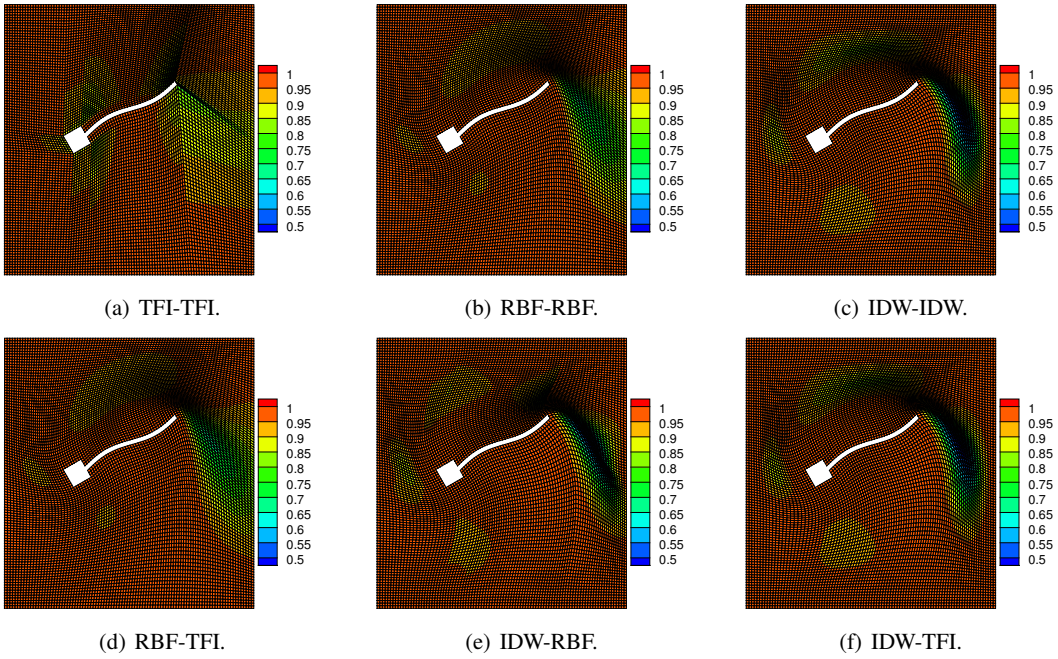
Method	Rel. comp. costs	Remark	$\langle f_{\text{skew}} \rangle$
TFI-TFI	1	Block interfaces strongly non-smooth	0.96
RBF-RBF	2013.71	Overall high grid quality	0.96
IDW-IDW	19.71	High grid quality near moving boundaries	0.96
RBF-TFI	2000.78	Overall high grid quality	0.96
IDW-RBF	418.00	Overall good grid quality	0.95
IDW-TFI	**5.46**	Best compromise of quality & CPU-time	0.95

As visible in Fig. 12(b) the RBF-RBF deformation strategy works quite well. The orthogonality is preserved in the deformed grid. Although at the far field, especially in the downstream direction, some nodes are skewed, their *skew quality metric* is still high and a reasonable deformation of the grid leading to an overall high-quality grid can be observed. As usual IDW generates a mesh,

DOI: 10.1002/nme.5465

which preserves the orthogonality in the vicinity of the deformed splitter plate to a great extent. However, in the region downstream of the splitter plate some cells with a relatively high skewness are observed in Fig. 12(c). This is consistent with earlier observations that the IDW interpolation pushes the skewness away from the deforming body.

The two hybrid versions involving IDW for the block interfaces (see Fig. 12(e) and 12(f)) lead to similar grid qualities as obtained by the pure IDW method. Similarly the hybrid RBF-TFI method (see Fig. 12(d)) delivers a grid of almost the same quality as the pure RBF technique. Thus, the lesson which could be learned from the present results is that the method applied for the deformation of the block interfaces is of major importance for the grid quality achieved, whereas the method used for the displacements of the inner grid points plays a minor role.



(a) TFI-TFI. (b) RBF-RBF. (c) IDW-IDW.

(d) RBF-TFI. (e) IDW-RBF. (f) IDW-TFI.

Figure 12. Deformed mesh and *skew quality metric* contours computed by the three elementary and the three hybrid grid deformation methods.

The corresponding relative computational efforts related to all six computations are given in Table IV. Similar to the previous cases huge differences are already found between the pure elementary methods. As expected, pure TFI produces deformed meshes with the lowest computational effort. TFI is about 2000 times faster than pure RBF and still about 20 times faster than pure IDW. As visible in Table IV the major disadvantage of any RBF implementation is the large computational effort required consumed. As anticipated a combination of RBF and TFI only slightly reduces the computational costs in comparison with the pure RBF algorithm, since the time-consuming determination of the coefficients of the basis functions are required in both variants. Hence, this hybrid technique is also not suitable. Combining IDW with RBF reduces the computational costs of the pure RBF method significantly. However, it is quite slow compared to the IDW-TFI method. The hybrid IDW-TFI method is very promising since it only takes about one quarter of the computational effort of the pure IDW technique while a grid of comparably high quality is achieved.

5. APPLICATION OF THE HYBRID METHOD TO A REAL-LIFE FSI PROBLEM

In the previous section the hybrid IDW-TFI method proves its ability to deform the mesh conserving a high quality with reasonable computational effort. This hybrid method is now evaluated based

DOI: 10.1002/nme.5465

on a real-life FSI problem (see Fig. 13): A flexible membranous hemisphere (diameter $D = 1.5 \times 10^{-1}$ m) is attached on a flat plate and placed in an approaching thick turbulent boundary layer (Re = $\rho_{\text{air}}\, D\, U_\infty \,/\, \mu_{\text{air}} = 50{,}000$) (see Fig. 14). This FSI-LES case is derived from the turbulent flow past a rigid hemisphere [46] with two modifications: First, the grid is coarsened and consists of about 4.5×10^6 control volumes. Second, the deformation of the hemisphere is imposed based on the following movement of the surface in spherical coordinates $(r_{\text{FSI}}, \theta, \phi)$:

$$r_{\text{FSI}} = A_{\text{def}} \frac{z}{D/2} \sin(\omega_t\, t)\, \cos(4\,\phi) \quad \text{with} \quad A_{\text{def}} = 1.1 \times 10^{-2}\ \text{m} \quad \text{and} \quad \omega_t = 2\pi\, 10^3\ \text{rad s}^{-1}.$$

Figure 13. Air inflated structure in Gentofte (Denmark).

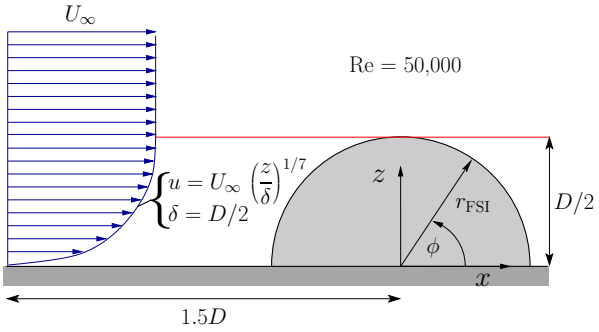


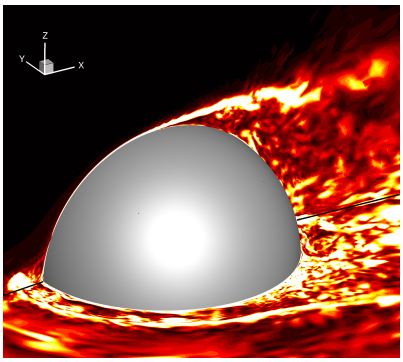
Figure 14. Surface-mounted hemisphere within a turbulent boundary layer.

Hence, it presently describes only a one-way coupled simulation which, however, is sufficient here for the evaluation of the hybrid technique. Note that this imposed movement is not physical. However, the chosen parameters leads to severe deformations in critical parts of the geometry: On the top of the hemisphere, on the sides and at the junction between the membrane and the bottom wall. The amplitude of the deformation A_{def} is set to a value much larger than what is expected in a real physical case in order to stress the robustness of the proposed hybrid method. A snapshot of the geometry and the flow is shown in Fig. 15 at the time instant of the maximum deformation ($t = 2.5 \times 10^{-4}$ s). The solver (FASTEST-3D) and the techniques used to compute this FSI-LES problem are described in Breuer et al. [1] and were validated by complementary experimental-numerical FSI investigations [2, 3].

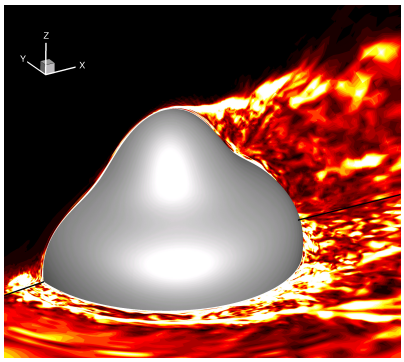
Figures 15(a) and 15(b) show the vorticity magnitude of the turbulent flow around the rigid and the flexible hemisphere in two slices ($y/D = 0$, $z/D = 0.014$). Non-physical numerical oscillations are not detected, which is a first rough indicator for a grid of high quality. To evaluate the grid quality, the *skew quality metric* is plotted in the slice $y/D = 0$ in Figs. 15(c) and 15(d). The undeformed grid possesses a high quality. The first cells are all perfectly orthogonal to the wall and the *skew quality metric* is close to unity in the whole mesh. After the deformation the quality of the grid is slightly altered in several regions around the hemisphere (see Fig. 15(d)). However, the *skew quality metric* remains high in these areas (> 0.9). Furthermore, it is important to mention that the high grid quality is maintained near the moving membrane (see Fig. 15(e)) and near the bottom wall (see Fig. 15(f)). As explained in Section 3.2.4 this is one of the main advantages of a method based on IDW: The deformation is pushed away from the FSI interface and the node distribution remains unchanged in the wall vicinity. This particularity ensures that the boundary layer can be correctly resolved by the LES solver.

The junction between the bottom wall and the flexible hemisphere is a critical region during the mesh deformation based on IDW. In case of a wall-resolved LES the mesh is very fine here. When the structure inflates, the area is reduced and degenerated cells can appear. In case of the hybrid IDW-TFI method, IDW is only applied on the block boundaries, which are far away from the FSI interface. Therefore, this is not an issue for the present flexible hemisphere. The mesh does not contain any bad cells and the orthogonality is more or less conserved also in this critical region.

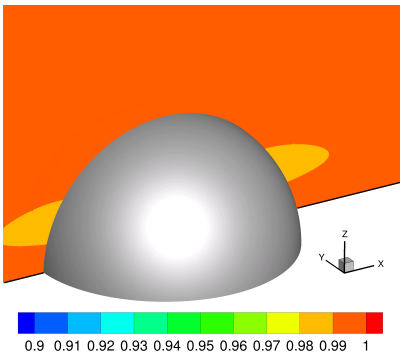
DOI: 10.1002/nme.5465



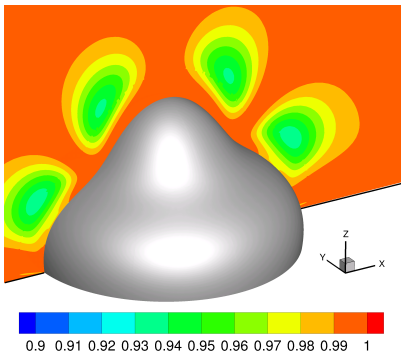
(a) Full view of the rigid hemisphere: Slices ($y/D = 0$, $z/D = 0.014$) with the vorticity magnitude.



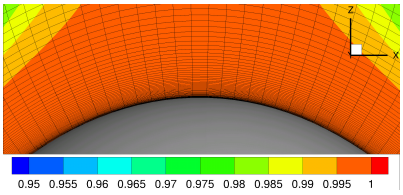
(b) Full view of the deformed hemisphere: Slices ($y/D = 0$, $z/D = 0.014$) with the vorticity magnitude.



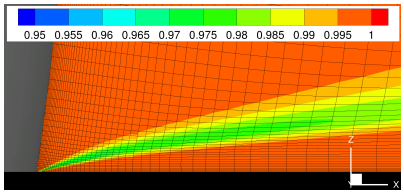
(c) Full view of the rigid hemisphere: Slice ($y/D = 0$) with the grid *skew quality metric*.



(d) Full view of the deformed hemisphere: Slice ($y/D = 0$) with the grid *skew quality metric*.



(e) Close view of the deformed grid at the top: Slice ($y/D = 0$) with the grid *skew quality metric*.



(f) Close view of the deformed grid at the bottom: Slice ($y/D = 0$) with the grid *skew quality metric*.

Figure 15. Snapshot of the flexible hemisphere in a turbulent flow predicted by LES.

Table V. Comparison of the floating-point operations (GFLOP for one time step) and of the relative computational costs for elementary and hybrid mesh deformation strategies applied to a real-life FSI problem.

Method	GFLOP	Rel. comp. costs	Remark
TFI-TFI	-	-	Cross-overs
RBF-RBF	-	-	System too large to be solved
IDW-IDW	9675	6.68	High grid quality near moving boundaries
IDW-TFI	1328	1	Best compromise of quality & CPU-time

Concerning the CPU-time consumption the hybrid IDW-TFI method is about 6.7 times faster than the pure IDW method for exactly the same grid quality (see Table V). Note that the corresponding

DOI: 10.1002/nme.5465

International Journal for Numerical Methods in Engineering, vol. 111 (3), pp. 273–300, (2017).

ratio of floating-point operations is about 7.3. Thus the pure IDW methods needs more floating-point operations but runs slightly more efficiently than the hybrid IDW-TFI method. No comparison with the elementary TFI and RBF methods is possible. The former leads to cross-overs, whereas for the latter the system of equations can not be solved in a reasonable time due to the large number of boundary points.

6. CONCLUSIONS

Three elementary mesh deformation methods are thoroughly investigated regarding their responses to situations involving substantial deformations. As expected, TFI is able to deform the grid quite fast, but is incapable of dealing with substantial rotation. Moreover, in case of a block-structured grid the deformed mesh is not smooth at the block interfaces. Nevertheless, in convex domains with prescribed deformations on the boundaries, its performance is reliable, although some loss of orthogonality is observed. RBF on the other hand produces a mesh of high quality in almost all situations, but is computationally expensive. Direct solvers for RBF are faster than iterative ones but limited to small problems. Thus, the latter is preferred for general utilization. At a glance of the test cases studied, the TPS basis function with global support and Wendland's C^2 function with local support are the best choices. W C^2 is able to preserve the orthogonality, but is time-consuming. TPS represents a good compromise between quality and speed. A well-thought-out use of IDW is also an attractive alternative guaranteeing a reasonable computational effort. An IDW method split into a rotational and translational deformation maintains a perfect orthogonality in the near-wall region, which is mandatory for LES. However, it requires to set many problem-dependent parameters, where unfavorable choices can lead to a degenerated mesh with cross-overs. The IDW grid deformation relying on a rotation and a translation step introduces a coordinate dependence. To solve this problem the time-consuming *transformation division steps* can be applied. However, for FSI-LES simulations involving small time steps and thus small relative displacements between subsequent time steps, IDW can renounce this additional procedure. For this purpose is has to be implemented in such a manner that it relies on the last deformed grid mimicking the *transformation division steps* and thus eliminating the coordinate dependence and the high computational costs.

Hybrid methods represent an appropriate compromise between quality requirements and computational effort. Since for block-structured grids the deformation consists of two steps (deformation of the block interfaces [$\sim N_v^{2/3}$] and displacement of the inner points [$\sim N_v$]), in principle six combinations of TFI, RBF and IDW are possible. Since the application of TFI for the block interfaces in not appropriate and the hybrid method RBF-IDW is canceled owing to the expected high effort, three possible hybrid variants remain: IDW-TFI, IDW-RBF and RBF-TFI. These are analyzed based on a realistic FSI scenario.

It is found that the deformation of the block boundaries plays the major role for the quality of block-structured grids. The block boundaries of a typical FSI-LES prediction are best adapted by IDW since it correctly captures the rotation of a deforming body. Once these block boundaries are deformed in an efficient manner, the deformation of the inner mesh is less critical and can be performed either by TFI or RBF. In case of substantial deformations RBF is favored for the inner deformation, but even in conjunction with the *greedy* algorithm the computational effort is still extremely high, which renders it unfeasible for a regular usage within each time step or even each FSI sub-iteration step of a coupled FSI-LES prediction. In many cases the application of TFI within the second step of the hybrid algorithm leads to deformed grids of high quality, while the CPU-time requirements stay at a reasonable level. Based on these arguments the hybrid IDW-TFI methodology is clearly favored and additionally tested based on a real-life FSI problem of a hemispherical membranous dome with an imposed time-dependent movement. It turns out that the proposed hybrid IDW-TFI method appropriately deforms the grid maintaining the original distribution and the orthogonality of the cells in the near-wall region. Thus, it generates high-quality grids in a reasonable time as mandatory for FSI-LES.

DOI: 10.1002/nme.5465

ACKNOWLEDGMENTS

The first author gratefully acknowledges the Indian National Science Academy (INSA), India and the Deutsche Forschungsgemeinschaft (DFG), Germany for their patronage and financial support under the INSA-DFG bilateral collaboration program. Financial support received from DFG under contract number BR 1847/15-1 allowed the first author to stay at the Helmut-Schmidt Universität, Hamburg for three months. Furthermore, the project is financially supported by the DFG under the contract number BR 1847/12-2.

A. DECOMPOSITION OF THE DISPLACEMENTS INTO ROTATION (MODELED WITH QUATERNIONS) AND TRANSLATION

The displacement at each boundary node can be decomposed into a rotation (quaternion Q) and a translation T. Q and T can be determined by using the steps outlined by Maruyama et al. [32]:

1. Determine unit normal vectors $\hat{\boldsymbol{n}}_u$ and $\hat{\boldsymbol{n}}_d$ at each surface grid point corresponding to the undeformed and deformed mesh, respectively. This can be accomplished in two steps. The first step is to represent the coordinates of the neighboring nodes with respect to the node under consideration (green node in Fig. 16). These vertices are now denoted $R'_u = [0, \boldsymbol{r}'_u]$ or $R'_d = [0, \boldsymbol{r}'_d]$ for the undeformed and deformed mesh, respectively. A schematic representation for one point can be found in Fig. 16(a) and 16(b), respectively. The second step is to predict the unit normals to the face corner position vectors and to take the average leading to the computation of $\hat{\boldsymbol{n}}_u$ and $\hat{\boldsymbol{n}}_d$, respectively.
2. The first quaternion Q_1 is then defined as

$$Q_1 = \left[\cos\frac{\theta_1}{2}, \hat{\boldsymbol{a}}\sin\frac{\theta_1}{2}\right], \quad \text{with} \quad \theta_1 = \cos^{-1}(\hat{\boldsymbol{n}}_u.\hat{\boldsymbol{n}}_d), \quad \hat{\boldsymbol{a}} = \frac{\hat{\boldsymbol{n}}_u \times \hat{\boldsymbol{n}}_d}{\| \hat{\boldsymbol{n}}_u \times \hat{\boldsymbol{n}}_d \|}. \tag{21}$$

   The quaternion Q_1 represents the change of the surface orientation during deformation and aligns $\hat{\boldsymbol{n}}_u$ to $\hat{\boldsymbol{n}}_d$ as depicted in Fig. 16(c).
3. Now the position vectors of the undeformed face are updated to $R''_u = [0, \boldsymbol{r}''_u]$ using the relation

$$R''_u = Q_1 R'_u Q_1^*. \tag{22}$$

4. Next, it is necessary to rotate the updated undeformed surface in order to align it as good as possible with the corners of the deformed face. The corresponding rotation axis is $\hat{\boldsymbol{n}}_d$ and the corresponding angle θ_2 can be calculated as follows:

$$\theta_2 = \frac{1}{N_{vrtx}} \sum_{n=1}^{N_{vrtx}} \pm\cos^{-1}\left[\frac{\boldsymbol{r}''_u.\boldsymbol{r}'_d}{\| \boldsymbol{r}''_u \| \| \boldsymbol{r}'_d \|}\right], \tag{23}$$

   where N_{vrtx} is the number of adjacent vertices. This process can be seen in Fig. 16(d). Thus, a second quaternion Q_2 which represents the rotation due to the cell shape modification during the mesh deformation procedure can be defined as

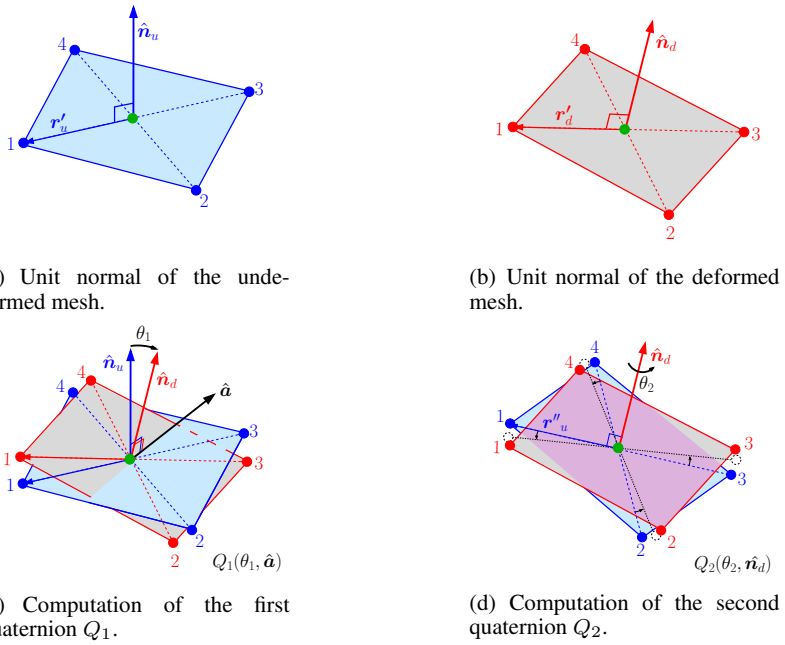
$$Q_2 = \left[\cos\frac{\theta_2}{2}, \hat{\boldsymbol{n}}_d\sin\frac{\theta_2}{2}\right]. \tag{24}$$

5. Finally, the quaternion Q is attained by using the composition of Q_1 and Q_2,

$$Q = Q_2\, Q_1. \tag{25}$$

Once the rotation quaternion Q is determined at a boundary node, the translation quaternion $T = R_d - QR_uQ^*$ can be calculated following Eq. (20). However, this translation vector is coordinate dependent since the rotation has been calculated with respect to a local coordinate system. To

DOI: 10.1002/nme.5465



(a) Unit normal of the undeformed mesh.

(b) Unit normal of the deformed mesh.

(c) Computation of the first quaternion Q_1.

(d) Computation of the second quaternion Q_2.

Figure 16. Representation of the procedure to obtain quaternions at a surface node having four neighboring nodes.

eliminate this coordinate dependence, a strategy called *transformation division* algorithm has been proposed by Maruyama et al. [32]. This algorithm consists of repeating the identical elementary process m times. Each iteration of this elementary transformation is composed of a rotation about the same axis but with an incremental angle θ/m, where θ is the angle of rotation prescribed by the quaternion Q. This is followed by the corresponding calculation of the translation vector. The entire procedure can be summarized as

6. Using $Q = [\cos(\frac{\theta}{2}), \hat{\boldsymbol{b}} \sin(\frac{\theta}{2})]$ one computes

$$Q^{1/m} = \left[\cos\left(\frac{\theta}{2m}\right), \hat{\boldsymbol{b}} \sin\left(\frac{\theta}{2m}\right)\right]. \tag{26}$$

The corresponding calculation of the translation vector is based on the relation

$$T_m = \mathcal{M}^{-1}(R_d - \mathcal{N} R_u), \tag{27}$$

where $\mathcal{N}$ is the 3×3 element matrix corresponding to the quaternion Q and

$$\mathcal{M} = \sum_{i=0}^{m-1} \mathcal{N}^i_{1/m}. \tag{28}$$

B. COORDINATE DEPENDENCY OF IDW AND TRANSFORMATION DIVISION STEPS

In order to illustrate the coordinate dependency of the interpolated quaternions used for IDW leading to different grid deformations for an identical structural movement, the simple test case of a bending beam in a rectangular region is considered. Furthermore, this setup allows to discuss the subsequent strategy developed to avoid this phenomenon.

Assuming a two-dimensional problem, a beam of unit length is immersed in a rectangle of dimension 4×2. Computations are carried out using two different grids, a coarse mesh of size

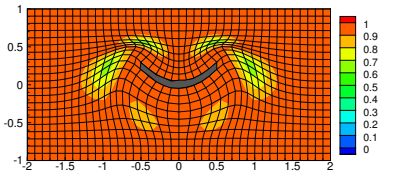
DOI: 10.1002/nme.5465

41×22 and a finer one of size 401×202. Since the beam is immaterial, its thickness has been set to one grid spacing in y-direction. The beam is deformed by prescribing a parabolic displacement $y = x^2$, where x is the distance from the center of the beam.

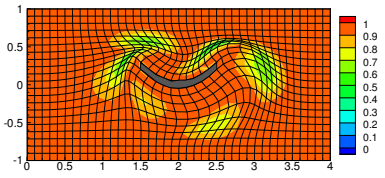
In IDW all the displacements are determined in terms of a global coordinate system, but the quaternions are calculated locally [32]. This leads to a situation where the translation vector, computed at a surface point, depends on its global position, whereas the rotation, expressed by quaternions, does not. Hence, the displacements of the interpolated volume grid points loses symmetry.

The so-called *transformation division* procedure introduced by Maruyama et al. [32] intends to make this interpolation independent of the coordinate system by dividing the deformation into small steps. Thus, it is a finite implementation of an essentially infinite procedure. In case of an unsteady computation it is found that with a small time step the deformations are most often small in each step. Hence the *transformation division* technique may not be necessary. To validate this procedure, the problem is tackled as a steady and an unsteady process, where the latter is completed in unit non-dimensional time.

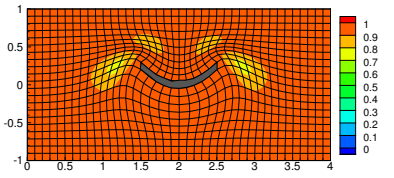
In Fig. 17 the deformed mesh and its *skew quality metric* predicted by various strategies are presented. Figures 17(a), 17(b) and 17(c) are obtained without using the *transformation division* technique. In Fig. 17(a) a symmetric deformation is visible because the origin of the coordinate system is located at the center of the geometry. However, a non-symmetric grid is predicted if this origin is shifted away from the center (see Fig. 17(b)). In Fig. 17(c) a time-marching strategy is used dividing the entire deformation into 25 time steps. Obviously, this leads to a symmetric mesh. The *transformation division* procedure is applied in the fourth case shown in Fig. 17(d). Again 25 steps similar to the time-marching procedure are used. Obviously the results obtained by the time-marching strategy and the *transformation division* algorithm are indistinguishable from each other. A particularly interesting observation that requires attention is that neither Fig. 17(c) nor Fig. 17(d) matches with Fig. 17(a). Both procedures relying on a splitting of the entire deformation into a finite number of sub-steps yield a higher grid quality than the original procedure. Thus, it can be concluded that making the node displacement procedure independent of the coordinate system is fundamental.



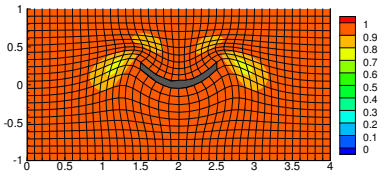
(a) IDW with origin located at the center of the mesh.



(b) IDW with origin located away from the center of the mesh.



(c) IDW with origin located away from the center of the mesh but based on an unsteady computation.



(d) IDW with origin located away from the center of the mesh but using the *transformation division* procedure.

Figure 17. Deformed mesh of the beam and *skew quality metric* computed by IDW with or without the *transformation division* procedure.

REFERENCES

DOI: 10.1002/nme.5465

1. Breuer M, De Nayer G, Münsch M, Gallinger T, Wüchner R. Fluid-structure interaction using a partitioned semi-implicit predictor-corrector coupling scheme for the application of large-eddy simulation. *Journal of Fluids and Structures* 2012; **29**:107–130.
2. De Nayer G, Kalmbach A, Breuer M, Sicklinger S, Wüchner R. Flow past a cylinder with a flexible splitter plate: a complementary experimental-numerical investigation and a new FSI test case (FSI-PfS-1a). *Computers & Fluids* 2014; **99**:18–43.
3. De Nayer G, Breuer M. Numerical FSI investigation based on LES: Flow past a cylinder with a flexible splitter plate involving large deformations (FSI-PfS-2a). *International Journal of Heat and Fluid Flow* 2014; **50**:300–315.
4. Apostolatos A, Schmidt R, Wüchner R, Bletzinger KU. A Nitsche-type formulation and comparison of the most common domain decomposition methods in isogeometric analysis. *International Journal for Numerical Methods in Engineering* 2014; **97**(7):473–504.
5. Breitenberger M, Apostolatos A, Philipp B, Wüchner R, Bletzinger KU. Analysis in computer aided design: Nonlinear isogeometric b-rep analysis of shell structures. *Computer Methods in Applied Mechanics and Engineering* 2015; **284**:401–457.
6. Eriksson LE. Generation of boundary-conforming grids around wing-body configurations using transfinite interpolation. *AIAA Journal* 1982; **20**(10):1313–1320.
7. Wang ZJ, Przekwas AJ. Unsteady flow computation using moving grid with mesh enrichment. *AIAA Paper* 1994; **285**.
8. Allen CB. Aeroelastic computations using algebraic grid motion. *Aeronautical Journal* 2002; **106**(1064):559–570.
9. Allen CB. An algebraic grid motion technique for large deformations. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 2002; **216**(1):51–58.
10. Batina JT. Unsteady Euler algorithm with unstructured dynamic mesh for complex aircraft aerodynamic analysis. *AIAA Journal* 1991; **29**(3):327–333.
11. Farhat CH, Degand C, Koobus B, Lesoinne M. Torsional springs for two-dimensional dynamic unstructured fluid meshes. *Computer Methods in Applied Mechanics and Engineering* 1998; **163**(1):231–245.
12. Degand C, Farhat C. A three-dimensional torsional spring analogy method for unstructured dynamic meshes. *Computers & Structures* 2002; **80**(3):305–316.
13. Lynch DR, O'Neill K. Elastic grid deformation for moving boundary problems in two space dimensions. *3rd International Conference on Finite Elements in Water Resources, University of Mississippi*, vol. 2, Wang SY, Alonso C, Brebbia CA, Gray WG, Pinder GF (eds.), WIT Press, 1980.
14. Stein K, Tezduyar T, Benney R. Mesh moving techniques for fluid-structure interactions with large displacements. *Journal of Applied Mechanics* 2003; **70**(1):58–63.
15. Löhner R, Yang C. Improved ALE mesh velocities for moving bodies. *Communications in Numerical Methods in Engineering* 1996; **12**:599–608.
16. Helenbrook BT. Mesh deformation using the biharmonic operator. *International Journal for Numerical Methods in Engineering* 2003; **56**(7):1007–1021.
17. Liu X, Qin N, Xia H. Fast dynamic grid deformation based on Delaunay graph mapping. *Journal of Computational Physics* 2006; **211**(2):405–423.
18. De Boer A, Van der Schoot MS, Bijl H. Mesh deformation based on radial basis function interpolation. *Computers & Structures* 2007; **85**(11):784–795.
19. Van Zuijlen AH, de Boer A, Bijl H. Higher-order time integration through smooth mesh deformation for 3D fluid–structure interaction simulations. *Journal of Computational Physics* 2007; **224**(1):414–430.
20. Rendall TCS, Allen CB. Unified fluid–structure interpolation and mesh motion using radial basis functions. *International Journal for Numerical Methods in Engineering* 2008; **74**(10):1519–1559.
21. Rendall TCS, Allen CB. Efficient mesh motion using radial basis functions with data reduction algorithms. *Journal of Computational Physics* 2009; **228**(17):6231–6249.
22. Sheng C, Allen CB. Efficient mesh deformation using radial basis functions on unstructured meshes. *AIAA Journal* 2013; **51**(3):707–720.
23. Wang G, Mian HH, Ye ZY, Lee JD. Improved point selection method for hybrid-unstructured mesh deformation using radial basis functions. *AIAA Journal* 2014; **53**(4):1016–1025.
24. Rendall TCS, Allen CB. Improved radial basis function fluid–structure coupling via efficient localized implementation. *International Journal for Numerical Methods in Engineering* 2009; **78**(10):1188–1208.
25. Rendall TCS, Allen CB. Parallel efficient mesh motion using radial basis functions with application to multi-bladed rotors. *International Journal for Numerical Methods in Engineering* 2010; **81**(1):89–105.
26. Rendall TCS, Allen CB. Evaluation of RBFs for volume data interpolation in CFD. *International Journal for Numerical Methods in Fluids* 2013; **72**(7):748–769.
27. Bos FM, van Oudheusden BW, Bijl H. Radial basis function based mesh deformation applied to simulation of flow around flapping wings. *Computers & Fluids* 2013; **79**:167–177.
28. Witteveen JAS, Bijl H. Explicit mesh deformation using inverse distance weighting interpolation. *19th AIAA Computational Fluid Dynamics Conference, AIAA, San Antonio, Texas, AIAA Paper*, vol. 3996, 2009.
29. Witteveen JAS. Explicit and robust inverse distance weighting mesh deformation for CFD. *48th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, Orlando, Florida, AIAA Paper*, vol. 165, 2010.
30. Luke E, Collins E, Blades E. A fast mesh deformation method using explicit interpolation. *Journal of Computational Physics* 2012; **231**(2):586–601.
31. Samareh JA. Application of quaternions for mesh deformation. NASA TM-2002-211646, NASA Langley Research Center: Hampton, VA, United States, 2002.
32. Maruyama D, Bailly D, Carrier G. High-quality mesh deformation using quaternions for orthogonality preservation. *AIAA Journal* 2014; **52**(12):2712–2729.

DOI: 10.1002/nme.5465

International Journal for Numerical Methods in Engineering, vol. 111 (3), pp. 273–300, (2017).

33. Stadler D, Kosel F, Čelič D, Lipej A. Mesh deformation based on artificial neural networks. *International Journal of Computational Fluid Dynamics* 2011; **25**(8):439–448.
34. Zhou X, Li S. A new mesh deformation method based on disk relaxation algorithm with pre-displacement and post-smoothing. *Journal of Computational Physics* 2013; **235**:199–215.
35. Lefrançois E. A simple mesh deformation technique for fluid–structure interaction based on a submesh approach. *International Journal for Numerical Methods in Engineering* 2008; **75**(9):1085–1101.
36. Xuan Z, Shuixiang L, Bin C. Spring-interpolation approach for generating unstructured dynamic meshes. *Acta Aeronautica Et Astronautica Sinica* 2010; **31**(7):1389–1395.
37. Landry J, Soulaïmani A, Ali ABH. Robust mesh-mover algorithms for hybrid meshes. *22nd Annual Conference of the CFD Society of Canada, Toronto, Ontario, Canada*, 2014.
38. Gordon WJ, Hall CA. Construction of curvilinear coordinate systems and applications to mesh generation. *International Journal for Numerical Methods in Engineering* 1973; **7**(4):461–477.
39. Thompson JF, Warsi ZUA, Wayne Mastin CW. *Numerical Grid Generation: Foundations and Applications*, vol. 45. Elsevier Science Pub. Co. (New York : North-Holland), 1985.
40. Smith RE. Transfinite interpolation (TFI) generation systems. *Handbook of Grid Generation*, Thompson J, Soni BK, Wheaterill NP (eds.), CRC Press, Boca Raton, 1998; 3–15.
41. Paige CC, Saunders MA. Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis* 1975; **12**(4):617–629.
42. Shepard D. A two-dimensional interpolation function for irregularly-spaced data. *Proceedings of the 23rd ACM National Conference, Las Vegas, Nevada, USA*, ACM, 1968; 517–524.
43. Knupp PM. Algebraic mesh quality metrics for unstructured initial meshes. *Finite Elements in Analysis and Design* 2003; **39**(3):217–241.
44. Treibig J, Hager G, Wellein G. Likwid: A lightweight performance-oriented tool suite for x86 multicore environments. *Proceedings of PSTI2010, the First International Workshop on Parallel Software Tools and Tool Infrastructures*, San Diego CA, 2010.
45. Sen S, Kalita JC, Gupta MM. A robust implicit compact scheme for two-dimensional unsteady flows with a biharmonic stream function formulation. *Computers & Fluids* 2013; **84**:141–163.
46. Wood JN, De Nayer G, Schmidt S, Breuer M. Experimental investigation and large-eddy simulation of the turbulent flow past a smooth and rigid hemisphere. *Flow, Turbulence and Combustion* 2016; **97**(1):79–119.

DOI: 10.1002/nme.5465