Enrichment Textures for Detailed Cutting of Shells

Peter Kaufmann, ETH Zurich
Sebastian Martin, ETH Zurich
Mario Botsch, Bielefeld University
Eitan Grinspun, Columbia University
Markus Gross, ETH Zurich

Abstract

We present a method for simulating highly detailed cutting and fracturing of thin shells using low-resolution simulation meshes. Instead of refining or remeshing the underlying simulation domain to resolve complex cut paths, we adapt the extended finite element method (XFEM) and enrich our approximation by custom-designed basis functions, while keeping the simulation mesh unchanged. The enrichment functions are stored in *enrichment textures*, which allows for fracture and cutting discontinuities at a resolution much finer than the underlying mesh, similar to image textures for increased visual resolution. Furthermore, we propose *harmonic enrichment functions* to handle multiple, intersecting, arbitrarily shaped, progressive cuts per element in a simple and unified framework. Our underlying shell simulation is based on discontinuous Galerkin (DG) FEM, which relaxes the restrictive requirement of C^1 continuous basis functions and thus allows for simpler, C^0 continuous XFEM enrichment functions.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling; I.6.8 [Simulation and Modeling]: Types of Simulation—Animation

1 Introduction

One of the most striking aspects of thin-walled structures is their dramatic failure via buckling, tearing, and fracturing modes. The introduced discontinuities can drastically alter the stability, deformation, and trajectory of the structure. Simulating such phenomena requires identifying the location of discontinuities and predicting the response of neighboring material. Yet even representing discontinuities is in itself considered awkward. Works in the graphics and mechanics communities have considered adaptive refinement methods, which are well-suited for gradual variations in the scale of relevant features, but require too many levels of refinement to sharply resolve creases or fractures. Other works take a remeshing approach, which avoids introducing many new unknowns at the cost of some additional implementation and a projection step to transfer data between the old and new representations.

In an alternative point of departure, the extended finite element method (XFEM) enriches the representation with the specific basis functions required to capture the desired discontinuity. In doing so, XFEM introduces (unlike refinement) only a negligible number of new unknowns, and (unlike remeshing) keeps intact the original mesh connectivity. No method is best for all applications, but the XFEM approach, as yet unexplored in the context of animation tools for flexible surfaces, offers an enticing way to represent highly-complex, sharp features, while keeping computation and implementation relatively simple.

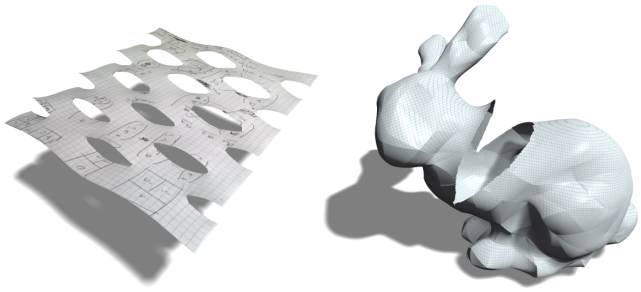

Figure 1: *Starting from a single 8-node quad shell element, our method is able to capture complex deformations and topological changes by simply enriching the element's basis functions through texture maps (left). Fracturing a bunny triangle mesh (right).*

The mechanics community has studied extensively the application of XFEM to thin shell fracture. However, mechanics research focuses on accurate prediction of crack growth and consequent material displacements, necessitating a fine spatial discretization and in particular a sparse number of discontinuous features relative to mesh elements. In the rare case where an element is cut a second or third time, current methods impose a hierarchical structure to keep track of the cuts. Furthermore, because the diameter of elements is small compared to the diameter of discontinuous features, existing methods can safely assume a simple (linear, low-order, or trigonometric) shape to the piecewise discontinuity per element.

By contrast, a coarse mesh is both economical and sufficient for computing plausible animations. That a graphics application demands less precision and predictive power does not imply that a coarser, less-detailed output is acceptable; in fact, the converse is true—if we compare the desired level of feature resolution to computational investment, graphics applications are far more demanding. An XFEM method for graphical simulation of shells operates in a different regime characterized by more dense and more complex discontinuities. In our setting, assuming a simple shape (a line or quadratic curve) for the discontinuity is unnecessarily limiting, and adopting a hierarchical structure to keep track of multiple cuts is needlessly cumbersome.

In this paper, we extend the current body of work on XFEM toward the goals of graphical simulation. We introduce recent developments in XFEM treatments of thin shells, and we describe several developments novel to both the graphics and engineering literature. Our main contributions are

- a unified XFEM framework based on *harmonic enrichment functions*, which allows for multiple, intersecting, and arbitrarily-shaped cuts per element, while being easy to define, compute, and use;
- a discrete representation of the enrichment functions using *enrichment textures*, which simplifies specification and computation of cuts and paves the way for the future incorporation of now-standard GPU-based techniques;

- a complementary representation of spatially-varying material properties (e.g., thickness, stiffness, mass density) in *material textures*, which expand the expressive power of the simulator while keeping the same number of unknowns and the same simple grid-based numerical quadrature;
- a simulation of the material's dynamic response to time-varying discontinuities, built on our co-rotational extension of Noels's linear DG FEM thin shells.

While our method does not require a-priori knowledge on the resulting discontinuity, we do not implement a full fracture model and restrict our examples to pre-scored fracture lines and cuts. Furthermore, we do not address collision handling, which remains an interesting direction for future work.

2 Related Work

The simulation of cutting and fracture of deformable objects dates back to the pioneering work of Terzopoulos et al. [1987; 1988], and was brought to the forefront by O'Brien and Hodgins [1999] in their work on brittle (and ductile [2002]) fracture of volumetric elastica. While some have considered purely procedural approaches [Desbenoit et al. 2005], we restrict our discussion to physically simulated material response in the presence of cuts and fracture.

When combined with FEM, the topological changes induced by cutting pose two distinct challenges: first, to adapt basis functions, boundary conditions, and to update the stiffness matrix; second, to restructure the underlying mesh connectivity so as to represent the newly-created surfaces at a high resolution. Most research has focused on solid objects, where the problem of updating mesh connectivity is particularly challenging. We can roughly distinguish between mesh-based methods, and, more recently, meshless cutting. The most simple mesh-based algorithms either remove the primitives of the underlying tetrahedral mesh touched by the cutting plane [Forest et al. 2002] or fracture the mesh along tetrahedral boundaries and predefined positions [Terzopoulos and Fleischer 1988]. Others, including Bielser et al. [1999], subdivide primitives to better represent the cut boundary. Molino et al. [2004] present a virtual node algorithm that keeps invariant both the shape of (reference) elements and the nodal masses, and accounts for partially-void elements, thus enabling stable simulation of sliver-producing fracture; building on this, Sifakis et al. [2007] embed a high-resolution two-dimensional material boundary mesh into a coarser tetrahedral mesh, allowing for fast, coarse simulation of elasticity while retaining fine boundary features. These two works served as the foundation for the efficient fracture of rigid materials proposed by Bao et al. [2007]. Another mesh-based approach allows for FEM bases over more general meshes, for example over arbitrary polyhedra [Wicke et al. 2007; Martin et al. 2008]. To avoid the computational complexity of mesh restructuring, Pauly et al. [2005] proposed a point based method to track fracture surfaces in solids. Steinemann et al. [2006] suggest a hybrid approach with meshless deformation and mesh-based surface representations.

The graphical simulation of shell cutting and fracture has recently received specific attention. Mesh-based methods such as those of Boux de Casson and Laugier [2000], who tear discrete models of cloth [Baraff and Witkin 1998; Choi and Ko 2002], and Gingold et al. [2004], who fracture discrete shells [Grinspun et al. 2003], split meshes along existing edges. Müller [2008] describes a fast method for simulating tearing cloth using position based dynamics. Guo et al. [2006] and Wicke et al. [2005] propose meshless methods so as not to be limited by mesh connectivity. Our approach differs in that we do target the widely-used, mesh-based setting, but seek to do so without restricting cuts to mesh resolution. For this we turn to basis enrichment in an FEM context.

The CHARMS framework enriches the FEM basis [Grinspun et al. 2002] and supports topological changes, but cutting and fracture are not considered. The XFEM method, introduced by Belytschko and Black [1999], explicitly targets fracture. Since XFEM has been explored over the past decade, a more comprehensive summary requires a thorough survey, such as the one by Abdelaziz and Hamouine [2008]; here we discuss representative works. Moës et al. [1999] explicitly consider multiple *straight-line* crack tips within one element. Moës et al. [2002] further studied curved crack tips in three dimensions. Huang et al. [2003] simulate multiple cracks, but the example problem (mudcracks) assumes that cracks do not intersect. Stazi et al. [2003] consider higher-order elements and quadratic cracks. In general, this body of work employs (in order to accurately resolve strain) elemental radii orders of magnitude smaller than the characteristic radii of crack shapes; furthermore, for improved quadrature, they partition a split element into subdomains (sometimes with a hierarchical construction). This makes it challenging to consider complex cut shapes. We consider the diametric opposite, finely-detailed cuts *inside* one or a few elements.

Beyond enriching the basis, we must also consider material response. Graphical models of thin shell behavior include Discrete Shells [Grinspun et al. 2003], cloth with non-flat rest state [Bridson et al. 2003], subdivision (Kirchhoff-Love) shells [Cirak et al. 2000], and most recently inextensible cloth and shells [Goldenthal et al. 2007; English and Bridson 2008]. The DG FEM method [Arnold et al. 2001; Cockburn 2003], which is widely applied to discretize elastica, was recently applied in graphics to simulate arbitrary polyhedral elements [Kaufmann et al. 2008], and was recently applied in mechanics to simulate linear [Noels and Radovitzky 2008] and non-linear [Noels 2009] shells. A combination of DG FEM and XFEM was studied in [Gracie et al. 2008].

3 DG FEM Thin Shells

This section gives a brief overview of the shell formulation and discretization used in our simulations. Since our main contribution is the XFEM enrichment framework, we will only give the key equations and refer the reader to the literature and the technical report [Kaufmann et al. 2009] for further details and implementation notes.

We employ Kirchhoff-Love thin shells, where shearing deformations are neglected, such that a vector normal to the shell surface always stays normal to the shell during deformation [Wempner and Talaslidis 2003]. As a consequence, the configuration of a shell can be fully described by the mid-surface of the shell.

The undeformed mid-surface is parameterized as a bivariate function $\boldsymbol{\varphi}_0(\xi^1, \xi^2) : \Omega \rightarrow \mathbb{R}^3$ embedded in $\mathbb{R}^3$. The current (deformed) state $\boldsymbol{\varphi}$ can be described by the undeformed configuration and a displacement field $\mathbf{u}$ as

$$\boldsymbol{\varphi}(\xi^1, \xi^2) = \boldsymbol{\varphi}_0(\xi^1, \xi^2) + \mathbf{u}(\xi^1, \xi^2). \tag{1}$$

From computing stresses and strains in shell coordinates and linearizing in the displacement $\mathbf{u}$, the bilinear form for thin shell elasticity is found to be

$$a(\mathbf{u}, \mathbf{v}) = \int_\Omega \epsilon(\mathbf{v}) : \mathcal{H}_n : \epsilon(\mathbf{u}) \, \mathrm{d}A + \int_\Omega \rho(\mathbf{v}) : \mathcal{H}_m : \rho(\mathbf{u}) \, \mathrm{d}A \,, \tag{2}$$

with ε and ρ the stretching and bending strain tensors and ":" denoting the tensor dot product (double contraction). $\mathcal{H}_n$ and $\mathcal{H}_m$ are the stretching and bending constitutive tensors, respectively, that describe the relation between stresses and strains. For a detailed derivation of the weak form, we refer to [Cirak et al. 2000].

In order to solve (2) using the finite element method (FEM), the domain Ω is discretized into a mesh of finite elements (typically triangles or quads). Each of the n mesh nodes $\boldsymbol{\xi}^a$ is associated a nodal basis function N^a. An approximate solution $\mathbf{u}$ is then specified by nodal displacements $\mathbf{u}^a$:

$$\mathbf{u}(\boldsymbol{\xi}) = \sum_{a=1}^{n} N^a(\boldsymbol{\xi})\mathbf{u}^a. \tag{3}$$

Representing the solution $\mathbf{u}$ and the test function $\mathbf{v}$ in terms of the basis functions N^a results in a linear system

$$\mathbf{K}\cdot\mathbf{U} = \mathbf{F}, \quad \text{with} \quad \begin{cases} \mathbf{K}_{ij} &= a(\mathbf{I}_3 N^i, \mathbf{I}_3 N^j) \\ \mathbf{F}_i &= \int_\Omega \mathbf{f}\ N^i \end{cases}, \tag{4}$$

which is solved for the degrees of freedom $\mathbf{U} = (\mathbf{u}^1, \ldots, \mathbf{u}^n)^T$, given external forces $\mathbf{f}$. For more details on FEM we refer the reader to [Hughes 2000].

Since thin shells are governed by a fourth order elliptic PDE, appropriate basis functions N^a must lie in the Sobolev space $H^2(\Omega)$, i.e., have square-integrable first and second order derivatives. Unfortunately, constructing suitable shell basis functions on irregular meshes is a rather complex task.

This motivates the use of so-called discontinuous Galerkin (DG) FEM [Arnold et al. 2001; Cockburn 2003], which relaxes the compatibility condition between incident elements in favor of a penalty term that *weakly* enforces the required level of continuity. While DG FEM is broadly explored in mechanics, its use in thin shell simulations is relatively recent [Noels and Radovitzky 2008].

DG FEM allows us to use C^0 continuous basis functions in (2), implemented as 8-node bi-quadratic elements (as proposed in [Noels and Radovitzky 2008]) and 6-node quadratic triangle elements. C^1 continuity across element edges is weakly enforced by penalizing the difference between the *change* of normal vectors on neighboring elements. In addition to the per-element bending and membrane contributions of (2), this results in a per-edge penalty term. An exact definition of the DG weak form as well as a detailed description of the assembly of the stiffness matrix $\mathbf{K}$ can be found in the technical report [Kaufmann et al. 2009]. Since the linear strain measure used in [Noels and Radovitzky 2008] is not suitable for the large deformations required in graphics applications, we extend their method by a corotational strain measure, as described in [Kaufmann et al. 2009].

Given the stiffness matrix $\mathbf{K}$, a dynamic simulation of time-varying forces and displacements is governed by the equations

$$\mathbf{M}\ddot{\mathbf{U}} + \mathbf{D}\dot{\mathbf{U}} + \mathbf{K}\mathbf{U} = \mathbf{F}\,, \tag{5}$$

which are solved by semi-implicit Euler integration. Note that instead of lumping element masses to nodes, the full mass matrix $\mathbf{M}_{ij} = \int_\Omega N^i N^j$ is used, such that nodal basis functions and XFEM enrichment functions can later be handled in a uniform way. Moreover, if an element will be split into multiple parts, these will not only have the correct mass, but also the correct center of mass and moment of inertia.

The DG FEM discretization of Kirchhoff-Love shells described so far allows for simple C^0 basis functions and shows a plausible, geometrically nonlinear deformation behavior (see Fig. 2). However, to enable the simulation of highly detailed cuts or creases, as shown in Fig. 1, without excessively refining the underlying simulation mesh, we propose an XFEM approach that is described in the following sections.

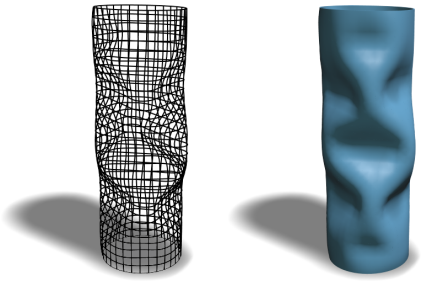

Figure 2: *Co-rotational DG FEM shells feature realistic deformations, including geometrically-nonlinear buckling effects.*

4 XFEM Basics

Introduced in [Belytschko and Black 1999], the extended finite element method (XFEM) builds up on the partition of unity concept [Babuska and Melenk 1996]. The basic idea of XFEM consists in *enriching* an element by splitting its basis functions along a desired discontinuity. This effectively doubles the element's degrees of freedom and decouples the solutions on either side of the discontinuity. In an elasticity simulation, this allows a single element to be cut or fractured into two independent parts.

Splitting the original basis functions along the discontinuity results in the *canonical basis*, as shown for a 1D example in Fig. 3. We can define the enriched basis functions as the product of an original basis function $N^a(\boldsymbol{\xi})$ and a so-called *enrichment function* $\psi^a(\boldsymbol{\xi})$. Arguably the simplest choice for the enrichment functions $\psi^a(\boldsymbol{\xi})$ is the Heaviside function $H_s(\boldsymbol{\xi})$, which assumes the value of 0 on one side of the cut and 1 on the other side.

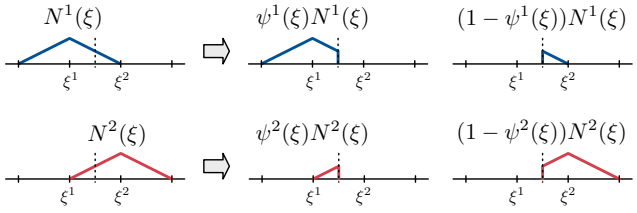

Figure 3: *The canonical basis: Basis functions N^a are split along the discontinuity to handle both parts separately.*

Note that the function space spanned by $\psi^a N^a$ and $(1-\psi^a)N^a$ is the same as the one spanned by N^a and $\psi^a N^a$, and it does not depend on which side of the discontinuity we choose the Heaviside function to vanish. The splitting of the basis functions can therefore be realized by adding to the original basis functions N^a the *enrichment basis functions* $\psi^a N^a$ with their corresponding degrees of freedom $\mathbf{a}^a$. This yields the enriched displacement field

$$\mathbf{u}(\boldsymbol{\xi}) = \sum_{a=1}^{n} N^a(\boldsymbol{\xi})\mathbf{u}^a + \sum_{a=1}^{n} \psi^a(\boldsymbol{\xi})N^a(\boldsymbol{\xi})\mathbf{a}^a. \tag{6}$$

XFEM therefore modifies the functional representation on a sub-element level instead of changing the topology of the element mesh. While traditional methods would have to divide existing elements into smaller elements in order to resolve the geometry of the cut, XFEM achieves the same goal without complex remeshing.

Instead of the simple Heaviside function $H_s(\boldsymbol{\xi})$, Zi and Belytschko [2003] proposed to use *shifted enrichment functions*

$$\psi^a(\boldsymbol{\xi}) = H_g(\boldsymbol{\xi}) - H_g(\boldsymbol{\xi}^a), \tag{7}$$

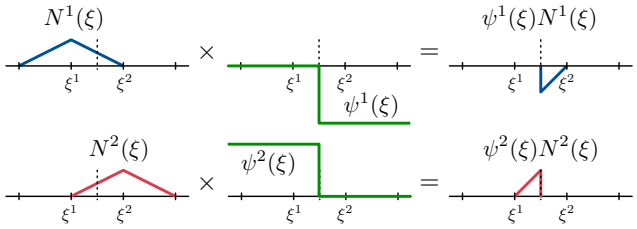

Figure 4: *Enrichment basis functions are obtained by multiplying the original basis functions by shifted Heaviside functions.*

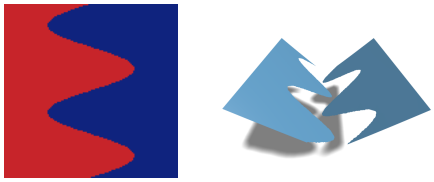

Figure 5: *A complete cut through a* single *element. The enrichment texture (*128×128*) is a generalized Heaviside (left), taking on values of* -1 *(red) or* $+1$ *(blue) on each side of the cut. A simulation of this enriched element is shown on the right.*

where $H_g(\boldsymbol{\xi}) = 2H_s(\boldsymbol{\xi}) - 1$ is the generalized Heaviside function (signum function), and $\boldsymbol{\xi}^a \in \Omega$ is the position of node a in parameter space. See Fig. 4 for a 1D illustration. The shifted enrichment basis functions together with the original basis functions span the same space as in the case of the unshifted enrichment functions. They are thus equivalent to the canonical basis.

Defining the enrichment functions this way has two desirable properties. First, it keeps the enrichment local, in the sense that the enrichment basis functions are only supported within the cut element. Compare this to Fig. 3, where the enrichment basis functions extend to all elements incident to the enriched node. This locality simplifies implementation and, according to [Jerabkova and Kuhlen 2009], also improves the stability of the simulation.

Second, shifting the enrichment functions recovers the Kronecker-delta property. The shifted enrichment function ψ^a vanishes at its associated node a, and as the nodal basis function N^a vanishes at all nodes other than a, the enrichment basis functions $\psi^a N^a$ vanish at *all* nodal positions $\boldsymbol{\xi}^a$. As a consequence, the displacement at each node a is fully defined by $\mathbf{u}^a$ alone. This simplifies the implementation of Dirichlet boundary conditions, as the displacement of a node can be fixed to a specific value by constraining a single degree of freedom.

5 Enrichment Textures

Many researchers have considered various ways to generalize the Heaviside function H_g and the shifted enrichment functions ψ^a from 1D to 2D or 3D. In most approaches, discontinuities are represented by simple analytical functions, such as cutting planes for 3D elements and linear or quadratic cutting paths for 2D elements. This representation, however, is too restrictive for graphics simulations, as it prevents us from having highly detailed, complex cuts through rather coarse elements of the simulation mesh.

To overcome this limitation, we represent the cut path as well as the generalized Heaviside function as piecewise linear functions on a regular 2D grid, which will be referred to as *enrichment texture* in the following. Discontinuities will be represented as an edge path between texture pixels (*texels*). The texture values are interpreted as (samples of) the generalized Heaviside function $H(\boldsymbol{\xi})$, and replaces $H_g(\boldsymbol{\xi})$ in (7).

Defining the enrichment function via a texture map has two important advantages. First, compared to analytic descriptions, it is better suited for modeling complex cutting paths, since the complexity of cuts is only limited by the texture resolution. Second, when dealing with progressive cuts, partial cuts, or multiple, intersecting cuts—required for cutting and fracturing simulations—it allows to handle the generation of the associated enrichment functions in a straightforward and uniform manner.

In this section we will focus on *complete cuts*, i.e., cuts that completely split an element into two pieces, or into several pieces in case of multiple cuts. The more complex problem of partial and progressive cuts will be discussed in Section 6.

Single Cut. In the simplest case, for a single, complete cut through an element, the enrichment texture stores a value of -1 or $+1$ for each texel, indicating which side of the cut the texel lies on (see Fig. 5, left). This defines the Heaviside $H(\boldsymbol{\xi})$, which we use to construct the enrichment functions $\psi^a(\boldsymbol{\xi})$ as in (7). Multiplication with the original basis functions yields the enriched basis functions $\psi^a(\boldsymbol{\xi})N^a(\boldsymbol{\xi})$. Their partial derivatives can easily be computed by finite differences on the regular texture grid. The functions and their gradients are then integrated over the element to construct $\mathbf{K}$ and $\mathbf{F}$, as shown in (4). To this end, the numerical integrator can simply sample the texture values at the quadrature points. These must be chosen densely enough so that all features of the enrichment texture are captured. In the limit, this corresponds to one quadrature point per texel.

Rendering. During a dynamic simulation, (5) is integrated in time, which in each time step yields the displacement $\mathbf{u}(\boldsymbol{\xi})$ at the spatial resolution of the enrichment textures. For rendering the deformed and cut shell surface, we tessellate each element of the simulation mesh into a fine triangulation, in order to more accurately approximate the quadratic displacement function $\mathbf{u}(\boldsymbol{\xi})$ as well as the high-resolution cuts. Note, however, that each element uses the same *static* triangulation, thereby minimizing storage and overhead.

One option would be to generate a vertex for each texel, and have the triangulation implicitly defined by the regular texture grid, which would be conceptually similar to geometry images [Gu et al. 2002]. However, in order to reduce the number of triangles, while still being able to resolve features at the texel level, we employ a geometry shader. This shader duplicates triangles that have been cut: Triangle vertices that lie on the opposite side of the cut are linearly extrapolated, and the triangle is then rendered with a fragment shader that masks out texels on the opposite side of the cut (see Fig. 6). An edge of a triangle is considered to be cut if two criteria are met. First, the magnitude of the difference between the enrichment function values at the edge's vertices must exceed a certain threshold, and second, the sign of the enrichment function must be different at these two vertices. Fig. 5 shows an example of a complete cut through a single element, using an enrichment texture of resolution 128×128. Note that this approach limits the resolu-

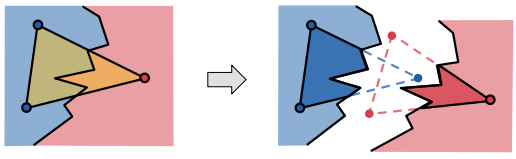

Figure 6: *A cut triangle (left) is duplicated and rendered once for each side of the cut (right)*

Figure 7: *Two complete, intersecting cuts through a single element, resulting in three separate connected components.*

Figure 8: *A* partial *cut in a single element: The corresponding Laplace problem (left), the resulting harmonic enrichment texture (center), and the element behavior in a simulation (right).*

tion of the cut to the texture resolution. Using texture filtering, the resulting aliasing artifacts can be mitigated to some degree.

Multiple Cuts. When considering *multiple* cuts within a single element, a cut is called complete if it starts and ends either at the element's boundary or at a junction with another cut. Multiple complete cuts therefore decompose the element into several connected components C_i.

The straightforward generalization of the canonical basis (see Fig. 3) is defined in terms of *canonical Heaviside* functions $H_c^i(\boldsymbol{\xi})$, which are 1 within their corresponding component C_i and 0 everywhere else. From these functions one can construct the shifted enrichment functions and from there on proceed like in the case of a single cut. Fig. 7 shows an example of two intersecting complete cuts, resulting in three physically independent surface components.

Note that thanks to our discrete texture representation the identification of connected components can be done by a trivial flood fill algorithm. In contrast, when representing cuts analytically, this task is more complex, such that often a hierarchical representation and classification of cuts has to be used.

6 Progressive Cutting

After showing the simplicity and flexibility of enrichment textures for *complete* cuts, we now discuss the more interesting case of *partial*, *progressive* cuts. In this case, a cut does not (yet) fully separate the element into two independent components.

For such configurations, works in the mechanics literature typically require to classify a discontinuity into either a *tip* (partial cut), a *complete* cut, or a *junction* (intersecting cuts), and generate special enrichment functions depending on the type of discontinuity. When furthermore allowing for multiple cuts within a single element, with cuts being either partial or complete, the combinatorics of handling these different cases in a hierarchical representation inevitably leads to rather complex implementations.

In this section we propose our *harmonic enrichment* approach, which is conceptually simpler, in that it uses only one kind of unified enrichment function that does not depend on the type of cut, and that generalizes to multiple, partial, progressive, and complete cuts in a natural and canonical manner. We again start with a single cut, then discuss multiple cuts within one element, and describe how to handle cuts that intersect multiple elements.

6.1 Single Cut

Since a partial cut does not fully split the element into two components, separation of material is only allowed along the curve covered by the cut so far. However, as soon as a progressive cut has traversed the whole element and hence became a complete cut, we want the parts on either side of the cut to be independent of each other, as shown in the previous section. This translates into two conditions for the generalized Heaviside function:

- For a partial cut, the enrichment (and the Heaviside) should be discontinuous along the cut but continuous everywhere else, allowing material to separate at the cut only.
- If the progressive cut becomes a complete cut, the value of the enrichment function must be constant on either side of the cut, replicating the behavior of the Heaviside functions of the previous section.

In mechanics, one of the classical crack tip functions is $\sqrt{r}\sin(\theta/2)$ [Belytschko and Black 1999], where (θ, r) are polar coordinates centered at the crack tip. This function takes on values of $\pm\sqrt{r}$ on either side of the crack. However, this function does not easily generalize to complex cut shapes and multiple cuts within a single element. We therefore generalize the classical enrichment function by observing that it is a harmonic function and defining *harmonic enrichments* as the solution to the Laplace equation

$$\Delta H(\boldsymbol{\xi}) = 0, \tag{8}$$

subject to suitable boundary conditions along the crack tip and the element boundary. On the element boundary, we prescribe vanishing Neumann constraints $\nabla H(\boldsymbol{\xi}) \cdot \mathbf{n}(\boldsymbol{\xi}) = 0$ (see Fig. 8).

Along the crack, prescribing the Dirichlet conditions $H(\boldsymbol{\xi}) = \pm\sqrt{r}$ recovers the classical harmonic enrichment function $\sqrt{r}\sin(\theta/2)$ in the vicinity of the crack tip. In particular, since a harmonic reconstruction reproduces given boundary conditions (tautologically), it follows that the influence of any other finitely-distant boundary can be ignored in a sufficiently small neighborhood of the tip. As a simplification, we assume that the $\sqrt{r}$-decay is concentrated within the crack tip texel; thus, we prescribe the Dirichlet conditions $H(\boldsymbol{\xi}) = \pm 1$ on the texels incident to the cut. The simplified condition works well in our experiments; however, if adaptive refinement of the enrichment texture is used, the $\sqrt{r}$-decay should be explicitly considered, since the Dirichlet conditions ± 1 are ill-posed in the smooth limit.

Thanks to our texture representation, (8) can be discretized by a simple finite difference scheme. As the cut corresponds to a series of connected edges between texels, the Dirichlet conditions set the texels incident to a cut edge to -1 or 1, respectively. The resulting sparse linear system has to be solved for the function values of $H(\boldsymbol{\xi})$ at each texel, which we do using either a multigrid solver or a sparse Cholesky factorization [Toledo et al. 2003].

As the solution of a Laplace equation, $H(\boldsymbol{\xi})$ will be harmonic, i.e., continuous and smooth, everywhere except at the Dirichlet constraints, where it shows the desired discontinuity. Hence, the first requirement is fulfilled. In the case of a complete cut, the vanishing Neumann boundary conditions cause $H(\boldsymbol{\xi})$ to replicate a Heaviside with constant ± 1 on either side of the cut. Furthermore, as the cut approaches the element boundary, the enrichment function converges to this Heaviside in a temporally smooth manner. As a consequence, the second criterion is satisfied. The enrichment texture of the partial cut in Fig. 8 would eventually converge to the one of the complete cut shown in Fig. 5.

Figure 9: Multiple progressive *cuts within an element: The harmonic enrichment textures $H^i(\boldsymbol{\xi})$ for each of the cuts (left, center), and the resulting simulation behavior (right).*

6.2 Multiple Cuts

With just a slight modification of the boundary constraints, the harmonic enrichment approach can be generalized from single cuts to multiple cuts within an element. Consider n (partial or complete) cuts $\mathbf{c}_i(\xi)$, $1 \leq i \leq n$, in an element. We have to construct an enrichment function for each cut, i.e., we have to find a generalized Heaviside $H^i(\boldsymbol{\xi})$ for each cut $\mathbf{c}_i(\xi)$. The two requirements formulated for single cuts can be extended to multiple cuts as follows:

- The enrichment function $H^i(\boldsymbol{\xi})$ should be discontinuous across its cut $\mathbf{c}_i(\xi)$, and continuous for unconstrained texels. $H^i(\boldsymbol{\xi})$ might be discontinuous across $\mathbf{c}_j(\xi)$ for $j \neq i$.
- If all cuts are *complete* cuts, they partition the element into $n+1$ separate components $C_1, \ldots, C_{n+1}$. The original basis functions and their n enriched versions should reproduce the behavior of the canonical basis (see Fig. 7), i.e., they should span the same function space.

We extend the definition of harmonic enrichment functions for multiple cuts as follows: Each enrichment function $H^i(\boldsymbol{\xi})$ is again defined as the solution of

$$\Delta H^i(\boldsymbol{\xi}) = 0. \tag{9}$$

As in the single cut case, the discontinuity across its corresponding cut $\mathbf{c}_i(\xi)$ is enforced through Dirichlet conditions of ± 1. However, now we enforce vanishing Neumann conditions not only on the element boundary, but also on all other cuts $\mathbf{c}_j(\xi)$, $j \neq i$, i.e.

$$\nabla H^i(\boldsymbol{\xi}) \cdot \mathbf{n}(\boldsymbol{\xi}) = 0, \tag{10}$$

where $\mathbf{n}$ is the normal of either the domain boundary or another cut $\mathbf{c}_j(\xi)$ respectively. Note that there might be components with Neumann conditions only. These components are independent of the cut $\mathbf{c}_i$ under consideration, and we therefore explicitly set the function values in these regions to zero. The resulting Heaviside functions $H^i(\boldsymbol{\xi})$ again trivially satisfy the first condition by construction. A simple proof showing that the second condition is also fulfilled is given in Appendix A. See Fig. 9 for an example of two partial cuts, which would eventually converge to the situation depicted in Fig. 7.

Harmonic enrichment textures therefore allow us to handle both single and multiple cuts, as well as partial, progressive, and complete cuts in a simple, unified, and canonical manner. In contrast to existing work, we do not require cuts to be classified into crack tips, joints, or complete cuts. Neither do we need a complex hierarchical representation to keep track of multiple cuts and the resulting connected components.

6.3 Multiple Elements

So far, we described how to handle single or multiple, partial or complete cuts within a *single element*. For realistic simulations, however, a simulation mesh of more than one element is of course required. In this section we extend our enrichment approach to multiple elements partitioning the simulation domain Ω. A 2D illustration of a multi-element mesh with two complete cuts $\mathbf{c}_1$, $\mathbf{c}_2$ and one partial cut $\mathbf{c}_3$ is shown in Fig. 10.

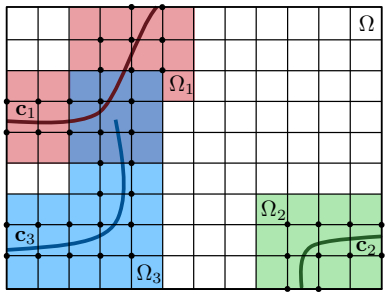

Figure 10: *Three cuts define nodes to be enriched (dots) and associated domains Ω_i on which the H^i are defined. Nodes in overlapping domains are enriched multiple times.*

In order to model the discontinuities caused by a cut, all elements intersected by this cut are enriched. This is achieved by enriching the nodal basis functions $N^a(\boldsymbol{\xi})$ of each of the element's vertices a, resulting in $\psi^a(\boldsymbol{\xi})N^a(\boldsymbol{\xi})$. This expression requires the enrichment function ψ^a, and therefore the Heaviside H (see (7)), to be defined on the whole support of N^a, which in our case is just the set of elements incident to vertex a. The (overlapping) colored regions Ω_i in Fig. 10 highlight the elements that are enriched due to the respective cuts $\mathbf{c}_i$, the dots represent the vertices to be enriched.

The main difference to the single element case is that the enrichment functions of neighboring elements have to be C^0 continuous across their shared edge, in order to guarantee the enrichment basis functions $\psi^a N^a$, and thus the enriched displacement $\mathbf{u}$ (see (6)), to be C^0, as required by our DG FEM formulation.

For *complete cuts*, such as $\mathbf{c}_1$ and $\mathbf{c}_2$ in Fig. 10, we can use the simple enrichment functions shown in Figs. 5 and 7 for the intersected elements, and constant enrichments of $+1$ or -1 for their neighboring elements. The resulting enrichment functions will be C^0 across mesh edges by construction.

The handling of *partial cuts*, such as $\mathbf{c}_3$ in Fig. 10, is slightly more involved. For the element containing the cut tip a harmonic enrichment is computed by solving (8) or (9), as discussed in Section 6.2. The vanishing Neumann boundary conditions for this system can, however, not guarantee C^0 continuity to its neighboring elements (as Dirichlet boundary constraints could do).

In order to ensure continuity, we additionally have to take the incident elements into account and solve the Laplace equation on this extended one-ring neighborhood (3×3 region around the tip of $\mathbf{c}_3$ in Fig. 10). On the boundary edges of this region we prescribe the values of neighboring enrichment functions, if they exist, as C^0 Dirichlet constraints, or vanishing Neumann conditions otherwise. As the neighboring enrichment function values were computed in previous simulation steps, this construction is free of cycles.

In Fig. 10 we would therefore compute the enrichment function(s) H^3 for the nine elements around the cut tip by a single Laplace system, prescribing ± 1 Dirichlet constraints along $\mathbf{c}_3$, vanishing Neumann constraints along $\mathbf{c}_1$, C^0 Dirichlet constraints on the boundary to the blue elements, and vanishing Neumann constraints on the other boundaries. In this 3×3 region the enrichment function H^1 for cut $\mathbf{c}_1$ also has to be computed through this system, by exchanging the roles of $\mathbf{c}_1$ and $\mathbf{c}_3$. With this slight modification of the domain and the constraints of the Laplace systems (8), (9), we are able to generalize the harmonic enrichment textures from single el-

ements to arbitrary simulation meshes, including quad and triangle meshes through the definition of appropriate texel neighborhoods at the element boundaries.

Note that our DG FEM formulation considerably simplifies this generalization as it only requires C^0 continuity on element edges. On the other hand, conforming thin shell discretizations using C^1 basis functions [Cirak et al. 2000; Thomaszewski et al. 2006] would require C^1 continuous enrichment functions, which cannot be computed through simple Laplace systems.

7 Results

We demonstrate our enrichment method on a variety of examples, including single elements, more complex element meshes, single progressive cuts, cuts consisting of multiple enrichments, and multiple cuts per element. The simulation meshes used consist of either 8-node bi-quadratic quadrilateral elements or 6-node quadratic triangular elements. All discontinuities were prescribed as polylines and then rasterized to texel edges. Dynamic simulations of the depicted examples can be found in the accompanying video.

Complex Cut Lines. Our method allows us to represent discontinuities on a fine sub-element level without the need to remesh. Fracture lines are a special instance of such discontinuity lines and are usually computed using local strain or stress measurements to predict their evolution. We do not directly simulate a complex fracturing model but generate the highly-detailed cut boundaries procedurally. Our approach is able to resolve the shells' reaction to the fracture discontinuity at the sub-element level, as shown in Fig. 11 and the accompanying video. Note that the simulation runs on a single quad element using bi-quadratic basis functions.

Figure 1, right, shows the same model applied to a non-trivial mesh with triangular quadratic elements. Note that the elements are shaded independently of each other in order to show the coarse resolution of the simulation mesh. To avoid these shading discontinuities, normals can be averaged at nodes and interpolated over the elements.

Small Features. The presented approach allows us to simulate small scale details down to the resolution of the texture. In Fig. 13, left, a circular cut is applied to a planar shell mesh consisting of four quads. The method can robustly handle the situation where the connection between the "hanging chad" and the rest of the element is only a few texels wide. Without requiring any special handling, the basis functions resulting from the harmonic enrichments introduce a specific additional mode of deformation, leading to the expected result. Figure 13, right, shows an example of a highly-detailed cut line demonstrating the capability to resolve small features.

Multiple Cuts. As described in Section 6.2, our method is also applicable to the case of multiple cuts per element. Figure 1, left, shows a single quad element that has been enriched 18 times, replicating a "paper accordion". In this example, the boundary conditions on the cuts are not purely ± 1 on either side of the cuts, but instead are blended to zero towards both ends of each cut to get an enrichment that better captures the desired deformation.

For long progressive cuts with complex trajectories, a single enrichment will not provide sufficient additional degrees of freedom to allow for the desired deformation. Such cases can be handled consistently by tracking the length or integral angle of the cut and by initiating a new cut once a predefined threshold is exceeded. While this criterion is mathematically not rigorous, it works very well in practice, and we employed it for our experiments.

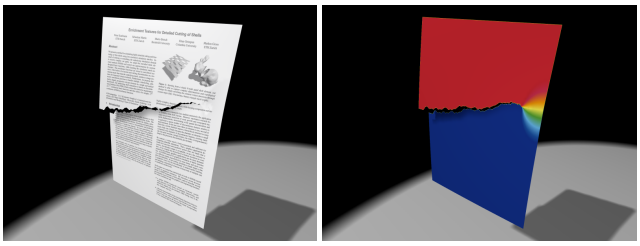

Figure 11: *A single textured quad element being fractured (left), and the underlying discontinuous enrichment function (right).*

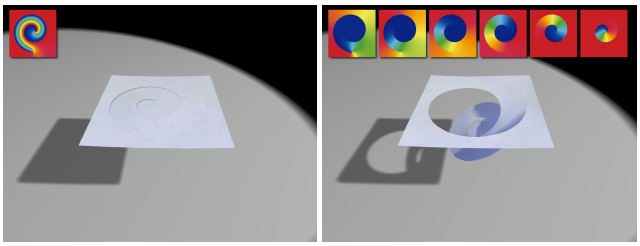

Figure 12: *A quad element is cut by a helical line, using a single enrichment (left) and six cut segments (right). The insets show the enrichment functions used.*

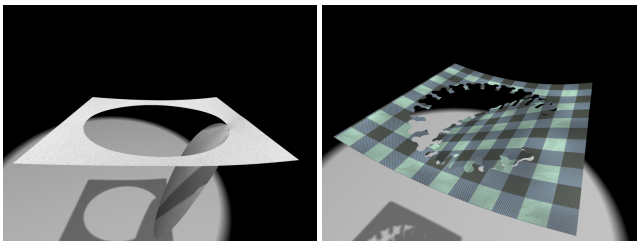

Figure 13: *Simulation of a chad failing to completely separate from a punchcard (left). Highly detailed cut on a tissue (right).*

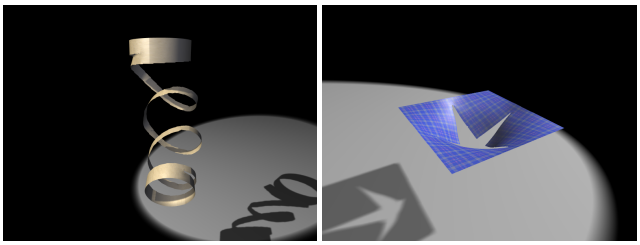

Figure 14: *A cylinder represented by four quad elements is cut into a slinky, deforms under gravity and uncoils (left). Simulation of intersecting cuts in a single element (right).*

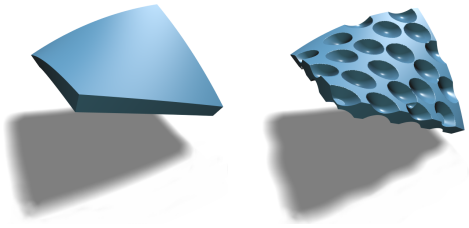

Figure 15: *The thickness of a shell element (left) is modulated by a thickness texture and deforms accordingly (right). The shell is rendered as a volumetric object in order to visualize its thickness.*

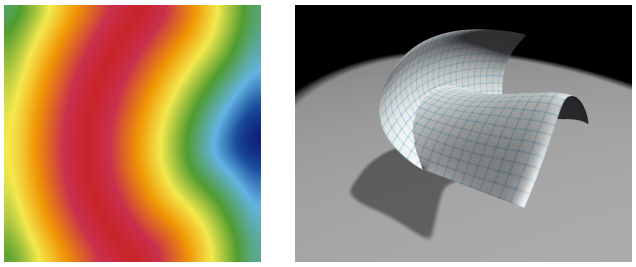

Figure 16: *A C^0 continuous enrichment texture with discontinuous first derivatives (left) allows adding a crease to an element (right).*

This results in a series of connected cuts that together make up the desired discontinuity and add the required local degrees of freedom. Figure 12 shows an example of a spiral cut enriched by a single enrichment texture (left), lacking the desired deformation, and the same curve divided into 6 cuts (right), resulting in a more plausible simulation. Applying the same principle to a less trivial mesh, a slinky is cut out of a cylinder in Fig. 14, left. Fig. 14, right, additionally shows an example of intersecting cuts. Note that once a cut crosses another cut, it becomes a complete cut that ends at the point of intersection and a new cut is started.

Material Textures. As an alternative application of textures besides cutting, we can define material properties at the texel level. In the case of Kirchhoff-Love shells, properties that can be modulated by such a *material texture* include the material's Young's modulus, Poisson's ratio, as well as the local thickness of the shell. During the integration of an element's stiffness matrix, the material texture is evaluated at the quadrature points. Figure 15 shows an example of a single shell element with locally reduced thickness, causing it to weaken and exhibit a stronger deformation under gravity. In this example, 12^2 quadrature points were used to integrate the element's stiffness matrix.

Creases. So far, we have only considered enrichment textures with discontinuities, allowing for cutting and separating shells into multiple independent components. However, we note that the concept of enrichment textures is more general. In particular, it can also be used to model sharp creases, at which the shell is free to bend. Creases can be modeled by keeping the enrichment textures C^0 continuous, but introducing discontinuities in the first derivatives. Fig. 16, left, shows an example of an enrichment texture for a curved crease line. The enrichment function is computed as the geodesic distance from the crease using a fast marching method [Sethian 1999]. Applying this enrichment texture to a shell element allows the element to bend around the crease, resulting in interesting states of minimal energy when external forces are applied (see Fig. 16, right, and the accompanying video).

Timings. Table 1 shows the computation times per simulation step for a representative selection of examples. The times for computing the enrichments are peak values, while the assembly and solve times were averaged over the whole simulation. For solving the Laplace problem during the computation of the enrichment functions, we employ a sparse Cholesky solver [Toledo et al. 2003].

Comparison to Discrete Shells. For typical meshes the cost of cutting and enrichment is only a small percentage of the total simulation time, e.g., 7% for the Bunny example. This percentage vanishes as mesh resolution increases, since the cost of dynamics increases while the texture area intersected by the cut decreases.

Model	#Els.	#Nodes	#Enr.	Tex.	t_{enr}	t_{asm}	t_{sol}
Chad	4	21	1	256	1433	16	3
Spiral 1	1	8	1	256	274	1.8	0.1
Spiral 6	1	8	6	256	1701	38	0.7
Accordion	1	8	18	256	4785	313	8
Bunny	552	1106	1	32	283	452	917
Slinky	4	20	12	128	5207	513	22

Table 1: *Comparison of timings for the computation of the enrichment functions (t_{enr}), assembly of the global stiffness matrix (t_{asm}), and solving the resulting linear system (t_{sol}). Timings are in milliseconds and were taken on an Intel Core2 Duo 2.4 GHz.*

Mesh (Method)	#Els.	#Nodes	#DOFs	#NNZ
Bunny (our method)	552	1106	3318	243954
Bunny (Discrete Shells)	4200	2102	6306	245934
Spiral (our method)	1	8	168	28224
Spiral (Discrete Shells)	385	274	822	32058
Chad (our method)	1	8	48	2304
Chad (Discrete Shells)	72	62	186	7254

Table 2: *Comparison of our method to Discrete Shells with a comparable computational budget (non-zero stiffness matrix entries).*

Focusing on the bulk of computational cost—simulating dynamics after cutting—Table 2 compares our method to Discrete Shells [Grinspun et al. 2003] in terms of computational effort. We use the number of non-zero entries in the stiffness matrix (NNZ) as a measure of complexity for solving the linear system during time integration. For Discrete Shells, the mesh resolutions were chosen to match the NNZs of our enriched simulations.

For the Discrete Shells simulations, the mesh resolution might be sufficient to simulate the elastic behavior, but it is not enough to resolve the cuts in sufficient detail, especially for highly detailed cuts. By contrast, our enrichment uses the same number of additional DOFs, whether the cut is highly jagged or straight. The comparison shows that our method excels in settings characterized by a higher ratio of cut to deformation complexity.

Discussion. The previous examples show that our method is able to represent complex cut lines and small features in single elements and element meshes, and that multiple cuts per element can be handled consistently.

However, we note that our current implementation of the method lacks a proper projection step. When the basis functions change due to enrichment, we simply apply the previous solution to the new basis. In most situations the introduced error is unnoticeable, since the enrichments change gradually over time. However, in some scenarios it leads to obvious "popping" artifacts, e.g., in the multiply enriched spiral cut in the accompanying video. These artifacts may be eliminated by projecting the displacements and velocities onto the new basis, as proposed, e.g., in [Réthoré et al. 2005].

8 Conclusion and Future Work

We presented a novel method for handling highly detailed discontinuity features such as cutting or fracture lines using a versatile, texture-based basis enrichment approach. The method spends new degrees of freedom in an economical manner and supports a uniform and general treatment of multiple progressive or complete discontinuities. While the proposed method focuses on introducing material discontinuities, we hope that the presented way of combining texture concepts with physical simulations opens exciting new areas for future work.

A promising direction for future work is to generalize the modeling of creases, building on the preliminary results presented in this work. We would also like to explore the synergies of textures representing both geometric and physical material properties. For example, a displacement texture map could represent both the physical rest state of a shell as well as allow for high surface detail while still only requiring a coarse simulation mesh.

We further note that most of the steps in the simulation pipeline are very amenable to parallelization, making them ideal candidates for computation on the GPU. In particular, the computation of the enrichment functions can be performed efficiently using a multigrid solver, whose pre- and post-smoothing steps can be interpreted as simple texture filtering operations [Bolz et al. 2003]. The corotated stiffness matrix integration could be computed on the GPU, since it parallelizes trivially, leading to a highly accurate integration at the texel level. At the rendering stage, the deformed geometry could easily be constructed on the GPU as well, by computing the displacement field from the enrichment textures and the solution to the dynamic simulation. We also note that at this stage, the rendered mesh could be enhanced trivially by adding fine-scale geometry.

Acknowledgments

The authors are grateful to N. Sukumar and Max Wardetzky for inspiring and helpful discussions and to the anonymous reviewers for their helpful comments and suggestions. This work was supported in part by the Swiss National Science Foundation (grant No. 200021-117756), by the NSF (MSPA Award No. IIS-05-28402, CSR Award No. CNS-06-14770, CAREER Award No. CCF-06-43268), and by the Deutsche Forschungsgemeinschaft (Center of Excellence in "Cognitive Interaction Technology", CITEC).

A Proof of 2nd Condition in Section 6.2

In this section we show for n complete cuts $\mathbf{c}_i$, the harmonic enrichment functions H^i, obtained by solving $\Delta H^i = 0$ for each cut $\mathbf{c}_i$ as described in Section 6.2, span the same function space as the canonical basis for the resulting $n+1$ components.

The *canonical Heaviside* functions H_c^i are the enrichment functions that produce the canonical basis. H_c^i is 1 in component C_i and 0 otherwise. The *canonical enrichment basis* of $n+1$ components is

$$\mathcal{C}_{n+1} \;=\; \{H_c^1, \ldots, H_c^{n+1}\}\,.$$

For an un-cut element, which is described by its original basis functions N^a only, the canonical enrichment function is $H_c^1 \equiv 1$. In order to proof that our enrichment basis functions span the same space as the canonical basis, it suffices to show that the space

$$\mathcal{H}_{n+1} \;=\; \{1, H^1, \ldots, H^n\}\,,$$

spanned by our harmonic enrichment functions H^i and the constant 1 function corresponding to the original basis, is equivalent to the space spanned by the canonical enrichment functions $\mathcal{C}_{n+1}$.

This can be shown by induction as follows: For the case with $n = 1$ components, the constant 1 function is the only enrichment function and all enrichments are trivially equivalent to the canonical basis.

In the inductive step we assume that for n components the set of enrichment functions $\mathcal{H}_n$ spans the same space as the canonical enrichments $\mathcal{C}_n$. Hence, $\mathcal{H}_n$ spans an n-dimensional space and the H^i are linearly independent.

Then a new cut $\mathbf{c}_n$ splits a component, say C_n, into two parts C_n and C_{n+1}, thus leading to $n+1$ components. The space $\mathcal{H}_{n+1}$ is computed by solving (9) for each $H^1, \ldots, H^n$. Note that the i-th functions of $\mathcal{H}_n$ and $\mathcal{H}_{n+1}$ ($1 \le i \le n-1$) are not equal, since the new cut imposes additional Neumann conditions at $\mathbf{c}_n$ for (9). However, the additional Neumann conditions cannot turn the new functions $H^1, \ldots, H^n$ linearly dependent. Consequently, $\mathcal{H}_{n+1} \setminus H^n$ spans an n-dimensional subspace. The additional new function H^n is linearly independent of $\mathcal{H}_{n+1} \setminus H^n$, since it is the only Heaviside that can control both sides of the new cut $\mathbf{c}_n$ (i.e., C_n and C_{n+1}) independently. This means that $\mathcal{H}_{n+1}$ must be a complete basis for an $(n+1)$-dimensional space and thus is equivalent to the canonical basis $\mathcal{C}_{n+1}$.

References

ABDELAZIZ, Y., AND HAMOUINE, A. 2008. A survey of the extended finite element. *Computers and Structures 86*, 11–12, 1141–1151.

ARNOLD, D. N., BREZZI, F., COCKBURN, B., AND MARINI, L. D. 2001. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM J. Numer. Anal. 39*, 5, 1749–1779.

BABUSKA, I., AND MELENK, J. M. 1996. The partition of unity finite element method: Basic theory and applications. *Comput. Meth. Appl. Mech. Eng., Special Issue on Meshless Methods 139*, 289–314.

BAO, Z., MO HONG, J., TERAN, J., AND FEDKIW, R. 2007. Fracturing rigid materials. *IEEE Transactions on Visualization and Computer Graphics 13*, 2, 370–378.

BARAFF, D., AND WITKIN, A. 1998. Large steps in cloth simulation. In *Proc. of ACM SIGGRAPH*, 43–54.

BELYTSCHKO, T., AND BLACK, T. 1999. Elastic crack growth in finite elements with minimal remeshing. *Int. J. Numer. Methods Eng. 45*, 5, 601–620.

BIELSER, D., MAIWALD, V., AND GROSS, M. 1999. Interactive cuts through 3-dimensional soft tissue. *Computer Graphics Forum (Proc. Eurographics) 18*, 3, 31–38.

BOLZ, J., FARMER, I., GRINSPUN, E., AND SCHRÖDER, P. 2003. Sparse matrix solvers on the GPU: conjugate gradients and multigrid. *ACM Trans. on Graphics (Proc. SIGGRAPH) 22*, 3, 917–924.

BRIDSON, R., MARINO, S., AND FEDKIW, R. 2003. Simulation of clothing with folds and wrinkles. In *Proc. of Symp. on Computer Animation '03*, 28–36.

CHOI, K.-J., AND KO, H.-S. 2002. Stable but responsive cloth. *ACM Trans. on Graphics (Proc. SIGGRAPH) 21*, 3, 604–611.

CIRAK, F., ORTIZ, M., AND SCHRÖDER, P. 2000. Subdivision surfaces: A new paradigm for thin-shell finite-element analysis. *Int. J. Numer. Methods Eng. 47*, 12, 2039–2072.

COCKBURN, B. 2003. Discontinuous Galerkin methods. *Z. Angew. Math. Mech. 83*, 11, 731–754.

DE CASSON, F. B., AND LAUGIER, C. 2000. Simulating 2D tearing phenomena for interactive medical surgery simulators. In *Computer Animation 2000*, 9–14.

DESBENOIT, B., GALIN, E., AND AKKOUCHE, S. 2005. Modeling cracks and fractures. *The Visual Computer 21*, 8-10, 717–726.

ENGLISH, E., AND BRIDSON, R. 2008. Animating developable surfaces using nonconforming elements. *ACM Trans. on Graphics (Proc. SIGGRAPH) 27*, 3, 1–5.

FOREST, C., DELINGETTE, H., AND AYACHE, N. 2002. Removing tetrahedra from a manifold mesh. In *Proc. of IEEE Computer Animation*, 225–229.

GINGOLD, Y., SECORD, A., HAN, J. Y., GRINSPUN, E., AND ZORIN, D. 2004. A discrete model for inelastic deformation of thin shells. Tech. rep., Courant Institute of Mathematical Sciences, New York University.

GOLDENTHAL, R., HARMON, D., FATTAL, R., BERCOVIER, M., AND GRINSPUN, E. 2007. Efficient simulation of inextensible cloth. *ACM Trans. on Graphics (Proc. SIGGRAPH) 26*, 3, 49:1–49:7.

GRACIE, R., WANG, H., AND BELYTSCHKO, T. 2008. Blending in the extended finite element method by discontinuous Galerkin and assumed strain methods. *Int. J. Numer. Methods Eng. 74*, 11, 1645–1669.

GRINSPUN, E., KRYSL, P., AND SCHRÖDER, P. 2002. CHARMS: A simple framework for adaptive simulation. *ACM Trans. on Graphics (Proc. SIGGRAPH) 21*, 3, 281–290.

GRINSPUN, E., HIRANI, A. N., DESBRUN, M., AND SCHRÖDER, P. 2003. Discrete shells. In *Proc. of Symp. on Computer Animation'03*, 62–67.

GU, X., GORTLER, S., AND HOPPE, H. 2002. Geometry images. *ACM Trans. on Graphics (Proc. SIGGRAPH) 21*, 3, 355–361.

GUO, X., LI, X., BAO, Y., GU, X., AND QIN, H. 2006. Meshless thin-shell simulation based on global conformal parameterization. *IEEE Transactions on Visualization and Computer Graphics 12*, 3, 375–385.

HUANG, R., SUKUMAR, N., AND PRÉVOST, J. 2003. Modeling quasi-static crack growth with the extended finite element method – Part II. *Int. J. of Solids and Structures 40*, 26, 7539–7552.

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

JERABKOVA, L., AND KUHLEN, T. 2009. Stable cutting of deformable objects in virtual environments using the XFEM. *IEEE Computer Graphics and Applications 29*, 2, 61–71.

KAUFMANN, P., MARTIN, S., BOTSCH, M., AND GROSS, M. 2008. Flexible simulation of deformable models using discontinuous Galerkin FEM. In *Proc. of Symp. on Computer Animation*, 105–115.

KAUFMANN, P., MARTIN, S., BOTSCH, M., AND GROSS, M. 2009. Implementation of discontinuous Galerkin Kirchhoff-Love shells. Tech. Rep. no. 622, Department of Computer Science, ETH Zurich.

MARTIN, S., KAUFMANN, P., BOTSCH, M., WICKE, M., AND GROSS, M. 2008. Polyhedral finite elements using harmonic basis functions. *Computer Graphics Forum (Proc. of Symp. on Geometry Processing) 27*, 5, 1521–1529.

MOËS, N., DOLBOW, J., AND BELYTSCHKO, T. 1999. A finite element method for crack growth without remeshing. *Int. J. Numer. Methods Eng. 46*, 1, 131–150.

MOËS, N., GRAVOUIL, A., AND BELYTSCHKO, T. 2002. Non-planar 3d crack growth by the extended finite element and level sets - Part I. *Int. J. Numer. Methods Eng. 53*, 11, 2549–2568.

MOLINO, N., BAO, Z., AND FEDKIW, R. 2004. A virtual node algorithm for changing mesh topology during simulation. *ACM Trans. on Graphics (Proc. SIGGRAPH) 23*, 3, 385–392.

MÜLLER, M. 2008. Hierarchical position based dynamics. In *Proc. of Virtual Reality Interactions and Physical Simulations*.

NOELS, L., AND RADOVITZKY, R. 2008. A new discontinuous Galerkin method for Kirchhoff-Love shells. *Computer Methods in Applied Mechanics and Engineering 197*, 2901–2929.

NOELS, L. 2009. A discontinuous Galerkin formulation of non-linear Kirchhoff-Love shells. *Int. J. Numer. Methods Eng. 78*, 3, 296–323.

O'BRIEN, J. F., AND HODGINS, J. K. 1999. Graphical modeling and animation of brittle fracture. In *Proc. of ACM SIGGRAPH*, 137–146.

O'BRIEN, J. F., BARGTEIL, A. W., AND HODGINS, J. K. 2002. Graphical modeling and animation of ductile fracture. *ACM Trans. on Graphics (Proc. SIGGRAPH) 21*, 3, 291–294.

PAULY, M., KEISER, R., ADAMS, B., DUTRE, P., GROSS, M., AND GUIBAS, L. J. 2005. Meshless animation of fracturing solids. *ACM Trans. on Graphics (Proc. SIGGRAPH) 24*, 3, 957–964.

RÉTHORÉ, J., GRAVOUIL, A., AND COMBESCURE, A. 2005. An energy-conserving scheme for dynamic crack growth using the extended finite element method. *Int. J. Numer. Methods Eng. 63*, 5, 631–659.

SETHIAN, J. 1999. *Level Set Methods and Fast Marching Methods*. Cambridge University Press.

SIFAKIS, E., DER, K. G., AND FEDKIW, R. 2007. Arbitrary cutting of deformable tetrahedralized objects. In *Proc. of Symp. on Computer Animation*, 73–80.

STAZI, F. L., BUDYN, E., CHESSA, J., AND BELYTSCHKO, T. 2003. An extended finite element method with higher-order elements for curved cracks. *Comput. Mech. 31*, 1, 38–48.

STEINEMANN, D., OTADUY, M. A., AND GROSS, M. 2006. Fast arbitrary splitting of deforming objects. In *Proc. of Symp. on Computer Animation*, 63–72.

TERZOPOULOS, D., AND FLEISCHER, K. 1988. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. In *Proc. of ACM SIGGRAPH*, 269–278.

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. 1987. Elastically deformable models. In *Proc. of ACM SIGGRAPH*, 205–214.

THOMASZEWSKI, B., WACKER, M., AND STRASSER, W. 2006. A consistent bending model for cloth simulation with corotational subdivision finite elements. In *Proc. of Symp. on Computer Animation*, 107–116.

TOLEDO, S., CHEN, D., AND ROTKIN, V., 2003. Taucs: A library of sparse linear solvers. http://www.tau.ac.il/∼stoledo/taucs.

WEMPNER, G., AND TALASLIDIS, D. 2003. *Mechanics of solids and shells: theories and approximations*. CRC Press.

WICKE, M., STEINEMANN, D., AND GROSS, M. 2005. Efficient animation of point-sampled thin shells. *Computer Graphics Forum (Proc. Eurographics) 24*, 667–676.

WICKE, M., BOTSCH, M., AND GROSS, M. 2007. A finite element method on convex polyhedra. *Computer Graphics Forum (Proc. Eurographics) 26*, 3, 355–364.

ZI, G., AND BELYTSCHKO, T. 2003. New crack-tip elements for XFEM and applications to cohesive cracks. *Int. J. Numer. Methods Eng. 57*, 15, 2221–2240.