Robust eXtended Finite Elements for Complex Cutting of Deformables

DAN KOSCHIER, RWTH Aachen University
JAN BENDER, RWTH Aachen University
NILS THUEREY, Technical University of Munich

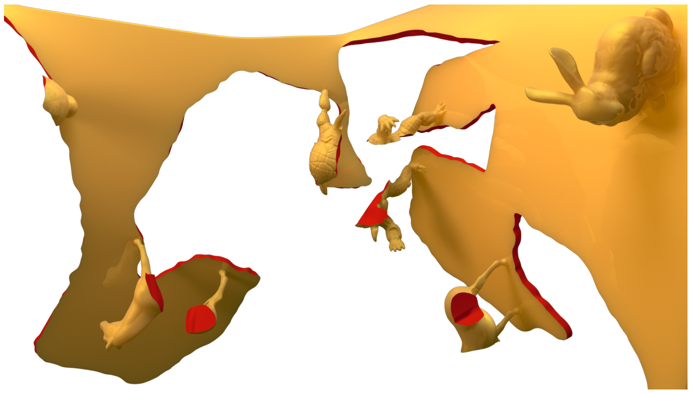
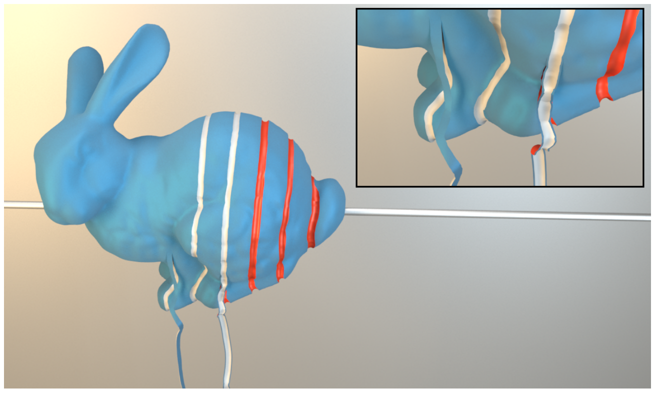



Fig. 1. Left: Plate with attached objects consisting of 159k tetrahedra is cut by long, bumpy cut surface. Right: A groove is carved into the Stanford bunny (18.5k tetrahedra) rotating around a fixed axis. Subsequently, the groove itself is peeled off by a second cut.

In this paper we present a robust remeshing-free cutting algorithm on the basis of the e**X**tended **F**inite **E**lement **M**ethod (XFEM) and fully implicit time integration. One of the most crucial points of the XFEM is that integrals over discontinuous polynomials have to be computed on subdomains of the polyhedral elements. Most existing approaches construct a cut-aligned auxiliary mesh for integration. In contrast, we propose a cutting algorithm that includes the construction of specialized quadrature rules for each dissected element without the requirement to explicitly represent the arising subdomains. Moreover, we solve the problem of ill-conditioned or even numerically singular solver matrices during time integration using a novel algorithm that constrains non-contributing degrees of freedom (DOFs) and introduce a preconditioner that efficiently reuses the constructed quadrature weights.

Our method is particularly suitable for fine structural cutting as it decouples the added number of DOFs from the cut's geometry and correctly preserves geometry and physical properties by accurate integration. Due to the implicit time integration these fine features can still be simulated robustly using large time steps. As opposed to this, the vast majority of existing approaches either use remeshing or element duplication. Remeshing based methods are able to correctly preserve physical quantities but strongly couple cut geometry and mesh resolution leading to an unnecessary large number of additional DOFs. Element duplication based approaches keep the number of additional DOFs small but fail at correct conservation of mass and stiffness properties. We verify consistency and robustness of our approach on simple and reproducible academic examples while stability and applicability are demonstrated in large scenarios with complex and fine structural cutting.

CCS Concepts: • **Computing methodologies** → **Physical simulation**;

Additional Key Words and Phrases: Finite elements, cutting, numerical integration

ACM Reference format:
Dan Koschier, Jan Bender, and Nils Thuerey. 2017. Robust eXtended Finite Elements for Complex Cutting of Deformables. *ACM Trans. Graph.* 36, 4, Article 55 (July 2017), 13 pages.
DOI: http://dx.doi.org/10.1145/3072959.3073666

1 INTRODUCTION

The animation of cutting of deformable bodies has been an active research topic for several years and is an essential tool in applications such as virtual surgery, special effects in feature films and computer games. Particularly challenging is the robust treatment of complex cut surfaces while maintaining physical plausibility at all times. Moreover, it is crucial to keep the computational overhead minimal even in highly complex cutting scenarios.

Most mesh-based approaches for the animation of cutting rely on element deletion, element duplication or adaptive remeshing in order to capture the cut's geometry within the simulation mesh. On the one hand, pure deletion or duplication of elements leads to a very robust simulation and is easy to implement but does not correctly represent the underlying cut geometry and introduces physical inconsistencies due to unintended loss or addition of mass and stiffness. On the other hand, proper remeshing of the affected regions correctly models the cut geometry and maintains a physically correct behavior. However, this class of approaches typically adds a large number of additional degrees of freedom (DOFs), in order to sufficiently embed the cut's geometry into the simulation mesh. This in turn leads to a substantial rise of computational cost especially when equation systems have to be solved for implicit time integration. Moreover, all persistent physical quantities, e.g. plastic deformation, temperature etc., stored on the elements have to be transferred to the new mesh. Besides the extra computational effort associated with the transfer process, this usually leads to unwanted diffusion effects, as discussed by Wicke et al. [2010].

In order to overcome these issues, we present a novel method for the physical simulation of cutting based on the e**X**tended **F**inite **E**lement **M**ethod (XFEM) with fully implicit time integration. Without the requirement to apply any topological changes to the simulation mesh, we capture the cut geometry with subcell accuracy using an appropriate enrichment function. This means that even for highly complex cuts, no additional geometry has to be created using our algorithm, and thus, geometric processing overhead is kept at a minimum. Following this strategy the overall algorithm is greatly simplified, as we only need to consider the enrichments during the assembly of the linear equation system for implicit time integration.

One of the key problems associated with the XFEM is the evaluation of integrals over discontinuous integrands on polyhedral domains. These are required to compute discretized force vectors as well as mass and tangent stiffness matrices. In order to solve the problem we propose a remeshing-free method to construct specialized quadrature rules for a particular cut surface. Using these rules we are able to accurately compute the desired integrals. Due to the accurate integration, we are able to completely maintain physical properties, i.e. mass and stiffness properties, during cut insertion.

Moreover, we keep the simulation numerically robust by introducing an algorithm that constrains nearly non-contributing DOFs and by introducing a method to precondition the solver matrix by direct use of the constructed quadrature weights. We demonstrate our method's consistency and advantages over existing approaches on the basis of easily reproducible, academic examples. Additionally, we thoroughly test robustness and applicability of our approach in highly complex scenarios with fine-structural cuts (see Figure 1).

2 RELATED WORK

The physically-based simulation of virtual cutting has been investigated for several years. Moreover, most methods are based on simulation methods for deformable solids which have been an active research topic for nearly three decades. For a general overview over simulation methods for deformable solids we would like to refer the reader to the survey of Nealen et al. [2006]. A state-of-the-art report for physically based cutting of deformable objects was presented by Wu et al. [2015].

In this section methods related to the presented approach will be reviewed. As our method is targeted towards applications in the field of computer graphics but based on XFEM originating from mechanical engineering, we organized the discussion respectively.

Cutting and Fracture in Computer Graphics. Methods for physically based simulation of cutting and fracture are strongly related as both applications require modeling of discontinuities within a solid. In this paragraph, we will therefore review recent developments in both areas.

In the pioneering work of O'Brien and Hodgins [1999] on brittle fracture, discontinuities are modeled by explicitly embedding the cut surface into the underlying tetrahedral discretization by static local remeshing rules. The concept was then adopted by O'Brien et al. [2002] for the animation of ductile fracture. A similar tetrahedral mesh based concept was developed by Bielser et al. [1999]. They embed the cut surface by static local remeshing using tetrahedral subdivision followed by topological disconnection of the subtetrahedra due to lookup-table entries. The concept was first improved by Bielser and Gross [2000] by introducing case-specific subdivision rules to reduce the number of added tetrahedra and was later generalized using a state-machine [Bielser et al. 2004]. Kaufmann et al. [2008] split cut elements, and continue the simulation directly on the resulting polyhedra. Moreover, they correctly account for volume integrals over polynomials on the polyhedra by applying the divergence-theorem and integrating over element faces and edges.

A particular disadvantage of all previously mentioned approaches is that the mesh quality can easily deteriorate due to the static remeshing. Therefore, Steinemann et al. [2006] decouple visual representation and simulation mesh and split the former only on existing faces. Following the same splitting strategy, Yeung et al. [2016] presented a static finite-element method for linear elastic materials featuring fast updates through matrix augmentation. While this strategy guarantees the conservation of mesh quality the approximation can be arbitrarily inaccurate especially for highly-detailed cut surfaces. In a more elaborate approach Wicke et al. [2010] maintain the mesh quality using a dynamic local remeshing algorithm involving topological splits and flips followed by vertex smoothing. A similar two-dimensional approach for tearing of thin sheets was proposed by Pfaff et al. [2014].

As opposed to embedding the cut surface into a tetrahedral mesh, Dick et al. [2010] discretize the solid using a regular hexahedral grid and approximately embed the cut by simple separation of elements subsequent to octree refinement. The method was later extended by Wu et al. [2011] using a composite finite-element discretization. An approach based on polyhedral finite elements was proposed by Martin et al. [2008]. Starting from a tetrahedral or hexahedral mesh, they locally split cut elements into polyhedra and construct harmonic basis functions on each subelement. In contrast to local adaptions, Busaryev et al. [2013] proposed an approach for fracture simulation of multi-layered thin plates using global constrained delaunay remeshing.

In order to maintain mesh quality and decouple the mesh resolution from the cut geometry, Molino et al. [2004] developed a Virtual Node Algorithm (VNA). The method is based on cell duplication resulting in a possibly non-manifold mesh topology that correctly accounts for the discontinuities and was adopted in several works, e.g. [Bao et al. 2007; Koschier et al. 2014; Mitchell et al. 2015]. However, the method is not able to handle an arbitrary number of cuts per tetrahedron resulting in a maximally-split configuration. This limitation was later addressed by Sifakis et al. [2007] by embedding a polygonal auxiliary mesh and by Wang et al. [2014] using an adaptive approach.

Despite the fact that most attention was paid to Lagrangian mesh based discretizations, Pauly et al. [2005] developed a meshless technique for fracturing solids using a transparency criterion while Hegemann et al. [2013] model discontinuities using level-sets in reference space using a hexahedral Eulerian background grid.

In contrast to the discussed approaches we present a Lagrangian mesh-based but remeshing-free simulation method. Using the enrichment concept of the XFEM, we are able to keep the number of additional DOFs low. Moreover, we are still able to accurately represent the shape of the cut within the discretization while correctly maintaining physical properties as opposed to methods based on cell duplication.

eXtended Finite Elements and VNAs in Engineering. In this paragraph we will give a brief overview over related methods developed in the field of mechanical engineering. For a detailed discussion of XFEMs for the simulation of solids we would like to refer the reader to the survey of Fries et al. [2010].

The concept of enriching finite element discretizations by discontinuous functions was first proposed by Belytschko and Black [1999] and later referred to as XFEM. They apply the method to model elastic crack growth in two-dimensional solids using enrichment functions based on asymptotic displacement fields near the crack tip. The method was later extended by Moës et al. [1999] for the simulation of long cracks using the piecewise-constant but discontinuous Heaviside enrichment, by Daux et al. [2000] for arbitrarily branching and intersecting cracks. A method based on harmonic enrichments was proposed by Mousavi et al. [2011a] in order to treat multiple, intersecting and branched cracks in a unified manner. They later extended the method for higher-order approximations [Mousavi et al. 2011b].

A, at first glance, highly similar method to XFEM approaches for discontinuities was proposed by Hansbo and Hansbo [2004]. They construct independent approximation bases for cut elements on each side of the cut. However, in a report of Areias and Belytschko [2006] it was proven that the kinematic decomposition in their method is in fact equivalent to the earlier proposed XFEM [Belytschko and Black 1999; Moës et al. 1999]. Moreover, we discovered a very interesting connection to the VNA, as cell and node duplication is equivalent to the basis modification proposed by Hansbo and Hansbo [2004]. For that reason, the general kinematic decomposition of the VNA and the XFEM is identical. However, major differences are that traditional VNA based approaches do not correctly account for integrals over the discontinuous functions and that the XFEM's canonical basis can be trivially extended by further enrichments for different phenomena.

A particular challenge in implementing XFEMs is the evaluation of integrals over discontinuous functions arising due to the enrichment strategy, since no standard quadrature rules are applicable anymore. The naïve approach to circumvent this issue is to generate a discontinuity-aligned submesh on the dissected elements and perform Gauss-Legendre quadrature (cf. [Moës et al. 1999]). However, as one of the main aims of the XFEM is to remove the necessity of remeshing, this strategy is often not desired. In an XFEM-based method for brittle fracture Richardson et al. [2011] correctly account for discontinuous integrands. They generate an explicit discontinuity-aligned boundary mesh and compute surface integrals using the divergence-theorem. VNA-based approaches following the same strategy were also developed in the engineering community [Bedrossian et al. 2010; Hellrung et al. 2012; Schroeder et al. 2014; Zhu et al. 2012]. Over the years several approaches based on equivalent polynomials [Ventura 2006], adaptive quadrature [Müller et al. 2012] or variable quadrature weights [Holdych et al. 2008] were developed. As adaptive rules are computationally expensive and as the remaining methods expect element-wise straight cuts, an approach based on hierarchical moment-fitting using predefined quadrature nodes was proposed by Müller et al. [2013] for discontinuities represented by higher-order level-sets. The method was later extended by a modification of the placement of the quadrature nodes [Müller et al. 2017]. The method is highly suitable for smooth cut surfaces but suffers from large errors for non-smooth, kinked cut surfaces.

In our work, we represent the discontinuity, i.e. the cut surface, by an explicit triangle mesh and present an approach for the evaluation of integrals over discontinuous polynomials. Similar to the level-set approaches Müller et al. [2017; 2013] we hierarchically construct specialized quadrature rules for accurate integration. However, our method builds on an explicit representation of the discontinuity where we place quadrature nodes such that curved and even kinked cut surfaces are accurately captured. Moreover, we allow the quadrature nodes to lie outside the actual finite element in order to avoid geometric operations and to reduce complexity.

eXtended Finite Elements in Graphics. The XFEM was also adopted by some works in the field of computer graphics. Jeřábková et al. [2009] developed an XFEM based simulator for interactive surgery using shifted sign enrichment. Moreover, they presented a novel method for mass-lumping in order to maintain a stable simulation based on explicit time integration. Kaufmann et al. [2009] introduced a simulation method for cutting of two-dimensional shells based on a discontinuous Galerkin finite element discretization and a semi-implicit Euler method for time integration. In order to model progressive cuts, they construct harmonic enrichments by solving the Laplace equation on textures using the GPU. The solution of the Laplace equation then mimics the behavior of asymptotic crack tip functions (cf. [Belytschko and Black 1999]). However, for each cut at least one Laplace equation has to be discretized, solved and stored in a texture resulting in a considerable computational and memory effort. Also, a numerical evaluation of the required integrals over the enriched region can be computationally expensive due to the yielded discontinuous, non-polynomial enrichment function. While Kaufmann et al. demonstrate that this strategy is still effective for two-dimensional structures, the computational effort and memory requirements become dominant for three-dimensional simulations. Moreover, the texture resolution poses a strict limitation on how fine-structural the geometry of a cut is allowed to be.

In this work we will present a novel method using fully implicit time integration. We overcome stability issues due to ill-conditioned matrices using our approach by introducing an effective preconditioner that directely reuses previously computed quadrature weights. Finally, we will demonstrate that our method produces stable results even in the case of large time-steps.

3 GOVERNING EQUATIONS

Let $\mathbf{u} : \Omega \times [0, \infty) \to \mathbb{R}^3$ be the displacement function that maps a material point ξ in the reference domain $\Omega \subset \mathbb{R}^3$ at time $t \in [0, \infty)$ to its displacement vector $\mathbf{u}$ pointing to the deformed location in world space. Then the deformation gradient is defined as $\mathbf{F} = \mathbf{I} + \nabla_\xi \mathbf{u}$, where $\mathbf{I}$ is the identity matrix. The mixed initial/boundary value problem describing the deformation behavior of a cut solid is defined

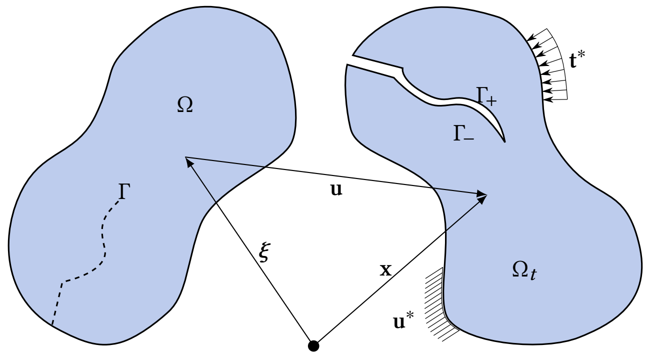


Fig. 2. Illustration of cut deformable continuum in reference (left) and world space (right).

as

$$\begin{aligned} \rho_0 \ddot{\mathbf{u}} &= \nabla_\xi \cdot \mathbf{P} + \mathbf{b} \\ \mathbf{u}(\xi, 0) &= \mathbf{u}_0(\xi) \\ \mathbf{u}(\xi, t) &= \mathbf{u}^*(\xi, t) \quad \text{on } \partial\Omega_u \\ \mathbf{t}(\xi, t) &= \mathbf{t}^*(\xi, t) \quad \text{on } \partial\Omega_t \cup \Gamma_\pm \end{aligned} \tag{1}$$

where $\mathbf{P}, \rho_0, \mathbf{b}, \mathbf{t}, \partial\Omega_u, \partial\Omega_t$ denote first Piola-Kirchhoff stress tensor, reference density, reference body force, traction and boundary with displacement as well as traction conditions, respectively. Further, Γ, Γ_- and Γ_+ represent the cut surface in reference space and the arising cut sides in world space while $\mathbf{u}^*$ and $\mathbf{t}^*$ denote displacement and traction boundary conditions (see Figure 2). As problems in computer graphics applications are usually stated without traction forces, we presume a traction free state with $\mathbf{t}^* = \mathbf{0}$ in the following. However, note that our formulation can also be extended to handle the case of $\mathbf{t}^* \neq \mathbf{0}$. Consequently, we develop the weak formulation of the problem which yields

$$\int_\Omega \rho_0 \ddot{\mathbf{u}} \cdot \mathbf{w} d\xi + \int_\Omega \mathbf{P} : \nabla_\xi \mathbf{w} d\xi = \int_\Omega \mathbf{b} \cdot \mathbf{w} d\xi, \tag{2}$$

where $\mathbf{w}$ denotes a test function. According to the Galerkin methodology of using the same function space for test functions as well as trial functions, the test function can be physically interpreted as virtual displacement $\mathbf{w} = \delta\mathbf{u}$ strongly fulfilling all essential boundary conditions. Note that the resulting formulation is often referred to as *Principle of Virtual Work*. In order to relate displacement and stress we use a hyperelastic material model where an energy density function Ψ is assumed such that $\mathbf{P} = \nabla_\mathbf{F}\Psi$. More specifically we used the St. Venant-Kirchhoff material model to compute all of our results. Please note that the formulation is not restricted to this specific material model and could easily be replaced by different hyperelastic models as discussed by Sin et al. [2013], or Sifakis and Barbic [2012].

4 DISCRETIZATION

In this section we will first briefly recapitulate the standard finite element discretization using linear Lagrange polynomials. Subsequently, we will discuss how we represent the cut surface's geometry and finally introduce the concept of basis enrichment. This enrichment will allow us to capture the discontinuities in the PDE's solution implied by the physical cuts.

4.1 Standard Finite Element Discretization

Consider a three-dimensional domain $\Omega \subset \mathbb{R}^3$ discretized using n_el elements. Let further $\mathcal{I}$ be the set of n_v nodes shared by the elements. Then a field function $\mathbf{u}$ is approximated by $\mathbf{u}^h$ using the given basis as

$$\mathbf{u}(\xi) \approx \mathbf{u}^h(\xi) = \sum_{i \in \mathcal{I}} N_i(\xi)\mathbf{u}_i, \quad \text{with } \xi \in \Omega, \tag{3}$$

where $N_i(\xi) : \mathbb{R}^3 \to \mathbb{R}$ is the shape function and $\mathbf{u}_i$ the field coefficient of the ith node. As we presume that Lagrange polynomial shape functions are used, they provide both the *Kronecker-δ Property* $N_i(\xi_j) = \delta_{ij}$ and the *Partition of Unity Property* $\sum_{i \in \mathcal{I}} N_i(\xi) = 1$ $\forall\ \xi \in \Omega$, where ξ_j represents the material coordinate of the jth node. For these reasons the approximation field is interpolating; hence, the field coefficients $\mathbf{u}_i$ can be interpreted as nodal displacements by means of a displacement field. Semi-discretizing Equation (2) using Equation (3) then yields

$$\mathbf{M}\ddot{\bar{\mathbf{u}}} + \mathbf{f}^\text{int} = \mathbf{f}^\text{ext}, \ \bar{\mathbf{u}} = \left(\mathbf{u}_1^T \ldots \mathbf{u}_{n_v}^T\right)^T \tag{4}$$

$$\sum_{e=1}^{n_\text{el}} \mathbf{m}_e \ddot{\bar{\mathbf{u}}} + \sum_{e=1}^{n_\text{el}} \mathbf{f}_e^\text{int} = \sum_{e=1}^{n_\text{el}} \mathbf{f}_e^\text{ext} \tag{5}$$

$$\mathbf{m}_e = \int_{\Delta_e} \rho_0 \mathbf{N}_e^T \mathbf{N}_e d\xi \tag{6}$$

$$\mathbf{f}_e^\text{ext} = \int_{\Delta_e} \mathbf{N}_e^T \mathbf{b} d\xi, \qquad \mathbf{f}_e^\text{int}(\bar{\mathbf{u}}) = \int_{\Delta_e} \mathbf{g}_e(\bar{\mathbf{u}}, \xi) d\xi \tag{7}$$

$$\mathbf{N}_e = \begin{bmatrix} N_0\mathbf{I}_3 & N_1\mathbf{I}_3 & N_2\mathbf{I}_3 & N_2\mathbf{I}_3 \end{bmatrix}, \tag{8}$$

where $\mathbf{M}, \mathbf{m}_e, \mathbf{N}_e, \mathbf{g}_e, \mathbf{f}_e^\text{int}, \mathbf{f}_e^\text{ext}$ denote the mass and element mass matrix, element shape function vector, specific elastic element force as well as the internal and external element force vector, respectively. Furthermore, $\Delta_e \subset \Omega$ represents the subdomain of the eth finite element with $\overline{\Omega} \approx \overline{\Omega}^h = \bigcup_e \overline{\Delta}_e$ and $\Delta_i \cap \Delta_j = \emptyset, \ \forall\ i \neq j$, where $\overline{\Omega}$ and $\overline{\Delta}_e$ are the closures of Ω and Δ_e, respectively. To finally discretize Equation (4) in time we employ a fully implicit Backward Euler method and add an additional Rayleigh damping term as suggested by Sin et al. [2013].

4.2 Cut Representation

In order to construct the enrichments, the location and geometry of each cut has to be defined. Especially in engineering, implicit cut representations, e.g. level-sets defined on grids or on the simulation mesh, are very popular. However, implicit representations are not well-suited to represent sharp features and are very memory consuming. Moreover, it is more than non-trivial to detect if, where and how often an implicit surface has dissected a tetrahedron. For the stated reasons we represent all cut surfaces as explicit triangle meshes. We denote the surface of the jth cut Γ^j while we further define its corresponding signed distance function

$$\Phi^j(\xi) = s(\xi) \inf_{\xi^* \in \Gamma^j} \|\xi - \xi^*\|, \tag{9}$$

where $s : \mathbb{R}^3 \rightarrow \{-1, 1\}$ determines the sign of the distance. Using this information we can now augment the approximation in Equation (3) in order to capture the discontinuity in the PDE's numerical solution as discussed in the next section.

4.3 XFEM Discretization

The main strategy of the (extrinsic) XFEM is to extend an existing polynomial approximation space by additional enrichment functions to capture certain features, e.g. strong or weak discontinuities, and to improve the quality of the PDE's numerical solution. Moreover, we would like to stress the fact that there is no need to perform any modification on the underlying mesh even when additional degrees of freedom are added. In order to keep the support of the enrichment functions local and to keep the *Kronecker-δ Property* and therefore the approximation field interpolating at nodes, we employ a *Shifted Sign Enrichment* for strong discontinuities. For m discontinuities $\Gamma_1, \ldots, \Gamma_m$ the discretized displacement field introduced in Equation (3) is extended yielding

$$\begin{aligned}
\mathbf{u}^h_{\text{XFEM}}(\xi) &= \sum_{i \in \mathcal{V}_0} N_i(\xi)\mathbf{u}^0_i + \sum_{j=1}^{m} \sum_{i \in \mathcal{V}_j} \psi^j_i(\xi) N_i(\xi) \mathbf{u}^j_i \\
&= \sum_{j=0}^{m} \sum_{i \in \mathcal{V}_j} \psi^j_i(\xi) N_i(\xi) \mathbf{u}^j_i, \\
\psi^j_i(\xi) &= \begin{cases} 1 & j = 0 \\ \frac{1}{2}\left(\operatorname{sgn}(\Phi^j(\xi)) - \operatorname{sgn}(\Phi^j(\xi_i))\right) & \text{otherwise,} \end{cases} \\
\operatorname{sgn}(x) &:= 2H(x) - 1,
\end{aligned} \tag{10}$$

where ψ^j_i, $\mathbf{u}^j_i$ denote the shifted enrichment function and the discontinuous part of the nodal displacement of the ith node and the jth cut, respectively, while $\mathbf{u}^0_i = \mathbf{u}_i$. Furthermore, $\mathcal{V}_0$ is the set of all mesh nodes while $\mathcal{V}_j$ represents the set of enriched nodes shared by the finite elements $\mathcal{E}_j$ that are completely cut by Γ_j for $j > 0$. Please note that the extending nature of the enrichment formulation only augments the approximation space without changing the standard term as well as previously added enrichment terms resulting in a very elegant expression.

Selecting Nodes for Enrichment. In order to determine $\mathcal{V}_j$, the set of nodes to be enriched by the jth cut, we first compute $\mathcal{E}_j$, i.e. the set of tetrahedra completely intersected by the jth cut. We call a tetrahedron partially intersected if the intersection path between cut surface, other intersecting cut surfaces and tetrahedron faces is open (cf. Figure 3, left). Correspondingly, we call a tetrahedron completely intersected by a cut if the intersection geometry forms a closed path (cf. Figure 3, right). Algorithmically, the decision whether a tetrahedron is completely cut is made using the following steps. We use the algorithm proposed by Baraff et al. [2003] in order to compute the intersection path (cf. orange path in Figure 3) as the cut surfaces are represented as triangle meshes and as a tetrahedron can be represented by four triangles. Finally, we need to determine if the cut is complete. This is realized by identifying cycles in the graph implied by the intersection path using a simple depth-first traversal.

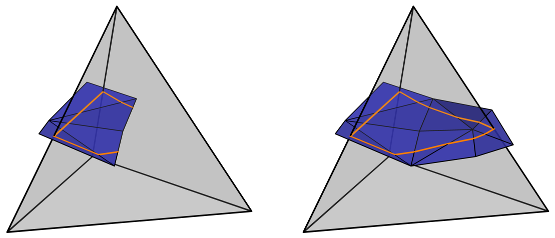


Fig. 3. Tetrahedral element intersected by cut surface represented by a triangle mesh. The orange polyline indicates the intersection path between the surface and the element.

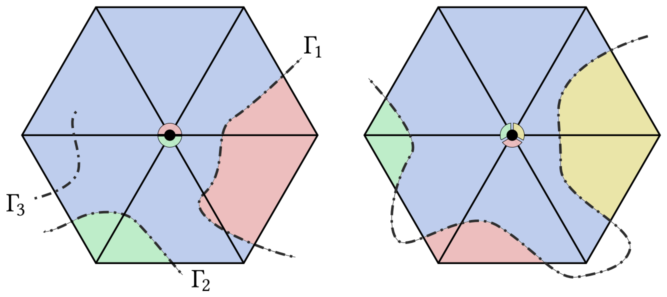


Fig. 4. Simplified 2D example for the enrichment test. Left: The central node is enriched by Γ_1 and Γ_2, as they completely separate the one-ring. The partial cut Γ_3 does not lead to an enrichment. Right: One-ring is cut by a single cut path. However, the node is enriched thrice as the one-ring is separated into four disjoint regions.

Once $\mathcal{E}_j$ is determined, we test for each vertex i associated with $\mathcal{E}_j$ whether the support domain of its according shape function N_i is either completely or partially cut. Based on the choice of Lagrange polynomials as shape functions, the support domain of a node i is exactly represented by its one-ring, i.e. the union of incident tetrahedra. We extract the boundary mesh of this one-ring represented by a triangle mesh and follow the exact same algorithm described in the previous paragraph. In case of a complete cut by surface j we enrich i using the enrichment function ψ^j_i. An example for the enrichment test is depicted in Figure 4, left using a simplified, two-dimensional example.

Special care must be taken when a node's one-ring is cut into multiple disjoint regions by a single cut surface (cf. Figure 4, right). In order to resolve this case, we treat each patch consisting of a triangle submesh that completely separates the one-ring as a distinct cut and add an enrichment for each of the patches.

Element Mass, Stiffness and Force. After determination of $\mathcal{V}_j$ all information is provided in order to construct the enriched approximation as given in Equation (10). Consequently, Equations (6) and (8) evolve into

$$\mathbf{m}_e = \int_{\Delta_e} \rho_0 \begin{bmatrix} \mathbf{N}^{0\,T}_e \\ \mathbf{N}^{1\,T}_e \\ \vdots \\ \mathbf{N}^{m\,T}_e \end{bmatrix} \left[\mathbf{N}^0_e, \mathbf{N}^1_e, \ldots, \mathbf{N}^m_e\right] d\xi \tag{11}$$

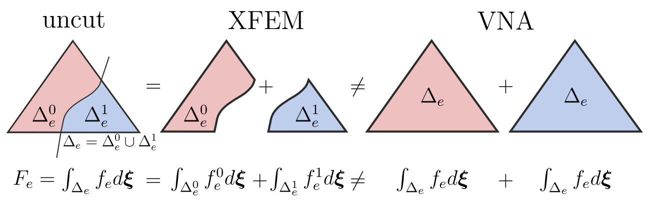

uncut
XFEM
VNA
Δ_e^0
Δ_e^1
$\Delta_e = \Delta_e^0 \cup \Delta_e^1$
Δ_e
$F_e = \int_{\Delta_e} f_e d\boldsymbol{\xi} = \int_{\Delta_e^0} f_e^0 d\boldsymbol{\xi} + \int_{\Delta_e^1} f_e^1 d\boldsymbol{\xi} \neq \int_{\Delta_e} f_e d\boldsymbol{\xi} + \int_{\Delta_e} f_e d\boldsymbol{\xi}$

Fig. 5. Integration domains of dissected element. During integration the arising subdomains left and right of the cut have to be treated separately. While XFEM based methods correctly account for this, VNA methods simply duplicate the element leading to an incorrect result.

$$\mathbf{f}_e^{\text{ext}} = \int_{\Delta_e} \begin{bmatrix} \mathbf{N}_e^{0\,T} \\ \mathbf{N}_e^{1\,T} \\ \vdots \\ \mathbf{N}_e^{m\,T} \end{bmatrix} \mathbf{b} d\boldsymbol{\xi}, \quad \mathbf{f}_e^{\text{int}}(\bar{\mathbf{u}}) = \int_{\Delta_e} \mathbf{g}_e^*(\bar{\mathbf{u}}, \boldsymbol{\xi}) d\boldsymbol{\xi} \tag{12}$$

$$\mathbf{N}_e^j = \begin{bmatrix} \psi_0^j N_0 \mathbf{I}_3 & \psi_1^j N_1 \mathbf{I}_3 & \psi_2^j N_2 \mathbf{I}_3 & \psi_3^j N_3 \mathbf{I}_3 \end{bmatrix}, \tag{13}$$

where $\mathbf{g}_e^*$ is the specific elastic element force based on the new XFEM discretization. Due to the complexity of the expression we will leave out the explicit definition of $\mathbf{g}_e^*$ and of the tangent stiffness matrix $\mathbf{K} = \partial \mathbf{f}_e^{\text{int}} / \partial \bar{\mathbf{u}}$. However, both terms can be easily determined depending on a specific hyperelastic potential using the given equations.

Due to the discontinuous nature of ψ_i^j the integrals required to compute these quantities contain discontinuous integrands. Figure 5 illustrates a single (two-dimensional) element separated by a single cut. When integral quantities have to be evaluated on the element one must pay special attention to the strong discontinuity induced by the enrichment function. This results in two major issues. The first issue is that standard quadrature rules are not applicable anymore which makes the evaluation of these quantities non-trivial. The second issue is that large volume ratios of the resulting subdomains can lead to badly conditioned mass and stiffness matrices. In the following, we will introduce a novel integration strategy and introduce a preconditioner based on the computed quadrature weights. Using this concept we address both issues in a robust manner.

At this point we would like to advise the reader with the information that similar cutting methods based on the VNA overcome both the quadrature and matrix condition issues by simply copying the element for each disjoint part that emerged while cutting (cf. [Molino et al. 2004]). While this will lead to a very stable simulation it will introduce physical inconsistencies (cf. Figure 5), i.e. mass and stiffness increments as well as a shifted center of mass, which becomes especially noticeable for cutting of fine structures. We demonstrate this in our results (cf. Figure 10).

5 NUMERICAL INTEGRATION OF DISCONTINUOUS INTEGRANDS

In order to determine element mass and element tangent stiffness matrices as well as internal and external element force vectors, integrals according to Equations (11) and (12) have to be evaluated. Based on the standard discretization using polynomial shape functions (cf. Equation (3)) all integrands are element-wise continuously differentiable polynomials. An exact evaluation of these can be conveniently achieved by using a Gauss-Legendre quadrature rule of adequate order. However, the extended discretization approach introduced in Equation (10) leads to polynomial but discontinuous integrands. Therefore, an evaluation using Gauss-Legendre quadrature is not an option as the resulting accuracy is not acceptable. The usage of adaptive techniques yields sufficient results but requires a large number of subdivisions to adequately capture the discontinuity resulting in poor performance (cf. [Fries and Belytschko 2010]). Another strategy is to remesh the integration domain in order to capture the cut surface and to apply standard Gauss-Legendre quadrature on the resulting subdomains. For one-dimensional integration domains this approach works very well in practice and is very accurate. However, for higher-dimensional domains the remeshing is computationally expensive as it involves triangulation or tetrahedralization of possibly non-convex domains and error-prone geometric intersection tests based on floating-point arithmetic.

In this section we will introduce a method to construct specialized quadrature rules for discontinuous integrands and will discuss how we compute the integrals on one- to three-dimensional domains. While a similar approach for quadrature rule construction for partially filled hexahedra was proposed by Patterson et al. (2012), they require the quadrature points to be positioned inside the filled domain, and rely on computing the integrals of the construction monomials using Monte-Carlo sampling. Also, an approach considering partially filled elements was proposed by Kim et al. [2011]. However, both approaches are not guaranteed to yield an accurate quadrature rule as the sampling of the cell may miss the partially filled regions. In contrast, we do not require the points to be inside the considered volume portion and hierarchically construct rules for integrals over the volume portion, the cut surface and mesh edges. Especially, the rule construction for the integration over the cut surface is beneficial if external tractions or boundary conditions are required on the arising surface.

In the following, $\mathcal{T}^d$ denotes the domain enclosed by a d-dimensional simplex. Further, $\overline{\mathcal{T}}^d$ is the closure of $\mathcal{T}^d$, i.e. the union of $\mathcal{T}^d$ and its boundary. Let $h : \mathcal{T}^d \to \mathbb{R}$ be a function that is only piecewise continuously differentiable on $\mathcal{T}^d$ but continuously differentiable on n_c subdomains $\mathcal{T}_i^d$, where $\overline{\mathcal{T}}^d = \bigcup_i \overline{\mathcal{T}}_i^d$ and $\mathcal{T}_i^d \cap \mathcal{T}_j^d = \emptyset \ \forall \ i \neq j$. Let further χ_i^d be the characteristic function of $\mathcal{T}_i^d$. We then additively decompose the following integral into a sum of individual integrals

$$\int_{\mathcal{T}^d} h(\mathbf{x}) d\mathbf{x} = \sum_{i=1}^{n_c} \int_{\mathcal{T}^d} \chi_i^d(\mathbf{x}) h(\mathbf{x}) d\mathbf{x}. \tag{14}$$

5.1 One-Dimensional Integration Domain

In order to numerically integrate a piecewise continuous polynomial we subdivide the integration domain such that the discontinuity is captured and integrated over each individual continuous segment:

$$\int_{\mathcal{T}^1} \chi_i(\xi)\, h(\xi) d\xi = \int_{\mathcal{T}_i^1} h(\xi) d\xi. \tag{15}$$

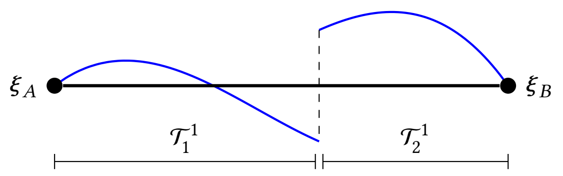


Fig. 6. Domain of 1D function is subdivided into $\mathcal{T}_1^1$ and $\mathcal{T}_2^1$ such that the function is continuous on both subdomains.

The position of each discontinuity is determined by a geometrical intersection test between the cut surface and the mesh edge. The integral of each subdomain $\mathcal{T}_i^1$ can then be numerically computed using Gauss-Legendre quadrature. Figure 6 shows a one-dimensional example of the discontinuous function and its corresponding subdomains.

5.2 Quadrature Rule Construction for Two and Three-Dimensional Domains

In order to numerically compute integrals over discontinuous integrands on two- or three-dimensional simplicial domains, we construct specialized quadrature rules depending on the discontinuties' geometries. More specifically, a single rule for each subdomain $\mathcal{T}_i^d$ where the integrand is continuously differentiable is determined, such that

$$\int_{\mathcal{T}^d} \chi_i(\xi)\, h(\xi) d\xi \approx \sum_{j=1}^{N} w_{i,j} h(\xi_j), \tag{16}$$

where ξ_j and $w_{i,j}$ denote quadrature points and weights corresponding to the ith subdomain, respectively.

Volume/Area Integrals. Given a set of integrands $\mathcal{P} = \{P_1, ..., P_M\}$ over a domain $\mathcal{T}^d$ a quadrature rule can be constructed by solving the following system of equations:

$$\begin{bmatrix} P_1(\xi_1) & \cdots & P_1(\xi_N) \\ \vdots & \ddots & \vdots \\ P_M(\xi_1) & \cdots & P_M(\xi_N) \end{bmatrix} \begin{bmatrix} w_{i,1} \\ \vdots \\ w_{i,N} \end{bmatrix} = \begin{bmatrix} \int_{\mathcal{T}_i^d} P_1 d\xi \\ \vdots \\ \int_{\mathcal{T}_i^d} P_M d\xi \end{bmatrix}. \tag{17}$$

As all occurring integrands in Equations (11) and (12) are polynomials of maximum order two in ξ, we choose $\mathcal{P}$ as a set of (linearly independent) polynomials up to order two. Moreover, choosing $\mathcal{P}$ as a basis that is orthonormal on the reference triangle/tetrahedron greatly improves the matrix condition number of Equation (17). The orthonormal basis can be easily constructed from a monomial basis, i.e. $\{1, x, y, x^2, xy, y^2\}$ for $d = 2$, using generalized Gram-Schmidt orthogonalization and subsequent normalization. The equation system is only linear in $w_{i,j}$ but nonlinear in the quadrature nodes ξ_j and for that reason hard to solve. In order to alleviate the problem, we predefine a set of quadrature nodes and keep their position constant which simplifies Equation (17) to a linear equation system.

While the construction of the matrix is trivial given the fixed quadrature points, the evaluation of the system's right-hand-side is challenging due to the integrals over the subdomain $\mathcal{T}_i^d$. In order to tackle this problem we first reformulate the right-hand-side by replacing the polynomials P_j with the divergence of their antiderivatives $\mathbf{P}_j^A$, such that $\nabla \cdot \mathbf{P}_j^A = P_j$. Please note, that there are multiple choices to choose the antiderivatives $\mathbf{P}_j^A$. However, we choose $\mathbf{P}_{j,i}^A = \int P_j d\xi_i$ for the sake of convenience, where $\mathbf{P}_{j,i}^A$ is the ith component of vector $\mathbf{P}_j^A$. Then, we can rewrite the integrals using the divergence theorem yielding

$$\begin{aligned} \int_{\mathcal{T}_i^d} \nabla_\xi \cdot \mathbf{P}_j^A d\xi &= \int_{\partial \mathcal{T}_i^d} \mathbf{P}_j^A \cdot \mathbf{n} ds \\ &= \underbrace{\int_{\partial \mathcal{T}^d} \chi_i\, \mathbf{P}_j^A \cdot \mathbf{n} ds}_{①} + \underbrace{\int_{\mathcal{I}_i} \mathbf{P}_j^A \cdot \mathbf{n} ds}_{②}, \end{aligned} \tag{18}$$

where $\mathcal{I}_i = \partial \mathcal{T}_i^d \setminus \partial \mathcal{T}^d$. Here $\mathcal{I}_i$ can be interpreted as the interface between the currently considered subdomain $\mathcal{T}_i^d$ and its neighboring subdomains.

In order to compute integral ① we distinguish between the three-dimensional ($d = 3$) and the two-dimensional ($d = 2$) simplicial case. For a tetrahedral domain we again construct quadrature rules as described in this section for each triangular face by computing the intersection of the cut surface's relevant triangles with the plane spanned by the tetrahedron's face. In the two-dimensional case, we can directly compute ① by integrating over the mesh edges as described in Section 5.1.

The second term ② represents a path/surface integral over the interface $\mathcal{I}_i$. For an evaluation of the term we construct an additional interface quadrature rule as discussed in the next paragraph.

Finally, the linear equation system (17) can be solved in order to obtain the quadrature weights $w_{i,j}$ required to construct the element vectors and matrices (see Equations (11) and (12)). It is important to state, that depending on the number of quadrature nodes and polynomials contained in $\mathcal{P}$ Equation (17) is generally over- or underdetermined. For that reason, we aim to guarantee the equation system to be underdetermined in order minimize the error in a least-squares sense and choose the number of quadrature nodes as approximately twice the number of polynomials according to symmetric quadrature rules. Finally, we have to decide where to place the quadrature points ξ_j. For both cases, i.e. $d = 2, 3$, we precompute ξ_j according to symmetric quadrature with the corresponding number of points following the method proposed by Zhang et al. [2009] and solve the underdetermined system by computing the matrix' Moore-Penrose pseudoinverse using a singular value decomposition.

In order to improve the efficiency when solving Equation (17) we us a linear transformation $Q : \mathbb{R}^d \rightarrow \mathbb{R}^d$ that maps the simplex onto a reference simplex with coordinates $(0,0,0)^T$, $(1,0,0)^T$, $(0,1,0)^T$ for a tetrahedron and $(0,0)^T$, $(1,0)^T$, $(0,1)^T$ for a triangle and additionally project the involved cut triangles/segments into the reference space. Then the matrix' pseudo-inverse has to be computed only once and can be reused for every element as the matrix is independent of the cut location. Subsequently, we scale the quadrature weights $w_{i,j}$ by $|\det(Q^{-1})|$ in order to obtain the correct weights in the original space.

Example. Given a triangular finite element with domain Δ_e that is cut by a single straight vertical line (cf. Figure 7). Using a fixed distribution of 18 quadrature points as depicted in Figure 8, left, we solve

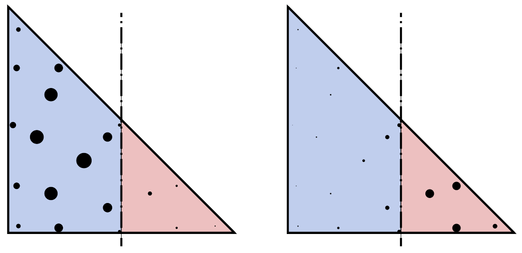


Fig. 7. Example for quadrature rules constructed on a 2D domain. A dot represents the position of a quadrature point while its radius represents its according weight $w_{i,j}$. Left: Weights to compute integrals on the blue subdomain. Right: Weights to compute integrals on the red subdomain.

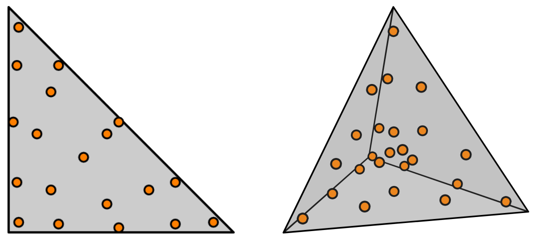


Fig. 8. Distribution example of quadrature on 2- and 3-simplex.

Equation (17) to compute the according moments $w_{1,1}, \ldots, w_{1,18}$ and $w_{2,1}, \ldots, w_{2,18}$. such that the basis polynomials of a quadratic orthonormal basis are correctly integrated over the two resulting subdomains depicted in blue and red in Figure 7 by evaluation of Equation (16). Then, the integral over a function h on Δ_e can be approximated by evaluating $\sum_j w_{1,j} h(\xi_j) + \sum_j w_{2,j} h(\xi_j)$.

Interface Integrals. As an integral over the subdomain interface $\mathcal{I}_i$ has to be computed in order to evaluate term ②, we construct a second quadrature rule with a set of predefined quadrature nodes such that

$$\int_{\mathcal{I}_i} h ds \approx \sum_{j=1}^{N_i^{\mathcal{I}}} \omega_{i,j} h(\xi_j), \tag{19}$$

where ξ_j and ω_j denote quadrature nodes and weights, respectively. Based on a vectorial, polynomial basis $\mathcal{P}^D = \left\{\mathbf{P}_1^D, \ldots, \mathbf{P}_{M_D}^D\right\}$, we can then write the according construction equations

$$\begin{bmatrix} \mathbf{P}_1^D(\xi_1)\cdot\mathbf{n}(\xi_1) & \cdots & \mathbf{P}_1^D(\xi_N)\cdot\mathbf{n}(\xi_1) \\ \vdots & \ddots & \vdots \\ \mathbf{P}_M^D(\xi_1)\cdot\mathbf{n}(\xi_1) & \cdots & \mathbf{P}_M^D(\xi_N)\cdot\mathbf{n}(\xi_N) \end{bmatrix} \begin{bmatrix} \omega_{i,1} \\ \vdots \\ \omega_{i,N} \end{bmatrix} = \begin{bmatrix} \int_{\mathcal{I}_i} \mathbf{P}_1^D \cdot \mathbf{n} ds \\ \vdots \\ \int_{\mathcal{I}_i} \mathbf{P}_M^D \cdot \mathbf{n} ds \end{bmatrix}. \tag{20}$$

Analogously to Equation (17) the evaluation of the linear system's right-hand-side is non-trivial for a choice of arbitrary polynomial vectors. Fortunately, a special choice of $\mathcal{P}^D$ as divergence-free basis allows us to elegantly simplify the right-hand-side. We construct the divergence-free basis following the method presented by Müller et al. [2013]. Assuming that $\nabla_\xi \cdot \mathbf{P}_j^D = 0$ we can rewrite the entries of Equation (20)'s right-hand-side using the divergence theorem as

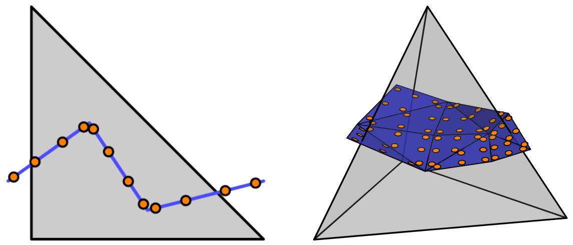


Fig. 9. Interface quadrature nodes. Blue line/surface represents the cut geometry while orange dots indicate the location of quadrature points. Left: 2D case; triangle intersected by polyline. Right: 3D case; tetrahedron intersected by triangle mesh.

follows:

$$\int_{\mathcal{I}_i} \mathbf{P}_j^D \cdot \mathbf{n} ds = \cancelto{0}{\int_{\partial\mathcal{T}^d} \nabla_\xi \cdot \mathbf{P}_j^D d\xi} - \int_{\partial\mathcal{T}^d} \chi_i \mathbf{P}_j^D \cdot \mathbf{n} ds. \tag{21}$$

The rewritten integral can then either be directly evaluated by piecewise integration over the mesh edges as described in Section 5.1 for $d = 2$ or using the already constructed quadrature rule used to evaluate ① for $d = 3$.

Again, we have to choose quadrature points ξ_j in order to evaluate the integrands in meaningful positions. For $d = 2$ the cut path is a polyline. Therefore, we place four quadrature points on each segment following the corresponding quadrature points of traditional Gauss-Legendre quadrature as illustrated in Figure 9, left. Due to the dependence of the system matrix on the cut geometry the matrix is not guaranteed to have full rank. We aim to still maintain an underdetermined system by using enough quadrature points. Therefore, using approximately twice as many points as polynomials basis vectors in $\mathcal{P}^D$ was a safe choice for all of our tests and simulations. Therefore, we subdivide one segment at a time until we have acquired the desired number of quadrature points. Similarly, we place the quadrature points for $d = 3$ on the triangles of the intersecting patch, again, using the abscissae of the symmetric quadrature rules for triangles determined following the approach of Zhang et al. [2009] (cf. Figure 9, right).

Finally, we can solve Equation (20) in order to obtain the interface quadrature weights ω_i. Please note, that we again choose the number of quadrature points to be approximately twice the number of polynomials in $\mathcal{P}^D$ which results in an undetermined system. Moreover, the system matrix is in general rank-deficient and has poor conditioning depending on the discontinuity's geometry. To robustly solve the underdetermined system, we compute the Moore-Penrose pseudoinverse using a singular value decomposition.

Discussion. Placing the quadrature nodes directly onto the cut path/surface rather than somewhere within the surrounding finite element (cf. [Müller et al. 2013]) ensures that the cut patch's geometry, if smooth or sharp, is captured sufficiently resulting in an accurate quadrature rule. It might, however, seem to be an interesting choice to place the quadrature nodes on the polyline segments/triangles without clipping the primitives on the considered triangular/tetrahedral element. Conducting several experiments showed that even if many quadrature points lay outside the element due to the cut path segments being large compared to the element size, still resulted in very accurate quadrature rules. We assume that this is the case because the integrands, i.e. polynomials, are infinitely often differentiable and therefore very smooth.

6 PRECONDITIONING AND AVOIDANCE OF ILL-CONDITIONING

The enrichment of nodes using shifted sign enrichment as described in Section 4.3 adds DOFs to the system and therefore leads to additional entries in the system matrix. A DOF's entries heavily depend on how the corresponding node's support domain, i.e. its one-ring, is dissected. If the ratio of the contributing subdomain's volume (the portion of cut-off material) and the one-ring's volume is high the system matrix' condition number deteriorates. This phenomenon is similar to the influence of element shapes on the matrix condition number in standard FE discretizations [Shewchuk 2002].

The convergence of iterative solvers is heavily influenced by the condition number of the system matrices (see e.g. [Shewchuk 1994]) but also modern direct solvers rely on well-conditioned matrices (see e.g. [Sonneveld 1989]). Even worse, a numerically singular system can cause a breakdown of the simulation. For the stated reasons keeping the system regular and moreover well-conditioned is a vital requirement and one of the key aspects regarding stability and robustness. We ensure a well-conditioned system by means of the following three steps:

1. Perturbation Step We check if the cut surface touches (but does not intersect) the support domain of the currently processed node's shape functions. As previously described, a vertex is enriched if its one-ring is completely cut. Due to the nature of floating point arithmetic, a complete cut may be detected but no actual intersection geometry is generated. This could cause the construction of the interface quadrature rule to fail, and result in a potentially singular system matrix. Therefore, we perturb the cut surface's vertices in order to improve robustness. As this problem is caused by numerical inaccuracies in geometric intersection tests, intersection algorithms based on robust predicates or exact arithmetic could be employed to resolve this problem.

2. Constraining Step If the support domain's volume of an enriched node is small compared to the volume of its shape functions' support domains, we constrain the DOF. In order to avoid the material "sticking" together in the region close to the constrained DOF we move the DOF with its according fragment to keep the afflicted region as-rigid-as-possible. Mathematically, the criterion for constraining a DOF $d_{i,j}$ reads

$$\frac{V_{i,j,\mathrm{enr}}}{V_{i,\mathrm{supp}}} < \epsilon, \tag{22}$$

where $V_{i,\mathrm{supp}}$ is the volume of the support domain of the ith node and $V_{i,j,\mathrm{enr}}$ the volume of the DOF $d_{i,j}$'s support domain volume associated with the ith node and the jth enrichment. Further, ϵ represents a scalar threshold that we chose 10^{-9} for all of our results. Generally, the computation of $V_{i,j,\mathrm{enr}}$ is problematic since we want to avoid to explicitly represent the DOF's support domain. However, we can fortunately reuse the previously computed weights to determine $V_{i,j,\mathrm{enr}}$ as the volume over the given domain is equal to $\sum_{e\in C_i}\int_{\Delta_e^j} 1 d\xi = \sum_{e\in C_i}\sum_{j=1}^{N} w_{e,j}$, where C_i is the set of elements incident to vertex i.

3. Preconditioning Step In the third and final step we construct a diagonal preconditioning matrix $\mathbf{T}$ to improve the system matrix $\mathbf{A}$'s condition number for the subsequent Newton iterations. In order to keep the matrix' symmetry we choose to bilaterally apply the diagonal preconditioning matrix $\mathbf{T}$ following

$$\mathbf{Ax} = \mathbf{c} \tag{23}$$

$$\mathbf{T}^T\mathbf{ATy} = \mathbf{T}^T\mathbf{c} \tag{24}$$

$$\mathbf{x} = \mathbf{Ty}. \tag{25}$$

As previously mentioned the area ratio of $V_{i,\mathrm{supp}}$ and $V_{i,j,\mathrm{enr}}$ has a big influence on the system matrix' condition number. Therefore, we aim to scale the diagonal entries of $\mathbf{A}$ with the inverse ratio:

$$T_{d_{i,j},d_{i,j}} = \frac{1}{\sqrt{v_{i,j}}} \tag{26}$$

$$v_{i,j} = \frac{V_{i,j,\mathrm{enr}}}{V_{i,\mathrm{supp}}} = \frac{\sum_{e\in C_i}\sum_{j=1}^{N} w_{e,j}}{V_{i,\mathrm{supp}}}. \tag{27}$$

7 RESULTS AND DISCUSSION

In this section we explain how we generate a representation for visualization purposes and discuss our results and comparisons. All measurements provided in this section were performed on an Intel i7-6700HQ processor with 2.6 GHz, 4 cores and 16GB RAM. We use an implicit Euler scheme for time integration and solve the resulting nonlinear equation systems using Newton's method in combination with the PARDISO solver implemented in Intel's Math Kernel Library. The number of quadrature points for volume, area quadrature were 24 and 19, respectively (cf. Figure 8). Analogously, we used 4 and 6 quadrature points per involved cut path segment and cut surface triangle, respectively (cf. Figure 9). Please note that we apply Dirichlet conditions following Wu et al. [2008] to fixate parts of the discretizations in all presented scenarios. Further, none of the presented simulations include collision handling.

Visualization. The simulation yields a numerical solution represented by the displacement field $\mathbf{u}_{\mathrm{XFEM}}^h(\xi, t)$ to the initial/boundary value problem described in Equation (1). Consequently, a suitable representation has to be found in order to visualize the result in an appealing way. At this point we would like to point out that the generation of a visualization can be understood as post-processing step and that the strategy presented in the following is by no means inextricably linked to the described simulation.

In order to represent the (initially uncut) simulation object's boundary surface we first extract the border mesh from the tetrahedral discretization in reference space. Subsequently, we compute the geometric intersection path between border mesh and cut surfaces using the algorithm proposed by Baraff et al. [2003]. The acquired path is then explicitly embedded into both the border mesh and the cut surface meshes by inserting intersection vertices and the according mesh edges followed by a retriangulation. As the cut surfaces overlap the border mesh we clip all triangles lying outside the border mesh. Each cut surface mesh is then duplicated in order to

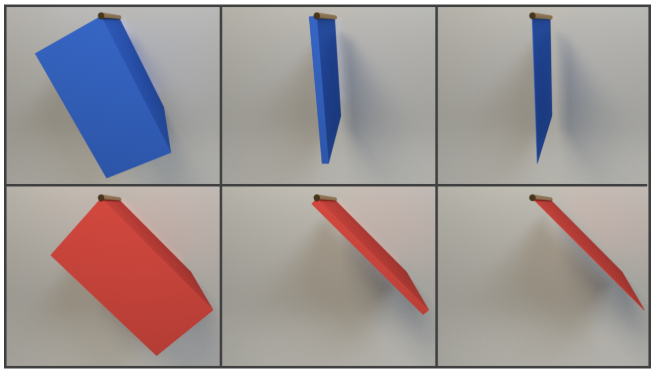


Fig. 10. Cubes consisting of five tetrahedra fixated on one edge are cut in differently sized slices. This image shows the resting position of the slices where all blue-shaded slices where simulated using our method and the red-shaded slices using the VNA.

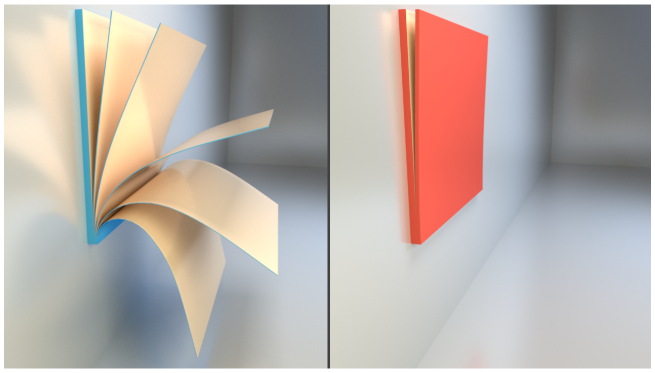


Fig. 11. A deformable slab is fixed to a wall and cut into five slices with increasing thickness. Left: Resting position resulting from our XFEM based simulation. Right: Result computed using the VNA with the exact same material and simulation parameters.

represents both sides of the cut. Finally, we use Equation (10) to compute the world positions for each vertex in the visualization mesh. If the tetrahedral mesh's boundary is too coarse for visualization purposes or if a higher resolution is desired an additional triangle mesh representing the object boundary may be used. The only requirement is that the mesh is entirely contained in the tetrahedral mesh in order to evaluate Equation (10) for mapping the vertices to world space positions. We used a high-resolution representation to generate the results shown in Figures 1 and 13 (right).

Our method vs. VNA. We performed three experiments where we compared our XFEM based method with the widely used VNA. In the first scenario depicted in Figure 10 we cut a cube consisting of five tetrahedra at three different locations. Because of the hinge fixation the resulting slices fold down and oscillate until they reach a resting position. In all simulations performed with our method (blue shading) the slices rest in a position where their centers of mass are exactly placed under the fixations. In contrast, the resting positions of the VNA based simulations (red shading) are noticeably displaced resulting in a physically incorrect and even implausible state. In the second experiment depicted in Figure 11 we cut a deformable slab fixed to a wall into several slices with increasing thickness. The slab is discretized with $61 \times 11 \times 1$ blocks consisting of five tetrahedra. Our method produces a realistic result where each distinct slice folds down separately due to the varying thicknesses. When the exact same scenario is simulated using the VNA each slice is as stiff as the uncut slab caused by inexact integration (cf. Figure 5) and therefore scarcely folds. In the third and final experiment we compared how well mass is conserved after inserting several cuts into an object. As illustrated in Figure 12 we simulated two blocks hanging on a comparably small strip of material. Due to the weight of the blocks the attached strips stretch and can thus be interpreted as nonlinear springs. The nonlinearity in the strips deformation behavior is caused by the nonlinear hyperelastic constitutive model as explained in Section 3. As a consequence of the insertion of several cuts into the blocks the objects unfold. Investigating the unfolded objects' resting states reveals that the simulation performed with our method conserves mass very well as the attachment connecting

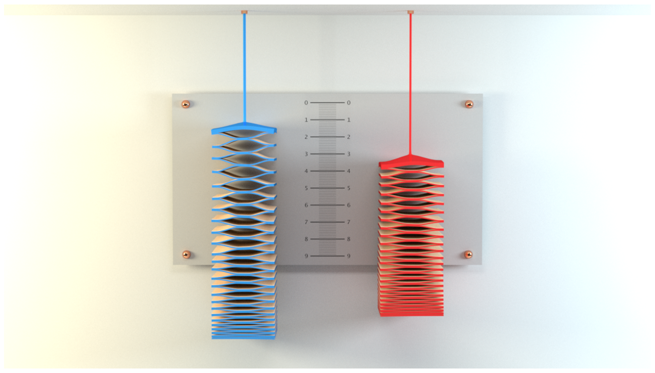


Fig. 12. Two blocks hanging on thin material strip are cut into an unfolding structure. The result produced by our method (left) conserves mass before and after the cut as indicated by the strip's extension. The result simulated using the VNA (right) drops significantly due to unphysically added mass and looks stiffer.

block and material strip still rests at the initial position (around one on the background measuring scale). In contrast, the block simulated using the VNA drops significantly, finally resting at three on the background measuring scale. This indicates a significant amount of additional mass gained caused by the element duplication strategy. Moreover, the resulting dynamic behavior of the object simulated with our method is noticeably "livelier" and looks less stiff compared to the VNA simulation.

Quadrature Comparison. We tested the quality of the acquired volume quadrature rules for 24 quadrature points by evaluating Equation (16) on the unit tetrahedron for several test polynomials. Three different cut surface shapes were used: a planar, a kinked and a spherical one. Furthermore, the results were compared against a regular sampling approach as used by Kaufmann et al. [2009] and a specialized adaptive quadrature approach for multidimensional discontinuous functions proposed by Müller et al. [2012]. We used a regular grid on the tetrahedron's bounding box consisting of 50^3 cells for the regular sampling. For the adaptive approach a maximum

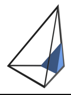
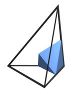
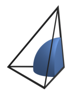
Scenario	Integrand $h(\xi, \eta, \zeta)$	Our Meth. ϵ_{rel} (24QP)	Reg. Samp. ϵ_{rel}	Reg. Samp. #QP	Adapt. ϵ_{rel}	Adapt. #QP
	1	3.32e−5	4.44e−3	560	5.79e−2	54
	$\xi + \eta + \zeta$	4.43e−5	4.44e−3	560	8.19e−2	54
	$\eta\zeta + \xi\zeta + \xi\eta$	5.54e−5	9.78e−1	560	9.25e−1	181
	$\xi^2 + \eta^2 + \zeta^2$	5.54e−5	5.00e−1	560	4.94e−1	181
	1	2.99e−6	4.94e−4	3795	1.54e−2	646
	$\xi + \eta + \zeta$	6.50e−6	4.89e−4	3795	2.16e−2	646
	$\eta\zeta + \xi\zeta + \xi\eta$	1.26e−5	2.50e−3	3795	1.86e−2	2512
	$\xi^2 + \eta^2 + \zeta^2$	1.38e−5	1.10e−4	3795	1.72e−2	2512
	1	3.88e−4	4.62e−3	8219	1.21e−2	788
	$\xi + \eta + \zeta$	5.15e−4	6.23e−3	8219	1.65e−2	788
	$\eta\zeta + \xi\zeta + \xi\eta$	6.43e−4	7.60e−3	8219	1.30e−3	3073
	$\xi^2 + \eta^2 + \zeta^2$	6.46e−4	7.71e−3	8219	1.79e−4	3073

Table 1. Results of our numerical integration test using regular sampling, an adaptive approach [Müller et al. 2012] and our method. The unit tetrahedron was cut using a planar (diagonal) cut, a kinked cut and a spherical cut. The integral over polynomials $h(\xi, \eta, \zeta)$ over the volume portion indicated in blue was computed. ϵ_{rel} and #QP represent the relative error to the analytic solution and the number of the contributing quadrature points used for quadrature, respectively.

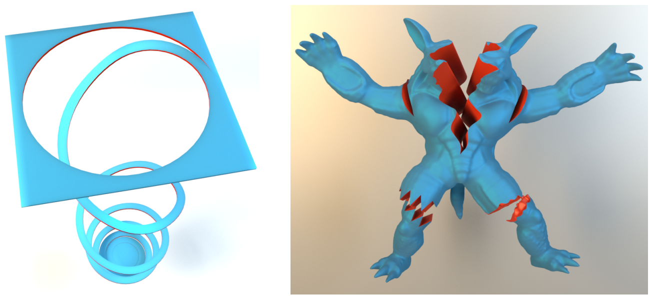


Fig. 13. Left: Cutting a plate with a smooth, helical surface. Right: Stanford armadillo is cut by several high- and low-frequent, smooth or sharp surfaces.

refinement depth of four was used. The results are summarized in Table 1. Please note the we counted only quadrature points evaluating to non-zero values in the other methods. The regular sampling consistently produced less accurate results compared to our method while a much larger number of quadrature points was required. Although the adaptive approach was in some cases able to achieve comparable accuracy, the number of required quadrature points was larger by several orders of magnitude.

Complex Cutting. Besides comparisons and academic examples we performed four simulations with multiple complex cut surfaces. In the first experiment we used a helical cut surface to dissect a plate discretized with $25 \times 25 \times 1$ hexahedral blocks each consisting of 5 tetrahedra (cf. Figure 13, left). Even though the object is discretized by a small number of linear elements the helical-shaped material deforms very smoothly while the cut is progressing. In the second scenario we simulated the Stanford armadillo that we fixated on its limbs (cf. Figure 13, right). Here, we demonstrate that our method is able to robustly handle cuts with low and high frequent smooth and/or sharp features while dissecting thousands of tetrahedra.

In the scenario illustrated in Figure 1 (right), we cut a circular groove into the Stanford bunny and subsequently peeled the groove off with a second cut. The result demonstrates that our method is able to robustly handle finely structured cut surfaces while producing a realistic result. Moreover, it shows that our simulation yields realistic results even when a coarse tetrahedral mesh is used. The tetrahedral discretization of the bunny consisted of only ~18.5k tetrahedra while the visualized triangle mesh had ~53k triangles. In the fourth and final scenario we simulated a plate with several attached objects as depicted in Figure 1 (left). We modeled a very long cut surface and complicated the scenario by displacing the cut surface' vertices using distorted noise. Due to the noise several tetrahedra are cut multiple times by the same cut surface resulting in the requirement to enrich several nodes multiple times (problem described in Section 4.3). In this highly complex example, we show that our method is able to accurately handle very complex cut surfaces and yields a realistic animation where all cut regions are properly separating without any artifacts.

We also measured the performance of the last two scenarios. Both scenarios were simulated using a time step width of $\Delta t = 5\text{ms}$. The simulation of the bunny scenario took on average 1.375s per step with initially 5326 DOFs, finally resulting in 9432 DOFs. Further, 47.27% of the time was spent to process the cuts and to enrich the nodes. Our quadrature rule construction is included in that portion but individually took only 8.5% of the time required per step. The plate scenario initially consisted of 47643 DOFs resulting in 53326 DOFs and took 9.365s per time step. 17.5% of the time was spent to process the cuts while our quadrature rule construction individually took 0.7% of the simulation time. The last scenario clearly shows the advantage of our approach in comparison to remeshing based methods since even for a very complex cut surface the number of DOFs increased only by 12%.

Preconditioning. In order to analyze the effect of our preconditioner on the system matrix' condition number we conducted an experiment where we cut the unit cube consisting of five tetrahedra with bounding coordinates $(-1, -1, -1)^T$ and $(1, 1, 1)^T$ using a plane with normal $(1, 0, 0)$. We then measured the condition number of the system matrix while varying the location of the cut in x-direction. We further chose the system matrix as $\mathbf{A} = \mathbf{M} + \Delta t^2 \mathbf{K}$ resulting from discretization and linearization in a single Newton step, where Δt and $\mathbf{K} = \partial \mathbf{f}^{\text{int}} / \partial \mathbf{u}$ represent time step width and tangent stiffness matrix, respectively. The cube was unconstrained and the matrix was assembled for Young's modulus $E = 10^6 N/m^2$, Poisson ratio $\nu = 0.3$, density $\rho = 1000 kg/m^3$ and time step width $\Delta t = 10^{-3} s$. The graph in Figure 14 shows the outcome of our experiment. While the condition number of the unpreconditioned system matrix approaches infinity when the cut is close to the bounds, the condition number of the preconditioned matrix is considerably lower. Even when the cube is cut exactly through its center the condition number of the unpreconditioned matrix is nearly two orders of magnitude higher compared to the uncut matrix' condition number. It should further be mentioned that the condition number of the preconditioned matrix will also approach infinity when the cut is so close to the cube boundary that some of the enriched vertices' DOFs have (almost) no support. Fortunately, this is prevented by constraining the

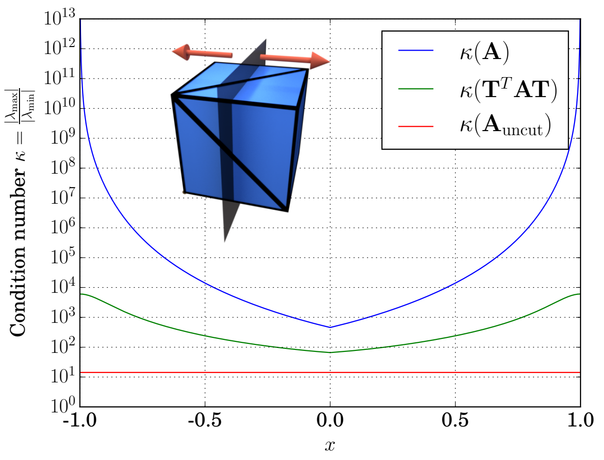


Fig. 14. Semi-logarithmic plot of the condition number of the system matrix $\mathbf{A} = \mathbf{M} + \Delta t^2\mathbf{K}$ over cut location x. Cube consisting of five tetrahedra was cut by a plane.

affected nodes as explained in Section 6 which keeps the condition number bounded.

8 CONCLUSION

We presented a novel approach for the simulation of complex cutting of three-dimensional deformable solids with fully implicit time integration. After introducing the concept of basis enrichment for the representation of cuts within the finite element discretization, an approach to construct specialized quadrature rules to compute arising integrals over discontinuous elements on polyhedral domains was proposed. Moreover, an algorithm was presented that keeps the equation system required for implicit time integration regular and well-conditioned. We demonstrated that our method robustly handles complex cut surfaces in large scenarios. Further, we showed that our method is able to realistically simulate finely structured cuts, even in the case of coarse tetrahedral discretizations. We compared the proposed method to the popular VNA where we could clearly show how the proposed method offers considerable advantages concerning preservation of physical plausibility such as mass conservation and correctly maintaining stiffness properties.

Limitations and Future Work. As explained before, our method only treats elements as cut if they are completely dissected by the cut surface. Therefore, we cannot simulate cuts progressively advancing within a single element. Building on the flexibility of the XFEM to incorporate different enrichments, we plan to enrich completely dissected elements using the presented shifted sign enrichment and treat regions near the crack-tip with the harmonic enrichment strategy following Kaufmann et al. [2009]. A localized Laplace enrichment near crack-tips would allow us to represent intra-element progressive cuts at moderate cost, and would result in a nice synergy of both approaches. A limitation of our implementation is that we did not treat the case of mutually intersecting and T-cuts. However, we see no reason why this should limit the generality of the proposed approach and we are confident that by using the specialized sign enrichments for branched cracks proposed by Daux et al. [2000] these cases can be treated in a straightforward manner. Further, we used the implicit backward Euler scheme together with Newton iterations to solve the resulting nonlinear equation system. We are aware of the fact that more elaborate and efficient implicit time integration schemes were developed within the computer graphics community. While performance was not the main focus of our work, we would still like to incorporate a more efficient time integrator and would like to investigate if our method can be further parallelized.

ACKNOWLEDGMENTS

This work is supported by the *ERC Starting Grant* 637014 and by the *German Research Foundation (DFG)* under contract numbers BE 5132/4-1 and TH 2034/1-1.

REFERENCES

P. Areias and T. Belytschko. 2006. A comment on the article "A finite element method for simulation of strong and weak discontinuities in solid mechanics" by A. Hansbo and P. Hansbo [Comput. Methods Appl. Mech. Engrg. 193 (2004) 3523–3540]. *Computer Methods in Applied Mechanics and Engineering* 195, 9-12 (2006), 1275–1276.

Z. Bao, J. Hong, J. Teran, and R. Fedkiw. 2007. Fracturing Rigid Materials. *IEEE Transactions on Visualization and Computer Graphics* 13, 2 (2007), 370–378.

D. Baraff, A. Witkin, and M. Kass. 2003. Untangling Cloth. *ACM Transactions on Graphics* 22, 3 (2003), 862–870.

J. Bedrossian, J. von Brecht, S. Zhu, E. Sifakis, and J. Teran. 2010. A second order virtual node method for elliptic problems with interfaces and irregular domains. *J. Comput. Phys.* 229, 18 (2010), 6405–6426.

T. Belytschko and T. Black. 1999. Elastic crack growth in finite elements with minimal remeshing. *Internat. J. Numer. Methods Engrg.* 45, 5 (1999), 601–620.

D. Bielser, P. Glardon, M. Teschner, and M. Gross. 2004. A State Machine for Real-Time Cutting of Tetrahedral Meshes. *Graphical Models* 66, 6 (2004), 398–417.

D. Bielser and M. H. Gross. 2000. Interactive Simulation of Surgical Cuts. In *Pacific Graphics*. 116–126.

D. Bielser, V. Maiwald, and M. Gross. 1999. Interactive Cuts through 3-Dimensional Soft Tissue. *Computer Graphics Forum* 18, 3 (1999), 31–38.

O. Busaryev, T. Dey, and H. Wang. 2013. Adaptive Fracture Simulation of Multi-Layered Thin Plates. *ACM Transactions on Graphics* 32, 4 (2013), 1.

C. Daux, N. Moes, J. Dolbow, N. Sukumar, and T. Belytschko. 2000. Arbitrary branched and intersecting cracks with the extended finite element method. *Internat. J. Numer. Methods Engrg.* 48, 12 (2000), 1741–1760.

C. Dick, J. Georgii, and R. Westermann. 2010. A Hexahedral Multigrid Approach for Simulating Cuts in Deformable Objects. *IEEE Transactions on Visualization and Computer Graphics* 17, 11 (2010), 1663–1675.

T.P. Fries and T. Belytschko. 2010. The extended/generalized finite element method: An overview of the method and its applications. *Internat. J. Numer. Methods Engrg.* 84, 3 (2010), 253–304.

A. Hansbo and P. Hansbo. 2004. A finite element method for the simulation of strong and weak discontinuities in solid mechanics. *Computer Methods in Applied Mechanics and Engineering* 193 (2004), 3523–3540.

J. Hegemann, C. Jiang, C. Schroeder, and J. Teran. 2013. A Level-Set Method for Ductile Fracture. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 193–201.

J. Hellrung, L. Wang, E. Sifakis, and J. Teran. 2012. A second order virtual node method for elliptic problems with interfaces and irregular domains in three dimensions. *J. Comput. Phys.* 231, 4 (2012), 2015–2048.

D. Holdych, D. Noble, and R. Secor. 2008. Quadrature rules for triangular and tetrahedral elements with generalized functions. *Internat. J. Numer. Methods Engrg.* 73, 9 (2008), 1310–1327.

L. Jeřábková and T. Kuhlen. 2009. Stable Cutting of Deformable Objects in Virtual Environments Using XFEM. *IEEE Computer Graphics and Applications* 29, 2 (2009), 61–71.

P. Kaufmann, S. Martin, M. Botsch, E. Grinspun, and M. Gross. 2009. Enrichment Textures for Detailed Cutting of Shells. *ACM Transactions on Graphics* 28, 3 (2009), 50:1–50:10.

P. Kaufmann, S. Martin, M. Botsch, and M. Gross. 2008. Flexible Simulation of Deformable Models Using Discontinuous Galerkin FEM. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 105–115.

J. Kim and N. Pollard. 2011. Fast simulation of skeleton-driven deformable body characters. *ACM Transactions on Graphics* 30, 5 (2011), 1–19.

D. Koschier, S. Lipponer, and J. Bender. 2014. Adaptive Tetrahedral Meshes for Brittle Fracture Simulation. In *ACM SIGGRAPH/Eurographics Symposium on Computer*

Animation. 1–10.

S. Martin, P. Kaufmann, M. Botsch, M. Wicke, and M. Gross. 2008. Polyhedral Finite Elements Using Harmonic Basis Functions. *Computer Graphics Forum* 27, 5 (2008), 1521–1529.

N. Mitchell, C. Cutting, and E. Sifakis. 2015. GRIDiron: An Interactive Authoring and Cognitive Training Foundation for Reconstructive Plastic Surgery Procedures. *ACM Transactions on Graphics* 34, 4 (2015), 43:1–43:12.

N. Moës, J. Dolbow, and T. Belytschko. 1999. A Finite Element Method for Crack Growth without Remeshing. *Internat. J. Numer. Methods Engrg.* 46, 1 (1999), 131–150.

N. Molino, Z. Bao, and R. Fedkiw. 2004. A Virtual Node Algorithm for Changing Mesh Topology During Simulation. *ACM Transactions on Graphics* 23, 3 (2004), 385–392.

S. Mousavi, E. Grinspun, and N. Sukumar. 2011a. Harmonic enrichment functions: A unified treatment of multiple, intersecting and branched cracks in the extended finite element method. *Internat. J. Numer. Methods Engrg.* 85, 10 (2011), 1306–1322.

S. Mousavi, E. Grinspun, and N. Sukumar. 2011b. Higher-order extended finite elements with harmonic enrichment functions for complex crack problems. *Internat. J. Numer. Methods Engrg.* 86, 4-5 (2011), 560–574.

B. Müller, S. Krämer-Eis, F. Kummer, and M. Oberlack. 2017. A high-order discontinuous Galerkin method for compressible flows with immersed boundaries. *Internat. J. Numer. Methods Engrg.* 110, 1 (2017), 3–30.

B. Müller, F. Kummer, and M. Oberlack. 2013. Highly accurate surface and volume integration on implicit domains by means of moment-fitting. *Internat. J. Numer. Methods Engrg.* 96, 8 (2013), 512–528.

B. Müller, F. Kummer, M. Oberlack, and Y. Wang. 2012. Simple multidimensional integration of discontinuous functions with application to level set methods. *Internat. J. Numer. Methods Engrg.* 92, 7 (2012), 637–651.

A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson. 2006. Physically Based Deformable Models in Computer Graphics. *Computer Graphics Forum* 25, 4 (2006), 809–836.

J. O'Brien, A. Bargteil, and J. Hodgins. 2002. Graphical Modeling and Animation of Ductile Fracture. *ACM Transactions on Graphics* 21, 3 (2002), 291–294.

J. O'Brien and J. Hodgins. 1999. Graphical Modeling and Animation of Brittle Fracture. In *Conference on Computer Graphics and Interactive Techniques*. 137–146.

M. Pauly, R. Keiser, B. Adams, P. Dutré, M. Gross, and L. Guibas. 2005. Meshless Animation of Fracturing Solids. *ACM Transactions on Graphics* 24, 3 (2005), 957–964.

T. Pfaff, R. Narain, J. de Joya, and J. O'Brien. 2014. Adaptive Tearing and Cracking of Thin Sheets. *ACM Transactions on Graphics* 33 (2014), 1–9.

C. Richardson, J. Hegemann, E. Sifakis, J. Hellrung, and J. Teran. 2011. An XFEM method for modeling geometrically elaborate crack propagation in brittle materials. *Internat. J. Numer. Methods Engrg.* 88, 10 (2011), 1042–1065.

C. Schroeder, A. Stomakhin, R. Howes, and J. Teran. 2014. A second order virtual node algorithm for Navier-Stokes flow problems with interfacial forces and discontinuous material properties. *J. Comput. Phys.* 265 (2014), 221–245.

J. Shewchuk. 1994. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain.* Technical Report.

J. Shewchuk. 2002. *What Is a Good Linear Finite Element? Interpolation, Conditioning, Anisotropy, and Quality Measures.* Technical Report.

E. Sifakis and J. Barbic. 2012. FEM Simulation of 3D Deformable Solids. In *ACM SIGGRAPH Courses*. 1–50.

E. Sifakis, K. Der, and R. Fedkiw. 2007. Arbitrary Cutting of Deformable Tetrahedralized Objects. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 73–80.

F. Sin, D. Schroeder, and J. Barbič. 2013. Vega: Non-linear FEM deformable object simulator. *Computer Graphics Forum* 32, 1 (2013), 36–48.

P. Sonneveld. 1989. CGS, a fast Lanczos-type solver for nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.* 10, 1 (1989), 36–52.

D. Steinemann, M. Harders, M. Gross, and G. Szekely. 2006. Hybrid Cutting of Deformable Solids. In *IEEE Virtual Reality*. 35–42.

G. Ventura. 2006. On the elimination of quadrature subcells for discontinuous functions in the eXtended Finite-Element Method. *Internat. J. Numer. Methods Engrg.* 66, 5 (2006), 761–795.

Y. Wang, C. Jiang, C. Schroeder, and J. Teran. 2014. An Adaptive Virtual Node Algorithm with Robust Mesh Cutting. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 1–9.

M. Wicke, D. Ritchie, B. Klingner, S. Burke, J. Shewchuk, and J. O'Brien. 2010. Dynamic local remeshing for elastoplastic simulation. In *ACM Transactions on Graphics*.

B. Wu, Z. Xu, and Z. Li. 2008. A note on imposing displacement boundary conditions in finite element analysis. *Communications in Numerical Methods in Engineering* 24, 9 (2008), 777–784.

J. Wu, C. Dick, and R. Westermann. 2011. Interactive High-Resolution Boundary Surfaces for Deformable Bodies with Changing Topology. In *Virtual Reality Interactions and Physical Simulations*. 29–38.

J. Wu, R. Westermann, and C. Dick. 2015. A Survey of Physically Based Simulation of Cuts in Deformable Bodies. *Computer Graphics Forum* 34, 6 (2015), 1–27.

Y.-H. Yeung, J. Crouch, and A. Pothen. 2016. Interactively Cutting and Constraining Vertices in Meshes Using Augmented Matrices. *ACM Transactions on Graphics* 35, 2 (2016), 18:1–18:17.

L. Zhang, T. Cui, and H. Liu. 2009. A Set of Symmetric Quadrature Rules on Triangles and Tetrahedra. *Journal of Computational Mathematics* 27, 1 (2009), 89–96.

Y. Zhu, Y. Wang, J. Hellrung, A. Cantarero, E. Sifakis, and J. Teran. 2012. A second-order virtual node algorithm for nearly incompressible linear elasticity in irregular domains. *J. Comput. Phys.* 231, 21 (2012), 7092–7117.