EUROPEAN JOURNAL OF COMPUTATIONAL MECHANICS, 2016
VOL. 25, NO. 5, 417–445
http://dx.doi.org/10.1080/17797179.2016.1249732

Taylor & Francis
Taylor & Francis Group

Assessment and practical application of mapping algorithms for beam elements in computational FSI

Tianyang Wang, Roland Wüchner and Kai-Uwe Bletzinger

Chair of Structural Analysis, Technische Universität München, München, Germany

ABSTRACT

If a beam structure is modelled by 1D finite elements in a fluid–structure interaction (FSI) simulation, due to dimensional reduction in beam theory, the actual structural surface is not available which results in non-matching discrete geometries of the FSI interface. A linearised and a co-rotating algorithms applying the corresponding beam kinematics are proposed in another work to map deformation and load between the 1D beam mesh and the 2D surface mesh. In this work, the convergence behaviour of both algorithms is assessed with analytically described deformation. Besides, efficient treatments for both structured and unstructured fluid wet-surface meshes as well as treatment for connected beams are introduced as a practical extension to the core algorithm. Moreover, mapping is applied in the FSI simulation of a wind turbine where the rotor blades are modelled by beam elements.

ARTICLE HISTORY
Received 9 May 2016
Accepted 14 October 2016

KEYWORDS
Non-matching meshes; fluid-structure interaction; aeroelasticity; nonlinear beam; co-rotating formulation; wet surface reconstruction

1. Introduction

Fluid–structure interaction (FSI) couples numerical models of fluid and structure at the spatial interface (Bazilevs et al., 2013a, 2013b; Blom, 1998; Onate, Idelsohn, Celigueta, and Rossi, 2008; Piperno and Farhat, 2001). The global equation system can be solved either with the monolithic Blom (1998), Fernández and Moubachir (2005), Michler, Van Brummelen, and De Borst (2005), Piperno and Farhat (2001), Ryzhakov, Rossi, Idelsohn, and Oñate (2010) or the partitioned Degroote, Bruggeman, Haelterman, and Vierendeels (2008), Farhat, Van der Zee, and Geuzaine (2006), Küttler and Wall (2009), Mittal and Tezduyar (1995), Tezduyar, Behr, and Liou (1992a), (1992b) strategies. One main task in solving FSI problems is data mapping at the fluid–structure interface, where grids from both sides usually do not coincide. For immersed FSI problems, data need to be coupled between fluid on Eulerian coordinates and structure on Lagrangian coordinates, related works on this topic can be found in e.g. De Hart, Peters, Schreurs, and Baaijens (2003), Hou, Wang, and Layton (2012), Gerstenberger and Wall (2008), Peskin (2002), Rabczuk, Gracie, Song, and Belytschko (2010),

CONTACT Tianyang Wang tianyang.wang@tum.de

Wang, Chessa, Liu, and Belytschko (2008), Zhu and Peskin (2002). For fluids with arbitrary Eulerian-Lagrangian formulation Donea, Huerta, Ponthot, and Rodriguez-Ferran (2004), Hirt, Amsden, and Cook (1974), the deformation of the wet surface is mapped from the structure mesh to the fluid mesh while the load on the wet surface is mapped the other way round. Different mapping methods using interpolation Beckert and Wendland (2001), Press, Teukolsky, Vetterline, and Flannery (2007), Smith, Cesnik, and Hodges (2000) or weighted residual methods (mortar methods) Bernardi, Maday, and Patera (1994), Felippa, Park, and Ross (2010), Jaiman, Jiao, Geubelle, and Loth (2006), Klöppel, Popp, Küttler, and Wall (2011), Puso (2004), Puso and Laursen (2004), Unger, Haupt, and Horst (2007), Wohlmuth (2000) are developed for 2D interface meshes. A review of them can be found in de Boer, van Zuijlen, and Bijl (2008), Felippa et al. (2010), Wang, Wüchner, Sicklinger, and Bletzinger (2016).

A beam structure has one spatially dominant direction and therefore can be modelled and analysed as an elastic curve which is discretised by 1D elements with the finite element method (FEM). 1D models are widely used in aeroelastic analysis of beam-like structures such as wind turbine blades (Hansen, Sørensen, Voutsinas, Sørensen, and Madsen, 2006; Streiner, 2011), airplane wings (Patil, Hodges, and Cesnik, 2001; Varello, Demasi, Carrera, and Giunta, 2010) or other composite beams (Belver, Foces Mediavilla, Lorenzana Iban, and Rossi, 2010), as a simplification of modelling them with solid or shell elements (Bottasso, Campagnolo, Croce, and Tibaldi, 2013; Hsu and Bazilevs, 2012; Sicklinger, 2014; Sicklinger, Lerch, Wüchner, and Bletzinger, 2015). Additional cross section analysis is required in this case to extract the structural properties at different cross sections along a beam (Blasques, Bitsche, Fedorov, and Eder, 2013; Chen, Yu, and Capellaro, 2010; Malcolm & Laird, 2007). In structural wind engineering, slender buildings and bridges are also modelled as 1D curves (especially in early design stages) to analyse the dynamical response to the wind loads (Holmes, 2015; Simiu and Scanlan, 1996).

In FSI simulation of a beam structure, data have to be mapped between 1D beam elements and a fluid surface mesh. Different from the case with both surface meshes, the relation between the 1D elements and the surface mesh needs to be built. Since the 1D mesh usually has only a small number of elements, the ability to represent the rotation degrees of freedom (DOFs) is important for the mapping accuracy and the smoothness of the deformed surface. In both Ahrem, Beckert, and Wendland (2007) and Cordero-Gracia, Gómez, and Valero (2014) radial basis function (RBF) model is applied for spatial interpolation. The former includes linearised rotations into the RBF model, while the latter creates a new intermediate mesh so that mapping techniques for surface meshes can be reused. The two methods require a big number of 1D elements and are restricted to small rotations. However, geometrical nonlinearity should be considered when mapping on flexible beam structures that undergo large deformation and rotation (Bathe and Bolourchi, 1979). A co-rotating mapping algorithm is introduced in

Wang et al. (2016) which can recover large displacements and rotations. It is inspired by the co-rotating formulation of nonlinear beam elements (De Rosis, Falcucci, Ubertini, and Ubertini, 2013; Felippa & Haugen, 2005; Krenk, 2009; Li and Vu-Quoc, 2010).

This work aims at providing a convergence analysis of the mapping algorithms for beam elements in Wang et al. (2016) and extending the core algorithm in order to treat different practical problems. The paper starts with the review of the linearised and co-rotating algorithms in Section 2. The convergence behaviour of both algorithms is systematically tested in Section 3. Besides, extension to the core algorithm with respect to efficient determination of cross section centre and treatment of connected beams is introduced in Section 4. Moreover, the mapping technique is applied in the FSI simulation of a wind turbine where the blades are modelled by beam elements. Finally, the work is concluded in Section 6.

2. Review of linearised and co-rotating algorithms

Both mapping algorithms are based on three assumptions: the cross sections of a beam remain rigid during deformation (no warping); the cross sections are always orthogonal to the beam axis during deformation; the shear centre and the elastic (tension) centre coincide and lie on the 1D beam model. The first two assumptions correspond to the Bernoulli beam theory. Based on these assumptions, motion of a cross-section can be described by a single rigid-body motion operator defined at the centre. These assumptions are also used in the theory of elastica for slender beams. But they are not valid in certain cases such as Timoshenko beams or warping cross sections, then the mapping algorithms should be extended by considering more parameters such as shear angles or warping functions. Nevertheless, the algorithms have already a wide range of applications and can give approximate solution when these factors are not severe.

The mapping algorithms apply an RBM operator $\Phi : R^3 \rightarrow R^3$ which is defined as:

$$\Phi(\mathbf{x}) = \mathbf{R}\mathbf{x} + \mathbf{t}, \tag{1}$$

where $\mathbf{R}$ is a rotation matrix (3×3), $\mathbf{t}$ a translation vector and $\mathbf{x}$ the coordinates of a point. Note that the operator is always associated with a certain coordinate system and especially, the axis of rotation always passes the origin of the coordinate system. A rotation matrix can be computed out of an angle θ around a unit vector $\mathbf{n}$, or a rotation vector $\boldsymbol{\theta} = \theta\mathbf{n} = (\theta_x, \theta_y, \theta_z)^T$ and *vice versa*. The relation between them can be expressed by Krenk (2009)

$$\mathbf{R} = \cos\theta\mathbf{I} + \sin\theta\hat{\mathbf{n}} + (1 - \cos\theta)\mathbf{n}\mathbf{n}^T, \tag{2}$$

where $\mathbf{I}$ is the identity matrix and $\hat{\mathbf{n}}$ (3×3 matrix) is a compact form of cross product, i.e.

$$\hat{\mathbf{n}}\mathbf{x} = \mathbf{n} \times \mathbf{x}.$$

For a small rotation (2) can be linearised as

$$\mathbf{R} \approx \mathbf{I} + \theta \hat{\mathbf{n}}. \tag{3}$$

In this case, the entries in the rotation vector can be regarded as three individual rotations around the axes of the coordinate system.

In the following, the centre of a cross section is denoted by P, and the RBM of it is denoted by Φ_P. The linearised and the co-rotating algorithms introduced in Wang et al. (2016) for interpolation of Φ_P are briefly reviewed here.

2.1. Linearised algorithm

The 1D beam element in this work has two end nodes. On each node, there are six DOFs including a displacement vector and a rotation vector. Before interpolation, the DOFs need to be transformed to the local coordinate system, so that axial displacements and torsional rotations are interpolated linearly along a finite element, whereas cubic interpolation can be performed for out of plane displacements and bending rotations. So displacement vector $(v_{Xi}, v_{Yi}, v_{Zi})^T$ and rotation vector $(\theta_{Xi}, \theta_{Yi}, \theta_{Zi})^T$ defined on the global coordinate system (GCS) are transformed to $(v_{xi}, v_{yi}, v_{zi})^T$ and $(\theta_{xi}, \theta_{yi}, \theta_{zi})^T$ which are defined on the element coordinate system (ECS) (see Figure 1). Here i is the node index with $i = 1$ or 2.

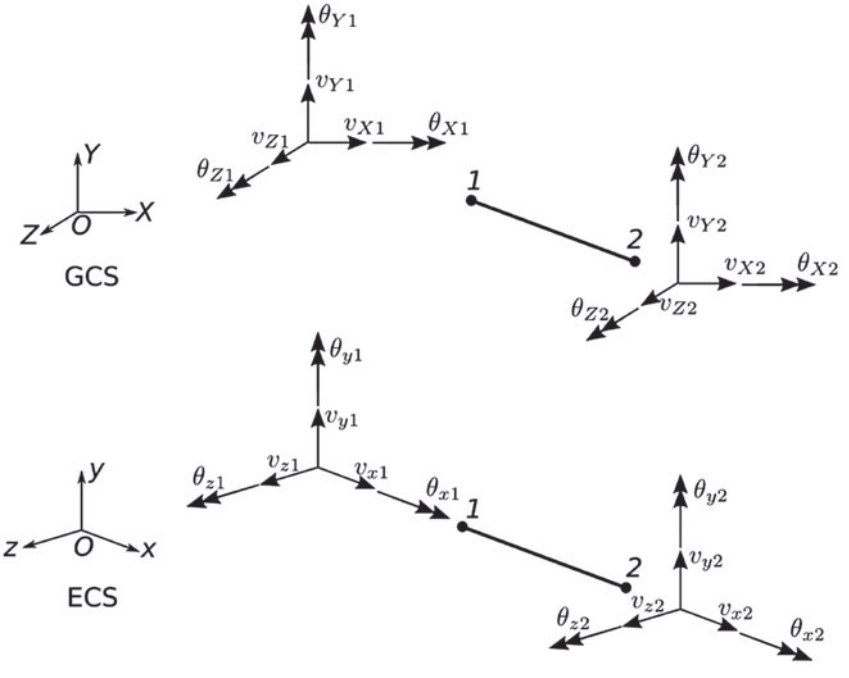


Figure 1. DOFs of a beam element corresponding to different coordinate systems Wang et al. (2016).

Assume the parametric coordinate ξ of P on an 1D element is

$$\xi = \frac{2(x - x_1)}{l} - 1, \tag{4}$$

where x and x_1 are, respectively, the x-coordinate of P and node 1 on the ECS, and l is the length of the element. ($l = x_2 - x_1$ where x_2 is the x-coordinate of node 2 on the ECS).

Define $(v_x, v_y, v_z)^T$ and $(\theta_x, \theta_y, \theta_z)^T$ as the displacement vector and the rotation vector on P. They can be computed through the interpolation on the 1D element. Related equations can be found in text books on FEM for beams. For convenience, they are summarised in Appendix 1.

$(v_x, v_y, v_z)^T$ and $(\theta_x, \theta_y, \theta_z)^T$ are finally used to construct Φ_P.

2.2. Co-rotating algorithm

The linearised algorithm cannot be used for large rotations. In the co-rotating algorithm, the beam element is first moved to a co-rotating configuration so that large rotations are reduced to relatively small ones. Then the linearised algorithm can be reused in this configuration for interpolation. The co-rotating algorithm contains three steps as shown in Figure 2:

(1) Displace the end nodes according to $(v_{xi}, v_{yi}, v_{zi})^T$. The beam element remains straight.
(2) Axial rotation. Now the co-rotating configuration is obtained.
(3) Twist and bend the element. Large rotations are already carried out in the first two steps. The third step includes only small rotations which can be linearised. So the linearised algorithm in Section 2.1 is reused for interpolation.

Define Φ_d, $\mathbf{R}_s$ and Φ_l as the RBM operators for the three steps as shown in Figure 2, then Φ_P can be computed as:

$$\begin{aligned} \Phi_P &= \Phi_d \circ \Phi_l \circ \mathbf{R}_s \circ \Phi_d^{-1} \circ \Phi_d \\ &= \Phi_d \circ \Phi_l \circ \mathbf{R}_s. \end{aligned} \tag{5}$$

For the details of the computation of Φ_d, $\mathbf{R}_s$ and Φ_l, the reader is referred to Wang et al. (2016).

2.3. Conservative load mapping

In load mapping, nodal forces on the surface are used to compute forces and moments on 1D elements. The force vector $\mathbf{f}_Q$ on a surface node Q is transferred to P first, as illustrated in Figure 3. This results in a force vector $\mathbf{f}_P$ and a moment

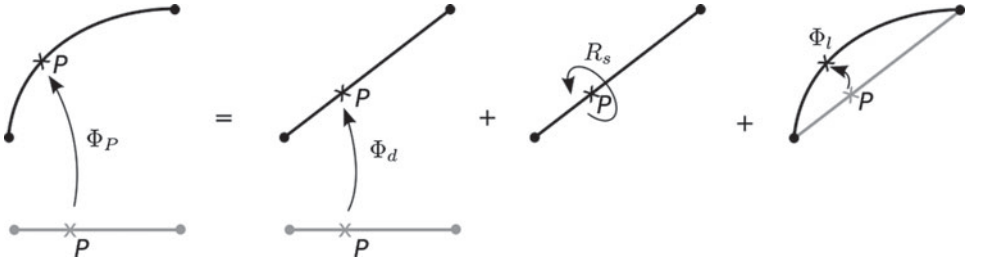


Figure 2. The rigid body motion of a cross section can be decomposed into three steps. The cross represents the cross section Wang et al. (2016).

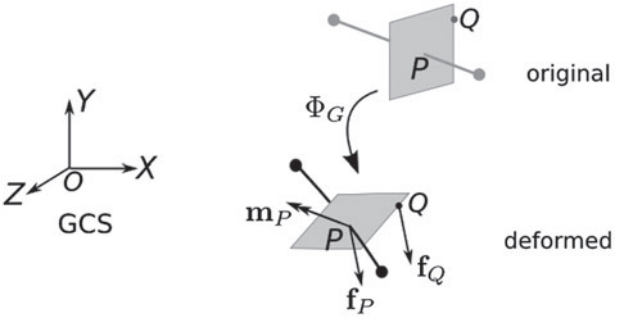


Figure 3. Transfer a nodal force to the centre of the cross section.

vector $\mathbf{m}_P$ on P with

$$\mathbf{f}_P = \mathbf{f}_Q, \tag{6a}$$

$$\mathbf{m}_P = \overrightarrow{PQ} \times \mathbf{f}_Q, \tag{6b}$$

where $\overrightarrow{PQ}$ is the difference vector from P to Q in the deformed configuration.

The next task is to transfer $\mathbf{f}_P$ and $\mathbf{m}_P$ to the end nodes of the corresponding 1D element. This can be achieved by computing the so-called *consistent nodal forces* and *moments* (Kindmann & Kraus, 2004; Wunderlich & Kiener, 2004), i.e. the forces and moments are allocated to the end nodes with the same shape functions as used for DOFs. The equations corresponding to linear beam elements are summarised in Appendix 1.

For nonlinear 1D elements, the increments in DOFs $(\mathrm{d}\, v_{xi}, \mathrm{d}\, v_{yi}, \mathrm{d}\, v_{zi})^T$ and $(\mathrm{d}\, \theta_{xi}, \mathrm{d}\, \theta_{yi}, \mathrm{d}\, \theta_{zi})^T$ need to be aligned with the co-rotating configuration first. Since the increments are small, the same shape functions used for the DOFs in (A1) and (A3) can be used for the increments. To apply conservative mapping, $\mathbf{f}_P$ and $\mathbf{m}_P$ are transformed to the co-rotating configuration first. Since they are conjugated with the increments now, they can be used to compute the consistent nodal forces and moments with (A5).

The above process is repeated for each surface node, so that the forces and moments are accumulated on the 1D elements.

3. Convergence tests

A beam with constant square cross section is used to test the convergence behaviour of the mapping algorithms. The length of it in the x direction is $l = 10$ and the square cross sections in the $y - z$ plane have a constant size as $a = 1 \times 1$ (x, y and z are the axes of the GCS from here on). The surface of the beam is uniformly meshed with 100 elements in the x direction and 10 elements in the other two directions. The 1D structure model locates at the centre of the beam. It is refined to study the convergence behaviour of mapping, i.e. there are

2^k elements with $k \in \{1, 2, 3, 4, 5, 6\}$. The 1D model with $k = 4$ and the surface mesh is shown in Figure 4.

3.1. Bending

The beam is clamped at the left end and bent into an arc in the $x-y$ plane without changing its length. An arbitrary point $\mathbf{P} = (\mathrm{P}_x, \mathrm{P}_y)$ in the initial configuration is defined as

$$\begin{aligned} \mathrm{P}_x &= tl, \\ \mathrm{P}_y &= 0, \end{aligned} \tag{7}$$

where t is the single parameter marking the same physical cross section in different configurations (or can be understood as a material point), with $0 < t < 1$. The point in the final configuration is defined as

$$\begin{aligned} \mathrm{P}_x &= r \sin(t\alpha), \\ \mathrm{P}_y &= -(r - r\cos(t\alpha)), \end{aligned} \tag{8}$$

where α is the rotation angle (slope) at the right end and $r = \frac{l}{\alpha}$ is the radius of the arc.

The displacements and the bending rotation at t can be derived as

$$\begin{aligned} \theta_z &= -t\alpha, \\ u_x &= r \sin(-\theta_z) - tl, \\ u_y &= -(r - r\cos(-\theta_z)). \end{aligned} \tag{9}$$

Three test cases are used with $\alpha = 20°, 40°$ and $60°$ resulting in the arcs plotted in Figure 5.

Since analytical functions describing the deformation of the beam are available, a *discrete* L2 norm de Boer et al. (2008) can be used to compute mapping errors: the functions are used to derive DOFs on beam elements; then the DOFs are mapped to the fluid surface to get nodal displacements; whereas exact nodal displacements can be derived directly from the analytical functions; the L2 norm of the deviation on all nodes is defined as the mapping error. 'discrete' means that the error is evaluated on discrete nodes instead of on a continuous domain. For this test case, the DOFs are computed according to (9) and are mapped to the surface nodes with both the linearised and the co-rotating algorithms. Specially, the exact displacements on surface nodes are computed from RBMs at the axis defined by (9). The equation of mapping error can be derived as

$$e = \sqrt{\frac{\sum_{i=1}^{n_s \times 3} (U_i - U_i')^2}{n_s \times 3}}, \tag{10}$$

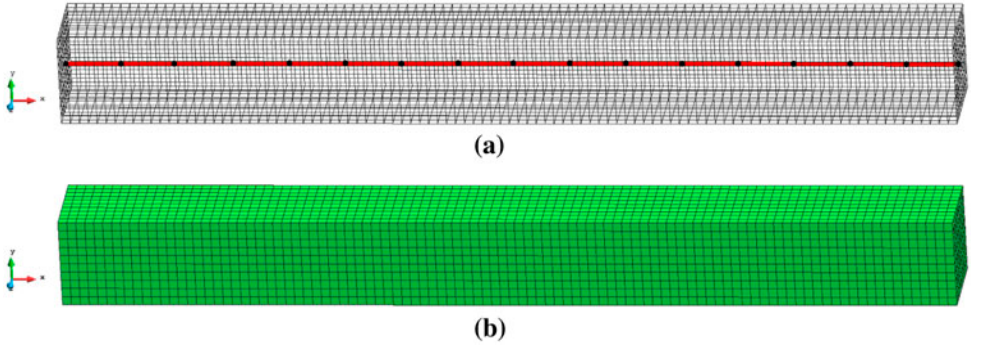


Figure 4. The curve and surface meshes of a beam with square cross section. The surface mesh is displayed with two different styles.

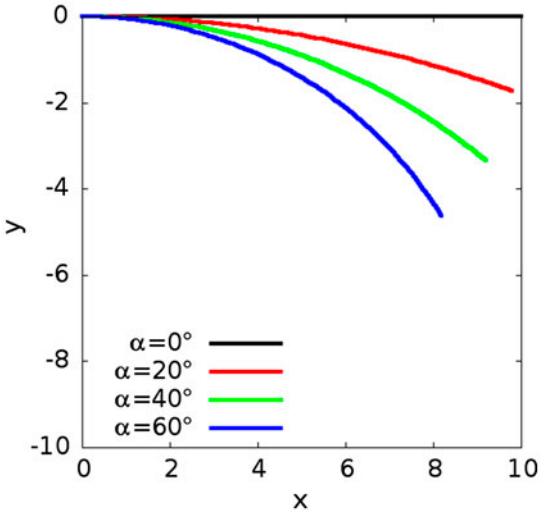


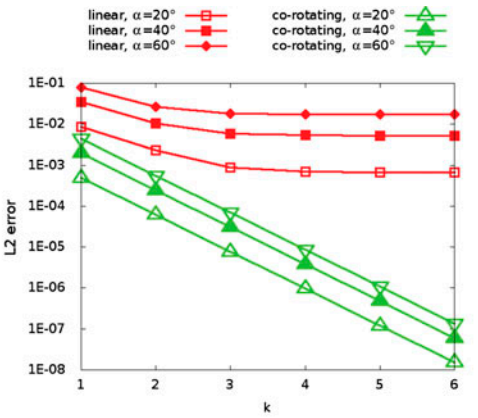
Figure 5. Plot of curves in (8) with different values of α.

where U are mapped displacements on all surface nodes and U' the exact ones.

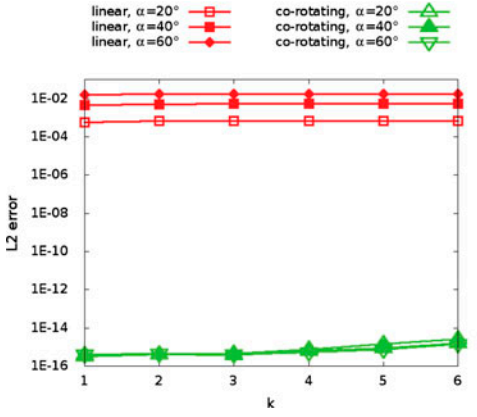
The mapping error with respect to different refinement levels is shown in Figure 6(a). Moreover, the errors from the rotation and the translation part are separately plotted in Figure 6(b) and (c). They are obtained by eliminating either translation or rotation in the interpolated RBM operators. Deformed surface meshes with $k = 4$ are shown in Figures 7 and 8 as examples. From the figures it can be seen that

- The linearised algorithm is not convergent and the rotations are not accurately interpolated leading to distorted elements;
- The co-rotating algorithm gives almost no error in rotations in this special case;
- The co-rotating algorithm gives smaller total error which converges with an order of 2.08.

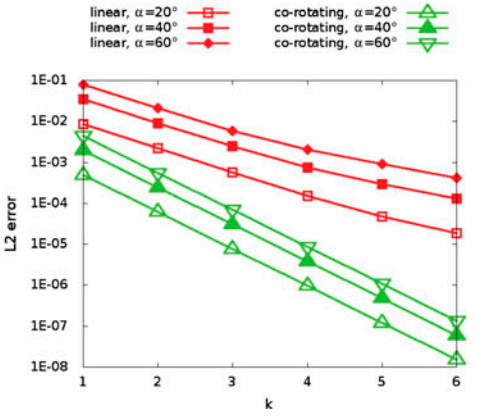
The convergence behaviour of the co-rotating algorithm is explained here. The two end nodes of an element are displaced first to obtain the co-rotating configuration as shown in Figure 9, where a new coordinate system is defined, whose x' axis is aligned with the element. Three functions need to be interpolated, namely the displacements $u_{x'}$, $u_{y'}$ and the bending angle $\theta_{z'}$. Since there are $u_{x'1} = u_{x'2} = u_{y'1} = u_{y'2} = 0$ and $\theta_{z'1} = -\theta_{z'2}$, Hermitian cubic interpolation with (A3) results in linear distribution of the bending angles. This agrees with the exact solution since the angle on an arc is exactly linearly distributed. As for the displacements, the result of linear interpolation is that material points in the co-rotating configuration are equally distributed (i.e. a linearly distributed parametric coordinate ξ), and moreover, axial displacements on them are zero. But since the arc lengths at all points are constant during deformation, the axial displacements are not zero in general as depicted in Figure 9. So linear



(a) Total error



(b) Error from the rotation part



(c) Error from the translation part

Figure 6. Mapping errors from the linearised and the co-rotating algorithms with different α.

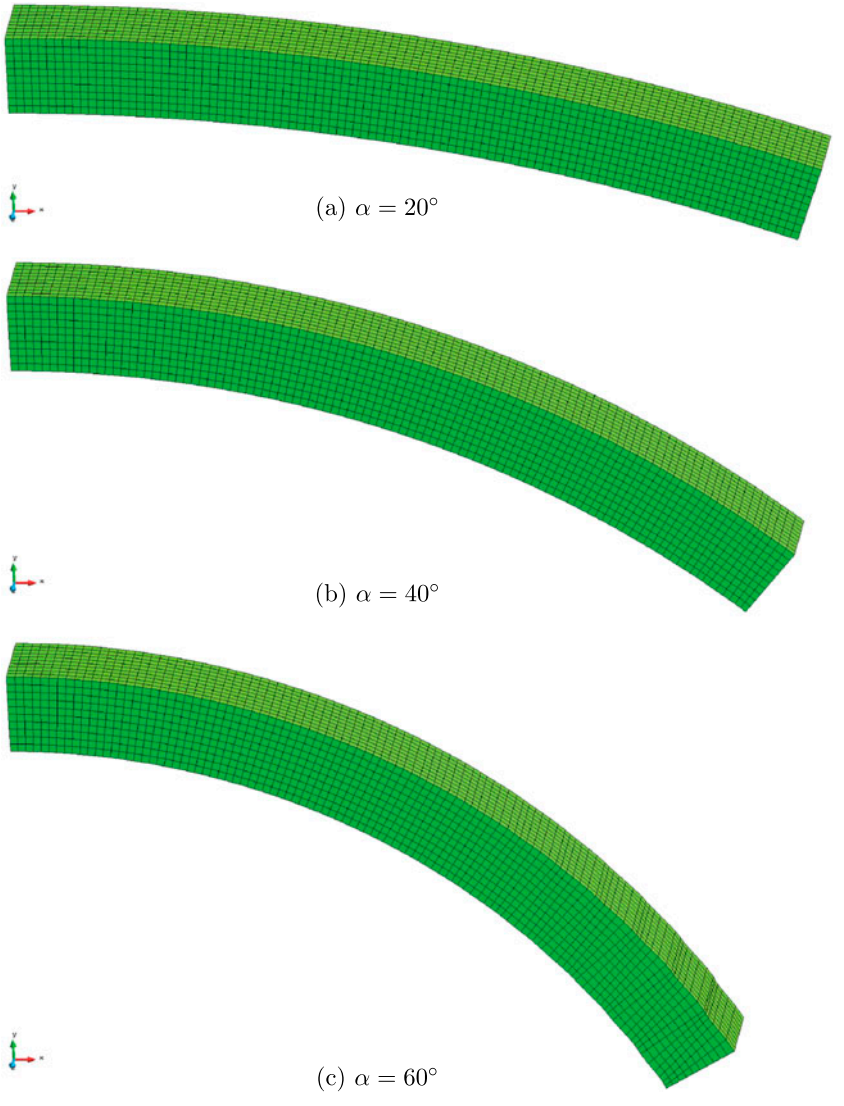


Figure 7. Results of the linearised algorithm with 16 1D elements under different α.

interpolation leads to second-order convergence rate, although Hermitian cubic interpolation is more accurate for displacements in the perpendicular direction.

3.2. Bending and twist

The beam is twisted first and then bent with the same analytical function as in Section 3.1. The twist angle is linearly distributed along the length as

$$\theta_x = t\beta, \tag{11}$$

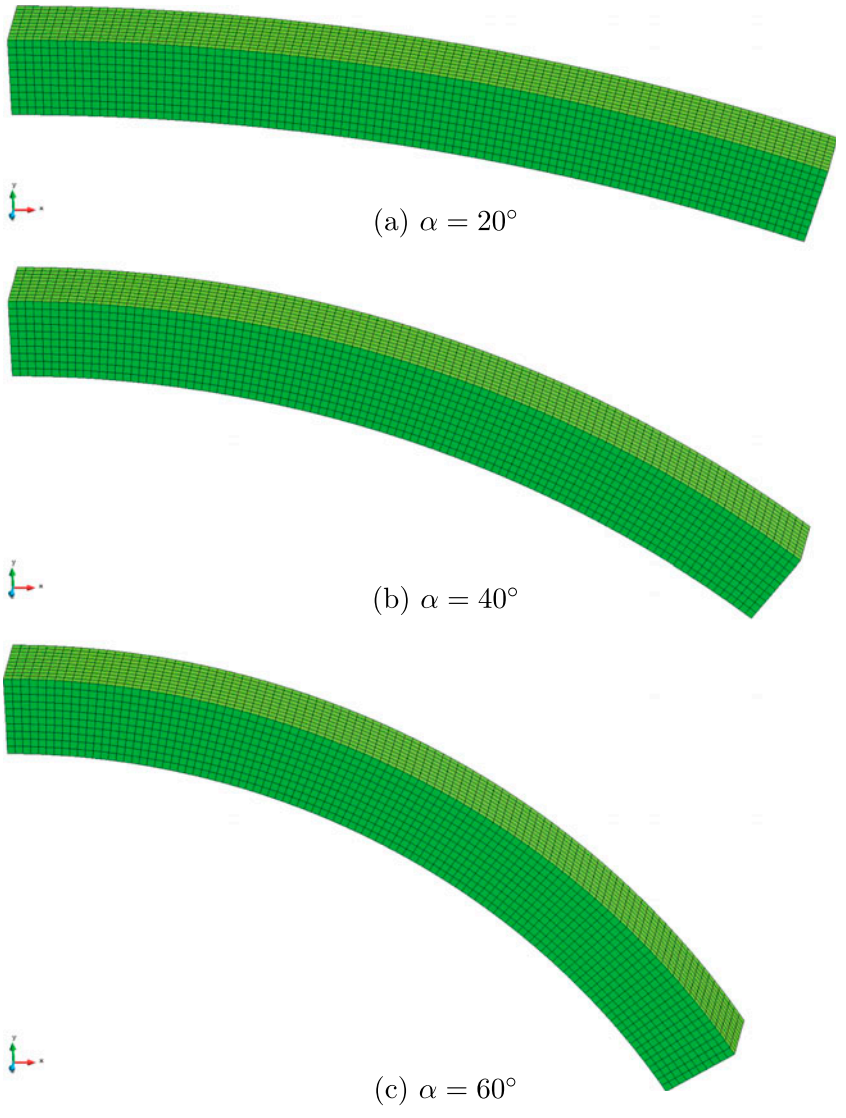


Figure 8. Results of the co-rotating algorithm with 16 1D elements under different α.

where β is the twist angle at the right end. Three tests with $\alpha = \beta = 20°$, $40°$ and $60°$ are performed. The mapping error is evaluated in the same way as in the pure bending case.

The mapping error with respect to different refinement levels of the 1D model is shown in Figure 10, where the total error as well as that from the translation and the rotation part are shown.

It can be seen that the co-rotating algorithm is again more accurate, but the convergence order is reduced compared with the pure bending case, e.g. it lies between 1.93 and 1.39 for $\alpha = \beta = 60°$. The reason is explained below.

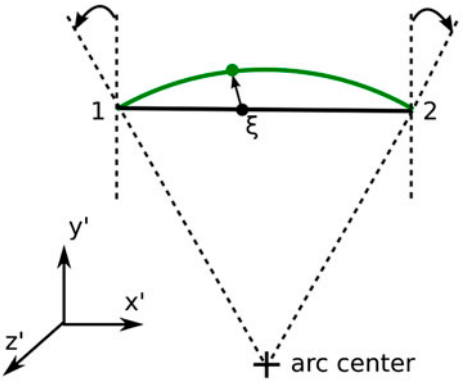


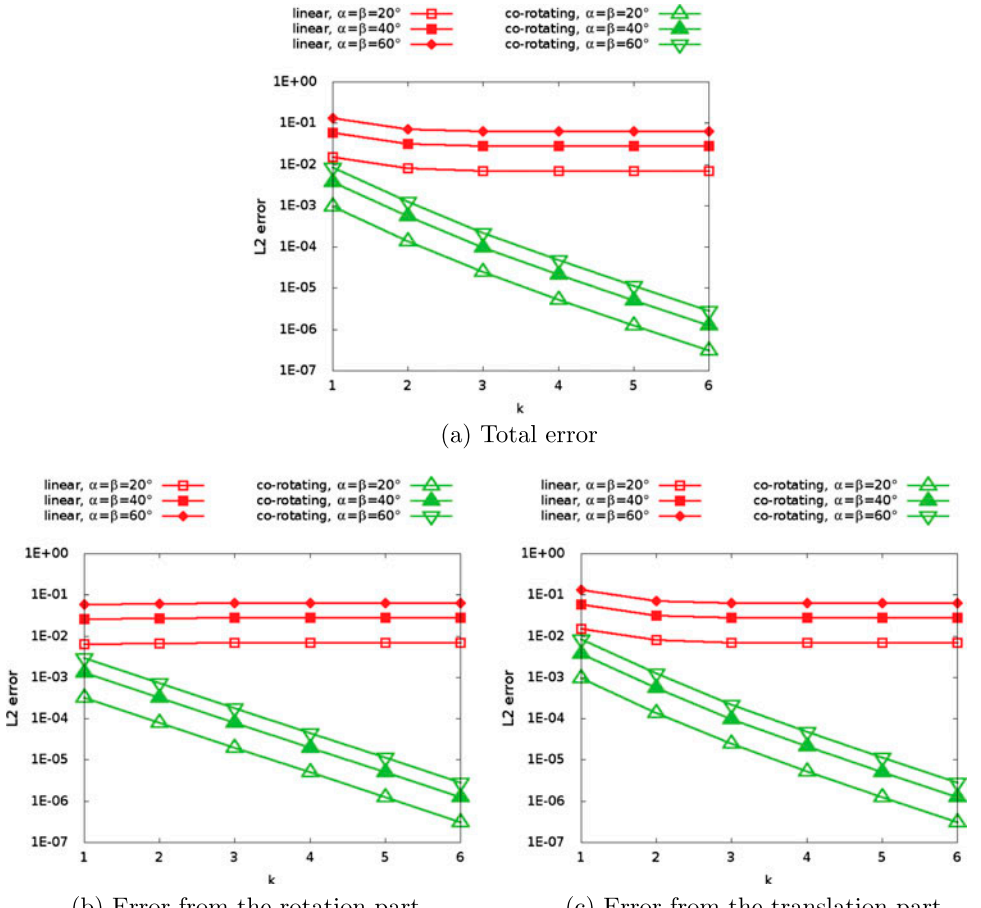
Figure 9. A beam element in the co-rotating configuration which will be further bended to an arc. The displacement vector on a general point ξ is not perpendicular to the element, because arc lengths at all points are prescribed as constant.

After the first two steps in the co-rotating algorithm, large rotations are removed and only relatively small rotations are left. They are separated into three rotations and become the rotational DOFs. It is only an approximation, since performing the three rotations sequentially is not equivalent to the original 3D rotation. From (2) and (3), it can also be seen that the approximation relies on the assumptions $\cos(\theta) \approx 1$ and $\sin(\theta) \approx 0$. Therefore, the separation brings errors: the bending angles on the two end nodes are not exactly opposite so an exact linear distribution of bending angles cannot be achieved; there is numerical bending in $x' - z'$ plane since $\theta_{y'1}$ and $\theta_{y'2}$ are non-zero. Compared with the pure bending problem where numerical errors come from interpolation, this is a new error source which leads to the reduction in the convergence order. This can also be observed from Figure 10(b) where the error form the rotation part is not close to zero any more compared with that in the pure bending case. However, the length of the beam is 10, while the edge length of all surface elements is .1, with the co-rotating algorithm the error in L2 norm for the largest bending and twist case is $\approx 10^{-2}$ with $k = 2$ and $\approx 10^{-6}$ with $k = 6$. This verifies the good quality of the results.

The deformed surface meshes with $k = 4$ are shown in Figures 11 and 12. It can be seen that the surface mesh from the co-rotating algorithm is again smooth and does not have distorted elements. This is due to fact that both translations and rotations computed from the co-rotating algorithm are convergent.

4. Practical treatments

To apply the linearised and co-rotating algorithms, each surface node needs to be linked to a point on the beam axis, which is the centre of the cross section passing the node. Efficient approaches for both structured and unstructured meshes are introduced in the first two sub-sections. When using these approaches on



(a) Total error

(b) Error from the rotation part

(c) Error from the translation part

Figure 10. Mapping errors from the linearised and the co-rotating algorithms with different α and β.

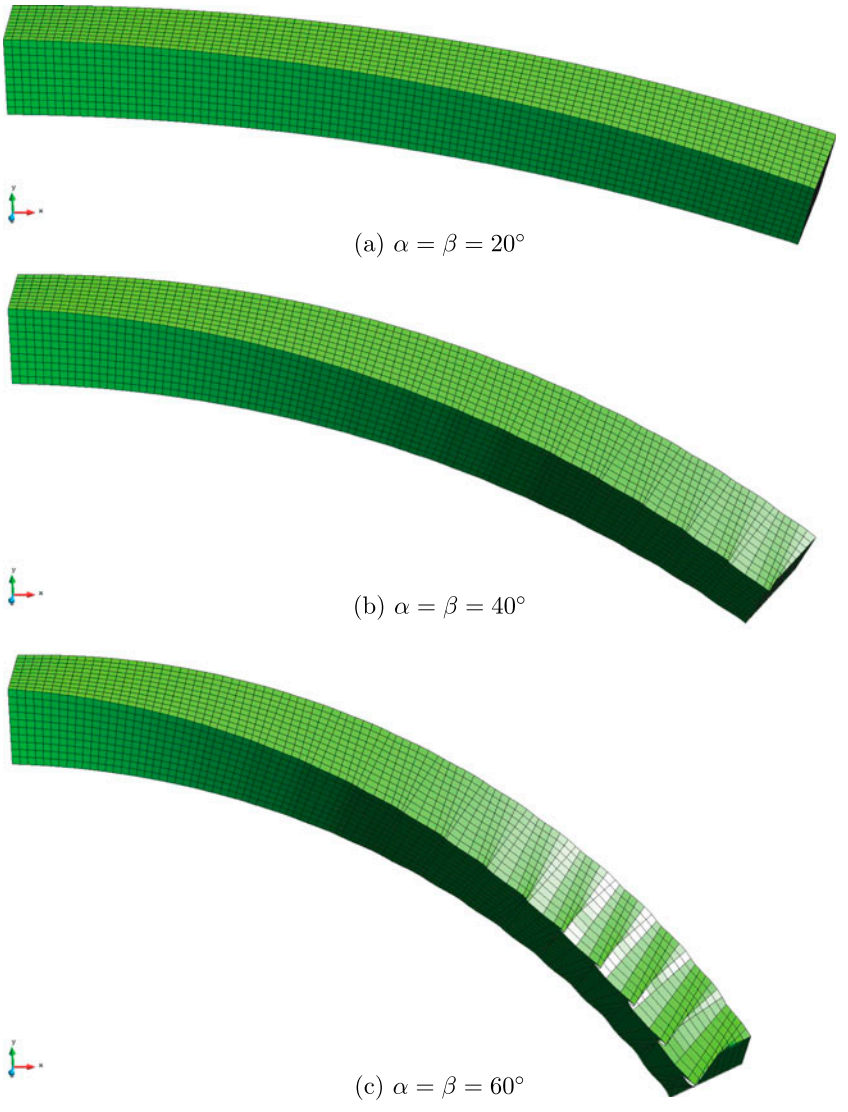


Figure 11. Results of the linearised algorithm with 16 1D elements under different α and β.

connected beams, surface nodes close to the joints cannot be linked to the axes of beams, therefore a practical treatment is suggested in the last sub-section.

4.1. Determine cross section centres

A simple and efficient algorithm is implemented to determine cross section centres, where all cross sections are assumed parallel in the original configuration. It is valid since a beam structure is straight or only slightly curved in most cases, i.e. the tangential direction of the beam axis does not change with a considerable amount. With the normal of the parallel cross sections $\mathbf{n}_{cs}$, the plane of the cross section passing a surface node Q can be obtained as shown in Figure 13. The intersection P between the plane and an 1D element is defined as the centre of

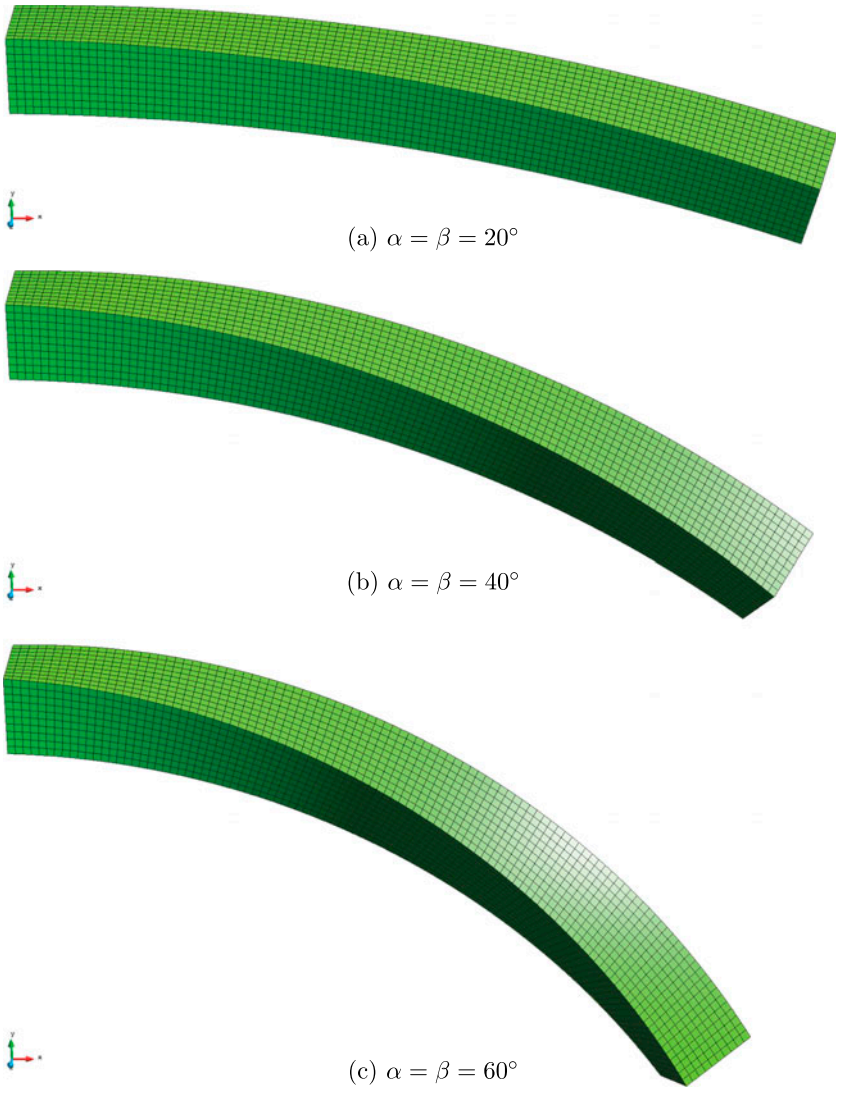


Figure 12. Results of the co-rotating algorithm with 16 1D elements under different α and β.

the cross section. The process above can be easily implemented: the nodes on both meshes are projected to the line aligned with $\mathbf{n}_{cs}$ (which is the axis of the beam if it is not curved), then P is simply equal to the new coordinate of Q in the direction $\mathbf{n}_{cs}$.

Some remarks regarding the accuracy are given below:

- There is numerical error if the computed centre is not exact, e.g. in case that some 1D elements do not locate exactly at the beam axis, or the elastic centre and the shear centre do not coincide. As a result, the translation of the cross section is accurate, while the rotation is inexact, i.e. the rotation radii are wrong though the orientation of the cross section after rotating is

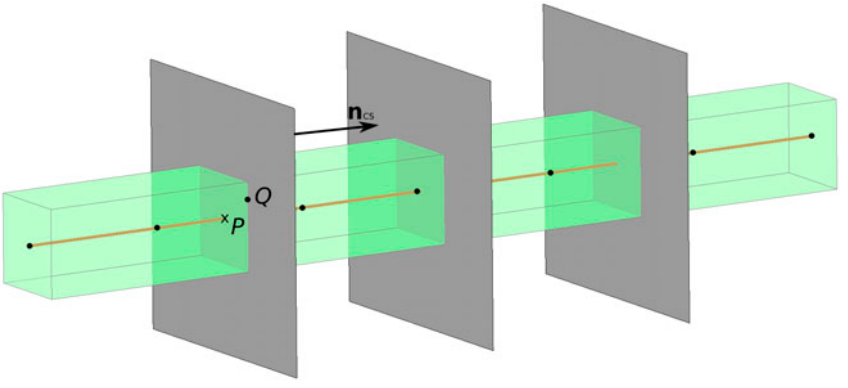


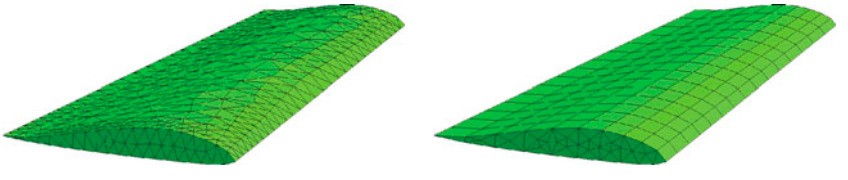
Figure 13. The planes of cross sections are assumed parallel. The cross section passing a surface point Q intersects with an 1D element at P, which is defined as the centre of the cross section Wang et al. (2016).

still correct. But the overall deformed shape is satisfactory except that some cross sections are shifted around the correct locations.

- There is also numerical error if the beam is curved, since cross section are assumed parallel. This leads to error in shape details, e.g. wrong distribution of angles of attack along an airwing. This is why only slightly curved beams are allowed by the approaches.
- The two error sources above can also lead to numerical error in load which is computed in an energy conjugated way. But similarly, the error can be regarded as a local effect which is bounded.

4.2. Treatment for structured mesh

The efficiency can be further improved if the surface mesh is a structured mesh, see Figure 14(b). In this case, the surface nodes are projected as before to the line aligned with $\mathbf{n}_{cs}$, then they are sorted by the new coordinate in the $\mathbf{n}_{cs}$ direction. A structured mesh usually has the same number of nodes n_r for each row along the beam length direction, therefore every n_r nodes are grouped together according to the position sorting results. The nodes at the tip or the root surface should be separately grouped if their number is different from n_r. The nodes belonging to the same group are regarded as also belonging to the same cross section and are

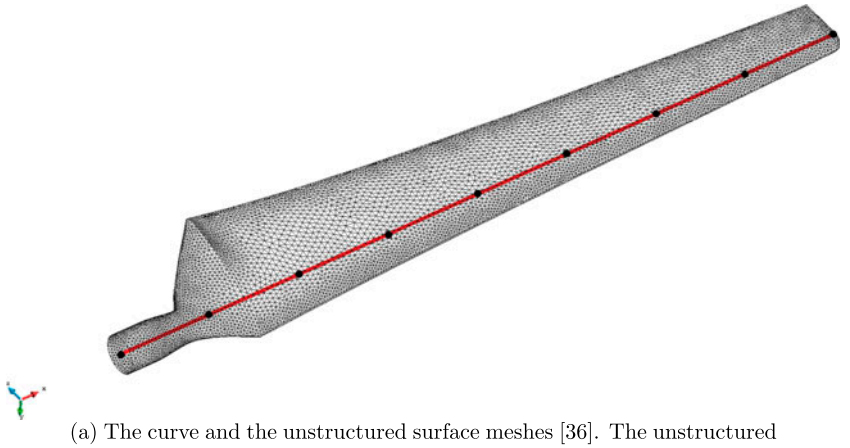


(a) unstructured (b) structured except at the root

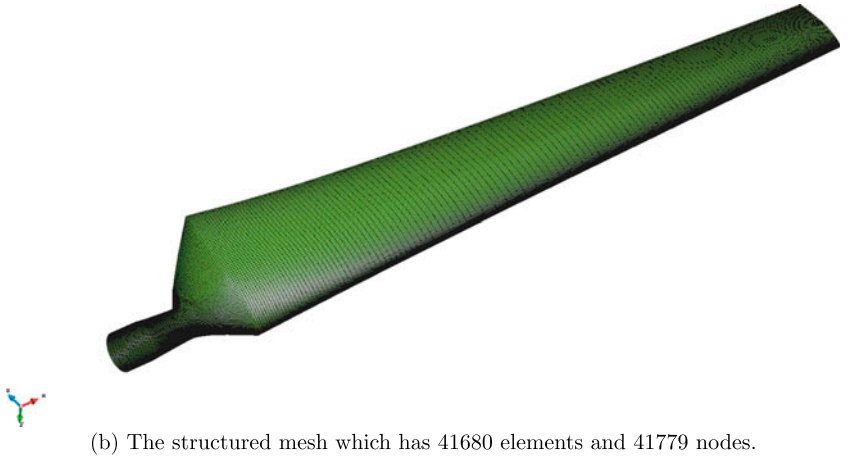
Figure 14. Two meshes of the same surface. On the structured mesh, nodes belonging to the same cross section can be grouped together, so that the number of RBM operators to be computed can be reduced.

assigned with the same RBM. Pre-processing/meshing usually cannot guarantee that nodes of the same group have the same coordinate in the $\mathbf{n}_{cs}$ direction, so only the average of all coordinates is used to determine the centre of the cross section.

The treatment for structured and unstructured meshes in combination with the co-rotating algorithm is tested on the blade of the NREL phase VI wind turbine (Hand et al., 2001). The beam axis is a straight line coinciding with the axis of the cylindrical root section. It is uniformly meshed by 8 elements. Two surface meshes are used as shown in Figure 15: an unstructured one with triangles and a structured one with quadrilaterals except a number of triangles at the tip. The mapping with the unstructured mesh is also presented in Wang et al. (2016), and it is shown here as a comparison to the mapping with the structured mesh. For the unstructured mesh, a rigid body motion operator has to be computed for each fluid node. For the structured mesh, each 198 fluid nodes along the beam



(a) The curve and the unstructured surface meshes [36]. The unstructured mesh has 20876 elements and 10458 nodes.



(b) The structured mesh which has 41680 elements and 41779 nodes.

Figure 15. The curve and the unstructured surface meshes Wang et al. (2016). The unstructured mesh has 20876 elements and 10458 nodes.

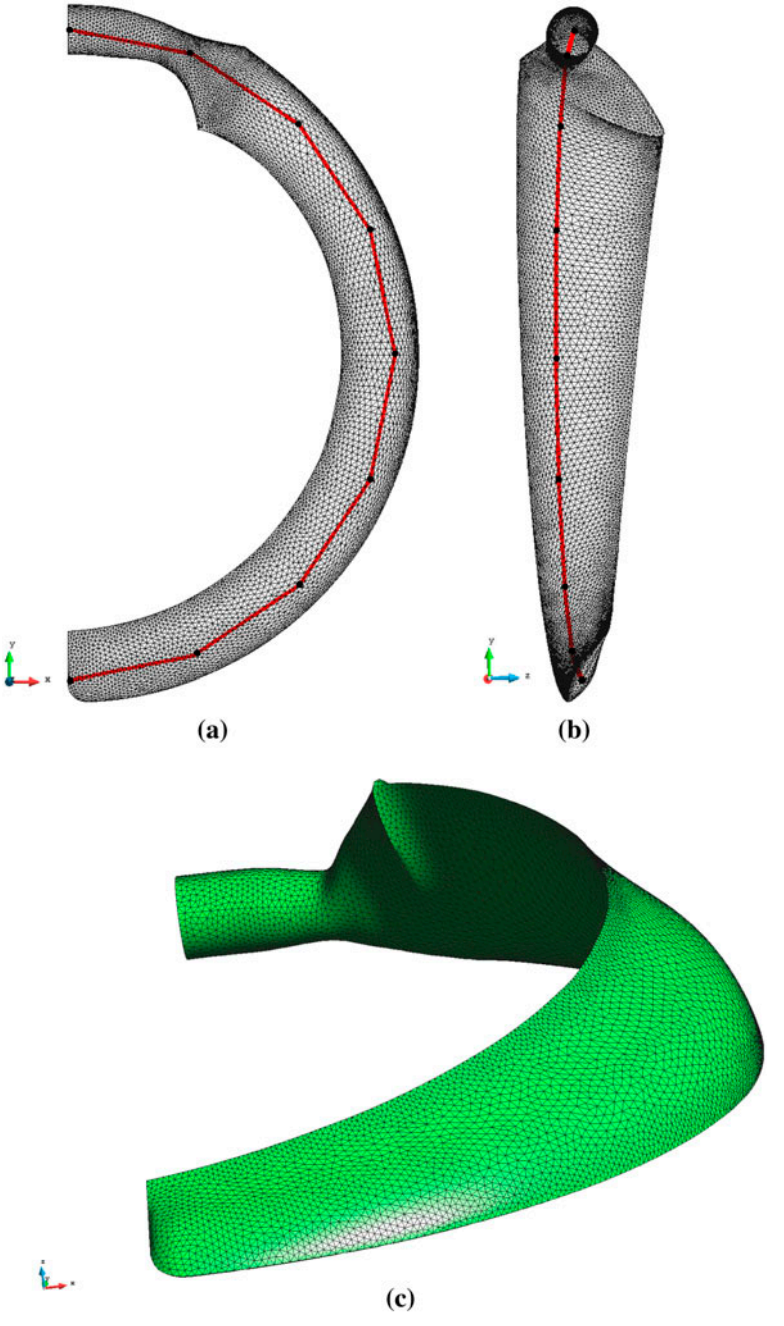


Figure 16. The curve and the unstructured surface meshes of the blade after deformation with three different views (Wang et al., 2016). The unstructured surface mesh is displayed with two different styles.

length direction are regarded as belonging to the same cross section, except that the 793 nodes at the blade tip are separately grouped. It results in only 208 rigid body motion operators for the 41779 nodes. The treatment is quite practical since resolving the boundary layer of turbulent flows usually leads to structured

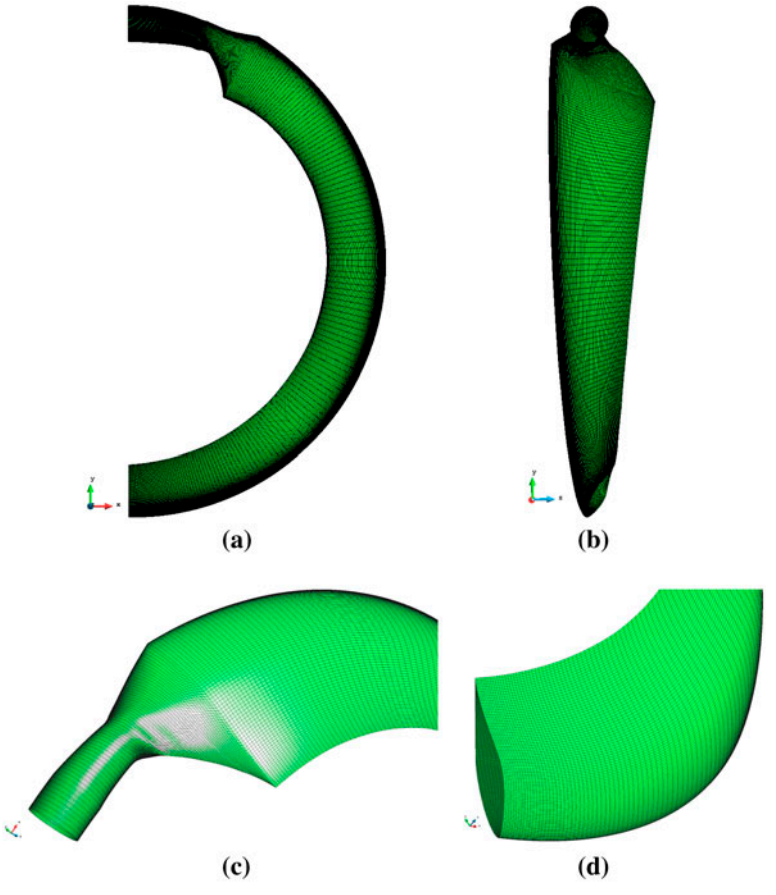


Figure 17. The curve and the structured surface meshes of the blade after deformation with four different views. The last two views zoom in at the root and tip due to the high refinement.

boundary meshes. The beam is applied with twist and flapwise bending that are described by the same analytical functions used in Section 3.2. Respective angles in these functions are set as $\beta = 60°$ and $\alpha = 180°$, which means the beam is bent into a half circle. Such big deformations do not happen in reality but it is used to test the performance of the co-rotating algorithm. The results are shown in Figures 16 and 17. It can be seen that the co-rotating algorithm is still able to deliver smooth surfaces.

4.3. Treatment for connected beams

As introduced in Section 4.1 and 4.2, surface nodes need to be projected to the beam axis in order to obtain the rigid-body motions on them. This works for single beams. But for connected beams as shown in Figure 18, some surface nodes around the joint either cannot be projected onto the beams (e.g. Q_1) or have projections on both beams (e.g. Q_2), so their rigid-body motions cannot be determined.

One treatment is to model the part of the beams close to the joint as rigid in structural analysis, and the rest part is still elastic. As a result, surface nodes around the joint share the same rigid-body motion, so they can be linked together to an arbitrary position on the rigid part of the beams. Taking again the example in Figure 18, partition b is modelled as infinitely rigid, which does not participate in the mapping. The mapping is performed in two groups: between a and $A \cup B$ as well as between c and C. The algorithm in Section 4.1 is extended for the mapping

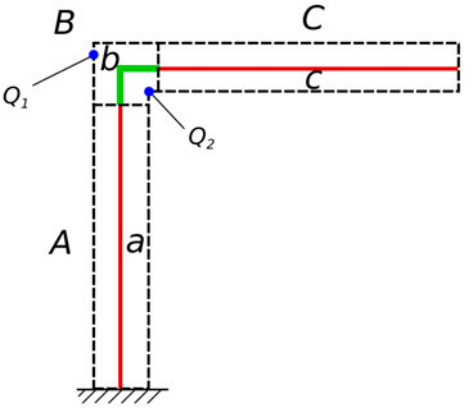


Figure 18. Two beams connected by a rigid joint. Surface node Q_1 cannot be projected to both beams while surface node Q_2 has projections on both beams. The beams are divided into three partitions where curve partitions are denoted by small letters a, b, c while surface partitions are denoted by capital letters A, B, C. b is modelled as rigid so all surface nodes in B share the same rigid body motion.

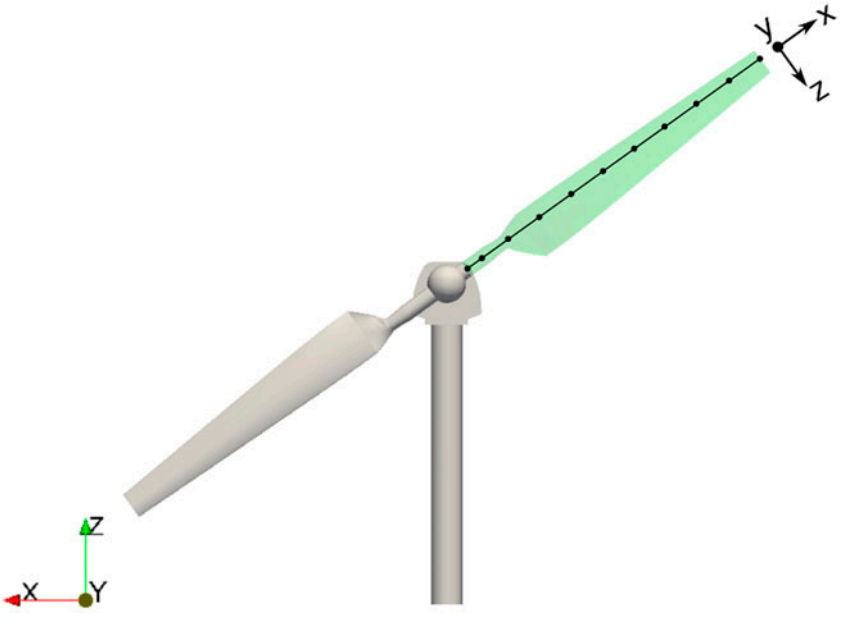


Figure 19. The rotating coordinate system defined on one blade.

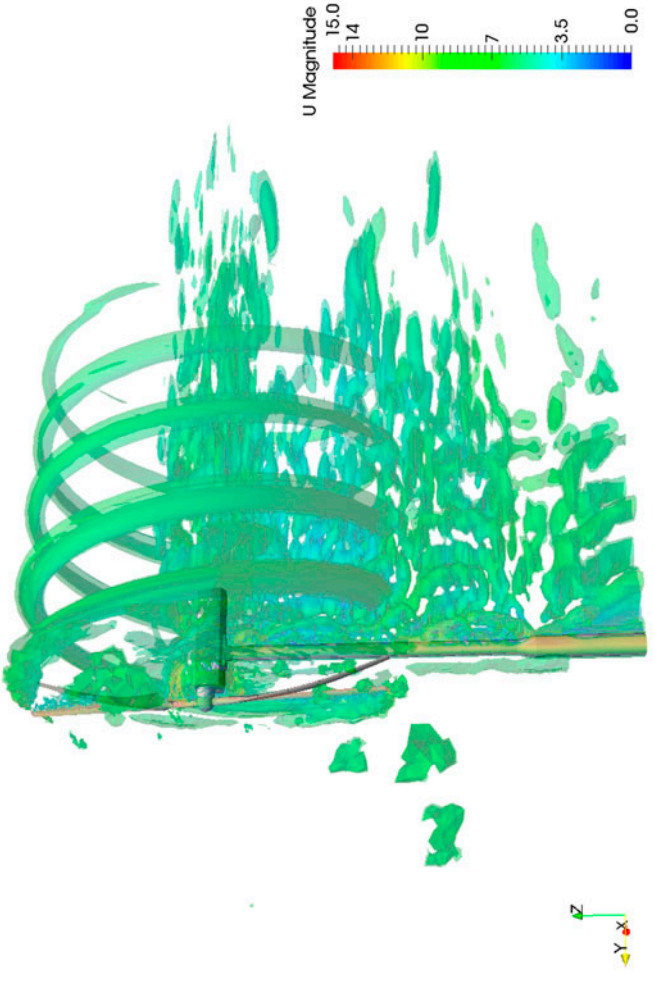


Figure 20. Q-criterion isosurface colored by velocity magnitude on the flow field at 3.15 s. The bending deformation is scaled with a factor of 70.

between a and $A \cup B$: when surface nodes cannot be projected to a beam, they will be linked together to the closest end node of the beam. Therefore, all surface nodes of partition B will take the rigid body motion at the connecting point between a and b.

One application of this treatment can be found in FSI simulations of wind turbines whose tower is also modelled as an elastic beam.

5. FSI example

In the following, the mapping for beam elements will be used in an FSI simulation of the NREL phase VI wind turbine. In Sicklinger et al. (2015), FSI simulation of the wind turbine is realised by modelling the rotor blades with shell elements. This work reuses the set-up in Sicklinger et al. (2015) for FSI and replaces the shell model by a beam model. Each blade is modelled by 10 beam elements which are defined on a rotating frame as shown in Figure 19. The distribution of structural properties along the length is obtained by linear interpolation of those documented in Hand et al. (2001) (at 26 different positions). The blades are modelled as cantilever beams that are clamped at the roots. Linear beam elements are used since the deformation of the blades under the wind load is small. Beside the wind load, additional centrifugal forces are also added to the blades.

Due to small deformation, the linearised mapping algorithm is used. The fluid surface mesh is the same as in Figure 15(b). For more details in the set-up of the FSI simulation, the reader is referred to Sicklinger et al. (2015). The results are shown in Figure 20.

It is found that there is considerable deviation between the tip displacement from the beam model and that from the shell model. This is due to the lack of accuracy of the cross section analysis tool as mentioned in Hand et al. (2001), i.e. the two structure models do not have enough close structural properties.

6. Conclusion

This work focuses on the mapping technique for beam elements in computational FSI. The linearised and the co-rotating algorithms which are based on the individual beam kinematics are reviewed. The co-rotating algorithm is convergent and more accurate thanks to the reduction in rotations. It can also give smooth deformed surfaces even under large displacements and rotations. But the linearised algorithm is limited to small deformation.

The efficient determination of the cross section centre is based on the assumption that all cross sections are parallel. This is valid for straight beams and is an approximation for slightly curved beams. The efficiency is further improved for structured meshes since surface nodes belonging to the same cross section can be grouped together. The approach is also extended to treat connected beams.

At the end, the mapping technique is applied in the FSI simulation of a wind turbine where the structure is modelled as a cantilever beam in a rotating frame.

Acknowledgements

The authors would like to acknowledge the contributions from the German Federal Ministry for the Environment, Nature Conservation, Building and Nuclear Safety through the project 'KonTest-Erstellung einer Konzeption eines Windenergie Testgeländes in bergig komplexem Terrain'.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by Federal Ministry for the Environment, Nature Conservation, Building and Nuclear Safety [0325656B].

References

Ahrem, R., Beckert, A., & Wendland, H. (2007). Recovering rotations in aeroelasticity. *Journal of Fluids and Structures, 23*, 874–884.

Bathe, K.-J., & Bolourchi, S. (1979). Large displacement analysis of three-dimensional beam structures. *International Journal for Numerical Methods in Engineering, 14*, 961–986.

Bazilevs, Y., Takizawa, K., & Tezduyar, T. E. (2013a). Challenges and directions in computational fluid–structure interaction. *Mathematical Models and Methods in Applied Sciences, 23*, 215–221.

Bazilevs, Y., Takizawa, K., & Tezduyar, T. E. (2013b). *Computational fluid–structure interaction: Methods and applications*. Wiley.

Beckert, A., & Wendland, H. (2001). Multivariate interpolation for fluid–structure–interaction problems using radial basis functions. *Aerospace Science and Technology, 5*, 125–134.

Belver, A. V., Foces Mediavilla, A., Lorenzana Iban, A., & Rossi, R. (2010). Fluid–structure coupling analysis and simulation of a slender composite beam. *Science and Engineering of Composite Materials, 17*, 47–77.

Bernardi, C., Maday, Y., & Patera, A. T. (1994). A new nonconforming approach to domain decomposition: The mortar element method. *Nonlinear Partial Differential Equations and Their Applications*.

Blasques, J. P. A. A., Bitsche, R., Fedorov, V., & Eder, M. A. (2013). *Applications of the beam cross section analysis software (becas)*. In Proceedings of the 26th Nordic Seminar on Computational Mechanics, (pp. 46–49).

Blom, F. J. (1998). A monolithical fluid–structure interaction algorithm applied to the piston problem. *Computer Methods in Applied Mechanics and Engineering, 167*, 369–391.

Bottasso, C., Campagnolo, F., Croce, A., & Tibaldi, C. (2013). Optimization-based study of bend-twist coupled rotor blades for passive and integrated passive/active load alleviation. *Wind Energy, 16*, 1149–1166.

Chen, H., Yu, W., & Capellaro, M. (2010). A critical assessment of computer tools for calculating composite wind turbine blade properties. *Wind Energy, 13*, 497–516.

Cordero-Gracia, M., Gómez, M., & Valero, E. (2014). A radial basis function algorithm for simplified fluid–structure data transfer. *International Journal for Numerical Methods in Engineering*, *99*, 888–905.

de Boer, A., van Zuijlen, A. H., & Bijl, H. (2008). Comparison of conservative and consistent approaches for the coupling of non-matching meshes. *Computer Methods in Applied Mechanics and Engineering*, *197*, 4284–4297.

De Hart, J., Peters, G., Schreurs, P., & Baaijens, F. (2003). A three-dimensional computational analysis of fluid–structure interaction in the aortic valve. *Journal of Biomechanics, 36*, 103–112.

De Rosis, A., Falcucci, G., Ubertini, S., & Ubertini, F. (2013). A coupled lattice boltzmann-finite element approach for two-dimensional fluid–structure interaction. *Computers & Fluids, 86*, 558–568.

Degroote, J., Bruggeman, P., Haelterman, R., & Vierendeels, J. (2008). Stability of a coupling technique for partitioned solvers in FSI applications. *Computers & Structures, 86*, 2224–2234.

Donea, J., Huerta, A., Ponthot, J.-P., & Rodriguez-Ferran, A. (2004). *Encyclopedia of computational mechanics, Vol. 1: Fundamentals, Chapter 14: Arbitrary Lagrangian–Eulerian methods*.

Farhat, C., Van der Zee, K. G., & Geuzaine, P. (2006). Provably second-order time-accurate loosely-coupled solution algorithms for transient nonlinear computational aeroelasticity. *Computer Methods in Applied Mechanics and Engineering*, *195*, 1973–2001.

Felippa, C., & Haugen, B. (2005). A unified formulation of small-strain corotational finite elements: I. Theory. *Computer Methods in Applied Mechanics and Engineering*, *194*, 2285–2335.

Felippa, C., Park, K., & Ross, M. (2010). *A classification of interface treatments for FSI, in: Fluid Structure Interaction II*. Springer. (pp. 27–51).

Fernández, M. Á., & Moubachir, M. (2005). A newton method using exact jacobians for solving fluid–structure coupling. *Computers & Structures, 83*, 127–142.

Gerstenberger, A., & Wall, W. A. (2008). Enhancement of fixed-grid methods towards complex fluid–structure interaction applications. *International Journal for Numerical Methods in Fluids, 57*, 1227–1248.

Hand, M., Simms, D., Fingersh, L., Jager, D., Cotrell, J., Schreck, S., & Larwood, S. (2001). Unsteady aerodynamics experiment phase VI: Wind tunnel test configurations and available data campaigns. *NREL/TP-500-29955*.

Hansen, M. O. L., Sørensen, J. N., Voutsinas, S., Sørensen, N., & Madsen, H. A. (2006). State of the art in wind turbine aerodynamics and aeroelasticity. *Progress in Aerospace Sciences, 42*, 285–330.

Hirt, C., Amsden, A. A., & Cook, J. (1974). An arbitrary Lagrangian–Eulerian computing method for all flow speeds. *Journal of Computational Physics, 14*, 227–253.

Holmes, J. D. (2015). *Wind loading of structures*. CRC Press.

Hou, G., Wang, J., & Layton, A. (2012). Numerical methods for fluid–structure interaction – A review. *Communications in Computational Physics, 12*, 337–377.

Hsu, M.-C., & Bazilevs, Y. (2012). Fluid–structure interaction modeling of wind turbines: Simulating the full machine. *Computational Mechanics, 50*, 821–833.

Jaiman, R., Jiao, X., Geubelle, P., & Loth, E. (2006). Conservative load transfer along curved fluid-solid interface with non-matching meshes. *Journal of Computational Physics, 218*, 372–397.

Kindmann, R., & Kraus, M. (2004). *Steel structures: Design using FEM*. Ernst & Sohn.

Klöppel, T., Popp, A., Küttler, U., & Wall, W. A. (2011). Fluid-structure interaction for non-conforming interfaces based on a dual mortar formulation. *Computer Methods in Applied Mechanics and Engineering, 200*, 3111–3126.

Krenk, S. (2009). *Non-linear modeling and analysis of solids and structures*. Cambridge University Press.

Küttler, U., & Wall, W. A. (2009). Vector extrapolation for strong coupling fluid–structure interaction solvers. *Journal of Applied Mechanics, 76*, 21–205.

Li, Z., & Vu-Quoc, L. (2010). A mixed co-rotational 3D beam element formulation for arbitrarily large rotations. *Advanced Steel Construction, 6*, 767–787.

Malcolm, D. J., & Laird, D. L. (2007). Extraction of equivalent beam properties from blade models. *Wind Energy, 10*, 135–157.

Michler, C., Van Brummelen, E., & De Borst, R. (2005). An interface Newton–Krylov solver for fluid-structure interaction. *International Journal for Numerical Methods in Fluids, 47*, 1189–1195.

Mittal, S., & Tezduyar, T. E. (1995). Parallel finite element simulation of 3D incompressible flows: Fluid-structure interactions. *International Journal for Numerical Methods in Fluids, 21*, 933–953.

Onate, E., Idelsohn, S. R., Celigueta, M. A., & Rossi, R. (2008). Advances in the particle finite element method for the analysis of fluid-multibody interaction and bed erosion in free surface flows. *Computer Methods in Applied Mechanics and Engineering, 197*, 1777–1800.

Patil, M. J., Hodges, D. H., & Cesnik, C. E. S. (2001). Nonlinear aeroelasticity and flight dynamics of high-altitude long-endurance aircraft. *Journal of Aircraft, 38*, 88–94.

Peskin, C. S. (2002). The immersed boundary method. *Acta Numerica, 11*, 479–517.

Piperno, S., & Farhat, C. (2001). Partitioned procedures for the transient solution of coupled aeroelastic problems-Part II: Energy transfer analysis and three-dimensional applications. *Computer Methods in Applied Mechanics and Engineering, 190*, 3147–3170.

Press, W., Teukolsky, S., Vetterline, W., & Flannery, B. P. (2007). *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge University Press.

Puso, M. A. (2004). A 3D mortar method for solid mechanics. *International Journal for Numerical Methods in Engineering, 59*, 315–336.

Puso, M. A., & Laursen, T. A. (2004). A mortar segment-to-segment frictional contact method for large deformations. *Computer Methods in Applied Mechanics and Engineering, 193*, 4891–4913.

Rabczuk, T., Gracie, R., Song, J.-H., & Belytschko, T. (2010). Immersed particle method for fluid–structure interaction. *International Journal for Numerical Methods in Engineering, 22*, 48.

Ryzhakov, P., Rossi, R., Idelsohn, S. R., & Oñate, E. (2010). A monolithic lagrangian approach for fluid–structure interaction problems. *Computational Mechanics, 46*, 883–899.

Sicklinger, S. (2014). *Stabilized co-simulation of coupled problems including fields and signals* (Ph.D. thesis). Technische Universität München.

Sicklinger, S., Lerch, C., Wüchner, R., & Bletzinger, K.-U. (2015). Fully coupled co-simulation of a wind turbine emergency brake maneuver. *Journal of Wind Engineering and Industrial Aerodynamics, 144*, 134–145.

Simiu, E., & Scanlan, R. H. (1996). *Wind effects on structures*. Wiley.

Smith, M. J., Cesnik, C. E., & Hodges, D. H. (2000). Evaluation of some data transfer algorithms for noncontiguous meshes. *Journal of Aerospace Engineering, 13*, 52–58.

Streiner, S. (2011). *Beitrag zur numerischen Simulation der Aerodynamik und Aeroelastik großer Windkraftanlagen mit horizontaler Achse*. Verlag Dr. Hut.

Tezduyar, T., Behr, M., & Liou, J. (1992a). A new strategy for finite element computations involving moving boundaries and interfaces-the deforming-spatial-domain/space-time

procedure: I. The concept and the preliminary numerical tests. *Computer Methods in Applied Mechanics and Engineering, 94*, 339–351.
Tezduyar, T., Behr, M., Mittal, S., & Liou, J. (1992b). A new strategy for finite element computations involving moving boundaries and interfaces-the deforming-spatial-domain/space-time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders. *Computer Methods in Applied Mechanics and Engineering, 94*, 353–371.
Unger, R., Haupt, M. C., & Horst, P. (2007). Application of lagrange multipliers for coupled problems in fluid and structural interactions. *Computers & structures, 85*, 796–809.
Varello, A., Demasi, L., Carrera, E., & Giunta, G. (2010). *An improved beam formulation for aeroelastic applications*. In Proceedings of the 51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, (pp. 12–15).
Wang, H., Chessa, J., Liu, W. K., & Belytschko, T. (2008). The immersed/fictitious element method for fluid–structure interaction: Volumetric consistency, compressibility and thin members. *International Journal for Numerical Methods in Engineering, 74*, 32–55.
Wang, T., Wüchner, R., Sicklinger, S., & Bletzinger, K.-U. (2016). Assessment and improvement of mapping algorithms for non-matching meshes and geometries in computational FSI. *Computational Mechanics, 57*, 793–816.
Wohlmuth, B. I. (2000). A mortar finite element method using dual spaces for the lagrange multiplier. *SIAM Journal on Numerical Analysis, 38*, 989–1012.
Wunderlich, W., & Kiener, G. (2004). *Statik der Stabtragwerke*. Vieweg+Teubner Verlag.
Zhu, L., & Peskin, C. S. (2002). Simulation of a flapping flexible filament in a flowing soap film by the immersed boundary method. *Journal of Computational Physics, 179*, 452–468.

Appendix 1. Interpolation of deformation and conservative computation of load

In Section 2.1, displacements and rotations on an arbitrary point P are denoted by $(v_x, v_y, v_z)^T$ and $(\theta_x, \theta_y, \theta_z)^T$. The interpolation of them on a linear beam element (see Figure 1) is shown in the following.

v_x and θ_x can be interpolated with linear shape functions:

$$v_x = \left(N_1^l \; N_2^l\right)\left(v_{x1} \; v_{x2}\right)^T, \tag{A1a}$$

$$\theta_x = \left(N_1^l \; N_2^l\right)\left(\theta_{x1} \; \theta_{x2}\right)^T, \tag{A1b}$$

where

$$N_1^l = \frac{1}{2}(1-\xi), \quad N_2^l = \frac{1}{2}(1+\xi). \tag{A2}$$

While v_y, θ_z, v_z and θ_y can be interpolated with Hermitian cubic shape functions:

$$v_y = \left(N_1^v \; N_1^\theta \; N_2^v \; N_2^\theta\right)\left(v_{y1} \; \theta_{z1} \; v_{y2} \; \theta_{z2}\right)^T, \tag{A3a}$$

$$\theta_z = v_y' = \left(N_1^{v\prime} \; N_1^{\theta\prime} \; N_2^{v\prime} \; N_2^{\theta\prime}\right)\left(v_{y1} \; \theta_{z1} \; v_{y2} \; \theta_{z2}\right)^T, \tag{A3b}$$

$$v_z = \left(N_1^v \; N_1^\theta \; N_2^v \; N_2^\theta\right)\left(v_{z1} \; -\theta_{y1} \; v_{z2} \; -\theta_{y2}\right)^T, \tag{A3c}$$

$$\theta_y = -v_z' = \left(N_1^{v\prime} \; N_1^{\theta\prime} \; N_2^{v\prime} \; N_2^{\theta\prime}\right)\left(-v_{z1} \; \theta_{y1} \; -v_{z2} \; \theta_{y2}\right)^T, \tag{A3d}$$

where

$$N_1^v = \frac{1}{4}(1-\xi)^2(2+\xi), \qquad N_1^\theta = \frac{1}{8}l(1-\xi)^2(1+\xi),$$
$$N_2^v = \frac{1}{4}(1+\xi)^2(2-\xi), \qquad N_2^\theta = -\frac{1}{8}l(1+\xi)^2(1-\xi), \tag{A4a}$$
$$N_1^{v\prime} = -\frac{3}{2l}(1-\xi)(1+\xi), \qquad N_1^{\theta\prime} = -\frac{1}{4}(1-\xi)(1+3\xi),$$
$$N_2^{v\prime} = \frac{3}{2l}(1+\xi)(1-\xi), \qquad N_2^{\theta\prime} = -\frac{1}{4}(1+\xi)(1-3\xi). \tag{A4b}$$

The shape functions can be reused in conservative computation of the loads on P. For linear 1D elements, $\mathbf{f}_P$ and $\mathbf{m}_P$ defined in Section 2.3 are firstly transformed to the ECS, resulting in $(f_x^P, f_y^P, f_z^P)^T$ and $(m_x^P, m_y^P, m_z^P)^T$. With (A1) and (A3) conservative mapping gives

$$f_x^P v_x = \left(f_x^P N_1^l \; f_x^P N_2^l\right)\left(v_{x1} \; v_{x2}\right)^T, \tag{A5a}$$
$$m_x^P \theta_x = \left(m_x^P N_1^l \; m_x^P N_2^l\right)\left(\theta_{x1} \; \theta_{x2}\right)^T, \tag{A5b}$$
$$f_y^P v_y = \left(f_y^P N_1^v \; f_y^P N_1^\theta \; f_y^P N_2^v \; f_y^P N_2^\theta\right) \left(v_{y1} \; \theta_{z1} \; v_{y2} \; \theta_{z2}\right)^T, \tag{A5c}$$
$$m_z^P \theta_z = \left(m_z^P N_1^{v\prime} \; m_z^P N_1^{\theta\prime} \; m_z^P N_2^{v\prime} \; m_z^P N_2^{\theta\prime}\right) \left(v_{y1} \; \theta_{z1} \; v_{y2} \; \theta_{z2}\right)^T, \tag{A5d}$$
$$f_z^P v_z = \left(f_z^P N_1^v \; f_z^P N_1^\theta \; f_z^P N_2^v \; f_z^P N_2^\theta\right) \left(v_{z1} \; -\theta_{y1} \; v_{z2} \; -\theta_{y2}\right)^T, \tag{A5e}$$
$$m_y^P \theta_y = \left(m_y^P N_1^{v\prime} \; m_y^P N_1^{\theta\prime} \; m_y^P N_2^{v\prime} \; m_y^P N_2^{\theta\prime}\right) \left(-v_{z1} \; \theta_{y1} \; -v_{z2} \; \theta_{y2}\right)^T. \tag{A5f}$$