The finite cell method for three-dimensional problems of solid mechanics

A. Düster a, J. Parvizian b, Z. Yang a, E. Rank a

a Computational Engineering, Faculty of Civil Engineering and Geodesy, Technische Universität München, Arcistr. 21, 80233 München, Germany
b Isfahan University of Technology, Isfahan 84156 83111, Iran

ABSTRACT

This article presents a generalization of the recently proposed finite cell method to three-dimensional problems of linear elasticity. The finite cell method combines ideas from embedding or fictitious domain methods with the p-version of the finite element method. Besides supporting a fast, simple generation of meshes it also provides high convergence rates. Mesh generation for a boundary representation of solids and for voxel-based data obtained from CT scans is addressed in detail. In addition, the implementation of non-homogeneous Neumann boundary conditions and the computation of cell matrices based on a composed integration is presented. The performance of the proposed method is demonstrated by three numerical examples, including the elastostatic computation of a bone biopsy.

© 2008 Elsevier B.V. All rights reserved.

1. Introduction

The finite cell method (FCM), which was recently proposed by the authors [1] can be interpreted as a combination of a fictitious or embedding domain approach with high-order finite element methods. It therefore combines the fast, simple generation of meshes with high convergence rates. In fictitious domain methods the original or physical domain is embedded in a geometrically larger domain of a simpler shape. Thanks to the simple geometry of the embedding domain it can be readily discretized with structured or Cartesian grids. Different discretization methods can be applied, ranging from finite difference to finite volume and finite element methods. The name “fictitious domain method” was coined by Saulev [2,3] in the early sixties. Since then the fictitious domain method has been further developed and applied to model problems arising in different areas of computational mechanics. For an overview of the huge body of literature, please refer to [4–6].

Perhaps the most relevant work on fictitious domain methods with regard to the current paper was published by Bishop [7] and Ramière et al. [5]. Bishop has used an implicit meshing for two-dimensional problems of linear elasticity to discretize the embedding domain. An algorithm to integrate the weak form exactly is suggested in order to account for elements which are cut by the physical boundary. It is shown that bi-cubic Hermite elements yield more accurate and efficient results of displacements and stresses than bi-quadratic Lagrange elements. This could provide a clue for increasing the order of approximation space for better results. Increase in accuracy from quadratic Lagrange elements, that are C1-continuous, to Hermite cubics, that are C2-continuous, may also be attributed to the increase in the level of continuity or smoothness of the underlying discretization. For the role of continuity in the discretization of solids and fluids in the context of isogeometric analysis the reader is referred to [8–10]. Despite interesting achievements of Bishop [7], the algorithm to integrate the weak form exactly is likely to become very expensive for three-dimensional problems. Since the volume integrals are converted to boundary integrals by means of the divergence theorem, the approach is restricted to element-wise constant data, which limits the approach to problems with homogeneous and linear material or precludes the application of high-order shape functions for nonlinear and inhomogeneous materials.

For Ramière et al., the core idea is again to immerse the original domain into a simpler, geometrically larger one. Both finite volume and bi-linear finite element methods are used to solve elliptic problems with general boundary conditions. Different methods for treating the boundary conditions are discussed. The literature provides a wide scope of ideas and techniques for imposing boundary conditions which are similar in nature but go by different names. For a review of such techniques the reader is referred to [4].

In the finite cell method, as proposed by the authors [1], the idea is to use an easily discretized domain in which the physical domain is embedded. Therefore, as in all similar methods that
are not based on a boundary-conforming mesh, the accuracy in discretizing the domain is replaced by an accurate integration scheme. Assuming a soft material which fills the void regions of the embedding domain makes a standard finite element discretization possible. However, the fast convergence to an accurate result is due to the fact that a high-order Ansatz space is used [11, 12]. A dense distribution of integration points serves both to capture the boundary, as in level sets [13], and to increase the accuracy of integration over cells that are independent of the physical domain.

The structure of the paper is as follows: Section 2 summarizes the finite cell formulation for three-dimensional problems of linear elastostatics. This includes the variational formulation, the implementation of boundary conditions and the computation of cell matrices. Section 3 discusses the generation of meshes for different types of geometric models. The performance of the proposed method is presented in Section 4 using three numerical examples.

2. The finite cell method

We closely follow the description in [1] to explain the finite cell method. For better clarity and understanding, the figures are presented in two dimensions but the formulation applies similarly in three dimensions, as demonstrated in the examples.

2.1. Variational formulation

Let us assume that on a three-dimensional physical domain \( \Omega \), a problem of linear elasticity is described by the weak form of equilibrium as

\[
\mathcal{A}(u, v) = \mathcal{F}(v),
\]

where the bilinear part is

\[
\mathcal{A}(u, v) = \int_\Omega [L v]^T C [L u] d\Omega,
\]

in which \( u \) is the displacement, \( v \) is the test function, \( L \) is the standard strain-displacement operator and \( C \) is the elasticity matrix. Note that the principles of the method are not restricted to linear differential operators. Without loss of generality we assume homogeneous Dirichlet boundary conditions \( u = 0 \) along \( \Gamma_D \) and a Neumann boundary \( \Gamma_N \) with prescribed tractions, \( \Gamma_D \cap \Gamma_N = \emptyset \). The linear functional

\[
\mathcal{F}(v) = \int_\Omega \nu^T \tilde{f} d\Omega + \int_{\Gamma_N} \tilde{t}^T \tilde{t} d\Gamma
\]

takes the volume loads \( \tilde{f} \) and prescribed tractions \( \tilde{t} \) into account.

The original physical domain can now be embedded in the domain \( \Omega_e \) with the boundary \( \partial \Omega_e \). For the sake of simplicity, the situation is depicted for a two-dimensional case in Fig. 1. The interface between \( \Omega \) and the embedding domain is defined as \( \Gamma_i = \partial \Omega \cap \partial \Omega_e \). Following Nettmitt and Tiba [14], the displacement variable is extended as

\[
u = \begin{cases} u^1 & \text{in } \Omega_e \setminus \Omega, \\ u^2 & \text{in } \Omega \setminus \Omega_e \end{cases}
\]

while the transition conditions guarantee continuity at the interface between \( \Omega \) and \( \Omega_e \setminus \Omega \):

\[
\begin{align*}
\mathbf{u}^1 &= \mathbf{u}^2 & \text{on } \Gamma_i, \\
\mathbf{t}^1 &= \mathbf{t}^2 & \text{on } \Gamma_i.
\end{align*}
\]

Boundary conditions are set for \( \Gamma_e \):

\[
\begin{align*}
\mathbf{u} &= 0 & \text{on } \Gamma_{e,D}, \\
\mathbf{t} &= 0 & \text{on } \Gamma_{e,N}.
\end{align*}
\]

in which \( \Gamma_{e,D} \) and \( \Gamma_{e,N} \) are the Dirichlet and Neumann boundaries of \( \Omega_e \) respectively, \( \Gamma_{e,D} = \Gamma_D \cup \Gamma_{e,N} \), and \( \Gamma_{e,D} \cap \Gamma_{e,N} = \emptyset \). The first condition in (6) is generally necessary to avoid rigid body motion. The weak form of the equilibrium equation for the embedding domain \( \Omega_e \) is given as

\[
\mathcal{A}_e(u, v) = \mathcal{F}_e(v),
\]

where the bilinear form is

\[
\mathcal{A}_e(u, v) = \int_{\Omega_e} [L v]^T C [L u] d\Omega,
\]

in which

\[
C_e = \varepsilon C
\]

is the elasticity matrix of the embedding domain, with

\[
\varepsilon(x) = \begin{cases} 1.0 & \forall x \in \Omega, \\ 0.0 & \forall x \in \Omega_e \setminus \Omega. \end{cases}
\]

Inserting (9) and (10) into (8), the bilinear functional turns to

\[
\mathcal{A}_e(u, v) = \int_{\Omega_e} [L v]^T C [L u] d\Omega = \int_{\Omega} [L v]^T C [L u] d\Omega + \int_{\Omega_e \setminus \Omega} [L v]^T C [L u] d\Omega = \mathcal{A}(u, v).
\]

The linear functional

\[
\mathcal{F}_e(v) = \int_{\Omega} \nu^T \tilde{f} d\Omega + \int_{\Gamma_N} \tilde{t}^T \tilde{t} d\Gamma + \int_{\Gamma_{e,N}} \tilde{t}^T d\Gamma
\]

considers the volume loads \( \tilde{f} \), prescribed traction along \( \Gamma_{e,N} \) interior to \( \Omega_e \), and prescribed traction at the boundary of the embedding domain. Due to Eq. (6), the last term in (12) can be assumed 0.

The embedding domain is now discretized in a mesh which is independent of the original domain. These "finite elements" of the embedding domain do not necessarily fulfill the usual geometric properties of elements for the original domain \( \Omega \), as they may be intersected by \( \partial \Omega \). To distinguish them from classical elements they will be called finite cells. It is simpler and more advantageous to initially assume cells to be rectangular hexahedrals (cuboids) resulting in a constant Jacobian matrix of the cell-wise mapping. Fig. 2 illustrates the situation for a two-dimensional setting. The union of all \( n_c \) cells forms the embedding domain

\[ \Omega_e = \bigcup_{c=1}^{n_c} \Omega_c. \]

Fig. 1. The domain \( \Omega \) is embedded in \( \Omega_e \).
where $\Omega_e$ is the domain represented by a cell. At the discretized level, the weak form (8) turns into

$$ \mathcal{M}_e(\mathbf{u}, \mathbf{v}) = \sum_{e=1}^{N_e} \int_{\Omega_e} [\mathbf{B}]^T [\mathbf{C}] [\mathbf{B}] \mathbf{u} \mathbf{v} \, d\Omega. $$

(14)

In each cell the displacement variable is approximated as

$$ \mathbf{u} = \mathbf{N} \mathbf{U}. $$

(15)

in which $\mathbf{N}$ denotes the matrix of shape functions and $\mathbf{U}$ is the vector of unknowns. The product $\mathbf{LN}$ is usually denoted as the standard strain–displacement matrix $\mathbf{B}$ which will be used in Section 2.3. Our implementation, [15], uses hexahedrals with hierarchical shape functions based on integrated Legendre polynomials [11,16,12]. This makes it possible to perform h-extension for both low and higher polynomial orders through the refinement of cells and a p-extension on a fixed mesh of (coarse) cells. Based on the Bubnov–Galerkin approach, $\mathbf{v} = \mathbf{N} \mathbf{v}$, inserting (15) into (14) produces a finite cell formulation of

$$ \mathbf{K} \mathbf{U} = \mathbf{F}, $$

(16)

where $\mathbf{K}$, the global stiffness matrix, and $\mathbf{F}$ the global load vector are obtained by assembling all cell matrices

$$ \mathbf{K} = A_e^{\eta} \mathbf{K}_e, $$

$$ \mathbf{F} = A_e^{\eta} \mathbf{F}_e. $$

(17)

(18)

The similarity between the finite cell formulation and the finite element formulation, as in [17,11] for example, exploits most of the solution methods and techniques of the latter method to the full.

2.2. Boundary conditions

In the finite cell, fictitious or embedding domain methods, the difficulty of generating meshes for complicated geometric models shifts to the problem of implementing boundary conditions. The difficulty arises when boundary conditions need to be defined on surfaces which do not conform to the mesh embedding the original domain $\Omega$. Boundary conditions can either be implemented by

penalization or Lagrangian methods can be applied. For an overview of existing methods without using Lagrange multipliers, the reader is referred to the recent work of Ramière et al. [5]. As regards Lagrange multipliers for the fictitious domain method, please refer to the recent review article on this subject presented by Glowinski and Kuznetsov [6].

2.2.1. Neumann boundary conditions

Let us first consider homogeneous Neumann conditions. This "zero traction condition” is equivalent to assuming material with zero stiffness in the embedding domain. Inhomogeneous Neumann boundary conditions can be realized by explicitly including the second term in (12), i.e. by integrating over the boundary $\Gamma_{N}$ which is on the interior of $\Omega$. The situation is depicted in Fig. 3 for a two-dimensional setting. Section 2.3 addresses the implementation for three-dimensional discretizations based on hexahedral elements in more detail.

2.2.2. Dirichlet boundary conditions

Homogeneous Dirichlet conditions can be approximated by assuming a "stiff strip" of material from $\Gamma_D$ up to the Dirichlet boundary $\Gamma_{e,D}$ of the embedding domain. In Fig. 4 the application of homogeneous Dirichlet boundary conditions is explained for a two-dimensional case, where the grey shaded area corresponds to a stiff strip of material. In [1] two-dimensional, numerical experiments have shown that $\epsilon = \epsilon(10)$, i.e. assuming a material that is one order of magnitude stiffer than in the original domain of computation produces reasonably accurate results.

A natural extension of the "stiff strip” approach to inhomogeneous Dirichlet boundary conditions is to prescribe the inhomogeneous displacement at the boundary $\Gamma_{D}$. In the literature several ways to impose Dirichlet boundary conditions have been proposed. An overview of the different methods for second order elliptic boundary value problems solved by the fictitious domain method can be found in [5,6].
2.3. Computation of cell matrices

The approximation of the original problem (2) over the domain \( \Omega \) has been replaced by a problem over an embedding domain \( \overline{\Omega} \), yet with discontinuous coefficients. Therefore the integrand in (14) may be discontinuous within cells that are cut by the boundary \( \partial \Omega \). The Gauss quadrature usually applied in finite element computations converges very fast for smooth integrands but shows a poor convergence behaviour for non-smooth functions. Therefore, an adapted integration scheme is necessary to capture the discontinuity of the integrand. We accordingly apply a composed integration scheme for cells intersected by the boundary.

Our implementation is based on a hexahedral element applying hierarchic high-order shape functions [11, 16, 12]. The mapping from the standard element to the global coordinate system is defined as

\[
x = \mathbf{Q}_c(\xi, \eta, \zeta) = \sum_{i=1}^{8} N_i(\xi, \eta, \zeta) \mathbf{x}_i,
\]

where \( \mathbf{x}_i = (x_i, y_i, z_i)^T \) denote the global coordinates of the eight nodes and

\[
N_i(\xi, \eta, \zeta) = \frac{1}{8} (1 + \xi_i)(1 + \eta_i)(1 + \zeta_i), \quad i = 1, \ldots, 8
\]

are the well-known tri-linear shape functions, where \((\xi_i, \eta_i, \zeta_i)\) are the local coordinates of the \(i\)th node. The discretization of the embedding domain is carried out with rectangular hexahedrals so the mapping reduces to

\[
x = \mathbf{Q}_c(\xi, \eta, \zeta) = \begin{bmatrix}
X_1 + \frac{1}{2}(1 + \xi)h_x \\
Y_1 + \frac{1}{2}(1 + \eta)h_y \\
Z_1 + \frac{1}{2}(1 + \zeta)h_z
\end{bmatrix}
\]

resulting in a constant Jacobian matrix

\[
J_c = \text{grad}^T \mathbf{Q}_c(\xi, \eta, \zeta) = \frac{1}{2} \begin{bmatrix}
h_x & 0 & 0 \\
0 & h_y & 0 \\
0 & 0 & h_z
\end{bmatrix}
\]

In (21) and (22) \( h_x, h_y, h_z \) denote the cell size with respect to the \( x, y, \) and \( z \) direction, respectively. Since the Jacobian matrix is constant, the hierarchic shape functions defined on the standard element which are mapped on the global coordinate system remain polynomials. If the material properties are constant and the cells are completely inside the domain \( \Omega \), i.e., are not intersected by the boundary \( \partial \Omega \), \( \mathbf{x} = \mathbf{r} \) within the cell and the stiffness matrix

\[
K_c = \int_{\Omega} \int_{\Omega} \int_{\Omega} B_i^T(\mathbf{r}) \mathbf{b}(\mathbf{r}) B_i(\mathbf{r}) \text{det} J_c \, d\mathbf{r} \, d\mathbf{r} \, d\mathbf{r}
\]

can be integrated exactly. Assuming a cell with hierarchic shape functions of degree \( p \) the integrand includes polynomials up to the order \( 2p \). Since a Gaussian integration with \( n \) points (in one direction) integrates polynomials of degree \( 2n - 1 \) exactly, \( n = p + 1 \) Gaussian integration points (in each direction) are sufficient to exactly integrate the stiffness matrix. Performing a linear elastic analysis, the stiffness matrix can only be computed in terms of the unknown element size and elasticity coefficients of the isotropic linear elastic material law. This computation can be carried out beforehand and the stiffness matrix can be tabulated, which significantly reduces the computational time. Since we apply a hierarchic set of shape functions, the stiffness matrix of degree \( p - 1 \) is a sub-matrix of the stiffness matrix corresponding to polynomial order \( p \). The hierarchy can be exploited when performing a p-extension.

If a cell happens to be cut by the boundary \( \partial \Omega \) then the integrand will be discontinuous and a standard Gaussian integration will not perform well. Therefore, we apply a composed integration, where the cell to be integrated is divided into \( n_{\text{sc}} \) sub-cells, see Fig. 5. These sub-cells are introduced for integration purposes only. In order to establish a relation between the coordinates of the block-shaped sub-cell \( \mathbf{r} = (r, s, t)^T \) and the cell, a linear mapping function

\[
\xi = \mathbf{Q}_c^e(r, s, t) = \begin{bmatrix}
\xi_1 + \frac{1}{2}(1 + r)h_x \\
\eta_1 + \frac{1}{2}(1 + s)h_y \\
\zeta_1 + \frac{1}{2}(1 + t)h_z
\end{bmatrix}
\]

is applied, where \((\xi_1, \eta_1, \zeta_1)\) are the local coordinates of the anchor point of the sub-cell and \( h_x, h_y, h_z \) denote the size of the sub-cell. Since the sub-cells are block-shaped, the mapping results in a constant Jacobian matrix

\[
J_c^e = \text{grad}^T \mathbf{Q}_c^e(r, s, t) = \frac{1}{2} \begin{bmatrix}
h_x & 0 & 0 \\
0 & h_y & 0 \\
0 & 0 & h_z
\end{bmatrix}
\]

The stiffness matrix of cell \( e \) is obtained by carrying out the composed integration over the \( n_{\text{sc}} \) sub-cells

\[
K_e = \sum_{i=1}^{n_{\text{sc}}} \int_{\Omega} \int_{\Omega} \int_{\Omega} B_i^T(\xi)(\mathbf{r}) \mathbf{b}(\xi(\mathbf{r})) \mathbf{B}_i(\xi(\mathbf{r})) \text{det} J_c^e \, d\mathbf{r} \, d\mathbf{r} \, d\mathbf{r}
\]

In (26) it is also necessary to account for the determinant of the Jacobian matrix \( J_c^e \) which is due to the change of variables. The shape functions are the same as in (23), however, the mapping \( \xi(\mathbf{r}) = \mathbf{Q}_c^e(r, s, t) \) has to be applied to establish the relation between the coordinates of the sub-cell \((r, s, t)\) and the cell \((\xi, \eta, \zeta)\). The sub-cells can either be created so as to have a uniform spacing or an adaptive algorithm based on an octree data structure can be applied instead. The adaptive algorithm may be applied to automatically control the refinement of the sub-cells, so that in those regions where the element exhibits a strong variation (e.g., jumps due to the incorporation of the domain's boundary) a more accurate quadrature is performed. Numerical experiments [18] in two dimensions that apply an adaptive quadtree technique have demonstrated that the adaptive scheme is superior to a uniform refinement. A different quadrature scheme may be chosen for each of the sub-cells. In the numerical examples presented in Section 4 a Gaussian quadrature with \( n \) integration points in each direction \((r, s, t)\) of the sub-cell is applied, for the sake of simplicity. In cases where a cell intersects only a small part of the domain, the condition of the global stiffness matrix (17) deteriorates. Therefore, the application of a standard preconditioned conjugate gradient method might yield problems. In order to overcome these problems we apply the direct solver...
Scholes [19] based on a LU factorization which turned out to be a very robust solution approach.

The cell load vector is composed of two parts
\[ \mathbf{F}_c = \mathbf{F}_{c1} + \mathbf{F}_{c2}, \]  
(27)

where \( \mathbf{F}_{c1} \) is due to a volume load and \( \mathbf{F}_{c2} \) results from a surface traction. The volume load vector \( \mathbf{F}_{c1} \) can be computed in a similar way as the stiffness matrix. To account for non-homogeneous Neumann boundary conditions defined on \( \Gamma_N \), the boundary integral has to be computed cell-wise to determine the load vector \( \mathbf{F}_{c2} \) due to a surface traction \( \mathbf{t} \) acting on \( \Gamma_N \). Since the boundary \( \Gamma_N \) will not normally coincide with the boundary of the embedding domain \( \Omega_e \), the computation of the cell load vector cannot be obtained by simply integrating over the boundary of the cells, since \( \Gamma_N \) will generally intersect the cells.

In order to compute the load vector we need to have a parametric description of the surface on which the traction is acting in order to perform the integration. Fig. 6 depicts a triangulated surface intersecting a hexahedral element. The area spanned by the \( n \) facets can be computed as
\[ A = \frac{1}{2} \sum_{j=1}^{n} \int_{a}^{b} x_j(u, v) \times x_{j+1}(u, v) \, du \, dv, \]  
(28)

with \( x_{j+1} = \mathbf{x}_{j+1}, x_j = \mathbf{x}_j \) where
\[ x_j = \mathbf{Q}(u, v) = \sum_{i=1}^{4} N_i(u, v) \mathbf{X}_i \]  
(29)
denotes the mapping from the local coordinates \((u, v)\) of the standard quadrilateral element \([-1, 1] \times [1, 1]\) to the global coordinates \((x, y, z)\) of the triangle. 
\( x_j = (X_j^1, X_j^2, X_j^3)^T \) denote the global coordinates of the three nodes of the triangle. The mapping function \( \mathbf{Q}(u, v) \) transforms the standard quadrilateral domain to fit the (triangular) facet, since we have chosen \( x_j = \mathbf{x}_j \).

In Eq. (29)
\[ N_i(u, v) = \frac{1}{4} (1 + u_i u)(1 + v_i v) \]  
(30)
are the bi-linear shape functions, where \((u_i, v_i)\) denote the local coordinates of the \(i\)th node of the standard quadrilateral. Applying the mapping function (29) makes it possible to perform the integration with a two-dimensional Gaussian quadrature defined on a standard quadrilateral. Although this is not as efficient as applying integration techniques developed especially for triangles, this drawback is negligible since the two-dimensional integration does not dominate the overall computational effort. It should be borne in mind that the process of triangulating a curved surface introduces a discretization error. Commercial CAD tools are designed to control this discretization error, i.e. the deviation from the (true) curved surface, by specifying the maximum chordal deviation tolerance of the facets.

To compute the cell load vector based on a traction \( \mathbf{t} \) acting on \( \Gamma_N \) we need to integrate the discrete form of the second term in (12):
\[ \mathbf{F}_{c2} = \sum_{j=1}^{n} \int_{a}^{b} N_j^T(u, v) [\mathbf{x}_j(u, v) \times \mathbf{x}_{j+1}(u, v)] \, du \, dv. \]  
(31)

Although in the case of plane facets the area can be computed precisely without any numerical integration, we apply the Gaussian quadrature. This is because the integrand is more complex, since it represents the discrete form of the inner product of the test functions with the traction vector. There are various accurate ways to integrate the cell load vector numerically. One involves the use of a very fine surface mesh consisting of many facets to perform a low-order integration by assuming that the integrand is constant over the whole of the facet. This procedure is applied when a fine triangular mesh is needed to obtain a sufficiently accurate description of the curved boundaries. However, there are situations where the surface is plane and a very coarse triangular mesh suffices to describe the geometry. In this case, the coarse triangulation would not provide an accurate integration of the high-order shape functions \( N_j^T \). We therefore apply a Gaussian quadrature on each of the facets to ensure that the integration error of the cell load vector can be controlled independently of the resolution needed for the discretization of geometry.

As can be seen from Fig. 6 it is very likely that the Gaussian points related to one facet fall within different cells (hexahedrals). It is accordingly important to ensure that, during the integration over each facet, the contribution of this integration point refers to the cell where the integration point is located. This means that the corresponding cell number \( c \) has to be determined for each integration point. Since a Cartesian grid of hexahedral cells is applied to discretize the embedding domain \( \Omega_e \), this cell number can be computed very efficiently. For each integration point, the global coordinates are computed from the mapping function \( \mathbf{x}_j = \mathbf{Q}(u, v) \) defined by the facet. Based on the global coordinates the corresponding hexahedral can be readily determined, since a structured cell arrangement with fixed spacing is applied. Having found the cell \( c \) which contains the present Gaussian point, the local coordinates \((\xi, \eta, \zeta)\) of the cell have to be determined in order to compute the shape function matrix \( N_j^T(\xi) \). It is also possible to compute the inverse mapping \( \xi = \mathbf{Q}_j^{-1}(\mathbf{x}_j(u, v)) \), very efficiently from global to local cell coordinates, since the mapping (21) is linear and can be inverted analytically.
3. Geometric models and grid generation for the FCM

The main advantage of the finite cell and embedding or fictitious domain methods in general is their extremely fast and simple grid generation. Since the Cartesian grids applied in the analysis do not need to be aligned to curved boundaries, the meshing process is straightforward. A structured mesh with a resolution of \( n_x \times n_y \times n_z \) cells in the \( x, y \) and \( z \) direction is created and cells which are completely outside the domain \( \Omega \) are disregarded. The starting point of a finite cell computation is a geometric model where different types can naturally be considered. We shall now proceed to discuss three different ways to represent the geometry and briefly explain how these geometric models are accounted for in the FCM. We will show computational examples, which nevertheless concentrate on the geometric aspects and not on the quality of the approximation, which will be in the center of Section 4.

3.1. Implicit representation of geometry

In our previous work on the two-dimensional FCM [1] we considered geometrical models given by an implicit representation. In the first example presented in Section 4.1, the circular hole of a perforated plate is also defined by an implicit representation, i.e., \( (x - b)^2 + y^2 - R^2 = 0 \). The quadrature points of cells cut by the (circular) boundary are tested to see whether they happen to lie in the hole. The geometry is accounted for during the quadrature of the element matrices as a zero extension of material. To this end, an accurate integration technique is needed, as described in detail in Section 2.3. Implicit representation is, however, rather an academic way of defining the geometry and only employed to investigate the accuracy of the FCM numerically. More realistic problems can be dealt with using one of the approaches described in the two subsequent Sections 3.2 and 3.3.

3.2. Voxel model derived from B-rep representation

The geometrical definition of solids by boundary representation (B-rep) is probably the most frequently applied approach in CAD systems. A very common and simple method to describe curved boundaries of solids is based on triangulation. There are lots of CAD tools which support a fully automatic triangulation of a curved surface. The STL format describes unstructured triangulated surfaces bounding a solid by means of the unit normal and vertices of each of the (flat) triangles. This faceted description of boundaries is used in the FCM to incorporate non-homogeneous Neumann boundary conditions, as shown in 2.2.1 and 2.3. It can nevertheless also be used to control integration of the cell stiffness matrices efficiently.

Starting from a B-rep model, the solid is discretized by a voxel model (which should not be confused with the sub-cell model for the composed integration, see Section 2.3) in order to provide a fast method for testing whether a given point, for example a Gaussian point, is located within a domain \( \Omega \) or not. The voxelization algorithm which has been implemented by Wenisch and Weisch [20] for fast grid generation in a computational steering approach for CFD applications [21–24], is based on the hierarchical space-partitioning concept of octrees. The algorithm maps the geometric objects onto a uniform Cartesian grid representation.

To demonstrate the efficiency of the voxelization algorithm let us consider the CAD-generated geometry of a passenger train compartment for which in [25,26] an air flow simulation has been carried out. The B-rep model of the passenger compartment consists of 173,085 triangles defined in the STL format, see left-hand side of Fig. 8. The voxelization with a resolution of 428 \( \times 83 \times 58 \) voxels is depicted in the right-hand side of Fig. 8. The CPU time to generate the voxel model amounts to 1.5 s on AMD Dual Opteron with 2.4 GHz only.

The aforementioned algorithm can be applied to derive a voxel model with a prescribed resolution from a given B-rep representation. Since the algorithm presented in [20] marks only the voxels that correspond to the boundary of the geometric model, it is also necessary to apply a fill algorithm in order to assign a unique identifier to the different volume sets. For this purpose, a seed fill algorithm has been implemented in this research. The non-recursive algorithm is based on a hierarchic data structure and starts from a given seed or starting point using the connectivity to mark all voxels lying in the same domain with the same identifier. The fill algorithm is repeated automatically until all voxels are assigned to a certain domain. Based on this voxel model, generating a Cartesian grid is a simple task. To this end the bounding box of the voxel model is sub-divided into \( n_x \times n_y \times n_z \) cells. During the generation of the grid, the cells that are completely outside the domain \( \Omega \) are disregarded. Since the geometry is now approximated in terms of voxels, the test to determine whether a cell is completely outside or not is very simple.

To visualize this concept let us take a look at the elastostatic computation of a dam. The left-hand side of Fig. 9 depicts the geometric model of the dam containing a tunnel. Based on the geometric model provided in the form of an STL file, the voxelization process is invoked, resulting in a resolution of 850 \( \times 238 \times 345 \) voxels. A Cartesian grid with a sub-division of \( 43 \times 12 \times 18 \) cells is generated from the voxel model. The dam is loaded by the pressure of the water acting on the convex side of the dam. This load is imposed by integrating over the faceted surface, as described in Section 2.3. The right-hand side of Fig. 9 shows a contour plot corresponding to the displacements for a cut plane. The results of the FCM are evaluated on a fine post-processing grid which is obtained by subdividing the cells. The original cells can be seen in the post-processing mesh. Although this Cartesian grid is fairly coarse, an accurate approximation can be obtained by elevating the order of the shape functions of the cells. Note that the accuracy of the geometric description depends on the resolution of the voxel model and the accuracy of the quadrature scheme which accounts for the cells cut by the boundary. Section 4.2 gives a more detailed example where the geometry is defined in terms of a B-rep model from which a voxel model is derived.

3.3. Voxel model obtained from a CT scan

As shown in the last section, transformation of a B-rep to a voxel model is an appropriate means to determine whether an integration point is inside or outside the original domain of computation \( \Omega \). It is therefore evident that FCM is especially well suited to problems where the original definition of the computational domain is already based on voxel models, like those obtained from patient-specific, quantitative computed tomography (QCT) scans. When using standard finite element methods, major difficulties arise in the creation of the geometrical model that is required for the process of mesh generation, as well as the determination of the constitutive models describing the various bone constituents. On the other hand, fast and reliable methods for predicting and monitoring in-vivo bone strength would be of major importance in clinical applications such as fracture fixation or endo-prostheses for joint replacement. Furthermore, the development of implants calls for highly efficient, robust computational tools which make it possible to match the mechanical properties of implants with those of the individual bone during a design loop, in order to avoid adaptive remodeling of the bone. Our approach, which is based on the finite
cell method, has the important advantage that the voxel model obtained from a QCT scan can be used as an immediate basis for the computation, without the necessity of segmentation [27,28] and derivation of a B-rep model as input to a finite element mesh generator.

Let us consider the elastostatic computation of a proximal femur, for example. The QCT scan of the proximal femur, which is depicted on the left-hand side of Fig. 10 was taken from Yosibash et al. [29]. The resolution of the QCT scan is 0.78125 mm in the in-plane and 0.75 mm in longitudinal direction. Each of the voxels

---

**Fig. 8.** Voxelization of a passenger train compartment, left-hand side: B-rep model (STL mesh with 173,085 triangles), right-hand side: voxel model with a resolution of $428 \times 83 \times 68$. Note that only part of the cabin is depicted.

**Fig. 9.** Geometric model of a dam, finite cell mesh with 4391 cells of degree $p = 5$, contour plot of the displacement of the dam.

**Fig. 10.** Voxel model obtained from a QCT scan, finite cell mesh and computed displacement of a proximal femur.
defined by the QCT scan gives the Hounsfield Unit, which can be correlated to the local density resulting in an inhomogeneous distribution, from which Young's modulus is computed by means of algebraic relations. The elastic modulus is therefore given as an inhomogeneous function, which can be computed from the data obtained from the QCT scan. An overview of the different models defining the relation between the Hounsfield Unit and the elastic modulus can be found in [29]. To summarize, it is possible to derive a functional description of the density and material properties from the QCT scan. It is consequently possible to automatically generate a Cartesian cell mesh by taking the inhomogeneous density of the bone into account. Cells that happen to lie completely outside the bone, where the density is below a certain threshold, are disregarded. The middle section of Fig. 10 depicts the corresponding cell mesh with a resolution of 12 × 6 × 18 resulting in 555 cells. The right-hand side of Fig. 10 shows a contour plot of the displacements due to a prescribed deformation on top of the bone. The bone was computed by applying a polynomial degree of p = 6 for the shape functions of the cells and the results were evaluated on a post-processing mesh obtained by subdividing the cells for a better visualization. Note that the inhomogeneous elastic modulus was directly computed from the data of the QCT scan. A composed integration (see Section 2.3) was applied to each cell and for each quadrature point the corresponding Young's modulus was directly computed from the data of the QCT scan. Future work will present a more detailed investigation of the FCM for biomechanical applications.

4. Numerical examples

In this section three numerical examples are presented to demonstrate the basic properties of the proposed method. All three examples are related to three-dimensional problems of linear elasticity. In order to compute the error of the FCM approximation, “overkill" FE solutions based on either h or p-refinements have been used. The computations were performed using the h and p-FEM code Adhoc [15]. The direct solver Spooltes [19] based on a LU factorization is applied to solve the overall equation system. The results are visualized by means of GID [30] using a fine post-processing mesh.

4.1. Thick-walled plate with circular hole

A classical benchmark for three-dimensional problems of solid mechanics is the thick-walled plate with a circular hole which has been considered by various research groups, see [31,32], for example. Here we assume a linear elastic isotropic material behaviour with Young's modulus E = 206900.00 N/mm² and Poisson's ratio ν = 0.29. The plate, depicted in Fig. 11, is loaded by a traction t0 = 100 N/mm² and symmetry boundary conditions are applied, reducing the model to an eighth of its original size.

Three different Cartesian grids are used to discretize the domain embedded in a cube, see Fig. 12. The number of elements is successively increased in x-y-plane whereas only one element is applied. When choosing a mesh with 16 × 16 × 1 cells, one element happens to lie completely in the circular hole and can therefore be disregarded. In this example, the embedding domain Ωh has been chosen so that Dirichlet boundary conditions and inhomogeneous Neumann boundary condition can be applied directly to the cell boundaries. The geometry of the hole is described exactly by an implicit representation and taken to be a zero extension of material during the quadrature of the element matrices. Therefore, in this first example the discretization by the FCM differs from a FE approach only with respect to the question of how accurately the circular hole can be accounted for with the composed integration scheme presented in Section 2.3.

To obtain an impression of the accuracy, let us take a look at the relative error in energy norm

\[
(G_t)_{L^2} = \frac{\int (u_{ex}(x, y, z) - u_h(x, y, z))^2 \, \mathrm{d}x \, \mathrm{d}y \, \mathrm{d}z}{\int (u_{ex}(x, y, z))^2 \, \mathrm{d}x \, \mathrm{d}y \, \mathrm{d}z}
\]

of the finite cell approximation u_h with respect to the exact solution u_{ex}. Both h and p-refinements are considered, where the element size is varied within the x-y-plane and the polynomial degree of the cells is uniformly increased from p = 1, . . . , 8 for each of the three meshes. Figs. 13–15 plot the relative error in energy norm against the number of degrees of freedom for each individual mesh. The influence of the integration error is investigated by successively increasing the number of sub-cells from 1 × 1 × 1 up to 32 × 32 × 32. A Gaussian quadrature with p = 1 integration points in each direction is performed on each sub-cell. Please note that cells that are not intersected by the boundary are integrated with one sub-cell only. From Figs. 13–15 it is evident that both h and p-refinement increase the accuracy of the approximation. The quadrature based on one sub-cell, which coincides with a standard Gaussian integration, yields oscillatory results, since the integrand related to cells intersected by the boundary is discontinuous. The Gaussian integration, which expands the integrand into a (smooth) polynomial accordingly produces oscillatory results, resulting in a poor convergence. Applying a composed integration by increasing the number of sub-cells corresponds to an h-refinement of the integrand making it possible to capture its discontinuous behaviour more efficiently. As the number of sub-cells is increased, the stiffness matrices (of the cells intersected by the boundary) are integrated more accurately and the contribution of the quadrature error to the overall error in energy norm is decreased. As soon as the integration error ceases to exceed the error in energy norm, the convergence due to h and p-refinement becomes visible. High convergence rates can be obtained by increasing the polynomial degree of the shape functions.

Fig. 16 depicts a contour plot of the von Mises stress computed with shape functions of degree p = 6 on a Cartesian grid with 8 × 8 × 1 cells and 16 × 16 × 16 sub-cells. The stresses have been directly computed for the nodes of the triangulated surface defining the geometry of the thick-walled plate with a circular hole. Note that no special post-processing techniques (like nodal averaging or smoothing-based SFR techniques, etc.) have been applied.

Since the exact solution is unknown, we replace u_{ex} with an "overkill" FE solution.
However, the directly computed stresses show a very smooth behaviour, indicating the high accuracy of the approximation.

To address the accuracy of the von Mises stress in more detail, we now take a look at the results along the cut-line A-B, as depicted in Fig. 11. Fig. 17 plots the reference solution (top) and the error of the von Mises stress for different discretizations. This shows that the moderately fine mesh with $8 \times 8 \times 1$ cells and polynomial degree $p = 8$ does not result in a (pointwise) error in the von Mises stress with respect to the reference solution that is greater than 1.5% at any point.

4.2. Sphere

The second example, which was presented by Bröker [32] in order to study the accuracy of different mapping techniques, depicts a sphere of radius $R = 5$ mm loaded by a traction $t_0 = 1.0$ N/mm$^2$ acting in normal direction, see left-hand side of Fig. 18. The material of the sphere is assumed to obey an isotropic linear elastic law with Young’s modulus $E = 1.0$ N/mm$^2$ and Poisson’s ratio $\nu = 0.3$. In the case of a hydrostatic stress state the normal components of the stress tensor are equal to the traction, $\sigma_{nx} = \sigma_{ny} = \sigma_{nz} = t_0$, and the shear components $\sigma_{xy} = \sigma_{yx} = \sigma_{zx} = 0$ vanish. Assuming that the center of the sphere is fixed, the displacement components $u_x = 0$, $u_y = Cy$ and $u_z = Cz$ with $C = \frac{1}{2}(1 - 2\nu)t_0$ are linear in $x$, $y$ and $z$.

The main point of this example is to investigate the procedure presented in Section 2.3 to apply inhomogeneous Neumann boundary conditions to a surface which intersects the finite cells. Before addressing this point, we will scrutinize the accuracy and efficiency with which the geometric model can be described by the finite cell method. Different approaches to discretize the geometry of the sphere will accordingly be considered by comparing the level of accuracy to which the volume of the sphere can be computed. The right-hand side of Fig. 18 shows a mesh consisting of 7 p-version finite elements. The geometric description of the elements is based on the quasi-regional mapping which was proposed by Kiralyfalvi and Szabó [34]. The idea is to combine the blending function method [11,33,16,35,12] with a polynomial interpolation of geometry, sampled at an optimized set of points. Lagrange polynomials are used to interpolate the geometry at the Babuska–Chen points [36,37], leading to a smooth description of geometry. The polynomial degree $p_2$ of the Lagrange polynomials is increased.
from 3 to 10 and the volume integral is evaluated by means of a Gaussian quadrature, applying \( n = p_v + 1 \) points in each direction of the hexahedral elements.

In addition, we also consider an \( h \)-version refinement by tri-linear (eight-noded) hexahedral elements. A series of successively refined meshes with up to 2401 hexahedrals is depicted in Fig. 19. The computation of the volume is performed using a Gaussian quadrature, where \( n = 2 \) integration points are applied to every element in each direction.

The geometric models for the FCM are depicted in Fig. 20. The left-hand side depicts the boundary representation (B-rep) of the sphere by its oriented boundary surface. The STL model consists of 33,994 facets. Starting from the B-rep model according to the procedure explained in Section 3, an octree representation is generated, from which a voxel model is then derived. The resolution of the voxel model is 160 x 160 x 160 resulting in a grid spacing of \( dx = dy = dz = 0.03125 \text{ mm} \). In Fig. 20, only that part of the voxel model is depicted that corresponds to the original domain \( \Omega \) in which \( x = 1 \), see Eqs. (11) and (10). In the remaining domain \( \Omega \setminus \Omega \), \( x \) is set at zero. The sphere is discretized by one cell only and the computation of the volume is carried out by a composed integration with \( n_{vc} = m \times m \times m \) sub-cells. A Gaussian quadrature with \( n = 4 \) integration points in each direction is performed on each sub-cell. The number of sub-cells \( n_{vc} = m^3 \) is increased according to \( m = 1, 2, 4, 8, 16, 32 \) and the relative error of the computed volume is depicted together with the results of the \( p \) and \( h \)-refinement in Fig. 21.

In Fig. 21 the results of the different methods are plotted against the total number of Gaussian points, representing a measure of the computational effort. Considering the finite cell method, the integration points of those sub-cells which are completely outside the domain \( \Omega \) are also counted. It would, of course, be fairly easy to reduce the effort by disregarding those sub-cells. Besides the computation based on the voxel model, the exact geometry is also taken into consideration for the finite cell method by applying an implicit description. It is evident from the comparison that the resolution of the voxel model provides sufficient accuracy in the range covered by Fig. 21. Since the geometry of the sphere is "smooth" it can be described efficiently by polynomials and accordingly the \( p \)-extension provides the most conclusive approach for the geometric description. The \( h \)-extension, where the geometry of the sphere is described by eight-noded hexahedrals, provides a less efficient approach. This corresponds to what is known from finite element computations, where \( p \)-extensions significantly outperform the \( h \)-extension for smooth problems. The computation of the volume based on the finite cell performs quite well, demonstrating that the composed integration can capture the geometry quite competently.
Having investigated the accuracy of the geometric description, we now consider the elastostatic computation of the sphere. The exact solution of the displacements is linear, and therefore one finite cell with shape functions of degree \( p = 1 \) should be sufficient to obtain accurate results. Since the mapping function (21) of the cell is linear and results in a constant Jacobian matrix (22), the polynomials defined on the standard element also remain polynomials in the global coordinate system. Symmetry boundary conditions are defined by suppressing the normal displacements of the faces of the cell. According to the explanation in Section 2.3, the cell load vector (31) resulting from the traction is obtained by integrating over the triangulated surface, which is defined in terms of the facet (STL) model. A contour plot of the displacements obtained with one cell of degree \( p = 1 \), \( n_{c} = 32^{2} \) sub-cells with \( n = 4 \) integration points in each local direction is depicted in Fig. 22. Please note that the exact solution corresponds to concentric contour lines, since \( |u_{c}| = \sqrt{u_{x}^{2} + u_{y}^{2} + u_{z}^{2}} = |(1 - 2\nu)k_{0}\sqrt{x^{2} + y^{2} + z^{2}}| \). The results presented in Fig. 22 have been evaluated on the surface

![Fig. 18. (a) Sphere with radius R and (b) one-eighth of the sphere discretized with a finite element mesh of seven p-elements.](image)

![Fig. 19. Different meshes with up to 2401 hexahedral elements applying tri-linear mapping (h-versions).](image)

![Fig. 20. Surface mesh, octree and voxel model of an eighth of a sphere.](image)

![Fig. 21. Relative error of computed volume.](image)
mesh depicted on the left-hand side of Fig. 20, which is rather coarse in terms of the symmetry planes. However, it shows that the results of the finite cell are in very good agreement with the exact solution.

To address the accuracy in more detail we consider the von Mises stress, which should vanish for a hydrostatic loading. We accordingly define the error in the von Mises stress in relation to the traction \( t_0 \)

\[
e_{\text{vM}} = \frac{|\sigma_{\text{vM}}|}{|t_0|} \times 100 \% \tag{33}
\]

and plot \( e_{\text{vM}} \) along a radial cut-line \( x = y = z = [0, R/\sqrt{3}] \) in Fig. 23. From this it is evident that the error in the von Mises stress is very low, provided that the (volume) integral is computed accurately enough by applying the composed integration. The integration of the surface integral determining the load vector was carried out sufficiently accurately so as not to affect the results. Even more accurate approximations might be obtained by taking a refined geometric model, i.e. a more dense surface mesh and voxel model.

Please note that the remarkably accurate results of the FCM for this problem are due to the fact that the exact solution can be represented by one cell of degree \( p = 1 \). Therefore, the quality of the approximation is mainly determined by the accuracy of the quadrature scheme applied to compute the stiffness matrix and the load vector. Applying the composed integration for the stiffness matrix computation allows to capture the geometry of the problem accurately enough (see also the results of the computation of the
volume, Fig. 21). The quadrature for the computation of the load vector is carried out by means of the fine face model which is depicted on the left-hand side of Fig. 20, and provides therefore also a high accuracy.

4.3. Human bone biopsy

In orthopaedic research, X-ray micro-tomography (microCT) is used to investigate bone specimens, see Perilli et al. [38], for example, and the literature listed therein. This way, trabecular and cortical bone samples can be examined non-destructively with a resolution in the micrometer range. This makes it possible to determine structural parameters such as bone volume fraction, trabecular thickness, etc. To this end, the cross-section grey-level images are binarized to separate bone from non-bone. It is of major importance for orthopaedic research and clinical applications to predict the mechanical response of bone, in order to assist orthopaedists in treatment planning. Therefore, three-dimensional finite element analysis in orthopaedics has been used for over three decades, see Yosibash et al. [29], for example, and the references cited therein. Besides the development of realistic material models describing the mechanical behaviour of bones, the fully automatic derivation of efficient computational models based on CT datasets is still a topic of ongoing research, cf. Section 3.

With this last numerical example we wish to demonstrate the capability of the FCM to automatically derive a computational model from CT scans. We therefore consider a human bone biopsy, taken from the femoral neck, see Fig. 24. This model has been created by Baruffaldi and Perilli [39]. It is based on an X-ray microCT examination and the corresponding voxel model has been converted into a B-rep model, described in terms of a triangulated surface. The STL model, depicted in Fig. 24 can be downloaded from [39]. Since the analysis designed for a detailed convergence study was too demanding without a parallelization of the solver [15] for the entire model, its size was further reduced, see Fig. 25, and the modified model was defined to describe the exact geometry from which all discretizations are derived. In the subsequent computations the bone specimen is clamped at the lower surface and subjected to a uniform compression acting on the upper surface. Assuming an isotropic linear elastic material law with Young’s modulus $E = 1000$ and Poisson’s ratio $\nu = 0.3$, we shall proceed to calculate the resulting, normal reaction force $F_n$ to the upper surface.

The commercial FE code Abaqus was used to generate a reference solution with which the results of the FCM will be compared. The discretization is based on 10-noded tetrahedral elements created with the mesh generator Negen [40]. Fig. 26 illustrates both the coarsest and the finest tetrahedral mesh of the FE computations.

The convergence of the reaction force $F_n$ for an $h$-refinement based on 10-noded tetrahedral elements is plotted in Fig. 27 against the number of degrees of freedom. Having obtained a reference solution for this problem we continue to investigate the FCM, beginning with the exact geometric model depicted in Fig. 25 the CT scan, which results in a voxel matrix, is "simulated" by applying the voxelization procedure described in Section 3. In this way, we can replace the original CT scan by a computational procedure in our study to make it easier to investigate CT scans or voxel models of different resolution, Fig. 28 shows a voxel model with a resolution of $238 \times 220 \times 90$ voxels. Please note that voxels outside the domain $\Omega$ are not dis-

---

3 It is available on the internet through the BIDI Repository managed by the Istituto Ortopedico Rizzoli, Bologna, Italy.
4 Abaqus is a trademark of ABAQUS, Inc., Rising Sun Mills, Providence, RI, US.
Cartesian grid. Three examples are given to show the efficiency of the proposed method. The influence of the quadrature error and geometry representation in terms of a voxel model is discussed in detail and it is shown that the meshing process can be dramatically simplified at the cost of a more accurate quadrature scheme. However, the main advantage of the method is the fully automatic generation of a discretization, reducing the labour-intensive aspect of pre-processing. We believe that the FCM is well suited for problems where the geometric model is given in terms of a voxel model. For this kind of problem, which tends to arise in biomechanics, for instance, where CT scans are used for patient-specific investigations, the FCM avoids the explicit generation of a geometric model which is needed for meshing algorithms applied in the standard FEM. Therefore, future work will concentrate on a seamless integration of the FCM into biomechanical simulation to support a fast, interactive numerical simulation for patient-specific surgery planning.

Examples presented in this investigation are represented by smooth solutions, i.e. no distinct singularities are present. In a two-dimensional setting, a proof was sketched showing that in this case even exponential rate of convergence can be expected [1]. It was also demonstrated that a simple local modification of the cell-grid yields very accurate results, even in the vicinity of singularities, and FCM outperforms convergence of the classical h-FEM by far. Similar investigations for problems including singularities in three-dimensional true remain to be done.

The FCM can be extended also to nonlinear problems of solid mechanics. In the case of geometrically linear problems with non-linear constitutive laws, the extension should be straightforward. Considering geometrically nonlinear problems with large deformations, the distortion of the cells needs to be taken into account and the situation is more involved. A detailed investigation of the FCM for nonlinear problems with large deformations will be subject of future work.

Acknowledgements

This work has partially been supported by the Alexander von Humboldt Foundation, the Excellence Initiative of the German federal and state governments, the TUM International Graduate School of Science and Engineering and SIEMENS AG. This support is gratefully acknowledged.

References
