



Three-dimensional image-based modeling by combining SBFEM and transfinite element shape functions

Hauke Gravenkamp¹ · Albert A. Saputra² · Sascha Eisentrger²

Received: 22 October 2019 / Accepted: 15 July 2020 / Published online: 4 August 2020
© The Author(s) 2020

Abstract

The scaled boundary finite element method (SBFEM) has recently been employed as an efficient tool to model three-dimensional structures, in particular when the geometry is provided as a voxel-based image. To this end, an octree decomposition of the computational domain is deployed, and each cubic cell is treated as an SBFEM subdomain. The surfaces of each subdomain are discretized in the finite element sense. We improve on this idea by combining the semi-analytical concept of the SBFEM with a particular class of *transition elements* on the subdomains' surfaces. Thus, a triangulation of these surfaces as executed in previous works is avoided, and consequently, the number of surface elements and degrees of freedom is reduced. In addition, these discretizations allow coupling elements of arbitrary order such that local p -refinement can be achieved straightforwardly.

Keywords Scaled boundary finite element method · Octree meshes · Transition elements · Transfinite mapping · Local mesh refinement

1 Introduction

The scaled boundary finite element method (SBFEM) is a semi-analytical technique—loosely based on finite elements—that involves only a boundary discretization of the computational (sub-)domains. Roughly speaking, this method aims at transforming a partial differential equation (PDE) in two or three spatial coordinates into a set of ordinary differential equations (ODE) in one coordinate by discretizing all but this one coordinate. In order to apply this idea effectively, a particular coordinate system is generally chosen in which one coordinate ξ points from the origin¹ to the boundary while the remaining one or two coordinate(s) (η , ζ) describe

a parametrization of the boundary. The ‘radial’ coordinate ξ is typically set to unity everywhere on the boundary.

The SBFEM was originally developed to model large and unbounded domains in the context of soil-structure interaction and was inspired by concepts such as similarity [64], cloning [65], as well as the thin layer method [30,32]. There, it was assumed that the entire computational domain was enclosed by a simply connected boundary which was discretized by finite elements. The analytical solution of the resulting ODE can then be applied to describe either a bounded ($\xi \leq 1$) or an unbounded domain ($\xi \geq 1$). A detailed description of the underlying formulation can be found in the early papers [57,58,66], as well as in the recent monograph by Song [56]. Later, it has been noticed that the SBFEM can be employed as a means to construct arbitrary star-convex² elements [8,45–47]. Each element is treated like an SBFEM domain, hence only its outer surface needs to be discretized by finite elements. Stiffness and mass matrices are computed based on the semi-analytical solution and coupled like in the finite element method. Since these subdo-

¹ In the context of the SBFEM, the origin of the coordinate system is usually positioned inside the domain and referred to as *scaling center*.

✉ Hauke Gravenkamp
hauke.gravenkamp@uni-due.de

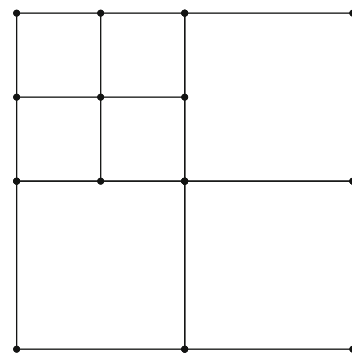
¹ Department of Civil Engineering, University of Duisburg-Essen, Universittsstrae 15, 45141 Essen, Germany

² School of Civil and Environmental Engineering, The University of New South Wales, Sydney, NSW 2052, Australia

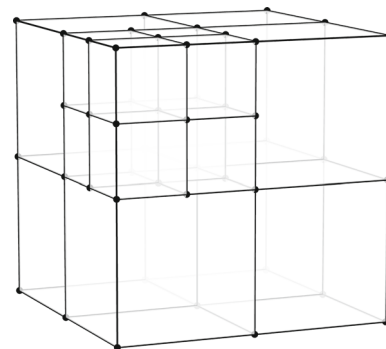
² A domain Ω is called star-convex if there exists a point $\mathbf{r}_0 \in \Omega$ such that the line segment from \mathbf{r}_0 to any point in Ω is contained in Ω . In other words, for the domain to be star-convex, there must be a point from where the whole boundary is ‘visible.’ Any convex domain is also star-convex.

mains require discretization of the surface only, they facilitate the coupling of different meshes and types of interpolants. Furthermore, such polygonal/polyhedral elements enable the application of rather flexible meshing procedures, compared to conventional finite elements, which are usually restricted to triangular/quadrilateral and tetrahedral/hexahedral shapes. A particular variant of such a meshing paradigm consists in the use of domain decompositions of the quadtree/octree type. The most common of this class of decompositions consists of square/cube-shaped cells (see Fig. 1a for a minimal example). While, from a geometrical viewpoint, these meshes only require quadrilateral/hexahedral subdomains, the SBFEM concept allows each side to be divided into an arbitrary number of surface elements on the boundary. Hence, coupling subdomains of different sizes is relatively simple. This idea has been used for the meshing of complex geometries, where flexible local refinement is desirable [24,27,43,44]. Such a meshing paradigm is particularly useful for *image-based analyses*, i.e., in applications where the geometry (the distribution of material parameters, etc.) is provided in a pixel graphics format. A number of approaches have been developed to mesh images, most of which can be divided into two categories (see [67] for a comprehensive literature review): The first type includes meshing based on boundary detection for each region in the image, where a region is assumed to be homogeneous. This can be straightforwardly achieved using techniques such as a marching cubes algorithm [40]. Subsequently, these boundaries are utilized to mesh the region inside using conventional meshing algorithms, such as the advancing front [39] or Delaunay tessellation [9]. The second type involves techniques that mesh the image directly. These techniques are generally quicker due to the straightforward nature of the meshing process. One very basic example is the pixel-based approach [33], where each pixel is modeled as a quadrilateral/hexahedral finite element. While such a meshing approach is absolutely trivial and fast, it leads to an unnecessarily large number of degrees of freedom. In order to improve the efficiency of the meshing, other direct meshing techniques can be deployed, such as the aforementioned quadtree/octree meshing structure [67].

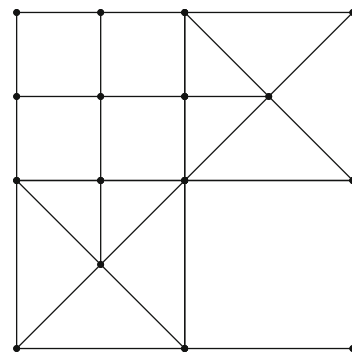
Quadtree meshing is performed by recursively dividing an image matrix into four equal-sized cells at a time. A criterion of homogeneity is established based on the difference in the maximum and minimum color intensity within a cell. If this difference exceeds a user-defined threshold, the cell is divided. The maximum and minimum cell size allowed are also specified by the user. As a result, the meshing scheme adaptively refines the regions around different interfaces. Simultaneously, the scheme retains relatively larger cells within each region where the material can be assumed to be homogeneous. However, since cells of different sizes exist in quadtree meshes, compatibility issues are encoun-



(a) Quadtree mesh, SBFEM.



(b) Octree mesh, SBFEM.



(c) Quadtree mesh, FEM.

Fig. 1 Discretization in the SBFEM: **a** In 2D, a quadtree-mesh is used to discretize the computational domain straightforwardly. **b** In 3D, surfaces need to be discretized, which feature a structure similar to the quadtree meshes in 2D. **c** In previous work, these surfaces have been triangulated using standard finite elements. Using the transition elements discussed in this paper, the surface meshes can be handled directly without further subdivision (see part (b) of the figure)

tered when using conventional finite elements. This issue was circumvented by utilizing the scaled boundary finite element method (SBFEM) for quadtree meshes, as this method allows the use of arbitrary (star-convex) polygonal subdomains. A detailed explanation of the image-based analysis using the SBFEM can be found in [27,53].

Notwithstanding the success of this method in two-dimensional image-based analyses, the situation in three-

dimensional modeling is somewhat different. The quadtree-based domain decomposition can be extended to 3D without much further effort [53]. In this case, we divide a cubic cell into eight smaller ones of equal size and refer to this technique as octree decomposition (Fig. 1b). From the viewpoint of the SBFEM, each cubic cell constitutes one subdomain, which implies that we have to discretize the surface of each cube in a finite element sense. Figure 1b reveals that the surfaces of 3D octree decompositions basically consist of 2D quadtree meshes and hence exhibit the same compatibility issues discussed above. That is, due to the two-dimensional finite element grid that has to be used for the discretization of the boundary, hanging nodes are generated. In this example, the mesh on the front surface of the structure in Fig. 1b is identical to the two-dimensional case illustrated in Fig. 1a. In finite element-applications, the hanging node problem can be circumvented by either using some kind of constraints (constraint equations, Lagrange multipliers, etc.) or triangular elements.

In previous works, this issue has been tackled by the second option, i.e., triangulating the surfaces of individual cubes to ensure compatibility with adjacent subdomains, see Fig. 1c. This approach has been applied to solve problems related to acoustics [37] and fracture [52]. However, this concept may be considered somewhat inelegant and unsatisfactory as the triangular elements require additional degrees of freedom and are intrinsically less accurate compared to quadrilateral elements. Furthermore, the previous approach does not allow combining elements of different orders within the same model, which is one of the major advantages exploited in 2D SBFEM models [25,27].

To improve on the previous formulation, we present a novel approach based on the use of special transition elements. Using this meshing paradigm, the surfaces of octree-based SBFEM models can be discretized without introducing additional nodes, and we are enabled to formulate shape functions on quadrilateral domains with an arbitrary number of subdivisions of each edge. The application of such elements was inspired by the papers on the so-called ‘pNh elements’ [15,20,62], as well as earlier formulations [3,6,16,18,18]. The basic idea of these formulations is that different interpolants can be chosen along the four edges of a quadrilateral element to ensure a compatible coupling between different types of elements. These four individual functions are then interpolated over the domain using certain projection operators. This concept also allows using a piecewise polynomial interpolation along any of the edges, such that elements of different sizes can be coupled. In a recent publication, we presented a detailed review and generalization of these classes of transition elements [13]. There, we refer to these general transition elements as xNy-elements to indicate that it is, in principle, possible to couple arbitrary element families. In the current paper, we shall only pro-

vide a very brief summary of the formulation (Sect. 2.4) and refer the reader to the pertinent literature. The purpose of this work is mainly to demonstrate that these transition elements can, in fact, be used to solve the hanging node problem in three-dimensional SBFEM models. The motivation for this approach is twofold:

1. We wish to avoid any triangulation of the surfaces in three-dimensional (octree-based) models. This goal can easily be achieved using said transition elements since they allow splitting each edge into an arbitrary number of sections. Consequently, we avoid introducing additional elements that are not required by the topological decomposition. Especially when using high-order interpolation, the reduction in degrees of freedom (DOFs) can be very significant.
2. We wish to allow different element orders within the same model, which is particularly useful in the case of inhomogeneous materials and wave propagation problems. Up to now, conventional Lagrange elements have been used on the surfaces of SBFEM subdomains, which required the element order to match between adjacent subdomains. Using the proposed approach generally allows connecting elements of arbitrary orders or even entirely different types of shape functions.

2 Theory

The overarching goal of this contribution is to combine the concepts of the SBFEM, octree-based mesh generation, and transition elements. This combination results in an efficient and robust numerical tool that can be applied to a wide range of problems in engineering and physics. However, we will demonstrate the essential concepts based on the example of linear elasticity. The problem statement is formulated in Sect. 2.1, after which we present a brief description of the fundamental principles underlying the SBFEM in Sect. 2.2. Essential features of octree-meshes are discussed in Sect. 2.3. Considering the formulation of transition elements, we sketch the basic ideas in Sect. 2.4, followed by an example for the case of linear interpolation in Sect. 2.5, as well as some details on the numerical integration in Sect. 2.6.

2.1 Problem statement

In the current work, we address boundary value problems related to three-dimensional linear elasticity, governed by the following system of PDEs for the displacement field $\mathbf{u} = \mathbf{u}(x, y, z)$:

$$\hat{\mathcal{L}}^T \mathbf{D} \hat{\mathcal{L}} \mathbf{u}(x, y, z) + \rho \ddot{\mathbf{u}}(x, y, z) = \mathbf{0} \quad (1)$$

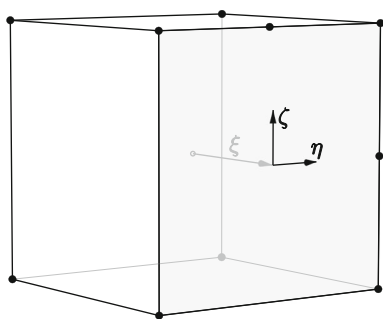


Fig. 2 Exemplary cubic cell within an octree decomposition. The ‘scaled boundary’ coordinates (ξ, η, ζ) are indicated on one of the cube’s surfaces

Here, ρ denotes the mass density and \mathbf{D} is the elasticity matrix, which, for an isotropic material with shear modulus G and Poisson’s ratio ν , reads

$$\mathbf{D} = \frac{2G}{(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & & & \\ \nu & 1-\nu & \nu & & & \\ \nu & \nu & 1-\nu & & & \\ & & & \frac{1-2\nu}{2} & & \\ & & & & \frac{1-2\nu}{2} & \\ & & & & & \frac{1-2\nu}{2} \end{bmatrix} \quad (2)$$

$\hat{\mathcal{L}}$ is the differential operator

$$\hat{\mathcal{L}} = \begin{bmatrix} \partial_x & 0 & 0 & \partial_y & \partial_z & 0 \\ 0 & \partial_y & 0 & \partial_x & 0 & \partial_z \\ 0 & 0 & \partial_z & 0 & \partial_x & \partial_y \end{bmatrix}^T \quad (3)$$

which may be re-written for later use as

$$\hat{\mathcal{L}} = \hat{\mathbf{b}}_1 \partial_x + \hat{\mathbf{b}}_2 \partial_y + \hat{\mathbf{b}}_3 \partial_z \quad (4)$$

with

$$\hat{\mathbf{b}}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}^T \quad \hat{\mathbf{b}}_2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T$$

$$\hat{\mathbf{b}}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T \quad (5)$$

By modifying the differential operator, as well as the material parameters, other common linear PDEs are obtained and can be solved by the SBFEM in a similar fashion [2,37].

2.2 SBFEM

The SBFEM constitutes a semi-analytical approach, in the sense that the governing PDEs (in three coordinates) are discretized in two directions while remaining analytical in the

third coordinate. In order to employ this idea effectively, a particular coordinate transformation is usually applied such that the two discretized directions (denoted as η, ζ) define a parametrization of the computational domain’s boundary, while the analytical coordinate ξ describes the direction from the origin to the boundary.³ Figure 2 depicts this coordinate system for the example of a cubical cell, which can be thought of as a subdomain in the octree decomposition (cf. Fig. 1). In this case, the local coordinates (η, ζ) are defined on each surface of the cube, similar to two-dimensional finite elements. A few key steps of the procedure are summarized in the remainder of this section. The complete formulation, including a detailed derivation of the basic coordinate transformation and the semi-discretization, can be found in [57]. For dynamic problems, we apply the solution scheme proposed in [55], which is based on a continued-fraction-expansion of the dynamic stiffness matrix. A concise summary of the procedures applied to dynamic problems is presented in [25].

In general, the coordinate transformation employed in the SBFEM is written as

$$\hat{x}(\xi, \eta, \zeta) = \xi x(\eta, \zeta) \quad (6a)$$

$$\hat{y}(\xi, \eta, \zeta) = \xi y(\eta, \zeta) \quad (6b)$$

$$\hat{z}(\xi, \eta, \zeta) = \xi z(\eta, \zeta) \quad (6c)$$

where $\hat{x}, \hat{y}, \hat{z}$ define a three-dimensional Cartesian coordinate system and x, y, z are the values of these Cartesian coordinates on the boundary of the domain, parametrized by the local coordinates η, ζ .⁴ Without loss of generality, the origin of the coordinate system is chosen to be inside the domain. Furthermore, the domain is assumed to be star-convex. The application presented in this paper involves only cubical cells, where the discretized surfaces are squares aligned with the Cartesian coordinates. Hence, the functions $x(\eta, \zeta), y(\eta, \zeta), z(\eta, \zeta)$ are trivial to implement based on the vertex coordinates of the cell. In other applications, the coordinates are interpolated using shape functions, especially when curved boundaries exist.

The Jacobian matrix corresponding to our coordinate transformation is given as

$$\hat{\mathbf{J}}(\xi, \eta, \zeta) = \begin{bmatrix} \hat{x}_{,\xi} & \hat{y}_{,\xi} & \hat{z}_{,\xi} \\ \hat{x}_{,\eta} & \hat{y}_{,\eta} & \hat{z}_{,\eta} \\ \hat{x}_{,\zeta} & \hat{y}_{,\zeta} & \hat{z}_{,\zeta} \end{bmatrix} = \begin{bmatrix} x & y & z \\ \xi x_{,\eta} & \xi y_{,\eta} & \xi z_{,\eta} \\ \xi x_{,\zeta} & \xi y_{,\zeta} & \xi z_{,\zeta} \end{bmatrix} \quad (7)$$

³ There are a few exceptions where such a coordinate transformation is not required, and the concept of semi-discretization can be applied directly in a Cartesian, polar, or cylindrical coordinate system to model structures of (piece-wise) constant cross-section [7,21–23,35,36,42].

⁴ This slightly peculiar choice of denoting the standard Cartesian coordinates by $\hat{\bullet}$ simply stems from the fact that the coordinates on the boundary occur far more frequently in the formulation of the SBFEM. This notation is adopted here for consistency with [53,57].

which is conveniently decomposed into

$$\hat{\mathbf{J}}(\xi, \eta, \zeta) = \begin{bmatrix} 1 & & \\ & \xi & \\ & & \xi \end{bmatrix} \begin{bmatrix} x & y & z \\ x, \eta & y, \eta & z, \eta \\ x, \xi & y, \xi & z, \xi \end{bmatrix} = \begin{bmatrix} 1 & & \\ & \xi & \\ & & \xi \end{bmatrix} \mathbf{J}(\eta, \zeta) \tag{8}$$

With these transformations, we re-formulate the differential operator $\hat{\mathcal{L}}$ in terms of the local coordinates

$$\mathcal{L} = \mathbf{b}_1 \partial_\xi + \frac{1}{\xi} (\mathbf{b}_2 \partial_\eta + \mathbf{b}_3 \partial_\zeta) \tag{9}$$

The matrices \mathbf{b}_1 , \mathbf{b}_2 , and \mathbf{b}_3 are given in Appendix A. In these ‘scaled boundary coordinates’, we now interpolate the displacement field in the η - and ζ -directions:

$$\mathbf{u}(\xi, \eta, \zeta) = \mathbf{N}(\eta, \zeta) \mathbf{u}_n(\xi) \tag{10}$$

Here, $\mathbf{N}(\eta, \zeta)$ consists of two-dimensional finite element shape functions—for which we will choose transfinite shape functions as explained in Sect. 2.4. Note that the displacement field remains analytical (i.e., not interpolated) in the ξ -direction.

Applying the method of weighted residuals in the next step, we multiply the governing PDEs (1)—with the transformed linear differential operator introduced in Eq. (9)—by a set of test functions and integrate over the computational domain. Our approach is based on the Bubnov-Galerkin approach; i.e., we use the same trial and test functions. This procedure is similar to a conventional finite element formulation; the difference being that the trial and test functions are functions of two coordinates, namely η and ζ , only. We obtain a matrix differential equation that represents a weak form of the governing equation in η and ζ while remaining ‘strong’ in ξ :

$$\mathbf{E}_0 \xi^2 \mathbf{u}_n(\xi)_{, \xi \xi} + (2 \mathbf{E}_0 - \mathbf{E}_1 + \mathbf{E}_1^T) \xi \mathbf{u}_n(\xi)_{, \xi} + (\mathbf{E}_1^T - \mathbf{E}_2) \mathbf{u}_n(\xi) + \omega^2 \xi^2 \mathbf{M}_0 \mathbf{u}_n(\xi) = \mathbf{0} \tag{11}$$

Here, the coefficient matrices \mathbf{E}_0 , \mathbf{E}_1 , \mathbf{E}_2 , \mathbf{M}_0 are similar to stiffness and mass matrices in the finite element method and involve an integration over the discretized surfaces, i.e.,

$$\mathbf{E}_0 = \int_S \mathbf{B}_1^T \mathbf{D} \mathbf{B}_1 |J| d\eta d\zeta \tag{12a}$$

$$\mathbf{E}_1 = \int_S \mathbf{B}_2^T \mathbf{D} \mathbf{B}_1 |J| d\eta d\zeta \tag{12b}$$

$$\mathbf{E}_2 = \int_S \mathbf{B}_2^T \mathbf{D} \mathbf{B}_2 |J| d\eta d\zeta \tag{12c}$$

$$\mathbf{M}_0 = \int_S \mathbf{N}^T \rho \mathbf{N} |J| d\eta d\zeta \tag{12d}$$

with

$$\mathbf{B}_1 = \mathbf{b}_1 \mathbf{N}(\eta, \zeta) \tag{13a}$$

$$\mathbf{B}_2 = \mathbf{b}_2 \mathbf{N}(\eta, \zeta)_{, \eta} + \mathbf{b}_3 \mathbf{N}(\eta, \zeta)_{, \zeta} \tag{13b}$$

Through this process of semi-discretization, the original boundary value problem is reduced to finding solutions to the ODE (11).⁵ In the static case ($\omega = 0$), the differential equation permits solutions of the form $\Psi \xi^{\lambda c}$ with a constant vector Ψ and exponent λ , which are the solutions to a quadratic eigenvalue problem. The integration constants c can be obtained by evaluating the boundary conditions. Details are provided in [54]. To obtain a somewhat more consistent formulation for statics and dynamics, we summarize here an apparently different (though equivalent) solution procedure. For this purpose, we may re-write the ODE (11) as a differential equation for the dynamic stiffness matrix. To achieve this, we first substitute the displacement field interpolation (10) into Eq. (9) to compute the components of the strain vector (in Voigt notation)

$$\boldsymbol{\varepsilon} = \mathbf{B}_1 \mathbf{u}_n(\xi)_{, \xi} + \frac{1}{\xi} \mathbf{B}_2 \mathbf{u}_n(\xi) \tag{14}$$

Integrating the corresponding components of the stress vector along η and ζ yields an expression for the nodal forces

$$\mathbf{R}(\xi) = \xi^2 \mathbf{E}_0 \mathbf{u}_n(\xi)_{, \xi} + \xi \mathbf{E}_1^T \mathbf{u}_n(\xi) \tag{15}$$

Noting that the (dynamic) stiffness matrix $\mathbf{S}(\omega, \xi)$ is defined as the relationship between displacements and forces, i.e.,

$$\mathbf{R}(\xi) = \mathbf{S}(\omega, \xi) \mathbf{u}(\xi) \tag{16}$$

we use Eqs. (11), (15), (16) to obtain the differential equation in dynamic stiffness on the boundary ($\xi = 1$)

$$(\mathbf{S}(\omega) - \mathbf{E}_1) \mathbf{E}_0^{-1} (\mathbf{S}(\omega) - \mathbf{E}_1^T) - \mathbf{E}_2 + \mathbf{S}(\omega) + \omega \mathbf{S}(\omega)_{, \omega} + \omega^2 \mathbf{M}_0 = \mathbf{0} \tag{17}$$

Considering again the static case, the above equation simplifies to

$$(\mathbf{K} - \mathbf{E}_1) \mathbf{E}_0^{-1} (\mathbf{K} - \mathbf{E}_1^T) - \mathbf{E}_2 + \mathbf{K} = \mathbf{0} \tag{18}$$

where we introduce the static stiffness matrix $\mathbf{K} = \mathbf{S}(0)$. Equation (18) is an algebraic Riccati equation that can be solved using standard procedures [57]. In the dynamic case,

⁵ Many details on the properties and the available solution procedures of similar differential equations are discussed in [31].

we may—similar to the approach taken for finite elements—assume an approximation of the form

$$\mathbf{S}(\omega) \approx \mathbf{K} - \omega^2 \mathbf{M} \quad (19)$$

where \mathbf{M} denotes the mass matrix of the SBFEM domain. Substituting Eq. (19) into Eq. (17) yields a Lyapunov equation for the mass matrix

$$(\mathbf{K} - \mathbf{E}_1) \mathbf{E}_0^{-1} \mathbf{M} + \mathbf{M} \mathbf{E}_0^{-1} (\mathbf{K} - \mathbf{E}_1^T) + 3 \mathbf{M} - \mathbf{M}_0 = \mathbf{0} \quad (20)$$

for which again standard solution procedures exist. The dynamic stiffness matrix in the form of Eq. (19) introduces an additional approximation in the sense that the semi-discretized differential equation (17) is not satisfied exactly. To improve accuracy, we apply the procedure first proposed in [55], which we briefly summarize in Appendix B for easier reference. Once the stiffness and mass matrices of the individual subdomains are computed, they can be assembled just like finite elements by enforcing continuity of displacements. Hence, the SBFEM subdomains can be treated as large elements without interior nodes and with an arbitrary number of surfaces. A global system (subscript ‘g’) is consequently assembled as

$$\mathbf{K}_g \mathbf{u}_g + \mathbf{M}_g \ddot{\mathbf{u}}_g = \mathbf{F}_g \quad (21)$$

The external load vector \mathbf{F} is obtained by integrating over the subdomains’ surfaces. Since those surfaces are discretized essentially by finite elements, the shape functions are integrated in the same way as in other finite-element-based techniques to obtain the load vector.

2.3 Octree mesh

While the concept of the SBFEM can generally be applied to any star-convex subdomain, the use of the quadtree/octree decomposition has become particularly popular. As already discussed in the introduction, octree-meshes can be obtained easily by starting with a coarse discretization consisting of a few cubes and recursively dividing each cube into eight smaller cubes according to specific criteria. These criteria may be based on whether the cube is intersected by a boundary or whether the variation of material parameters within the cube exceeds a predefined threshold. For further details, we refer again to [53]. Also, the reader may note that such a meshing procedure will not lead to a smooth boundary representation of complex geometries due to the ‘stair-case’ approximation of interfaces. These geometry errors are acceptable in many applications—particularly in image-based analyses, where the available initial geometry description is a pixel-based image. In other cases, it is desirable to improve the description of the boundaries by applying

smoothing algorithms, cutting the cubes that are intersected by the boundary [38], or combining the octree decomposition with fictitious domain concepts [24]. Such approaches are the subject of other publications and will not be discussed here in detail. In the current contribution, we restrict ourselves to pure octree-meshes since this is the part where our approach differs from previous works.

The proposed application of transition elements changes the way the meshes on the surfaces of the SBFEM subdomains are handled. For now, let us assume that the octree decomposition is ‘balanced,’ which means that adjacent subdomains differ by at most one refinement level. Consequently, each edge of each cube is either split into two segments or not. Taking into account the symmetries under rotation, there exist only six distinct meshing patterns on the cube surfaces. These patterns are depicted in Fig. 3 and differ in the number of sides that are split in order to ensure conformal coupling to the adjacent subdomain. Note that the first and last pattern in each row all consist of standard quadrilateral elements. Thus, in the proposed approach, there are only five different types of surface elements to be considered.⁶ As will be shown in the following sections, we can construct shape functions with nodes at the desired positions. For comparison, Fig. 3 also displays the subdivision of the patterns using conventional triangular and quadrilateral elements, as has been done in previous works.

The difference in both approaches becomes prominent when applying high-order interpolations. As an example, Fig. 4 shows the same patterns with additional nodes using a polynomial order of 3. Note that the transition elements do not require internal nodes to achieve complete polynomials up to an order of 3. This is in contrast to the Lagrange elements employed previously. In that sense, the shape functions are similar to that of serendipity finite elements. Furthermore, it should also be noted that, when using transition elements, there is no need to restrict the decomposition to balanced meshes. In fact, subdomains of arbitrary sizes can be coupled straightforwardly. This would be rather tedious when using conventional elements since adequate triangulations would have to be defined for each of the (arbitrarily large number of) different patterns.

⁶ To be more precise, there are two different cases in which all of the four edges of a given element are split into two segments: If smaller elements are to be connected only to the edges, these edges need to be segmented, while an additional node on the surface is not actually required. If, however, the surface itself is connected to smaller cubes, we need to split this surface element into four. In our current implementation, these two cases are treated the same way by splitting the surface into four elements.

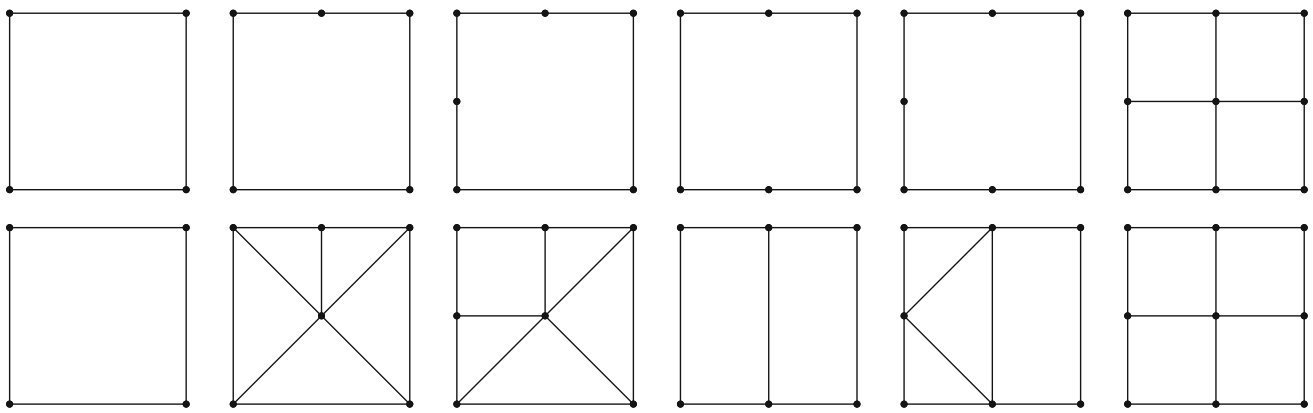


Fig. 3 Mesh patterns on the surfaces of a balanced octree decomposition. Transition elements allow us to use quadrilateral elements in all cases (top), while conventional elements require a subdivision of the surfaces to treat hanging nodes (bottom).

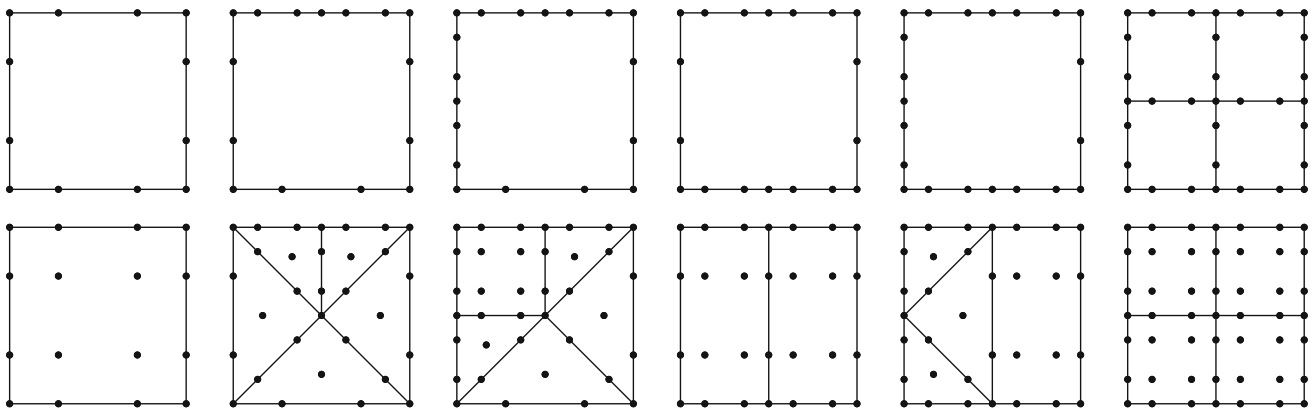


Fig. 4 Mesh patterns for the case of third order interpolation: Transition elements (top) and Lagrange elements (bottom)

2.4 Transition elements based on the xNy element concept

As discussed in the previous sections, transition elements are being used in the context of finite element models to realize local mesh refinement. In the current work, we will, for the first time, use the transition elements to discretize each face of a hexahedral SBFEM. This enables us to connect subdomains in an octree decomposition. In principle, such transition elements can be derived for various types of interpolants, including Legendre polynomials or Fourier shape functions, which have been previously used in the context of SBFEM applications [26]. In this paper, our focus is, however, on high-order shape functions based on Lagrange polynomials that are defined on a non-equidistant grid of points [29,48]. Typically, a Gauss-Lobatto-Legendre nodal distribution is chosen to generate the high-order Lagrange shape functions. This particular set of nodes includes the element (interval) boundaries at ± 1 , while the interior nodes are the roots of the Lobatto polynomials of order $p - 1$ (when constructing shape functions of order p) [10].

2.4.1 Projection: Fundamental idea

The idea of transition elements, as implemented in this paper, can be traced back to the works of Gordon and co-workers [16–19]. In their articles, the transition element was referred to as *transfinite element*, due to the special type of mapping that was applied. In the remainder of the article, we still prefer to use the term *transition element* instead. It is worth mentioning that such an approach to deriving shape functions for transition elements can also be used to achieve an accurate approximation of the geometry in high-order FEMs. In the p -FEM, this type of geometry description is commonly referred to as *blending function method* [5,34,59] and is widely used to incorporate the exact geometry of the structure stemming, for example, from computer-aided-design (CAD) software. More recent applications of the transition element technology can be found in [49–51,61,62]

In the current contribution, the main goal is to couple SBFEM-subdomains of different sizes, where each boundary of each element is discretized using two-dimensional quadrilateral elements (again, refer to Fig. 1b). In other words, we would like to have a quadrilateral element that allows dif-

ferent numbers of nodes along all four edges (e.g., on the surfaces of the cube indicated in Fig. 2). Therefore, the task is to develop a transition element that features piece-wise polynomial shape functions such that it can couple to multiple smaller elements, each with an arbitrary order of interpolation. At this point, only the basic idea is sketched and we refer to the pertinent literature [16–19,49–51,61,62] and the references cited therein. In particular, the authors recently presented a comprehensive analysis and generalization of this class of elements in [13]. The interested reader may refer to this publication for further details on the concepts outlined in the current work.

From a more abstract viewpoint, the task to derive suitable finite element shape functions corresponds to finding an interpolating function on a given domain (element) that is conforming with adjacent elements. That is to say, the interpolation along the edges should be identical between neighboring elements. In the current work, the interpolation at the element edges will be represented by one-dimensional Lagrange polynomials (like in conventional finite elements in 1D). In a nutshell, we choose (potentially distinct) interpolations along each for the four edges and interpolate those over the domain by using *blending operators*.

Formally, we need to interpolate a function $\Xi(\eta, \zeta)$ over the two-dimensional reference domain, chosen as $\Omega: [-1, 1] \times [-1, 1]$. To this end, we project the arbitrary bivariate function $\Xi(\eta, \zeta)$ onto a different (carefully selected) space of bivariate functions [18]. Such a projection will be denoted as $\mathcal{P}[\Xi(\eta, \zeta)]$ and is composed of two projectors $\mathcal{P}_\eta[\Xi(\eta, \zeta)]$, $\mathcal{P}_\zeta[\Xi(\eta, \zeta)]$ that interpolate $\Xi(\eta, \zeta)$ along the local directions η and ζ . Additionally, a mixed projector $\mathcal{P}_{\eta\zeta}[\Xi(\eta, \zeta)]$ is introduced to ensure that redundant terms cancel out. Hence, the projection is written as

$$\mathcal{P}[\Xi(\eta, \zeta)] = \mathcal{P}_\eta[\Xi(\eta, \zeta)] + \mathcal{P}_\zeta[\Xi(\eta, \zeta)] - \mathcal{P}_{\eta\zeta}[\Xi(\eta, \zeta)] \tag{22}$$

where the mixed projection operator is defined as

$$\mathcal{P}_{\eta\zeta}[\Xi(\eta, \zeta)] = \mathcal{P}_\eta[\mathcal{P}_\zeta[\Xi(\eta, \zeta)]] \tag{23}$$

Note that the individual projection operators are both linear

$$\mathcal{P}_s[f(\eta, \zeta) + g(\eta, \zeta)] = \mathcal{P}_s[f(\eta, \zeta)] + \mathcal{P}_s[g(\eta, \zeta)] \tag{24}$$

and idempotent

$$\mathcal{P}_s[\mathcal{P}_s[f(\eta, \zeta)]] = \mathcal{P}_s[f(\eta, \zeta)] \tag{25}$$

where $f(\eta, \zeta)$ and $g(\eta, \zeta)$ are (continuous) bivariate functions and the subscript $s \in \{\eta, \zeta\}$. The projection \mathcal{P} can be thought of as an operator that interpolates the four edge-

interpolations over the domain. Hence, the interpolants along the four edges can be chosen independently of this projection.

2.4.2 Projection: Definition of operators

We now construct conformal finite element shape functions that allow elements of different sizes to be coupled. This is achieved by introducing piece-wise polynomial shape functions along the element edges, which are then interpolated over the domain using the methodology sketched in the previous subsection. To achieve this goal, a simple definition of the projection operators suffices. As already introduced in the early works of Gordon and Hall [17,18], a linear interpolation is appropriate to define the projectors as

$$\mathcal{P}_\eta[\Xi(\eta, \zeta)] = \psi_1(\eta) \Xi(\eta_1, \zeta) + \psi_2(\eta) \Xi(\eta_2, \zeta) \tag{26a}$$

$$\mathcal{P}_\zeta[\Xi(\eta, \zeta)] = \psi_1(\zeta) \Xi(\eta, \zeta_1) + \psi_2(\zeta) \Xi(\eta, \zeta_2) \tag{26b}$$

where η_i and ζ_i denote two values of each local coordinate used to define the linear interpolation. We make the obvious choice of using the four element corners; thus, the functions $\psi_i(s)$ are identical to the linear Lagrange polynomials. In this context, they are referred to as linear *blending functions*

$$\psi_1(s) = \frac{1}{2}(1 - s) \tag{27a}$$

$$\psi_2(s) = \frac{1}{2}(1 + s) \tag{27b}$$

Note that the projections $\mathcal{P}_\eta[\Xi(\eta, \zeta)]$, $\mathcal{P}_\zeta[\Xi(\eta, \zeta)]$ as defined in Eqs. (26) involve the values of the interpolated function at the edges ($\Xi(\eta_1, \zeta)$ etc.). These function values will later be approximated using adequate (piece-wise) interpolation functions along the element edges. The definition of the mixed projection operator immediately follows from the successive application of Eqs. (26a) and (26b) to the function $\Xi(\eta, \zeta)$:

$$\begin{aligned} \mathcal{P}_{\eta\zeta}[\Xi(\eta, \zeta)] &= \psi_1(\eta) \psi_1(\zeta) \Xi(\eta_1, \zeta_1) + \psi_2(\eta) \psi_1(\zeta) \Xi(\eta_2, \zeta_1) \\ &\quad + \psi_1(\eta) \psi_2(\zeta) \Xi(\eta_1, \zeta_2) + \psi_2(\eta) \psi_2(\zeta) \Xi(\eta_2, \zeta_2) \end{aligned} \tag{28}$$

The mixed projector interpolates $\Xi(\eta, \zeta)$ only at discrete nodal locations. Obviously, in the presented form, the mixed projector by itself is identical to an interpolation using standard linear 2D shape functions. Based on the presented concept, we are now able to derive the shape functions for arbitrary transition elements. To this end, $\Xi(\eta, \zeta)$ is chosen such that it represents the piece-wise polynomial function that enables coupling elements of different sizes. Recall that the goal is to couple three-dimensional SBF-subdomains

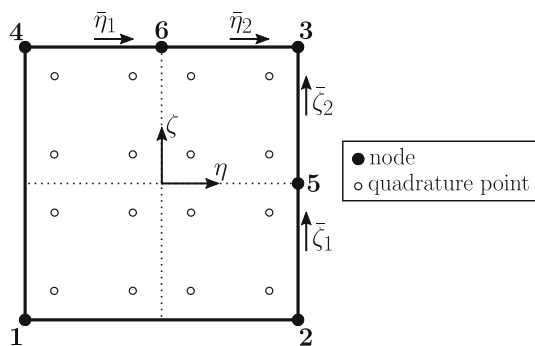


Fig. 5 Linear transition element with two ‘hanging’ nodes—illustration of local coordinate systems and integration points

of different sizes. In this case, it suffices to ensure that the edges feature conformal shape functions, as long as polynomials of degree $p \leq 3$ are used. If high-order elements or tensor product formulations are deployed, we also have to account for interior shape functions (‘bubble functions’) to ensure a conformal coupling of surfaces [11, 12]. Note that the bubble functions do not need to be included in the projection process since they vanish on all edges by construction, i.e., they are identical to those of standard finite elements. One possibility is to only add the hierarchical bubble modes that are known from the p -version of the FEM [59] to complete the polynomial.

2.5 Example: linear shape functions

Since this procedure of creating finite element shape functions may look rather unfamiliar to most readers, let us clarify the notation by considering a simple four-noded linear element. Keep in mind that we *always* employ linear blending, irrespective of the element order we ultimately wish to obtain. Hence, the projectors are still given by Eqs. (26) or, in explicit form

$$\mathcal{P}_\eta[\Xi(\eta, \zeta)] = \frac{1 - \eta}{2} \Xi(-1, \zeta) + \frac{1 + \eta}{2} \Xi(1, \zeta) \quad (29a)$$

$$\mathcal{P}_\zeta[\Xi(\eta, \zeta)] = \frac{1 - \zeta}{2} \Xi(\eta, -1) + \frac{1 + \zeta}{2} \Xi(\eta, 1) \quad (29b)$$

$$\begin{aligned} \mathcal{P}_{\eta\zeta}[\Xi(\eta, \zeta)] &= \frac{(1 - \eta)(1 - \zeta)}{4} \Xi(-1, -1) + \frac{(1 + \eta)(1 - \zeta)}{4} \Xi(1, -1) \\ &+ \frac{(1 - \eta)(1 + \zeta)}{4} \Xi(-1, 1) + \frac{(1 + \eta)(1 + \zeta)}{4} \Xi(1, 1) \end{aligned} \quad (30)$$

Equations (29) contain the function values along the four edges $\Xi(-1, \zeta)$, $\Xi(1, \zeta)$, $\Xi(\eta, -1)$, and $\Xi(\eta, 1)$, for which we can now choose an interpolation. For illustration purposes,

let us assume standard linear interpolation along all edges, i.e.,

$$\Xi(1, \zeta) = \psi_1(\zeta) \Xi(1, -1) + \psi_2(\zeta) \Xi(1, 1) \quad (31a)$$

$$\Xi(\eta, 1) = \psi_1(\eta) \Xi(-1, 1) + \psi_2(\eta) \Xi(1, 1) \quad (31b)$$

$$\Xi(-1, \zeta) = \psi_1(\zeta) \Xi(-1, -1) + \psi_2(\zeta) \Xi(-1, 1) \quad (31c)$$

$$\Xi(\eta, -1) = \psi_1(\eta) \Xi(-1, -1) + \psi_2(\eta) \Xi(1, -1) \quad (31d)$$

Substituting Eqs. (31) into (29) and using Eq. (22), we immediately obtain the two-dimensional interpolation as

$$\mathcal{P}[\Xi(\eta, \zeta)] = \mathcal{P}_{\eta\zeta}[\Xi(\eta, \zeta)] \quad (32)$$

which again is identical to the interpolation by standard linear shape functions. Similarly, by employing quadratic interpolation with evenly spaced nodes along all edges, we recover the well-known shape functions of the eight-node ‘Serendipity’ element. A graphical derivation of similar shape functions is given in [68], illustrating the more general concept developed in [13] for simple elements. By further increasing the polynomial degree, we can construct elements with an arbitrary interpolation order on the edges. However, if the polynomial degree exceeds three, the resulting shape functions are incomplete and must be enriched by bubble functions to ensure high-order completeness.

Returning to the current purpose of creating transition elements, we can now choose *piece-wise* polynomial interpolation along any of the edges. As an example, let us consider the surface indicated in Fig. 2 with two hanging nodes (pattern 3 in Fig. 3). Of course, the projection operators Eqs. (29) and (30) are still valid. Also, the linear interpolation along the edges $\Xi(-1, \zeta)$ and $\Xi(\eta, -1)$ remains the same. Instead of Eqs. (31a) and (31b), we apply the piece-wise interpolation as

$$\Xi(1, \zeta) = \quad (33a)$$

$$\begin{cases} \psi_1(\bar{\zeta}_1) \Xi(1, -1) + \psi_2(\bar{\zeta}_1) \Xi(1, 0), & -1 < \zeta \leq 0 \\ \psi_1(\bar{\zeta}_2) \Xi(1, 0) + \psi_2(\bar{\zeta}_2) \Xi(1, 1), & 0 < \zeta < 1 \end{cases}$$

$$\Xi(\eta, 1) = \quad (33b)$$

$$\begin{cases} \psi_1(\bar{\eta}_1) \Xi(-1, 1) + \psi_2(\bar{\eta}_1) \Xi(0, 1), & -1 < \eta \leq 0 \\ \psi_1(\bar{\eta}_2) \Xi(0, 1) + \psi_2(\bar{\eta}_2) \Xi(1, 1), & 0 < \eta < 1 \end{cases}$$

Here, $\bar{\eta}_1, \bar{\eta}_2, \bar{\zeta}_1, \bar{\zeta}_2$ are the local coordinates of the individual sections, i.e.,

$$\bar{\eta}_1 = 2\eta + 1 \quad (34a)$$

$$\bar{\eta}_2 = 2\eta - 1 \quad (34b)$$

$$\bar{\zeta}_1 = 2\zeta + 1 \quad (34c)$$

$$\bar{\zeta}_2 = 2\zeta - 1 \quad (34d)$$

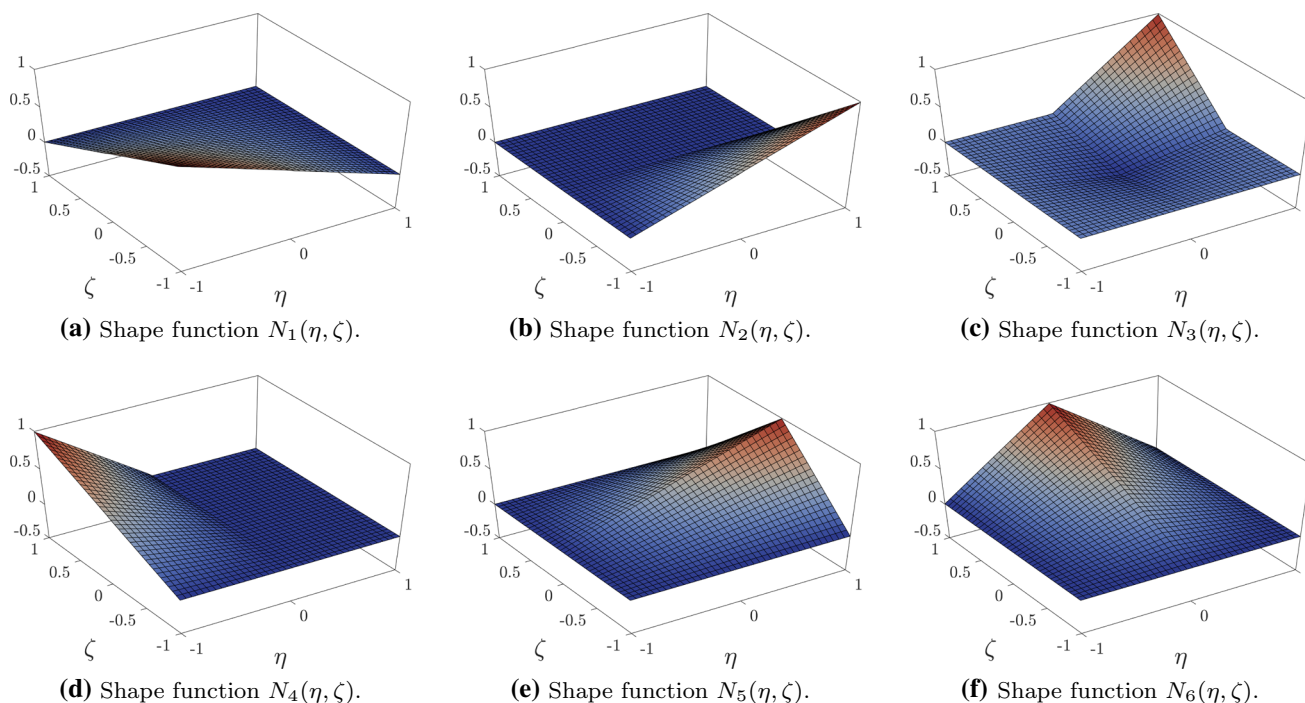


Fig. 6 Example of linear transfinite element shape functions on a quadrilateral with two hanging nodes

The local coordinates along with the nodal numbering scheme are depicted in Fig. 5. Generally, once the (piece-wise) interpolations along the edges have been chosen, they are substituted into the definition of the projection operators to obtain the 2D shape functions. Considering this simple linear example, closed-form expressions of the shape functions can be obtained directly in the coordinates η, ζ , as provided by Gupta [28]:

$$N_1(\eta, \zeta) = \frac{1}{4}(1 - \eta)(1 - \zeta) \tag{35a}$$

$$N_2(\eta, \zeta) = \frac{1}{4}(1 + \eta)(|\zeta| - \zeta) \tag{35b}$$

$$N_3(\eta, \zeta) = \frac{1}{4}[(1 + \eta)(|\zeta| + \zeta) - (1 - |\eta|)(1 + \zeta)] \tag{35c}$$

$$N_4(\eta, \zeta) = \frac{1}{4}(|\eta| - \eta)(1 + \zeta) \tag{35d}$$

$$N_5(\eta, \zeta) = \frac{1}{2}(1 + \eta)(1 - |\zeta|) \tag{35e}$$

$$N_6(\eta, \zeta) = \frac{1}{2}(1 - |\eta|)(1 + \zeta) \tag{35f}$$

The shape functions obtained for this example are plotted in Fig. 6. Note that they fulfil the ‘Kronecker-delta-property’, i.e., each shape function equals one at exactly one of the six nodes and vanishes at all other nodes. Using the above recipes, we can apply any combination of piece-wise interpo-

lation of arbitrary order on each of the edges. Many examples and detailed analyses can be found in [13].

2.6 Integration of piece-wise shape functions

When the shape functions are created based on piece-wise polynomial interpolation, they are typically not continuously differentiable everywhere inside the element. This is obviously the case in the example presented in Fig. 6. Hence, to accurately integrate the coefficient matrices according to Eq. (12), it is indispensable to divide the element into ‘patches’ for the numerical quadrature. In our example, it is sufficient to divide the integration domain into four equal-sized patches (see the quadrature points indicated in Fig. 5 for a piece-wise linear element). The number of integration points per patch is related to the polynomial order of the shape functions as $(p + 1)^2$. If edges are divided into more than two sections, the number of patches needs to be increased accordingly. Within each of the patches, the shape functions are polynomials and are integrated accurately using standard Gauss quadrature of adequate order (corresponding to the maximum order of interpolation on the edges).

3 Numerical examples

In this section, we present the results of several numerical experiments that we performed in order to verify the proposed

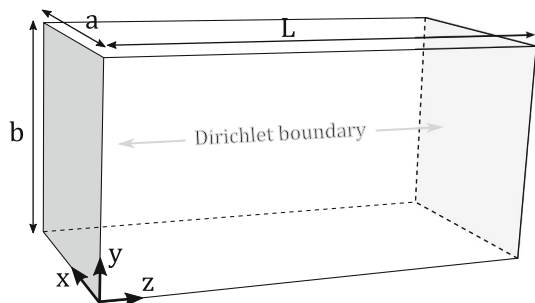


Fig. 7 Geometry used to perform patch tests

approach. We begin by analyzing simple geometries—namely a rectangular cuboid and a cube—and conduct standard patch tests, as well as a modal analysis. We then proceed to demonstrate the applicability of the approach to more complex geometries. In all examples, we assume linear elastic material behavior as well as small deformations. We denote by Ω the computational domain, and Γ_u, Γ_q are the parts of the boundary where Dirichlet and Neumann boundary conditions are applied, respectively. Hence, the general problem statement may be written as

$$\mathcal{L}^T \mathbf{D} \mathcal{L} \mathbf{u} - \rho \ddot{\mathbf{u}} - \mathbf{f} = \mathbf{0} \quad \text{in } \Omega \quad (36a)$$

$$\mathbf{u} = \mathbf{u}_\Gamma \quad \text{on } \Gamma_u \quad (36b)$$

$$\mathbf{n}^T \boldsymbol{\sigma} = \mathbf{n}^T \boldsymbol{\sigma}_\Gamma \quad \text{on } \Gamma_q \quad (36c)$$

where \mathbf{u}_Γ and $\boldsymbol{\sigma}_\Gamma$ denote the boundary conditions and \mathcal{L} is the differential operator as given in Eq. (3).

3.1 Static analyses and patch tests

We begin by conducting linear and higher-order patch tests to check the validity of the proposed approach and evaluate the rate of convergence under optimal conditions. These tests are performed for a cuboid with a width and height of $a = b = 2$ and a length of $L = 4$, see Fig. 7. Hence, the computational domain is given as

$$\Omega = \left\{ (x, y, z) \in \mathbb{R}^3 \mid 0 \leq x \leq 2, 0 \leq y \leq 2, 0 \leq z \leq 4 \right\}$$

In all tests, Dirichlet boundary conditions are applied to the two faces indicated in Fig. 7, while all other surfaces are traction-free. The material is isotropic and homogeneous with the following properties:

Young's modulus:	$E = 1$
Poisson's ratio:	$\nu = 0$

Despite the fact that the material is homogeneous, we choose to divide the domain into two regions, where different ele-

ment sizes are employed. Thus, transition elements are incorporated at the interface between the two regions (see Fig. 8 for examples). The element order is varied between $p = 1$ and $p = 3$. We perform h -refinement by dividing each subdomain into eight in each refinement step. The element size h is defined as the edge length of the largest element in the mesh. To assess the accuracy of the computed results, we evaluate the L_2 norm of the relative error in displacements with respect to an analytical solution.⁷ We study the following three cases:

Uniaxial tension We assume a state of uniaxial tension, such that the exact solution of this problem is

$$\mathbf{u}^{\text{ref}} = (0, 0, z)^T \quad (37)$$

and consequently, the following Dirichlet boundary conditions are applied:

$$\mathbf{u}_{\Gamma_1} = \mathbf{u}^{\text{ref}}(x, y, 0) = (0, 0, 0)^T \quad (38)$$

$$\mathbf{u}_{\Gamma_2} = \mathbf{u}^{\text{ref}}(x, y, L) = (0, 0, L)^T \quad (39)$$

Hence, the numerical errors should be negligible, as long as the elements on the surface of each subdomain are capable of representing a linear variation of the displacement field exactly. Figure 8a shows the computed errors when performing h -refinement with varying element order. The results show that the proposed approach passes the linear patch test with error levels in the order of 10^{-13} .

Beam bending As a 2nd order patch test, we analyze the cuboid (with the same properties as before) in a state of pure bending (with a unit radius of curvature). The analytical solution of the displacement field is given in [60], Chapter 10, as

$$u_x^{\text{ref}} = -\frac{1}{2}(z^2 + \nu(x^2 - y^2)) \quad (40)$$

$$u_y^{\text{ref}} = -xy\nu \quad (41)$$

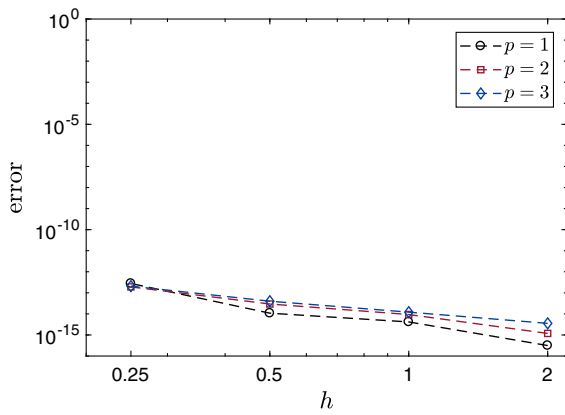
$$u_z^{\text{ref}} = xz \quad (42)$$

Hence, for the case of $\nu = 0$, our Dirichlet boundary conditions read

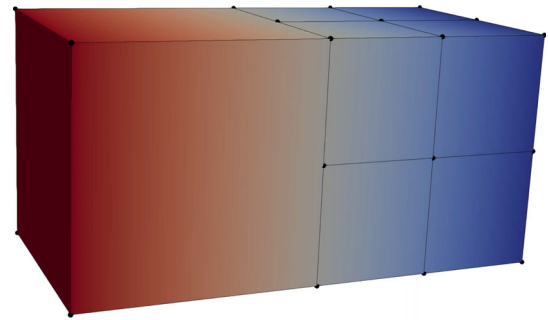
$$\mathbf{u}_{\Gamma_1} = \mathbf{u}^{\text{ref}}(x, y, 0) = (0, 0, 0) \quad (43)$$

$$\mathbf{u}_{\Gamma_2} = \mathbf{u}^{\text{ref}}(x, y, L) = \left(-\frac{1}{2}L^2, 0, xL \right) \quad (44)$$

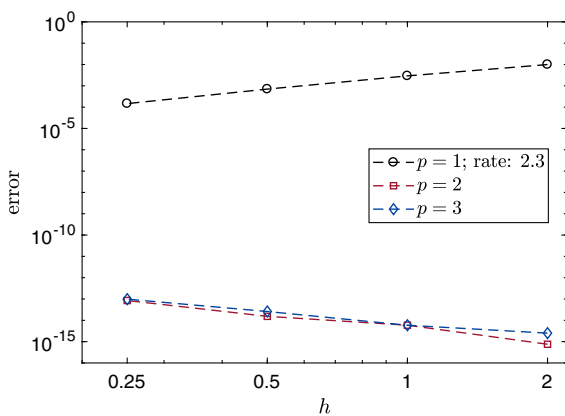
⁷ Note that, when computing stresses, one has to keep in mind that the transfinite shape functions are not continuously differentiable, as mentioned in Sect. 2.6. Evaluating stresses is, however, not conceptually different from conventional finite elements, where stresses are discontinuous between adjacent elements [13]. Furthermore, to obtain stresses in an SBFE subdomain, the analytical solution in the ξ -direction has to be taken into account. Details on stress analyses are provided in [53].



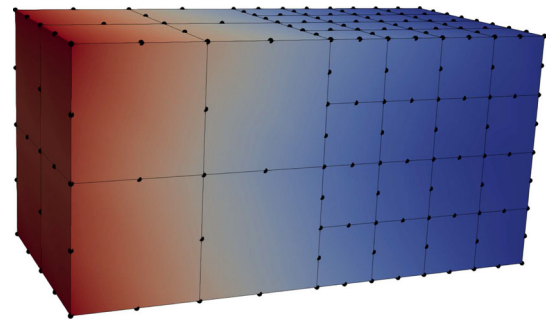
(a) 1st order patch test — uniaxial tension.



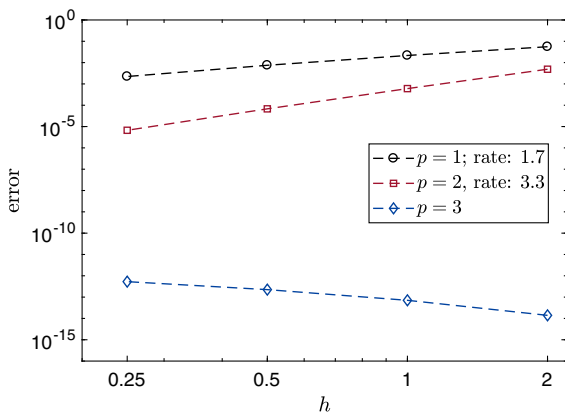
(b) Displacement field for the uniaxial tension test illustrated on an exemplary mesh: $h = 2, p = 1$.



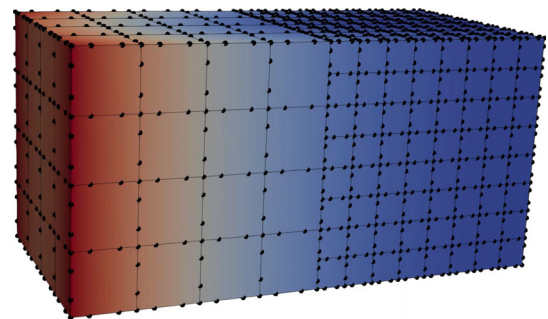
(c) 2nd order patch test — bending.



(d) Displacement field for the beam bending test illustrated on an exemplary mesh: $h = 1, p = 2$.



(e) 3rd order patch test — cantilever beam.



(f) Displacement field for the cantilever beam test illustrated on an exemplary mesh: $h = 0.5, p = 3$.

Fig. 8 Results of the 1st-, 2nd-, and 3rd-order patch tests: Convergence of the relative errors in displacement (left); computed displacement field and exemplary meshes (right)

Results are presented in Fig. 8c. Since the exact solution is a quadratic function of x, y, z , the errors are negligible when using an element order of $p \geq 2$. On the other hand, if an element order of $p = 1$ is chosen, the results are not exact. In this case, the numerical convergence rate—computed based on the last two points of the graphs—is obtained with a value of 2.3.

Cantilever beam Finally, we perform a 3rd order patch test by applying boundary conditions that result in a cubic variation of the displacement field. The analytical reference solution of the displacement field $\mathbf{u}^{\text{ref}}(x, y, z) = (u_x^{\text{ref}}, u_y^{\text{ref}}, u_z^{\text{ref}})$ is based on a cantilever beam, which is weakly fixed on one end and subject to a shear force F applied at the other end (at $z = 4$ in our example). The analytical solution can be found in [4]:

$$u_x^{\text{ref}} = -\frac{F\nu}{EI}xyz \tag{45}$$

$$u_y^{\text{ref}} = \frac{F}{EI} \left[\frac{\nu}{2}(x^2 - y^2)z - \frac{1}{6}z^3 \right] \tag{46}$$

$$u_z^{\text{ref}} = \frac{F}{EI} \left[\frac{1}{2}y(\nu x^2 + z^2) + \frac{1}{6}\nu y^3 + (1 + \nu) \left(b^2y - \frac{1}{3}y^3 \right) - \frac{1}{3}a^2\nu y - \frac{4a^3\nu}{\pi^3} \sum_{n=1}^{\infty} \frac{(-1)^n}{n^3} \cos(n\pi x/a) \frac{\sinh(n\pi y/a)}{\cosh(n\pi b/a)} \right] \tag{47}$$

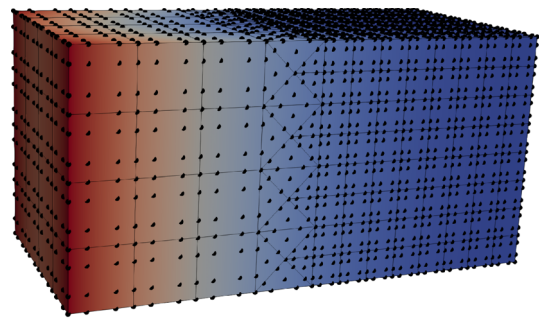
According to this analytical solution, Dirichlet boundary conditions are applied at the two ends of the cantilever as

$$\mathbf{u}_{\Gamma_1} = \frac{F}{EI} \left(0, 0, b^2y - \frac{1}{3}y^3 \right)^T \tag{48}$$

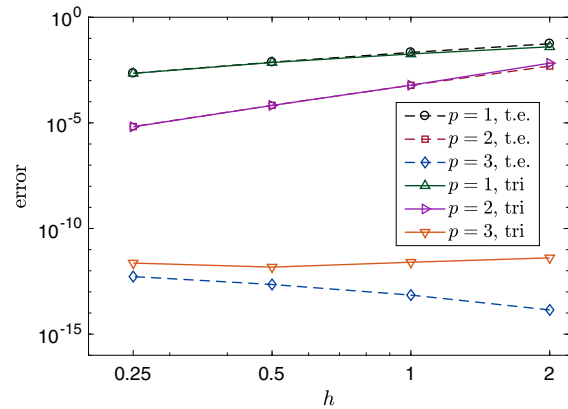
$$\mathbf{u}_{\Gamma_2} = \frac{F}{EI} \left(0, -\frac{1}{6}L^3, -\frac{1}{3}y^3 + \left[\frac{1}{2}L^2 + b^2 \right] y \right)^T \tag{49}$$

Again, all other surfaces are traction-free. As expected, the error is negligible when using an element order of $p = 3$, see Fig. 8e. For linear and quadratic elements, we obtain a numerical convergence rate of 1.7 and 3.3, respectively.

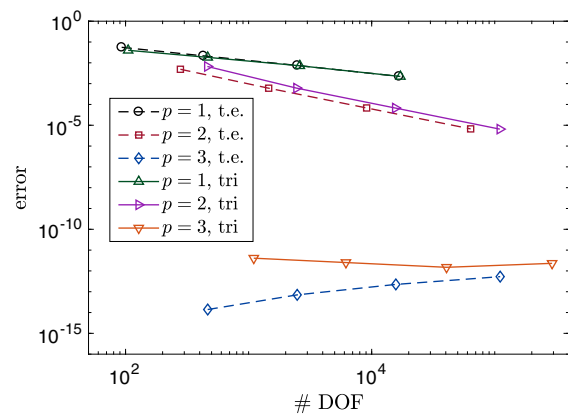
Comparison against triangulation In addition to the above verification against analytical solutions, we compare the proposed approach to a previous implementation, in which surfaces are subdivided into triangular and rectangular elements rather than using transition elements (see the explanations in Sect. 2.3). Such a discretization is depicted in Fig. 9a and can be compared directly with the corresponding mesh in Fig. 8f. For the sake of brevity, we present here only the results for the cantilever beam. A comparison of the error in displacements using both approaches is presented in Fig. 9b. The results obtained based on the proposed discretization



(a) Exemplary mesh with triangulation.



(b) Error vs. element size.



(c) Error vs. degrees of freedom.

Fig. 9 Third-order patch tests—comparison between transition elements (‘t.e.’) and triangulated surfaces (‘tri’)

(denoted as ‘t.e.’ in the figure) are, of course, identical to the ones in Fig. 8e and are included here to facilitate the comparison. It can be seen that the application of triangulated surfaces (‘tri’) results in very similar error levels for the same element size. Only when using an element order

⁸ For conciseness, we refer to these meshes as ‘triangulated’ even though some of the surfaces are subdivided into rectangles to create conforming meshes.

of $p = 3$, we observe a significant increase of the error levels—particularly for large element sizes—when using triangular elements. Plotting the same errors against the number of degrees of freedom (Fig. 9c), the triangulated mesh obviously requires more DOFs to achieve the same error level as the transition elements. This is due to the fact that the quadtree patterns lead to a larger number of interior nodes when applying the triangulation (cf. Figs. 3 and 4). While in the case of linear elements (in the current example), only a few additional nodes are required at the centroids of triangulated surfaces, the difference in the number of DOFs can be quite significant for high-order elements. For instance, when comparing the finest discretization at $p = 3$, the numbers of DOFs differ by a factor of approximately 2.6 (292,539 vs. 110,715 DOFs).

3.2 Modal analysis: cube

We continue by performing a modal analysis in order to validate the proposed approach for dynamic problems. The computational domain is a cube of width 8:

$$\Omega = \left\{ (x, y, z) \in \mathbb{R}^3 \mid 0 \leq x \leq 8, 0 \leq y \leq 8, 0 \leq z \leq 8 \right\}$$

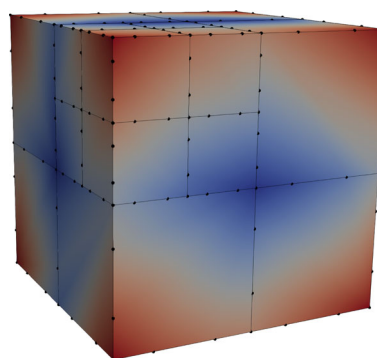
The material parameters are chosen as

Young's modulus:	$E = 1$
Poisson's ratio:	$\nu = 0$
Mass density:	$\rho = 1$

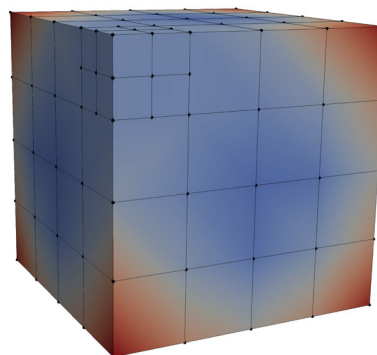
All surfaces are traction-free. The initial mesh is defined by $(8/h)^3$ cubic subdomains, where h denotes the subdomains' side length. To ensure that transition elements are present in the mesh, one of these SBFE-subdomain at one of the corners of the cube is divided into eight, see Fig. 10 for exemplary meshes. A numerical reference solution has been computed by utilizing the conventional spectral element method on a uniform grid of 8^3 elements of order 5, leading to 206,763 DOFs. The eigenfrequencies are listed in Table 1. The relative error of the first ten nonzero eigenfrequencies (modes 7–16) is plotted in Fig. 10 with respect to the element size h for different element orders p . When using $p = 1, 2, 3$, the computed numerical convergence rates are obtained as 1.8, 3.8, and 4.5, respectively (Fig. 11).

3.3 Modal analysis: crane tower

To demonstrate the applicability of the proposed approach to more complex geometries, the model of a crane tower, as depicted in Fig. 12, is analyzed. The model is based on a



(a) Exemplary mesh, $h = 4, p = 3$, Mode 7 (1st nonzero mode).



(b) Exemplary mesh, $h = 2, p = 1$, Mode 9 (3rd nonzero mode).

Fig. 10 Modal analysis of a cube—exemplary meshes and mode shapes

Table 1 Modal analysis of a cube—Reference solution for the first ten nonzero eigenfrequencies

Mode no.	Eigenfrequency
7	0.063666938067
8	0.063666949380
9	0.108860021116
10	0.108860021166
11	0.108860027908
12	0.108860036839
13	0.108860080965
14	0.108861627176
15	0.117218751959
16	0.117218866414

stereolithography (STL) file obtained from the online repository Thingiverse [41].⁹ From the STL-file, we generated a voxel-based model with the help of the online converter Voxelizer [63]. We slightly modified the design—mainly to remove unconstrained parts as well as to assign two different

⁹ The particular design *crane tower* has been uploaded by Tung Nguyen (username *tungnt*) and published under the terms of the GNU General Public License. The design can be retrieved from www.thingiverse.com/thing:2440385.

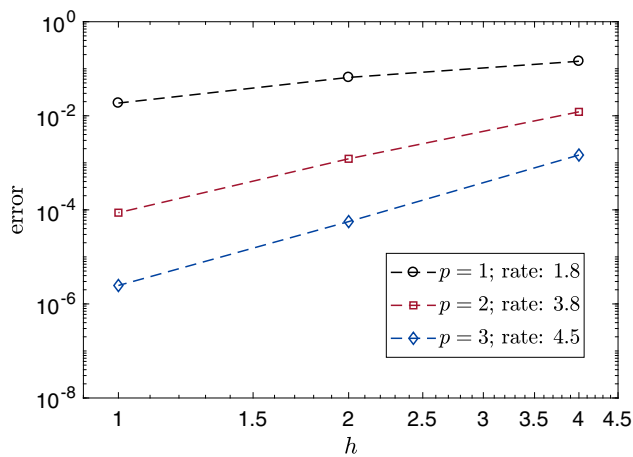


Fig. 11 Modal analysis of a cube—relative error of the eigenfrequencies

colors that correspond to different materials in the context of the image-based analysis. The modifications have been made using the software *MagicaVoxel* [14]. The material parameters of the tower and the base (as indicated by the different colors) are assumed as

	material 1	material 2
Young’s modulus E :	70 GPa	20 GPa
Poisson’s ratio ν :	0.35	0.3
mass density ρ :	2.7 kg/m ³	2.4 kg/m ³

The total height of the model is 22.5 m. We apply fixed boundary conditions at the bottom of the base, as indicated in Fig. 12. The mesh has been created automatically by means of an octree decomposition of the voxel-based model, leading to 7158 subdomains. As can be inferred from Fig. 12, the decomposition leads to subdomains of four different sizes and, therefore, the side length of the largest element is eight times larger than that of the smallest elements. A rapid and consistent transition from large to small element sizes is easily achieved by the octree meshing algorithm. Figure 13 shows the mode shapes of the first four modes and lists the corresponding eigenfrequencies. We found that for computing these modes, it suffices to employ quadratic elements on the surfaces of each subdomain, which results in a total of 615,960 DOFs (Table 2).

3.4 Structure under self-weight loading

As a final example, we study the behavior of the structure depicted in Fig. 14 under the influence of self-weight loading. The model of a castle is based on a sample provided in the software *MagicaVoxel* [14], which we slightly modified and placed on a base of a different material. We scaled the model

Table 2 Eigenfrequencies of the crane tower, computed using varying element order p

Mode	Frequency (Hz)		
	$p = 1$	$p = 2$	$p = 3$
1	1.28	1.12	1.08
2	1.85	1.74	1.72
3	3.64	3.49	3.47
4	4.20	4.02	4.00
5	5.79	5.02	4.95
6	8.02	7.24	7.10
7	9.87	8.85	8.60
8	10.55	10.12	10.08
9	13.48	12.81	12.68
10	18.66	18.43	18.41

such that the total height of the structure is 33 m. The base is a homogeneous layer with a thickness of 8 m. The material parameters of the castle (material 1) and foundation (material 2) are chosen as

	material 1	material 2
Young’s modulus E :	10 GPa	0.5 GPa
Poisson’s ratio ν :	0.3	0.2
mass density ρ :	2.4 kg/m ³	2 kg/m ³

The acceleration of gravity is set to $g = -9.81 \text{ m/s}^2$. As a proof of concept, this example represents a situation where it can be useful to apply a different element order to the two distinct materials and thus better capture the larger deformations in the softer material. This feature will be particularly interesting for more complicated problems, such as wave propagation through layered soils, which have been addressed previously in two dimensions [27]. In those cases, it can also be useful to adjust the element order not only based on the material properties but also on the size of the subdomain. In the current rather simple example, we found it sufficient to use elements of order 3 in the base material, while the comparably rigid material of the castle is discretized using linear elements only. The resulting mesh is depicted in Fig. 14a. The deformed geometry due to self-weight can be seen in Fig. 14b, where the colors indicate absolute values of displacement.

4 Conclusion

The combination of the three-dimensional SBFEM with transition elements allows a consistent discretization of an octree decomposition. Each surface of each subdomain is

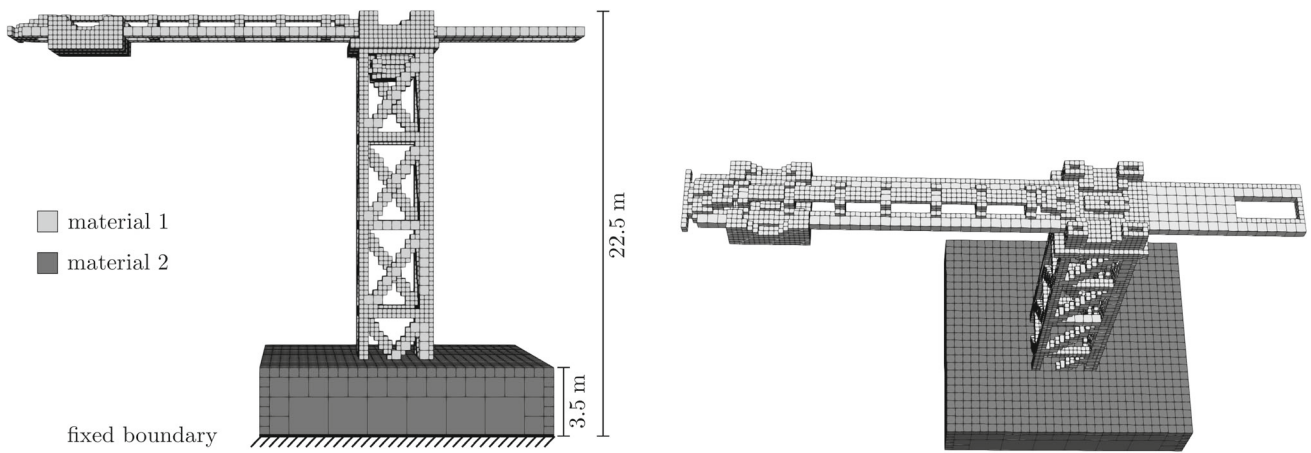
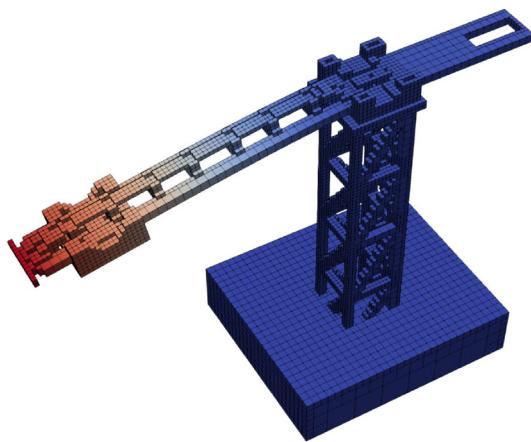
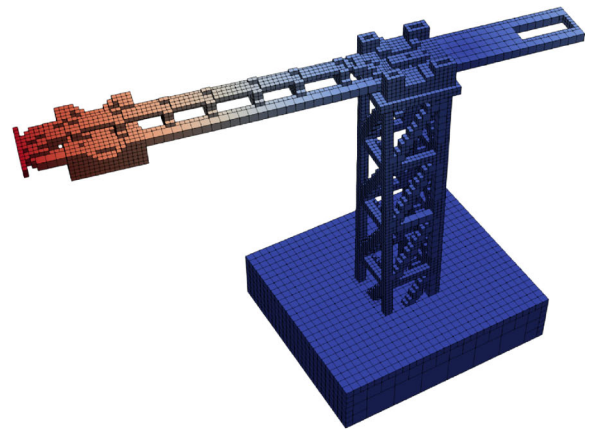


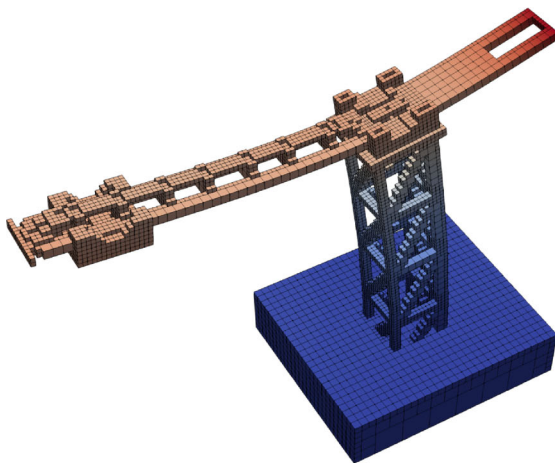
Fig. 12 Modal analysis of a crane tower—octree decomposition



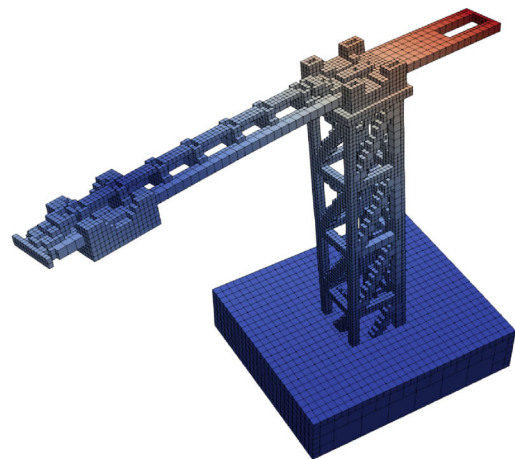
(a) Mode 1, $f = 1.1$ Hz.



(b) Mode 2, $f = 1.7$ Hz.



(c) Mode 3, $f = 3.5$ Hz.



(d) Mode 4, $f = 4.0$ Hz.

Fig. 13 Modal analysis of a crane tower—Mode shapes and eigenfrequencies of the first four modes

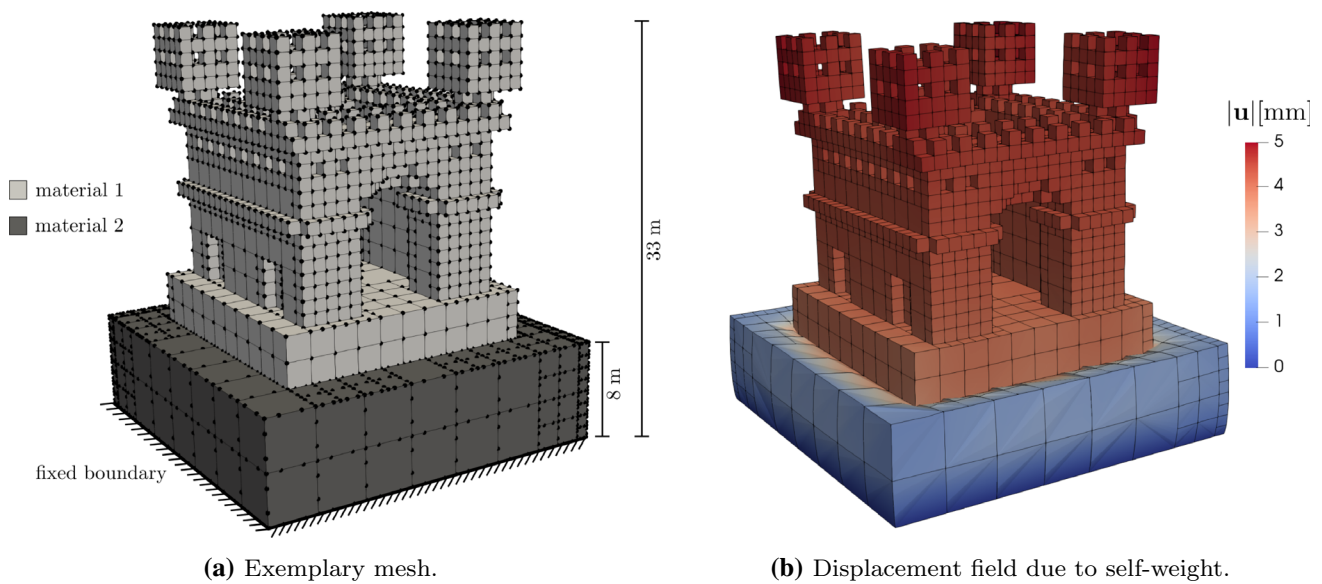


Fig. 14 Structure under self-weight loading—**a** mesh and **b** displacement solution

discretized by quadrilateral elements only. Hence, we avoid any unnecessary subdivision into smaller surface elements as had been done in previous publications. The numerical results demonstrate that the computational models created using the proposed technique pass the linear as well as higher-order patch tests. Compared to the previous meshing paradigm (involving triangulation), there is no loss of accuracy due to the transition elements, while the number of surface elements and degrees of freedom is reduced. We also demonstrated that the proposed approach allows coupling elements of different interpolation orders straightforwardly. Thus, we are now able to implement local *p*-refinement, which could previously only be exploited in two-dimensional SBFEM models.

Acknowledgements Open Access funding provided by Projekt DEAL.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

A Coordinate transformation

In Eq. (7), we provided the Jacobian matrix representing the transformation between Cartesian and ‘scaled boundary’ coordinates, which we decomposed into an ξ -dependent part

and the Jacobian matrix on the boundary $\mathbf{J}(\eta, \zeta)$. The corresponding determinant is obtained as

$$|\mathbf{J}| = x(y_{,\eta}z_{,\zeta} - z_{,\eta}y_{,\zeta}) + y(z_{,\eta}x_{,\zeta} - x_{,\eta}z_{,\zeta}) + z(x_{,\eta}y_{,\zeta} - y_{,\eta}x_{,\zeta}) \tag{50}$$

and its inverse reads

$$\mathbf{J}^{-1} = \frac{1}{|\mathbf{J}|} \begin{bmatrix} y_{,\eta}z_{,\zeta} - z_{,\eta}y_{,\zeta} & zy_{,\zeta} - yz_{,\zeta} & yz_{,\eta} - zy_{,\eta} \\ z_{,\eta}x_{,\zeta} - x_{,\eta}z_{,\zeta} & xz_{,\zeta} - zx_{,\zeta} & zx_{,\eta} - xz_{,\eta} \\ x_{,\eta}y_{,\zeta} - y_{,\eta}x_{,\zeta} & yx_{,\zeta} - xy_{,\zeta} & xy_{,\eta} - yx_{,\eta} \end{bmatrix} \tag{51}$$

The transformation of the spatial derivatives yields

$$\begin{bmatrix} \partial_{\xi} \\ \partial_{\eta} \\ \partial_{\zeta} \end{bmatrix} = \begin{bmatrix} 1 \\ \xi \\ \xi \end{bmatrix} \mathbf{J}(\eta, \zeta) \begin{bmatrix} \partial_{\hat{x}} \\ \partial_{\hat{y}} \\ \partial_{\hat{z}} \end{bmatrix} \tag{52}$$

Using Eqs. (50)–(52), we transform the governing equation into the coordinate system (ξ, η, ζ) with the transformed differential operator as given in Eq. (9). The transformation matrices $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ are obtained after some lengthy but straightforward algebra as

$$\mathbf{b}_1 = \frac{1}{|\mathbf{J}|} \begin{bmatrix} y_{,\eta}z_{,\zeta} - z_{,\eta}y_{,\zeta} & 0 & 0 \\ 0 & z_{,\eta}x_{,\zeta} - x_{,\eta}z_{,\zeta} & 0 \\ 0 & 0 & x_{,\eta}y_{,\zeta} - y_{,\eta}x_{,\zeta} \\ 0 & x_{,\eta}y_{,\zeta} - y_{,\eta}x_{,\zeta} & z_{,\eta}x_{,\zeta} - x_{,\eta}z_{,\zeta} \\ x_{,\eta}y_{,\zeta} - y_{,\eta}x_{,\zeta} & 0 & y_{,\eta}z_{,\zeta} - z_{,\eta}y_{,\zeta} \\ z_{,\eta}x_{,\zeta} - x_{,\eta}z_{,\zeta} & y_{,\eta}z_{,\zeta} - z_{,\eta}y_{,\zeta} & 0 \end{bmatrix} \tag{53a}$$

$$\mathbf{b}_2 = \frac{1}{|\mathbf{J}|} \begin{bmatrix} zy,\zeta - yz,\zeta & 0 & 0 \\ 0 & xz,\zeta - zx,\zeta & 0 \\ 0 & 0 & yx,\zeta - xy,\zeta \\ 0 & yx,\zeta - xy,\zeta & xz,\zeta - zx,\zeta \\ yx,\zeta - xy,\zeta & 0 & zy,\zeta - yz,\zeta \\ xz,\zeta - zx,\zeta & zy,\zeta - yz,\zeta & 0 \end{bmatrix} \quad (53b)$$

$$\mathbf{b}_3 = \frac{1}{|\mathbf{J}|} \begin{bmatrix} yz,\eta - zy,\eta & 0 & 0 \\ 0 & zx,\eta - xz,\eta & 0 \\ 0 & 0 & xy,\eta - yx,\eta \\ 0 & xy,\eta - yx,\eta & zx,\eta - xz,\eta \\ xy,\eta - yx,\eta & 0 & yz,\eta - zy,\eta \\ zx,\eta - xz,\eta & yz,\eta - zy,\eta & 0 \end{bmatrix} \quad (53c)$$

B Continued-fraction-expansion for high frequencies

The static stiffness matrix \mathbf{K} , as presented in Sect. 2.2, is obtained as an *exact* solution of the semi-discretized matrix differential equation for vanishing frequency. That is to say, in the static case, the accuracy of the solution is governed by the quality of the interpolation in the (η, ζ) directions, while the solution along the ξ -direction does not introduce additional errors. For the dynamic case, on the other hand, we applied an approximation of the inertia term by considering only terms that are quadratic in the frequency—see Eq. (19). This assumption is valid for small frequencies (meaning that the wavelength is considerably larger than the size of the subdomain under consideration). To enhance the accuracy for higher frequencies, high-order stiffness and mass matrices can be computed of the form

$$\mathbf{K}_h = \text{diag}(\mathbf{K}, \mathbf{S}_0^{(1)}, \mathbf{S}_0^{(2)}, \dots, \mathbf{S}_0^{(M_{cf})}) \quad (54a)$$

$$\mathbf{M}_h = \begin{bmatrix} \mathbf{M} & -\mathbf{X}^{(1)} & 0 & \dots & 0 \\ -[\mathbf{X}^{(1)}]^T & \mathbf{S}_1^{(1)} & -\mathbf{X}^{(2)} & \dots & 0 \\ 0 & -[\mathbf{X}^{(2)}]^T & \mathbf{S}_1^{(2)} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \mathbf{S}_1^{(M_{cf})} \end{bmatrix} \quad (54b)$$

where the matrices $\mathbf{S}_0^{(i)}, \mathbf{S}_1^{(i)}$ are derived from a continued fraction expansion of the dynamic stiffness matrix

$$\begin{aligned} \mathbf{S}(\omega) = & \mathbf{K} - \omega^2 \mathbf{M} - \omega^4 \mathbf{X}^{(1)} (\mathbf{S}_0^{(1)} - \omega^2 \mathbf{S}_1^{(1)}) \\ & - \omega^4 \mathbf{X}^{(2)} (\mathbf{S}_0^{(2)} - \omega^2 \mathbf{S}_1^{(2)}) - \dots - \omega^4 \mathbf{X}^{(M_{cf})} (\mathbf{S}_0^{(M_{cf})} \\ & - \omega^2 \mathbf{S}_1^{(M_{cf})})^{-1} [\mathbf{X}^{(M_{cf})}]^T)^{-1} [\mathbf{X}^{(2)}]^T)^{-1} [\mathbf{X}^{(1)}]^T \end{aligned} \quad (55)$$

and the matrices $\mathbf{X}^{(i)}$ are introduced for preconditioning. The details of this approach can be found in [1,55].

References

1. Birk C, Prempramote S, Song C (2012) An improved continued-fraction-based high-order transmitting boundary for time-domain analyses in unbounded domains. *Int J Numer Methods Eng* 89:269–298
2. Birk C, Song C (2009) A continued-fraction approach for transient diffusion in unbounded medium. *Comput Methods Appl Mech Eng* 198:2576–2590
3. Birkhoff G, Cavendish JC, Gordon WJ (1974) Multivariate approximation by locally blended univariate interpolants. *Proc Natl Acad Sci USA* 71(9):3423–3425
4. Bishop JE (2014) A displacement-based finite element formulation for general polyhedra using harmonic shape functions. *Int J Numer Methods Eng* 97:1–31
5. Bröker H (2001) Integration von geometrischer Modellierung und Berechnung nach der p-Version der FEM. Shaker Verlag, Berichte aus dem Bauwesen
6. Cavendish JC (1975) Local mesh refinement using rectangular blended finite elements. *J Comput Phys* 19:211–228
7. Chen X, Birk C, Song C (2014) Numerical modelling of wave propagation in anisotropic soil using a displacement unit-impulse-response-based formulation of the scaled boundary finite element method. *Soil Dyn Earthq Eng* 65:243–255
8. Chiong I, Ooi ET, Song C, Tin-Loi F (2014) Scaled boundary polygons with application to fracture analysis of functionally graded materials. *Int J Numer Methods Eng* 98:562–589
9. Du Q, Wang D (2006) Recent progress in robust and quality Delaunay mesh generation. *J Comput Appl Math* 195(1):8–23
10. Duzcek S (2014) Higher order finite elements and the fictitious domain concept for wave propagation analysis. *VDI Fortschritt-Berichte Reihe 20 Nr. 458*
11. Duzcek S, Gravenkamp H (2019) Critical assessment of different mass lumping schemes for higher order serendipity finite elements. *Comput Methods Appl Mech Eng* 350:836–897
12. Duzcek S, Gravenkamp H (2019) Mass lumping techniques in the spectral element method: on the equivalence of the row-sum, nodal quadrature, and diagonal scaling methods. *Comput Methods Appl Mech Eng* 353:516–569
13. Duzcek S, Saputra AA, Gravenkamp H (2020) High order transition elements: the xNy-element concept—part I: statics. *Comput Methods Appl Mech Eng* 362:112833
14. Ephtracy (2019) *MagicaVoxel 0.99.3a*. <https://ephtracy.github.io/>
15. Gabbert U, Graeff-Weinberg K (1999) Adaptive local-global analysis by pN_h transition elements. *Tech Mech* 19(2):115–126
16. Gordon WJ (1971) Blending-function methods of bivariate and multivariate interpolation and approximation. *SIAM J Numer Anal* 8:158–177
17. Gordon WJ, Hall CA (1973) Construction of curvilinear coordinate systems and applications to mesh generation. *Int J Numer Methods Eng* 7:461–477
18. Gordon WJ, Hall CA (1973) Transfinite element methods: blending-function interpolation over arbitrary curved element domains. *Numer Math* 21:109–129
19. Gordon WJ, Thiel LC (1982) Transfinite mappings and their application to grid generation. Elsevier, Amsterdam
20. Graeff-Weinberg K, Berger H (1996) Verbesserte FE-Diskretisierung bei Kontaktaufgaben. *Tech Mech* 16(3):250–270
21. Gravenkamp H (2018) Efficient simulation of elastic guided waves interacting with notches, adhesive joints, delaminations and inclined edges in plate structures. *Ultrasonics* 82:101–113
22. Gravenkamp H, Bause F, Song C (2014) On the computation of dispersion curves for axisymmetric elastic waveguides using the scaled boundary finite element method. *Comput Struct* 131:46–55

23. Gravenkamp H, Birk C, Song C (2015) Simulation of elastic guided waves interacting with defects in arbitrarily long structures using the scaled boundary finite element method. *J Comput Phys* 295:438–455
24. Gravenkamp H, Duczek S (2017) Automatic image-based analyses using a coupled quadtree-SBFEM/SCM approach. *Comput Mech* 60:559–584
25. Gravenkamp H, Natarajan S (2018) Scaled boundary polygons for linear elastodynamics. *Comput Methods Appl Mech Eng* 333:238–256
26. Gravenkamp H, Saputra AA, Duczek S (2019) High-order shape functions in the scaled boundary finite element method revisited. *Arch Comput Methods Eng*. <https://doi.org/10.1007/s11831-019-09385-1>
27. Gravenkamp H, Saputra AA, Song C, Birk C (2017) Efficient wave propagation simulation on quadtree meshes using SBFEM with reduced modal basis. *Int J Numer Methods Eng* 110:1119–1141
28. Gupta AK (1978) A finite element for transition from a fine to a coarse grid. *Int J Numer Methods Eng* 12:35–45
29. Karniadakis GE, Sherwin SJ (2005) Spectral/hp element methods for computational fluid dynamics. Oxford Science Publications, Oxford
30. Kausel E (1994) Thin-layer method: formulation in the time domain. *Int J Numer Methods Eng* 37:927–941
31. Kausel E, Gravenkamp H (2019) On the numerical solution of matrix Bessel equations. *ZAMM Zeitschrift für Angewandte Mathematik und Mechanik* 99(8):e201800288
32. Kausel E, Roësset JM, Roesset JM (1981) Stiffness matrices for layered soils. *Bull Seismol Soc Am* 71(6):1743–1761
33. Keyak J, Meagher J, Skinner H, Mote C (1990) Automated three-dimensional finite element modelling of bone: a new method. *J Biomed Eng* 12(5):389–397
34. Királyfalvi G, Szabó B (1997) Quasi-regional mapping for the p-version of the finite element method. *Finite Elem Anal Des* 27:85–97
35. Krome F, Gravenkamp H (2017) A semi-analytical curved element for linear elasticity based on the scaled boundary finite element method. *Int J Numer Methods Eng* 109:790–808
36. Krome F, Gravenkamp H, Birk C (2017) Prismatic semi-analytical elements for the simulation of linear elastic problems in structures with piecewise uniform cross section. *Comput Struct* 192:83–95
37. Liu L, Zhang J, Song C, Birk C, Gao W (2019) An automatic approach for the acoustic analysis of three-dimensional bounded and unbounded domains by scaled boundary finite element method. *Int J Mech Sci* 151:563–581
38. Liu Y, Saputra AA, Wang J, Tin-Loi F, Song C (2017) Automatic polyhedral mesh generation and scaled boundary finite element analysis of STL models. *Comput Methods Appl Mech Eng* 313:106–132
39. Löhner R, Parikh P (1988) Generation of three-dimensional unstructured grids by the advancing-front method. *Int J Numer Methods Fluids* 8(10):1135–1149
40. Lorenzen WE, Cline HE (1987) Marching cubes A high resolution 3d surface construction algorithm. *ACM SIGGRAPH Comput Graph* 21:163–169 ACM
41. MakerBot Industries: Thingiverse. <https://www.thingiverse.com/>
42. Man H, Song C, Gao W, Tin-Loi F (2012) A unified 3D-based technique for plate bending analysis using scaled boundary finite element method. *Int J Numer Methods Eng* 91:491–515
43. Man H, Song C, Natarajan S, Ooi ET, Birk C, Tat Ooi E, Birk C (2014) Towards automatic stress analysis using Scaled Boundary Finite Element Method with quadtree mesh of high-order elements. *ArXiv e-prints p. math.NA/1402.5186*
44. Ooi ET, Man H, Natarajan S, Song C (2015) Adaptation of quadtree meshes in the scaled boundary finite element method for crack propagation modelling. *Eng Fract Mech* 144:101–117
45. Ooi ET, Shi M, Song C, Tin-Loi F, Yang Z (2013) Dynamic crack propagation simulation with scaled boundary polygon elements and automatic remeshing technique. *Eng Fract Mech* 106(2012):1–21
46. Ooi ET, Song C, Tin-Loi F (2014) A scaled boundary polygon formulation for elasto-plastic analyses. *Comput Methods Appl Mech Eng* 268:905–937
47. Ooi ET, Song C, Tin-Loi F, Yang Z (2012) Automatic modelling of cohesive crack propagation in concrete using polygon scaled boundary finite elements. *Eng Fract Mech* 93:13–33
48. Pozrikidis C (2014) Introduction to finite and spectral element methods using MATLAB, 2nd edn. Chapman and Hall/CRC, Boca Raton
49. Provatidis CG (2006) Coons-patch macroelements in two-dimensional parabolic problems. *Appl Math Model* 30(4):319–351
50. Provatidis CG (2011) Two-dimensional elastostatic analysis using Coons-Gordon interpolation. *Meccanica* 47(4):951–967
51. Provatidis CG (2019) Precursors of isogeometric analysis. Springer, Berlin
52. Saputra AA, Birk C, Song C (2015) Computation of three-dimensional fracture parameters at interface cracks and notches by the scaled boundary finite element method. *Eng Fract Mech* 148:213–242
53. Saputra AA, Talebi H, Tran D, Birk C, Song C (2017) Automatic image-based stress analysis by the scaled boundary finite element method. *Int J Numer Methods Eng* 109:697–738
54. Song C (2004) A matrix function solution for the scaled boundary finite-element equation in statics. *Comput Methods Appl Mech Eng* 193:2325–2356
55. Song C (2009) The scaled boundary finite element method in structural dynamics. *Int J Numer Methods Eng* 77:1139–1171
56. Song C (2018) The scaled boundary finite element method: introduction to theory and implementation. Wiley, New York
57. Song C, Wolf JP (1997) The scaled boundary finite-element method—alias consistent infinitesimal finite-element cell method - for elastodynamics. *Comput Methods Appl Mech Eng* 147:329–355
58. Song C, Wolf JP (2000) The scaled boundary finite-element method—a primer: solution procedures. *Comput Struct* 78:211–225
59. Szabó B, Babuška I (1991) Finite element analysis. Wiley, New York
60. Timoshenko S (1951) Theory of elasticity. McGraw-Hill Book Company, New York
61. Weinberg K (1996) Ein Finite-Elemente-Konzept zur lokalen Netzverdichtung und seine Anwendung auf Koppel- und Kontaktprobleme. Ph.D. thesis, Otto von Guericke University Magdeburg
62. Weinberg K, Gabbert U (2002) An adaptive pNh-technique for global-local finite element analysis. *Eng Comput* 19:485–500
63. Westerdiep A (2019) Online Voxelizer. <http://drububu.com/miscellaneous/voxelizer/>
64. Wolf JP, Song C (1994) Dynamic-stiffness matrix in time domain of unbounded medium by infinitesimal finite element cell method. *Earthq Eng Struct Dyn* 23:1181–1198
65. Wolf JP, Song C (1994) Dynamic-stiffness matrix of unbounded soil by finite-element multi-cell cloning. *Earthq Eng Struct Dyn* 23:233–250
66. Wolf JP, Song C (2000) The scaled boundary finite-element method—a primer: derivations. *Comput Struct* 78:191–210

67. Young P, Beresford-West T, Coward S, Notarberardino B, Walker B, Abdul-Aziz A (2008) An efficient approach to converting three-dimensional image data into highly accurate computational models. *Philos Trans R Soc A Math Phys Eng Sci* 366(1878):3155–3173
68. Zienkiewicz OC, Taylor RL (2000) *The finite element method—volume 1: the basis*. Butterworth Heinemann, Oxford

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.