Eurographics Symposium on Geometry Processing 2008
Pierre Alliez and Szymon Rusinkiewicz
(Guest Editors)

Volume 27 (2008), Number 5

Polyhedral Finite Elements Using Harmonic Basis Functions

Sebastian Martin[1] Peter Kaufmann[1] Mario Botsch[1,2] Martin Wicke[3] Markus Gross[1]

[1] Computer Graphics Laboratory, ETH Zurich [2] Computer Graphics Group, Bielefeld University [3] Stanford University

Abstract

Finite element simulations in computer graphics are typically based on tetrahedral or hexahedral elements, which enables simple and efficient implementations, but in turn requires complicated remeshing in case of topological changes or adaptive refinement. We propose a flexible finite element method for arbitrary polyhedral elements, thereby effectively avoiding the need for remeshing. Our polyhedral finite elements are based on harmonic basis functions, which satisfy all necessary conditions for FEM simulations and seamlessly generalize both linear tetrahedral and trilinear hexahedral elements. We discretize harmonic basis functions using the method of fundamental solutions, which enables their flexible computation and efficient evaluation. The versatility of our approach is demonstrated on cutting and adaptive refinement within a simulation framework for corotated linear elasticity.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling

1. Introduction

Finite element methods have become tremendously important in computer graphics. They are used to animate scenes showing "real-world" behavior, for example in virtual reality applications like surgery simulations, or computer animations for feature films. Its versatility and rigorous mathematical foundation make the finite element method an indispensable tool, which consequently has been applied to a wide variety of problems.

Traditionally, FEM simulations in computer graphics rely on strictly tetrahedral or hexahedral meshes, which simplifies the finite element approximation and significantly speeds up the involved computations. However, allowing a single element shape only can be too restrictive, since it requires complex remeshing in case of topological changes, for instance due to cutting, fracture, or adaptive refinement.

One class of approaches [MBF04, SDF07, SSIF07] therefore avoids remeshing after cutting by embedding each resulting cut part into an individual copy of the original tetrahedron. An interesting alternative is the approach of Wicke et al. [WBG07]: They directly support more general convex polyhedral elements in finite element simulations by employing mean-value coordinates as a generalization of linear barycentric FEM shape functions.

In this paper we extend their approach to arbitrary convex and non-convex polyhedral elements using *harmonic coordinates* [JMD*07] as FEM basis functions (cf. Fig. 1). Harmonic basis functions naturally generalize linear basis functions for tetrahedral elements and trilinear basis functions for hexahedral elements. Hence, our method seamlessly integrates into existing FEM frameworks, such that standard tetrahedral or hexahedral elements can be used in regular parts of the model, whereas irregular polyhedral elements are used in regions of cutting or adaptive refinement.

Harmonic coordinates, as the solution of a Laplace PDE with Dirichlet boundary constraints, have an analytic solution for simple element shapes only. For general polyhedra they therefore have to be approximated numerically, using for instance finite differences, finite elements, or the boundary element method. As a simple and flexible alternative we propose an approximation using radial basis functions, which guarantees the resulting basis functions to be harmonic and to furthermore exactly satisfy important physical conservation properties.

The use of general polyhedral elements significantly increases the flexibility in generating and manipulating the simulation mesh. We demonstrate this versatility in examples of cutting and adaptive refinement within a simulation framework for elastically deformable models.

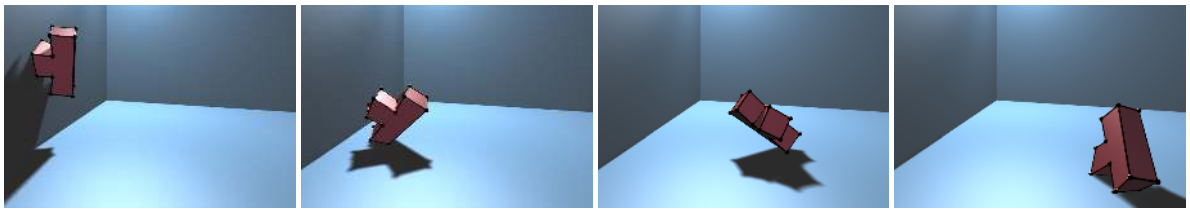

Figure 1: *Using harmonic basis functions, even non-convex polyhedral elements can be used directly in FEM simulations.*

2. Related Work

Starting with Terzopoulos et al. [TPBF87,TF88], physically-based methods have been used extensively in computer graphics. In this paper, we focus on elastically deformable solids that are simulated using the finite element method. From the vast body of literature in the field, we only cite key references, and refer the interested reader to [NMK*06] for a more comprehensive overview.

In computer graphics, finite element methods are implemented almost exclusively using tetrahedral (e.g. [OH99]) or hexahedral meshes (e.g. [MTG04, JBT04]). These discretizations allow for simple and efficient implementations of the finite element method, but in turn require rather complex mesh restructuring in case of topological changes, for instance due to fracture, cutting, or mesh refinement.

In the context of fracture or cutting simulation, one class of approaches tackled the arising problems using continuous remeshing [OH99,OBH02,SOG06]. In contrast, the virtual node algorithm [MBF04] and its generalization [SDF07, SSIF07] duplicates elements instead of splitting them, and embeds the surface parts into those copies. Point-based approaches do not have to maintain a consistent simulation mesh [MKN*04], but on the other hand have to update special shape functions [PKA*05] or distance graphs [SOG06]. In contrast, Wicke et al. [WBG07] avoid costly remeshing by supporting general convex polyhedra in FEM simulations.

The need to accurately simulate highly detailed models lead to the development of adaptive and hierarchical simulation techniques [DDCB01, GKS02, CGC*02, OGRG07]. Hanging nodes, or T-junctions, in an adaptively refined simulation mesh often pose problems, and hence have to be either avoided or specially treated, for instance by refining basis functions instead of elements [GKS02, CGC*02], or by constraining hanging nodes to edge midpoints [SSIF07].

We propose to handle the complex elements arising during cutting as well as hanging nodes due to adaptive refinement within a single, consistent simulation framework based on arbitrary polyhedral elements. The key to such a generalization is to find basis functions that fulfill all necessary requirements for convergent FEM schemes [Hug00], but are defined for a larger class of element shapes.

Our approach is inspired by, and hence most similar to, [WBG07], who employed 3D mean-value coordinates [FKR05,JSW05,JLW07] as shape functions for convex polyhedra. The main drawback of mean-value coordinates, as well as of Wachspress coordinates [Wac75,War96], is that they are defined on simplicial polyhedra only, i.e., on *convex* elements with *triangulated* faces. The restriction to convex elements might even be advantageous in practice, e.g., for collision detection. However, having to triangulate element faces can cause erroneous (slight) asymmetries for otherwise perfectly symmetric configurations (e.g., symmetric deformations of symmetric hexahedral meshes).

To overcome these limitations, we propose to use harmonic coordinates [JMD*07] as FEM basis functions, which enables the simulation of arbitrary convex and non-convex polyhedral elements with planar (not necessarily triangulated) faces. In contrast to mean-value coordinates, harmonic functions generalize both linear basis functions for tetrahedra and trilinear basis functions for hexahedra, and therefore can be considered a seamless generalization of linear basis functions to arbitrary polyhedral elements.

3. Elastic Deformations

Before introducing harmonic basis functions (Section 4), discussing their numerical approximation (Section 5), and describing their integration into a dynamic simulation framework (Section 6), we will first review the basic concepts of continuum elasticity and its Galerkin discretization in this section. A more detailed overview of deformable models and finite element discretizations can be found in the survey [NMK*06] or in the textbooks [Chu96, Hug00].

In the following we consider an object with material coordinates $\mathbf{x} = [x,y,z]^T$, which is deformed by a displacement field $\mathbf{u} = [u(\mathbf{x}), v(\mathbf{x}), w(\mathbf{x})]^T$. The local deformations of the material are measured by a 3×3 strain tensor, where typical choices are the nonlinear Green-Lagrange strain

$$\varepsilon_G(\mathbf{u}) = \frac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T + \nabla\mathbf{u}(\nabla\mathbf{u})^T\right) \qquad (1)$$

or the linearized Cauchy strain

$$\varepsilon_C(\mathbf{u}) = \frac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right). \qquad (2)$$

We employ the linear Cauchy strain $\varepsilon = \varepsilon_C$ in our framework, since it greatly simplifies the simulation, and the resulting linearization artifacts can be mostly eliminated by a co-rotational formulation [MG04]. Note, however, that the harmonic basis functions we introduce in Section 4 are equally valid for nonlinear strain measures.

In an elastic material, strain leads to restoring forces represented by a 3×3 stress tensor σ. Since stress and strain are symmetric tensors, we can represent their independent degrees of freedom by 6-vectors. Assuming a Hookean material yields a linear relation between stress and strain

$$\sigma(\mathbf{u}) = \mathbf{C}\,\varepsilon(\mathbf{u})\,, \tag{3}$$

where the symmetric 6×6 matrix $\mathbf{C}$ contains the elastic coefficients of the material. Finally, the elastic energy $\mathrm{E}(\mathbf{u})$ stored in a deformed object is given by stress times strain, integrated over the object volume Ω:

$$\mathrm{E}(\mathbf{u}) \;=\; \frac{1}{2}\int_\Omega \varepsilon(\mathbf{u})^T\,\mathbf{C}\,\varepsilon(\mathbf{u})\;\mathrm{d}\mathbf{x}\,. \tag{4}$$

The displacement field $\mathbf{u}(\mathbf{x})$ is then discretized using a finite dimensional space spanned by basis functions $N_i(\mathbf{x})$. To this end, the object is partitioned into finite elements e, i.e., $\bigcup e = \Omega$, and $\mathbf{u}(\mathbf{x})$ is approximated by interpolating the displacements $\mathbf{u}_i$ of the nodes $\mathbf{x}_i$ within elements. The interpolated displacement $\mathbf{u}^e$ in an element e of k nodes is

$$\mathbf{u}(\mathbf{x})\,|_e \;\approx\; \mathbf{u}^e(\mathbf{x}) \;:=\; \sum_{i=1}^{k} \mathbf{u}_i N_i^e(\mathbf{x})\,, \tag{5}$$

where the shape functions $N_i^e = N_i|_e$ determine the influence of the nodal displacements $\mathbf{u}_i$ in the element. From the gradient $\nabla\mathbf{u}^e(\mathbf{x})$ one computes strain and stress, such that the elastic energy stored in the deformed element e

$$\mathrm{E}^e\big(\mathbf{u}^e\big) \;=\; \frac{1}{2}\int_e \varepsilon\big(\mathbf{u}^e\big)^T\,\mathbf{C}\,\varepsilon\big(\mathbf{u}^e\big)\;\mathrm{d}\mathbf{x}\,, \tag{6}$$

can be accumulated into the total energy $\mathrm{E}(\mathbf{u}) = \sum_e \mathrm{E}^e(\mathbf{u}^e)$. Its variational derivative $\partial\mathrm{E}/\partial\mathbf{u}$, respectively the partial derivatives $\partial\mathrm{E}/\partial\mathbf{u}_i$, constitute the internal restoring forces that drive the dynamic elasticity simulation.

The basis functions N_i have to fulfill a number of requirements in order to guarantee convergence of the method under refinement of the discretization [Hug00]: For linear elasticity, they have to be in the Sobolev space H^1. In the special case of linear FEM they have to be C^1 smooth within elements and C^0 continuous across element boundaries. Furthermore, the basis functions need to exactly reproduce linear functions (e.g., rigid motions). These conditions are satisfied by linear shape functions for tetrahedra and trilinear shape functions for hexahedra — the element types and basis functions most frequently used in computer graphics. However, since those require complex remeshing for topological changes, we introduce more flexible shape functions for general polyhedral elements in the following section.

4. Harmonic Basis Functions

We propose to use harmonic basis functions as a generalization of linear barycentric basis functions to general polyhedral elements. A shape function $N_i^e : e \to \mathbb{R}$ is *harmonic* if its Laplacian vanishes in e, in which case it is uniquely determined by Dirichlet boundary constraints $b(\mathbf{x})$ on ∂e:

$$\Delta N_i^e(\mathbf{x}) = 0\,, \qquad \text{for} \quad \mathbf{x} \in e\,, \tag{7}$$

$$N_i^e(\mathbf{x}) = b_i(\mathbf{x})\,, \qquad \text{for} \quad \mathbf{x} \in \partial e\,. \tag{8}$$

For a finite element simulation we need nodal basis functions N_i^e for interpolating quantities within each element e. If these functions are chosen to be harmonic, they are fully determined by the values $b_i(\mathbf{x})$ on the element boundary ∂e, which we set up following [JMD*07]: First, in order to interpolate nodal quantities, the basis function N_i^e of node i has to equal 1 at the node $\mathbf{x}_i$ and 0 at all others, i.e.,

$$N_i^e\big(\mathbf{x}_j\big) \;=\; \delta_{ij} \quad \forall\, i,j = 1,\ldots,k\,. \tag{9}$$

Additionally, in order to ensure continuity across element boundaries, the values of the basis function for node i defined in neighboring elements e_1 and e_2 should coincide on the face or edge shared by the two elements:

$$N_i^{e_1}(\mathbf{x}) \;=\; N_i^{e_2}(\mathbf{x}) \quad \text{for} \quad \mathbf{x} \in e_1 \cap e_2\,. \tag{10}$$

This can be guaranteed by choosing the values on the faces of a d-dimensional element to be $(d-1)$-dimensional harmonic coordinates. For a trivariate harmonic basis function N_i^e on a 3D element e the boundary conditions are bivariate harmonic coordinates on its faces, which themselves are determined by univariate harmonic (i.e., linear) interpolation of the nodal values $N_i^e\big(\mathbf{x}_j\big) = \delta_{ij}$ along the edges.

It follows from these recursively defined boundary constraints and the uniqueness of harmonic functions for fixed Dirichlet constraints, that harmonic shape functions reproduce linear triangles and bilinear quads in 2D, as well as linear tetrahedra and trilinear hexahedra in 3D. The harmonic basis for a more complex 2D element is shown in Fig. 2.

Harmonic basis functions satisfy all requirements for admissible FEM basis functions (we refer the reader to [JMD*07] for short proofs of some of these properties):

- Since 3D harmonic shape functions degenerate to 2D harmonic coordinates on element faces, they are continuous across element boundaries: $N_i \in \mathrm{C}^0(\Omega)$.
- As solution to Laplace's equation (7) they are smooth within elements: $N_i^e \in \mathrm{C}^\infty(e)$.
- For fixed constraints b_i, (7) characterizes the minimizer of the Dirichlet energy $\int_e \|\nabla N_i^e\|^2$. Hence, the gradients of harmonic functions are square integrable: $\nabla N_i^e \in \mathrm{L}^2(e)$. Combining the last three points we get $N_i \in \mathrm{H}^1(\Omega)$.
- They are non-negative $N_i(\mathbf{x}) \in [0,1]$, build a partition of unity $\sum_i N_i(\mathbf{x}) = 1$, and reproduce linear functions.

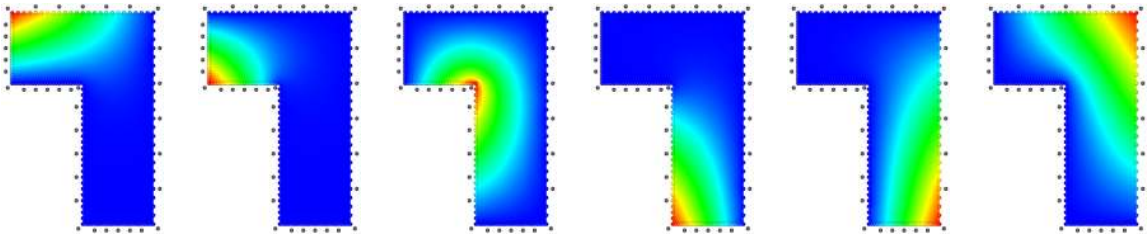

Figure 2: *Harmonic basis functions for the six nodes of a non-convex 2D element. The constraint collocation points* $\mathbf{c}_i$ *are visualized as small spheres along the element boundary, the kernel centers* $\mathbf{k}_i$ *are shifted slightly outside and are shown in gray.*

5. Numerical Approximation

Closed form expressions for harmonic basis functions exist for simple element shapes only, such as tetrahedra or hexahedra. For more general elements, harmonic basis functions N_i^e have to be computed numerically as the solution of (7), (8), which is valid for both 2D faces and 3D elements. To this end, several well-established techniques for solving the 2D and 3D Laplacian PDEs exist, each having their own respective advantages and drawbacks.

Finite Differences. Overlaying the element by a regular 3D grid and using a finite difference discretization leads to the solution of a sparse linear system for a piecewise trilinear approximation of N_i^e [JMD*07]. While this method is comparatively easy to implement, an accurate solution requires a sufficiently dense grid in order to resolve the smallest edges/faces of the element. In particular for cutting, where small edges occur frequently, the cubic growth of volumetric grids leads to very complex systems, which require advanced multi-grid methods for their solution [JMD*07].

Finite Elements could be used to solve (7) on an adaptive tessellation of polyhedral elements, thereby overcoming the limitations of regular grids. However, since the major goal of our approach is to enable adaptive FEM computations *without* complex remeshing of elements, the recursive application of adaptive FEM to each polyhedral element is a chicken-and-egg problem and contradicts our goals.

Boundary Element Method. The boundary element method (BEM) is also well suited to solve the PDE (7). By formulating the solution as an integral of fundamental solutions over the element's boundary, it avoids a volumetric tessellation and therefore needs a boundary discretization only. Due to our experiments the major drawback of BEM is performance: In its exact formulation, each function evaluation requires a full integral over the element's boundary, which makes the numerical integration of (6) very expensive.

Fundamental Solutions. While all the above methods can be employed for solving (7), (8) we found the method of fundamental solutions (MFS) [FK98] to be a more flexible, easier-to-implement, yet sufficiently accurate alternative.

MFS is closely related to BEM: It is also a boundary method, and thus does not require a volumetric tessellation, and it also represents the approximate solution in terms of fundamental solution kernels. However, instead of the boundary integrals in BEM, MFS employs a simple, meshless collocation. A shape function $N_i^e(\mathbf{x})$, simply denoted by $N(\mathbf{x})$ in the following, is represented in the following form:

$$N(\mathbf{x}) \;=\; \sum_{j=1}^{n} w_j \cdot \psi\big(\|\mathbf{x}-\mathbf{k}_j\|\big) \;+\; \mathbf{a}_1^T\mathbf{x} + a_0\,, \qquad (11)$$

where the first part is a superposition of n weighted radial basis functions ψ (RBFs), centered at $\mathbf{k}_j$, and the second part is a linear polynomial in $\mathbf{x}$. The kernel function ψ is chosen as fundamental solution of the Laplace PDE, which is $\psi(r) = \log r$ in 2D and $\psi(r) = 1/r$ in 3D. As a consequence, the function (11) is harmonic by construction [Duc77], in the whole domain except at the kernels' singularities $\mathbf{k}_j$.

Hence, the kernel $\mathbf{k}_j$ have to be placed outside the element. A standard method is to first sample the boundary by $\mathbf{s}_j \in \partial e$, and to move the kernels outward in (interpolated) normal direction by a small fraction ξ of the element size:

$$\mathbf{k}_j \;=\; \mathbf{s}_j \;+\; \xi \cdot \mathrm{size}(e) \cdot \mathbf{n}\big(\mathbf{s}_j\big)\,. \qquad (12)$$

For non-convex elements one additionally has to take care that this simple offsetting does not generate centers in the element's interior. For generating the n samples $\mathbf{s}_j$, we select the element's nodes, about 3–5 samples on each edge, and a uniform sampling of its faces of about the same density.

The function (11) satisfies (7) by construction, thus we solve for the best approximation of the Dirichlet constraints (8). To this end, we approximate the boundary integral of the L^2 error by a sum of m collocation points $\mathbf{c}_i$:

$$\int_{\partial e} |N(\mathbf{x}) - b(\mathbf{x})|^2 \approx \frac{1}{m}\sum_{i=1}^{m} |N(\mathbf{c}_i) - b(\mathbf{c}_i)|^2 \to \min\,. \qquad (13)$$

These collocation points $\mathbf{c}_i \in \partial e$ are generated equivalently to the samples $\mathbf{s}_j$, but at a higher resolution of $m \approx 3n$. The distribution of kernel centers $\mathbf{k}_i$ and collocation constraint points $\mathbf{c}_i$, as well as the resulting basis functions are shown for a non-convex L-shaped 2D element in Fig. 2.

Given the kernel centers $\mathbf{k}_i$, the minimization of the L^2 error (13) amounts to solving an overdetermined linear least squares system for the coefficients of (11):

$$\begin{bmatrix} \psi_{11} & \dots & \psi_{1n} & \mathbf{c}_1^T & 1 \\ \vdots & & \vdots & \vdots & \vdots \\ \psi_{m1} & \dots & \psi_{mn} & \mathbf{c}_m^T & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_n \\ \mathbf{a}_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} b(\mathbf{c}_1) \\ \vdots \\ b(\mathbf{c}_m) \end{bmatrix}, \quad (14)$$

where $\psi_{ij} = \psi(\|\mathbf{c}_i - \mathbf{k}_j\|)$. This least squares system can be solved in a numerically robust manner using the QR factorization or the SVD pseudo-inverse [GL89].

In order to compute a 3D basis function N_i^e for an element e, we first solve the above linear system on each of its faces. The resulting 2D harmonic functions constitute the boundary constraints for the final 3D linear system, which yields the coefficients of N_i^e. Notice that in order to compute all k basis functions $N_1^e, \dots, N_k^e$ of an element e with k nodes, the same 2D and 3D systems are solved for k different right-hand sides $b_i(\mathbf{x})$. After factoring each matrix once, these systems can be solved efficiently by back-substitution.

Discussion. As described in Section 4, *exact* solutions of (7), (8) satisfy the conditions for admissible FEM shape functions. The numerical approximation N from (11) satisfies all but one exactly, and one up to small numerical errors.

Since the singularities $\mathbf{k}_j$ are located outside of e, we have $N \in C^1(e)$ and $\nabla N \in L^2(e)$. However, the C^0 continuity across elements is only satisfied approximately through the Dirichlet conditions (8), resp. (13). Our typical choice of 5 edge samples for generating kernels through (12) leads to L^2 boundary errors of about 2–3%. More accurate results can be achieved by using more kernels in (11). However, as we show in Section 8, the accuracy of the global solution is not limited by the individual basis functions' errors.

Typically, MFS approximations are based on fundamental solution kernels $\psi(\|\cdot - \mathbf{k}_j\|)$ only, and do not include a linear polynomial as in (11). This polynomial, however, is crucial in our case, since it guarantees *exact* reproduction of linear functions, independent of the number n of kernels used. Due to our experiments even small errors in the linear reproduction (L^2 error around 10^{-3}) would cause ghost forces, thereby destroying the preservation of linear and angular momenta and resulting in counter-intuitive behavior.

The offset distance ξ in (12) has to be chosen heuristically. In our experiments, we found $\xi = 0.1$ to be a reliable setting, as similarly stated in [LGW*07]. Moreover, due to our dense sampling of collocation points $\mathbf{c}_i$ ($m \approx 3n$) the solution of the least squares system (14) is hardly influenced by ξ. In contrast, an exact interpolation ($m = n+3$) cannot prevent oscillations on the boundary between the $\mathbf{c}_i$ and would be much more sensitive to the offset distance (cf. Fig. 3).

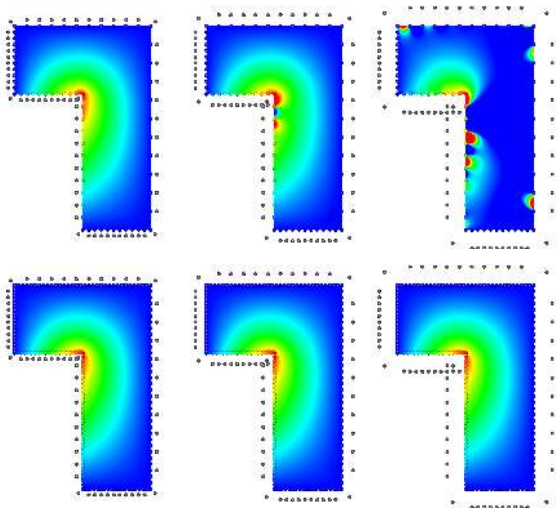

Figure 3: *Top: If we enforce the boundary conditions exactly, the solution oscillates and is very sensitive to the offset distance ($\xi = 0.05, 0.1, 0.15$). Bottom: In contrast, dense least-squares boundary conditions are more robust and considerably less affected by different parameter choices.*

Degenerate elements cause numerical problems for harmonic shape functions, just as they do for standard shape functions. Two (almost) *coincident kernels* $\mathbf{k}_i$, $\mathbf{k}_j$ lead to linearly dependent rows i, j in (14), yielding a rank deficient matrix. Simple cases, like the clustered kernels near the concave corner in Fig. 3, can be resolved explicitly by merging kernels, or implicitly through the SVD pseudo-inverse.

For *degenerate edges* with coincident nodes $\mathbf{x}_i$, $\mathbf{x}_j$, we conceptually merge $\mathbf{x}_i$ and $\mathbf{x}_j$ by computing one joint basis function ϕ_{ij}^e from the constraints $b_i + b_j$ in (8) and using a joint nodal displacement $\mathbf{u}_{ij} = \mathbf{u}_i = \mathbf{u}_j$. Note that this does not change the simulation mesh, and therefore preserves, e.g., planarity of faces. Almost planar *sliver elements* can be merged with their neighbors as proposed in [WBG07].

6. Finite Element Simulation

After introducing harmonic basis functions and their numerical approximation, we insert them into the Galerkin discretization of Section 3 and set up the matrix equations for the FEM simulation. We again give the main equations only, and refer the reader to [NMK*06, Hug00] for more details.

Once the basis functions N_i^e are computed as described in the previous section, the displacement $\mathbf{u}^e$ within e can be approximated as in (5), which can be written in matrix notation

$$\mathbf{u}^e(\mathbf{x}) := \sum_{i=1}^{k} N_i^e(\mathbf{x})\, \mathbf{u}_i = \mathbf{H}_e(\mathbf{x})\, \mathbf{U}_e,$$

with a $3 \times 3k$ matrix $\mathbf{H}_e(\mathbf{x})$ of basis function values $N_i^e(\mathbf{x})$ and a vector $\mathbf{U}_e = [\mathbf{u}_1^T, \dots, \mathbf{u}_k^T]^T$ of e's nodal displacements.

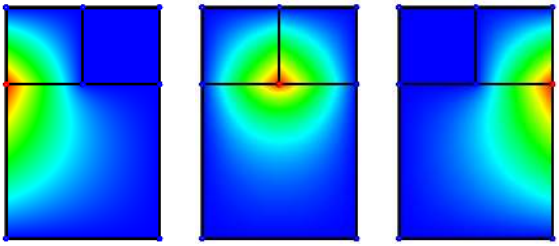

Figure 4: *A quadtree element with neighbors at a higher refinement level. Shown are the basis functions for the shared nodes, where the center one is not a hanging node, but instead is part (and DOF) of all three elements.*

Since the Cauchy strain is linear in the displacements, it can also be written in matrix notation as

$$\varepsilon\big(\mathbf{u}^e(\mathbf{x})\big) \;=\; \mathbf{B}_e(\mathbf{x})\,\mathbf{U}_e\,,$$

with a $6\times 3k$ matrix $\mathbf{B}_e(\mathbf{x})$ built from gradients $\nabla N_i^e(\mathbf{x})$. Using this matrix notation of each element's strain, the global elastic energy (6) of the deformed model can be written as

$$\mathrm{E}(\mathbf{u}) \;=\; \sum_e \mathbf{U}_e^T \underbrace{\left(\frac{1}{2}\int_e \mathbf{B}_e^T\,\mathbf{C}\,\mathbf{B}_e\,\mathrm{d}\mathbf{x}\right)}_{=:\,\mathbf{K}_e}\mathbf{U}_e \;=\; \frac{1}{2}\,\mathbf{U}^T\mathbf{K}\mathbf{U}\,, \quad (15)$$

with $\mathbf{K}_e$ denoting the $3k\times 3k$ stiffness matrix of element e, which are assembled into the global stiffness matrix $\mathbf{K}$, and $\mathbf{U}=[\ldots\mathbf{u}_i^T\ldots]^T$ the global vector of nodal displacements.

For a general polyhedral element e, the computation of its stiffness matrix $\mathbf{K}_e$ requires numerical integration, since the derivative matrix $\mathbf{B}_e(\mathbf{x})$ is not constant, as in the special case of linear tetrahedra. We employ either bounding box subdivision and Gaussian quadrature or Monte Carlo integration instead of the heuristic integration proposed by [WBG07], since the latter degrades for non-convex elements. For linear elasticity, the matrices $\mathbf{K}_e$ can be precomputed, such that the integration has to be performed only once for each element. As a consequence, the run-time complexity of our simulation is not higher than that of a simulation using a tetrahedral or hexahedral discretization.

Since the linear strain is not rotation-invariant, even rigid-body motions will give rise to strain, which in turn causes ghost forces. This can be remedied by adapting the stiffness warping of [MG04] to general polyhedral elements [WBG07]: The rotation $\mathbf{R}_e$ of the element's displacement $\mathbf{U}_e$ is extracted using shape matching, and is factored out by correcting the stiffness matrix as $\mathbf{K}_e \leftarrow \mathbf{R}_e\,\mathbf{K}_e\,\mathbf{R}_e^T$. This correction has to be performed in each time step, and the global stiffness matrix $\mathbf{K}$ needs to be updated accordingly. Again, the complexity for these computations is of the same order as for tetrahedral or hexahedral simulations.

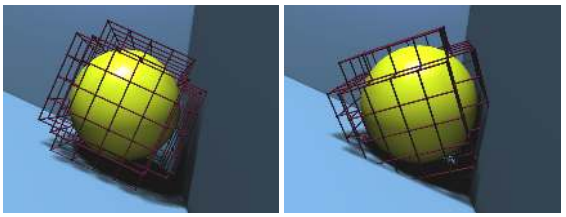

Figure 5: *Left: Collision handling on the simulation mesh's nodes. Right: Collision handling on the embedded surface.*

With the discrete energy (15), the governing equation for a dynamically deforming elastic solid becomes

$$\mathbf{M}\ddot{\mathbf{U}} + \mathbf{D}\dot{\mathbf{U}} + \mathbf{K}\mathbf{U} \;=\; \mathbf{F}_{\text{ext}}\,, \quad (16)$$

with mass matrix $\mathbf{M}$, damping matrix $\mathbf{D}$, and the vector $\mathbf{F}_{\text{ext}}$ containing external forces. We use a standard semi-implicit Euler method for the robust time-integration of (16). In our framework we employ simple nodal collision detection and handling, based on linear penalty forces being added to $\mathbf{F}_{\text{ext}}$ in order to resolve collisions. The collision handling for harmonic basis functions is performed in exactly the same way as for standard linear or trilinear basis functions.

7. Embedded Simulation

Using the method described so far, we can simulate deformable objects that are discretized by general polyhedral elements. However, a straightforward volume tessellation works for clean, moderately complex objects only, but becomes problematic for highly complex or topologically inconsistent models (e.g., scanned data, point-based models).

To be able to also handle such objects, we adapt a space embedding technique [FvdPT97, CGC*02, MG04, JBT04, SSIF07]. In a preprocessing step, we voxelize the object into hexahedral elements, and then simulate the elastic deformation on the resulting voxels only. The high resolution surface mesh is deformed by interpolating the displacement within the voxels according to (5). However, since the complexity of regular grids grows cubically under refinement, this approach can handle moderate grid resolutions only.

Exploiting the flexibility we gain from arbitrary element shapes, a hierarchical, *adaptive* refinement is very easy to implement. Similar to [BPWG07], we employ an octree-like discretization that refines nodes near the embedded surface. Note that this does not lead to hanging nodes in our discretization. Since the elements need not be strictly hexahedral, faces between octree cells of different depth do not require special handling, as illustrated in Fig. 4.

For embedded simulations we perform collision handling on the vertices of the embedded surface, instead of on the simulation nodes (cf. Fig. 5). Similar to [SSIF07], (penalty) forces applied to an embedded vertex $\mathbf{x}=\sum_i \mathbf{x}_i N_i(\mathbf{x})$ simply have to be distributed to the simulation nodes $\mathbf{x}_i$, weighted by the (generalized) barycentric coordinates $N_i^e(\mathbf{x})$.

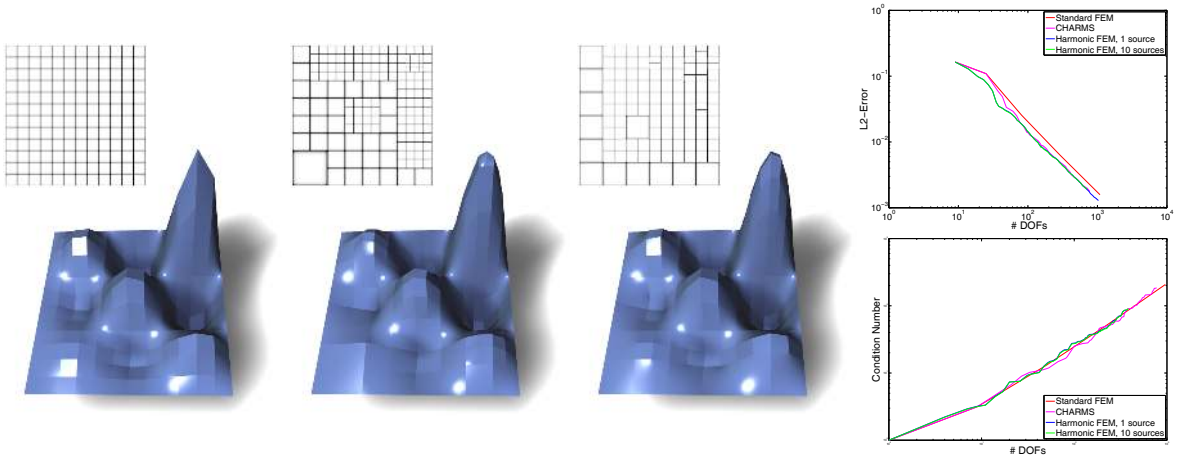

Figure 6: *For a 2D Poisson problem with known solution, we compare bilinear FEM (left), adaptive basis function refinement of CHARMS (middle), and our adaptive element refinement (right). The graphs compare approximations for grids of about the same number of DOFs. The plots show* L^2 *errors and condition numbers of* $\mathbf{K}$ *for increasing numbers of DOFs.*

8. Results

In this section we demonstrate the versatility of our polyhedral finite element framework on adaptive refinement and progressive cutting, and give statistics and comparisons of our method. Most examples show individual frames of simulations that are also included in the accompanying video.

Non-Convex Elements. As a proof of concept, Fig. 1 shows a simulation of a simple, non-convex element, whose shape functions are computed using $n = 97$ RBF centers in (11). While non-convex elements undoubtedly increase the meshing flexibility, we also note that in practice a restriction to convex elements might be preferable, for instance for efficient collision detection.

2D Adaptive Refinement. Fig. 6 shows a quantitative analysis of our method based on a 2D Poisson problem $-\Delta u = f$ with known analytic solution. We compare convergence behavior and condition numbers of uniform refinement of standard bilinear FEM, adaptive refinement of bilinear basis functions with CHARMS [GKS02], and our quadtree-like adaptive element refinement (cf. Fig. 4).

While condition numbers increase similarly with the number of DOFs for all methods, adaptive refinement makes better use of the DOFs than uniform refinement. When comparing CHARMS to our refinement technique, the former can also be used for higher order basis functions, whereas our method is a generalization of linear shape functions only. However, it allows for more flexible splits without the need to balance neighboring elements' refinement levels (see below). The plots also show that the number of sources $\mathbf{s}_i$ (resp. of RBF kernels $\mathbf{k}_i$) used for individual basis functions N_i^e has practically no effect on the global approximation error.

Adaptive Mesh Generation. In Fig. 7, the Stanford bunny is embedded in an adaptive octree-like simulation mesh, which concentrates the DOFs at the more interesting boundary surface. Supporting general polyhedral elements makes this kind of adaptive embedding both easy to implement and to maintain, thereby enabling the efficient simulation of highly complex or topologically inconsistent meshes.

Stress-Based Dynamic Refinement. Polyhedral elements allow for dynamic element refinement: Similar to fracture [OH99, OBH02], we sample the stress tensor $\sigma(\mathbf{x})$ at a few points within each element, compute the principal stress as the largest absolute eigenvalue, and refine an element once a certain threshold is reached. Fig. 8 illustrates this for two kinds of adaptive refinement: A uniform 1-to-8 subdivision of voxels, and the more flexible 1-to-2 splitting perpendicular to the maximum stress direction, which results in fewer elements for the same refinement threshold.

Progressive Cutting. As proposed in [WBG07], supporting general polyhedra in FEM simulations effectively avoids the need for complex remeshing during cutting and thus considerably simplifies the implementation. Our harmonic basis functions seamlessly integrate into both tetrahedral and hexahedral simulations, where then only the cut elements have to be computed as harmonic polyhedral elements (cf. Fig. 9). Arbitrary cuts can lead to non-convex elements with small opening angles, which complicates off-setting RBF centers. To avoid this problem, and to simplify the actual element splitting and collision detection, our cutting algorithm generates convex elements only, following [WBG07].

Timings. For the examples shown, Table 1 summarizes model complexities and timings, taken on an Intel Core2 Duo, 2.4 GHz. Solving the linear systems takes about the

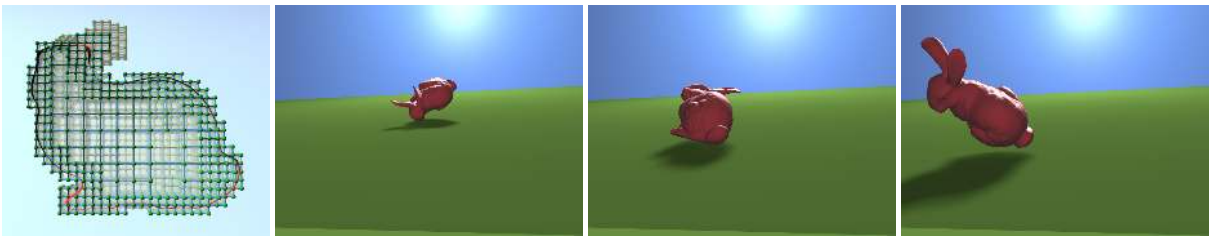

Figure 7: *The bunny model is embedded into an adaptively refined, octree-like simulation mesh, shown on the left. The degrees of freedom are concentrated on the surface, wasting little computing power on the less interesting, invisible interior.*

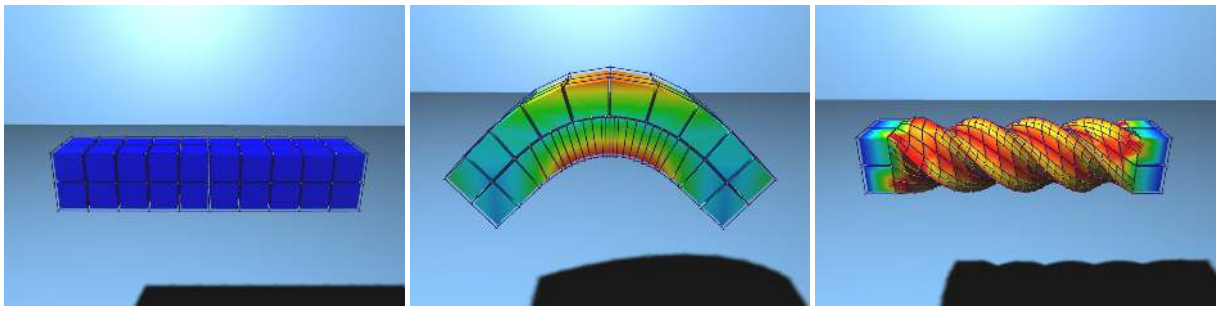

Figure 8: *Dynamic, stress-based refinement of a hexahedral bar model, using 1-to-2 splits for the bending deformation and 1-to-8 subdivision for twisting. The bar is constrained at both ends, the color visualizes the maximum principal stress.*

same time as for standard FEM, with only a slight increase in matrix density in case of complex polyhedra with high vertex count. Solving for and numerically integrating shape functions N_i^e is considerably more expensive than for simple linear tetrahedra or trilinear hexahedra. Note, however, that general polyhedral elements are employed in irregular regions of adaptivity and cutting only, whereas in regular regions we can use standard elements. Our approach therefore trades the combinatorial complexity of remeshing for the computational complexity of polyhedral elements.

9. Conclusion

We have introduced an FEM framework for arbitrary polyhedral elements based on harmonic basis functions, and proposed the method of fundamental solutions as a simple and flexible method for computing these basis functions. Being able to use general polyhedral elements in FEM simulations considerably simplifies topological changes of the simulation domain, as illustrated for adaptive mesh generation, dynamic refinement, and progressive cutting.

While we demonstrated harmonic polyhedral elements mainly in the context of corotated linear elasticity, it is important to note that they can as well be used with nonlinear strain measures and nonlinear material behavior, which therefore constitutes an interesting direction for future work. Furthermore, extending our approach to both adaptive and hierarchical discretizations and solvers has the potential to improve runtime performance of the simulation.

Acknowledgments. This research was supported by the Swiss National Science Foundation grant 200021-117756.

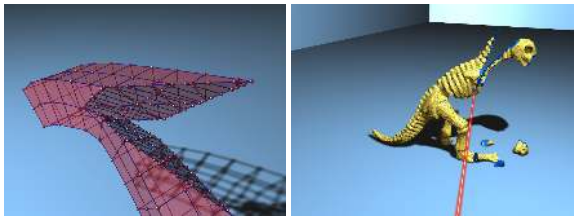

Figure 9: *Left: Progressive cutting of a hexahedral bar model. Right: Cutting a tetrahedral dinosaur mesh. Tetrahedra are visualized in yellow, general polyhedra in blue.*

Scene	Start #N/#E	End #N/#E	t_{init}	t_{solve}	t_{total}
Collision (Fig. 5)	274 / 153	274 / 153	105	5.2	20
Bunny (Fig. 7)	4.8k / 3k	4.8k / 3k	59	221	247
Bending (Fig. 8)	99 / 40	256 / 88	302	1.6	60
Twisting (Fig. 8)	99 / 40	1392 / 719	170	37	276
Bar Cut (Fig. 9)	99 / 40	391 / 63	688	3	235
Dino Cut (Fig. 9)	5.6k / 19k	9.3k / 21k	48	113	752

Table 1: *Statistics and timinigs for the examples shown in this paper. We list initial and final number of nodes (#N) and elements (#E), avg. time to compute N_i^e and setup $\mathbf{K}_e$ per* polyhedral *element, and for the linear solve per time-step (both [ms]), and the total simulation time [s].*

References

[BPWG07] BOTSCH M., PAULY M., WICKE M., GROSS M.: Adaptive space deformations based on rigid cells. *Computer Graphics Forum (Proc. Eurographics) 26*, 3 (2007), 339–347.

[CGC*02] CAPELL S., GREEN S., CURLESS B., DUCHAMP T., POPOVIĆ Z.: A multiresolution framework for dynamic deformations. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2002), pp. 41–47.

[Chu96] CHUNG T. J.: *Applied Continuum Mechanics*. Cambridge University Press, New York, 1996.

[DDCB01] DEBUNNE G., DESBRUN M., CANI M.-P., BARR A. H.: Dynamic real-time deformations using space and time adaptive sampling. In *Proceedings of SIGGRAPH'01* (2001), pp. 31–36.

[Duc77] DUCHON J.: Spline minimizing rotation-invariant seminorms in Sobolev spaces. In *Constructive Theory of Functions of Several Variables*, Schempp W., Zeller K., (Eds.), no. 571 in Lecture Notes in Mathematics. Springer Verlag, 1977, pp. 85–100.

[FK98] FAIRWEATHER G., KARAGEORGHIS A.: The method of fundamental solutions for elliptic boundary value problems. *Advances in Computational Mathematics 9*, 1–2 (1998), 69–95.

[FKR05] FLOATER M. S., KOS G., REIMERS M.: Mean value coordinates in 3d. *Computer Aided Geometric Design 22* (2005), 623–631.

[FvdPT97] FALOUTSOS P., VAN DE PANNE M., TERZOPOULOS D.: Dynamic free-form deformations for animation synthesis. *IEEE Transactions on Visualization and Computer Graphics 3*, 3 (1997), 201–214.

[GKS02] GRINSPUN E., KRYSL P., SCHRÖDER P.: Charms: A simple framework for adaptive simulation. In *Proceedings of SIGGRAPH'02* (2002), pp. 281–290.

[GL89] GOLUB G. H., LOAN C. F. V.: *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1989.

[Hug00] HUGHES T. J. R.: *The Finite Element Method. Linear Static and Dynamic Finite Element Analysis*. Dover Publications, 2000.

[JBT04] JAMES D., BARBIČ J., TWIGG C.: Squashing cubes: Automating deformable model construction for graphics. In *Proceedings of SIGGRAPH '04 Sketches and Applications* (2004).

[JLW07] JU T., LIEPA P., WARREN J.: A general geometric construction of coordinates in a convex simplicial polytope. *Computer Aided Geometric Design* (2007). preprint.

[JMD*07] JOSHI P., MEYER M., DEROSE T., GREEN B., SANOCKI T.: Harmonic coordinates for character articulation. *ACM Trans. on Graphics (Proc. SIGGRAPH) 26*, 3 (2007), 71.

[JSW05] JU T., SCHAEFER S., WARREN J.: Mean value coordinates for closed triangular meshes. In *Proceedings of SIGGRAPH'05* (2005), pp. 561–566.

[LGW*07] LI X., GUO X., WANG H., HE Y., GU X., QIN H.: Harmonic volumetric mapping for solid modeling applications. In *Proceedings of symposium on Solid and physical modeling* (2007), pp. 109–120.

[MBF04] MOLINO N., BAO Z., FEDKIW R.: A virtual node algorithm for changing mesh topology during simulation. In *Proceedings of SIGGRAPH'04* (2004), pp. 385–392.

[MG04] MÜLLER M., GROSS M.: Interactive virtual materials. In *Proceedings of Graphics Interface'04* (2004), pp. 239–246.

[MKN*04] MÜLLER M., KEISER R., NEALEN A., PAULY M., GROSS M., ALEXA M.: Point-based animation of elastic, plastic and melting objects. In *Proceedings of the Symp. on Computer Animation'04* (2004), pp. 141–151.

[MTG04] MÜLLER M., TESCHNER M., GROSS M.: Physically based simulation of objects represented by surface meshes. In *Proceedings of Computer Graphics Int.'04* (2004), pp. 26–33.

[NMK*06] NEALEN A., MULLER M., KEISER R., BOXERMAN E., CARLSON M.: Physically based deformable models in computer graphics. *Computer Graphics Forum 25*, 4 (2006), 809–836.

[OBH02] O'BRIEN J. F., BARGTEIL A. W., HODGINS J. K.: Graphical modeling and animation of ductile fracture. In *Proceedings of SIGGRAPH'02* (2002), pp. 291–294.

[OGRG07] OTADUY M. A., GERMANN D., REDON S., GROSS M.: Adaptive deformations with fast tight bounds. In *Proceedings of the Symp. on Computer Animation'07* (2007), pp. 181–190.

[OH99] O'BRIEN J. F., HODGINS J. K.: Graphical modeling and animation of brittle fracture. In *Proceedings of SIGGRAPH'99* (1999), pp. 137–146.

[PKA*05] PAULY M., KEISER R., ADAMS B., DUTRé P., GROSS M., GUIBAS L. J.: Meshless animation of fracturing solids. In *Proceedings of SIGGRAPH'05* (2005), pp. 957–964.

[SDF07] SIFAKIS E., DER K. G., FEDKIW R.: Arbitrary cutting of deformable tetrahedralized objects. In *Proceedings of the Symp. on Computer Animation'07* (2007), pp. 73–80.

[SOG06] STEINEMANN D., OTADUY M. A., GROSS M.: Fast arbitrary splitting of deforming objects. In *Proceedings of the Symp. on Computer Animation'06* (2006), pp. 63–72.

[SSIF07] SIFAKIS E., SHINAR T., IRVING G., FEDKIW R.: Hybrid simulation of deformable solids. In *Proceedings of the Symp. on Computer Animation'07* (2007), pp. 81–90.

[TF88] TERZOPOULOS D., FLEISCHER K.: Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. In *Proceedings of SIGGRAPH'88* (1988), pp. 269–278.

[TPBF87] TERZOPOULOS D., PLATT J., BARR A., FLEISCHER K.: Elastically deformable models. In *Proceedings of SIGGRAPH'87* (1987), pp. 205–214.

[Wac75] WACHSPRESS E. L.: *A Rational Finite Element Basis*. Academic Press, 1975.

[War96] WARREN J.: Barycentric coordinates for convex polytopes. *Advances in Computational Mathematics 6* (1996), 97–108.

[WBG07] WICKE M., BOTSCH M., GROSS M.: A finite element method on convex polyhedra. In *Proceedings of Eurographics'07* (2007), pp. 355–364.