Simulation and Visualization of Ductile Fracture with the Material Point Method

STEPHANIE WANG, University of California – Los Angeles
MENGYUAN DING, University of California – Los Angeles
THEODORE F. GAST, JIXIE EFFECTS
LEYI ZHU, University of Science and Technology of China
STEVEN GAGNIERE, University of California – Los Angeles
CHENFANFU JIANG, University of Pennsylvania
JOSEPH M. TERAN, University of California – Los Angeles

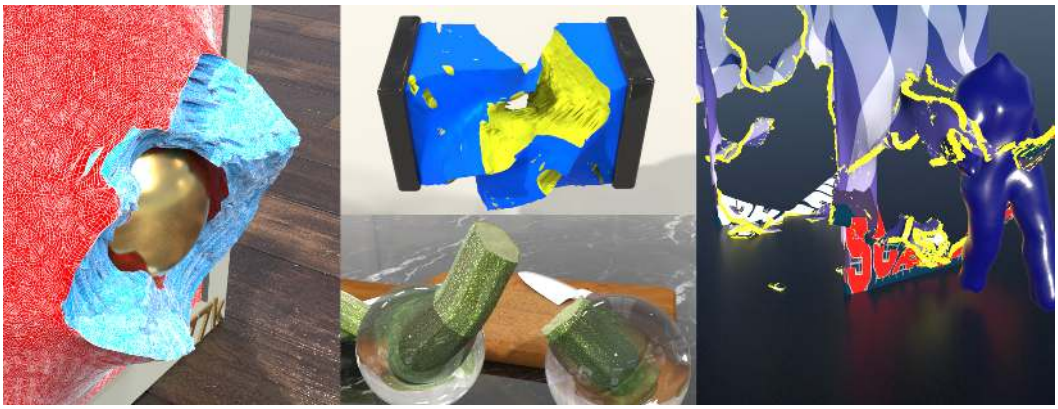

Fig. 1. Left: Meshing an elastic wall shot by a projectile. Bottom: Breaking a zucchini with brute force. Top: Twisting a cube until it breaks. Right: Ductile walls fracture as a mannequin walks through.

We present novel techniques for simulating and visualizing ductile fracture with the Material Point Method (MPM). We utilize traditional particle-based MPM [Stomakhin et al. 2013; Sulsky et al. 1994] as well as the Lagrangian energy formulation of [Jiang et al. 2015] that utilizes a tetrahedron mesh, rather than particle-based estimation of the deformation gradient and potential energy. We model failure and fracture via elastoplasticity with damage. Material is elastic until its deformation exceeds a Rankine or von Mises yield condition, at which point we use a softening model that shrinks the yield surface until a damage threshold is reached. Once damaged, the material Lamé coefficients are modified to represent failed material. We design visualization techniques for rendering the boundary of the material and its intersections with evolving crack surfaces.

Authors' addresses: Stephanie Wang, University of California – Los Angeles, evast@g.ucla.edu; Mengyuan Ding, University of California – Los Angeles, mengyuanding@ucla.edu; Theodore F. Gast, JIXIE EFFECTS, tfg@math.ucla.edu; Leyi Zhu, University of Science and Technology of China, zhuleyi@mail.ustc.edu.cn; Steven Gagniere, University of California – Los Angeles, sgagniere@math.ucla.edu; Chenfanfu Jiang, University of Pennsylvania, cffjiang@seas.upenn.edu; Joseph M. Teran, University of California – Los Angeles, jteran@math.ucla.edu.

2577-6193/2019/7-ART18 $15.00
https://doi.org/10.1145/3340259

Our approach uses a simple and efficient element splitting strategy for tetrahedron meshes to represent crack surfaces that utilizes an extrapolation technique based on the MPM simulation. For traditional particle-based MPM we use an initial Delaunay tetrahedralization to connect randomly initialized MPM particles. Our visualization technique is a post-process and can be run after the MPM simulation for efficiency. We demonstrate our method with a number of challenging simulations of ductile failure with considerable and persistent self-contact.

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

Additional Key Words and Phrases: MPM, Elasticity, Fracture, Plasticity, Ductile

ACM Reference Format:
Stephanie Wang, Mengyuan Ding, Theodore F. Gast, Leyi Zhu, Steven Gagniere, Chenfanfu Jiang, and Joseph M. Teran. 2019. Simulation and Visualization of Ductile Fracture with the Material Point Method. *Proc. ACM Comput. Graph. Interact. Tech.* 2, 2, Article 18 (July 2019), 20 pages. https://doi.org/10.1145/3340259

1 INTRODUCTION

Ductile materials behave elastically until a yield stress condition is met, at which point they yield plastically and at some point fail completely. Whether it be the distinctive patterns exhibited while tearing a piece of fruit or twisted metal after a high-velocity impact, the fracture and failure of ductile materials are ubiquitous and indispensable when creating visually interesting virtual worlds for computer graphics applications. Indeed, some of the earliest methods for simulating elasticity in computer graphics included treatment for tearing and failure of materials [Terzopoulos and Fleischer 1988]. O'Brien et al. [2002] demonstrated that using the Finite Element Method (FEM) with continual domain remeshing after fracture events allowed for a wide range of ductile behaviors and incredibly detailed simulations. Since this pioneering approach, many others have used FEM and remeshing to achieve similar behaviors [Molino et al. 2004; Müller and Gross 2004; Wicke et al. 2010; Wojtan et al. 2009]. Particle methods based on Smoothed Particle Hydrodynamics (SPH) [Chen et al. 2013; Gerszewski et al. 2009] and Moving Least Squares (MLS) [Müller et al. 2004; Pauly et al. 2005] have also been used with impressive effect, since their unstructured nature naturally allows for topological change. Procedural approaches have also achieved good results when computational cost is limited [Choi 2014; Jones et al. 2016; Müller et al. 2007].

The Material Point Method (MPM) is another unstructured particle technique that naturally resolves topological changes and fracture, and also naturally accommodates elastoplastic phenomena. Furthermore, a key advantage of MPM is that the hybrid Lagrangian/Eulerian nature of the method naturally resolves collisions between fragments of material. These aspects make MPM an ideal candidate for simulating fracture and failure of ductile materials. However, while MPM naturally allows for topological changes, they can be difficult to control. Particles are connected in the domain when they are in the support of the same Eulerian grid node interpolating function. Particles that do not interact with the same grid nodes in this way are decoupled. This is advantageous in that topology change requires no special treatment; however, fracture is therefore a numerical error that is not influenced by a material property but rather by discretization-related parameters like particle sampling density and Eulerian grid resolution.

Numerical fracture can be addressed by utilizing particle resampling techniques as in [Yue et al. 2015] or by using the Lagrangian energy technique of Jiang et al. [2015] in which a tetrahedron mesh is used to compute deformation gradients. This treatment naturally couples meshed objects with MPM-based materials, and also gives an automated treatment of self-collision between meshed objects and other materials. However, in either the resampling or Lagrangian energy approaches, an additional model must be provided to allow for fracture.

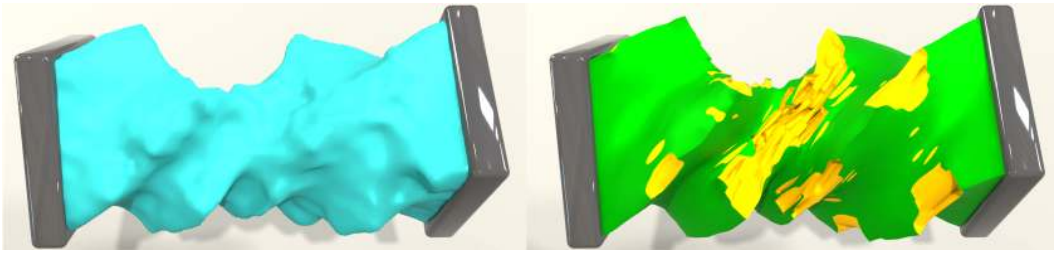

Fig. 2. A twisted cube with 8,000 particles was surfaced with Houdini particle fluid surface (left) and our mesh visualization (right).

A second issue hindering MPM adoption for ductile fracture is largely common to all particle-based techniques: defining and rendering material boundary surfaces in a visually sharp manner is difficult. While particle-based simulation techniques naturally allow for topological change, they generally have a more vague notion of material boundaries that complicates the process of rendering. FEM and mesh-based techniques require more intervention (remeshing) to resolve topological change, however in the process material boundaries are sharp and well defined. This is important for preserving the surface of objects created by users, and for transferring textures as the material fails.

The most common techniques for visualizing particle-based simulation data define the boundary of the particle domain as the zero isocontour of a level set function, or as a threshold value of a density function. This goes back to at least Blinn [1982]. Many other authors have provided improvements on these techniques over the years, including sharper surface resolution, reduction of noise and temporal coherence of surfaces, resolution of anisotropic features, and many more [Adams et al. 2007; Ando et al. 2013; Müller et al. 2003; Museth 2014; Museth et al. 2007; Solenthaler et al. 2007; Yu and Turk 2013; Zhu and Bridson 2005]. However, these types of techniques are much more appropriate for fluid simulations, and cannot support initialization from a high-resolution textured input surface mesh without complicated texture transfer at each frame, etc.

Surface tracking techniques can provide the desired preservation of sharp features and surface details. These techniques have been used with great effect in simulations of fluid [Brochu and Bridson 2009; Da et al. 2014; Müller 2009; Wojtan et al. 2010; Yu et al. 2012] and viscoelastic materials [Dagenais et al. 2017; Wojtan et al. 2009]. These approaches are extremely powerful, but computationally expensive. However, much of the implementation and computational overhead is associated with material merging. Much simpler techniques can be used if only splitting is required. Fracture of ductile materials typically only involves failure without cohesive merging, so fully-general surface tracking techniques are not necessary.

Pre-scoring-based surfacing approaches are generally more efficient than surface tracking, and can be used when merging is not needed. These techniques predefine the maximally split configuration of the material, and only separation between components can occur. For example, the virtual node algorithm of Molino et al. [2004] is a pre-scoring technique where each vertex in a tetrahedron mesh represents a portion of the material in the elements in its one ring. Choi [2014] use a pre-scoring approach for visualizing shape-matching-based ductile fracture where each node is assigned material as a union of elements, gathered via K-means, from a tetrahedron mesh. Chen et al. [2018] assign a single tetrahedron to each particle by initializing particles at the barycenters of an input tetrahedron mesh. In these techniques, material separation is introduced when connectivity between adjacent particle regions is severed. Crack surfaces are then defined as a subset of the

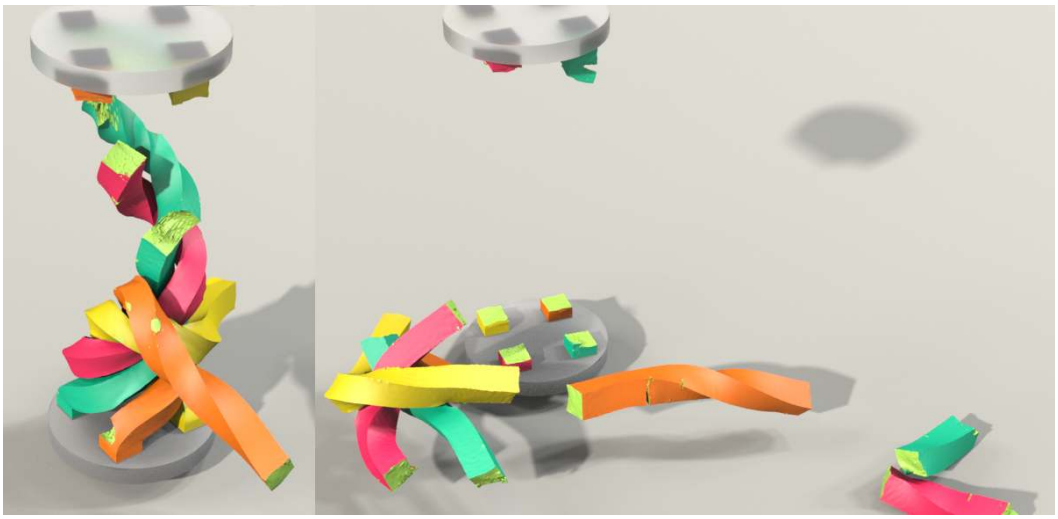

Fig. 3. Four columns braided to fracture.

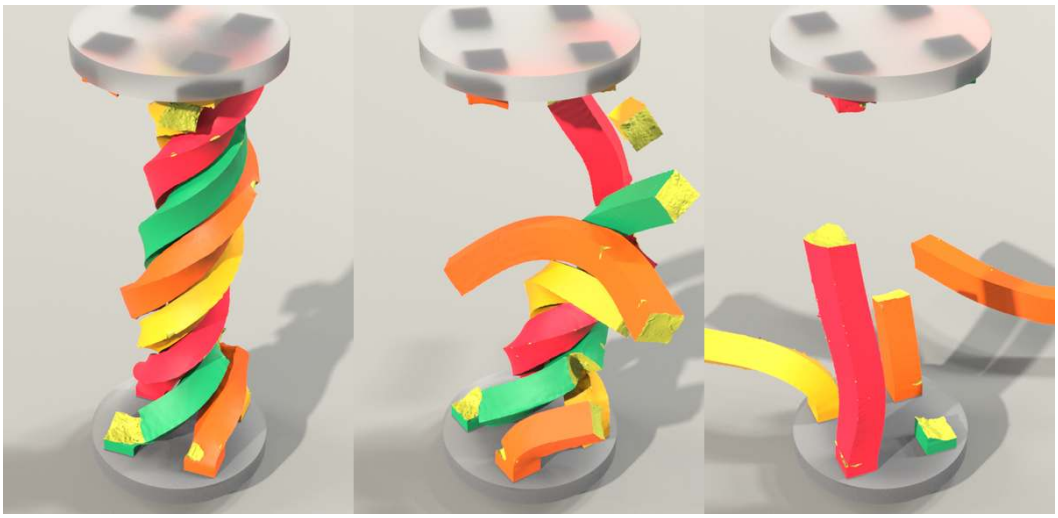

Fig. 4. Four stiffer columns braided to fracture.

boundary of the maximally split configuration. Generally, pre-scoring techniques suffer from mesh-based aliasing, since the crack paths must lie on the predefined maximally split configuration. Fracture surfaces are usually much smoother than they will appear when the sampling bias in the predefined maximally split configuration is imposed on the visualization.

We provide two options to remove the barriers preventing MPM adoption for ductile fracture simulation in graphics applications. First, we provide an extension of the mesh based strategy of Jiang et al. [2015] that removes numerical fracture and introduces failure through the elastoplastic constitutive equations alone. Second, when traditional particle-based MPM with numerical fracture suffices, we overcome limitations of existing surfacing strategies with a pre-scoring approach. We

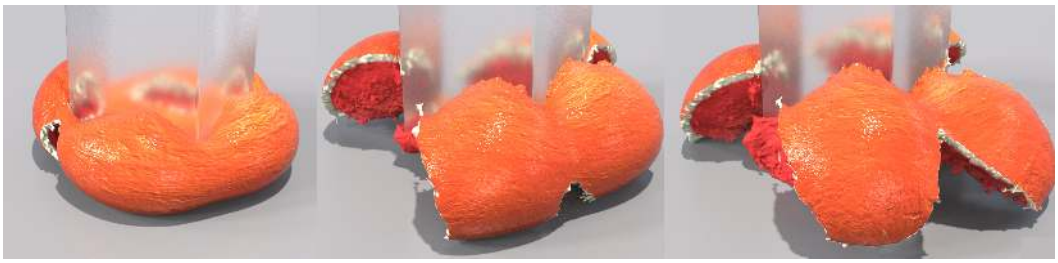

Fig. 5. "Hydraulic press". The orange is simulated with a meshed hollow sphere filled with guts made by MPM particles.

note that our surfacing approach is a post-process that can be implemented on data generated from standard MPM simulations. In summary, our contributions include:

- An elastoplasticity and damage model for ductile fracture that works easily with existing MPM code bases.
- A generalization of the Lagrangian energy approach of Jiang et al. [2015] for removing numerical fracture with ductile materials.
- A novel particle surfacing technique that preserves input surface details like texture and high-curvature regions, while removing mesh-based aliasing inherent in pre-scoring surfacing strategies.

2 PREVIOUS WORK

Here we discuss works from the computer graphics and computational physics literature related to simulation of ductile fracture and visualization of particle-based simulation data.

Following the seminal approach of O'Brien et al. [2002], many authors have used FEM simulation of elastoplasticity with continual domain remeshing for ductile fracture. Müller et al. [2004] use warped stiffness with a Rankine condition on the principal stress to define per-tetrahedron element fracture planes. Pfaff et al. [2014] use an adaptive mesh to simulate tearing and cracking of thin sheets. Parker and O'Brien [2009] use the separation tensor from [O'Brien and Hodgins 1999] but split along element boundaries rather than cutting elements for the sake of efficiency. Wicke et al. [2010] dynamically remesh tetrahedron meshes to allow for efficient simulation of behaviors ranging from purely elastic to extremely plastic with fracture. Other remeshing approaches include [Bargteil et al. 2007; Busaryev et al. 2013; Wojtan et al. 2009; Wojtan and Turk 2008]. Wicke et al. [2008; 2007] developed interpolating functions for convex polyhedral elements to allow for easy splitting of elements in fracture simulations. Gissler et al. [2007] introduce a notion of constraint sets for fracture simulation. Koschier et al. [2017] use XFEM and improve the mass matrix treatment by integrating over partially empty enriched elements. Zhang et al. [2006] use tetrahedron mesh-based FEM with elastoplasticity driven damage, element splitting (at damage threshold), and molecular dynamics for debris simulation.

Pauly et al. [2005] use a meshfree MLS approach to simulate elastoplastic ductile fracture with Heaviside-enriched interpolating functions, as in the XFEM approaches of Belytschko [2003]. They create domain and crack boundary surfaces at render time using the surfels approach in [Pauly et al. 2003; Wicke et al. 2004]. Müller et al. [2004] use a similar approach. Steineman et al. [2009] use visibility graphs to further improve the modification of MLS interpolating functions in the presence of splitting and merging defined by explicitly tracked failure surfaces. Gerszewski et al. [2009] also compute the deformation gradient in a weighted least squares sense.

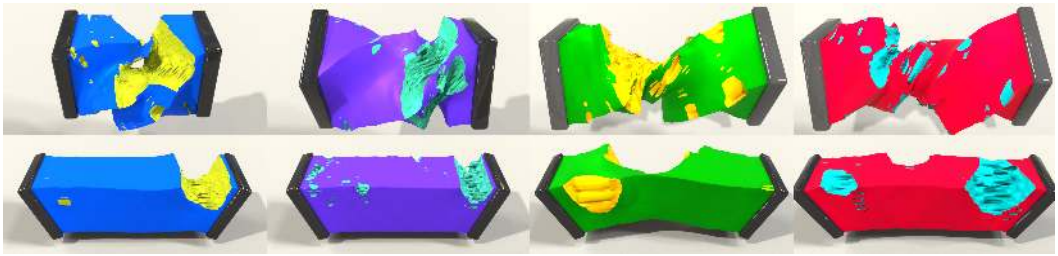

Fig. 6. Four identical cubes of different resolution undergoing twisting and pulling motions. From left to right: 60K, 17K, 8K, 4K particles.

Other notable ductile fracture techniques include the peridynamics approach of Chen et al. [2018]. Bußler et al. [2017] visualize crack surfaces in peridynamics particle data by computing Delaunay tetrahedralizations that respect height ridges in the damage field. Choi [2014] uses shape-matching to simulate procedural ductile fracture. Ohta et al. [2009] use an adaptive regular lattice with shape matching-based elasticity to simulate ductile fracture. Jones et al. [2016] simulate ductile fracture using shape matching.

Various approaches for ductile fracture with MPM exist in the computational physics literature. Wretborn et al. [2017] simulate fracture with MPM by pre-scoring materials into pieces held together by massless particle constraints. They resolve collisions between fragments by using the MPM N-body approach of [2011]. Nairn et al. [2006; 2003] developed the CRAMP MPM technique for simulating velocity and displacement discontinuities on the grid. Other MPM techniques utilize grid node duplication [2007]. They then resolve frictional contact on the duplicated Eulerian grid nodes.

Surfacing particle-based simulation data is a long-standing problem. Most approaches define the boundary of the particle domain as the zero isocontour of a level set function or as a threshold value of a density function [Adams et al. 2007; Blinn 1982; Desbrun and Cani 1998; Müller et al. 2003; Museth 2014; Solenthaler et al. 2007; Zhu and Bridson 2005]. Yu and Turk developed an anisotropic approach to more accurately capture sharp features [2013]. Bhattacharya et al. [2015] fit signed distance functions to particle data by minimizing a biharmonic thin shell energy over a surface constrained between interior and exterior CSG surfaces, and support anisotropic capture of sharp features as in [Yu and Turk 2013]. Williams [2008] similarly solves the surfacing problem with a constrained minimization. Shen and Shah [2007] address temporal discontinuities by blending adjacent frames. Museth et al. [2007] incorporate a variety of post-processing techniques including temporal and spatial anti-aliasing. Adams et al. [2007] use a semi-Lagrangian contouring method similar to that proposed by Bargteil et al. [2006]. Dagenais et al. [2017] improves and extends surface tracking to retain surface details. Mercier et al. [2015] develop a post-process approach for surfacing particle-based fluid simulation data. They create an up-res particle surface using a generalization of the approach in [Williams 2008] and then apply a surface-only Lagrangian wave simulation to provide realistic, detailed motion.

Pre-scoring bodies into precomputed pieces is useful for simulation and visualization. Müller et al. [2013] decompose objects into convex pieces and generate fracture patterns of space using Voronoi diagrams. CSG operations are used to resolve the initial convex decomposition with the fracture patterns. Su et al. [2009] also fracture all of space to generate rigid body fragment pieces for real time simulation of brittle fracture. Liu et al. [2011] also pre-score the material along Voronoi boundaries to add user control over fracture patterns. Schvartzman and Otaduy [2014] use Voronoi-based pre-scoring of fracture boundaries with rigid body simulation to simulate brittle fracture. Zheng

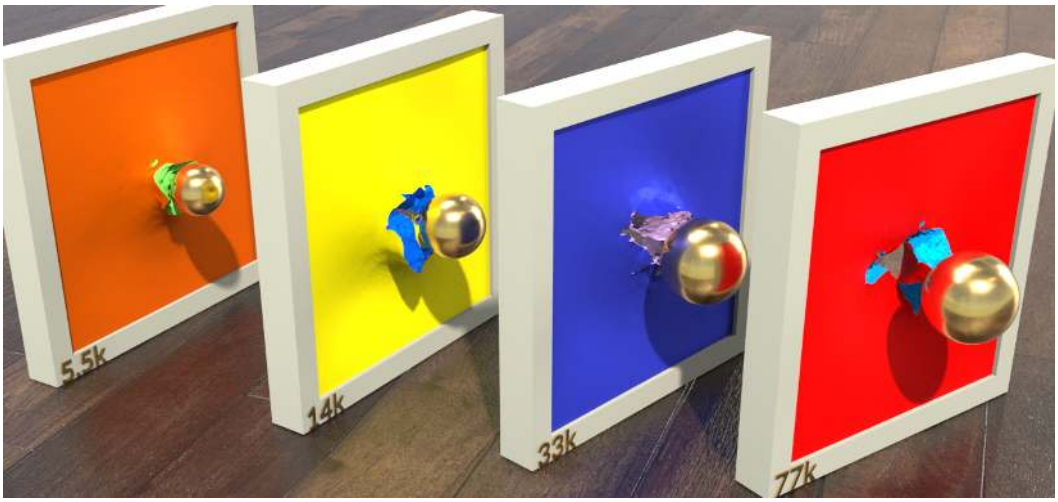

Fig. 7. Shooting projectiles at ductile walls with 5.5K (orange), 14K (yellow), 33K (blue), and 77K (red) particles.

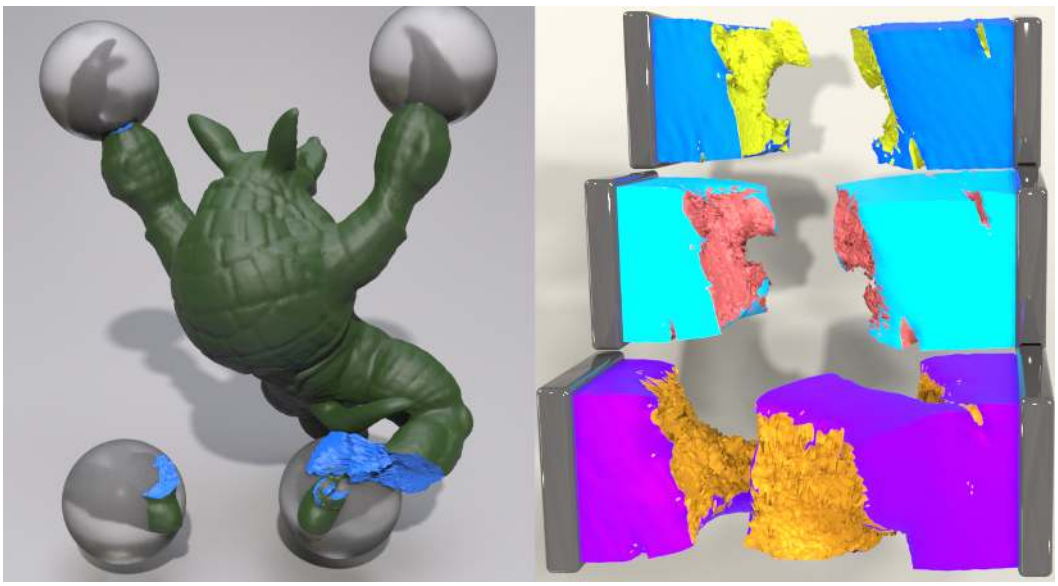

Fig. 8. Left: an armadillo twisted to fracture. Right: twisting cubes with different von Mises yield surfaces. We use $\tau_C = E$, (blue), $0.7E$ (cyan), and $0.5E$, for Young's modulus E.

and James [2010] use the strain energy density to adapt Voronoi fracture regions. Raghavachary [2002] defines fragments in polygon meshes by splitting into Voronoi regions.

3 MATHEMATICAL BACKGROUND

We define the deformation of a continuum body as a map from its undeformed configuration consisting of points $\mathbf{X}$ to its deformed configuration consisting of points $\mathbf{x}$ at time t by $\mathbf{x}(t) = \phi(\mathbf{X}, t)$. We refer to the spatial derivative of this map as the deformation gradient $\mathbf{F} = \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{X}}$ and decompose

it into elastic and plastic parts $\mathbf{F} = \mathbf{F}^E\mathbf{F}^P$. Here $\mathbf{F}^E$ is the elastic deformation and $\mathbf{F}^P$ is the plastic deformation associated with inelastic yielding at large stresses [Bonet and Wood 2008]. The potential energy in the system increases as $\mathbf{F}^E$ deviates from orthogonality, meaning that the motion from the plastic/damaged state is non-rigid. The governing equations for the deformation mapping are derived from conservation of mass and momentum

$$\frac{D\rho}{Dt} = -\rho\nabla\cdot\mathbf{v}, \tag{1}$$

$$\rho\frac{D\mathbf{v}}{Dt} = \nabla\cdot\boldsymbol{\sigma} + \rho\mathbf{g}, \tag{2}$$

where $\boldsymbol{\sigma}$ denotes the Cauchy stress, $\mathbf{g}$ the gravity, and $\frac{D}{Dt} = \frac{\partial}{\partial t} + \mathbf{v}\cdot\nabla$ the material derivative.

3.1 Elastic constitutive model

We use the isotropic hyperelastic potential energy density of [Klár et al. 2016]. This model is quadratic in elastic Hencky strain $\boldsymbol{\epsilon}^E = \frac{1}{2}\ln(\mathbf{F}_E\mathbf{F}_E^T)$,

$$\psi(\mathbf{F}^E) = \mu\boldsymbol{\epsilon} : \boldsymbol{\epsilon} + \frac{\lambda}{2}\mathrm{tr}(\boldsymbol{\epsilon})^2 = \mu\sum_{i=1}^{3}\ln(\sigma_i^E)^2 + \frac{\lambda}{2}\left(\sum_{i=1}^{3}\ln(\sigma_i^E)\right)^2 \tag{3}$$

where $\mathbf{F}^E = \mathbf{U}^E\Sigma^E(\mathbf{V}^E)^T$ is the singular value decomposition of $\mathbf{F}_E$ and σ_i^E denote the entries in Σ^E. Here μ and λ are the Lamé coefficients which control the amount of resistance to deformation and volume change. The Cauchy stress is defined in terms of the elastic potential as

$$\boldsymbol{\sigma} = \frac{1}{\det(\mathbf{F})}\frac{\partial\psi}{\partial\mathbf{F}_E}{\mathbf{F}_E}^T, \tag{4}$$

$$\frac{\partial\psi}{\partial\mathbf{F}^E} = \mathbf{U}^E\Sigma^{E^{-1}}\left(2\mu\ln(\Sigma^E) + \lambda\ln(\Sigma)\right)(\mathbf{V}^E)^T. \tag{5}$$

This choice of potential energy is primarily for the sake of simplifying the return mapping process (see [Wang et al. 2019]), as discussed in [Jiang et al. 2017; Klár et al. 2016].

3.2 Plasticity

Ductile materials behave elastically until a critical stress is reached, at which point deformation becomes permanent and the material achieves a new local rest state. We express this notion of critical stress in terms of a yield surface in stress space defined implicitly as $y(\boldsymbol{\sigma}) = 0$ using a yield function y. When $y(\boldsymbol{\sigma}) < 0$, the critical stress has not been achieved and the material behaves elastically. When $y(\boldsymbol{\sigma}) = 0$, the elastic limit is reached and the plastic deformation defined via $\mathbf{F}^P$ becomes non-trivial. Mathematically, we can view the dynamics of $\mathbf{F}^P$ as being chosen to satisfy the stress constraint $y(\boldsymbol{\sigma}) = 0$ through its dependence on $\mathbf{F}^E$.

Although the Cauchy stress $\boldsymbol{\sigma}$ is more physically intuitive, the Kirchhoff stress $\boldsymbol{\tau} = \det(\mathbf{F})\boldsymbol{\sigma}$ is often more convenient when working with plasticity. It is particularly convenient for defining the plastic deformation in a manner that is consistent with the second law of thermodynamics and when enforcing the yield condition discretely during time stepping, a process which is typically referred to as the return mapping (see [Wang et al. 2019]). Henceforth, we will assume the yield surface is defined in terms of the Kirchhoff stress $y(\boldsymbol{\tau})$.

3.2.1 Yield surface. We use two different yield surfaces to model different fracture modes. The Rankine yield surface [Anderson 2017] is given by

$$y(\boldsymbol{\tau}) = \max_{\|\mathbf{u}\|=\|\mathbf{v}\|=1}\mathbf{u}^T\boldsymbol{\tau}\mathbf{v} - \tau_C \le 0, \tag{6}$$

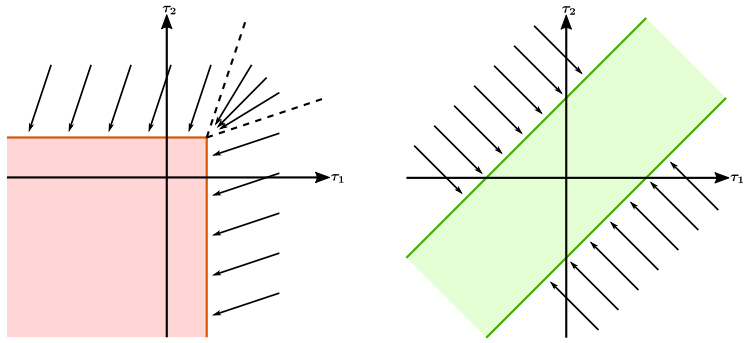

Fig. 9. Left: Rankine yield surface and its return mapping. Right: von Mises yield surface and its return mapping.

where τ_C is a scalar parameter that represents the maximum allowed tensile strength, since the expression $\max_{\|\mathbf{u}\|=\|\mathbf{v}\|=1} \mathbf{u}^T \boldsymbol{\tau} \mathbf{v}$ measures the tensile stress among all directions and corresponds to the largest eigenvalue of $\boldsymbol{\tau}$. Constraining the maximal tension in all directions enables the material to go through mode I yielding, where permanent deformation is induced in response to local tension.

The von Mises yield surface given by

$$y(\boldsymbol{\tau}) = \|\boldsymbol{\tau} - \mathrm{tr}(\boldsymbol{\tau})\mathbf{I}\|_F - \tau_C \leq 0 \tag{7}$$

provides plastic response to mode II and mode III shearing deformations by constraining the deviatoric (shear) stress; here $\|\mathbf{A}\|_F = \sqrt{\mathbf{A} : \mathbf{A}}$ denotes the Frobenius norm. By combining the two yield surfaces or using them independently, we can simulate a wide range of fracturing and plastic materials.

In practice, the yield condition $y(\boldsymbol{\tau}) \leq 0$ is enforced per time step. In this process, the trial strain ($\tilde{\boldsymbol{\epsilon}}^E$) is mapped from a state whose corresponding stress violates the condition to one whose corresponding stress is on the boundary of the yield surface ($\boldsymbol{\epsilon}^{E,n+1}$) in a process referred to as the return mapping. We illustrate the different yield surfaces and the associative direction for return mappings in Figure 9. We provide detailed derivation in [Wang et al. 2019].

3.2.2 Softening and damage. As the material undergoes plastic deformation, we decrease τ_C to shrink the yield surface towards the origin. This limits the strength of the material as smaller and smaller stresses are admissible. For each projection $\tilde{\boldsymbol{\epsilon}}^E \rightarrow \boldsymbol{\epsilon}^{E,n+1}$ in the return mapping (see [Wang et al. 2019]), we decrease τ_C by $\theta\|\boldsymbol{\epsilon} - \mathrm{proj}(\boldsymbol{\epsilon})\|_F$, where $\theta > 0$ is a material constant that defines the rate of softening. When τ_C reaches zero, we model the material as completely damaged and set the Lamé coefficients to zero.

4 NUMERICAL METHOD

We use MPM to discretize the governing equations and cover both standard particle-based MPM as in [Stomakhin et al. 2013; Sulsky et al. 1994] as well as the mesh-based Lagrangian energy techniques used to prevent numerical fracture [Jiang et al. 2015]. In the Lagrangian energy case, we modify the approach of Jiang et al. [2015] to include the effects of plasticity and damage.

In MPM, the discrete state consists of a collection of particles that partition the domain based on initial volumes V_p^0, with time t^n positions $\mathbf{x}_p^n$ and with masses m_p computed from the initial mass density as $\rho(\mathbf{x}_p^0, t^0)V_p^0$ and linear and affine time velocities $\mathbf{v}_p^n$, $\mathbf{C}_p^n$ used for APIC particle/grid transfers [Jiang et al. 2015]. In the case of traditional particle-based MPM, each particle additionally

stores the elastic portion of the deformation gradient $\mathbf{F}_p^{E,n}$ and yield surface size τ_{Cp}. In the case of mesh-based MPM, we assume there additionally exists a tetrahedron mesh connecting the particles $\mathbf{x}_p^n$. We use e to denote elements in the mesh and store $\mathbf{F}_e^{E,n}$ and τ_{Ce} per tetrahedron element, rather than per particle. Furthermore, in the mesh-based case, we must also store the plastic part of the deformation gradient $\mathbf{F}_e^{P,n}$.

An MPM time step from time t^n to t^{n+1} typically consists of three steps: (1) mass (m_p) and momentum ($m_p\mathbf{v}_p^n$) are transferred from particles to the grid using weights ($w_{\mathbf{i}p}^n = N(\mathbf{x}_p^n - \mathbf{x}_\mathbf{i})$) defined by Eularian grid interpolating functions $N(\mathbf{x})$ that describe the degree of interaction between particle p and grid node $\mathbf{i}$, (2) the grid momentum ($m_\mathbf{i}^n\mathbf{v}_\mathbf{i}^n$) is then updated in a variational way from the potential energy in the system, and finally (3) the motion of the grid under the updated momentum is interpolated to the particles. In step (2), the discretization is done differently in the cases of standard particle-based MPM versus the mesh-based approach. The difference lies in how the deformation gradient is computed. In the case of standard particle-based MPM, the deformation gradient is stored per particle and is updated using an updated Lagrangian view. With this assumption the deformation gradient is computed as the product of the time t^n deformation gradient $\mathbf{F}_p^n$ and the deformation of the grid (evaluated at the particle) over the time step $\hat{\mathbf{F}}_p^{n+1} = (\mathbf{I} + \Delta t \sum_\mathbf{i} \mathbf{v}_\mathbf{i}^{n+1}\nabla w_{\mathbf{i}p}^n)$ where $\nabla w_{\mathbf{i}p}^n = \frac{\partial N}{\partial \mathbf{x}}(\mathbf{x}_p^n - \mathbf{x}_\mathbf{i})$ is the derivative of the grid interpolating functions. In the case of mesh-based elasticity, the deformation gradient is computed using mesh connectivity as in standard FEM [Jiang et al. 2015; Sifakis and Barbic 2012] $\mathbf{F}_e^{n+1} = \sum_p \mathbf{x}_p^{n+1}\nabla\tilde{N}_p(\mathbf{X}_e)$ where $\tilde{N}_p(\mathbf{X})$ is the piecewise linear interpolating function associated with particle p evaluated at the tetrahedron barycenter in the initial configuration of the mesh. We summarize this below as

$$m_\mathbf{i}^n = \sum_p w_{\mathbf{i}p}^n m_p \tag{8}$$

$$\mathbf{v}_\mathbf{i}^n = \frac{1}{m_\mathbf{i}^n}\sum_p w_{\mathbf{i}p}^n m_p(\mathbf{v}_p^n + \mathbf{C}_p^n(\mathbf{x}_\mathbf{i} - \mathbf{x}_p^n)) \tag{9}$$

$$\mathbf{v}_\mathbf{i}^{n+1} = \mathbf{v}_\mathbf{i}^n + \frac{dt}{m_\mathbf{i}^n}\mathbf{f}_\mathbf{i} + \Delta t\mathbf{g} \tag{10}$$

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t\sum_\mathbf{i} \mathbf{v}_\mathbf{i}^{n+1} w_{\mathbf{i}p}^n \tag{11}$$

$$\mathbf{v}_p^{n+1} = \sum_\mathbf{i} \mathbf{v}_\mathbf{i}^{n+1} w_{\mathbf{i}p}^n \tag{12}$$

$$\tilde{\mathbf{C}}_p^{n+1} = \frac{12}{\Delta x^2(b+1)}\sum_\mathbf{i} w_{\mathbf{i}p}^n \mathbf{v}_\mathbf{i}^{n+1} \otimes (\mathbf{x}_\mathbf{i} - \mathbf{x}_p^n) \tag{13}$$

$$\mathbf{C}_p^{n+1} = (1-\nu)\,\tilde{\mathbf{C}}_p^{n+1} + \frac{\nu}{2}\left(\tilde{\mathbf{C}}_p^{n+1} - \tilde{\mathbf{C}}_p^{n+1T}\right) \tag{14}$$

$$\tilde{\mathbf{F}}_e^E = \left(\sum_p \mathbf{x}_p^{n+1}\nabla\tilde{N}_p(\mathbf{X}_e)\right)(\mathbf{F}_e^{P,n})^{-1} \tag{15}$$

$$\tilde{\mathbf{F}}_p^E = (\mathbf{I} + \Delta t\sum_\mathbf{i} \mathbf{v}_\mathbf{i}^{n+1}\nabla w_{\mathbf{i}p}^n)\mathbf{F}_p^{E,n} \tag{16}$$

$$\mathbf{F}_q^{E,n+1} = \text{returnMap}(\tilde{\mathbf{F}}_q^E). \tag{17}$$

Here the transfer to grid in step (1) consists of Equations (8)-(9), the grid-based momentum update in step (2) consists of Equations (10)-(12) and the interpolation from grid to particles in step (3) consists

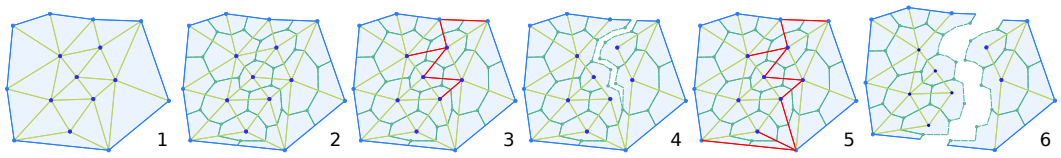

Fig. 10. **Mesh cutting**. From left to right: 1: Initial simplex mesh (Delaunay or quality mesh generated for Lagrangian simulation). 2: Particle core partitioning. 3: Identify failed edges (marked red). 4: The corresponding partially split mesh to the set of failed edges in 3. 5: A different set of failed edges (marked red). 6: The corresponding split mesh to the set of failed edges in 5.

of Equations (12)-(14). This is using APIC transfers [Jiang et al. 2015] for Equations (9) and (13) as well as the RPIC damping of [Jiang et al. 2017] in Equation (14) where ν controls the amount of damping. Note that in Equation (10), $\alpha = 0$ corresponds to symplectic Euler for the grid momentum update and $\alpha = 1$ corresponds to backward Euler. Equations (15) and (16) represent the deformation gradient update in the cases of mesh-based and standard MPM respectively. Equation (17) projects the elastic state to satisfy the plasticity constraints. The equation is indexed by q to indicate that it is either e for mesh-based or p for particle-based MPM.

In Equation (10), $\mathbf{f_i}$ is the force on grid node $\mathbf{i}$ which is computed as the variation of the total potential with respect to grid nodes moving as $\mathbf{x_i} + \Delta t \mathbf{v}_\mathbf{i}^{n+\alpha}$, where $\alpha = 0$ corresponds to symplectic Euler and $\alpha = 1$ corresponds to backward Euler time stepping. The value varies based on the choice of mesh- or particle-based MPM as

$$\mathbf{f_i} = \begin{cases} \sum_p w_{\mathbf{i}p}^n \mathbf{f}_p(\mathbf{x}^{n+\alpha}) + \Delta t \mathbf{g}, \\ -\sum_p \frac{\partial \psi}{\partial \mathbf{F}^E}(\tilde{\mathbf{F}}_p^E(\tilde{x}^{n+\alpha}))(\mathbf{F}_p^{E,n})^T \nabla w_{\mathbf{i}p}^n V_P^0 + \Delta t \mathbf{g} \end{cases} \tag{18}$$

respectively, where $\mathbf{x}^{n+\alpha} \in \mathbb{R}^{2n_P}$ is the vector consisting of all particle time $t^{n+\alpha}$ positions $\mathbf{x}_p^{n+\alpha}$ according to Equation (11). In the case of standard particle MPM, $\tilde{\mathbf{x}}^{n+\alpha}$ is the vector of all Eulerian grid node positions, moved according to

$$\mathbf{x}_\mathbf{i}^{n+\alpha} = \begin{cases} \mathbf{x_i}, & \alpha = 0 \\ \mathbf{x_i} + \Delta t \mathbf{v}_\mathbf{i}^{n+1}, & \alpha = 1 \end{cases} \tag{19}$$

In the case of mesh-based MPM, the particle force $\mathbf{f}_p$ in Equation (18) is related to the variation of the potential as estimated over the tetrahedron mesh, rather than the particles

$$\mathbf{f}_p = \sum_e \frac{\partial \psi}{\partial \mathbf{F}^E}(\tilde{\mathbf{F}}_e^E(\mathbf{x}^{n+\alpha}))\nabla \tilde{N}(\mathbf{X}_e) \tag{20}$$

where $\tilde{\mathbf{F}}_e^E(\mathbf{x}^{n+\alpha})$ is given by Equation (16).

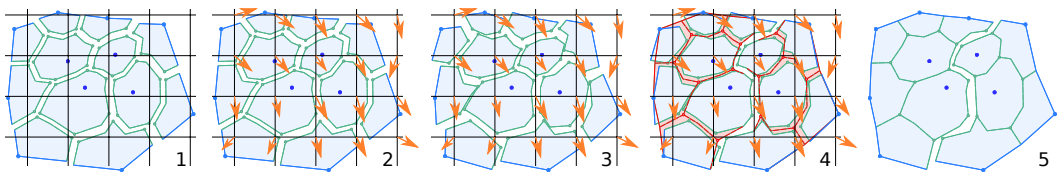

Fig. 11. **Extrapolation**. From left to right: 1. Initial particle core partition. 2. Velocity field defined on grid. 3. Particle cores positioned and oriented by local rigid body transform. 4. Sewing connected cells. 5. Final deformed fractured mesh.

5 MATERIAL SURFACE DEFINITION AND VISUALIZATION

We provide a novel pre-scoring strategy for visualization of material boundary and crack surfaces as a post-process for ductile fracture simulations. Our approach can easily be used for most standalone MPM solvers. Our technique works with either traditional particle-based MPM, or Lagrangian energy mesh-based MPM [Jiang et al. 2015]. In the case of mesh-based MPM, we assume the user provides a tetrahedron mesh of quality suitable for FEM simulation of elasticity. In the case of traditional particle-based MPM, we assume the user provides interior points that are sampled with a Poisson disc, or similar initial random spacing. We also assume that the user provides a triangulation of the boundary of the domain from which the internal particles are sampled. The vertices of the boundary (triangle) mesh and the randomly sampled interior particles are treated as MPM particles for simulation. If the user does not provide a triangle mesh, we can generate one by surfacing the interior particles using an existing technique like [Yu and Turk 2013]. We assume that most users will define the boundary of the initial domain for ductile materials using a triangle mesh, typically with texture etc. and our approach is designed to preserve those details throughout the simulation. Once in possession of the boundary triangle mesh and the interior particles, we create a Delaunay tetrahedralization connecting the interior and boundary points and preserving triangles on the original boundary.

5.1 Visualization mesh topology

With our initialization strategy, in either the traditional particle-based MPM or Lagrangian energy mesh-based MPM cases, we can assume we have a tetrahedralization of the particles used in the MPM calculation. The mesh is used to define a particle-wise partition of the material domain. Each tetrahedron in the mesh is split into four cuboids, one for each of its particles. To create the particle-wise partitioning, each particle in the MPM calculation receives a cuboid from each of the tetrahedron elements it belongs to. We note that this is essentially the same as the per-particle cores of material used in the virtual node approach of Molino et al. [2004]. We adopt this name and refer to the particle's union of cuboids as its core of the domain. With this convention, each particle is responsible for updating its core over the course of the simulation.

The boundary of each particle core initially shares faces with cores of particles that it is connected to in the tetrahedron mesh. We define material failure on a per-initial-tetrahedron-mesh-edge basis. That is, common faces on cores of material associated with particles initially connected in the tetrahedron mesh are treated as identical until material failure occurs. To define material failure, we label core faces between particles connected along an edge in the tetrahedron mesh as broken. We use a simple union-find data structure to manage the topological connectivity and create a hexahedron mesh that respects the failed core faces. To do this we start with a mesh that is completely broken into the maximally split configuration and merge unbroken faces using the union-find data structure. See Figure 10 for details. One could use an element wise splitting strategy where core faces within a damaged element are broken, but we found that this gave inferior results to this edge-wise criterion.

We manage all topological aspects of the material and crack surface visualization with this simple strategy. Next we discuss our criteria for deciding when an edge (and its associated core faces) are broken as well as the geometric aspects of the crack surface evolution.

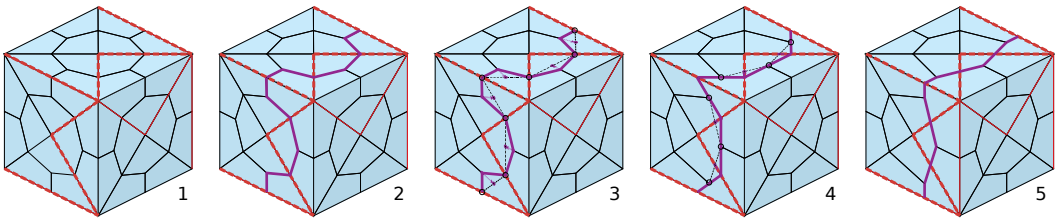

Fig. 12. **Crack boundary curve smoothing**. From left to right: 1: Identify broken edges (red dashed line). 2: Identify boundary curve of the crack surface (purple solid line). 3-4: Smooth crack boundary curve while remaining on the original boundary surface: triangle centers move to average of neighbors, edge centers move to the intersection of its associated edge and the path joined by its neighbors. 5: Crack boundary curve after one iteration of smoothing.

5.2 Topology evolution

We use a history-based maximal stretching criteria to define broken edges. We define the maximum relative stretching of an edge for times before a given time t as

$$\zeta_t = \max_{s<t} \frac{\|\phi(\mathbf{X}_1, t) - \phi(\mathbf{X}_2, t)\|}{\|\mathbf{X}_1 - \mathbf{X}_2\|}. \tag{21}$$

When this value is larger than a threshold, we consider the cores associated with $\mathbf{X}_1$ and $\mathbf{X}_2$ as separated from each other and break the edge connecting them. Note that if any edge is broken at a given time $\hat{t}$ it will be broken for all times $t > \hat{t}$.

5.3 Visualization mesh geometry: extrapolation

Each particle is responsible for updating the geometry of its core. We do this with a simple extrapolation strategy. We use a rigid transform local to each particle to extrapolate the motion of the particle to the rest of its core. For each core vertex $\mathbf{y}_p^n$ associated with a particle center $\mathbf{x}_p^n$, we compute the time t^n position as

$$\mathbf{y}_p^n = \mathbf{R}_p^n(\mathbf{y}_p^0 - \mathbf{x}_p^0) + \mathbf{x}_p^n, \tag{22}$$

where $\mathbf{R}_p^n$ is the rotation associated with the simulated particle p at time t^n. We use the MPM grid velocity to update the local rotation matrix on each particle

$$\mathbf{Z}_p^{n+1} = \left(\mathbf{I} + \sum_i \tilde{\mathbf{v}}_i^n \nabla \omega_{ip}^n\right) \mathbf{R}_p^n, \tag{23}$$

$$\mathbf{R}_p^{n+1}\mathbf{S}_p^{n+1} = \mathbf{Z}_p^{n+1}. \tag{24}$$

where the polar decomposition $(\mathbf{R}_p^{n+1})^T\mathbf{R}_p^{n+1} = \mathbf{I}$, $\mathbf{S}_p^{n+1} = (\mathbf{S}_p^{n+1})^T$ is used to enforce orthogonality. This creates a rigid core translating and rotating with the particle. However, when the vertices on the boundary of the core are associated with multiple cores, we take the average of the extrapolated positions given by each core. This introduces visually realistic deformation when material is not fully failed, while reverting to translation and rotation in the event of a fully separated core.

The accuracy of the update in Equation (23) is affected by the particle sampling density. If the grid resolution is too high relative to the particle density, the update can be noisy. For traditional particle-based MPM this is not an issue, however for Lagrangian energy MPM we found it advantageous to add traditional MPM particles in each element to help resolve update in Equation (23). These particles are not used to compute forces until their parent elements fail. In the event of failure, they function as standard elastic MPM particles. See Figure 13 on the right for details.

5.4 Visualization mesh geometry: crack smoothing

There is considerable flexibility when defining the initial geometry of each particle core. The geometry of the cuboid is most naturally chosen by setting its vertices as the edge, face and tetrahedron centers. However, these points may be chosen anywhere in their respective submanifolds. The only points on the cuboids without flexibility are those corresponding to MPM particles (tetrahedron mesh vertices). We take advantage of this flexibility to remove sampling based biasing in the crack paths. Note that the flexibility is only in the initial geometry of the cuboids. Once set, they must always evolve according to the per-particle extrapolation in Section §5.3.

A limitation of our pre-scoring visualization approach is that all possible crack paths are determined from the initial particle partitioning of the domain. This will lead to sampling bias of the crack surface in general. This tends to make the crack surfaces appear more jaggy in the case of randomly sampled initial points. In the case of structured initial points, the structure is imposed on the crack paths. In order to remove initial sampling bias, we iteratively smooth the crack surface in the initial configuration. Smoothing the surface tends to remove sampling bias as is usually visible through regions of locally high curvature. Because our visualization technique is a post-process, we can assume that we know the topology of the crack surface at the final time from the condition in Section §5.2. We can therefore smooth the entire surface in the initial configuration, as required.

The first step of our approach smoothes the intersection of the initial material boundary surface and the crack surface. Care must be taken in this step to ensure that the boundary crack curves

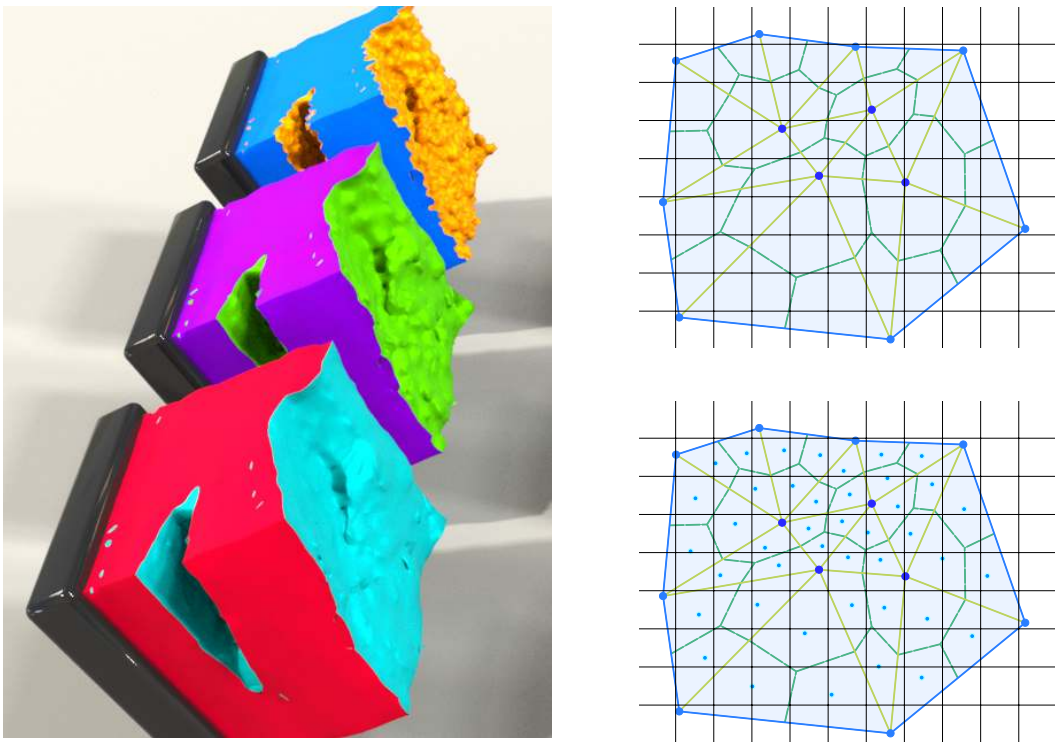

Fig. 13. Left: original crack surface (yellow), crack surface smoothed with 2 iterations (green), crack surface smoothed with 20 iterations (cyan). Right: we sample extra particles in each quadrilateral/cuboid to help reduce noise.

remain on the initial boundary during the smoothing process. See Figure 12 for details. Next, we smooth the crack surface interior by assigning each vertex to the average of its neighbors while the curve processed in the first step remains unchanged. We do this in a Gauss-Seidel fashion. Our approach quickly removes high-frequency noise while preserving the general shape of the crack pattern.

6 RESULTS

We demonstrate our ductile fracture simulation and surface visualization techniques with a variety of simulations exhibiting a wide range of representative behaviors. We list our computational performance and simulation details in Table 1. We note that in many of our examples, remarkably detailed fracture patterns are produced with comparatively low resolutions. This is advantageous because surfacing limitations often require simulations with artificially high resolution in many MPM applications. Our results were run on an Intel Xeon E5-2687W v4 with 48 threads. Time stepping was adaptively chosen according to the CFL condition, i.e. Δt was set so no particle travels more than a portion of a grid cell in each time step. For particle-based MPM, the grid resolution was chosen so that there are initially approximately six particles per grid cell. For Lagrangian energy MPM, the grid resolution reflects the tetrahedron mesh resolution, i.e. grid Δx was chosen roughly the same as the average edge length of the tetrahedron mesh. In our examples, we used TetWild to generate the tetrahedron mesh for Lagrangian MPM [Hu et al. 2018].

6.1 Capturing different fracture modes

We test our method with fracture simulations in which excessive tension or shear force is applied. In Figure 14, we simulate the process of pulling on a cube and demonstrate how Lagrangian MPM prevents numerical fracture caused by excessive deformation. In Figure 6, we twist and pull a cube until the shearing forces cause material failure and the material becomes disconnected. On the left of Figure 8, we pull the 4 limbs of the armadillo until they break and observe how the fracture introduces momentum to the torso. On the right of Figure 8, we added the von Mises plasticity model to the particles to capture more shear-induced plastic deformation.

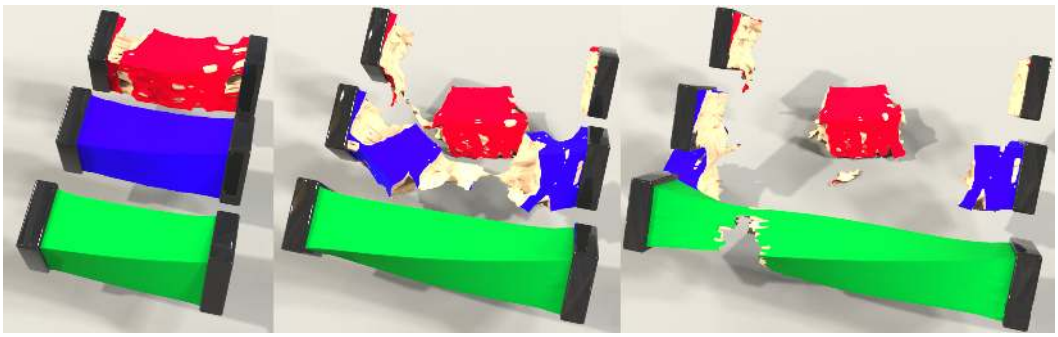

Fig. 14. We illustrate our treatment of numerical fracture with three simulations using the same particles. The red cube and blue cubes are simulated using traditional particle-based MPM with fine grid resolution (approximately 1 particle per grid cell) and coarse grid resolution (approximately 6 particles per grid cell) respectively. The green cube is simulated with our Lagrangian approach and fine grid resolution.

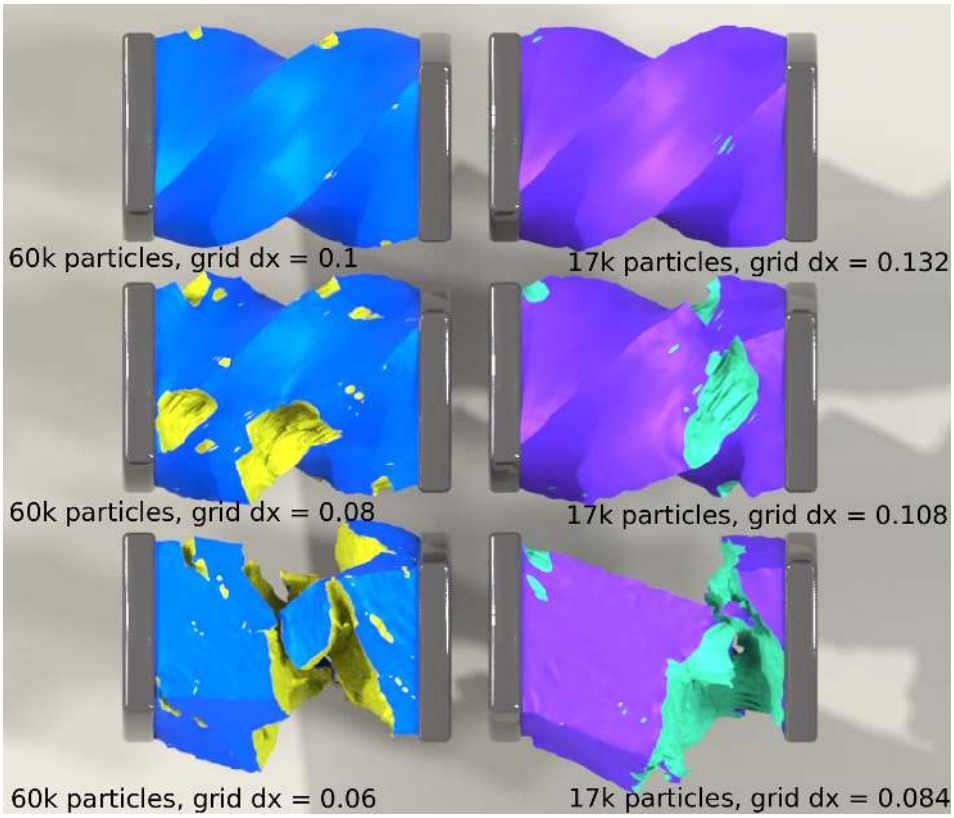

Fig. 15. We compare the same twisting cube simulation with different particle count and grid size. The sims with smaller grid dx to particle count ratio experience more fracture than the ones with larger ratio in the same frame.

6.2 Texturing objects

Our mesh visualization technique has the advantage that it naturally accommodates texturing based on an input mesh. E.g. all particles from the initial mesh are in the cut mesh and it is trivial to obtain a consistent vertex ordering based on the initial mesh for simplified texturing. In Figure 1, we simulated a zucchini being broken in half and demonstrated that its detailed texture is preserved. Also in Figure 1, we textured the ductile walls broken by the walking mannequin with SCA logos. In Figure 5, we textured the ductile sphere and created convincing details in the fracture scene.

6.3 Relaxed resolution requirements

In Figure 2, we simulated twisting of a cube with 8,000 particles. We compared two different renders: conventional particle fluid surface reconstruction and our approach. Our result captures significantly more detail and does not suffer from reconnection due to proximity. We also provide similar resolution comparison in Figure 6, and Figure 7. With our meshing technique, the results still look comparable even with comparatively low resolution.

Table 1. Our simulations and post-processes were run with 48 threads and 128 GB of RAM. Simulation and post-process time are measured in averaged seconds per frame, and resolution is measured by particle count.

	Sim	Post-process	Res
Pull - MPM (Fig. 14 red and blue)	0.6	0.5	8K
Pull - Lagrangian (Fig. 14 green)	0.6	0.5	8K
Projectile - 77K (Fig. 7 red)	2	5	77K
Projectile - 33K (Fig. 7 blue)	0.9	2	33K
Projectile - 14K (Fig. 7 yellow)	0.4	0.7	14K
Projectile - 5.5K (Fig. 7 orange)	0.2	0.3	5.5K
Twist - 60K (Fig. 6 blue)	11	5	60K
Twist - 17K (Fig. 6 purple)	4	1	17K
Twist - 8K (Fig. 6 green)	2	0.4	8K
Twist - 4K (Fig. 6 red)	2	0.2	4K
Twist von Mises (Fig. 8)	11	4	60K
Pulling with angle - 60K (Fig. 6 blue)	11	5	60K
Pulling with angle - 17K (Fig. 6 purple)	8	1	17K
Pulling with angle - 8K (Fig. 6 green)	8	0.4	8K
Pulling with angle - 4K (Fig. 6 red)	5	0.2	4K
Braiding Columns (see supplementary video)	2	3	50K
Braiding Columns (Fig. 3 and Fig. 4)	35	16	200K
Crushing Orange (Fig. 5)	15	8	130K
Zucchini (Fig. 1 bottom)	16	13	207K
Stretching Armadillo (see supplementary video)	49	27	299K
Tearing Armadillo (Fig. 8 left)	48	26	299K
Wall breaking (Fig. 1 right)	50	5	933K

7 DISCUSSION AND LIMITATIONS

Many existing FEM approaches for simulating ductile materials rely on the creation of a sufficiently high quality tetrahedron mesh to be used in the simulation. In the case of traditional particle based MPM, our mesh quality demands are practically non-existent. Indeed we simply use Delaunay tetrahedralization. In the case of Lagrangian mesh-based MPM our approach requires a mesh with the same quality constraints as traditional FEM. In either case, the MPM conception of our approach automatically resolves self-collision allowing us to simulate ductile fracture with comparably low implementation and computational complexity. Our approach does have a number of clear limitations. First, crack patterns are affected by particle sampling density/tetrahedron mesh topology and grid resolution. See Figure 15. Also, choosing appropriate parameters for edge splitting thresholds and crack surface smoothing iteration counts can vary from example to example.

ACKNOWLEDGMENTS

The work is supported by NSF CCF-1422795, ONR (N000141110719, N000141210834), DOD (W81XWH-15-1-0147), Intel STC-Visual Computing Grant (20112360) as well as a gift from Adobe Inc.

REFERENCES

B. Adams, M. Pauly, R. Keiser, and L. Guibas. 2007. Adaptively sampled particle fluids. *ACM Trans Graph* 26, 3 (2007). https://doi.org/10.1145/1276377.1276437

T. Anderson. 2017. *Fracture mechanics: fundamentals and applications.* CRC Press.

R. Ando, N. Thürey, and C. Wojtan. 2013. Highly adaptive liquid simulations on tetrahedral meshes. *ACM Trans Graph* 32, 4 (2013), 103:1–103:10. https://doi.org/10.1145/2461912.2461982

A. Bargteil, T. Goktekin, J. O'Brien, and J. Strain. 2006. A semi-Lagrangian contouring method for fluid simulation. *ACM Trans Graph* 25, 1 (2006).

A. Bargteil, C. Wojtan, J. Hodgins, and G. Turk. 2007. A finite element method for animating large viscoplastic flow. *ACM Trans Graph* 26, 3 (2007).

T. Belytschko, H. Chen, J. Xu, and G. Zi. 2003. Dynamic crack propagation based on loss of hyperbolicity and a new discontinuous enrichment. *Int'J Num Meth Eng* 58, 12 (2003), 1873–1905. https://doi.org/10.1002/nme.941

H. Bhattacharya, Y. Gao, and A. Bargteil. 2015. A level-set method for skinning animated particle data. *IEEE Trans Vis Comp Graph* 21 (2015), 315–327. Issue 3.

J. Blinn. 1982. A generalization of algebraic surface drawing. *ACM Trans Graph* 1, 3 (1982), 235–256. https://doi.org/10.1145/357306.357310

J. Bonet and R. Wood. 2008. *Nonlinear continuum mechanics for finite element analysis.* Cambridge University Press.

T. Brochu and R. Bridson. 2009. Robust topological operations for dynamic explicit surfaces. *SIAM J Sci Comp* 31, 4 (2009), 2472–2493. https://doi.org/10.1137/080737617

M. Buler, P. Diehl, D. Pflger, S. Frey, F. Sadlo, T. Ertl, and M. Schweitzer. 2017. Visualization of fracture progression in peridynamics. *Comp Graph* 67, C (2017), 45–57. https://doi.org/10.1016/j.cag.2017.05.003

O. Busaryev, T. Dey, and H. Wang. 2013. Adaptive fracture simulation of multi-layered thin plates. *ACM Trans Graph* 32, 4 (2013), 52:1–52:6. https://doi.org/10.1145/2461912.2461920

F. Chen, C. Wang, B. Xie, and H. Qin. 2013. Flexible and rapid animation of brittle fracture using the smoothed particle hydrodynamics formulation. *Comp Anim Virt Worlds* 24, 3-4 (2013), 215–224. https://doi.org/10.1002/cav.1514

W. Chen, F. Zhu, J. Zhao, S. Li, and G. Wang. 2018. Peridynamics-based fracture animation for elastoplastic solids. *Comp Graph Forum* 37, 1 (2018), 112–124. https://doi.org/10.1111/cgf.13236

M. Choi. 2014. Real-time simulation of ductile fracture with oriented particles. *Comp Anim Virt Worlds* 25, 3-4 (2014), 457–465. https://doi.org/10.1002/cav.1601

F. Da, C. Batty, and E. Grinspun. 2014. Multimaterial mesh-based surface tracking. *ACM Trans Graph* 33, 4 (2014), 112:1–112:11. https://doi.org/10.1145/2601097.2601146

F. Dagenais, J. Gagnon, and E. Paquette. 2017. Detail-preserving explicit mesh projection and topology matching for particle-based fluids. *Comp Graph Forum* 36, 8 (2017), 444–457. https://doi.org/10.1111/cgf.13091

N. Daphalapurkar, H. Lu, D. Coker, and R. Komanduri. 2007. Simulation of dynamic crack growth using the generalized interpolation material point (GIMP) method. *Int J Frac* 143, 1 (2007), 79–102. https://doi.org/10.1007/s10704-007-9051-z

M. Desbrun and M. Cani. 1998. Active implicit surface for animation. In *Graph Int.* 143–150.

D. Gerszewski, H. Bhattacharya, and A. Bargteil. 2009. A point-based method for animating elastoplastic solids. In *Proc 2009 ACM SIGGRAPH/Eurograph Symp Comp Anim.* ACM, 133–138. https://doi.org/10.1145/1599470.1599488

M. Gissler, M. Becker, and M. Teschner. 2007. Constraint sets for topology-changing finite element models. In *VRIPHYS.* 21–26.

J. Guo and J. Nairn. 2006. Three-Dimensional Dynamic Fracture Analysis Using the Material Point Method. *Comp Mod Eng Sci* 16 (2006).

Y. Hu, Q. Zhou, X. Gao, A. Jacobson, D. Zorin, and D. Panozzo. 2018. Tetrahedral Meshing in the Wild. *ACM Trans. Graph.* 37, 4, Article 60 (July 2018), 14 pages. https://doi.org/10.1145/3197517.3201353

P. Huang, X. Zhang, S. Ma, and X. Huang. 2011. Contact algorithms for the material point method in impact and penetration simulation. *Int J Num Meth Eng* 85, 4 (2011), 498–517. https://doi.org/10.1002/nme.2981

C. Jiang, T. Gast, and J. Teran. 2017. Anisotropic elastoplasticity for cloth, knit and hair frictional contact. *ACM Trans Graph* 36, 4 (2017), 152.

C. Jiang, C. Schroeder, A. Selle, J. Teran, and A. Stomakhin. 2015. The Affine Particle-In-Cell Method. *ACM Trans Graph* 34, 4 (2015), 51:1–51:10.

B. Jones, A. Martin, J. Levine, T. Shinar, and A. Bargteil. 2016. Ductile fracture for clustered shape matching. In *Proc ACM SIGGRAPH Symp Int 3D Graph Games.* ACM, 65–70.

P. Kaufmann, S. Martin, M. Botsch, and M. Gross. 2008. Flexible simulation of deformable models using discontinuous Galerkin FEM. In *Proc 2008 ACM SIGGRAPH/Eurograph Symp Comp Anim.* Eurographics Association, 105–115.

G. Klár, T. Gast, A. Pradhana, C. Fu, C. Schroeder, C. Jiang, and J. Teran. 2016. Drucker-prager Elastoplasticity for Sand Animation. *ACM Trans Graph* 35, 4 (2016), 103:1–103:12.

D. Koschier, J. Bender, and N. Thuerey. 2017. Robust eXtended Finite Elements for complex cutting of deformables. *ACM Trans Graph* 36, 4 (2017), 55:1–55:13. https://doi.org/10.1145/3072959.3073666

N. Liu, X. He, S. Li, and G. Wang. 2011. Meshless simulation of brittle fracture. *Comp Anim Virt Worlds* 22, 2-3 (2011), 115–124. https://doi.org/10.1002/cav.412

O. Mercier, C. Beauchemin, N. Thuerey, T. Kim, and D. Nowrouzezahrai. 2015. Surface turbulence for particle-based liquid simulations. *ACM Trans Graph* 34(6) (Nov 2015), 10.

N. Molino, Z. Bao, and R. Fedkiw. 2004. A virtual node algorithm for changing mesh topology during simulation. *ACM Trans Graph* 23, 3 (2004), 385–392. https://doi.org/10.1145/1015706.1015734

Matthias Müller. 2009. Fast and robust tracking of fluid surfaces. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim.* ACM, 237–245. https://doi.org/10.1145/1599470.1599501

M. Müller, D. Charypar, and M. Gross. 2003. Particle-based fluid simulation for interactive applications. In *Proc 2003 ACM SIGGRAPH/Eurograph Symp Comp Anim.* Eurographics Association, 154–159.

M. Müller, N. Chentanez, and T. Kim. 2013. Real-time dynamic fracture with volumetric approximate convex decompositions. *ACM Trans Graph* 32, 4 (2013), 115:1–115:10. https://doi.org/10.1145/2461912.2461934

M. Müller and M. Gross. 2004. Interactive virtual materials. In *Proc Graph Int.* Canadian Human-Computer Communications Society, 239–246.

M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff. 2007. Position based dynamics. *J Vis Comm Im Rep* 18, 2 (2007), 109–118.

M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa. 2004. Point based animation of elastic, plastic and melting objects. In *Proc ACM SIGGRAPH/Eurograp Symp Comp Anim.* 141–151. https://doi.org/10.1145/1028523.1028542

K. Museth. 2014. *A flexible image processing approach to the surfacing of particle-based fluid animation.* 81–84. https://doi.org/10.1007/978-4-431-55007-5_11

K. Museth, M. Clive, and B. Zafar. 2007. Blobtacular: surfacing particle system in "Pirates of the Caribbean 3". In *ACM SIGGRAPH 2007 Sketches (SIGGRAPH '07).* ACM. https://doi.org/10.1145/1278780.1278804

John A. Nairn. 2003. Material point method calculations with explicit cracks.

J. O'Brien, A. Bargteil, and J. Hodgins. 2002. Graphical modeling and animation of ductile fracture. In *Proc ACM SIGGRAPH 2002.* 291–294.

J. O'Brien and J. Hodgins. 1999. Graphical modeling and animation of brittle fracture. In *Proc 26th Conf Comp Graph Int Tech (SIGGRAPH '99).* ACM Press/Addison-Wesley Publishing Co., 137–146. https://doi.org/10.1145/311535.311550

M. Ohta, Y. Kanamori, and T. Nishita. 2009. Deformation and fracturing using adaptive shape matching with stiffness adjustment. *Comp Anim Virt Worlds* 20, 2–3 (2009), 365–373. https://doi.org/10.1002/cav.v20:2/3

E. Parker and J. O'Brien. 2009. Real-time deformation and fracture in a game environment. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim.* ACM, 165–175. https://doi.org/10.1145/1599470.1599492

M. Pauly, R. Keiser, B. Adams, P. Dutré, M. Gross, and L. Guibas. 2005. Meshless animation of fracturing solids. *ACM Trans Graph* 24, 3 (2005), 957–964. https://doi.org/10.1145/1073204.1073296

M. Pauly, R. Keiser, L. Kobbelt, and M. Gross. 2003. Shape modeling with point-sampled geometry. *ACM Trans Graph* 22, 3 (2003), 641–650. https://doi.org/10.1145/882262.882319

T. Pfaff, R. Narain, J. de Joya, and J. O'Brien. 2014. Adaptive tearing and cracking of thin sheets. *ACM Trans Graph* 33, 4 (2014), 110:1–110:9. https://doi.org/10.1145/2601097.2601132

S. Raghavachary. 2002. Fracture generation on polygonal meshes using Voronoi polygons. In *ACM SIGGRAPH 2002 Conf Abstracts App (SIGGRAPH '02).* ACM, 187–187. https://doi.org/10.1145/1242073.1242200

S. Schvartzman and M. Otaduy. 2014. Fracture animation based on high-dimensional voronoi diagrams. In *Proc ACM SIGGRAPH Symp Int 3D Graph Games.* ACM, 15–22. https://doi.org/10.1145/2556700.2556713

C. Shen and A. Shah. 2007. Extracting and parametrizing temporally coherent surfaces from particles.. In *SIGGRAPH Sketches.* 66.

E. Sifakis and J. Barbic. 2012. FEM simulation of 3D deformable solids: a practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 Courses (SIGGRAPH '12).* ACM, New York, NY, USA, 20:1–20:50. https://doi.org/10.1145/2343483.2343501

B. Solenthaler, J. Schläfli, and R. Pajarola. 2007. A unified particle model for Fluid-solid interactions. *Comp Anim Virt Worlds* 18, 1 (2007), 69–82. https://doi.org/10.1002/cav.v18:1

D. Steinemann, M. Otaduy, and M. Gross. 2009. Splitting meshless deforming objects with explicit surface tracking. *Graph Models* 71, 6 (2009), 209–220. https://doi.org/10.1016/j.gmod.2008.12.004

A. Stomakhin, C. Schroeder, L. Chai, J. Teran, and A. Selle. 2013. A Material Point Method for snow simulation. *ACM Trans Graph* 32, 4 (2013), 102:1–102:10.

J. Su, C. Schroeder, and R. Fedkiw. 2009. Energy stability and fracture for frame rate rigid body simulations. In *Proc 2009 ACM SIGGRAPH/Eurograph Symp Comp Anim.* ACM, 155–164. https://doi.org/10.1145/1599470.1599491

D. Sulsky, Z. Chen, and H. Schreyer. 1994. A particle method for history-dependent materials. *Comp Meth App Mech Eng* 118, 1 (1994), 179–196.

D. Terzopoulos and K. Fleischer. 1988. Modeling inelastic deformation: viscolelasticity, plasticity, fracture. *SIGGRAPH Comp Graph* 22, 4 (1988), 269–278.

S. Wang, M. Ding, T. Gast, L. Zhu, S. Gagniere, C. Jiang, and J. Teran. 2019. *Supplementary Technical Document* (2019).

M. Wicke, M. Botsch, and M. Gross. 2007. A finite element method on convex polyhedra. *Comp Graph Forum* 26 (2007), 355–364.

M. Wicke, D. Ritchie, B. Klingner, S. Burke, J. Shewchuk, and J. O'Brien. 2010. Dynamic local remeshing for elastoplastic simulation. *ACM Trans Graph* 29, 4 (2010), 49:1–11.

M. Wicke, M. Teschner, and M. Gross. 2004. CSG tree rendering for point-sampled objects. In *12th Pac Graph.* 160–168. https://doi.org/10.1109/PCCGA.2004.1348346

B. Williams. 2008. *Fluid surface reconstruction from particles.* Ph.D. Dissertation. University of British Columbia.

C. Wojtan, N. Thürey, M. Gross, and G. Turk. 2009. Deforming meshes that split and merge. *ACM Trans Graph* 28, 3 (2009), 76:1–76:10. https://doi.org/10.1145/1531326.1531382

C. Wojtan, N. Thürey, M. Gross, and G. Turk. 2010. Physics-inspired topology changes for thin fluid features. *ACM Trans Graph* 29, 4 (2010), 50:1–50:8. https://doi.org/10.1145/1778765.1778787

C. Wojtan and G. Turk. 2008. Fast viscoelastic behavior with thin features. *ACM Trans Graph* 27, 3 (2008), 1–8. https://doi.org/10.1145/1360612.1360646

J. Wretborn, R. Armiento, and K. Museth. 2017. Animation of crack propagation by means of an extended multi-body solver for the material point method. *Comp Graph* 69, C (2017), 131–139. https://doi.org/10.1016/j.cag.2017.10.005

J. Yu and G. Turk. 2013. Reconstructing surfaces of particle-based fluids using anisotropic kernels. *ACM Trans. Graph.* 32, 1 (2013), 5:1–5:12. https://doi.org/10.1145/2421636.2421641

J. Yu, C. Wojtan, G. Turk, and C. Yap. 2012. Explicit mesh surfaces for particle based fluids. *Comp Graph Forum* 31, 2pt4 (2012), 815–824. https://doi.org/10.1111/j.1467-8659.2012.03062.x

Y. Yue, B. Smith, C. Batty, C. Zheng, and E. Grinspun. 2015. Continuum foam: a material point method for shear-dependent flows. *ACM Trans Graph* 34, 5 (2015), 160:1–160:20.

N. Zhang, X. Zhou, D. Sha, X. Yuan, K. Tamma, and B. Chen. 2006. Integrating mesh and meshfree methods for physics-based fracture and debris cloud simulation. In *Symp Point-Based Graph*, M. Botsch, B. Chen, M. Pauly, and M. Zwicker (Eds.). Eurograph Assoc. https://doi.org/10.2312/SPBG/SPBG06/145-154

C. Zheng and D. James. 2010. Rigid-body fracture sound with precomputed soundbanks. *ACM Trans Graph* 29, 4 (2010), 69:1–69:13. https://doi.org/10.1145/1778765.1778806

Y. Zhu and R. Bridson. 2005. Animating sand as a fluid. *ACM Trans Graph* 24, 3 (2005), 965–972.