Proceedings of the Seventeenth International Conference on
Civil, Structural and Environmental Engineering Computing
Edited by: P. Iványi, J. Kruis and B.H.V. Topping
Civil-Comp Conferences, Volume 6, Paper 14.1
Civil-Comp Press, Edinburgh, United Kingdom, 2023
doi: 10.4203/ccc.6.14.1

Towards Fully-automated High-performance Scaled Boundary Finite Element Analysis

Ch. Song[1], J.Q. Zhang[2], A.S. Kumar[1] and Y.F. Zhan[1]

[1]Centre for Infrastructure Engineering and Safety, University of New South Wales, Sydney, Australia
[2]Key Laboratory of Urban Security and Disaster Engineering of Ministry of Education,
Beijing University of Technology, Beijing, China

Abstract

This paper presents the development of the scaled boundary finite element method to benefit from modern technologies for geometrical modelling and high-performance computing. The scaled boundary finite element method allows the use of arbitrarily shaped star-convex polyhedral elements. The greater flexibility in spatial discretization than standard finite elements facilitates automatic mesh generation. A simple and efficient octree algorithm is developed to mesh geometric models given in common formats such as conventional CAD, STL, digital images, and point clouds. By identifying suitable transformations of the octree cells, a mesh can be deconstructed into a limited number of unique cell patterns. A pattern-by-pattern method for computing matrix-vector products in explicit dynamics and iterative solvers is developed. The operations grouping elements of the same pattern reduce the memory requirement and improve the parallel computation efficiency. Numerical examples of large-scale problems with complex geometries are presented. A significant speedup is observed for these examples with up to 1 billion degrees of freedom and running on up to 16,384 computing cores.

Keywords: scaled boundary finite element method, mesh generation, octree mesh, parallel computing, statics, dynamics

1 Introduction

Computer simulation using the finite element method [1] is nowadays routinely employed in engineering practice. The process may require extensive human intervention in mesh generation and is often time-consuming for engineers. At the same time, digital technologies for geometrical acquisition and modelling are being rapidly adopted in engineering practice. Using terrestrial laser scanning and close-range photogrammetry-based techniques, it becomes possible to complete near real-time shape acquisition of structures that do not possess a CAD model or that have been subjected to damaging effects. X-ray computed tomography (XCT) can accurately obtain digital images of the internal structures of materials [2]. The data formats commonly employed in these digital technologies include STereoLithography (STL), voxel data, and point clouds. These data formats cannot be used directly to generate finite element models by employing the current mesh generation methods that have been developed for CAD models. An involved surface reconstruction must be conducted to convert the model to a typical CAD file format [3]. The process is not only very time consuming, but more importantly, also rather prone to errors.

In the finite element analysis of large-scale problems, the primary computational bottleneck is the solution of large linear systems of equations [4]. Modern high-performance computing (HPC) systems are equipped with a vast number of Central Processing Units (CPUs) and Graphics Processing Units (GPUs). Compared with the computational speed of the processors, the memory requirement and memory access are frequently the most critical factors affecting the computational efficiency of a finite element analysis on an HPC system. Iterative solvers and explicit dynamics solvers that rely on element-by-element operations of matrix-vector products and do not need the assembly of global matrices are often preferred due to their high parallel efficiency.

The scaled boundary finite element method has been developed into a general-purpose numerical method for the solution of PDE problems [5–17] over the last few years. This paper aims to present a scaled boundary finite element framework that automates mesh generation and is suitable to high-performance computing.

2 The Scaled Boundary Finite Element Method

The concept of the scaled boundary finite element method is briefly summarized. The reader is referred to [18].

The construction of a scaled boundary finite element is illustrated in Figure 1. The geometry of the element needs to satisfy the star convexity criterion, that is, any point on its boundary must be directly visible from a point called the scaling centre. The element can be a polyhedron with no restriction on the number of nodes, edges, and faces. The boundary of the element is discretized into two dimensional surface (polygonal) elements with the nodal coordinates $\boldsymbol{x}_e$ and the shape functions $\boldsymbol{N}_S(\eta, \zeta)$ formulated

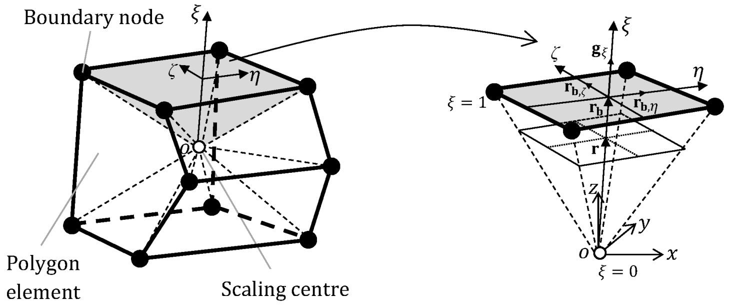


Figure 1: A scaled boundary finite element with surface discretization.

in local coordinates η and ζ. The polyhedral element is described by scaling its faces with respect to a scaling centre and expressed as

$$\boldsymbol{x}(\xi, \eta, \zeta) = \xi \boldsymbol{N}_S(\eta, \zeta)\boldsymbol{x}_e \tag{1}$$

where ξ is the radial coordinate chosen as 0 at the scaling centre and 1 on the surface. The displacement field $\boldsymbol{u} = \boldsymbol{u}(\xi, \eta, \zeta)$ is represented semi-analytically by introducing nodal functions $\boldsymbol{u}_r = \boldsymbol{u}_r(\xi)$ along the radial lines connecting the scaling centre and a node on the boundary and numerically interpolating between the radial lines. The solution of scaled boundary finite element equation is obtained analytically [19]. This leads to the semi-analytical solution interpolating the displacement $\boldsymbol{u}(\xi, \eta, \zeta)$ in the polyhedron from the nodal values $\boldsymbol{u}_e = \boldsymbol{u}_r(\xi = 1)$ and shape functions $\boldsymbol{N}_V(\xi, \eta, \zeta)$

$$\boldsymbol{u} = \boldsymbol{N}_V(\xi, \eta, \zeta)\boldsymbol{u}_e, \qquad \text{with} \quad \boldsymbol{N}_V(\xi, \eta, \zeta) = \boldsymbol{N}_S(\eta, \zeta)\boldsymbol{V}\xi^{-\boldsymbol{S}}\boldsymbol{V}^{-1} \tag{2}$$

with the displacement modes $\boldsymbol{V}$ and the diagonal matrix $\boldsymbol{S}$. The strain field $\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}(\xi, \eta, \zeta)$ is obtained from the displacement field in Eq. (2) and the B-matrix $\boldsymbol{B}_V = \boldsymbol{B}_V(\xi, \eta, \zeta)$ is defined as ($\boldsymbol{V}_\varepsilon$: strain modes)

$$\boldsymbol{\varepsilon} = \boldsymbol{B}_V\boldsymbol{u}_e; \qquad \text{with} \quad \boldsymbol{B}_V = \boldsymbol{V}_\varepsilon\xi^{-S}\boldsymbol{V}^{-1} \tag{3}$$

Standard finite element procedures can be followed to compute the element stiffness matrix, mass matrix and internal force vector

$$\boldsymbol{k}_e = \int_V \boldsymbol{B}_V^T\boldsymbol{D}\boldsymbol{B}_V\mathrm{d}V; \boldsymbol{m}_e = \int_V \boldsymbol{N}_V^T\rho\boldsymbol{N}_V\mathrm{d}V; \quad \boldsymbol{p}_e(\boldsymbol{\sigma}) = \int_V \boldsymbol{B}_V^T\boldsymbol{\sigma}\mathrm{d}V \tag{4}$$

3 Automatic Mesh Generation

The flexibility of the scaled boundary finite elements are highly complementary with the efficient octree algorithm for mesh generation. A problem domain is divided into cells by recursively bisecting the cell edges until specified stopping criteria are met [2]. An example of octree mesh in shown in Figure 2a. An octree mesh is usually balanced by limiting the length ratio between two adjacent cells to 2 ($2:1$ rule). All the possible

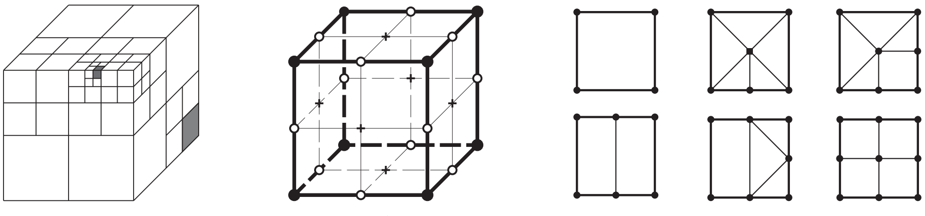


Figure 2: (a) Example of balanced octree mesh; (b) A balanced octree cell indicating all possible corner (•), mid-edge (○) and center ($+$) nodes; (c) Surface discretization eliminating hanging nodes.

locations of nodes on a balanced octree cell are indicated in Figure 2b. There are $2^{12} = 4096$ cell patterns in total [18]. Each octree cell is modelled as a scaled boundary finite element. On the faces of the octree cells, only 6 unique patterns exist. They are easily divided into triangular and quadrilateral elements as shown in Figure 2c to eliminate the hanging nodes between polyhedral elements of the different sizes.

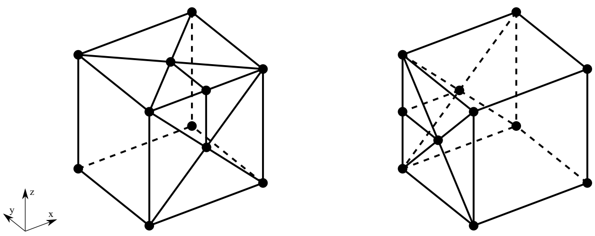
4 Pattern-by-Pattern Matrix-vector Product

The 4096 octree cell patterns can be further transformed into 144 unique cases using rotation and mirroring operators. An example is illustrated in Figure 3. The octree cell in Figure 3b can be transformed into the corresponding master cell in Figure 3a by rotating 90 degrees about the x axis and then rotating 90 degrees about the z axis. The rotation matrix can be constructed as in Figure 3c. Additional cell patterns can be defined for given rules. Some examples of element patterns obtained by trimming through the corner nodes and mid-edge nodes are shown in Figure 4.

In a large-scale problem, many elements of the same patterns but different sizes will exist. Geometrically, these elements are similar and can be obtained by scaling the element of unit size (master-element). The stiffness matrix and other properties of the elements can also be obtained by appropriate scaling. For example, the stiffness matrix of an element is proportional to the edge length, and the mass matrix of an element is proportional to the cube of edge length. When performing the matrix-vector product at the element level, the operations of elements of the same pattern can be grouped and performed using a single matrix-multiplication. For example, the product of the stiffness matrix with the nodal displacement can be expressed as

$$\mathbf{R}_n^{\text{int}} = \sum_{p=1}^{n_p} \mathbf{K}_p^e \begin{bmatrix} \mathbf{U}_{p1}^e & \mathbf{U}_{p2}^e & \cdots & \mathbf{U}_{pk}^e \end{bmatrix}_n \operatorname{diag}\left(\mathbf{S}_p^e\right) \tag{5}$$

where n_p is the number of unique patterns in the octree mesh, $\mathbf{K}_p^e$ is the stiffness matrix of a master element of pattern p with unit size, $\mathbf{U}_{pi}^e$ is the nodal displacement vector of the i-th element of pattern p, and $\operatorname{diag}\left(\mathbf{S}_p^e\right)$ is a diagonal matrix containing the edge



$$\mathbf{Q} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

(a) Octree master cell (b) Octree cell with identical pattern (c) Transformation matrix

Figure 3: Identify unique octree pattern by transformation.

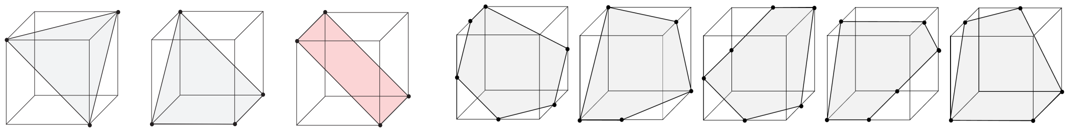


Figure 4: Examples of elements trimming through corner nodes and mid-edge nodes.

lengths of each element of pattern p. Each column obtained from the right-hand side will be assembled into $\mathbf{R}_n^{\mathrm{int}}$ using the element connectivity data.

Since the master-element matrices have a small data size, they can easily fit inside the cache-memory and can be reused during the matrix computation exploiting the temporal locality of the cache. This approach saves a significant amount of time typically required in fetching the individual element matrices from the main-memory. Additionally, the memory requirement is drastically reduced as the storage of individual element matrices is avoided. Therefore, the octree pattern-based approach provides an efficient way of implementing iterative solvers on modern supercomputers usually relying on cache-based hierarchical memory architecture.

5 Numerical Examples

5.1 Implicit dynamics: Seismic wave propagation response of a castle on a mountain [20]

A mountain with a castle at the peak is modelled. The model of the mountain spans a region of $1024\,\mathrm{m} \times 1024\,\mathrm{m} \times 576\,\mathrm{m}$. The height of the castle is $64\,\mathrm{m}$. An octree mesh is automatically generated from an STL model and shown in Figure 5. A pressure load is applied at the base of the region. A temporal Ricker wavelet distribution with the maximum frequency of interest at $f_{\max} = 4.9$ Hz is assumed.

The maximum element size is $16\,\mathrm{m}$ and the minimum element size of $0.0625\,\mathrm{m}$. It contains 638,707 elements and 1,353,006 nodes (4,059,018 DOFs). To consider the large-scale seismic wave propagation in the mountain and the fine-scale dynamic

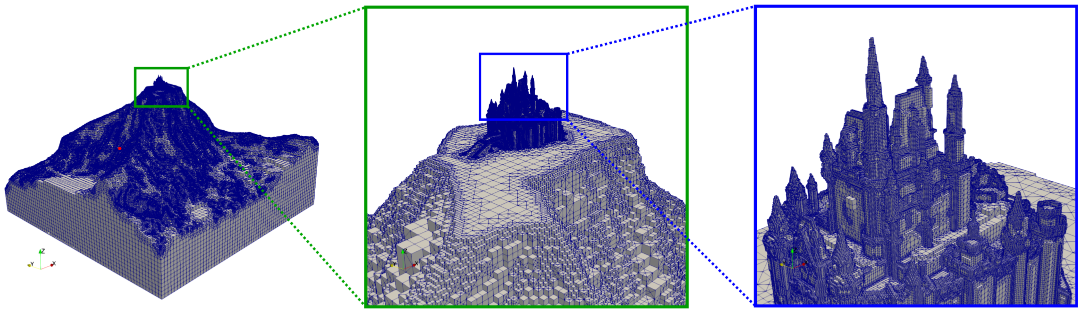


Figure 5: Mesh of a castle on a mountain

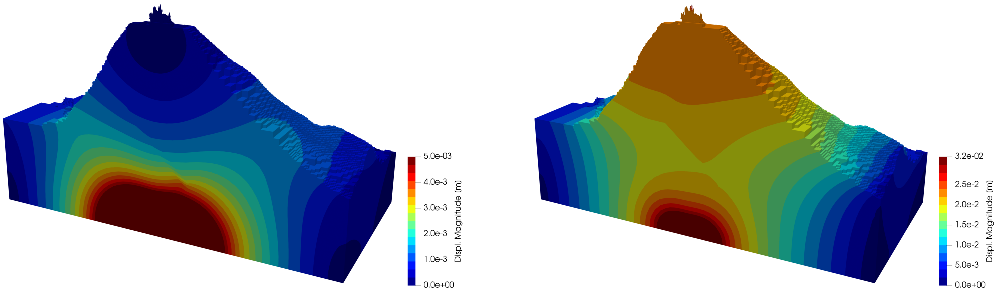


Figure 6: Displacement magnitude contour plots for the castle on a mountain at 0.44 s (left) and 1.04 s (right).

response of the structure in one model, an implicit method, the HHT-α method, is chosen to perform the time integration. The linear equations are solved using the octree pattern-based parallel preconditioned conjugate gradient (PCG) solver. The time step size is chosen as $\Delta t = 2 \times 10^{-2}$ s, allowing 10 time steps per period at the maximum frequency of interest. The contours of the displacement magnitude for half of the mesh are plotted at the time 0.44 s and 1.04 s in Figure 6. It can be seen that as the wave travels from the source to the rest of the region, diffracted by a fault under the mountain.

5.2 Explicit dynamics: Nonlocal damage analysis of concrete specimen [21]

A 3D image of a concrete specimen with the resolution 1024×1024×1024 (∼1.07 billion voxels) is obtained using XCT-scan as shown in Figure 7. Each voxel in the image has a side length 70 μm. The specimen has a cubic shape with side length approximately L =72 mm. The specimen is segmented into aggregates, mortar, voids, and interface transition zones between the aggregates and mortar.

A modified exponential model is used for the damage. The internal length $l_0 = 0.14\,\mathrm{mm}$ is chosen to consider the non-local effects. The left surface of the specimen

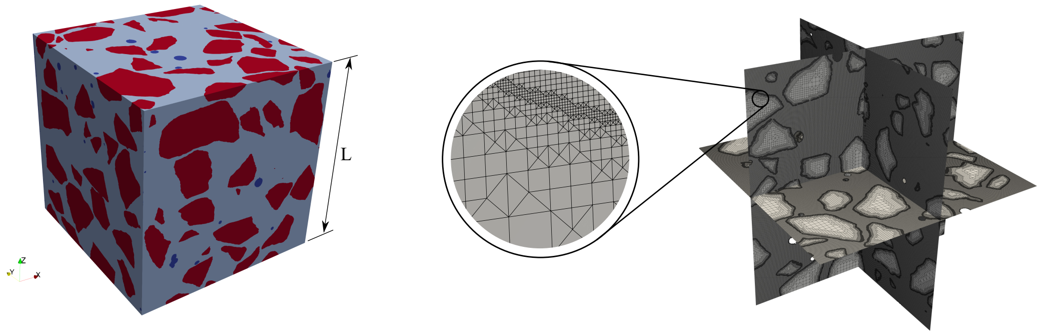


Figure 7: 3D image of concrete specimen and octree mesh

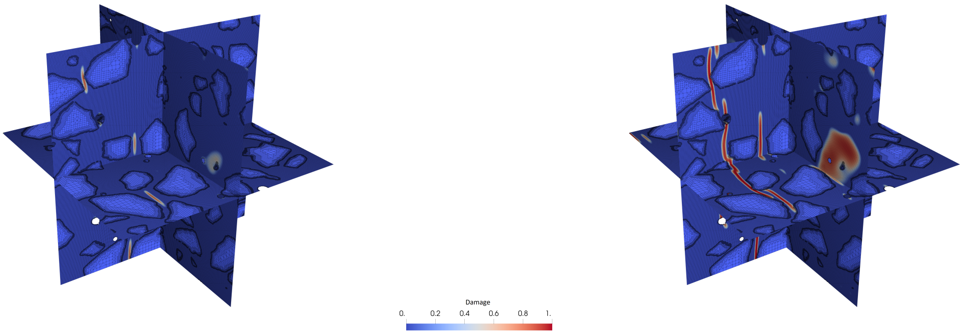


Figure 8: Damage contours plotted on mid-plane slices of the concrete specimen at $t = 38.5\,\mu$s(left) and $t = 60.9\,\mu$s(right).

is constrained against displacement in its normal direction. A uniformly distributed traction P (tensile) is suddenly applied on the opposite surface. The explicit dynamic analysis is carried out for about 61 μs, with the stable time-step size computed as 12.8 ns and the total number of time-steps used is 4800.

The various stages of the damage evolution in the concrete specimen sliced along the mutually perpendicular mid-planes are shown in Figure 8. It is observed that the damage process zones (DPZs) initiate at multiple locations; however, they are mostly located near voids (where stress concentration exists) and ITZs (weaker material) in the specimen. As the loading continues, the DPZs in the specimen begin to grow and coalesce leading to the formation of macro cracks.

5.3 Explicit dynamics: Wave propagation in a sandwich panel [22]

A sandwich panel with two steel cover sheets and a foam-like aluminum core is modelled. The digital image, shown in Figure 9, is obtained by X-ray CT scans. The size of the panel is $288\,\mathrm{mm} \times 288\,\mathrm{mm} \times 57.6\,\mathrm{mm}$, while the thickness of the cover sheet is $4.8\,\mathrm{mm}$. An octree mesh is generated with a maximum element size of $0.96\,\mathrm{mm}$ and a minimum element size of $0.24\,\mathrm{mm}$. The panel is mirrored in the x and y directions

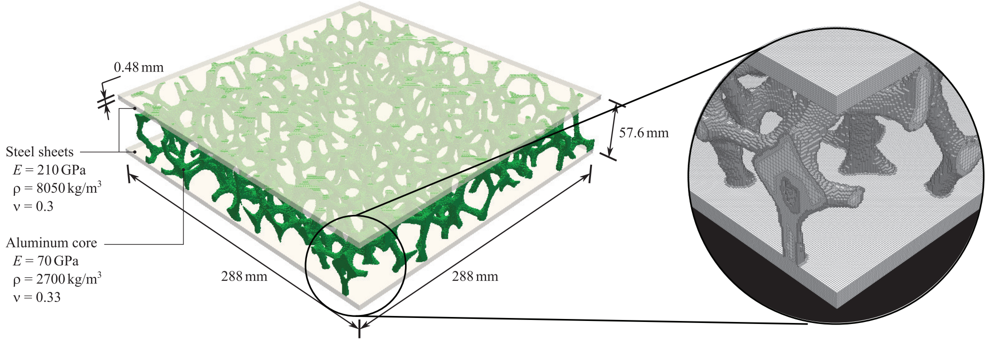
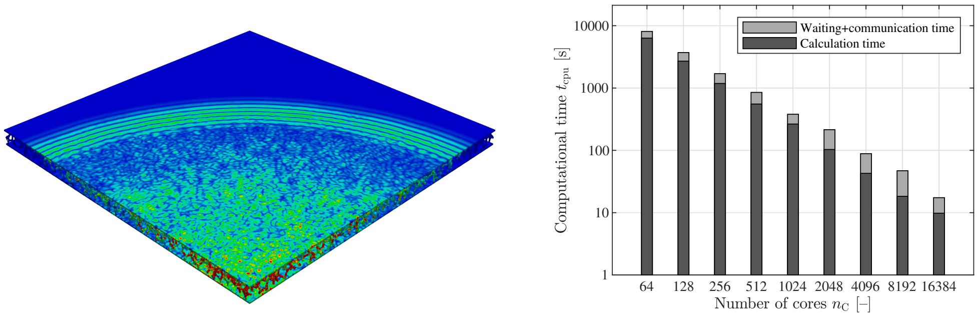


Figure 9: Geometry and mesh of the sandwich panel



(a) Displacement contour at $t = 0.2\,\text{ms}$

(b) Computational time vs. number of cores

Figure 10: Wave propagation in a sandwich panel.

twice. Hence, the edge length of the panel becomes $1152\,\text{mm}$. There are 216,323,104 elements and 1,064,602,902 DOFs in the octree mesh of the panel.

A sine-burst excitation is applied at one corner of the plate. The Lamb wave propagation in the panel is simulated using an explicit central difference method for a total duration of $0.4\,\text{ms}$. The time step size $\Delta t = 2.98 \times 10^{-5}\,\text{ms}$ is used. The magnitude of the displacement responses at two time instances are shown in Figure 10a. The speedup of the multiple cores is illustrated in Figure 10b.

When 100% efficiency is assumed at 64 cores, it is observed that a speedup factor of 29,760 is obtained with 16,384 cores. This super-linear speedup is caused by the cache effect, i.e., as the mesh is partitioned and distributed to more cores, more data can be fit into the cache memory of the compute nodes, which has significantly higher data processing speed than RAM.

6 Concluding remarks

A scaled boundary finite element framework is established for fully automated analysis on high-performance computing systems. The salient features of this approach include:

1. The scaled boundary finite element method allows the construction of star-convex polyhedral elements. An element may have an arbitrary number of faces, edges and nodes.
2. An efficient octree algorithm is utilized for automatic mesh generation. The same algorithm can handle not only traditional CAD models but also STL models, digital images, and point cloud data. The issue of hanging nodes is maturely resolved using the polyhedral elements.
3. A pattern-by-pattern algorithm for performing matrix-vector products is developed. It reduces the amount of memory and memory access in the analysis and improves the computational efficiency on HPC systems.

The scaled boundary finite element framework incorporating fully automated mesh generation and high-performance computing implementation is capable of handling large-scale problems with complex geometry, as demonstrated by examples.

References

[1] K. J. Bathe, Finite element procedures, 2nd Edition, Prentice Hall, Pearson Education, Inc., 2014.

[2] A. Saputra, H. Talebi, D. Tran, C. Birk, C. Song, Automatic image-based stress analysis by the scaled boundary finite element method, International Journal for Numerical Methods in Engineering 109 (2017) 697–738.

[3] M. Berger, A. Tagliasacchi, L. Seversky, P. Alliez, G. Guennebaud, J. Levine, A. Sharf, C. Silva, A survey of surface reconstruction from point clouds, Computer Graphics Forum 36 (2017) 301–329.

[4] P. Yadav, K. Suresh, Large scale finite element analysis via assembly-free deflated conjugate gradient, Journal of Computing and Information Science in Engineering 14 (4) (2014).

[5] Z. J. Yang, F. Yao, E. T. Ooi, X. W. Chen, A scaled boundary finite element formulation for dynamic elastoplastic analysis, International Journal for Numerical Methods in Engineering 120 (4) (2019) 517–536.

[6] D. Zou, K. Chen, X. Kong, J. Liu, An enhanced octree polyhedral scaled boundary finite element method and its applications in structure analysis, Engineering Analysis with Boundary Elements 84 (2017) 87 – 107.

[7] Y.-M. Jia, C.-J. Li, Y. Zhang, J. Chen, The high-order completeness analysis of the scaled boundary finite element method, Computer Methods in Applied Mechanics and Engineering 362 (2020) 112867.

[8] B. Yu, P. Hu, A. A. Saputra, Y. Gu, The scaled boundary finite element method based on the hybrid quadtree mesh for solving transient heat conduction problems, Applied Mathematical Modelling 89 (2021) 541–571.

[9] K. O. Coelho, P. R. Devloo, S. M. Gomes, Error estimates for the scaled boundary finite element method, Computer Methods in Applied Mechanics and Engineering 379 (2021) 113765.

[10] C. Wang, Y. He, H. Yang, Adaptive recursive scaled boundary finite element method for forward and inverse interval viscoelastic analysis, Engineering Analysis with Boundary Elements 128 (2021) 257–273.

[11] S. Pfeil, H. Gravenkamp, F. Duvigneau, E. Woschke, Scaled boundary finite element method for hydrodynamic bearings in rotordynamic simulations, International Journal of Mechanical Sciences 199 (2021) 106427.

[12] Z. Y. Li, Z. Q. Hu, G. Lin, J. B. Li, A scaled boundary finite element method procedure for arch dam-water-foundation rock interaction in complex layered half-space, Computers and Geotechnics 141 (2022) 104524.

[13] S. Jiang, C. Wan, L. Sun, C. Du, Flaw classification and detection in thin-plate structures based on scaled boundary finite element method and deep learning, International Journal for Numerical Methods in Engineering 123 (19) (2022) 4674–4701.

[14] R. Reichel, S. Klinkel, A non-uniform rational b-splines enhanced finite element formulation based on the scaled boundary parameterization for the analysis of heterogeneous solids, International Journal for Numerical Methods in Engineering 124 (9) (2023) 2068–2092.

[15] J. Zhang, P. Wang, M. Zhao, L. Liu, Y. Qu, X. Du, A scaled boundary finite element method for soil dynamic impedance of pile groups using hybrid quadtree mesh considering horizontal vibration, Engineering Analysis with Boundary Elements 153 (2023) 226–241.

[16] Q. Zang, S. P. Bordas, J. Liu, S. Natarajan, NURBS-enhanced polygonal scaled boundary finite element method for heat diffusion in anisotropic media with internal heat sources, Engineering Analysis with Boundary Elements 148 (2023) 279–292.

[17] D. Lozano, J. Bulling, A. Asokkumar, H. Gravenkamp, C. Birk, 3D simulations of ultrasonic waves in plates using the scaled boundary finite element method and high-order transition elements, Wave Motion 120 (2023) 103158.

[18] C. Song, The Scaled Boundary Finite Element Method: Introduction to Theory and Implementation, Wiley, 2018.

[19] B. Xiao, S. Natarajan, C. Birk, E. H. Ooi, C. Song, E. Ooi, Construction of generalized shape functions over arbitrary polytopes based on scaled boundary finite element method's solution of Poisson's equation, International Journal for Numerical Methods in Engineering (2023) 1– 34.

[20] A. Ankit, J. Zhang, S. Eisenträger, C. Song, An octree pattern-based massively parallel PCG solver for elasto-static and dynamic problems, Computer Methods in Applied Mechanics and Engineering 404 (2023) 115779.

[21] A. Ankit, C. Song, S. Eisenträger, S. Zhang, E. Hamed, Dynamic non-local damage analysis using an octree pattern-based massively parallel explicit solver, Computer Methods in Applied Mechanics and Engineering 400 (2022) 115598.

[22] J. Zhang, A. Ankit, H. Gravenkamp, S. Eisenträger, C. Song, A massively parallel explicit solver for elasto-dynamic problems exploiting octree meshes, Computer Methods in Applied Mechanics and Engineering 380 (2021) 113811.