# A parallel high-order fictitious domain approach for biomechanical applications

Martin Ruess, Vasco Varduhn, Ernst Rank
*Technische Universität München*
*Chair for Computation in Engineering*
*Munich, Germany*
{*ruess, varduhn, rank*}*@bv.tu-muenchen.de*

Zohar Yosibash
*Ben-Gurion University of the Negev*
*Dept. of Mechanical Engineering*
*Beer Sheva, Israel*
*zohary@bgu.ac.il*

***Abstract*—The focus of this contribution is on the parallelization of the Finite Cell Method (FCM) applied for biomechanical simulations of human femur bones. The FCM is a high-order fictitious domain method that combines the simplicity of Cartesian grids with the beneficial properties of hierarchical approximation bases of higher order for an increased accuracy and reliablility of the simulation model. A pre-computation scheme for the numerically expensive parts of the finite cell model is presented that shifts a significant part of the analysis update to a setup phase of the simulation, thus increasing the update rate of linear analyses with time-varying geometry properties to a range that even allows user interactive simulations of high quality. Paralellization of both parts, the pre-computation of the model stiffness and the update phase of the simulation is simplified due to a simple and undeformed cell structure of the computation domain. A shared memory parallelized implementation of the method is presented and its performance is tested for a biomedical application of clinical relevance to demonstrate the applicability of the presented method.**

***Keywords*-shared memory parallelization, pre-integration scheme, Finite Cell Method, fictitious domain, high-order approximation, biomechanics**

## I. Introduction

Computer-aided medical procedures are of increasing relevance in clinical practice and have already established in many clinical centres worldwide for minimal-invasive surgeries, surgical navigation and particularly for surgical pre-planning [21], [5], [8]. The need for patient-specific simulations controls the need for accurate, reliable, fast and parallel algorithms in this field that predict the in-vivo response of medical treatment and surgical interventions [22], [7].

Voxel models derived from x-ray images and quantitative computer tomography scans (QCT-scan), respectively, are the basis for any patient specific simulation e.g. in the field of bone or dental mechanics. In various other fields they are often generated by recursive bisection [20] to overcome a time-consuming and error-prone mesh-generation for the numerical simulation of structures of high geometric complexity. Such fields include e.g. seismic analyses and ground water flow in soil mechanics [10], [6], shape and topology optimization of complicated structures [12], [2], [19] or stability analyses of foam structures [15]. For Finite Element schemes the voxel approach avoids fine granular meshes but still is often limited in terms of accuracy due to a constant element-wise material assignment. In particular in terms of computational efficiency it often requires means of parallelization to generate solutions with a reasonable time effort [13], [11].

In this constribution we present a shared memory parallelization of the Finite Cell Method, a high order fictitious domain approach, that exploits the advantageous properties of Cartesian grids. An extension of the method allows a pre-computation of cell properties that shifts a significant part of the numerical effort from the simulation loop to an independent setup phase and turns out to be highly suited for parallelization due to its simple algorithmic structure. Based on the pre-computed cell characteristics the assembly of the governing system of equations is reduced in large parts and even allows user interactive simulations with acceptable update rates.

## II. Numerical simulation model

This section provides an overview about the numerical fundamentals of the implemented simulation platform, the numerical simulation method and the application for patient-specific biomechanical analyses. We give a brief overview about the Finite Cell Method and an extension for an effective parallelization for linear analyses of voxel-based, heterogeneous material models.

### *A. The Finite Cell Method*

In the following we summarize the essential idea of the Finite Cell Method in compact form. The method is formulated for solids of linear elasticity on the basis of the principle of virtual work, independent of the applied Ansatz space. In this contribution we apply a hierachic Ansatz space based on integrated Legendre polynomials as known from the p-version of the Finite Element method[3], [18].

It is shown that the method is well-suited for complex geometries and the multi-material interfaces of heterogeneous voxel models. A detailed description of the method is provided e.g. in [16].

Based on the principle of virtual displacements, the FCM satisfies the governing integral equations within a simplified

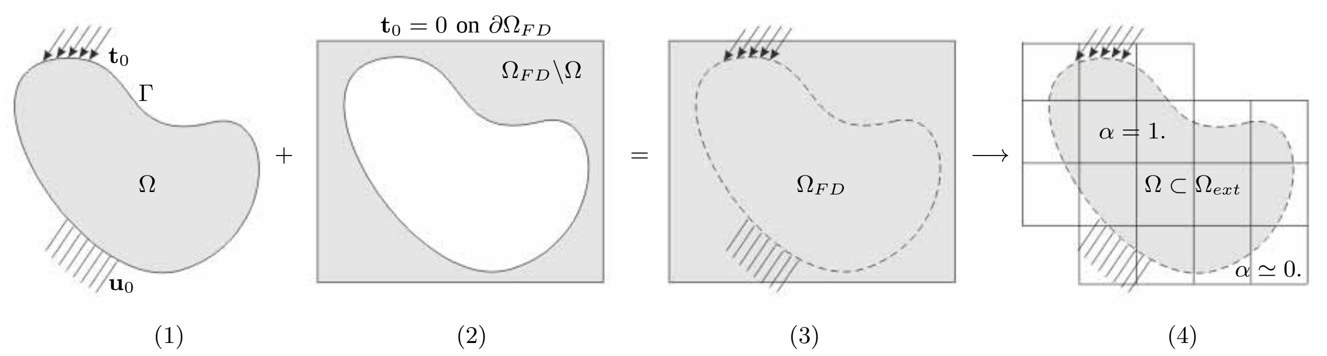

Figure 1. (1) Physical domain $\Omega$ with prescribed traction $\mathbf{t}_0$ along the Neumann boundary $\Gamma_t$ and prescribed displacements $\mathbf{u}_0$ along the Dirichlet boundary $\Gamma_u$, (2) extended cell domain $\Omega_{FD}\backslash\Omega$ with zero traction $\mathbf{t}_0$ on the cell domain surface $\partial\Omega_{FD}$, (3) embedded domain with implicit domain support for $\Omega_{FD}$ from prescribed displacement constraints on $\Gamma_u$ and (4) finally applied cell grid structure on $\Omega_{ext}$ with location function $\alpha(\mathbf{x})$.

domain of computation. The method embeds the physical domain of interest $\Omega$ in an extended cell domain $\Omega_C$ that is generated on a cartesian grid. The governing integral equations are evaluated on the physical domain $\Omega$ by a refined numerical integration scheme that captures the true boundary $\Gamma$ (Fig. 3).

In general, Dirichlet boundary conditions are prescribed on a part of the physical domain's boundary $\Gamma_u$ and traction boundary conditions are prescribed on the other part $\Gamma_t$ of the boundary $\Gamma$:

$$\mathbf{u} = \mathbf{u}_0 \qquad on\ \Gamma_u \tag{1}$$

$$\mathbf{t} = \mathbf{t}_0 \qquad on\ \Gamma_t \tag{2}$$

with displacements $\mathbf{u}$, traction forces $\mathbf{t}$ and their prescribed values $\mathbf{u}_0$ and $\mathbf{t}_0$, respectively, where

$$\Gamma = \Gamma_u \cup \Gamma_t \quad \wedge \quad \Gamma_u \cap \Gamma_t \ = \ \emptyset$$

The embedding character of the fictitious domain method requires a penalization of the stresses and volume forces in the extension domain $\Omega_{FD}$ to confine their influence on the true solution domain $\Omega$. Following Hooke's law for linear elastic material, the stress-strain relation is coupled by the elasticity tensor $\mathbf{C}$. Scaling $\mathbf{C}$ by a location dependent factor $\alpha(\mathbf{x})$ penalizes the stresses in the fictitious domain and retains their full contribution in the solution domain.

$$\boldsymbol{\sigma} \ = \ \alpha\,\mathbf{C}\boldsymbol{\epsilon} \tag{3}$$

with

$$\alpha(\mathbf{x}) \quad \begin{cases} \alpha \ = \ 1 \ \forall\ \mathbf{x} \in \Omega \\ \alpha \ = \ \epsilon \ \forall\ \mathbf{x} \in \Omega_C \setminus \Omega \end{cases} \tag{4}$$

The value for $\epsilon$ is chosen smallest possible to confine the influence of the extension domain but to ensure sufficient numerical stability w.r.t. the conditioning of the governing system of equations. Typical values for $\epsilon$ range problem dependent between $10^{-14}$ and $10^{-4}$. According to the stresses in (5), volume loads are penalized with the same $\alpha$ to restrict them to the solution domain $\Omega$.

In general, the boundary of the cell domain $\partial\Omega_C$ is assumed traction free. Traction forces are directly applied on the true domain boundary $\Gamma_t$. The absence of boundary fitted elements in fictitious domain methods requires a special treatment of Dirichlet boundary conditions (cf Fig. 3). A penalty approach [16] and a weak, Nitsche-like boundary formulation have proven to be a stable and accurate approach, the latter even providing a variationally consistent formulation of the problem. A detailled description of a weak boundary formulation for the FCM can be found in [14], [16].

Considering the formentionend extensions, the weak form of the elasticity problem on $\Omega$ is consistently extended to a formulation on the extended domain $\Omega_C$ according to the principle of virtial work:

$$\begin{aligned} \int_{\Omega_C} \delta\boldsymbol{\epsilon}^T \alpha\mathbf{C}\,\boldsymbol{\epsilon}\,dv \ &= \ \int_{\Omega_C} \delta\mathbf{u}^T \alpha\mathbf{p}_V\,dv + \int_{\Gamma_u} \delta\mathbf{u}^T\mathbf{t}\,da \\ &\quad + \int_{\Gamma_t} \delta\mathbf{u}^T\mathbf{t}_0\,da \\ \mathbf{x} \in \Gamma_u \ \Rightarrow \ \mathbf{u} \ &= \ \mathbf{u}_0 \end{aligned} \tag{5}$$

with $\boldsymbol{\epsilon}$ and $\boldsymbol{\sigma}$ representing the strain and stress vectors in Voigt notation [9], $\mathbf{u}$ the displacement vector and $\mathbf{p}_V$ the volume load vector.

The Finite Cells are implemented as hexahedral elements according to the principles of tensor product elements of the Finite Element Method. The unknown displacement field $\mathbf{u}(\mathbf{x})$ is approximated with hierarchical piecewise defined polynomials of higher order $N_i(\xi,\eta,\zeta)$ specified in the standard hexahedral $(-1 \leq \xi,\eta,\zeta \leq 1)$ (see [17])

$$\mathbf{u} \ = \ \sum_a \mathbf{N}_a(\xi,\eta,\zeta)\,\mathbf{U}_a \tag{6}$$

$$\delta\mathbf{u} \ = \ \sum_a \mathbf{N}_a(\xi,\eta,\zeta)\,\delta\mathbf{U}_a \tag{7}$$

with $\mathbf{U}_a$ and $\delta\mathbf{U}_a$ denoting the unknown degrees of freedom. The approximation of the linear strain tensor $\boldsymbol{\epsilon}$ and corresponding virtual quantity, applies the standard strain operator $\mathbf{B}(\xi, \eta, \zeta)$ that is obtained from differentiation of (6) with respect to the global coordinates $(x, y, z)$, applying the chain rule.

Following the Bubnov-Galerkin approach equations (6) and (7) are substituted into the weak form (5) providing a discrete finite cell formulation

$$\mathbf{K}\,\mathbf{U}_a \quad = \quad \mathbf{P} \tag{8}$$

with the $(N \times N)$-matrix $\mathbf{K}$ representing the system stiffness and corresponding system load vector $\mathbf{P}$.

## *B. Efficient pre-computation of cell properties*

The shape of voxelized domains is directly given either by the assembly of voxels or by the value of each voxel in the case of QCT-derived voxel data. For both cases a regular sub-cell scheme is applied, decomposing each finite cell into $(m_x \times m_y \times m_z)$ sub-cells for integration. The integrals (9) are computed numerically with Gauss quadrature. The number of quadrature points required to exactly integrate polynomials depends on the polynomial degree. Despite the polynomial character of the integral (9) that has a constant and diagonal Jacobian $\mathbf{J}$, $(p+1)^3$ quadrature points are applied for integration of cells with shape functions of degree $p$.

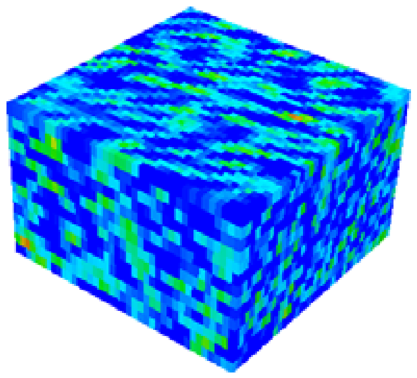

Figure 2. QCT-derived voxel cube of a trabecular bone sample, extracted from a human femur head. Each voxel is represented by a sub-cell, used for a composed integration of the cell characteristics during the pre-computation phase.

With the quantities $\mathbf{u}$, $\delta\mathbf{u}$ and $\boldsymbol{\epsilon}$ defined in the previous section, the cell stiffness matrix is represented by

$$\mathbf{K}_c \quad = \quad \int_\xi \int_\eta \int_\zeta \mathbf{B}^T \alpha\, \mathbf{C}\, \mathbf{B}\, det\mathbf{J}\, d\zeta d\eta d\xi. \tag{9}$$

For a pre-computation of the integrals (9) the following strategy is applied: The material properties of the $(n_x \times n_y \times n_z)$ voxels per finite cell are assumed constant for each voxel. This is especially true for QCT-derived voxel data and is favorably exploited to model a discrete and heterogeneous material distribution. With this property the cell stiffness integrals (9) are replaced by the sum over $(n_x \times n_y \times n_z)$-integrals, each defined over the domain of a single voxel:

$$\hat{\mathbf{K}}_c \quad = \quad \sum_{i=1}^{n_z} \sum_{j=1}^{n_y} \sum_{k=1}^{n_x} \mathbf{K}_{ijk} \tag{10}$$

$$\hat{\mathbf{P}}_c \quad = \quad \sum_{i=1}^{n_z} \sum_{j=1}^{n_y} \sum_{k=1}^{n_x} \mathbf{P}_{ijk} \tag{11}$$

The material properties change with every voxel and are uniquely identified by the indices $i, j, k$ that are used to define the integration limits of each voxel integral, according to the normalized coordinate directions $\xi, \eta, \zeta$

$$t_\xi^i \quad = \quad -1.0 + 2\,\tfrac{i-1}{n_z} \qquad i \quad = \quad 1, \ldots, n_z \tag{12}$$

$$t_\eta^i \quad = \quad -1.0 + 2\,\tfrac{j-1}{n_y} \qquad j \quad = \quad 1, \ldots, n_y \tag{13}$$

$$t_\zeta^k \quad = \quad -1.0 + 2\,\tfrac{k-1}{n_x} \qquad k \quad = \quad 1, \ldots, n_x \tag{14}$$

The voxel integrals then follow as

$$\mathbf{K}_{ijk} \;=\; \int_{t_\xi^i}^{t_\xi^{i+1}} \int_{t_\eta^i}^{t_\eta^{i+1}} \int_{t_\zeta^i}^{t_\zeta^{i+1}} \mathbf{B}^T \alpha\, \mathbf{C}_{ijk}\, \mathbf{B}\, det\mathbf{J}\, d\zeta d\eta d\xi \tag{15}$$

In the special case of isotropic material properties, the elasticity matrix $\mathbf{C}_{ijk}$ for each voxel is split into two independent parts according to the Lamé constants $\lambda_{ijk}$ and $\mu_{ijk}$ (see [4])

$$\mathbf{C}_{ijk} \quad = \quad \lambda_{ijk}\, \mathbf{C}^\lambda + \mu_{ijk}\, \mathbf{C}^\mu \tag{16}$$

where $\mathbf{C}^\lambda$ and $\mathbf{C}^\mu$ are constant matrices. Substitution of (16) in (15) results in

$$\hat{\mathbf{K}}_c \quad = \quad \sum_{i=1}^{n_z} \sum_{j=1}^{n_y} \sum_{k=1}^{n_x} (\lambda_{ijk}\, \mathbf{K}_{ijk}^\lambda + \mu_{ijk}\, \mathbf{K}_{ijk}^\mu) \tag{17}$$

for the cell stiffness with matrices $\mathbf{K}_{ijk}^\lambda$ and $\mathbf{K}_{ijk}^\mu$ only depending on $\mathbf{C}^\lambda$ and $\mathbf{C}^\mu$, respectively.

It is worth to note that the quantities $\mathbf{K}_{ijk}^\lambda$, $\mathbf{K}_{ijk}^\mu$ are independent of the material properties of each voxel and can be pre-computed in dependence of the polynomial degree of the Ansatz, the number of voxels per cell $(n_x \times n_y \times n_z)$ and the voxel spacing $(s_x, s_y, s_z)$. Thus the impact of changing material properties during the simulation of structures reduces to a modification of the Lamé constants that simply scale the pre-computed voxel stiffness, followed by summation over all stiffness contributions. Contributions from the extension domain are penalized with $\alpha$ as defined in (4).

The formentioned pre-computation scheme is examplarily shown for the cell stiffness but also holds for other cell quantities of linear analyses as cell loads, cell masses etc.

## C. Biomechanical in-vitro simulation

A fresh-frozen femur of a 63-year-old male donor was tested with a total compression load of 1000N on top of the femur head. The boundary $\Gamma_t$ where the load is applied corresponds with the boundary of the upper most voxel layer (cf Fig. 3) and allows the computation of the corresponding integral of (5) over a circular plane loading area. Before the non-destructive test the bone was QCT scanned by a clinical CT with a resolution of $1024 \times 1024$ pixel in the plane, resulting in a pixel spacing $s_x = s_y = 0.195mm$ and a slice thickness of $1.25mm$. Over the bone's surface 13 strain gauges were bonded to record the surface strain during the compression test. In addition the deflection of the bone was measured by two linear displacement sensors (Fig. 3).

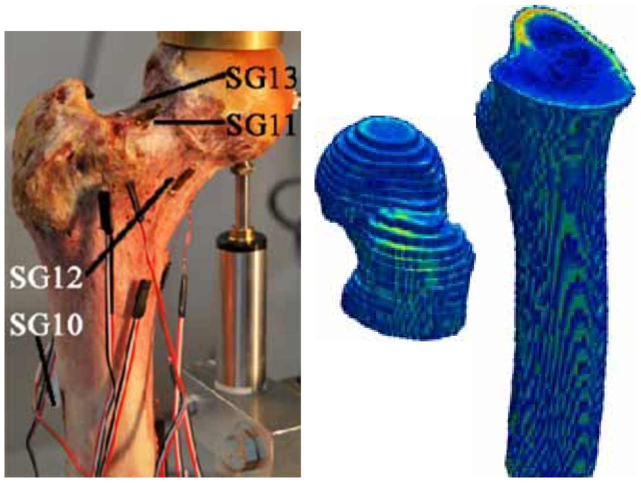

Figure 3. Experimental setup for a human femur compression test, loaded with $1000N$ (left). Cut through the corresponding QCT-derived voxel model (right).

The voxel model was embedded in 678 finite cells. According to the experiment a surface compression load was applied within the embedding top cells. The bone was fully clamped at its distal face. The principal strains at the strain gauge locations were computed pointwise and as a surface averaged result including several locations adjacent to the point result to account for the dimensions of the measuring patch of each strain gauge. A uniform $p$-refinement was applied to control the approximation error. At a polynomial degree of $p = 4$ an overall good correlation was found with a correlation factor $R^2 = 0.975$ and $R^2 = 0.962$ for the pointwise and the surface averaged results, respectively.

A complete description of the testing environment, the specific setup of the experiment, the Finite Cell model including all measured and numerically predicted results is reported in [14].

Figure 4. Algorithm *kcPrecompute()*
Pre-computation of $(nSC := (n_x \times n_y \times n_z))$ sub-cell stiffness contributions $\mathbf{K}_{ijk}$ according to Eq. (15). The sub-cells of hexahedral type apply Gauss integration over $(nGPz \times nGPy \times nGPx)$ quadrature points. The array $sc$ denotes the Gauss coordinates, mapped to the sub-cell coordinate system. The constant value $detJSC$ denotes the determinant of the Jacobian to account for the sub-cell geometry mapping to the normalized standard reference element. The following variables were declared private in the omp parallel scope : $i, j, k_{ijk}^{(\lambda/\mu)}, sc$.

```
**#pragma omp** parallel default(shared) private(...)
**#pragma omp** for
**for** $n = 1$ to $nSC$ **do**
    $k_{ijk} \leftarrow new\ Matrix(N)$
    **for** $i = 1$ to $nGPz$ **do**
        **for** $j = 1$ to $nGPy$ **do**
            **for** $k = 1$ to $nGPx$ **do**
                $sc \leftarrow gaussCoordSC(i, j, k)$
                $k_{ijk}^{(\lambda/\mu)} \leftarrow \{B(sc), (C^{(\lambda/\mu)}, detJ, detJSC\}$
            **end for**
        **end for**
    **end for**
    *fcmModel.store*$(k_{ijk})$
    **end omp** for
**end for**
**end omp** parallel
```

## III. Shared memory parallelization

In the following section we present a shared memory parallelization of the Finite Cell Method extended by a pre-computation scheme. We further document the parallel performance of our implementation.

### A. Implementation

The Finite Cell Method was implemented within the object-oriented C++ framework FELINA++ –Finite Elements for LInear and Nonlinear Analyses. In the following the basic routines for the computation of the cell stiffness, including the pre-computation scheme and the stiffness update are summarized as well as the assembly process (Figures 4 and 5). The implementation and OpenMP extension [1] is straightforward and profits from the simple structural properties of the pre-computation and the assembly, respectively. Due to the applied orthogonal cell grid the Jacobian of the finite cell and its sub-cells is constant as well as the material matrices $\mathbf{C}^\lambda$ and $\mathbf{C}^\lambda$. The strain interpolation matrix $\mathbf{B}$ is the only non-constant quantity of significance in the pre-computation that depends on the Gauss coordinates mapped to the corresponding sub-cell coordinate system.

At a first glance the pre-computation scheme seems to be memory intensive, particularly for higher polynomial degrees with polynomially increasing cell matrix dimensions. Fortunately it turns out that a reasonable number of sub-cells

Figure 5. Algorithm *systemStiffness()*
Assembly of all finite cell stiffness contributions to the system matrix $\mathbf{K}$ of the governing system of equations (cf Eq. (8)). In lines 5-11 the stiffness for each finite cell is assembled according to Eq. (17). Line 13 assembles the system stiffness matrix by the cell contributions. The following variables were declared private in the omp parallel scope : $n, i, j, kc_n, \lambda_{ijk}, \mu_{ijk}$.

```
1: $kcPrecompute()$
2: #pragma omp parallel default(shared) private(...)
3: #pragma omp for
4: for $n = 1$ to $nCells$ do
5:     for $i = 1$ to $nz$ do
6:         for $j = 1$ to $ny$ do
7:             for $k = 1$ to $nx$ do
8:                 $kc_n \leftarrow \{k_{ijk}^{(\lambda/\mu)}, \lambda_{ijk}, \mu_{ijk}\}$
9:             end for
10:        end for
11:    end for
12:    #pragma omp critical
13:    $K \leftarrow addKcStiffness(kc_n)$
14:    end omp critical
15: end for
16: end omp for
17: end omp parallel
```

per finite cell is limited by the CT-voxel resolution of the solution domain and an adequate number of cells that fully cover the solution domain, following the classical properties and rules for high-order FEM discretizations. For the simulation model described in II-C a already coarse granular cell resolution was chosen that considers $(40 \times 40 \times 10)$-voxel per finite cell, resulting in 678 finite cells.

The Lamé constants $\lambda_{ijk}$ and $\mu_{ijk}$ are precomputed in the setup phase of the simulation according to a linear relation between the hounsfield units (HU) of the QCT-data and the bone mineral density (BMD) that is callibrated by a QCT-scanned phantom device of known BMD values.

Write access to the system matrix still results in a race condition in our implementation and requires additional effort to remove the necessary critical section.

## B. Performance analysis

The focus of the following performance test lies on the pre-computation scheme and the complete assembly of the system stiffness matrix. For time-critical computations as e.g. user interactive simulations in the framework of a steering environment the number of sub-cells necessary to update after each change in geometry or material properties is typically significantly smaller and requires only a fraction of the computational effort. The solution process of the governing system of equations is out of the scope of this analysis and will not be reported in the following.

Speedup and parallel efficiency are derived from wall time measurements. All computations were performed on two six

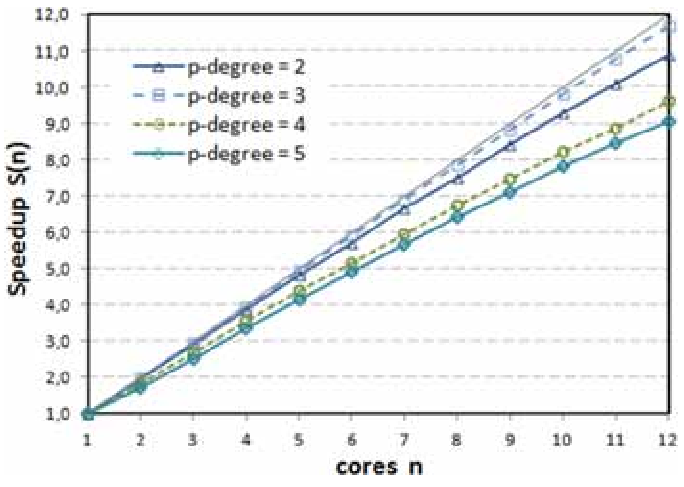

Figure 6. Speedup for the pre-computation scheme.

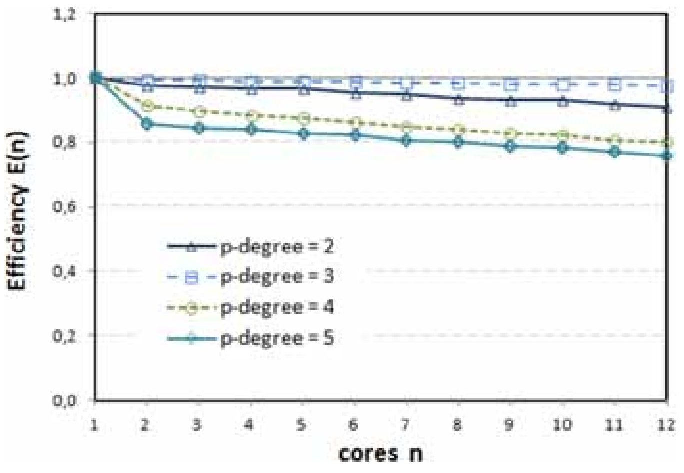

Figure 7. Parallel efficiency for the pre-computation scheme.

core Intel(R) Xeon(R) CPU X5690 @ 3.47GHz with $12MB$ L2 cache. At a polynomial degree of $p = 4$ a sufficient convergence of the simulation result was observed by the relative error in energy norm.

Speedup and parallel efficiency for the pre-computation scheme clearly show a linear tendency thus exploiting the algorithmic structure at a high level. Obviously the speedup for a polynomial degree $p = 3$ combines best the load distribution, chache efficiency and algorithmic structure of the computation resulting in $> 97\%$ parallel efficiency. Between $p = 3$ and $p = 4$ a jump can be observed most likely to the increased number of degrees of freedom that results from a high number of additional edge-modes in the applied approximation basis thus having direct influence on the cache load. Still a satisfying speedup above $9$ is noticed on twelve cores.

The assembly of the system stiffness performes well up to eight cores. Due to the critical section constraining the

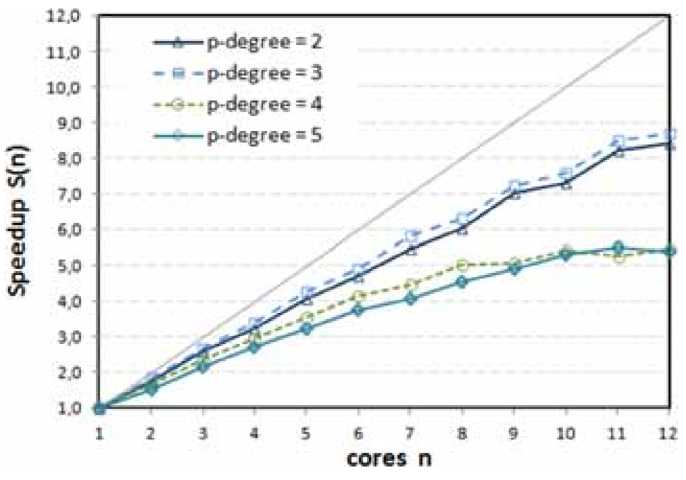

Figure 8. Speedup for the total computation: pre-computation and assembly

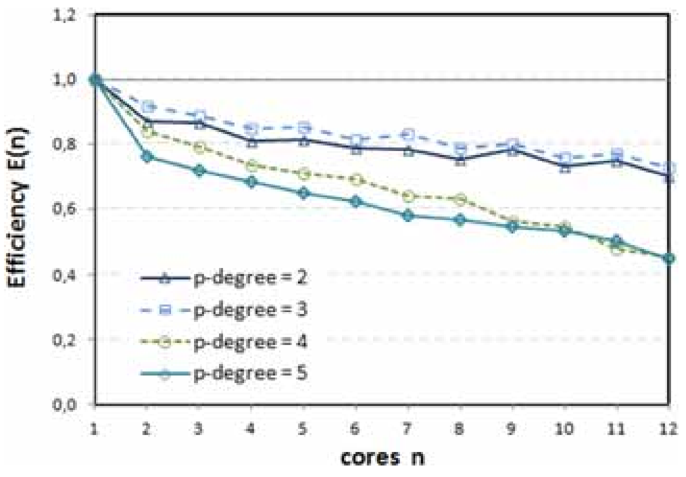

Figure 9. Parallel efficiency for the total computation: pre-computation and assembly

assignment of the cell stiffness coefficients into the relatively sparse system matrix the assembly speedup stagnates with an increasing matrix dimension on cell level. At $p = 4$ already $150$ degrees of freedom determine the cell matrix size and the resulting assembly effort. More sophisticated datastructures on cell and system level are necessary to overcome this drawback. Still the total efficiency including pre-computation and assembly is around $60\%$ on eight cores for $p = 4, 5$. The reduced cell matrix dimensions for $p = 2$(60 dof) and $p = 3$(96 dof) even perform above a parallel efficiency of $60\%$ on all twelve cores, a result that is of special relevance for models of homogeneous material properties and problems that fully exploit the hierarchy of the p-version Ansatz space for refinement in a user interactive simulation loop. At $p = 3$ reliable results are already avilable that provide sufficient accuracy for a steering approach, in general even superior to results from a classcal h-version Finite Element analysis.

The time effort for pre-computation and assembly, measured on all 12 cores, provides a first impression about update rates in a user interactive steering environment though none of the routines has been optimized for this task. For the pre-computation of $(40 \times 40 \times 10)$ sub-cells the time effort varies between $7\,s(p = 3)$ and $42\,s(p = 4)$. The total assembly of the cell matrices of the complete model into the system stiffness matrix requires $26\,s(p = 3)$ and $150\,s(p = 4)$, respectively. Since interactive changes are restricted to a few cells, the update of a patch of 10 cells was measured with $< 0.3\,s(p = 3)$ and $< 2\,s(p = 4)$. A detailled time effort analysis based on highly optimized routines and libraries is out of the scope of this contribution and is reported e.g. in [21].

## IV. Conclusion

In this contribution we have introduced an efficient pre-computation scheme as an extension to the Finite Cell Method, a high order fictitious domain method that turns out to be highly suited for problems of high complex geometry and multi-material interfaces. The simplicity of the algorithm allows a straightforward parallelization, still providing results on a very high accuracy level. Speedup and parallel efficiency of the implementation were demonstrated for a biomechanical problem of clinical relevance showing reasonable results that underline the algorithmic fitness of the method for parallelization. Refined datastructures and an extension to a distributed memory architecture are under development showing already promising results.

## References

[1] http://openmp.org/wp/.

[2] T. Adachi, K. Hiromichi, and T. Yoshihiro. Shape optimization based on traction method using voxel-fem. *Transactions of the Japan Society of Mech. Engineers*, 70(691):426–433, 2004.

[3] I. Babuška, B. A. Szabo, and I. N. Katz. The $p$-Version of the Finite Element Method. *SIAM Journal on Numerical Analysis*, 18:515–545, 1981.

[4] K.J. Bathe. *Finite element procedures*. Prentice Hall, 1996.

[5] D. Bongini, M. Carfagni, and L. Governi. A semiautomatic computer program for selecting hip prosthesis femoral components. *Computer Methods and Programs*, 63(2):105–115, 2000.

[6] Q. Cai, S. Kollmannsberger, R. Mundani, and E. Rank. The finite cell method for solute transport problems in porous media. In *Proceedings of the International Conference on Finite Elements in Flow Problems*, Garching, Germany, 2011.

[7] C. Dick, Georgii J., R. Burgkart, and R. Westermann. Stress Tensor Field Visualization for Implant Planning in Orthopedics.

[8] A.M. DiGioia, D. Simon, B. Jaramaz, and M. Blackwell. The value of preoperative planning for total hip arthroplasty. *Computer Assisted Orthopaedic Surgery Symposium*, 80B:382, 1995.

[9] P. Helnwein. Some remarks on the compressed matrix representation of symmetric second-order and fourth-order tensors. *Computer Methods in Applied Mechanics and Engineering*, 190(22–23):2753–2770, 2001.

[10] K. Koketsu, H. Fujiwara, and Y. Ikegami. Finite-element simulation of seismic ground motion with a voxel mesh. *Pure and Applied Geophysics*, 161:2183–2198, 2004.

[11] S. Margenox and Y. Vutov. Comparative analysis of pcg solvers for voxel fem systems. In *Proc Int'l Multiconference Comp. Science Information Techn.*, pages 591–598. 2007.

[12] J. Parvizian, A. Düster, and E. Rank. Topology optimization using the finite cell method. *Optimization and Engineering*, in press, 2011.

[13] B. van Rietbergen. Computational Strategies for Iterative Solutions of Large FEM Applications Employing Voxel Data. *International Journal for Numerical Methods in Engineering*, 39:2743–2764, 1996.

[14] M. Ruess, D. Tal, N. Trabelsi, Z. Yosibash, and E. Rank. The Finite Cell Method for bone simulations: Verifcation and validation. *Biomechanics and Modeling in Mechanobiology*, 11(3):425–437, 2012.

[15] D. Schillinger, A. Düster, and E. Rank. The $hp$-$d$ adaptive Finite Cell Method for geometrically nonlinear problems of solid mechanics. *International Journal for Numerical Methods in Engineering*, DOI: 10.1002/nme.3289, 2011.

[16] D. Schillinger and E. Rank. An unfitted $hp$ adaptive finite element method based on hierarchical B-splines for interface problems of complex geometry. *Computer Methods in Applied Mechanics and Engineering*, 200(47-48):3358–3380, 2011.

[17] B.A. Szabó and I. Babuška. *Finite element analysis*. John Wiley & Sons, 1991.

[18] B.A. Szabó, A. Düster, and E. Rank. The p-version of the Finite Element Method. In E. Stein, R. de Borst, and T. J. R. Hughes, editors, *Encyclopedia of Computational Mechanics*, volume 1, chapter 5, pages 119–139. John Wiley & Sons, 2004.

[19] T. Torigaki and K. Fujitani. Power of a voxel approach to structural analysis and topology-shape optimization in automobile industries. *Japan J. Indust. App. Math.*, 17:129–147, 2000.

[20] P. Wenisch and O. Wenisch. Fast octree-based voxelization of 3D boundary representation-objects. Technical report, Lehrstuhl für Bauinformatik, Technische Universität München, 2004.

[21] Z. Yang, S. Kollmannsberger, A. Düster, M. Ruess, R. Burgkart, E. Garcia, and E. Rank. Non-standard bone simulation: Interactive numerical analysis by computational steering. *Computing and Visualization in Science*, accepted, 2011.

[22] Z. Yang, M. Ruess, S. Kollmannsberger, A. Düster, and E. Rank. An efficient integration technique for the voxel-based Finite Cell Method. *International Journal for Numerical Methods in Engineering*, accepted, 2011.