A Level Set Method for Ductile Fracture

Jan Hegemann°* Chenfanfu Jiang⋆† Craig Schroeder⋆‡ Joseph M. Teran⋆§

°University of Münster ⋆University of California, Los Angeles

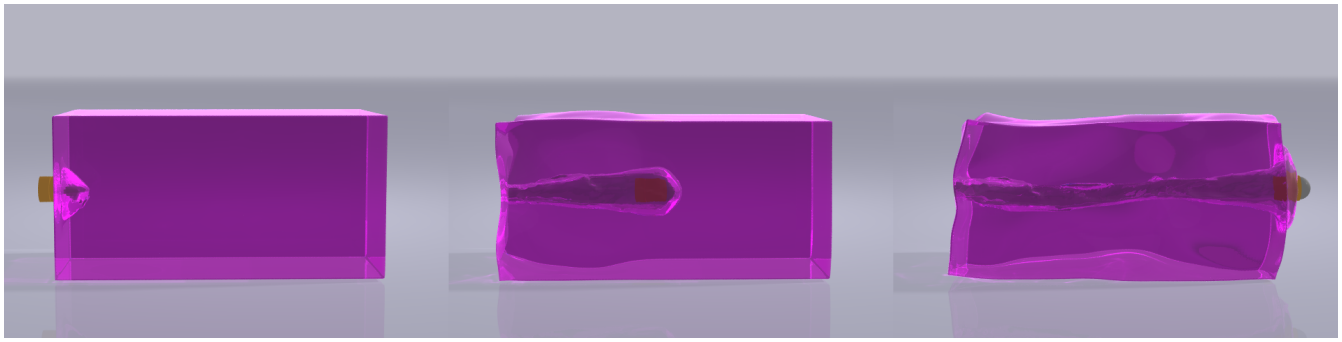


Figure 1: *Shooting a bullet through Jell-O™.*

Abstract

We utilize the shape derivative of the classical Griffith's energy in a level set method for the simulation of dynamic ductile fracture. The level set is defined in the undeformed configuration of the object, and its evolution is designed to represent a transition from undamaged to failed material. No re-meshing is needed since the resulting topological changes are handled naturally by the level set method. We provide a new mechanism for the generation of fragments of material during the progression of the level set in the Griffith's energy minimization. Collisions between different material pieces are resolved with impulses derived from the material point method over a background Eulerian grid. This provides a stable means for colliding with embedded interfaces. Simulation of corotational elasticity is based on an implicit finite element discretization.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations;

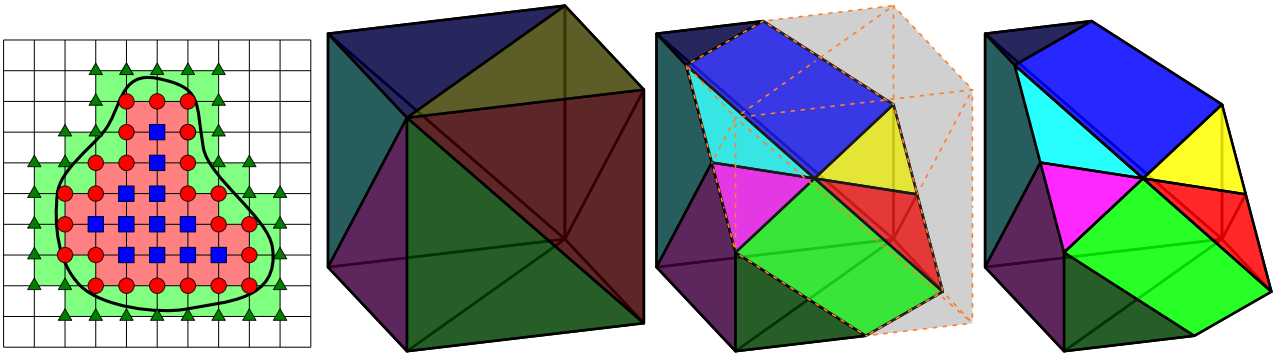
Keywords: ductile fracture, level set method, physically based modeling, collisions, corotational elasticity, plasticity, finite element method

1 Introduction

Our focus is on ductile fracture of elasto-plastic solids. We use a level set method to evolve damaged regions of material with an embedded approach to reduce meshing complexity. Level set methods have proven very effective for handling topological changes for fluids, and we show that they can also be used to reduce remeshing efforts for failure of solids. The level set evolves in material space to minimize Griffith's energy as an alternative to the principle stress criteria popular in computer graphics. This is a generalization of the work in [Allaire et al. 2007] to large-strain, ductile materials. The level set description of the material region is used to simplify the determination of material connectivity in the embedded meshing approach from [Teran et al. 2005; Sifakis et al. 2007] and is similar to the ideas used in [Losasso et al. 2006]. Also, we accurately compute the integrals in the FEM discretization of the elastic forces taking into account sub-cell geometric detail as is commonly done with XFEM discretizations [Belytschko et al. 2009]. We provide a new mechanism for generating fragments of material in damaged regions (as defined by the level set evolution). Finally, we employ a material point method treatment of collision response.

Here we highlight our novel contributions. First, we extend the method of [Allaire et al. 2007] from quasistatic, linear elasticity to dynamics and arbitrary constitutive models. Second, we generalize this work to embedded geometries where the material boundary is initially defined from a level set. Also, we provide a new fragment generation algorithm to prevent volume loss inherent in [Allaire et al. 2007]. This fragment generation procedure is specifically designed for an evolving level set definition of healthy and damaged material. The resulting algorithm is significantly less complex than the explicit remeshing strategies commonly used in computer graphics. Lastly, we demonstrate the application of the material point method (MPM) to the longstanding problem of embedded surface collisions. Collisions with these types of surfaces (e.g. resulting from marching tetrahedra) are notoriously difficult to resolve due to the inherently ill-conditioned sliver triangles arising from isosurface contouring. We also provide an additional improvement to the MPM approach with the introduction of barycentrically bound ghost particles. These are used to improve material coverage of the background MPM grid in large deformation scenarios. Our algorithm is robust and easy to implement; however, this partially comes from representing thin crack structures as having finite thickness, which may prohibit the simulation of some phenomena.

*e-mail:j.hegemann@uni-muenster.de
†e-mail:cffjiang@cs.ucla.edu
‡e-mail:craig@math.ucla.edu
§e-mail:jteran@math.ucla.edu



(a) Embedded domain. (b) Cube divided into tetrahedra. (c) Tetrahedra cut by a level set. (d) Material portion of cut tetrahedra.

Figure 2: *The leftmost image illustrates our level set based embedding in a regular grid in 2D. Boundary cells are shown in green, virtual nodes are depicted with green triangles, nodes that have a discrete stencil independent of embedding are in blue. The images at the right depict tetrahedralization and embedding on a 3D uniform grid (with cells cut into 6 tetrahedra).*

2 Related work

Simulation of fracture and failure phenomena was introduced to computer graphics in the pioneering work of [Terzopoulos and Fleischer 1988]. Early approaches typically made use of simple separation along mesh element boundaries [Norton et al. 1991; Mazarak et al. 1999; Smith et al. 2001; Müller et al. 2001] or even element deletion [Forest et al. 2002]. The available geometric detail in this type of approach was increased somewhat by subdivision of elements in the mesh prior to splitting [Mor and Kanade 2000; Bielser and Gross 2000]; however, this tended to introduce elements with poor aspect ratios. More geometrically rich fracture patterns were generated by allowing failure along more arbitrary paths (albeit with the expense of re-meshing) [Neff and Fiume 1999; O'Brien and Hodgins 1999; O'Brien et al. 2002]. Recently, such approaches have been used to create some very compelling results for a variety of materials [Wicke et al. 2010; Clausen et al. 2013; Wojtan and Turk 2008; Wojtan et al. 2009; Goktekin et al. 2004; Bargteil et al. 2007]. Embedded methods have been developed to minimize the complexity of re-meshing by embedding material surfaces into the existing mesh [Müller and Gross 2004; Molino et al. 2004; Bao et al. 2007; Sifakis et al. 2007; Gissler et al. 2007; Parker and O'Brien 2009]. Although these works generalized the approach to fracture, the embedding idea goes back at least to free form deformations [Sederberg and Parry 1986; Faloutsos et al. 1997; Capell et al. 2002; Teran et al. 2005]. Also, particle-based methods can provide flexibility for topology change [Pauly et al. 2005]. Computer graphics approaches primarily use a principal stress failure criterion [Molino et al. 2004; O'Brien and Hodgins 1999; O'Brien et al. 2002; Müller et al. 2001; Müller and Gross 2004; Kaufmann et al. 2009]. This has also been used for nearly rigid materials where an instantaneous linear elastic response after collision events was used to determine stresses [Bao et al. 2007; Zheng and James 2010; Su et al. 2009]. Grain boundaries in this type of treatment can help to quickly create plausible fracture patterns [Bao et al. 2007; Hellrung et al. 2009; Zheng and James 2010]. Other interesting models for crack patterns were developed in [Iben and O'Brien 2006; Iben and O'Brien 2009; Neff and Fiume 1999].

3 Level set based embedded meshing

As in [Losasso et al. 2006], we use a signed distance function ϕ in material coordinates $\boldsymbol{X}$ to create an embedded Lagrangian mesh for our material (see Figure 2). The mesh consists of all tetrahedral elements in a regular background lattice with at least one node $\boldsymbol{X}_i$ having $\phi(\boldsymbol{X}_i) < 0$. However, our method produces connectivity in this mesh that is slightly different than that of the background lattice. We refer to any node incident on a boundary tetrahedron that has a positive ϕ value as a virtual node since it is outside but still participates in the discretization by virtue of the embedding. We create a linear approximation of the sub-element location of the zero isocontour to define the boundary of the material region. That is, we introduce either a triangle or quadrilateral on each boundary tetrahedron depending on the number of edge crossings. We define a boundary tetrahedron as one having nodes with both positive and negative ϕ values. Note that the vertices of the embedded surface are not degrees of freedom in our discretization. Only the tetrahedron nodes give rise to actual degrees of freedom.

The level set will evolve to minimize Griffith's energy, which we describe in Section 5. During this process, we treat the evolution as a phase change from "healthy" to "damaged" material. As this process occurs, we create a mesh for both damaged and healthy regions. To illustrate this, let ϕ define the healthy region prior to evolution as that where $\phi < 0$, and let $\hat{\phi}$ denote the new level set after evolution. Our energy evolution is defined so that material cannot transition from damaged to healthy. Therefore, the region with $\hat{\phi} < 0$ is contained in the region with $\phi < 0$ (and in fact $\hat{\phi} \geq \phi$). To create the healthy and damaged material meshes, we first create sub-element approximations to the zero isocontour of both ϕ and $\hat{\phi}$ using the previously described process of triangle and quadrilateral insertion on boundary elements (as defined by the respective level sets). Note that both level sets will often cut the same tetrahedron, and therefore some elements may have material surfaces introduced by both level sets. Since our strict evolution from healthy to damaged enforces $\hat{\phi} \geq \phi$, there will never be any crossings between these surfaces.

We combine sub-element geometric information with the level set values to define the meshes in a variation of the material connectivity criteria of [Teran et al. 2005; Sifakis et al. 2007]. First, we create a copy of each tetrahedron that has at least one node with $\hat{\phi} < 0$. That is, this copy has its nodes disconnected from its neighbors with the introduction of potentially temporary virtual nodes. These elements will form the healthy mesh. Second, we create a copy for all tetrahedra that either have (i) a vertex with the product $\hat{\phi}\phi < 0$ (these nodes transitioned from healthy to damaged in the evolution to $\hat{\phi}$) or (ii) an incident edge that is cut by both $\hat{\phi}$ and ϕ (these edges correspond to where the transition occurred at a material node, but without incurring a sign change). These tetrahedra comprise the newly damaged region as defined by the evolution and are added to the system as fragment pieces. Note that some tetrahe-

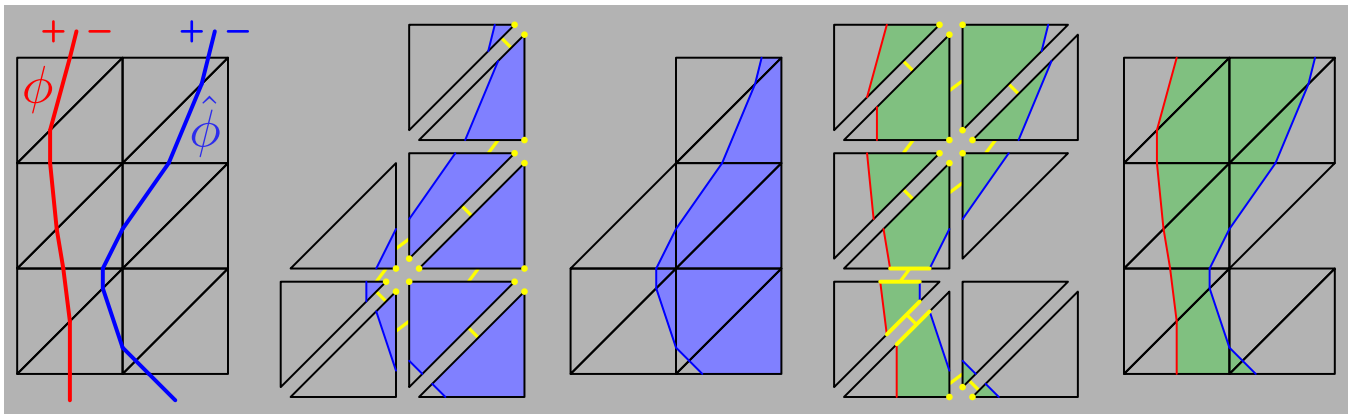


Figure 3: *Elements are duplicated for the healthy (blue) and damaged (green) regions. The degrees of freedom along segments are merged if an endpoint is material on both segments (yellow dots) or if the segment is cut by both level sets for the damaged region (yellow lines).*

dra will give rise to copies for both damaged and healthy regions. This copying procedure is the same as in [Teran et al. 2005; Sifakis et al. 2007]. In the final step, we merge nodes across element faces based on material connectivity. However, our means of establishing this is simplified greatly by our level set and sub-element geometric information. Specifically, faces of adjacent healthy tetrahedral elements (those copied based on the one $\hat{\phi} < 0$ criterion) are merged if at least one original node on the face has $\hat{\phi} < 0$. Faces of adjacent damaged copies are merged if they either share a node that transitioned ($\hat{\phi}\phi < 0$) or if they share an edge that was cut twice. This process is illustrated in Figure 3.

During any time step, the portion of the domain undergoing fracture can be considered as composed of two regions: a healthy region Ω_h^0 and a damaged region Ω_d^0 that is being shed from the healthy region through fracture. The fragments created from the damaged region may be subdivided into smaller pieces via pre-scored grain boundaries as in [Bao et al. 2007], and we do so in the examples seen in Figures 9 and 10; this post-processing does not affect the definition of the damaged region itself. In some cases (e.g. Figures 5, 6 and 8) it may be more visually pleasing to omit the damaged region and forgo fragments altogether. In the latter case, some loss of material occurs as a consequence. In addition to the new fragments (those created in the transition from ϕ to $\hat{\phi}$), there are fragments created in previous time steps which are simulated but are not subject to further fracture.

4 Elasto-plastic dynamics

We use a hyperelastic idealization of the material response to deformation augmented with a simple finite-strain multiplicative plasticity law combined with a Finite Element Method (FEM) discretization. We make only slight modifications to the standard tetrahedral discretization outlined in e.g. [Sifakis and Barbic 2012], which we outline below.

For hyperelastic materials, the first Piola-Kirchoff stress $\boldsymbol{P}$ is related to an energy density Ψ as $\boldsymbol{P} = \frac{\partial \Psi}{\partial \boldsymbol{F}}$, where $\boldsymbol{F}$ is the deformation gradient. In our case, the total elastic energy e is defined in terms of the hyperelastic energy density as

$$e = \int_{\Omega_h^0 \cup \Omega_d^0} \Psi(\boldsymbol{F}(\boldsymbol{X}, t)) d\boldsymbol{X}. \tag{1}$$

Assuming we introduce discontinuities between regions as outlined in Section 3, the FEM elastic force on node j is

$$\boldsymbol{f}_j(\boldsymbol{x}) = -\frac{\partial e}{\partial \boldsymbol{x}_j}(\boldsymbol{x}) = -\int_{\Omega_h^0 \cup \Omega_d^0} \boldsymbol{P} \nabla N_j \, d\boldsymbol{X} = -\sum_k V_k \boldsymbol{P}_k \nabla N_j \tag{2}$$

where $\boldsymbol{x}_j$ is the position of node j, $\boldsymbol{x}$ is the vector of all $\boldsymbol{x}_j$ and N_j is the piecewise linear interpolating function associated with node j. Note that if j belongs to the healthy region then N_j is supported only on Ω_h^0, and if j belongs to the damage region then N_j is supported only on Ω_d^0. The integrands in Equation 2 are piecewise constant because they are functions of the gradients of piecewise linear interpolating functions and therefore we can compute them exactly by computing the volume of the polyhedral material region in each element V_k.

4.1 Time stepping

We use a backward-Euler update of the particle velocities and positions to allow for larger time steps. This is done by solving the non-linear system

$$\boldsymbol{M}\left(\frac{\boldsymbol{v}^{n+1} - \boldsymbol{v}^n}{\Delta t}\right) = \boldsymbol{f}(\boldsymbol{x}^n + \Delta t \boldsymbol{v}^{n+1}) \tag{3}$$

for the time $n+1$ velocities $\boldsymbol{v}^{n+1}$. The linearization of the elastic forces $\boldsymbol{f}$ simply requires the linearization of the first Piola-Kirchoff stress $\boldsymbol{P}$, and we refer the reader to [McAdams et al. 2011] and [Sifakis and Barbic 2012] for the details. We use the FEM mass matrix whose entries are

$$M_{ij} = \int_{\Omega_h^0 \cup \Omega_d^0} N_i N_j d\boldsymbol{X}. \tag{4}$$

Note that $\boldsymbol{M}$ decouples into a healthy part and a damage part and is sparse since the support of any N_i and N_j only overlap if node i and j are incident on the same tetrahedron. We evaluate these integrands analytically over the polyhedral subelement embedded material regions. This is done with the divergence theorem as in [Hellrung et al. 2012].

4.2 Elasto-plastic constitutive model

In practice we define our elasto-plastic constitutive relation from the corotational energy density in [McAdams et al. 2011] as $\tilde{\Psi}(\boldsymbol{F}) = \mu \|\boldsymbol{F} - \boldsymbol{R}\|_F + \frac{\lambda}{2}(\mathrm{tr}(\boldsymbol{S} - \boldsymbol{I}))^2$. Here, $\boldsymbol{F} = \boldsymbol{R}\boldsymbol{S}$ is the polar decomposition of the deformation gradient.

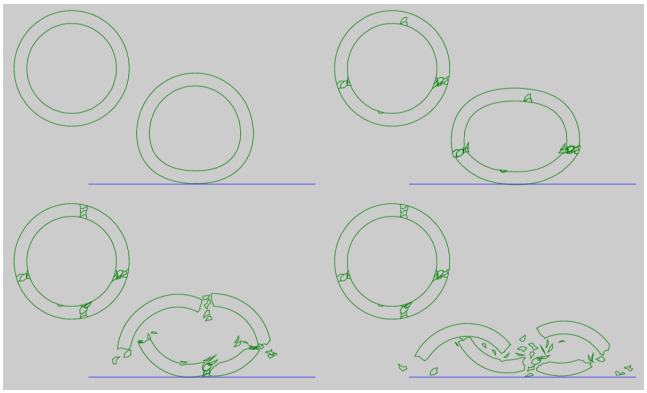


Figure 4: *Ring fractures upon hitting ground with material configuration (left) and the deformed configuration (right) shown.*

After each update of Equation 3 we incorporate the plastic response of [Irving et al. 2004]. This amounts to decomposing the element wise deformation gradient into elastic $\boldsymbol{F}_e$ and plastic $\boldsymbol{F}_p$ parts $\boldsymbol{F}^{n+1} = \boldsymbol{F}_e^{n+1}\boldsymbol{F}_p^{n+1}$. This effects the elastic response via the energy density Ψ as we then consider it to be defined as $\Psi(\boldsymbol{F}) = \tilde{\Psi}(\boldsymbol{F}(\boldsymbol{F}_p^{n+1})^{-1})$ in the next time step.

5 Griffith's energy evolution

We use a Griffith's energy minimization as our fracture evolution criterion as in [Allaire et al. 2007]. Reusing the notation for the healthy region $\Omega_h^0 = \{\boldsymbol{X} | \phi(\boldsymbol{X}) < 0\}$ and the damaged material Ω_d^0 as described in Section 3, the Griffith's energy is defined as

$$e_g(\phi) = \int_{\Omega_h^0} \Psi \, d\boldsymbol{X} + \int_{\Omega_d^0} \kappa \, d\boldsymbol{X}. \tag{5}$$

The coefficient κ is the rate of Griffith's energy release. It can be used to limit the tendency to shrink Ω_h^0 and is therefore intuitively used to increase the material's resistance to fracture. Without this term, we could easily release the elastic energy stored in Ω_h^0 by evolving until we had $\Omega_h^0 = \emptyset$.

A straightforward derivation (see Appendix A for details) shows that the Fréchet derivative of $e_g(\phi)$ in the direction of $\delta\eta$ is given by

$$\int_{\partial\Omega_h^0} (\kappa - \Psi)\delta\eta \, dA. \tag{6}$$

We use this shape derivative in a gradient descent approach as

$$\frac{\partial \phi}{\partial s} = -\delta(\phi)(\kappa - \Psi), \tag{7}$$

where the delta function $\delta(\phi)$ localizes the evolution to the interface and s is a pseudo-time evolution parameter. The transition from ϕ to the new interface, described by $\hat{\phi}$, can then be implemented with a forward-Euler scheme, yielding the update step

$$\hat{\phi} = \phi + \Delta s \delta_\epsilon(\phi)(\Psi - \kappa). \tag{8}$$

Note that a spatially varying function can be used for κ, thus allowing to guide the fracture pattern whenever a more directed evolution is preferred. Figure 5 shows how different choices for κ lead to different propagation speeds.

Since the level set function is defined on the mesh nodes, we need to compute the energy density Ψ on the nodes as well. We do this

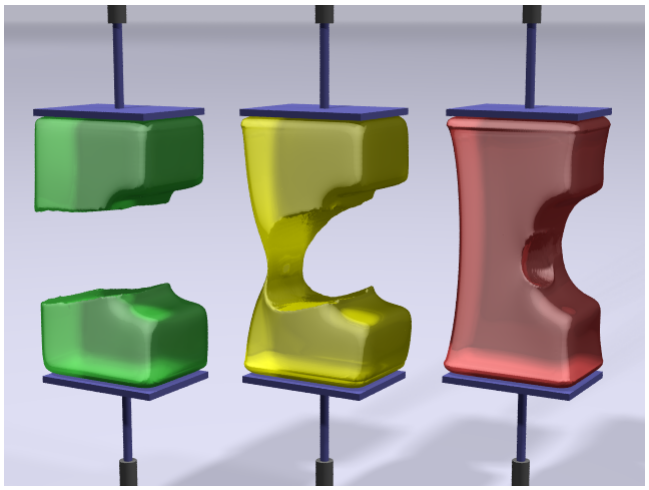


Figure 5: *Stretching Jell-O™ with different energy release rates.*

by computing its value as specified in Section 4.2 within each mesh tetrahedron T_k (where it is piecewise constant), and then employing a weighted average over all elements that contain this node:

$$\Psi(\mathbf{X}_p) = \frac{\sum_{k,\mathbf{X}_p \in T_k} \Psi(T_k) \int_{T_k} N_p(\mathbf{X}) d\mathbf{X}}{\sum_{k,\mathbf{X}_p \in T_k} \int_{T_k} N_p(\mathbf{X}) d\mathbf{X}}, \tag{9}$$

with N_p being the linear basis function of node p.

To ensure that material only transitions from healthy to damaged, we disregard any changes to ϕ at nodes where its value decreases as this would correspond to a growth in the healthy region and thus would violate the irreversibility of fracture. For the delta function, we use the representation of [Vese and Chan 2002], $\delta_\epsilon(\phi) = \frac{1}{\pi}\frac{\epsilon}{\epsilon^2+\phi^2}$. In practice, we enforce a CFL restriction on Δs so that $\hat{\phi}$ does not move the boundary of the healthy region by more than a grid cell in the undeformed configuration. This results in the necessity for multiple executions of (8), though sometimes it might be more visually pleasing to use only a limited number of steps (as we did in the case of Figure 5). An illustration of the energy evolution is given in Figure 6.

To avoid artefacts, we must reinitialize $\hat{\phi}$ during the pseudo-time evolution after each change so that it maintains its signed-distance property. To do so, we first identify the location of the new interface, as defined by $\hat{\phi}$, within the boundary elements of the mesh as detailed in Section 3. We then use the embedded surface triangles and quadrilaterals to recompute the exact distance from the surface to the mesh vertices of the containing boundary elements. This re-evaluation procedure is only necessary at nodes where the level set value, and thus the interface, has changed, i.e. if $|\phi(\boldsymbol{X}_i) - \hat{\phi}(\boldsymbol{X}_i)| > \epsilon$, where ϵ is be chosen as a multiple of machine precision. We then use these exact distances as initialization for a more effective fast sweeping or fast marching reinitialization (e.g. [Zhao 2005]) to propagate the signed distance property to all other mesh nodes.

6 Collisions

We use an impulse-based response for rigid collision bodies and self collision. This is difficult because the embedded meshes that define the material regions have many sliver triangles since they are generated by marching tetrahedra. We found that the material point method for collision impulses outlined in [Huang et al. 2011] was

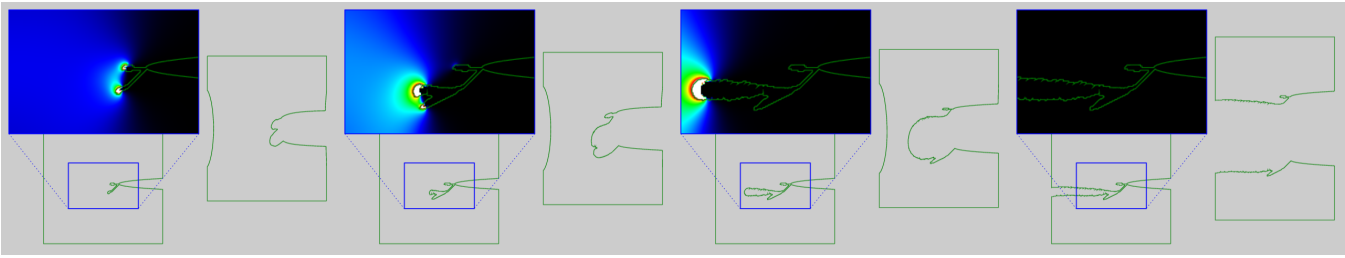


Figure 6: *Notch tears when stretched with energy density (top left), material configuration (bottom left) and deformed configuration (right).*

effective at applying impulses when the surface geometry contained sliver elements. This technique uses the gradients of interpolating functions on a regular background grid to estimate normals to the boundary of the material region. This is convenient because it does not rely on high-quality boundary geometry. However, this robustness does come at the expense of accuracy, and we alleviate this by augmenting the original method by seeding barycentrically bound ghost particles.

We will now outline this collisions handling. For simplicity, we will restrict our description to two colliding bodies, but any number of objects is possible. We compute the connected components of all mesh particles based on material connectivity as given by the mesh. We will use a subscript b to indicate that we store the contributions to a grid node i separately for each body.

Since the accuracy of the collision algorithm that follows is improved by an increase of the number of material particles that contribute to any affected grid node, we create *ghost particles* in every mesh element in addition to the degrees of freedom of the mesh. We use R and G to denote the real and ghost particles, with $P = R \cup G$ (or R_b, G_b, and P_b when referring only to those particles in body b). These dependent particles do not represent any new degrees of freedom but are solely defined by their barycentric relation to their parent mesh vertices, with the barycentric weight $w_p^q = 0$ if q is not bound to p. The positions and candidate velocities of ghost particle are computed in a straightforward manner as the linear combination of their parents. That is,

$$\boldsymbol{x}_q^n = \sum_{p \in R} w_p^q \boldsymbol{x}_p^n \qquad \boldsymbol{v}_q^{n+1} = \sum_{p \in R} w_p^q \boldsymbol{v}_p^{n+1}, \tag{10}$$

where $q \in G$. The computation of the mass for ghost particles has to be done sightly more carefully to conserve total mass. The particles masses m_{pk}^n corresponding to an element T_k and a real particle p are computed respecting the embedded boundaries as the integral

$$m_{pk}^n = \int_{T_k} \rho_0 N_p d\mathbf{X}. \tag{11}$$

The mass associated with a node p can thus change if material gets damaged or breaks off during the fracture evolution. The mass $\sum_k m_{pk}^n$ of real particle p is distributed to p and embedded particles q proportional to their barycentric weights w_p^q (where $w_p^p = 1$) so that

$$m_q^n = \sum_{p \in R} \frac{w_p^q m_{pk}^n}{W_{pk}} \quad m_p^n = \sum_k \frac{m_{pk}^n}{W_{pk}} \quad W_{pk} = 1 + \sum_{q \in G \cap T_k} w_p^q, \tag{12}$$

where $p \in R$, $q \in G$, $q \in T_k$, and W_{pk} was chosen so that

$$\begin{aligned}
\sum_{q \in P} m_q^n &= \sum_{p \in R} m_p^n + \sum_k \sum_{q \in G \cap T_k} m_q^n \\
&= \sum_{p \in R} \sum_k \frac{m_{pk}^n}{W_{pk}} + \sum_k \sum_{q \in G \cap T_k} \sum_{p \in R} \frac{w_p^q m_{pk}^n}{W_{pk}} \\
&= \sum_{p \in R} \sum_k \frac{m_{pk}^n}{W_{pk}} \left(1 + \sum_{q \in G \cap T_k} w_p^q\right) \\
&= \sum_{p \in R} \sum_k m_{pk}^n \\
&= \sum_k \int_{T_k} \rho_0 d\mathbf{X}
\end{aligned}$$

accounts for the total mass.

Next, we rasterize the particle masses to the collision grid as

$$m_{bi}^n = \sum_{p \in P_b} m_p^n S_i(\boldsymbol{x}_p^n), \tag{13}$$

where the S_i are standard trilinear (grid-based) nodal shape functions. We use the positions of the previous time step, $\boldsymbol{x}_p^n$, as the values of the current time step will depend on any changes we make to the velocity field to avoid collisions.

The nodal velocity on the grid is computed as the ratio of rasterized momentum to mass. Using the candidate velocities $\boldsymbol{v}_p^{n+1}$ from the backward Euler update of Equation (3),

$$\bar{\boldsymbol{v}}_{bi}^{n+1} = \frac{\sum_{p \in P_b} m_p^n \boldsymbol{v}_p^{n+1} S_i(\boldsymbol{x}_p^n)}{m_{bi}^n}. \tag{14}$$

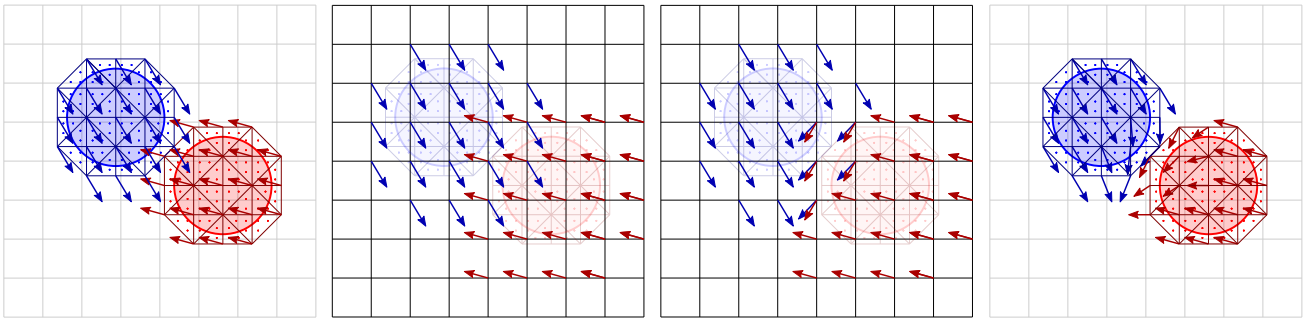
We use the gradient of the nodal mass to find the outward normals of grid node i for body b

$$\boldsymbol{n}_{bi}^n = \frac{\nabla m_{bi}^n}{\|\nabla m_{bi}^n\|} = \frac{\sum_{p \in P_b} m_p^n \nabla S_i(\boldsymbol{x}_p^n)}{\|\sum_{p \in P_b} m_p^n \nabla S_i(\boldsymbol{x}_p^n)\|}. \tag{15}$$

Note that for some grid nodes, the material barely overlaps with the support of the associated shape function, and ghost particles cannot offer any improvements. These nodes also barely contribute to the velocity field due to their low mass weights, so in practice this ghost particle strategy leads to acceptable results.

If particles from two different bodies register at the same grid node, a collision may occur. All contact velocities are relative to the velocity of the center of mass at this grid node

$$\boldsymbol{v}_{(bc),i}^{\text{com},n+1} = \frac{\bar{\boldsymbol{v}}_{bi}^{n+1} m_{bi}^n + \bar{\boldsymbol{v}}_{ci}^{n+1} m_{ci}^n}{m_{bi}^n + m_{ci}^n}. \tag{16}$$



(a) Candidate particles velocities (b) Rasterized grid velocities (c) Corrected grid velocities (d) Corrected particle velocities

Figure 7: *Collisions are processed by transferring velocities to a background grid, where collisions are detected and corrected. These corrected velocities are then transferred back to the simulation mesh.*

To detect and process a collision, we need a single collision normal direction. In practice, however, if bodies b and c are colliding, we will have $\boldsymbol{n}_{bi}^n \neq -\boldsymbol{n}_{ci}^n$. We can correct this by computing an shared direction for the collision

$$\boldsymbol{n}_{(bc)i}^n = \frac{\boldsymbol{n}_{bi}^n - \boldsymbol{n}_{ci}^n}{\|\boldsymbol{n}_{bi}^n - \boldsymbol{n}_{ci}^n\|} \qquad \boldsymbol{n}_{(cb)i}^n = -\boldsymbol{n}_{(bc)i}^n. \tag{17}$$

An approach of the two bodies happens if

$$(\bar{\boldsymbol{v}}_{bi}^{n+1} - \bar{\boldsymbol{v}}_{ci}^{n+1}) \cdot \boldsymbol{n}_{(bc)i}^n > 0. \tag{18}$$

In this case, we project out the normal components of each approaching velocity

$$\boldsymbol{v}_{bi}^{n+1} = \bar{\boldsymbol{v}}_{bi}^{n+1} - [(\bar{\boldsymbol{v}}_{bi}^{n+1} - \boldsymbol{v}_i^{\text{com},n+1}) \cdot \boldsymbol{n}_{(bc)i}^n]\boldsymbol{n}_{(bc)i}^n. \tag{19}$$

This detection based on separate bodies does not cover collisions of different parts of the same piece of material. However, this limitation could be circumvented by subdividing a body into smaller regions that register separately onto the Eulerian grid. These subregions could then in turn collide with each other (neighboring ones excluded to not interfere with elasticity forces). However, for our examples such an extension necessary was not necessary.

After all grid velocities are corrected we interpolate the new velocities back to the degrees of freedom of the Lagrangian mesh as

$$\begin{aligned} \boldsymbol{v}_{p,new}^{n+1} = \xi[\boldsymbol{v}_p^n + \sum_i S_i(\boldsymbol{x}_p^n)(\boldsymbol{v}_{bi}^{n+1} - \boldsymbol{v}_{bi}^n)] \\ + (1-\xi)\sum_i S_i(\boldsymbol{x}_p^n)\boldsymbol{v}_{bi}^{n+1}, \end{aligned} \tag{20}$$

with b such that $p \in b$; $\xi \in [0,1]$ is a control parameter that defines the ratio between PIC (Particle-In-Cell method [Evans et al. 1957]) versus FLIP (Fluid-Implicit-Particle method [Brackbill and Ruppel 1986]). For our simulations, we found that full FLIP, i.e. $\xi = 1$, serves our purposes best. Since typically only a small portion of the Lagrangian particles are involved in collisions, we do not need the numerical viscosity introduced by PIC for stability.

Lastly, we update all particle velocities with the new values and correct the positions accordingly to be consistent with the backward Euler time discretization:

$$\boldsymbol{v}_p^{n+1} \leftarrow \boldsymbol{v}_{p,new}^{n+1} \qquad \boldsymbol{x}_p^{n+1} = \boldsymbol{x}_p^n + \Delta t \boldsymbol{v}_p^{n+1}. \tag{21}$$

The collision algorithm is summarized in Figure 7.

7 Full algorithm

The complete update at each time step reads as follows:

1. **Elasticity update:** solve Equation (3) for candidate velocities
2. **Collisions handling:** rasterize masses and momenta (using current positions and newly-acquired candidate velocities) of mesh vertices and ghost particles; compute grid-based velocities and body normals; detect grid-based collisions and resolve them by projecting out the respective normal components; interpolate velocities back to mesh dofs
3. **Level set evolution:** compute energy density as stated in Section 4.2; move the interface according to Equation (7)
4. **Mesh cutting and fragment generation:** embed the surface of the new healthy region into the mesh; use the old and new interfaces to generate new fragments as detailed in Section 3; reinitialize level set function to a signed distance field.

8 Results

We demonstrate the compelling effects possible with this approach with a number of complex fracture scenarios. Representative runtimes are given in Table 1. Figures 1, 9 and 10 demonstrate our ability to resolve collisions with an external projectile while simultaneously simulating the failure response. Figure 4 depicts the sequential generation of fragments of material during a collision-induced failure process. Figures 8 and 5 demonstrate failure resulting from external loading (Figure 6 illustrates the effect of the elastic energy in this process). In the examples in Figure 9 and 10 we further split damage fragments using pre-scored grain boundaries as in [Bao et al. 2007].

9 Limitations and discussion

Our level set approach to ductile fracture requires no Lagrangian re-meshing effort and is very easy to implement. Furthermore, the

example	dofs	level set grid	min/frame
Stretching Jell-O™	1.9M	128 × 128 × 128	4.1
Shooting Jell-O™	1.0M	128 × 128 × 128	1.1
Stretching armadillo	1.0M	96 × 96 × 96	1.1

Table 1: *Example degree of freedom counts, level set grid resolutions and simulation times. Simulations were performed on a 16-core Intel Xeon E5-2690 2.90GHz machine.*

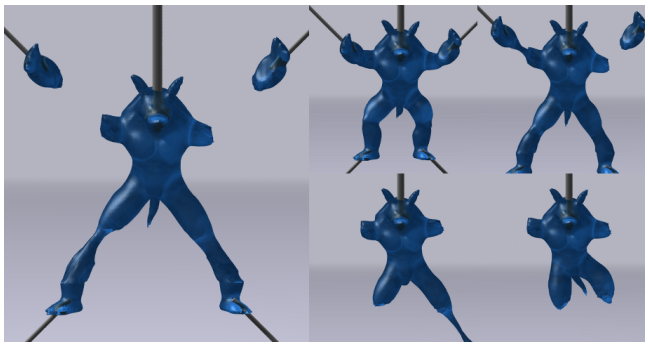


Figure 8: *An armadillo is stretched until its limbs tear off.*

level set evolution in the Griffith's energy minimization requires little more information than is already needed during standard simulation of deformable objects. Although this makes the method easier to implement than many methods that utilize aggressive remeshing, it also brings with it some limitations. The primary limitation is caused by the level set definition of the material regions, which precludes the representation of infinitely thin failure regions, i.e. we cannot represent individual crack tip curves. Our grid based collision handling is not as precise as [Bridson et al. 2002] and can cause small regions of overlap in some areas and separation distances in others. However in contrast to said work, the MPM method provides the capability to resolve collisions between embedded interface, independent of the aspect ratios of the embedded triangles. Nonetheless, the method presents a nice balance between accuracy and complexity of implementation. Its foundation in Griffith's energy and its ability to employ arbitrary fracture patterns lead to compelling, realistic fracture effects, as demonstrated in our results.

Acknowledgements

All authors were partially supported by NSF (DMS-0502315, DMS-0652427, CCF-0830554, DOE (09-LR-04-116741-BERA), ONR (N000140310071, N000141010730, N000141210834), Intel STCVisual Computing Grant (20112360) as well as a gift from Disney Research.

A Derivation of the level set speed

We will now detail the derivation of Equation (6) and our level set speed. The problem is of the general form

$$J(\Omega) := \int_{\Omega} f(\boldsymbol{X}) d\boldsymbol{X},$$

for a smooth, open set $\Omega \subset \mathbb{R}^d$ and sufficiently well-behaved function f. The problem at hand is the analysis of the behavior of J under small perturbations of the domain of integration Ω. For a small, suitable perturbation $(I+\theta)(\Omega) = \{\boldsymbol{X} + \theta(\boldsymbol{X}) | \boldsymbol{X} \in \Omega\}$ the rate of change to J can be expressed as the *shape derivative* (see [Sokolowski and Zolesio 1992]), which we will denote as $J^{'}(\Omega)[\theta]$, where the direction of change, θ, is indicated in brackets.

A very useful way to compute the shape derivative for this case is the following result (cf. e.g. [Sokolowski and Zolesio 1992]). For

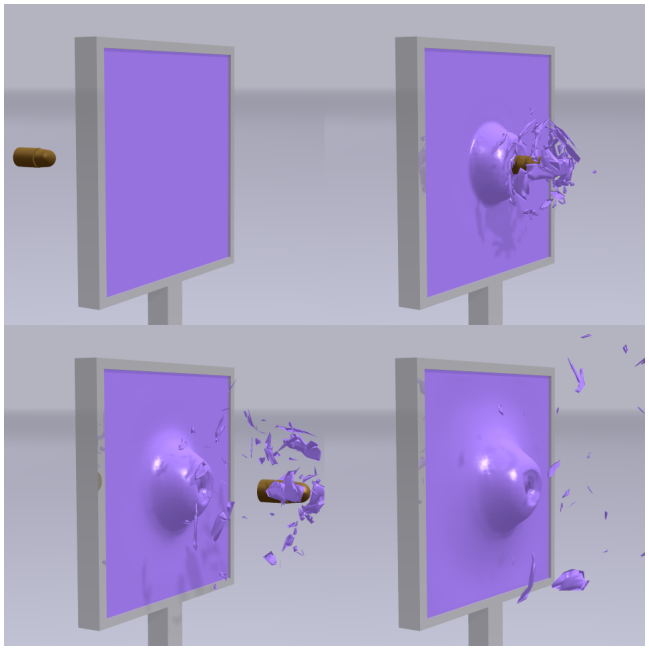


Figure 9: *Shooting a bullet through a plastic wall.*

the above definitions, the shape derivative of J is given by

$$\begin{aligned} J'(\Omega)[\theta] &= \int_{\Omega} \nabla \cdot (\theta(\boldsymbol{X}) f(\boldsymbol{X})) d\boldsymbol{X} \\ &= \int_{\partial\Omega} \theta(\boldsymbol{X}) \cdot \boldsymbol{n}(\boldsymbol{X}) f(\boldsymbol{X}) dA(\boldsymbol{X}), \end{aligned} \tag{22}$$

where $\boldsymbol{n}$ denotes the outward normal to $\partial\Omega$. This result can also be interpreted as a special case of the Reynolds' transport theorem

$$\frac{d}{d\tau} \int_{\Omega(\tau)} f dV = \int_{\Omega(\tau)} \frac{\partial f}{\partial \tau} dV + \int_{\partial\Omega(\tau)} (\boldsymbol{v}^b \cdot \boldsymbol{n}) f dA.$$

To show that the statement of Equation (22) holds, let $\Omega^{'} = (I + \epsilon\theta)(\Omega)$ be a small perturbation of Ω in the direction of θ, and let γ: $\mathbb{R}^d \to \mathbb{R}^d$, $\Omega \mapsto \Omega^{'}$ the mapping between the two sets, i.e. $\gamma(\boldsymbol{X}) = (I + \epsilon\theta)(\boldsymbol{X})$. Then, we can apply a change of coordinates to obtain

$$\begin{aligned} J((I+\epsilon\theta)(\Omega)) &= \int_{\Omega'} f(\boldsymbol{Y}) d\boldsymbol{Y} \\ &= \int_{\Omega} f(\gamma(\boldsymbol{X})) \det(\mathrm{D}\gamma)(\boldsymbol{X}) d\boldsymbol{X}, \end{aligned}$$

where $\mathrm{D}\gamma$ denotes the Jacobian of γ. The directional derivative of J is then given by

$$\begin{aligned} J'(\Omega)[\theta] &= \frac{\partial}{\partial\epsilon}|_{\epsilon=0} J((I+\epsilon\theta)(\Omega)) \\ &= \frac{\partial}{\partial\epsilon}|_{\epsilon=0} \int_{\Omega} f(\gamma(\boldsymbol{X})) \det(\mathrm{D}\gamma)(\boldsymbol{X}) d\boldsymbol{X} \\ &= \int_{\Omega} \nabla f(\boldsymbol{X}) \cdot \theta(\boldsymbol{X}) + f(\boldsymbol{X}) \nabla \cdot \theta d\boldsymbol{X} \\ &= \int_{\Omega} \nabla \cdot (f\theta) d\boldsymbol{X} \end{aligned}$$

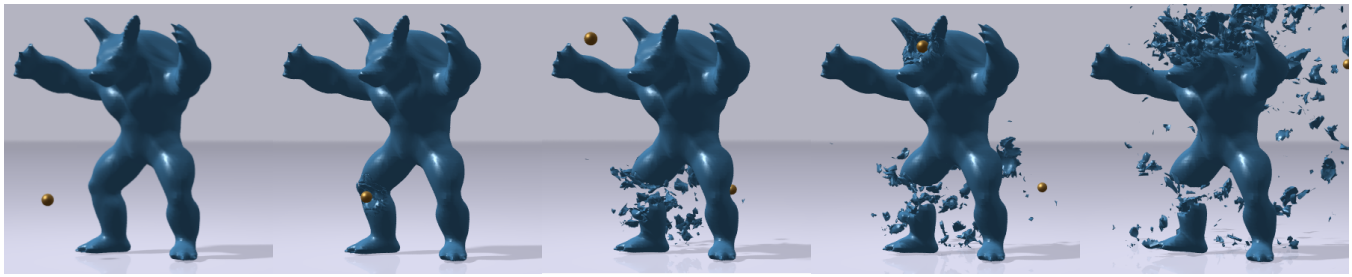


Figure 10: *Shooting two spheres at an armadillo.*

where we used $\det(\mathrm{D}\gamma)|_{\epsilon=0} = \det(\mathrm{D}(I+\epsilon\theta))|_{\epsilon=0} = 1$ as well as

$$\begin{aligned}\frac{\partial}{\partial\epsilon}|_{\epsilon=0}\det(\mathrm{D}\gamma) &= \frac{\partial}{\partial\epsilon}|_{\epsilon=0}\det(\mathrm{D}(I+\epsilon\theta)) \\ &= \frac{\partial}{\partial\epsilon}|_{\epsilon=0}(1+\epsilon\nabla\cdot\theta+O(\epsilon^2)) \\ &= \nabla\cdot\theta,\end{aligned}$$

and $\nabla f\cdot\theta + f\nabla\cdot\theta = \nabla\cdot(f\theta)$ for any differentiable scalar valued function f and vector field θ. The divergence theorem completes the derivation.

To apply this result, we first interpret the energy e_g defined by Equation (5) as a functional in the healthy region Ω_h^0 and then use Equation (22) to differentiate the energy in the direction of a small perturbation θ:

$$e_g'(\Omega_h^0)[\theta] = \int_{\partial\Omega_h^0}\Psi\theta\cdot\boldsymbol{n}_h dA + \int_{\partial(\Omega\setminus\Omega_h^0)}\kappa\theta\cdot\boldsymbol{n}_d dA.$$

Since $\partial(\Omega\setminus\Omega_h^0) = \partial\Omega_h^0$ and $\boldsymbol{n}_h = -\boldsymbol{n}_d$ where $\theta \neq \mathbf{0}$, we can rewrite this as

$$e_g'(\Omega_h^0)[\theta] = \int_{\partial\Omega_h^0}(\Psi-\kappa)\theta\cdot\boldsymbol{n}_h dA.$$

By setting $\theta(x) = -\delta\eta\,\boldsymbol{n}_h(x)$ (in the notation of Section 5), we obtain Equation (6), which allows us to minimize the energy in a gradient descent. This level set speed, $v_{\mathrm{ls}} = \Psi - \kappa$ corresponds to the idea of Griffith that a transition from healthy phase to damaged phase only occurs if the release of elastic energy exceeds a threshold κ.

References

ALLAIRE, G., JOUVE, F., AND GOETHEM, N. V. 2007. A levelset method for the numerical simulation of damage evolution. In *Proc. ICIAM*, 3–22.

BAO, Z., HONG, J., TERAN, J., AND FEDKIW, R. 2007. Fracturing rigid materials. *IEEE Trans. Vis. Comp. Graph. 13*, 370–378.

BARGTEIL, A., WOJTAN, C., HODGINS, J., AND TURK, G. 2007. A finite element method for animating large viscoplastic flow. *ACM Trans. Graph. 26*, 19–38.

BELYTSCHKO, T., GRACIE, R., AND VENTURA, G. 2009. A review of extended/generalized finite element methods for material modeling. *Mod. Sim. Mater. Sci. Eng. 17*, 043001.

BIELSER, D., AND GROSS, M. H. 2000. Interactive simulation of surgical cuts. In *Proc. Pac. Conf. Comp. Graph. App.*, 116–442.

BRACKBILL, J., AND RUPPEL, H. 1986. Flip: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal Comp. Phys. 65*, 314–343.

BRIDSON, R., FEDKIW, R., AND ANDERSON, J. 2002. Robust treatment of collisions, contact and friction for cloth animation. In *ACM Trans. Graph.*, vol. 21, ACM, 594–603.

CAPELL, S., GREEN, S., CURLESS, B., DUCHAMP, T., AND POPOVIĆ, Z. 2002. Interactive skeleton-driven dynamic deformations. *ACM Trans. Graph. 21*, 586–593.

CLAUSEN, P., WICKE, M., SHEWCHUK, J. R., AND O'BRIEN, J. F. 2013. Simulating liquids and solid-liquid interactions with lagrangian meshes. *ACM Trans. Graph. 32*, 17:1–15.

EVANS, M., HARLOW, F., AND BROMBERG, E. 1957. The particle-in-cell method for hydrodynamic calculations. Tech. rep., DTIC Document.

FALOUTSOS, P., VAN DE PANNE, M., AND TERZOPOULOS, D. 1997. Dynamic free-form deformations for animation synthesis. *IEEE Trans. Vis. Comp. Graph. 3*, 201–214.

FOREST, C., DELINGETTE, H., AND AYACHE, N. 2002. Removing tetrahedra from a manifold mesh. In *Proc. Comp. Anim.*, 225–229.

GISSLER, M., BECKER, M., AND TESCHNER, M. 2007. Constraint sets for topology-changing finite element models. In *Virt. Real. Inter. Phys. Sim.*, 21–26.

GOKTEKIN, T., BARGTEIL, A., AND O'BRIEN, J. 2004. A method for animating viscoelastic fluids. *ACM Trans. Graph. 23*, 463–468.

HELLRUNG, J., SELLE, A., SHEK, A., SIFAKIS, E., AND TERAN, J. 2009. Geometric fracture modeling in Bolt. In *SIGGRAPH 2009: Talks*, 7:1.

HELLRUNG, J. L., WANG, L., SIFAKIS, E., AND TERAN, J. M. 2012. A second order virtual node method for elliptic problems with interfaces and irregular domains in three dimensions. *Journal Comp. Phys. 231*, 2015 – 2048.

HUANG, P., ZHANG, X., MA, S., AND HUANG, X. 2011. Contact algorithms for the material point method in impact and penetration simulation. *Int. Journal Num. Meth. Eng. 85*, 498–517.

IBEN, H. N., AND O'BRIEN, J. F. 2006. Generating surface crack patterns. In *Proc. Symp. Comp. Anim.*, 177–185.

IBEN, H., AND O'BRIEN, J. 2009. Generating surface crack patterns. *Graph. Mod. 71*, 198–208.

IRVING, G., TERAN, J., AND FEDKIW, R. 2004. Invertible finite elements for robust simulation of large deformation. In *Proc. Symp. Comp. Anim.*, 131–140.

KAUFMANN, P., MARTIN, S., BOTSCH, M., GRINSPUN, E., AND GROSS, M. 2009. Enrichment textures for detailed cutting of shells. *ACM Trans. Graph. 28*, 50:1–50:10.

LOSASSO, F., IRVING, G., GUENDELMAN, E., AND FEDKIW, R. 2006. Melting and burning solids into liquids and gases. *IEEE Trans. Vis. Comp. Graph. 12*, 343–352.

MAZARAK, O., MARTINS, C., AND AMANATIDES, J. 1999. Animating exploding objects. In *Proc. Graph. Int.*, 211–218.

MCADAMS, A., ZHU, Y., SELLE, A., EMPEY, M., TAMSTORF, R., TERAN, J., AND SIFAKIS, E. 2011. Efficient elasticity for character skinning with contact and collisions. *ACM Trans. Graph. 30*.

MOLINO, N., BAO, Z., AND FEDKIW, R. 2004. A virtual node algorithm for changing mesh topology during simulation. In *ACM SIGGRAPH*, 385–392.

MOR, A. B., AND KANADE, T. 2000. Modifying soft tissue models: Progressive cutting with minimal new element creation. In *Proc. MICCAI*, 598–607.

MÜLLER, M., AND GROSS, M. 2004. Interactive virtual materials. In *Proc. Graph. Int.*, 239–246.

MÜLLER, M., MCMILLAN, L., DORSEY, J., AND JAGNOW, R. 2001. Real-time simulation of deformation and fracture of stiff materials. In *Proc. Eurographics Workshop Comp. Anim. Sim.*, 113–124.

NEFF, M., AND FIUME, E. 1999. A visual model for blast waves and fracture. In *Proc. Graph. Int.*, 193–202.

NORTON, A., TURK, G., BACON, B., GERTH, J., AND SWEENEY, P. 1991. Animation of fracture by physical modeling. *Vis. Comp. 7*, 210–219.

O'BRIEN, J., AND HODGINS, J. 1999. Graphical modeling and animation of brittle fracture. In *Proc. annual Conf. Comp. Graph. interactive Tech.*, 137–146.

O'BRIEN, J., BARGTEIL, A., AND HODGINS, J. 2002. Graphical modeling and animation of ductile fracture. *ACM Trans. Graph. 21*, 291–294.

PARKER, E., AND O'BRIEN, J. 2009. Real-time deformation and fracture in a game environment. In *Proc. Symp. Comp. Anim.*, 165–175.

PAULY, M., KEISER, R., ADAMS, B., DUTRÉ, P., GROSS, M., AND GUIBAS, L. 2005. Meshless animation of fracturing solids. *ACM Trans. Graph. 24*, 957–964.

SEDERBERG, T., AND PARRY, S. 1986. Free-form deformation of solid geometric models. In *ACM SIGGRAPH*, 151–160.

SIFAKIS, E., AND BARBIC, J. 2012. Fem simulation of 3d deformable solids: a practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH Courses*, 20:1–20:50.

SIFAKIS, E., DER, K., AND FEDKIW, R. 2007. Arbitrary cutting of deformable tetrahedralized objects. In *Proc. Symp. Comp. Anim.*, 73–80.

SMITH, J., WITKIN, A., AND BARAFF, D. 2001. Fast and controllable simulation of the shattering of brittle objects. *Comp. Graph. Forum 20*, 81–90.

SOKOLOWSKI, J., AND ZOLESIO, J. 1992. *Introduction to shape optimization: shape sensitivity analysis*.

SU, J., SCHROEDER, C., AND FEDKIW, R. 2009. Energy stability and fracture for frame rate rigid body simulations. In *Proc. Symp. Comp. Anim.*, 155–164.

TERAN, J., SIFAKIS, E., BLEMKER, S., NG-THOW-HING, V., LAU, C., AND FEDKIW, R. 2005. Creating and simulating skeletal muscle from the visible human data set. *IEEE Trans. Vis. Comp. Graph. 11*, 317–328.

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

VESE, L., AND CHAN, T. 2002. A multiphase level set framework for image segmentation using the mumford and shah model. *Inter. Journal Comp. Vis. 50*, 271–293.

WICKE, M., RITCHIE, D., KLINGNER, B., BURKE, S., SHEWCHUK, J., AND O'BRIEN, J. 2010. Dynamic local remeshing for elastoplastic simulation. *ACM Trans. Graph. 29*, 49:1–49:11.

WOJTAN, C., AND TURK, G. 2008. Fast viscoelastic behavior with thin features. In *ACM SIGGRAPH*, 47:1–47:8.

WOJTAN, C., THÜREY, N., GROSS, M., AND TURK, G. 2009. Deforming meshes that split and merge. *ACM Trans. Graph. 28*, 76:1–76:10.

ZHAO, H. 2005. A fast sweeping method for eikonal equations. *Math. Comp. 74*, 603–628.

ZHENG, C., AND JAMES, D. 2010. Rigid-body fracture sound with precomputed soundbanks. *ACM Trans. Graph. 29*, 69:1–69:13.