Continuum Foam: A Material Point Method for Shear-Dependent Flows

YONGHAO YUE and BREANNAN SMITH
Columbia University
CHRISTOPHER BATTY
University of Waterloo
CHANGXI ZHENG and EITAN GRINSPUN
Columbia University

We consider the simulation of dense foams composed of microscopic bubbles, such as shaving cream and whipped cream. We represent foam not as a collection of discrete bubbles, but instead as a continuum. We employ the Material Point Method (MPM) to discretize a hyperelastic constitutive relation augmented with the Herschel-Bulkley model of non-Newtonian viscoplastic flow, which is known to closely approximate foam behavior. Since large shearing flows in foam can produce poor distributions of material points, a typical MPM implementation can produce non-physical internal holes in the continuum. To address these artifacts, we introduce a particle resampling method for MPM. In addition, we introduce an explicit tearing model to prevent regions from shearing into artificially-thin, honey-like threads. We evaluate our method's efficacy by simulating a number of dense foams, and we validate our method by comparing to real-world footage of foam.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Animation*; I.6.8 [**Simulation and Modeling**]: Types of Simulation—*Animation*

General Terms: Algorithms

Additional Key Words and Phrases: Foam, material point method, particle resampling, tearing, shear thinning, shear thickening, viscoplasticity

1. INTRODUCTION

Foams are a unique class of materials that arise in many problem domains and that exhibit a remarkable range of behaviors. Application domains as diverse as firefighting, oil recovery, chemical filtration, and even industrial textile processing employ foams [Weaire and Hutzler 2001]. In computer graphics, foams are a useful component of many animated scenarios; indeed, the recent Pixar animated short "Partysaurus Rex" prominently showcases soap bubbles and foams [Harder and Mangnall 2013]. The simulation of foam is thus an important problem in both engineering and graphics contexts.

Foam bubbles encompass a wide spectrum of spatial scales, ranging from the macroscopic bubbles formed by bath soaps on the order of centimeters, to the immense number of bubbles that compose shaving foams on the order of tens of micrometers. We focus on the relatively unexplored case of dense foams, exemplified by materials such as shaving cream and whipped cream. As our focus is on dense foams, we do not model the drainage [Koehler et al. 1998] and collapse [Garrett 1993] effects characteristic of macro-scale bubbles. To guide the design of our method, we first consider the empirical properties of foams.

Dense foams consist of myriad individual bubbles, each individually difficult to see due to its small size; a typical shaving cream bubble is $40\mu m$ in diameter [Ovarlez et al. 2010]. Despite consisting of many small bubbles interacting with each other, the observed macroscopic behavior of a dense foam is that of a seamless *continuum*. This observation suggests that we can simulate foam in a homogenized way to resolve large-scale and visually compelling dynamics.

When viewed as a continuous medium, foam exhibits a non-Newtonian force response. Due to the microscopic stochastic rearrangement of bubbles, foams exhibit *viscoplasticity*, meaning they undergo rate-dependent permanent deformation. Foams also tend to exhibit *shear thinning* effects, in which the continuum flows more easily under larger applied stresses. Shear thinning, which also occurs in blood, motor oil, and paint, plays a significant role in the utility of many materials. For example, shear thinning allows paint to readily flow off of a brush, but not off of a wall. Viscoplasticity and shear thinning can induce large shearing flows and topological changes with the potential to radically rearrange a material's configuration.

With these observations in mind, we simulate bulk foam as a continuous medium, allowing us to treat larger volumes of foam than would be tractable at the individual bubble level. We adopt the *Herschel-Bulkley* constitutive relation for foam [Weaire and Hutzler 2001]. While a precise understanding of the relation between microscopic bubble mechanics and the aggregate behavior of foam at the macroscopic scale remains an open problem [Weaire 2008], empirical experiments have shown the Herschel-Bulkley model to be a highly effective phenomenological approximation. As an added bonus, we will demonstrate that the Herschel-Bulkley constitutive relation is general enough to model a wide variety of other interesting materials, including shear *thickening* "oobleck."

To address the challenges posed by foam's extreme shape changes, we adopt the hybrid particle/grid method known as the Material Point Method (MPM), which has been noted for the relative ease with which it handles large deformations and topology changes [Stomakhin et al. 2013; Stomakhin et al. 2014]. A direct application of MPM to our setting falls short, however. Large shearing flows can produce poor material point distributions, creating non-physical internal holes (see Figure 6). We remedy this problem with a novel particle resampling method for MPM that maintains uniform particle distributions while retaining the physical properties and the geometry of the foam.

Finally, we observe that an explicit tearing model is required to realistically simulate foam; without a tearing model, regions of foam can become progressively thinner, yielding artificially-slender, honey-like threads. These unnatural threads form because the grid maintains artificial topological connections at the grid resolution through the stencil of the shape functions; the particles are only regarded as topologically distinct when they are separated by a

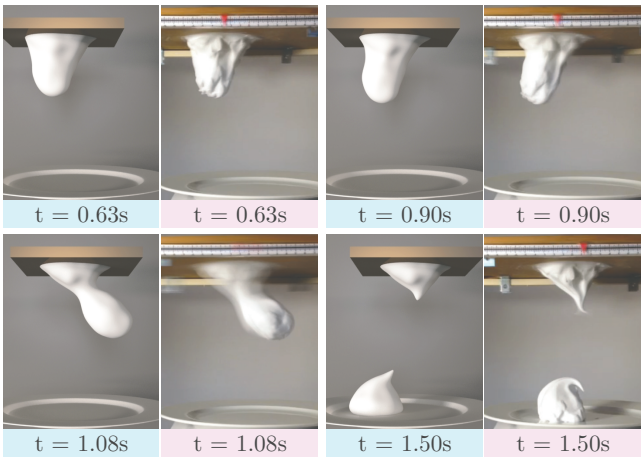

Fig. 1. **Comparison of our method to real-world video footage** of a dollop of shaving cream falling from an oscillating block. Our simulation results (left of each pair with blue labels) match closely with the captured video footage (right of each pair with pink labels).

sufficiently large distance. We therefore propose an extension to directly detect and tear weak regions based on their accumulated plasticity.

In this work we offer three core contributions: we extend the MPM framework to support the general viscoplastic Herschel-Bulkley material model, we introduce an effective particle resampling method for MPM, and we introduce an explicit tearing model. Taken together these features enable our continuum-based method to realistically simulate the familiar behaviors of dense foams, as well as a wide variety of related materials. We demonstrate the method's efficacy with several animated examples and with a comparison to recordings of real foam (see Figure 1).

2. RELATED WORK

The Physics of Foam. Due to the wide array of domains in which foams play a role, there has been substantial research into their properties and behavior. The text by Weaire and Hutzler [2001] provides a broad overview of earlier theoretical, experimental, and computational approaches to foam mechanics. The field of *rheology* is the study of flowing matter, including foams, although a systematic rheological understanding of foams remains to be fully developed [Weaire 2008]. As Weaire notes, the dominant continuum model is the nonlinear Herschel-Bulkley fluid model, a generalized model for non-Newtonian fluids that we adopt. While the aggregate behavior of foam is ultimately dictated by the small-scale mechanics of its constituent bubbles, for many applications, including ours, the Herschel-Bulkley model is well-suited.

Navier-Stokes Approaches. Multiphase fluid simulation methods [Hong and Kim 2005; Losasso et al. 2006] have been adapted to simulate the detailed deformations of small collections of bubbles [Zheng et al. 2006; Kim et al. 2007; Kim et al. 2010]. Multiphase extensions of the level set method are used to distinguish individual pockets of air, with the Navier-Stokes equations applied to evolve the complete system. Sethian et al. [2013] recently extended a similar framework with more advanced drainage and film rupture modeling.

Multiphase Smoothed Particle Hydrodynamics (SPH) techniques can likewise model submerged bubbles [Müller et al. 2005; Solenthaler and Pajarola 2008], though they have not been applied to foam structures.

Bubble Films as Triangle Meshes. An alternate approach is to model bubble interfaces as dynamic triangle meshes. This is exemplified by the *Surface Evolver* model [Brakke 1992], which seeks equilibrium configurations of surfaces under surface tension and various constraints including volume preservation. Ďurikovič [2001] developed a different triangle surface model for dynamic bubbles, treating the film as an elastic deformable object with surface tension and using Lennard-Jones forces for bubble interactions.

Particle Models for Foams and Bubbles. To scale up to denser foams composed of small bubbles, for which the deformations of individual films become less relevant, many authors have adopted models in which each bubble is represented by a single particle, typically with spring-like interparticle forces. These approaches are often layered atop a single-phase fluid solver in order to increase the apparent level of detail and realism. Focusing on foams themselves, Kück et al. [2002] applied spring forces between bubble particles to approximate the Plateau conditions of soap films at equilibrium. Greenwood and House [2004] later coupled this model to a Navier-Stokes fluid solver, while Thürey et al. [2007] coupled particle-based bubbles to a shallow water model. Cleary et al. [2007] augmented an SPH liquid simulator with a discrete bubble model, using cohesion forces to encourage the formation of foam "rafts" on the liquid surface, while Ihmsen et al. proposed a method to layer foam onto an SPH animation as a post-process [Ihmsen et al. 2012]. To better approximate the foam geometry and to improve volume preservation, Busaryev et al. [2012] exploited the properties of Voronoi diagrams. Hong et al. coupled an SPH bubble model with an Eulerian liquid to enliven the bubble behavior [Hong et al. 2008]. Patkar et al. [2013] tightly coupled a particle-based subgrid compressible model to a two-phase fluid simulator.

Multiphase Continua and Homogenization. Similar in spirit to our work, Kim et al. [2010] took a step away from individual interparticle forces in the context of dispersed bubble flows, instead adopting a variable density multiphase continuum model that accounts for combined bubble/liquid behavior. Nielsen et al. [2013] applied a continuum model to a similar problem in which small droplets of water are dispersed in air to produce a spray.

Homogenization methods [Nesme et al. 2009; Kharevych et al. 2009] develop coarse-scale discretizations that approximate aggregate fine-scale inhomogeneities in material properties and geometry. The extreme nonlinearities exhibited by foams makes this strategy inapplicable.

Methods for Viscoplastic and Non-Newtonian Materials. Various non-Newtonian and viscoplastic models have been developed to animate materials whose behavior cannot be neatly categorized as fluid or solid. Early on, Terzopoulos and Fleischer [1988] developed deformable models that incorporated viscoplasticity and fracturing effects. Goktekin et al. [2004] gradually accumulated strain within an Eulerian fluid solver to add elastic forces, while Bargteil et al. [2007] augmented a Lagrangian finite element method (FEM) elasticity solver with remeshing to robustly capture dramatic viscoplastic flows. Subsequent work in the Lagrangian FEM setting has focused on accelerating remeshing and reducing the associated numerical smearing [Wojtan and Turk 2008; Wicke et al. 2010]. Point-based methods using both SPH and moving-least-squares (MLS) approaches have also been extended to non-Newtonian materials [Müller et al. 2004; Keiser et al. 2005; Solenthaler et al. 2007; Gerszewski et al. 2009].

Independent of the particular underlying numerical technique, graphics authors have adapted a range of constitutive models for capturing non-Newtonian and viscoplastic effects. O'Brien et al. [2002] proposed a simple additive plasticity model, while later authors adopted a more powerful multiplicative plasticity model [Irving et al. 2004]. Bargteil et al. [2007] added support for creep effects and work hardening and softening. While we are motivated by foam in particular, the constitutive model we propose offers increased flexibility to handle a variety of plastic effects, including perfect plasticity, viscoplasticity, shear thinning and shear thickening.

FLIP and MPM Methods. We build on a hybrid particle/grid framework for elastoplastic materials called the Material Point Method (MPM) [Sulsky et al. 1995]. MPM uses Lagrangian particles to handle the transport of information through space, but exploits a background regular grid structure to compute elastic forces. As we discuss below (§3), this confers many of the advantages of both Lagrangian and Eulerian methods. MPM was originally developed as an application of the related FLIP method for compressible flow to elastic media [Brackbill and Ruppel 1986]. As with FEM, MPM is derived in the weak form and can treat history-dependent quantities as material points. Zhu and Bridson applied FLIP to reduce dissipation in incompressible fluid animations [Zhu and Bridson 2005]; notably, they also abstracted away individual grains to apply a continuum model for granular media, later extended by Narain et al. [2010]. Recently, Stomakhin et al. [2013] developed an MPM technique and constitutive model to animate the unique behavior of snow, likewise treating it as a continuum. Stomakhin et al. [2014] subsequently extended their MPM technique to incorporate phase changes due to heat flow and to enforce incompressibility with a projection.

Sampling. Nonuniform point distributions can pose problems for numerical methods that advect particles. Closest to our work is that of Edwards et al. [2012], who pointed out that shearing flows can quickly introduce anisotropic point distributions for particle-in-cell methods, like MPM; they proposed a (non-conservative) method to resample points. Stomakhin et al. [2014] also pointed to the need for a resampling technique for MPM as a direction for future work. Particle-based vortex methods [Cottet and Koumoutsakos 2000] can also suffer from this problem. Particle position adjustment, particle resampling, and the use of source/sink terms have all been suggested in the context of FLIP methods [Losasso et al. 2008; Narain et al. 2010; Ando et al. 2012]. Surface representation methods, such as particle/level set hybrids [Enright et al. 2002; Hieber and Koumoutsakos 2005], and surfel-based methods [Keiser et al. 2005], can also require resampling to avoid holes or loss of accuracy. A largely orthogonal but related problem arises in adaptive simulation, for both FLIP and SPH, in which particles must be merged or split to achieve different scales (e.g., [Adams et al. 2007; Ando et al. 2012]). Schechter and Bridson [2012] proposed using Poisson-disk sampling to place ghost air particles at each step and to seed the initial SPH particles according to a level set description of the geometry; we build on these ideas to resample the interior region of our MPM foam.

Material Failure and Fracture. Foams exhibit behaviors characteristic of both fluids and solids, leading to phenomena such as the formation of stiff peaks. Similar to solids, foams can tear and form stiff peaks, yet similar to fluids, detached samples of foam can reconnect under contact. While methods to explicitly handle material failure and fracture for stiff materials have been proposed for MPM [Zhou 1998; Schreyer et al. 2002; Sulsky and Schreyer 2004; Banerjee 2004; Guo and Nairn 2006], these techniques typically consider "permanent" fracture, and can be overkill for foam simulation. To this end, we propose a simple technique to explicitly model tearing in foams that also allows foams to reconnect.

3. BACKGROUND: MATERIAL POINT METHOD

Notation. In the following, we depict scalars in regular roman type (e.g., ρ and J), vectors in bold italic type (e.g., $\boldsymbol{v}$ and $\boldsymbol{g}$), and rank-two tensors in bold sans serif type (e.g., $\boldsymbol{\sigma}$ and $\mathbf{S}$).

Continuous Setting. Consider a continuous medium over a domain $\Omega \subset \mathbb{R}^3$, with scalar density field $\rho(x, y, z)$, vectorial velocity field $\boldsymbol{v}(x, y, z)$, tensorial Cauchy stress field $\boldsymbol{\sigma}(x, y, z)$, and subject to gravitational acceleration $\boldsymbol{g}$. The dynamics of this medium evolve over time t under the Euler-Lagrange partial differential equation

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

subject to the mass conservation condition

$$\frac{D\rho}{Dt} + \rho \nabla \cdot \boldsymbol{v} = 0$$

where $\frac{D\cdot}{Dt}$ denotes the material derivative, i.e., $\frac{D\phi}{Dt} = \frac{\partial \phi}{\partial t} + \boldsymbol{v} \cdot \nabla \phi$ for a time-varying field $\phi(x, y, z)$. The details of our constitutive model, which defines $\boldsymbol{\sigma}$, are presented in §5.

Discretization. The material point method [Sulsky et al. 1995] discretizes the Euler-Lagrange equations in both space and time via a sequence of Eulerian and Lagrangian steps. This discretization approach is ideally suited for non-Newtonian and viscoplastic materials, which possess characteristics of both fluids and solids.

Material geometry and kinematic properties (density, velocity, deformation gradient, etc.) are associated with particles, which allows them to be transported easily through space in a Lagrangian manner. This circumvents traditional challenges of Eulerian methods, especially artificial numerical dissipation.

Material forces and particle accelerations are computed using a static regular (Cartesian) *background grid*, and then applied to the Lagrangian particles. Recourse to a *static* grid avoids the potentially complex and expensive remeshing required by methods that transport the mesh/grid itself in a Lagrangian manner, while the *regular* grid structure simplifies discretization and optimization. The use of a grid to *increment* rather than overwrite particle data reduces numerical dissipation.

MPM also makes various geometric aspects of large-deformation foam dynamics easier to handle: the Lagrangian particles implicitly account for topological splitting and merging; the Eulerian grid implicitly accounts for (self-)collision.

The net result is a method that is highly effective at modeling both solid and fluid properties. Like mesh-based Lagrangian techniques, MPM can be used with an arbitrary constitutive law to model complex materials; this flexibility allowed Stomakhin et al. [2013] to achieve impressive snow effects by combining MPM with a specially-designed snow constitutive model. Like many discretization methods, MPM is also able to provide exact conservation of certain quantities. In our implementation, which generally follows Stomakhin et al. [2013], mass is exactly conserved.

4. OVERVIEW

We begin by summarizing the procedures that comprise each time step of our method. Figure 2 depicts this sequence, which mirrors the overall structure of the method of Stomakhin et al. [2013]. We make substantial modifications in the procedures highlighted in

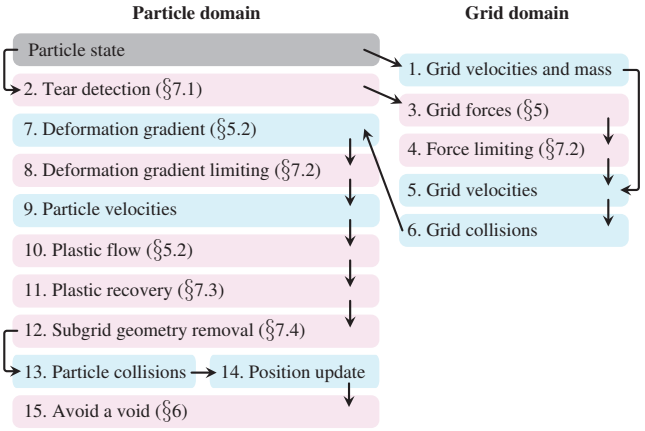

Fig. 2. **An overview of the stages of the MPM algorithm**, with our additions highlighted in pink. The arrows indicate the data flow.

pink, which implement our constitutive model (procedures 3 and 10), our tearing model (procedures 2, 4, 8, 11, 12), and our particle resampling technique (procedure 15).

Each particle carries several pieces of data: position $\boldsymbol{x}$, velocity $\boldsymbol{v}$, mass m, (initial) volume V, deformation gradient $\mathbf{F}$, and normalized elastic Green strain tensor $\overline{\mathbf{b}}^e = \det(\mathbf{b}^e)^{-1/3}\mathbf{b}^e$. *Here and henceforth the bar operator isotropically rescales a tensor to have unit determinant*, i.e., $\overline{\mathbf{x}} = \det(\mathbf{x})^{-1/3}\mathbf{x}$. Our Eulerian grid-based FEM discretization relies on the one-dimensional cubic B-spline basis functions advocated by Stomakhin et al. [2013].

Briefly, the steps of our algorithm are:

(1) **Rasterize particle data to the grid.** At the start of every time step we transfer each particle's mass and velocity to the Eulerian grid.

(2) **Detect tearing regions.** We examine each particle's accumulated plasticity to detect regions that are tearing; this determines where the material should be *weakened* during the next step (§7.1).

(3) **Compute grid forces.** Returning to the grid, we compute the grid-based forces according to our constitutive model (§5).

(4) **Limit forces.** We modify the stress and the force in tearing regions according to our tearing model (§7.2).

(5) **Update grid velocities.** We integrate the resulting grid-based forces to yield new grid-based velocities.

(6) **Resolve grid-based collisions.** We update the grid-based velocities to account for collisions with objects ([Stomakhin et al. 2013] §8).

(7) **Update particle deformation data.** We update the per-particle deformations based on the velocities (§5.2).

(8) **Limit deformation gradient.** We limit the change in the deformation gradient in the tearing regions according to our tearing model (§7.2).

(9) **Update particle velocities.** We apply the incremental *change* in the grid velocities back to the particle velocities (à la FLIP methods).

(10) **Compute plastic flow.** We compute a *plastic update* from the deformation gradient, transferring excess elastic strain into permanent plastic strain (§5.2).

(11) **Compute plastic recovery.** We relax the plasticity history to account for the strengthening of bonds between recently adhered bubbles (§7.3).

(12) **Remove subgrid geometry.** We remove thin, weakened particles whose physics cannot be resolved on the grid (§7.4).

(13) **Resolve particle-based collisions.** We update the particle-based velocities to account for collisions with objects ([Stomakhin et al. 2013] §8).

(14) **Update particle positions.** We update the particle positions according to the particle velocities.

(15) **Avoid a void.** Finally, we resample the material points to fill non-physical voids and to maintain a uniform particle distribution (§6).

5. CONSTITUTIVE MODEL

Foams (such as Figure 3, bottom) consist of concentrated dispersions of gas bubbles immersed in a surfactant solution [Weaire and Hutzler 2001]. Foams exhibit features over a broad range of length scales, from the microscopic liquid-gas interfaces of individual bubbles, to the macroscopic body which is well approximated as a continuous medium.

At the macroscopic scale, foams behave as shear thinning fluids for which the relationship between shear stress and strain rate is well approximated by the Herschel-Bulkley phenomenological law [Herschel and Bulkley 1926; Cohen-Addad et al. 2013]. As described by Weaire and Hutzler [2001], the application of a small stress deforms individual bubbles, which is reflected at a macroscopic scale as a reversible, elastic behavior. Increasing the stress, however, induces a rearrangement of bubbles, which is in turn reflected as a macroscopic plastic deformation, or a change in rest state. A continued increase in stress eventually induces a shear thinning flow of the surfactant solution and its immersed bubbles.

We model foam macroscopically. To capture a wide range of observed foam phenomena, we desire a constitutive relation that can model elasticity, plasticity and shear thinning flow.

Deformation Gradient and Tensor. We begin our presentation of the constitutive model with the *deformation gradient*, $\mathbf{F}(\boldsymbol{x})$, a linear mapping of an infinitesimal material neighborhood from its undeformed to its deformed position [Simo and Hughes 1998]. The deformation gradient maps every infinitesimal vector $\delta\boldsymbol{x}$ anchored at material point $\boldsymbol{x}$ in the undeformed configuration to its corresponding vector $\mathbf{F}(\boldsymbol{x})\delta\boldsymbol{x}$ in the deformed configuration. We typically omit the argument, $\boldsymbol{x} = (x, y, z)$, while bearing in mind that $\mathbf{F} : \Omega \mapsto \mathbb{R}^{3\times3}$ is in general a heterogeneous field over the material domain.

In the treatment of plasticity, it is convenient to decompose $\mathbf{F}$ into parts associated to the elastic, $\mathbf{F}^e$, and the plastic, $\mathbf{F}^p$, deformation, via the (multiplicative) decomposition [Simo and Hughes 1998; Irving et al. 2004; Bargteil et al. 2007; Wicke et al. 2010; Jones et al. 2014]

$$\mathbf{F} = \mathbf{F}^e\mathbf{F}^p \,.$$

Plasticity serves as the macroscopic description of a *volume-preserving* bubble rearrangement, thus by ansatz $\det(\mathbf{F}^p) = 1$. In addition, our treatment of elasticity will also refer to the *left Cauchy-Green tensor* $\mathbf{b} = \mathbf{F}\mathbf{F}^T$ and its elastic part $\mathbf{b}^e = \mathbf{F}^e\mathbf{F}^{eT}$. The tensors $\mathbf{b}$ and $\mathbf{b}^e$ are objective (i.e., frame-indifferent).

Elasticity. The application of a slight strain on the foam performs reversible elastic work, captured by the hyperelastic stored energy

density ([Simo and Hughes 1998]-§9.2)

$$W = W_v(J) + W_s(\overline{\mathbf{b}}^e) \tag{1}$$

whose terms account for volumetric and shear deformation, weighted by the material-dependent *bulk modulus* κ and the *shear modulus* μ, respectively; correspondingly, $\int_\Omega W \, \mathrm{dVol}$ is the stored energy, which is also objective.

The volume-dependent energy density

$$W_v(J) = \frac{1}{2}\kappa \left[\frac{1}{2}(J^2 - 1) - \ln J\right]$$

is expressed in terms of the square and natural logarithm of the scalar $J = \det(\mathbf{F})$, whereas the shear-dependent energy density

$$W_s(\overline{\mathbf{b}}^e) = \frac{1}{2}\mu(\mathrm{Tr}[\overline{\mathbf{b}}^e] - 3)$$

is expressed in terms of the *volume preserving left-Cauchy Green tensor* $\overline{\mathbf{b}}^e = \det(\mathbf{b}^e)^{-1/3}\mathbf{b}^e = J^{-2/3}\mathbf{b}^e$. This energy function alone characterizes the constitutive relation for finite stresses. In particular, it gives rise to the Kirchhoff stress tensor (see Appendix B for the derivation)

$$\boldsymbol{\tau} = \frac{\partial W}{\partial \mathbf{F}^e}\mathbf{F}^{eT} = \frac{\kappa}{2}(J^2 - 1)\mathbf{I} + \mu \, \mathrm{dev}[\overline{\mathbf{b}}^e] \, , \tag{2}$$

whose two terms encode an isotropic response resulting from volumetric and shear strains, respectively. In the expression above, $\mathrm{dev}[\mathbf{x}] = \mathbf{x} - \frac{\mathrm{Tr}[\mathbf{x}]}{3}\mathbf{I}$, and $\mathbf{I}$ denotes the 3×3 identity tensor. For any tensor $\mathbf{x}$, the deviatoric operator $\mathrm{dev}[\mathbf{x}]$ returns the trace-free part of $\mathbf{x}$, i.e., $\mathrm{Tr}[\mathrm{dev}[\mathbf{x}]] = 0$. Thus, $\mathrm{dev}[\mathbf{I}] = \mathbf{0}$, and $\mathrm{dev}[\mathrm{dev}[\mathbf{x}]] = \mathrm{dev}[\mathbf{x}]$. We will use all of these identities in §5.2. The Kirchhoff tensor is related to the Cauchy stress by $\boldsymbol{\sigma} = \boldsymbol{\tau}/J$. The internal force $\boldsymbol{f}$ is given by $\boldsymbol{f} = \nabla \cdot \boldsymbol{\sigma}$.

The treatment of plasticity shall make use *only* of the deviatoric part of the Kirchhoff stress tensor, $\mathbf{s} = \mathrm{dev}[\boldsymbol{\tau}] = \mu \, \mathrm{dev}[\overline{\mathbf{b}}^e]$, its (scalar) magnitude $s = \|\mathbf{s}\|_F$, and its normalized form $\hat{\mathbf{s}} = \mathbf{s}/s$; here $\|\cdot\|_F$ denotes the Frobenius norm. Intuitively, s encodes the magnitude of the material's elastic response to shear, ignoring the orientation of the shear mode.

Plastic Yield. The *von Mises yield condition* [Simo and Hughes 1998; O'Brien et al. 2002] makes precise the limits of the elastic regime, in terms of a material-dependent *yield stress* σ_Y; note that because we employ an *isotropic elastic* response, σ_Y is a scalar. So long as the condition

$$\Phi(s) = s - \sqrt{\frac{2}{3}}\sigma_Y \leq 0 \tag{3}$$

is satisfied, the material response remains elastic. Intuitively, the material cannot sustain a purely elastic response when the magnitude of the elastic response to shear deformation exceeds the threshold σ_Y. We refer to the quantity $\max(0, \Phi(s))$ as the *yield excess*, and the inequality $\Phi(s) \leq 0$ as the *yield condition*.

For more general plastic behavior, work hardening and softening are modeled by an update of the yield stress [Simo and Hughes 1998]; however, in quasistatic experiments of foams [Weaire and Hutzler 2001] it was observed that there is no apparent tendency for hardening or softening. We therefore safely neglect this effect.

5.1 Plastic Model

When the von Mises condition is violated, the foam deforms plastically; mathematically, this is modeled by updating the plastic strain

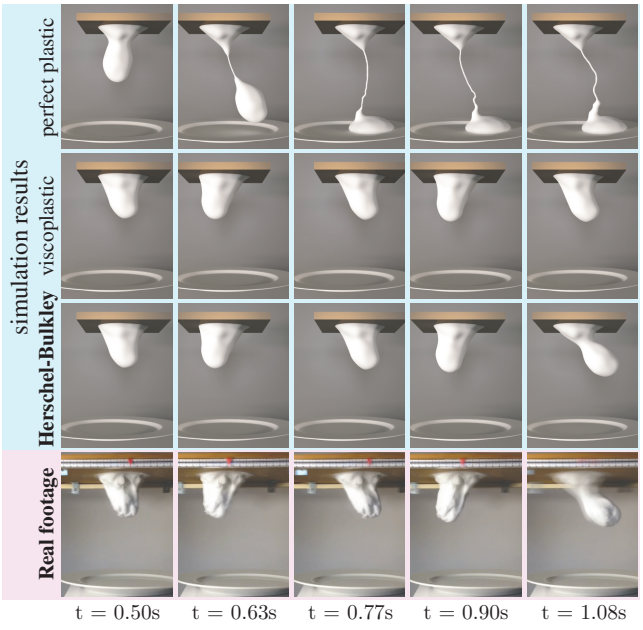

Fig. 3. **Comparison** of real footage (bottom row) of a dollop of dense shaving cream falling from an oscillating block to simulations with three plasticity models (perfect plastic, viscoplastic and Herschel-Bulkley in the top three rows). The behavior of the Herschel-Bulkley model most closely matches that of the real experiment. For the sake of comparison, we only vary the yield stress σ_Y, the viscosity η, and the Herschel-Bulkley power h. The thin filament visible in the perfect plasticity model can be removed by modifying the tearing parameters.

(and thus reducing the elastic strain) in the decomposition of the strain tensor. The time derivative of $\mathbf{b}^e$ is given by the identity

$$\dot{\mathbf{b}}^e = \mathbf{L}\mathbf{b}^e + \mathbf{b}^e\mathbf{L}^T + \mathcal{L}_{\boldsymbol{v}}\mathbf{b}^e \, , \tag{4}$$

which involves both the spatial velocity gradient $\mathbf{L} = \nabla \boldsymbol{v} = \dot{\mathbf{F}}\mathbf{F}^{-1}$, as well as the Lie derivative of $\mathbf{b}^e$, $\mathcal{L}_{\boldsymbol{v}}\mathbf{b}^e$ (see Simo and Hughes [1998]). The former captures the change in the strain due to the flow field itself, while the latter Lie derivative captures the change relative to the flow induced by $\boldsymbol{v}$, i.e., due to the *plastic* flow.

In rheology, one adopts a flow rule to describe the plastic deformation, in which excess elastic strain *flows* into plastic strain *over time*. We use the flow rule of Simo and Hughes (see [Simo 1988]-§1.4, [Simo and Hughes 1998]-§9)

$$\mathcal{L}_{\boldsymbol{v}}\mathbf{b}^e = -\frac{2}{3}\,\mathrm{Tr}[\mathbf{b}^e]\gamma\hat{\mathbf{s}} \, , \tag{5}$$

where γ is the flow rate, and $\hat{\mathbf{s}}$ is the flow direction. This flow rule is objective and can be derived by applying the principle of maximum plastic dissipation to the stored energy function (1) and the yield condition (3) [Simo 1988]-§1.4.

Perhaps the simplest update model is *perfect plasticity*, in which the *excess* elastic strain (i.e., the amount that exceeds the von Mises condition) is *instantaneously* transferred from the elastic to the plastic strain; in the context of perfect plasticity, γ in Eq. (5) is called the *consistency parameter*, and is obtained by solving Eq. (5) with the condition that the elastic strain, and the attendant stress, never exceeds the yield stress (see Figure 4).

Albeit simple, *perfect* plasticity fails to capture the *visco*plastic behavior of foam. Consider an experiment in which we place real

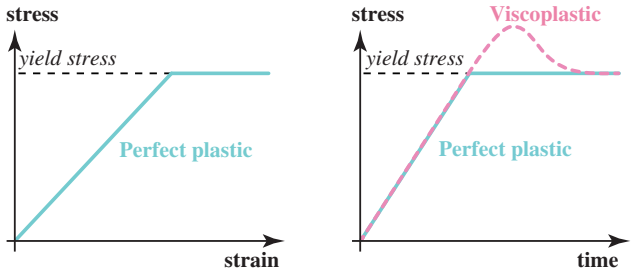

Fig. 4. **Comparison of constitutive relations:** (Left) An illustration of the stress-strain relation for the perfectly plastic model. (Right) A comparison of the perfect plastic and viscoplastic models as we increase the total strain at a constant strain rate. While the perfectly plastic model immediately removes any excess stress, the viscoplastic model acts over a finite timescale. Hence, under the viscoplastic model, the total stress can exceed the yield stress. This (large) stress is subject to elastic deformation, hence the material resists flowing.

foam on the underside of a wooden block (see Figure 3 and the accompanying video) and shake vigorously in a horizontal motion. Initially, the foam vibrates in an elastic manner, and the material flows slowly; over time, however, the foam begins to rapidly flow, leading to the formation of a thin neck that eventually tears away. Perfect plasticity fails to capture this behavior: the foam flows from the onset of agitation.

The block experiment motivates the need for a *visco*plastic model (see Figure 4), where the flow rate is replaced by a function of the viscosity-normalized yield excess as

$$\gamma = \gamma(s) = \max\left(0, \frac{\Phi(s)}{\eta}\right), \tag{6}$$

where η is the viscosity (or relaxation) coefficient. Eq. (6) is known as the *Bingham plastic* model [Bingham 1922], a common model for toothpaste, mayonnaise, and similar materials.

With a viscoplastic model in hand, we revisit the block experiment (see Figure 3). When the foam is agitated, the excess strain cannot be instantaneously eliminated, and thus early vibrations are seemingly elastic; over a time scale determined by the viscosity η, the rate-limited plastic flow leads to the formation of a neck and then to tearing.

While a considerable improvement over perfect plasticity, a viscoplastic law does not capture the *shear thinning* behavior of foam; the formation of the neck happens too slowly. Refer to Fig. 5 and the accompanying video for a comparison of the plasticity models.

Herschel-Bulkley Model. The *Herschel-Bulkley* model replaces (6) with a power law relating the plastic flow rate to the yield excess:

$$\gamma(s) = \max\left(0, \left(\frac{\Phi(s)}{\eta}\right)\right)^{1/h} \tag{7}$$

Shear *thinning* and *thickening* materials are characterized by $h < 1$ and $h > 1$, respectively.

To reveal the advantage of the Herschel-Bulkley model, we rewrite $(\Phi/\eta)^{1/h}$ as

$$\left(\frac{\Phi}{\eta}\right)^{1/h} = \frac{\Phi}{\eta^{1/h}\Phi^{1-1/h}}. \tag{8}$$

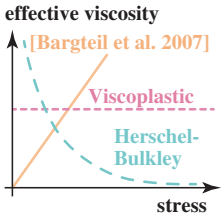

Fig. 5. A comparison of the effective viscosity for three plasticity models. While the effective viscosity of a viscoplastic material is independent of stress, the effective viscosity of a shear thinning Herschel-Bulkley material shrinks with increasing stress. We also note that the effective viscosity increases as a function of stress (see Appendix A) in the model of Bargteil et al. [2007].

This expression shows that the denominator $\eta^{1/h}\Phi^{1-1/h}$ is the effective viscosity coefficient from the vantage point of the standard viscoplastic model (6); thus, a shear thinning material such as ours has an effective viscosity that decreases as stress increases (see Figure 5). When a larger stress is imposed, the foam flows more easily.

The Herschel-Bulkley model captures a wide range of viscoplastic behaviors. While we use $h < 1$ for foams, the model can also capture shear thickening behaviors as exemplified by materials such as *oobleck*, a mixture of cornstarch and water (for a related treatment via finite elements, see earlier work by Bargteil et al. [2007]).

Each of the increasingly sophisticated viscoplastic models we have discussed generalizes its predecessor. Setting $h = 1$ recovers the viscoplastic (Bingham) model; taking the limit $\eta \to 0$ recovers perfect plasticity.

Temporal Evolution of the Left Cauchy-Green Tensor. The temporal evolution of the left Cauchy-Green tensor $\mathbf{b}^e$ typically involves an *elastic prediction* that applies the elastic deformation followed by a *plastic correction*. Substituting our flow rule (5) into (4), we obtain

$$\dot{\mathbf{b}}^e = \mathbf{L}\mathbf{b}^e + \mathbf{b}^e\mathbf{L}^T - \frac{2}{3}\,\mathrm{Tr}[\mathbf{b}^e]\gamma(s)\hat{\mathbf{s}}\,. \tag{9}$$

We treat Eq. (9) through operator splitting, which first integrates the elastic prediction

$$\dot{\mathbf{b}}^e = \mathbf{L}\mathbf{b}^e + \mathbf{b}^e\mathbf{L}^T \tag{10}$$

followed by a plastic correction

$$\dot{\mathbf{b}}^e = -\frac{2}{3}\,\mathrm{Tr}[\mathbf{b}^e]\gamma(s)\hat{\mathbf{s}}\,. \tag{11}$$

5.2 Herschel-Bulkley Temporal Discretization

For numerical simulations, we must temporally discretize (10) and (11). Because foam and more generally Herschel-Bulkley materials undergo large shears, our discretization departs from that of Stomakhin et al. [2013]. We extend the *return mapping method*, described by Simo and Hughes [1998] in the context of viscoplasticity, to accommodate the Herschel-Bulkley power law (7). Within this framework it is sufficient to track $\bar{\mathbf{b}}^e$, but to implement our tearing model, we additionally track $\mathbf{F}$.

Elastic Prediction. At time step $n+1$, we compute the *incremental* deformation gradient

$$\mathbf{f}_{n+1} = (\mathbf{I} + \Delta t \nabla \boldsymbol{v}_n) \tag{12}$$

due to all factors (e.g., elasticity, gravity) *other than plasticity*; $\boldsymbol{v}_n$ is the particle velocity, and Δt is the time step. We refer to the work of Stomakhin et al. [2013] for details on the computation of $\nabla \boldsymbol{v}_n$.

Next we discretize (10) and *predict* a change *only* in the elastic strain (see Appendix C), yielding an update to the volume preserving

elastic strain $\overline{\mathbf{b}}^e_{n+1}$ of

$$\overline{\mathbf{b}}^{e,\text{pre}}_{n+1} = \bar{\mathbf{f}}_{n+1}\overline{\mathbf{b}}^e_n\bar{\mathbf{f}}^T_{n+1} \,. \tag{13}$$

Likewise, we update $\mathbf{F}_{n+1}$ via $\mathbf{F}_{n+1} = \mathbf{f}_{n+1}\mathbf{F}_n$.

Since $\mathbf{s}$ is the deviatoric part of the Kirchhoff stress tensor $\boldsymbol{\tau}$, we have $\mathbf{s} = \mu\,\text{dev}[\overline{\mathbf{b}}^e]$, and the corresponding predicted stress is $\mathbf{s}^{\text{pre}}_{n+1} = \mu\,\text{dev}[\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}]$. If the predicted stress' magnitude s^{pre}_{n+1} satisfies the yield condition (3), we accept the prediction.

Plastic Correction. If the yield condition is violated, we correct the strain by discretizing (11) using fully implicit backward Euler.

Given the condition that volume does not change during plastic correction, the discretization of (11) is given by

$$\overline{\mathbf{b}}^e_{n+1} - \overline{\mathbf{b}}^{e,\text{pre}}_{n+1} = -\frac{2}{3}\Delta t\,\text{Tr}[\overline{\mathbf{b}}^e_{n+1}]\gamma(s_{n+1})\hat{\mathbf{s}}_{n+1} \,. \tag{14}$$

As shown in Appendix D, we transform this equation into a form that requires the solution of a single *scalar* formula

$$s_{n+1} - s^{\text{pre}}_{n+1} = -2\tilde{\mu}\Delta t\gamma(s_{n+1}) \,, \tag{15}$$

where $\tilde{\mu} = \frac{1}{3}\,\text{Tr}[\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}]\mu$, or equivalently

$$\eta^{1/h}\left(s_{n+1} - s^{\text{pre}}_{n+1}\right) + 2\tilde{\mu}\Delta t\left(s_{n+1} - \sqrt{\frac{2}{3}}\sigma_Y\right)^{1/h} = 0 \,. \tag{16}$$

This equation is well-defined as $\eta \to 0$. When $\eta = 0$ or $h = 1$, we obtain s_{n+1} explicitly as

$$s_{n+1} = s^{\text{pre}}_{n+1} - \left(s^{\text{pre}}_{n+1} - \sqrt{\frac{2}{3}}\sigma_Y\right) \Big/ \left(1 + \frac{\eta}{2\tilde{\mu}\Delta t}\right) . \tag{17}$$

Otherwise, we solve Eq. (16) numerically. Denote the left hand side of Eq. (16) as a function $g(s_{n+1})$. The yield condition is violated when we perform a plastic correction which implies that $s^{\text{pre}}_{n+1} > \sqrt{\frac{2}{3}}\sigma_Y$, hence $g(\sqrt{\frac{2}{3}}\sigma_Y) < 0$ and $g(s^{\text{pre}}_{n+1}) > 0$. Additionally, $g(s_{n+1})$ is monotonic in the interval $\sqrt{\frac{2}{3}}\sigma_Y \le s_{n+1} \le s^{\text{pre}}_{n+1}$. We thus solve for the unique root s_{n+1} via bisection.

Finally, we complete the correction by updating the Kirchhoff stress tensor via $\mathbf{s}_{n+1} = s_{n+1}\hat{\mathbf{s}}^{\text{pre}}_{n+1}$ and by updating the volumetric left Cauchy-Green strain tensor via

$$\overline{\mathbf{b}}^e_{n+1} = \frac{1}{\mu}\mathbf{s}_{n+1} + \frac{1}{3}\,\text{Tr}[\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}]\mathbf{I} \,. \tag{18}$$

Note that when we update $\overline{\mathbf{b}}^e_{n+1}$ in this way (i.e., by shrinking the deviatoric part while keeping the trace part the same), the determinant of the result will always have a value equal to or slightly larger than 1. We thus renormalize $\overline{\mathbf{b}}^e_{n+1}$ after the update.

We summarize this numerical update in Algorithm 1.

Algorithm 1 Updating the left Cauchy-Green tensor (9)

Compute the change in the deformation gradient $\mathbf{f}_{n+1}$ using (12)
Predict the volume-preserving part of the elastic strain using (13)
Compute the predicted Kirchhoff stress using (2)
if the yield condition is not violated ($\Phi \le 0$) **then**
  $\overline{\mathbf{b}}^e_{n+1} \leftarrow \overline{\mathbf{b}}^{e,\text{pre}}_{n+1}$ and **return**
else
  Solve the nonlinear equation (16) for s_{n+1}
  Update $\overline{\mathbf{b}}^e_{n+1}$ using equation (18)
  Renormalize $\overline{\mathbf{b}}^e_{n+1}$ to ensure that $\det(\overline{\mathbf{b}}^e_{n+1}) = 1$
end if

6. AVOID A VOID

Particle Resampling Under Large Deformation. In typical foams, the bulk modulus dramatically exceeds the shear modulus: according to Weaire and Hutzler [2001], the shear modulus is on the order of 10^1 Pa, while the bulk modulus is on the order of 10^5 Pa. As a result, foams can undergo substantial shearing deformations.

When material points are numerically advected through large deformations, their distribution can become highly nonuniform. For example, when a sample of foam is compressed along one axis and stretched along a perpendicular axis, the uniformity of the material point distribution can rapidly degrade, producing a sparse sampling along the stretched axis. In a method for which inter-particle distance governs material connectivity, such as FLIP or MPM, this sparsity results in the formation of numerically induced, nonphysical *voids* (Figure 6). In order to maintain a uniform point distribution, we develop a resampling approach for the material point method. As described below, we employ Poisson disk sampling to insert new points (§6.1), and we detect and merge points that are too close to one another (§6.2). We further exploit an estimated distance function to restrict particle resampling to interior regions, effectively distinguishing true exterior regions from artificial interior voids.

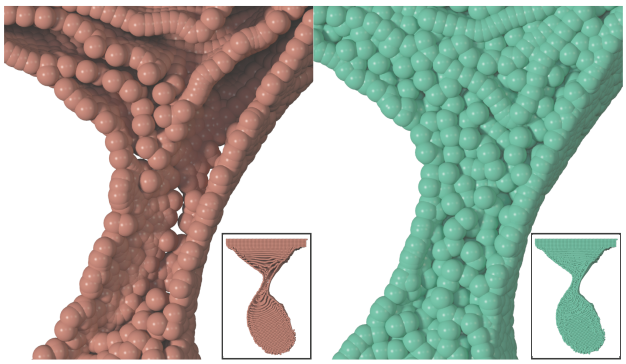

Fig. 6. **Efficacy of our particle resampling scheme**: (Left) A sheared blob of foam develops voids if additional material points are not introduced to the simulation. (Right) Our material point insertion technique prevents the formation of voids.

6.1 Inserting Points

We insert new material points to prevent the formation of numerically induced voids. We first compute an approximate narrowband *signed distance function* (SDF) in the vicinity of the foam-air interface, enabling us to classify points as foam or air. We then identify undersampled regions of the interior and insert new points to increase the sampling density.

Estimating the Interior Region. We compute the signed distance function [Sethian 1999] on a uniform grid (distinct from the MPM background grid) of cell size $r = \frac{1}{2}$(cell width of the background grid). For the purpose of this operation, we consider each material *point* to be a *sphere* of radius r, approximating the foam volume as a union of spheres. By convention, the signed distance of a grid point to the surface of a sphere is

Algorithm 2 `Compute_Signed_Distance`

$S \leftarrow \emptyset$ ▷ Initialize interface as empty sample set
for each grid point p **do**
  $p.d \leftarrow \infty$
end for
Register all spheres into a uniform grid for range queries
for each grid point p **do**
  $S' \leftarrow$ all spheres satisfying $dist(p, center) < 2r$
  $p.d \leftarrow$ min distance over all spheres $\in S'$, minus r
end for
for each grid edge e **do**
  if $\mathsf{sgn}(e.p1.d) \neq \mathsf{sgn}(e.p2.d)$ **then** ▷ Zero crossing
    S.append zeroCrossing$(e.p1, e.p2)$
  end if
end for
for each grid point p **do**
  $p.d \leftarrow \mathsf{sgn}(p.d) \cdot \infty$
end for
for each zeroCrossing C in S **do**
  for each grid point p satisfying $dist(p, C) < 4r$ **do**
    if $dist(p, C) < |p.d|$ **then**
      $p.d \leftarrow \mathsf{sgn}(p.d) \cdot dist(p, C)$
    end if
  end for
end for

positive when outside the sphere, zero on the surface, and negative when inside the sphere. Algorithm 2 consists of three steps: first, we categorize each grid point as interior or exterior based on its sign. We then find the interpolated zero crossing points along the grid edges, and treat these as samples of the interface. Finally, we use these interface samples to compute and assign improved estimates of the distance magnitude for all nearby grid points (within $4r$).

While the SDF magnitude is only meaningful in the vicinity of the boundary, its sign is meaningful everywhere. A point with negative distance is without a doubt inside the material.

Octree-Based Dart Throwing. Poisson disk sampling [Cook 1986] provides a simple technique to produce uniform sample distributions, while avoiding the insertion of samples too close to existing points. Our method is the first application of Poisson disk sampling to MPM simulation, to our knowledge. We apply the octree method proposed by Ebeida et al. [2012] in 3D. Whereas Ebeida et al. randomly draw a sample from the active domain in a probability proportional to the cell volume, we instead draw one sample uniformly from each active octree cell; refer to Algorithm 3 for pseudocode.

Put simply, we take a coarse to fine sampling approach. We begin with one, *active*, coarse cell, and we repeat the following until no active cells remain:

For each active cell, we generate a position sample at random. We retain the sample if it lies inside of the foam, as determined by thresholding the SDF, *and* if it lies outside of all material spheres, determined by using a modified radius of αr, where $\alpha < 1$. This ensures that material points can slightly overlap, thus preventing the formation of numerically induced voids. We employ a value of $\alpha \approx \frac{\sqrt{3}}{2} + \frac{1}{100}$ for all simulations. We choose α to be slightly larger than half the diagonal length of a unit cube. With this choice, if the particles remain within their initial distribution (8 particles per cell), we do not insert additional particles; if the distance between the particles grows larger than the initial setting, however, we insert new particles. To avoid inserting particles too close to the interface,

Algorithm 3 `Create_Particles_with_Poisson_Sampling`

$\Omega \leftarrow$ {set of all material points}
$A \leftarrow$ {set with one active cell that covers entire domain}
while $A \neq \emptyset$ **do** ▷ While there are active cells
  for $c \in A$ **do**
    Uniformly sample a point $\boldsymbol{x}$ in cell c
    if $\boldsymbol{x}$ is not covered by any sphere of radius αr **then**
      if $dist(\boldsymbol{x}) < -2.2 \cdot r$ **then** ▷ Sample the interior
        Accept $\boldsymbol{x}$ as a new sample
      end if
    end if
    Append all children of c to A
    Remove c from A
  end for
  for $c \in A$ **do**
    if c is covered by any sphere at $p \in \Omega$ of radius αr **then**
      Remove c from A
    end if
    if $c \cap \Omega = \emptyset$ ▷ Cell is void of material points
      or width$(c) < \frac{2r}{\sqrt{3}}$ **then** ▷ Cell is small
      if ($\nexists$ corner x of c) dist$(x) \leq -\alpha r$ **then**
        Remove c from A
      end if
    end if
  end for
end while

which would induce popping artifacts, we only insert points if the interpolated signed distance value is below a threshold of $-2.2r$. We empirically observe that thresholds larger than this value introduce noticeable popping artifacts. The threshold should be as large as possible to maximize the resampling region, however. We then deactivate the cell and recursively activate its eight children.

Before terminating the current iteration, we deactivate cells that we deem unnecessary. We deactivate cells that are fully covered by a set of material spheres, as well as cells that are sufficiently far from the interior and that are too small or empty.

Determining Physical Values. We need to assign physical values to newly inserted material points. For conserved quantities (i.e., mass m and volume V), we redistribute values from the surrounding points to the new point. For each introduced point, we first search for its neighboring points (e.g., within a distance r). Given N neighboring points with mass m_q and volume V_q, where $q = 1, ..., N$, we set the mass and volume of the added point to $m = \frac{1}{N+1}\sum_q m_q$ and $V = \frac{1}{N+1}\sum_q V_q$, respectively. We set the masses and volumes of the neighboring points to $m'_q = m_q - \frac{1}{N+1}m_q$ and $V'_q = V_q - \frac{1}{N+1}V_q$, respectively.

For field values that are not inherently conserved ($\boldsymbol{v}$, $\mathbf{F}$, $\overline{\mathbf{b}}^e$), we employ a mass-weighted interpolation. First, we compute a field value q_i at each grid point i using all of the material points that existed before resampling as

$$q_i = \frac{\sum_p w_{ip} m_p q_p}{\sum_p w_{ip} m_p} , \tag{19}$$

where p is the index of the material point, w_{ip} is the shape function, m_p is the mass of the point, and q_p is the field value at the point. Next, we compute the field value $\tilde{q}_p$ for the new point p as

$$\tilde{q}_p = \sum_i w_{ip} q_i . \tag{20}$$

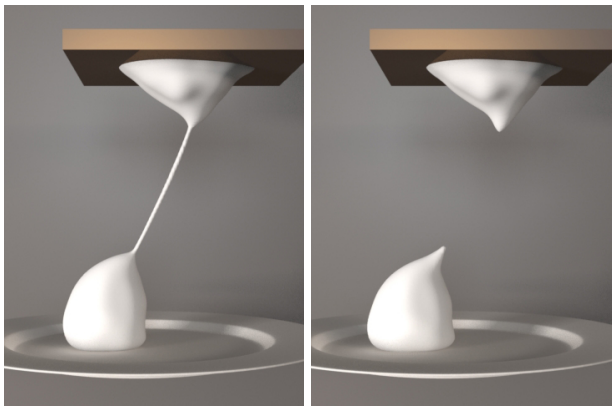

Fig. 7. **The importance of an explicit tearing model**: (Left) Without explicit tearing, a thin, non-physical tendril forms. (Right) Our explicit tearing model permits the thin tendril to break.

To ensure that $\det(\overline{\mathbf{b}}^e) = 1$, we renormalize $\overline{\mathbf{b}}^e$. To avoid a volume change in $\mathbf{F}$, which would correspond to change in its determinant, after interpolating both $\mathbf{F}$ and its determinant J, we rescale $\mathbf{F}$ to ensure that its determinant is equal to J. Specifically, given the interpolated value J and the initial interpolated value of $\mathbf{F}$ denoted $\tilde{\mathbf{F}}$, we set $\mathbf{F} = J^{1/3}(\det(\tilde{\mathbf{F}})^{-1/3}\tilde{\mathbf{F}})$.

6.2 Merging Points

To avoid an overly dense sampling, we merge points that are too close to one another. Like Ando et al. [2012], for each point we compute its closest neighbor, and if the separating distance is below a threshold ($0.03r$ in our implementation), we add the pair to a list. We then greedily find a set of disjoint point pairs, and merge each pair into a single new point.

To assign values to a new merged point, we simply sum the values of the conserved quantities, and compute a mass-weighted average for the remaining quantities. As in the preceding section, we again rescale $\mathbf{F}$ for consistency with J and we renormalize $\overline{\mathbf{b}}^e$.

7. TEARING

When a specimen of foam is increasingly stretched, it will grow thin and eventually tear. Standard MPM approaches lack an appropriate explicit tearing model, however: as long as particles remain close to one another relative to the grid spacing, they will exchange forces through the background grid and behave as a continuous body. As a result, foam simulated with MPM tends to become excessively thin before finally separating, and in the process forms long, unrealistic threads as shown in Figure 7.

To address this shortcoming, we propose to explicitly model tearing in our foam simulations. Physically, tearing occurs when the connections between incident bubbles are broken, yielding an effect analogous to particle dislocation in crystal structures. We first identify regions that are in a "weakened" or torn state. In these regions we directly modify the computed forces and the accumulation of strain so that the material no longer resists separation, and so that the observed geometric stretching (i.e., the separation) does not accumulate as additional physical strain. Finally, after tearing has occurred, we gradually adjust the accumulated plasticity to model the recovery of bubble neighbor connectivity in these regions. This allows the foam to return to a state in which it once again resists stretching.

7.1 Detecting Weak Particles

We first iterate over all particles and label those whose accumulated plasticity exceeds a threshold. Given a particle, its accumulated plasticity is $\mathbf{b}^p = \mathbf{F}^p(\mathbf{F}^p)^T$, where $\mathbf{F}^p$ is the plastic deformation gradient. Similar to the yield condition, a particle should be labeled as *weak* if $\|\operatorname{dev}[\mathbf{b}^p]\|_F > \sigma_T$, where the *tear yield* σ_T is a phenomenological material threshold.

To perform this check we need to compute $\|\operatorname{dev}[\mathbf{b}^p]\|_F$. For each particle, since we explicitly track the deformation gradient $\mathbf{F}$, its determinant J, and the normalized elastic strain $\overline{\mathbf{b}}^e$, we can compute $\|\operatorname{dev}[\mathbf{b}^p]\|_F$ as follows. First, we recover $\mathbf{b}^e = J^{2/3}\overline{\mathbf{b}}^e$. Next, denoting the *right* Cauchy-Green plastic deformation tensor by $\mathbf{C}^p = (\mathbf{F}^p)^T\mathbf{F}^p$, we compute it as $\mathbf{C}^p = \mathbf{F}^T(\mathbf{b}^e)^{-1}\mathbf{F}$. Finally, the Frobenius norm of the left and right Cauchy-Green tensors are the same, yielding

$$\|\operatorname{dev}[\mathbf{b}^p]\|_F = \|\operatorname{dev}[\mathbf{C}^p]\|_F \,.$$

We compare this scalar quantity to σ_T to identify weakened particles.

7.2 Modeling the Weakening Effect

Once a region of foam is weakened, it should no longer resist separation and it should freely tear; weakened foam, however, should continue to resist compression. To model this effect, we modify the stress $\boldsymbol{\sigma}$ (computed in §5). First, we compute the eigendecomposition $\boldsymbol{\sigma} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{V}^T$, where $\mathbf{V}$ is an orthonormal matrix, and $\boldsymbol{\Sigma} = diag(s_1, s_2, s_3)$ is a diagonal matrix. We then set each eigenvalue s_i to $min(0, s_i)$, and reassemble $\boldsymbol{\sigma}$. This modification sets the stress corresponding to expansion to 0, while permitting the material to resist compression. (This is loosely similar to the work of Wang et al. [2010], who clamp the singular values of strain to perform isotropic strain limiting.)

In addition, when we update the deformation gradient, we want to prevent regions that are in a weakened state from accumulating additional strain since we view them as torn. We therefore modify the incremental deformation gradient $\mathbf{f}$ in Eq. (12) to prevent the shear from increasing and to prevent volumetric expansion, as follows.

To detect if material is accumulating shear, we consider the shear before and after the proposed deformation. (Note that because the preexisting shear and the incremental shear may have different directions, the incremental shear alone does not provide the necessary information.) We compute the eigendecompositions of the normalized left Cauchy-Green tensor for the previous time step, $\overline{\mathbf{b}}^e_n$, and of the *estimated* left Cauchy-Green tensor for the current time step, $\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}$ (from Eq. (13)), and obtain their maximum eigenvalues λ_n and $\lambda^{\text{pre}}_{n+1}$, respectively. If $\lambda^{\text{pre}}_{n+1} \leq \lambda_n$, we let $\bar{\mathbf{f}}^{corr}_{n+1} = \mathbf{f}_{n+1}$. Otherwise, the material is accumulating shear, so we compute $J_f = \det(\mathbf{f}_{n+1})$, we find the singular value decomposition $\bar{\mathbf{f}}_{n+1} = J_f^{-1/3}\mathbf{f}_{n+1}$, and we set its singular values to 1 to obtain a corrected version $\bar{\mathbf{f}}^{corr}_{n+1}$. We then rescale to obtain the correct incremental deformation gradient $\mathbf{f}^{corr}_{n+1} = J_f^{1/3}\bar{\mathbf{f}}^{corr}_{n+1}$. This effectively disables shear deformations without affecting rotations.

Next, we disable the accumulation of expansion. We compute $J_n = \det(\mathbf{F}_n)$ and $J^{corr}_f = \det(\mathbf{f}^{corr}_{n+1})$. If $J_n > 1$ and $J^{corr}_f > 1$, this indicates that the material has a volume greater than its rest state and is continuing to expand. In this case we simply normalize $\mathbf{f}^{corr}_{n+1}$ to remove the expansion.

Finally, we proceed to use $\mathbf{f}_{n+1}^{corr}$ in place of $\mathbf{f}_{n+1}$ in the steps that follow Eq. (12).

7.3 Modeling Connectivity Recovery

After foam has torn, we do not want the material to stretch without resistance indefinitely. Instead, we model the bubbles comprising the foam as establishing connections with neighbors in the new configuration, with the strength of these connections gradually recovering over a finite time scale. We consider the new state to be a bubble rearrangement without dislocation and therefore without plasticity effects. In other words, we allow $\mathbf{b}^p$ to gradually reduce to the identity tensor over the time scale of the rearrangement.

We therefore need a method to update $\mathbf{F}$ that incorporates this change in the plasticity. Effectively, this requires manipulating $\mathbf{F}^p$, yet our particles do not explicitly carry this quantity. We will instead compute $\mathbf{F}^p$ from $\mathbf{C}^p$, which we can determine for each particle as in §7.1: $\mathbf{C}^p = {\mathbf{F}^p}^T\mathbf{F}^p = \mathbf{F}^T(\mathbf{b}^e)^{-1}\mathbf{F}$. While $\mathbf{C}^p$ is symmetric, $\mathbf{F}^p$ is in general not symmetric, and we can thus only estimate $\mathbf{F}^p$ up to some unknown rotation, *i.e.*, $\mathbf{F}^p = \mathbf{RB}$, where $\mathbf{R}$ is an arbitrary orthonormal matrix, and $\mathbf{B}$ is a symmetric matrix. Fortunately, this extra rotational freedom will not affect our update of the plasticity: we will only need to modify $\mathbf{B}$.

To construct the above factorization, we first compute the eigendecomposition $\mathbf{C}^p = \mathbf{USU}^T$. We then obtain $\mathbf{B}$ as the principal root of $\mathbf{C}^p$ via $\mathbf{B} = \mathbf{U}\sqrt{\mathbf{S}}\mathbf{U}^T$. Next, we let $\mathbf{X} = \mathbf{FB}^{-1}$, giving us the desired factorization of $\mathbf{F}$ into $\mathbf{F} = \mathbf{XB}$.

To update the plasticity, we would like to replace $\mathbf{B} = \mathbf{U}\sqrt{\mathbf{S}}\mathbf{U}^T$ with a "relaxed" version. We relax $\mathbf{B}$ by replacing $\sqrt{\mathbf{S}} = diag(\lambda_1, \lambda_2, \lambda_3)$, where $\lambda_1, \lambda_2, \lambda_3 > 0$, with a new diagonal matrix $\mathbf{P} = diag(\tilde{\lambda}_1, \tilde{\lambda}_2, \tilde{\lambda}_3)$. We need to ensure that $\det(\mathbf{F}^p) = 1$ after this update, which in turn requires that $\det(\mathbf{P}) = 1$. Additionally, regarding $\mathbf{P}$ as a function of time, we require that $\mathbf{P}(0) = \sqrt{\mathbf{S}}$ and $\mathbf{P}(\infty) = \mathbf{I}$ to allow the foam to steadily transition from the current plasticity level to zero plasticity. To model this behavior, we first emphasize the fact that $\det(\mathbf{F}^p) = \lambda_1\lambda_2\lambda_3 = 1$ and, that for any real value α, $\lambda_1^\alpha\lambda_2^\alpha\lambda_3^\alpha = 1$. If we regard α as a function of time and if we let $\tilde{\lambda}_i(t) = \lambda_i^{\alpha(t)}$, then $\det(\mathbf{P}(t)) = 1$ is automatically satisfied. For $\alpha(t)$, we want to select a function that ensures that $\alpha(0) = 1$ and that $\alpha(\infty) = 0$. $\mathbf{P}(t)$ then further satisfies $\mathbf{P}(0) = \sqrt{\mathbf{S}}$ and $\mathbf{P}(\infty) = \mathbf{I}$. Specifically, we choose $\alpha(t) = exp(-t/\eta_p)$, where we call η_p the *plasticity relaxation coefficient*. Combining these ingredients, we obtain $\tilde{\lambda}_i(t) = \lambda_i^{exp(-t/\eta_p)}$, which is the solution to the differential equation $\dot{\tilde{\lambda}}_i(t) = -\frac{1}{\eta_p}\tilde{\lambda}_i(t)log\tilde{\lambda}_i(t)$ with the initial condition $\tilde{\lambda}_i(0) = \lambda_i$. In matrix form this is $\mathbf{P}(t) = \sqrt{\mathbf{S}}^{exp(-t/\eta_p)}$. We compute the updated $\mathbf{F}$ as $\mathbf{F}' = \mathbf{XB}' = \mathbf{XUPU}^T$, where $\mathbf{P} = \sqrt{\mathbf{S}}^{exp(-\Delta t/\eta_p)}$.

Without our plastic recovery model, tearing regions are permanently weakened, resulting in a "brittle" effect. Setting η_p to a large value gives brittle-type fracture behavior, whereas setting η_p to a small value gives a more "sticky" tearing behavior. Refer to Figure 8 for a comparison of various η_p values.

7.4 Subgrid Geometry Removal

When a piece of foam grows excessively thin, the underlying discrete representation could, in the worst case, be a single particle wide. The physics governing these severely slender slivers can be strongly under-resolved when computed on the background grid, giving rise to artificially stiff thread-like artifacts as in Figure 8 (center). While a fully adaptive MPM treatment could ameliorate these issues, we defer the exploration of this topic to future work.

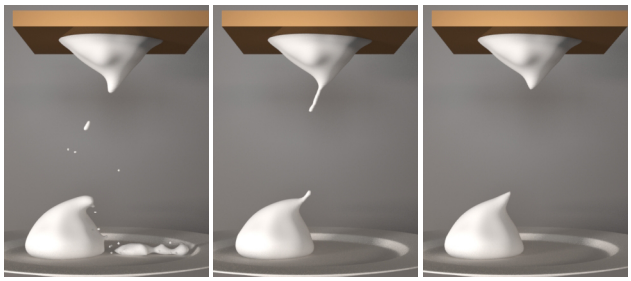

Fig. 8. "Brittle"-type fracture (left, $\eta_p = 100$) v.s. sticky tearing (center, $\eta_p = 0.35$). Subgrid geometry removal erases geometry whose physics is not resolved by the MPM background grid (right, $\eta_p = 0.35$). The mass reduction due to subgrid geometry removal is less than 0.28% over the entire simulation.

Table I. **Material Parameters**

Material	density $[kg/m^3]$	κ [kPa]	μ [kPa]	σ_Y [Pa]	η	h	σ_T	η_p
shaving cream	77.7	109.0	0.29	31.9	27.2	0.22	217.5	0.35
s'more interior	50.0	109.0	0.08	10.0	16.0	0.43	15.0	0.25
s'more exterior	50.0	109.0	50.00	1,000.0	0.1	1.00	0.3	0.50
pie	275.0	109.0	1.60	120.0	5.0	0.27	10.0	0.30
oobleck	1,000.0	109.0	11.20	0.1	10.0	2.80	1.0	0.30
viscoplastic	1,000.0	109.0	11.20	0.1	10.0	1.00	1.0	0.30

Parameters for all materials, including shaving cream (Figure 1), the interior and exterior of a toasted s'more (Figure 11), a pie (Figure 12), a shear thickening oobleck material (Figures 13, and 14), and a viscoplastic material (Figure 13) for comparison with the oobleck.

Instead, we present a simple method to remove such under-resolved particles by examining the anisotropy of the local region in a manner similar to Yu and Turk [2013]. For each weakened particle p, we identify all neighbors $\mathcal{N}(p)$ within a radius of $2h$, where h is the grid spacing. We then compute a mass-weighted mean position $\boldsymbol{p}_p$ and a mass-weighted covariance matrix $\mathbf{V}_p$ for each weakened particle p as

$$\boldsymbol{p}_p = \frac{\sum_{q\in\mathcal{N}(p)} w(\boldsymbol{x}_q - \boldsymbol{x}_p)m_q\boldsymbol{x}_q}{\sum_{q\in\mathcal{N}(p)} w(\boldsymbol{x}_q - \boldsymbol{x}_p)m_q} \tag{21}$$

and

$$\mathbf{V}_p = \frac{\sum_{q\in\mathcal{N}(p)} w(\boldsymbol{x}_q - \boldsymbol{x}_p)^2 m_q^2(\boldsymbol{x}_q - \boldsymbol{p}_p)(\boldsymbol{x}_q - \boldsymbol{p}_p)^T}{\sum_{q\in\mathcal{N}(p)} w(\boldsymbol{x}_q - \boldsymbol{x}_p)^2 m_p^2}, \tag{22}$$

where w is the weighting function employed by Stomakhin et al. [2013]. We compute the eigendecomposition of $\mathbf{V}_p$ and consider the ratio between the largest and the smallest eigenvalues, where a smaller ratio indicates a more anisotropic distribution of neighboring particles. If this ratio is beneath a threshold (10^{-5} in our tests) or the largest eigenvalue is too small (smaller than 10^{-5}), we remove the weakened particle from the simulation. Figure 8 (right) illustrates the effect of subgrid geometry removal.

8. RESULTS

We now apply our model to simulations of foam and foam-like materials, as well as other Herschel-Bulkley materials, such as "oobleck."

We list the material parameters from our simulations in Table I. The physical parameters for real foam depend on a number of factors, including freshness, subtle differences in formulation, temperature, etc., and hence can vary from product to product and from time to time. While the material density and the bulk modulus are rather

easy to set, there are a number of key parameters that dictate the resulting motion, including the shear modulus μ, the yield stress σ_Y, the Herschel-Bulkley power h, the viscosity η, the tearing yield σ_T, and the plastic relaxation η_p. We set the physical parameters (shear modulus μ, yield stress σ_Y, Herschel-Bulkley power h, viscosity η) of shaving cream based on rheological measurements from the literature [Durian et al. 1990; Ovarlez et al. 2010]. In order to achieve more compelling motion or to closely match the dynamics of a particular physical sample, we adjusted some of these parameters by at most 40% from the reported values, which is reasonable given the likelihood of differences in our physical samples from those employed in the rheology literature.

For materials whose true physical parameters are difficult to obtain, we were able to approximate any given desired behavior by tuning the six parameters according to the following procedure. We begin by tuning the parameters μ and σ_Y, which determine the behavior of the elastic regime, before tuning h and η to adjust the behavior of the plastic flow. Finally, we adjust σ_T and η_p to control the tearing. The accompanying video provides an example of this parameter tuning process.

Simulation statistics are given in Tables II and III. We timed all examples on an 8-core Intel Xeon E5-2680 v2 2.8GHz processor. Particle resampling is performed every 50 time steps. We show the amortized costs per time step in Fig. 9. At present we employ an explicit integrator for simplicity, but we note that a semi-implicit time stepping method is straightforward to implement by taking the chain rule with respect to $\mathbf{b}^e$ as derived in Stomakhin et al. [2013]. Finally, we employ a level-set based skinning method to extract surfaces from the particles [Bhattacharya et al. 2011]. Please refer to our video for the animation results.

Shaving Foam. To test the validity of our model, we attached a sample of real shaving foam to the base of a platform.

We acquired an approximation of the shape of the foam using a 3D scanner, and subsequently recorded the motion of the foam when subject to an oscillatory motion. We then simulated the acquired geometry, and compared the motion of our simulations to the recorded video. Initially, the foam oscillates and slowly begins to stretch downwards due to accumulated plasticity. As the oscillations continue, the foam exhibits typical nonlinear shear thinning behavior, and rapidly tears apart. When the foam finally collides with the ground plane, it exhibits a characteristic jiggle. As shown in the supplemental video and in Figure 1, our simulations exhibit the same characteristic behaviors of real foam noted above.

The maximum volume deviation of the skinned shaving cream model was 2.7%. Note that the density estimation (and the volume estimation) can be erroneous near the interface. We investigated the density deviation for the interior of the foam. The deviation of the average density is at most 0.89% from the initial material density of 77.7. The minimum and maximum densities were 18.0 and 93.5, respectively. 95.7% of the grid density values were within 5% of the average density.

Table II. **Simulation Statistics**

Example	#points	grid resolution	min frame	dt	subgrid geom. rem.
shaving cream	54,695- 72,681	84× 84× 84	57.5	1.0×10^{-5}	✓
uniform s'more	202,278- 203,053	88×135× 88	111.9	1.0×10^{-5}	-
toasted s'more	981,554- 988,355	140× 85×140	160.2	1.0×10^{-5}	-
pie	725,052- 726,853	225×225×225	245.7	1.0×10^{-5}	-
viscoplastic sphere	519,171- 534,871	157×157×157	303.0	0.5×10^{-5}	-
oobleck sphere	519,171- 529,365	157×157×157	303.0	0.5×10^{-5}	-
viscoplastic penguin	83,340- 87,030	209×209×209	84.3	0.5×10^{-5}	-
oobleck penguin	83,340- 85,716	209×209×209	85.0	0.5×10^{-5}	-
oobleck pachinko	0-1,615,550	389×662× 56	324.6	0.5×10^{-5}	-

Statistics for the shaving cream shake (Figure 1), the toasted s'more (Figure 11), the pie toss (Figure 12), the viscoplastic and oobleck spheres (Figures 13 top row), the viscoplastic and oobleck penguins (Figure 13 bottom row), and the oobleck penguin pachinko game (Figure 14). A frame is 1/30 of a second.

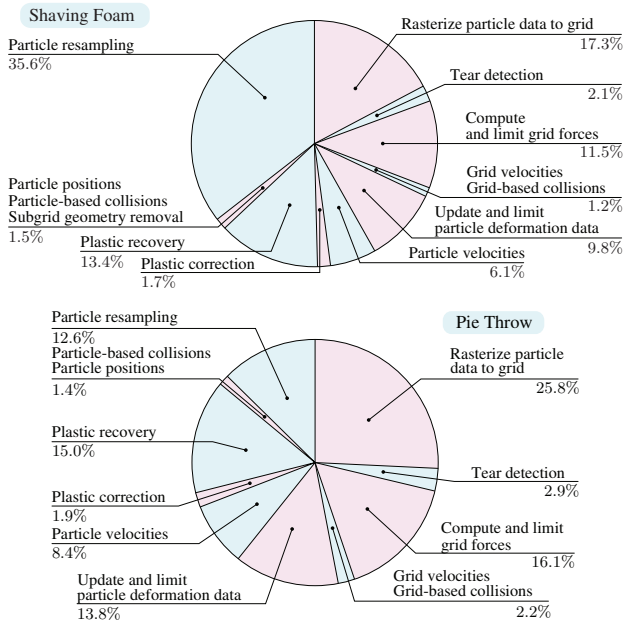

Fig. 9. Timing breakdowns for a typical step in two simulations.

Fig. 10. **S'more:** A marshmallow made from a uniform material is crushed between two graham crackers and begins to ooze through the cracks. We then lift the upper cracker, resulting in the formation of a characteristic stiff peak.

Pie Throw. In Figure 12, we throw a whipped cream pie at a face. When the pie collides with the face, the collisions induce large internal stresses, and the whipped cream flows. The pie subsequently

Table III. **Grid Statistics**

Example	cell spacing [mm]	initial #points/cell	#non-empty cells
shaving cream	3.0	8	467,788-490,813
uniform s'more	1.6	8	37,755- 53,529
toasted s'more	1.0	8	133,806-164,153
pie	1.6	8	135,628-181,445
viscoplastic sphere	1.6	8	85,455- 97,751
oobleck sphere	1.6	8	85,743- 90,629
viscoplastic penguin	1.2	8	21,559- 25,295
oobleck penguin	1.2	8	21,167- 23,681
oobleck pachinko	1.8	8	0-461,329

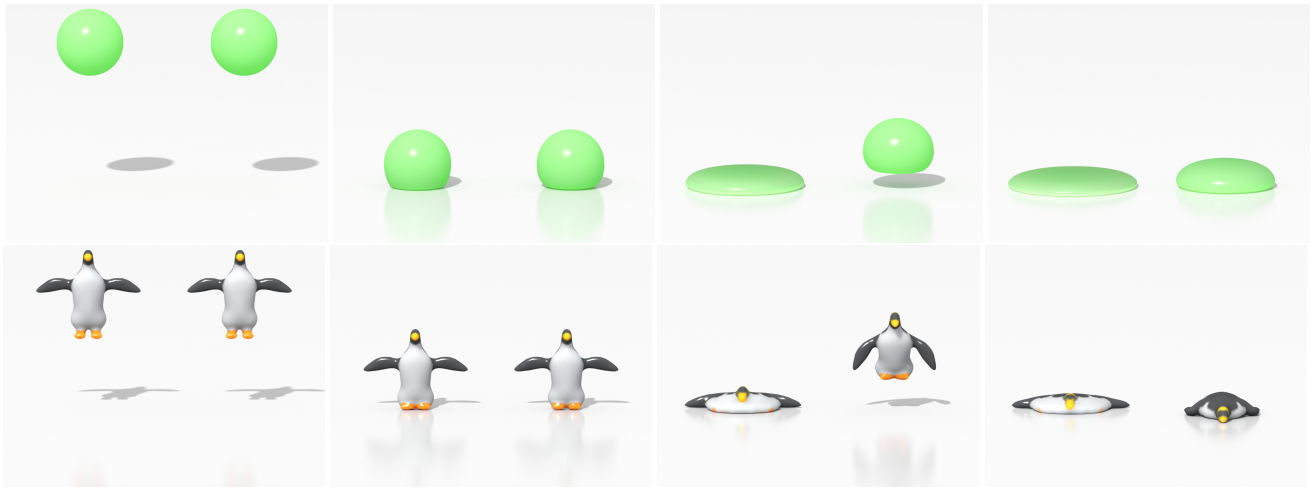

Fig. 13. **Side by side comparisons** between viscoplastic materials (left object in each pair) and shear thickening (oobleck) materials (right object in each pair). While the viscoplastic materials flow immediately after impact with the ground plane, the shear thickening materials rebound elastically after the fast initial impact, and only flow after the second, lower-velocity impact.

Fig. 11. **Multi-material s'more:** A toasted marshmallow composed of a crunchy exterior and a gooey interior is crushed between two graham crackers. The stiff outer layer cracks, allowing the softer interior to flow out.

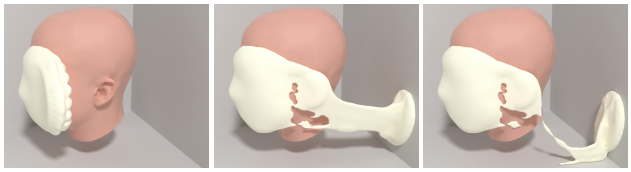

Fig. 12. **Pie throw:** A whipped cream pie is thrown at a face and exhibits the characteristic clumping and tearing behavior of such a system.

accumulates plasticity, stretches, and tears. Afterwards, chunks of whipped cream adhere to the face.

S'more. In Figure 10, we simulate a s'more by squeezing a marshmallow between two graham crackers and a block of chocolate. The marshmallow is modeled as a "moderately warm" foam, that is, as a soft material (we do not simulate the heat transfer). As we compress the marshmallow, it begins to deform and flow, eventually seeping through the holes in the graham cracker. As we subsequently lift the upper cracker, the marshmallow forms a thin neck and eventually tears, leaving a characteristic stiff peak.

In Figure 11 we simulate a more complex marshmallow. We model the marshmallow using two layers of MPM particles; this allows us to mimic a toasted marshmallow, where a stiff outer layer approximates the crispy exterior, and a soft inner layer approximates the molten marshmallow core (again, we do not simulate the heat transfer). After a critical amount of compression, the outer layer cracks, allowing the softer inner layer to flow outwards.

Oobleck. Inspired by Bargteil et al. [2007], we also explore the application of our method to shear thickening materials such as "oobleck," a solution of cornstarch in water. In Figure 13, we show a side by side comparison between viscoplastic and shear thickening materials. In this comparison, we only modify the Herschel-Bulkley power parameter h; other parameters are the same for both materials. We drop both materials onto a ground plane; upon impact they experience a large stress. At large stresses, the flow rate of the shear thickening material is low, and it behaves like an elastic material and bounces off the ground. In contrast, the flow rate of the viscoplastic material is much higher and it flows immediately.

Fig. 14. **Oobleck pachinko:** Penguins composed of shear thickening oobleck are dropped onto a set of pachinko pins. The penguins bounce after initial high-stress collisions, but later experience substantial plastic flow after low-stress collisions.

In Figure 14, we use the same oobleck material to simulate a pachinko machine. Due to shear thickening, the rapidly falling penguins initially bounce off of the pachinko pins, with only an occasional wing or foot being ripped away. As the penguins slow down due to additional collisions with the increasingly tightly spaced pins, the resulting gentle collisions induce significantly lower stresses, and the viscoplastic penguins flow and lose their shape.

9. CONCLUSIONS AND FUTURE WORK

We have presented a method to simulate dense foams and other complex materials that exhibit nonlinear, viscoplastic behaviors. In particular, we adapted the highly flexible Herschel-Bulkley constitutive model to the similarly powerful MPM framework in order to realistically simulate such materials. To robustly treat large shearing effects characteristic of dense foams, we developed a particle resampling technique for MPM to prevent the formation of nonphysical voids. Finally, we extended MPM to treat tearing effects and to avoid the generation of artificial thin strands.

While we successfully simulated dense foams and validated our model against experimental results, there remain several opportunities for future research. First, it would be interesting to consider additional types of foams. Our method simulates dense foams as a continuum, homogenizing over a vast number of small bubbles in order to conduct simulations at macroscopic scales. For less dense foams where some individual bubbles are distinguishable (e.g., soap foams in a bathtub) hybrid continuum-discrete strategies for both simulation and rendering may be required, in order to achieve the appearance of high density bubbles without extreme computational costs.

Many real foams also exhibit additional phenomena such as coarsening and drainage, which modify the observed bulk properties of the material. Although the time scales of these phenomena are often much longer than our simulated time scale, it would be interesting to incorporate such time-dependent material changes.

Finally, since a systematic rheological understanding of many foam phenomena is still lacking, we hope that the simulation tools we have developed may aid in studying these phenomena from a numerical perspective.

Acknowledgements

We thank Keenan Crane for providing his digital and real face for the pie throw example, as well as Henrique Teles Maia, Nora Wixom and Michelle Ming-Yen Lee for helping to prepare the final version of this paper and the supplementary video. We thank Intel for donating computing hardware, The Foundry for donating MODO licenses, and Adobe for donating Creative Cloud licenses.

This work was supported in part by the JSPS Postdoctoral Fellowships for Research Abroad, NSF (Grants IIS-13-19483, CMMI-11-29917, CAREER-1453101), NSERC (Grant RGPIN-04360-2014), Intel, The Walt Disney Company, Autodesk, Side Effects, NVIDIA, Adobe, and The Foundry.

REFERENCES

ADAMS, B., PAULY, M., KEISER, R., AND GUIBAS, L. J. 2007. Adaptively Sampled Particle Fluids. *ACM Trans. Graph. 26,* 3 (jul), 48:1–48:7.

ANDO, R., THÜREY, N., AND TSURUNO, R. 2012. Preserving Fluid Sheets with Adaptively Sampled Anisotropic Particles. *IEEE Transactions on Visualization and Computer Graphics 18,* 8 (aug), 1202–1214.

BANERJEE, B. 2004. Material Point Method Simulations of Fragmenting Cylinders. In *Proc. of 17th ASCE Engineering Mechanics Conference*.

BARGTEIL, A. W., WOJTAN, C., HODGINS, J. K., AND TURK, G. 2007. A Finite Element Method for Animating Large Viscoplastic Flow. *ACM Trans. Graph. 26,* 3 (jul), 16:1–16:8.

BHATTACHARYA, H., GAO, Y., AND BARGTEIL, A. 2011. A Level-set Method for Skinning Animated Particle Data. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2011*. 17–24.

BINGHAM, E. C. 1922. *Fluidity and Plasticity*. McGraw-Hill, 219.

BRACKBILL, J. U. AND RUPPEL, H. M. 1986. FLIP: A Method for Adaptively Zoned, Particle-in-Cell Calculations of Fluid Flows in Two Dimensions. *J. Comp. Phys. 65,* 2, 314–343.

BRAKKE, K. A. 1992. The Surface Evolver. *Experimental Mathematics 1,* 2, 141–165.

BUSARYEV, O., DEY, T. K., WANG, H., AND REN, Z. 2012. Animating Bubble Interactions in a Liquid Foam. *ACM Trans. Graph. 31,* 4 (jul), 63:1–63:8.

CLEARY, P. W., PYO, S. H., PRAKASH, M., AND KOO, B. K. 2007. Bubbling and Frothing Liquids. *ACM Trans. Graph. 26,* 3 (jul), 97:1–97:6.

COHEN-ADDAD, S., HÖHLER, R., AND PITOIS, O. 2013. Flow in Foams and Flowing Foams. *Annual Review of Fluid Mechanics 45*, 241–267.

COOK, R. L. 1986. Stochastic Sampling in Computer Graphics. *ACM Trans. Graph. 5,* 1 (jan), 51–72.

COTTET, G.-H. AND KOUMOUTSAKOS, P. D. 2000. *Vortex Methods: Theory and Practice*. Cambridge University Press.

DURIAN, D. J., WEITZ, D. A., AND PINE, D. J. 1990. Dynamics and Coarsening in Three-Dimensional Foams. *Journal of Physics: Condensed Matter 2,* Supplement, SA433.

ĎURIKOVIČ, R. 2001. Animation of Soap Bubble Dynamics, Cluster Formation and Collision. *Computer Graphics Forum 20,* 3 (sep), 67–76.

EBEIDA, M. S., MITCHELL, S. A., PATNEY, A., DAVIDSON, A. A., AND OWENS, J. D. 2012. A Simple Algorithm for Maximal Poisson-Disk Sampling in High Dimensions. *Computer Graphics Forum 31,* 2 (may), 785–794.

EDWARDS, E. AND BRIDSON, R. 2012. A High-Order Accurate Particle-in-Cell Method. *Int. J. Numer. Meth. Engng. 90,* 9 (jun), 1073–1088.

ENRIGHT, D., FEDKIW, R., FERZIGER, J., AND MITCHELL, I. 2002. A Hybrid Particle Level Set Method for Improved Interface Capturing. *J. Comp. Phys. 183,* 1 (nov), 83–116.

GARRETT, P. 1993. Recent developments in the understanding of foam generation and stability. *Chemical Engineering Science 48,* 2, 367 – 392.

GERSZEWSKI, D., BHATTACHARYA, H., AND BARGTEIL, A. W. 2009. A Point-Based Method for Animating Elastoplastic Solids. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2009*. 133–138.

GOKTEKIN, T. G., BARGTEIL, A. W., AND O'BRIEN, J. F. 2004. A Method for Animating Viscoelastic Fluids. *ACM Trans. Graph. 23,* 3 (aug), 463–468.

GREENWOOD, S. T. AND HOUSE, D. H. 2004. Better with Bubbles: Enhancing the Visual Realism of Simulated Fluid. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2004*. 287–296.

GUO, Y. J. AND NAIRN, J. A. 2006. Three-Dimensional Dynamic Fracture Analysis Using the Material Point Method. *Computer Modeling in Engineering & Sciences 1,* 1, 11–25.

HARDER, A. AND MANGNALL, C. 2013. Bubbles and Foam in Partysaurus Rex. In *ACM SIGGRAPH 2013 Talks*. 17:1–17:1.

HERSCHEL, W. H. AND BULKLEY, R. 1926. Konsistenzmessungen von Gummi-Benzollösungen. *Kolloid-Zeitschrift & Zeitschrift für Polymere 39,* 4 (aug), 291–300.

HIEBER, S. E. AND KOUMOUTSAKOS, P. 2005. A Lagrangian Particle Level Set Method. *J. Comp. Phys. 210,* 1 (nov), 342 – 367.

HONG, J.-M. AND KIM, C.-H. 2005. Discontinuous Fluids. *ACM Trans. Graph. 24,* 3 (jul), 915–920.

HONG, J.-M., LEE, H.-Y., YOON, J.-C., AND KIM, C.-H. 2008. Bubbles Alive. *ACM Trans. Graph. 27,* 3 (aug), 48:1–48:4.

IHMSEN, M., AKINCI, N., AKINCI, G., AND TESCHNER, M. 2012. Unified Spray, Foam and Air Bubbles for Particle-Based Fluids. *The Visual Computer 28,* 6-8 (jun), 669–677.

IRVING, G., TERAN, J., AND FEDKIW, R. 2004. Invertible Finite Elements for Robust Simulation of Large Deformation. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2004*. 131–140.

JONES, B., WARD, S., JALLEPALLI, A., PERENIA, J., AND BARGTEIL, A. W. 2014. Deformation Embedding for Point-Based Elastoplastic Simulation. *ACM Trans. Graph. 33,* 2 (mar), 21:1–21:9.

KEISER, R., ADAMS, B., GASSER, D., BAZZI, P., DUTRÉ, P., AND GROSS, M. 2005. A Unified Lagrangian Approach to Solid-Fluid Animation. In *Proc. of the Second Eurographics/IEEE VGTC Conference on Point-Based Graphics 2005*. 125–133.

KHAREVYCH, L., MULLEN, P., OWHADI, H., AND DESBRUN, M. 2009. Numerical Coarsening of Inhomogeneous Elastic Materials. *ACM Trans. Graph. 28,* 3 (aug), 51:1–51:8.

KIM, B., LIU, Y., LLAMAS, I., JIAO, X., AND ROSSIGNAC, J. 2007. Simulation of Bubbles in Foam with the Volume Control Method. *ACM Trans. Graph. 26,* 3 (jul), 98:1–98:10.

KIM, D., SONG, O.-Y., AND KO, H.-S. 2010. A Practical Simulation of Dispersed Bubble Flow. *ACM Trans. Graph. 29,* 4 (jul), 70:1–70:5.

KOEHLER, S. A., STONE, H. A., BRENNER, M. P., AND EGGERS, J. 1998. Dynamics of foam drainage. *Phys. Rev. E 58*, 2097–2106.

KÜCK, H., VOGELGSANG, C., AND GREINER, G. 2002. Simulation and Rendering of Liquid Foams. In *Proc. of Graphics Interface 2002*. 81–88.

LOSASSO, F., SHINAR, T., SELLE, A., AND FEDKIW, R. 2006. Multiple Interacting Liquids. *ACM Trans. Graph. 25,* 3 (jul), 812–819.

LOSASSO, F., TALTON, J. O., KWATRA, N., AND FEDKIW, R. 2008. Two-Way Coupled SPH and Particle Level Set Fluid Simulation. *IEEE Transactions on Visualization and Computer Graphics 14,* 4 (jul-aug), 797–804.

MÜLLER, M., KEISER, R., NEALEN, A., PAULY, M., GROSS, M., AND ALEXA, M. 2004. Point Based Animation of Elastic, Plastic and Melting Objects. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2004*. 141–151.

MÜLLER, M., SOLENTHALER, B., KEISER, R., AND GROSS, M. 2005. Particle-Based Fluid-Fluid Interaction. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2005*. 237–244.

NARAIN, R., GOLAS, A., AND LIN, M. C. 2010. Free-Flowing Granular Materials with Two-Way Solid Coupling. *ACM Trans. Graph. 29,* 6 (dec), 173:1–173:9.

NESME, M., KRY, P. G., JEŘÁBKOVÁ, L., AND FAURE, F. 2009. Preserving Topology and Elasticity for Embedded Deformable Models. *ACM Trans. Graph. 28,* 3 (aug), 52:1–52:9.

NIELSEN, M. B. AND ØSTERBY, O. 2013. A Two-Continua Approach to Eulerian Simulation of Water Spray. *ACM Trans. Graph. 32,* 4 (jul), 67:1–67:10.

O'BRIEN, J. F., BARGTEIL, A. W., AND HODGINS, J. K. 2002. Graphical Modeling and Animation of Ductile Fracture. *ACM Trans. Graph. 21,* 3 (jul), 291–294.

OVARLEZ, G., KRISHAN, K., AND COHEN-ADDAD, S. 2010. Investigation of shear banding in three-dimensional foams. *EPL (Europhysics Letters) 91,* 6, 68005:1–68005:6.

PATKAR, S., AANJANEYA, M., KARPMAN, D., AND FEDKIW, R. 2013. A Hybrid Lagrangian-Eulerian Formulation for Bubble Generation and Dynamics. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2013*. 105–114.

SAYE, R. I. AND SETHIAN, J. A. 2013. Multiscale Modeling of Membrane Rearrangement, Drainage, and Rupture in Evolving Foams. *Science 340,* 6133 (may), 720–724.

SCHECHTER, H. AND BRIDSON, R. 2012. Ghost SPH for Animating Water. *ACM Trans. Graph. 31,* 4 (jul), 61:1–61:8.

SCHREYER, H. L., SULSKY, D. L., AND ZHOU, S.-J. 2002. Modeling Delamination as a Strong Discontinuity with the Material Point Method. *Computer Methods in Applied Mechanics and Engineering 191,* 23–24 (mar), 2483 – 2507.

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

SIMO, J. C. 1988. A Framework for Finite Strain Elastoplasticity Based on Maximum Plastic Dissipation and the Multiplicative Decomposition: Part I. Continuum Formulation. *Computer Methods in Applied Mechanics and Engineering 66,* 2 (feb), 199–219.

SIMO, J. C. AND HUGHES, T. J. R. 1998. *Computational Inelasticity*. Springer.

SOLENTHALER, B. AND PAJAROLA, R. 2008. Density Contrast SPH Interfaces. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2008*. 211–218.

SOLENTHALER, B., SCHLÄFLI, J., AND PAJAROLA, R. 2007. A Unified Particle Model for Fluid-Solid Interactions. *Computer Animation and Virtual Worlds 18,* 1 (feb), 69–82.

STEFFEN, M., KIRBY, R. M., AND BERZINS, M. 2008. Analysis and Reduction of Quadratire Errors in the Material Point Method (MPM). *Int. J. Numer. Meth. Engng. 76,* 6 (nov), 922–948.

STOMAKHIN, A., SCHROEDER, C., CHAI, L., TERAN, J., AND SELLE, A. 2013. A Material Point Method for Snow Simulation. *ACM Trans. Graph. 32,* 4 (jul), 102:1–102:9.

STOMAKHIN, A., SCHROEDER, C., JIANG, C., CHAI, L., TERAN, J., AND SELLE, A. 2014. Augmented MPM for Phase-Change and Varied Materials. *ACM Trans. Graph. 33,* 4 (jul), 138:1–138:11.

SULSKY, D. AND SCHREYER, L. 2004. MPM Simulation of Dynamic Material Failure with a Decohesion Constitutive Model. *European Journal of Mechanics - A/Solids 23,* 3 (may-jun), 423 – 445.

SULSKY, D., ZHOU, S.-J., AND SCHREYER, H. L. 1995. Application of a Particle-in-Cell Method to Solid Mechanics. *Computer Physics Communications 87,* 1-2 (may), 236–252.

TERZOPOULOS, D. AND FLEISCHER, K. 1988. Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture. *Computer Graphics 22,* 4 (aug), 269–278.

THÜREY, N., SADLO, F., SCHIRM, S., MÜLLER-FISCHER, M., AND GROSS, M. 2007. Real-time Simulations of Bubbles and Foam within a Shallow Water Framework. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2007*. 191–198.

WANG, H., O'BRIEN, J., AND RAMAMOORTHI, R. 2010. Multi-Resolution Isotropic Strain Limiting. *ACM Trans. Graph. 29,* 6 (dec), 156:1–156:10.

WEAIRE, D. 2008. The Rheology of Foam. *Current Opinion in Colloid & Interface Science 13,* 3, 171–176.

WEAIRE, D. AND HUTZLER, S. 2001. *The Physics of Foams*. Oxford University Press.

WICKE, M., RITCHIE, D., KLINGNER, B. M., BURKE, S., SHEWCHUK, J. R., AND O'BRIEN, J. F. 2010. Dynamic Local Remeshing for Elastoplastic Simulation. *ACM Trans. Graph. 29,* 4 (jul), 49:1–49:11.

WOJTAN, C. AND TURK, G. 2008. Fast Viscoelastic Behavior with Thin Features. *ACM Trans. Graph. 27,* 3 (aug), 47:1–47:8.

YU, J. AND TURK, G. 2013. Reconstructing Surfaces of Particle-Based Fluids Using Anisotropic Kernels. *ACM Trans. Graph. 32,* 1 (jan), 5:1–5:12.

ZHENG, W., YONG, J.-H., AND PAUL, J.-C. 2006. Simulation of Bubbles. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2006*. 325–333.

ZHOU, S. 1998. The Numerical Prediction of Material Failure Based on the Material Point Method. Ph.D. thesis, The University of New Mexico.

ZHU, Y. AND BRIDSON, R. 2005. Animating Sand as a Fluid. *ACM Trans. Graph. 24,* 3 (jul), 965–972.

APPENDIX

A. THE EFFECTIVE VISCOSITY IN THE MODEL OF BARGTEIL ET AL.

Bargteil et al. [2007] proposed the function

$$\gamma = \min\left(\nu\frac{\|\mathbf{P}\|_F - P_y}{\|\mathbf{P}\|_F}, 1\right),$$

where $\mathbf{P}$ gives the stress, P_y gives the yield stress, and ν gives the flow rate. We have omitted work hardening and work softening for simplicity. The effective viscosity is thus $\|\mathbf{P}\|_F/\nu$, an increasing function of the stress.

B. DERIVATION OF EQ.(2)

From (1), we have

$$\boldsymbol{\tau} = \frac{\partial W}{\partial \mathbf{F}^e}\mathbf{F}^{eT} = \left(\frac{\partial W_v(J)}{\partial \mathbf{F}^e} + \frac{\partial W_s(\overline{\mathbf{b}}^e)}{\partial \mathbf{F}^e}\right)\mathbf{F}^{eT}. \tag{23}$$

Next, using the chain rule, we obtain

$$\begin{aligned}\frac{\partial W_v(J)}{\partial \mathbf{F}^e} &= \frac{\partial W_v(J)}{\partial J}\frac{\partial J}{\partial \mathbf{F}^e} = \frac{1}{2}\kappa(J-\frac{1}{J})\frac{\partial \det(\mathbf{F}^e)}{\partial \mathbf{F}^e}\\ &= \frac{1}{2}\kappa(J-\frac{1}{J})\det(\mathbf{F}^e)\mathbf{F}^{e-T} = \frac{1}{2}\kappa(J^2-1)\mathbf{F}^{e-T}.\end{aligned} \tag{24}$$

Likewise,

$$\frac{\partial W_s(\overline{\mathbf{b}}^e)}{\partial \mathbf{F}^e} = \frac{\partial W_s(\overline{\mathbf{b}}^e)}{\partial \operatorname{Tr}[\overline{\mathbf{b}}^e]}\frac{\partial \operatorname{Tr}[\overline{\mathbf{b}}^e]}{\partial \overline{\mathbf{b}}^e} : \frac{\partial \overline{\mathbf{b}}^e}{\partial \mathbf{F}^e}, \tag{25}$$

where we have

$$\frac{\partial W_s(\overline{\mathbf{b}}^e)}{\partial \operatorname{Tr}[\overline{\mathbf{b}}^e]}\frac{\partial \operatorname{Tr}[\overline{\mathbf{b}}^e]}{\partial \overline{\mathbf{b}}^e} = \frac{1}{2}\mu\mathbf{I} \tag{26}$$

and

$$\begin{aligned}\frac{\partial \overline{\mathbf{b}}^e{}_{ij}}{\partial \mathbf{F}^e{}_{uv}} &= J^{-2/3}\frac{\partial \mathbf{b}^e{}_{ij}}{\partial \mathbf{F}^e{}_{uv}} + \mathbf{b}^e{}_{ij}\frac{\partial J^{-2/3}}{\partial \mathbf{F}^e{}_{uv}}\\ &= J^{-2/3}(\mathbf{F}^e{}_{iv}\delta_{ju} + \mathbf{F}^e{}_{jv}\delta_{iu}) - \frac{2}{3}J^{-2/3}\mathbf{b}^e{}_{ij}\mathbf{F}^{e-T}_{uv}.\end{aligned} \tag{27}$$

Substituting (26) and (27) into (25), we obtain

$$\frac{\partial W_s(\overline{\mathbf{b}}^e)}{\partial \mathbf{F}^e} = \mu J^{-2/3}\left(\mathbf{F}^e - \frac{1}{3}\operatorname{Tr}[\mathbf{b}^e]\mathbf{F}^{e-T}\right). \tag{28}$$

Finally, substituting (24) and (28) into (23), we obtain (2).

C. DERIVATION OF EQ.(13)

Discretizing Eq. (10), we obtain

$$\mathbf{b}^{e,\text{pre}}_{n+1} = \mathbf{b}^e_n + \Delta t(\mathbf{L}_n\mathbf{b}^e_n + \mathbf{b}^e_n\mathbf{L}^T_n).$$

We can add a higher order term $\Delta t^2(\mathbf{L}_n\mathbf{b}^e_n\mathbf{L}^T_n)$ to the right hand side, which vanishes as $\Delta t \to 0$. We then obtain

$$\begin{aligned}\mathbf{b}^{e,\text{pre}}_{n+1} &= \mathbf{b}^e_n + \Delta t(\mathbf{L}_n\mathbf{b}^e_n + \mathbf{b}^e_n\mathbf{L}^T_n) + \Delta t^2(\mathbf{L}_n\mathbf{b}^e_n\mathbf{L}^T_n)\\ &= (I + \Delta t\mathbf{L}_n)\mathbf{b}^e_n(I + \Delta t\mathbf{L}^T_n) = \mathbf{f}_{n+1}\mathbf{b}^e_n\mathbf{f}_{n+1}.\end{aligned}$$

Normalizing both sides, we obtain Eq. (13).

D. DERIVATION OF EQ. (15)

Taking the trace of both sides of Eq. (14) and noting that $\operatorname{Tr}[\hat{\mathbf{s}}_{n+1}] = 0$, we find that

$$\operatorname{Tr}[\overline{\mathbf{b}}^e_{n+1}] - \operatorname{Tr}[\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}] = 0.$$

Recalling that $\mathbf{s}^{\text{pre}}_{n+1} = \mu\operatorname{dev}[\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}]$, we have $\overline{\mathbf{b}}^{e,\text{pre}}_{n+1} = \frac{1}{\mu}\mathbf{s}^{\text{pre}}_{n+1} + \frac{1}{3}\operatorname{Tr}[\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}]\mathbf{I}$. Likewise, $\overline{\mathbf{b}}^e_{n+1} = \frac{1}{\mu}\mathbf{s}_{n+1} + \frac{1}{3}\operatorname{Tr}[\overline{\mathbf{b}}^e_{n+1}]\mathbf{I} = \frac{1}{\mu}\mathbf{s}_{n+1} + \frac{1}{3}\operatorname{Tr}[\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}]\mathbf{I}$. Thus, Eq. (14) becomes

$$\mathbf{s}_{n+1} - \mathbf{s}^{\text{pre}}_{n+1} = -2\tilde{\mu}\Delta t\gamma(s_{n+1})\hat{\mathbf{s}}_{n+1}.$$

Equivalently, we have

$$s_{n+1}\hat{\mathbf{s}}_{n+1} - s^{\text{pre}}_{n+1}\hat{\mathbf{s}}^{\text{pre}}_{n+1} = -2\tilde{\mu}\Delta t\gamma(s_{n+1})\hat{\mathbf{s}}_{n+1},$$

which reveals that $\hat{\mathbf{s}}_{n+1} = \hat{\mathbf{s}}^{\text{pre}}_{n+1}$. That is, $\mathbf{s}$ changes in *magnitude* but not in direction. It thus suffices to solve the scalar equation (15).

E. ALGORITHM REFERENCE

E.1 MPM Discretization

As in [Stomakhin et al. 2013], we use a regular grid as the background Eulerian mesh for the computation of stress-induced forces, which requires the spatial discretization of $\nabla\cdot\boldsymbol{\sigma}$. We fix the grid position and the grid spacing throughout the simulation to avoid remeshing and to simplify computations.

We use cubic B-splines [Steffen et al. 2008; Stomakhin et al. 2013] for the grid basis functions $N^h_{\boldsymbol{i}}(\boldsymbol{x}_p)$:

$$N^h_{\boldsymbol{i}}(\boldsymbol{x}_p) = N(\frac{1}{h}(x_p - ih))N(\frac{1}{h}(y_p - jh))N(\frac{1}{h}(z_p - kh))$$

Here $\boldsymbol{i} = (i,j,k)$ is the grid index, $\boldsymbol{x}_p = (x_p, y_p, z_p)$ is the evaluation position (or the particle position), h is the grid spacing, and

$$N(x) = \begin{cases}\frac{1}{2}|x|^3 - x^2 + \frac{2}{3}, & 0 \le |x| < 1\\ -\frac{1}{6}|x|^3 + x^2 - 2|x| + \frac{4}{3}, & 1 \le |x| < 2\\ 0, & \text{otherwise}\end{cases}.$$

For compact notation, let $w_{\boldsymbol{ip}} = N^h_{\boldsymbol{i}}(\boldsymbol{x}_p)$ and $\nabla w_{\boldsymbol{ip}} = \nabla N^h_{\boldsymbol{i}}(\boldsymbol{x}_p)$ as in [Stomakhin et al. 2013].

Physical quantities along the flow are tracked with Lagrangian particles, which are also useful for tracking history dependent quantities required for the computation of plasticity. Such physical quantities include the position $\boldsymbol{x}$, the velocity $\boldsymbol{v}$, the mass m, the total deformation gradient $\mathbf{F}$, and the normalized strain $\overline{\mathbf{b}}^e$. A complete list of the quantities stored for each particle can be found in §E.4.

E.2 Initial Set Up

As in [Stomakhin et al. 2013], we compute each particle's volume and density once during the first time step. A particle's density and volume are estimated as $\rho_p = \sum_{\boldsymbol{i}} w_{\boldsymbol{ip}}m_{\boldsymbol{i}}/h^3$ and $V_p = m_p/\rho_p$, respectively.

E.3 Per Step Computation

To update the physical quantities at time step $n+1$ given those at time step n, we perform the following steps in order. For brevity, we drop the subscript $n+1$ and n when it is clear from the context.

1. Rasterize particle data to the grid. We transfer mass from the particles to the grid via $m_i = \sum_p w_{ip} m_p$, where m_p is the mass of particle p. Likewise, we transfer momentum via $m_i \boldsymbol{v}_i = \sum_p w_{ip} m_p \boldsymbol{v}_p$, where $\boldsymbol{v}_p$ is the velocity of particle p. This yields a grid velocity of $\boldsymbol{v}_i = \sum_p w_{ip} m_p \boldsymbol{v}_p / m_i$.

2. Detect tearing regions. For each particle p, we compute $\| \operatorname{dev}[\mathbf{C}_p^p] \|_F = \| \operatorname{dev}[\mathbf{F}^T (\mathbf{b}^e)^{-1} \mathbf{F}] \|_F$. We label p as *weak* only if $\| \operatorname{dev}[\mathbf{C}_p^p] \|_F > \sigma_T$.

3. Compute grid forces. We compute the internal forces from the stress ($\nabla \cdot \boldsymbol{\sigma}$) at each grid node $\boldsymbol{i}$ via

$$\boldsymbol{f}_i = -\sum_p V_p J_p \boldsymbol{\sigma}_p \nabla w_{ip} \ ,$$

where V_p is the initial particle volume and $J_p = \det(\mathbf{F}_p)$ is the determinant of the particle total deformation gradient $\mathbf{F}_p$. $\boldsymbol{\sigma}_p$ is computed from the constitutive relation via

$$\boldsymbol{\sigma}_p = \frac{1}{J_p} \left[\frac{\kappa}{2} (J_p^2 - 1) \mathbf{I} + \mu \operatorname{dev}[\overline{\mathbf{b}}_p^e] \right] \ ,$$

where $\overline{\mathbf{b}}_p^e$ is the normalized strain of particle p.

4. Limit forces. If a particle is labeled as weak in step 3, we modify its stress $\boldsymbol{\sigma}_p$ by first computing the eigendecomposition $\boldsymbol{\sigma}_p = \mathbf{V}_p \boldsymbol{\Sigma}_p \mathbf{V}_p^T$, where $\mathbf{V}_p$ is a unitary matrix and $\boldsymbol{\Sigma}_p = diag(s_1, s_2, s_3)$ is a diagonal matrix. Next, we replace each eigenvalue s_i by $\min(0, s_i)$, and reassemble $\boldsymbol{\sigma}_p$.

5. Update grid velocities. We set the grid velocities $\boldsymbol{v}^\star$ to

$$\boldsymbol{v}_i^\star = \Delta t \boldsymbol{f}_i / m_i + \Delta t \boldsymbol{g} \ .$$

6. Resolve grid-based collisions. To model adhesive obstacles, we set the relative velocity between a grid node and an obstacle to 0 if the grid node is inside an obstacle.

We model non-adhesive obstacles as in [Stomakhin et al. 2013]-§8.

7. Update particle deformation data. We compute the incremental deformation gradient via $\mathbf{f}_{p,n+1} = (\mathbf{I} + \Delta t \nabla \boldsymbol{v}_{p,n})$, where $\nabla \boldsymbol{v}_{p,n}$ is given by $\nabla \boldsymbol{v}_{p,n} = \sum_i \boldsymbol{v}_{i,n} (\nabla w_{ip})^T$.

We compute the predicted strain via $\overline{\mathbf{b}}_{p,n+1}^{e,\text{pre}} = \bar{\mathbf{f}}_{p,n+1} \overline{\mathbf{b}}_{p,n}^e \bar{\mathbf{f}}_{p,n+1}^T$. Likewise, we update $\mathbf{F}_{p,n+1}$ to $\mathbf{F}_{p,n+1} = \mathbf{f}_{p,n+1} \mathbf{F}_{p,n}$.

8. Limit deformation gradient. Based on the results of step 7, if a particle is weak and if the shear component is increasing, we disable the shear deformation by modifying $\mathbf{f}$. In addition, if the volume continues to expand, we disable the expansion.

To detect accumulation, we compute the eigendecompositions of $\overline{\mathbf{b}}_{p,n}^e$ and $\overline{\mathbf{b}}_{p,n+1}^{e,\text{pre}}$, and compare their maximum eigenvalues λ_n and $\lambda_{n+1}^{\text{pre}}$. If $\lambda_{n+1}^{\text{pre}} \leq \lambda_n$, we set $\mathbf{f}_{n+1}^{corr} = \mathbf{f}_{n+1}$. Otherwise, the material is accumulating shear, and we compute $J_f = \det(\mathbf{f}_{n+1})$. We compute the singular value decomposition of $\bar{\mathbf{f}}_{n+1} = J_f^{-1/3} \mathbf{f}_{n+1}$, setting its singular values to 1 to obtain a corrected version, $\bar{\mathbf{f}}_{n+1}^{corr}$. We then rescale again to obtain $\mathbf{f}_{n+1}^{corr} = J_f^{1/3} \bar{\mathbf{f}}_{n+1}^{corr}$.

To detect continued volume expansion, we compute $J_n = \det(\mathbf{F}_n)$ and $J_f^{corr} = \det(\mathbf{f}_{n+1}^{corr})$. If $J_n > 1$ and $J_f^{corr} > 1$, the material was previously expanding and continues to expand. In this situation, we disable this additional expansion by normalizing $\mathbf{f}_{n+1}^{corr}$.

Finally, we update the predicted strain to $\overline{\mathbf{b}}_{p,n+1}^{e,\text{pre}} = \mathbf{f}_{n+1}^{corr} \overline{\mathbf{b}}_{p,n}^e \mathbf{f}_{n+1}^{corr\,T}$.

9. Update particle velocities. Following [Stomakhin et al. 2013], we compute new particle velocities as $\boldsymbol{v}_{p,n+1} = (1 - \alpha) \boldsymbol{v}_{\text{PIC}p} + \alpha \boldsymbol{v}_{\text{FLIP}p}$, where $\boldsymbol{v}_{\text{PIC}_p} = \sum_i w_{ip} \boldsymbol{v}_i^\star$ and $\boldsymbol{v}_{\text{FLIP}_p} = \boldsymbol{v}_{p,n} + \sum_i w_{ip} (\boldsymbol{v}_i^\star - \boldsymbol{v}_i)$. We employ $\alpha = 0.95$.

10. Compute plastic flow. We compute $s_{n+1}^{\text{pre}} = \mu \| \operatorname{dev}[\overline{\mathbf{b}}_{n+1}^{e,\text{pre}}] \|_F$. If the yield condition is violated, i.e., if $\Phi(s_{n+1}^{\text{pre}}) > 0$, we compute the plastic flow by solving for s_{n+1} in (16). When $\eta = 0$ or $h = 1$, an explicit solve is possible, with s_{n+1} given by (17). Otherwise, we solve Eq. (16) numerically via bisection starting from the interval $\left[\sqrt{\frac{2}{3}} \sigma_Y, s_{n+1}^{\text{pre}} \right]$.

Next, we complete the correction by updating $\mathbf{s}_{n+1} = s_{n+1} \hat{\mathbf{s}}_{n+1}^{\text{pre}}$ and by updating the volumetric left Cauchy-Green tensor via (18). Finally, we renormalize $\overline{\mathbf{b}}_{n+1}^e$.

11. Compute plastic recovery. We first compute $\mathbf{C}^p = \mathbf{F}^T (\mathbf{b}^e)^{-1} \mathbf{F}$. Next, we compute the eigendecomposition $\mathbf{C}^p = \mathbf{U} \mathbf{S} \mathbf{U}^T$ to obtain $\sqrt{\mathbf{C}^p}$ as $\mathbf{U} \sqrt{\mathbf{S}} \mathbf{U}^T$.

Afterwards, we let $\lambda_i = (\sqrt{\mathbf{S}})_{ii}$ and compute $\tilde{\lambda}_i = \lambda_i^{exp(-\Delta t / \eta_p)}$ to obtain $\mathbf{R} = diag(\tilde{\lambda}_i / \lambda_i)$.

Finally, we update the deformation gradient to $\mathbf{F}' = \mathbf{F} \mathbf{U} \mathbf{R} \mathbf{U}^T$.

12. Remove subgrid geometry. For each weakened particle p, we identify all neighbors $\mathcal{N}(p)$ within a radius of $2h$, where h is the grid spacing. We then compute a mass-weighted mean position $\boldsymbol{p}_p$ and a mass-weighted covariance matrix $\mathbf{V}_p$ for each weakened particle p via (21) and (22), respectively. The weighting function is given by

$$w(\boldsymbol{x}_q - \boldsymbol{x}_p) = N(\frac{1}{h}(x_q - x_p)) N(\frac{1}{h}(y_q - y_p)) N(\frac{1}{h}(z_q - z_p)) \ .$$

We compute the eigendecomposition of $\mathbf{V}_p$ and consider the ratio between the largest and smallest eigenvalues. If this ratio is beneath a threshold (10^{-5} in our tests) or the largest eigenvalue is too small (smaller than 10^{-5}), we remove the weakened particle from the simulation.

13. Resolve particle-based collisions. To model adhesive obstacles, we set the relative velocity between a particle and an obstacle to 0 if the particle is inside an obstacle.

We model non-adhesive obstacles as in [Stomakhin et al. 2013]-§8.

14. Update particle positions. We set the particle positions to $\boldsymbol{x}_{p,n+1} = \boldsymbol{x}_{p,n} + \Delta t \boldsymbol{v}_{p,n+1}$.

15. Avoid a Void. We fill voids for a sparse subset of all time steps. We consider each material *point* to be a *sphere* of radius $r = \frac{1}{2} h$, where h is the spacing of the MPM background grid. We then employ Algorithm 2 to compute an approximate signed distance function (SDF) on a uniform grid (distinct from the MPM background grid) of cell size r. We fill potential void regions by first reducing the radius of each point to αr. We use a value of $\alpha \approx \frac{\sqrt{3}}{2} + \frac{1}{100}$ in our simulations. We then employ octree-based dart throwing (Algorithm 3) to populate interior regions ($2.2r$ away from the interface) with new particles.

We need to assign physical values to newly inserted material points. For conserved quantities (i.e., mass m and volume V), we redistribute values from the surrounding points to the new point. For field values that are not inherently conserved ($\boldsymbol{v}$, $\mathbf{F}$, $\overline{\mathbf{b}}^e$), we employ mass-weighted interpolation. We renormalize $\overline{\mathbf{b}}^e$ to ensure

that $\det(\overline{\mathbf{b}}^e) = 1$. After interpolating both $\mathbf{F}$ and its determinant J, we rescale $\mathbf{F}$ to ensure that its determinant is equal to J.

Finally, we merge points that are too close to one another. To assign values to a new merged point, we simply sum the values of the conserved quantities, and compute a mass-weighted average for the remaining quantities. We again rescale $\mathbf{F}$ for consistency with J, and renormalize $\overline{\mathbf{b}}^e$.

E.4 Internal Data Structure and Variables

MPM Background Grid. At each MPM grid node $\boldsymbol{i}$, we store the mass $m_{\boldsymbol{i}}$, the velocity $\boldsymbol{v}_{\boldsymbol{i}}$, the updated velocity $\boldsymbol{v}_{\boldsymbol{i}}^\star$, and the internal force $\boldsymbol{f}_{\boldsymbol{i}}$. In addition, we allocate memory for the deformation gradient $\mathbf{F}_{\boldsymbol{i}}$, for the determinant of the deformation gradient $J_{\boldsymbol{i}}$, for the strain $\overline{\mathbf{b}}_{\boldsymbol{i}}^e$, and for the sum of weights $w_{\boldsymbol{i},\text{sum}}$ (in order to compute field values at newly created particles).

Material Points. Each material point p stores its mass m_p, its volume V_p, its velocity $\boldsymbol{v}_p$, its normalized strain $\overline{\mathbf{b}}^e$, and its total deformation gradient $\mathbf{F}$. In addition, we store the binary flags $weak$ and $valid$. The flag $weak$ indicates tearing, and is true if and only if $\| \operatorname{dev}[\mathbf{C}_p^p] \|_F > \sigma_T$. The flag $valid$ indicates whether or not we will remove a given particle.

Auxiliary Uniform Grid. We employ an auxiliary uniform grid to accelerate neighboring particle-particle queries. Note that other neighbor queries do not require an auxiliary uniform grid. That is, neighboring grid node queries for particles can be directly computed by converting particle positions to grid node IDs, and neighboring particle queries for grid nodes can be avoided by inverting the computation order: we scatter particle data onto the grid.

At each node of the uniform grid, we store a list of the neighbor particles.

SDF Grid. At each node of the SDF grid, we store the minimum distance to the interface and a pointer to the closest zero-crossing point.

E.5 Pseudocode

Algorithm 4 `Simulate_Foam`

```
Compute_Particle_Volumes_and_Densities
time_step ← 0
while time_step < num_steps do
    Rasterize_Particle_Data_to_Grid
    Detect_Tearing_Regions
    Compute_and_Limit_Grid_Forces
    Update_Grid_Velocities
    Resolve_Grid_Based_Collisions
    Update_and_Limit_Deformation_Gradient
    Update_Particle_Velocities
    Compute_Plastic_Flow
    Compute_Plastic_Recovery
    Remove_Subgrid_Geometry
    Resolve_Particle_Based_Collisions
    Update_Particle_Positions
    if time_step mod Avoid_a_Void_Interval = 0 then
                                ▷ Avoid_a_Void_Interval = 50
        Avoid_a_Void
    end if
    time_step ← time_step + 1
end while
```

Algorithm 5 `Compute_Particle_Volumes_and_Densities`

```
Rasterize_Particle_Data_to_Grid
for each particle p do
    ρ_p ← 0
    M_p ← all MPM grid nodes satisfying w_ip > 0
    for each MPM grid node i ∈ M_p do
        ρ_p ← ρ_p + w_ip m_i                ▷ Accumulate mass
    end for
    ρ_p ← ρ_p / h^3
    V_p ← m_p / ρ_p
end for
```

Algorithm 6 `Rasterize_Particle_Data_to_Grid`

```
for each MPM grid node i do
    m_i ← 0
    v_i ← 0
end for
for each particle p do
    M_p ← all MPM grid nodes satisfying w_ip > 0
    for each MPM grid node i ∈ M_p do
        m_i ← m_i + w_ip m_p                ▷ Accumulate mass
        v_i ← v_i + w_ip m_p v_p            ▷ Accumulate momentum
    end for
end for
for each MPM grid node i do
    v_i ← v_i / m_i
end for
```

Algorithm 7 `Detect_Tearing_Regions`

```
for each particle p do
    C_p^p ← det(F_p)^{-2/3} F_p^T (b̄_p^e)^{-1} F_p
    if ‖ dev[C_p^p] ‖_F > σ_T then
        Label p as weak
    else
        Label p as not weak
    end if
end for
```

Algorithm 8 `Compute_and_Limit_Grid_Forces`

for each MPM grid node $\boldsymbol{i}$ **do**
  $\boldsymbol{f_i} \leftarrow 0$
end for
for each particle p **do**
  $J_p \leftarrow \det(\mathbf{F}_p)$
  $\boldsymbol{\sigma}_p \leftarrow \frac{1}{J_p}\left[\frac{\kappa}{2}(J_p^2-1)\mathbf{I} + \mu\,\mathrm{dev}[\bar{\mathbf{b}}_p^e]\right]$
  if p is labeled as weak **then**
    $\mathbf{V}_p\boldsymbol{\Sigma}_p\mathbf{V}_p^T \leftarrow \boldsymbol{\sigma}_p$ ▷ Eigendecompostion
    $s_i \leftarrow (\boldsymbol{\Sigma}_p)_{ii}$ ▷ Eigenvalues
    $\boldsymbol{\Sigma}_p \leftarrow diag(min(0, s_i))$ ▷ Limit stress
    $\boldsymbol{\sigma}_p \leftarrow \mathbf{V}_p\boldsymbol{\Sigma}_p\mathbf{V}_p^T$
  end if
  $\mathcal{M}_p \leftarrow$ all MPM grid nodes $\boldsymbol{i}$ satisfying $w_{ip} > 0$
  for each MPM grid node $\boldsymbol{i} \in \mathcal{M}_p$ **do**
    $\boldsymbol{f_i} \leftarrow \boldsymbol{f_i} - V_p J_p \sigma_p \nabla w_{ip}$ ▷ Accumulate force
  end for
end for

Algorithm 9 `Update_Grid_Velocities`

for each MPM grid node $\boldsymbol{i}$ **do**
  $\boldsymbol{v_i^\star} \leftarrow \Delta t \boldsymbol{f_i}/m_i + \Delta t \boldsymbol{g}$
end for

Algorithm 10 `Resolve_Grid_Based_Collisions`

for each MPM grid node $\boldsymbol{i}$ **do**
  if $\boldsymbol{i}$ is inside an obstacle **then**
    $\boldsymbol{v_i^\star} \leftarrow \boldsymbol{v}_{\text{obstacle}}$
  end if
end for

Algorithm 11 `Update_and_Limit_Deformation_Gradient`

for each particle p **do**
  $\nabla \boldsymbol{v}_{p,n} \leftarrow \mathbf{0}$
  $\mathcal{M}_p \leftarrow$ all MPM grid nodes $\boldsymbol{i}$ satisfying $w_{ip} > 0$
  for each MPM grid node $\boldsymbol{i} \in \mathcal{M}_p$ **do**
    $\nabla \boldsymbol{v}_{p,n} \leftarrow \nabla \boldsymbol{v}_{p,n} + \boldsymbol{v}_{i,n}(\nabla w_{ip})^T$
  end for
  $\mathbf{f}_{p,n+1} \leftarrow (\mathbf{I} + \Delta t \nabla \boldsymbol{v}_{p,n})$
  $\bar{\mathbf{b}}_{p,n+1}^{e,\text{pre}} \leftarrow \bar{\mathbf{f}}_{p,n+1}\bar{\mathbf{b}}_{p,n}^{e}\bar{\mathbf{f}}_{p,n+1}^T$
  if p is not labeled as weak **then**
    $\mathbf{F}_{p,n+1} \leftarrow \mathbf{f}_{p,n+1}\mathbf{F}_{p,n}$
    continue
  end if
  $\lambda_n \leftarrow$ largest eigenvalue of $\bar{\mathbf{b}}_{p,n}^{e}$
  $\lambda_{n+1}^{\text{pre}} \leftarrow$ largest eigenvalue of $\bar{\mathbf{b}}_{p,n+1}^{e,\text{pre}}$
  $\mathbf{f}_{n+1}^{corr} = \mathbf{f}_{p,n+1}$
  if $\lambda_{n+1}^{\text{pre}} > \lambda_n$ **then** ▷ Shear is accumulating
    $J_f \leftarrow \det(\mathbf{f}_{p,n+1})$
    $\bar{\mathbf{f}}_{n+1} \leftarrow J_f^{-1/3}\mathbf{f}_{p,n+1}$
    $\mathbf{U}\mathbf{S}\mathbf{V}^T \leftarrow \bar{\mathbf{f}}_{n+1}$ ▷ SVD
    $\bar{\mathbf{f}}_{n+1} \leftarrow \mathbf{U}\mathbf{V}^T$
    $\bar{\mathbf{b}}_{p,n+1}^{e,\text{pre}} \leftarrow \bar{\mathbf{f}}_{n+1}\bar{\mathbf{b}}_{p,n}^{e}\bar{\mathbf{f}}_{n+1}^T$
    $\mathbf{f}_{n+1}^{corr} \leftarrow J_f^{1/3}\bar{\mathbf{f}}_{n+1}$ ▷ Only retain the rotation
  end if
  $J_n = \det(\mathbf{F}_{p,n})$
  $J_f^{corr} = \det(\mathbf{f}_{n+1}^{corr})$
  if $J_n J_f^{corr} > 1$ and $J_f^{corr} > 1$ **then**
    ▷ Previously and continuing to expand
    $\mathbf{f}_{n+1}^{corr} \leftarrow (J_f^{corr})^{-1/3}\mathbf{f}_{n+1}^{corr}$ ▷ Discard expansion
  end if
  $\mathbf{F}_{p,n+1} \leftarrow \mathbf{f}_{n+1}^{corr}\mathbf{F}_{p,n}$
end for

Algorithm 12 `Update_Particle_Velocities`

for each particle p **do**
  $\boldsymbol{v}_{\text{PIC}p} \leftarrow 0$
  $\boldsymbol{v}_{\text{FLIP}p} \leftarrow \boldsymbol{v}_p$
  $\mathcal{M}_p \leftarrow$ all MPM grid nodes $\boldsymbol{i}$ satisfying $w_{ip} > 0$
  for each MPM grid node $\boldsymbol{i} \in \mathcal{M}_p$ **do**
    $\boldsymbol{v}_{\text{PIC}p} \leftarrow \boldsymbol{v}_{\text{PIC}p} + w_{ip}\boldsymbol{v_i^\star}$
    $\boldsymbol{v}_{\text{FLIP}p} \leftarrow \boldsymbol{v}_{\text{FLIP}p} + w_{ip}(\boldsymbol{v_i^\star} - \boldsymbol{v_i})$
  end for
  $\boldsymbol{v}_p \leftarrow (1-\alpha)\boldsymbol{v}_{\text{PIC}p} + \alpha\boldsymbol{v}_{\text{FLIP}p}$ ▷ $\alpha = 0.95$
end for

Algorithm 13 `Compute_Plastic_Flow`

for each particle p **do**
  $\mathbf{s}^{\text{pre}}_{n+1} \leftarrow \mu \operatorname{dev}[\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}]$
  $s^{\text{pre}}_{n+1} \leftarrow \|\mathbf{s}^{\text{pre}}_{n+1}\|_F$
  if $s^{\text{pre}}_{n+1} - \sqrt{\frac{2}{3}}\sigma_Y \leq 0$ **then** ▷ Elastic regime
    continue
  end if
  $\tilde{\mu} \leftarrow \frac{1}{3}\operatorname{Tr}[\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}]\mu$
  if $\eta = 0$ or $h = 1$ **then** ▷ Direct solve
    $s_{n+1} \leftarrow s^{\text{pre}}_{n+1} - \left(s^{\text{pre}}_{n+1} - \sqrt{\frac{2}{3}}\sigma_Y\right) \Big/ \left(1 + \frac{\eta}{2\tilde{\mu}\Delta t}\right)$
  else ▷ Numerical solve via bisection
    $[s_m, s_M] \leftarrow [\sqrt{\frac{2}{3}}\sigma_Y, s^{\text{pre}}_{n+1}]$ ▷ Initial search range
    repeat
      $s \leftarrow (s_m + s_M)/2$
      $g \leftarrow \eta^{\frac{1}{h}}\left(s - s^{\text{pre}}_{n+1}\right) + 2\tilde{\mu}\Delta t\left(s - \sqrt{\frac{2}{3}}\sigma_Y\right)^{\frac{1}{h}}$
      if $g < 0$ **then**
        $s_m \leftarrow s$ ▷ Update search range
      else
        $s_M \leftarrow s$ ▷ Update search range
      end if
      $E \leftarrow g/s^{\text{pre}}_{n+1}$ ▷ Relative error
    until $|E| < \epsilon$ ▷ $\epsilon = 10^{-6}$
  end if
  $\hat{\mathbf{s}}^{\text{pre}}_{n+1} \leftarrow \mathbf{s}^{\text{pre}}_{n+1}/s^{\text{pre}}_{n+1}$ ▷ Flow direction
  $\mathbf{s}_{n+1} \leftarrow s_{n+1}\hat{\mathbf{s}}^{\text{pre}}_{n+1}$
  $\overline{\mathbf{b}}^{e}_{n+1} \leftarrow \frac{1}{\mu}\mathbf{s}_{n+1} + \frac{1}{3}\operatorname{Tr}[\overline{\mathbf{b}}^{e,\text{pre}}_{n+1}]\mathbf{I}$
  $\overline{\mathbf{b}}^{e}_{n+1} \leftarrow \det(\overline{\mathbf{b}}^{e}_{n+1})^{-1/3}\overline{\mathbf{b}}^{e}_{n+1}$ ▷ Renormalize
end for

Algorithm 14 `Compute_Plastic_Recovery`

for each particle p **do**
  $\mathbf{C}^p_p \leftarrow \det(\mathbf{F}_p)^{-2/3}\mathbf{F}^T_p(\overline{\mathbf{b}}^e_p)^{-1}\mathbf{F}_p$
  $\mathbf{USU}^T \leftarrow \mathbf{C}^p_p$ ▷ Eigendecomposition
  $\lambda_i \leftarrow \mathbf{S}_{ii}^{(exp(-\Delta t/\eta_p)-1)/2}$
  $\mathbf{R} \leftarrow diag(\lambda_i)$
  $\mathbf{F}_p \leftarrow \mathbf{F}_p\mathbf{URU}^T$
end for

Algorithm 15 `Remove_Subgrid_Geometry`

for each particle p **do**
  if p is labeled as weak **then**
    $\mathcal{N}_p \leftarrow$ all particles q satisfying $w(\boldsymbol{x}_p - \boldsymbol{x}_q) > 0$ ▷ Use uniform grid for query
    $\boldsymbol{p}_p \leftarrow \mathbf{0}$ ▷ Initialize mean position
    $w_{p,\text{sum}} \leftarrow 0$ ▷ Initialize weight sum
    for each particle $q \in \mathcal{N}_p$ **do**
      $w \leftarrow w(\boldsymbol{x}_q - \boldsymbol{x}_p)m_q$ ▷ Compute weight
      $\boldsymbol{p}_p \leftarrow \boldsymbol{p}_p + w\boldsymbol{x}_q$ ▷ Accumulate position
      $w_{p,\text{sum}} \leftarrow w_{p,\text{sum}} + w$ ▷ Accumulate weight
    end for
    $\boldsymbol{p}_p \leftarrow \boldsymbol{p}_p/w_{p,\text{sum}}$ ▷ Compute mean position
    $\mathbf{V}_p \leftarrow \mathbf{0}$ ▷ Initialize covariance matrix
    $w_{p,\text{sum}} \leftarrow 0$ ▷ Initialize weight sum
    for each particle $q \in \mathcal{N}_p$ **do**
      $w \leftarrow (w(\boldsymbol{x}_q - \boldsymbol{x}_p))^2 m_q^2$ ▷ Compute weight
      $\mathbf{V}_p \leftarrow \mathbf{V}_p + w(\boldsymbol{x}_q - \boldsymbol{p}_p)(\boldsymbol{x}_q - \boldsymbol{p}_p)^T$ ▷ Accumulate covariance
      $w_{p,\text{sum}} \leftarrow w_{p,\text{sum}} + w$ ▷ Accumulate weight
    end for
    $\mathbf{V}_p \leftarrow \mathbf{V}_p/w_{p,\text{sum}}$ ▷ Covariance matrix
    $\mathbf{USU}^T \leftarrow \mathbf{V}_p$ ▷ Eigendecomposition
    $(\lambda_M, \lambda_m) \leftarrow (\max_i\{\mathbf{S}_{ii}\}, \min_i\{\mathbf{S}_{ii}\})$
    if $\lambda_m/\lambda_M < \epsilon$ **then** ▷ $\epsilon = 10^{-5}$ ▷ Neighbor distribution is highly anisotropic
      Label p as invalid
    end if
  end if
end for
for each particle p **do**
  if p is invalid **then**
    Remove p from the simulation
  end if
end for

Algorithm 16 `Resolve_Particle_Based_Collisions`

for each particle p **do**
  if p is inside an obstacle **then**
    $\boldsymbol{v}_p \leftarrow \boldsymbol{v}_{\text{obstacle}}$
  end if
end for

Algorithm 17 `Update_Particle_Positions`

for each particle p **do**
  $\boldsymbol{x}_p \leftarrow \boldsymbol{x}_p + \Delta t\boldsymbol{v}_p$
end for

Algorithm 18 `Avoid_a_Void`

`Compute_Signed_Distance` ▷ Algorithm 2
`Create_Particles_with_Poisson_Sampling` ▷ Algorithm 3
`Compute_New_Conserved_Values` ▷ Algorithm 19
`Compute_New_Field_Values` ▷ Algorithm 20
`Merge_Particles` ▷ Algorithm 21

Algorithm 19 `Compute_New_Conserved_Values`

for each newly sampled particle p **do**
  $\mathcal{N}_p \leftarrow$ all existing particles q satisfying $|\boldsymbol{x}_q - \boldsymbol{x}_p| < r$
  $N \leftarrow |\mathcal{N}_p|$ ▷ The number of neighbor particles
  $m_{p,\text{sum}} \leftarrow 0$ ▷ Initialize total mass
  $V_{p,\text{sum}} \leftarrow 0$ ▷ Initialize total volume
  for each particle $q \in \mathcal{N}_p$ **do**
    $m_{p,\text{sum}} \leftarrow m_{p,\text{sum}} + m_q$ ▷ Accumulate mass
    $V_{p,\text{sum}} \leftarrow V_{p,\text{sum}} + V_q$ ▷ Accumulate volume
    $m_q \leftarrow \frac{N}{N+1} m_q$ ▷ Reduce mass
    $V_q \leftarrow \frac{N}{N+1} V_q$ ▷ Reduce volume
  end for
  $m_p \leftarrow \frac{1}{N+1} m_{p,\text{sum}}$ ▷ Gathered mass
  $V_p \leftarrow \frac{1}{N+1} V_{p,\text{sum}}$ ▷ Gathered volume
end for

Algorithm 20 `Compute_New_Field_Values`

for each node $\boldsymbol{i}$ in MPM grid **do**
  $\boldsymbol{v_i} \leftarrow \mathbf{0}$ ▷ Initialize velocity
  $\mathbf{F_i} \leftarrow \mathbf{0}$ ▷ Initialize deformation gradient
  $J_{\boldsymbol{i}} \leftarrow 0$ ▷ Initialize determinant
  $\overline{\mathbf{b}}^e_{\boldsymbol{i}} \leftarrow \mathbf{0}$ ▷ Initialize normalized strain
  $w_{\boldsymbol{i},\text{sum}} \leftarrow 0$ ▷ Initialize weight sum
end for
for each particle p that existed before resampling **do**
  $\mathcal{M}_p \leftarrow$ all MPM grid nodes $\boldsymbol{i}$ satisfying $w_{\boldsymbol{ip}} > 0$
  for each MPM grid node $\boldsymbol{i} \in \mathcal{M}_p$ **do**
    $w \leftarrow w_{\boldsymbol{ip}} m_p$ ▷ Weight
    $w_{\boldsymbol{i},\text{sum}} \leftarrow w_{\boldsymbol{i},\text{sum}} + w$ ▷ Accumulate weight
    $\boldsymbol{v_i} \leftarrow \boldsymbol{v_i} + w\boldsymbol{v}_p$ ▷ Accumulate velocity
    $\mathbf{F_i} \leftarrow \mathbf{F_i} + w\mathbf{F}_p$ ▷ Accumulate deformation gradient
    $J_{\boldsymbol{i}} \leftarrow J_{\boldsymbol{i}} + wJ_p$ ▷ Accumulate determinant
    $\overline{\mathbf{b}}^e_{\boldsymbol{i}} \leftarrow \overline{\mathbf{b}}^e_{\boldsymbol{i}} + w\overline{\mathbf{b}}^e_p$ ▷ Accumulate normalized strain
  end for
end for
for each node $\boldsymbol{i}$ in MPM grid **do**
  $\boldsymbol{v_i} \leftarrow \boldsymbol{v_i}/w_{\boldsymbol{i},\text{sum}}$ ▷ Velocity
  $\mathbf{F_i} \leftarrow \mathbf{F_i}/w_{\boldsymbol{i},\text{sum}}$ ▷ Deformation gradient
  $J_{\boldsymbol{i}} \leftarrow J_{\boldsymbol{i}}/w_{\boldsymbol{i},\text{sum}}$ ▷ Determinant
  $\overline{\mathbf{b}}^e_{\boldsymbol{i}} \leftarrow \overline{\mathbf{b}}^e_{\boldsymbol{i}}/w_{\boldsymbol{i},\text{sum}}$ ▷ Normalized strain
end for
for each newly sampled particle p **do**
  $\boldsymbol{v}_p \leftarrow \mathbf{0}$ ▷ Initialize velocity
  $\mathbf{F}_p \leftarrow \mathbf{0}$ ▷ Initialize deformation gradient
  $J_p \leftarrow 0$ ▷ Initialize determinant
  $\overline{\mathbf{b}}^e_p \leftarrow \mathbf{0}$ ▷ Initialize normalized strain
  $\mathcal{M}_p \leftarrow$ all MPM grid nodes $\boldsymbol{i}$ satisfying $w_{\boldsymbol{ip}} > 0$
  for each MPM grid node $\boldsymbol{i} \in \mathcal{M}_p$ **do**
    $\boldsymbol{v}_p \leftarrow \boldsymbol{v}_p + w_{\boldsymbol{ip}}\boldsymbol{v_i}$ ▷ Accumulate velocity
    $\mathbf{F}_p \leftarrow \mathbf{F}_p + w_{\boldsymbol{ip}}\mathbf{F_i}$ ▷ Accumulate deformation gradient
    $J_p \leftarrow J_p + w_{\boldsymbol{ip}} J_{\boldsymbol{i}}$ ▷ Accumulate determinant
    $\overline{\mathbf{b}}^e_p \leftarrow \overline{\mathbf{b}}^e_p + w_{\boldsymbol{ip}}\overline{\mathbf{b}}^e_{\boldsymbol{i}}$ ▷ Accumulate normalized strain
  end for
  $\overline{\mathbf{b}}^e_p \leftarrow \det(\overline{\mathbf{b}}^e_p)^{-1/3}\overline{\mathbf{b}}^e_p$ ▷ Renormalize strain
  $\mathbf{F}_p \leftarrow J_p^{1/3}(\det(\mathbf{F}_p)^{-1/3}\mathbf{F}_p)$ ▷ Rescale deformation gradient
end for

Algorithm 21 `Merge_Particles`

$\mathcal{P} \leftarrow \emptyset$ ▷ Set of potential particle pairs to merge
for each particle p **do**
  $p.selected \leftarrow$ false
end for
for each particle p **do**
  $\mathcal{N}_p \leftarrow$ all particles q satisfying $|\boldsymbol{x}_q - \boldsymbol{x}_p| < 0.03r$
  for each particle q **do**
    $\mathcal{P} \leftarrow \mathcal{P} \cup (p, q)$ ▷ Particles p and q are too close, ▷ add the pair to the potential merge set
  end for
end for
$\mathcal{D} \leftarrow \emptyset$ ▷ Set of particle pairs to merge
for each particle pair $s \in \mathcal{P}$ **do**
  if not $s.p.selected$ and not $s.q.selected$ **then**
    $\mathcal{D} \leftarrow s$ ▷ Append the particle pair
    $s.p.selected \leftarrow$ true ▷ Skip $s.p$ in future iterations
    $s.q.selected \leftarrow$ true ▷ Skip $s.q$ in future iterations
  end if
end for
for each particle pair $s \in \mathcal{D}$ **do**
  $m \leftarrow m_{s.p} + m_{s.q}$ ▷ Merge mass
  $V \leftarrow V_{s.p} + V_{s.q}$ ▷ Merge volume
  $\boldsymbol{v} \leftarrow (m_{s.p}\boldsymbol{v}_{s.p} + m_{s.q}\boldsymbol{v}_{s.q})/m$ ▷ Merge velocity
  $\mathbf{F} \leftarrow (m_{s.p}\mathbf{F}_{s.p} + m_{s.q}\mathbf{F}_{s.q})/m$ ▷ Merge deformation gradient
  $J \leftarrow (m_{s.p}J_{s.p} + m_{s.q}J_{s.q})/m$ ▷ Merge determinant
  $\overline{\mathbf{b}}^e \leftarrow (m_{s.p}\overline{\mathbf{b}}^e_{s.p} + m_{s.q}\overline{\mathbf{b}}^e_{s.q})/m$ ▷ Merge strain
  $\overline{\mathbf{b}}^e \leftarrow \det(\overline{\mathbf{b}}^e)^{-1/3}\overline{\mathbf{b}}^e$ ▷ Renormalize strain
  $\mathbf{F} \leftarrow J^{1/3}\det(\mathbf{F})^{-1/3}\mathbf{F}$ ▷ Rescale deformation gradient
  delete $s.p$ and $s.q$
  add a particle with m, V, $\boldsymbol{v}$, $\mathbf{F}$, $\overline{\mathbf{b}}^e$
end for