


A Novel Plastic Phase-Field Method for Ductile Fracture with GPU Optimization

Zipeng Zhao¹, Kemeng Huang^{1†}, Chen Li^{1‡}, Changbo Wang¹, Hong Qin²

¹School of Computer Science and Technology, East China Normal University, Shanghai, China

²Department of Computer Science, Stony Brook University, Stony Brook, USA



Figure 1: Applications of PPF in different scenarios. Our method can simulate and visualize the various ductile fracture phenomena for different elastoplastic materials, including plastic, dough, and multi-layer hamburger.

Abstract

In this paper, we articulate a novel plastic phase-field (PPF) method that can tightly couple the phase-field with plastic treatment to efficiently simulate ductile fracture with GPU optimization. At the theoretical level of physically-based modeling and simulation, our PPF approach assumes the fracture sensitivity of the material increases with the plastic strain accumulation. As a result, we first develop a hardening-related fracture toughness function towards phase-field evolution. Second, we follow the associative flow rule and adopt a novel degraded von Mises yield criterion. In this way, we establish the tight coupling of the phase-field and plastic treatment, with which our PPF method can present distinct elastoplasticity, necking, and fracture characteristics during ductile fracture simulation. At the numerical level towards GPU optimization, we further devise an advanced parallel framework, which takes the full advantages of hierarchical architecture. Our strategy dramatically enhances the computational efficiency of preprocessing and phase-field evolution for our PPF with the material point method (MPM). Based on our extensive experiments on a variety of benchmarks, our novel method's performance gain can reach $1.56\times$ speedup of the primary GPU MPM. Finally, our comprehensive simulation results have confirmed that this new PPF method can efficiently and realistically simulate complex ductile fracture phenomena in 3D interactive graphics and animation.

CCS Concepts

• **Computing methodologies** → **Physical simulation; Parallel algorithms;**

1. Introduction

In recent years, the fracture of elastoplastic materials (e.g., iron, rubber, bread, and cheese, etc.) has been simulated in graphics and

animation, starting to gain popularity. Such research attempts also find widespread applications in movies and games industry (thanks to the magnificent visual effects showcased in distinct physical phenomena). Unfortunately, existing simulation methods still cannot accurately capture representative and subtle details pertinent to ductile fracture (e.g., elastic-plastic response, localized damage, crack failure, etc.) due to both the underlying complex physical principles and the lack of time-consuming and robust numerical analysis.

[†] Z. Zhao and K. Huang contribute equally to this work.

[‡] Corresponding author. Email: cli@cs.ecnu.edu.cn.

Therefore, it is of vital importance to develop a user-friendly and graphics-efficient simulation framework for ductile fracture based on correct physical models and GPU optimization schemes. This paper aims to offer an efficient solution.

In graphics, the mesh-based method, represented by the finite element method (FEM) [TF88], is popular in fracture simulation for its numerical accuracy and stability. Later on, Hahn et al. [HW15, HW16] developed the boundary element method (BEM) to efficiently model the brittle fracture and produce rich details for cracked surfaces. However, the mesh-based techniques usually suffer from the time-consuming re-meshing process when severe deformation is unavoidable. Moreover, some graphics researchers introduced meshless methods such as smoothed particle hydrodynamics (SPH) [CWXQ13, LWQ15]. But due to the lack of physics-accurate constraints, it remains challenging for SPH to deal with elastic-plastic responses in a stable and robust way. In recent years, the material point method (MPM) was introduced as a meshfree method with a background grid based on continuum mechanics theory, and recent results had shown that MPM not only facilitates the handling of complex physical models, but also overcomes the aforementioned problems.

Relevant to the aforementioned numerical techniques, the phase-field methods become a popular topic most recently. By smoothing the crack surface into the material, the phase-field helps avoid the crack interface tracking or re-meshing completely, achieving excellent results for fracture simulation in both FEM [Bor12] and MPM [KT17, WFL*19a]. Nevertheless, in the absence of plastic treatment, the ductile fracture phenomena, such as necking, are usually ignored in the existing literature. What is even worse is that, the phase-field evolution requires an additional iterative optimization process, which makes both the time cost increase and the memory overhaul unavoidable.

In this paper, our ambitious goal is to develop a novel physics-based fracture simulation framework for ductile material, which is further enhanced by advanced GPU parallelization. Towards this objective, we articulate a new plastic phase-field (PPF) method coupled with MPM, that could integrate the phase-field fracture theory and the von Mises yield criterion. Once the stress on the material reaches a critical threshold, the underlying material starts to yield and harden. As this process continues, the sensitivity of phase-field will increase and make the material more prone to fracture. Moreover, to maximize the utilization efficiency of GPU memory and threads, we advocate a novel strategy for multiple calculation processes in MPM. In particular, our salient contributions in this paper include:

- (1) **Plastic phase-field with strain accumulation that overcomes the limitations of the classical Griffith theory.** In order to capture more complicated phenomena involving ductile fracture, we modify the original evolution scheme of the phase-field fracture method with our newly-proposed degradation function and hardening-related fracture toughness function. In such way, our PPF can apply the accumulated plastic strain to the evolution of phase-field, so that the plastic deformation can directly and accurately drive the material damaging procedure.
- (2) **Degraded return mapping with the evolution of the phase-field.** Once the material is damaged and deformed, its elastic

response becomes weaker and leads the degraded stress to return into the yield surface. During this evolution, the yield condition should degrade accordingly to simulate the complete process of ductile fracture and prevent the purely elastic failure. Thus we adopt a degraded von Mises yield criterion, which dynamically projects the elastic strain to the degraded yield surface. Consequently, when the strain is excessive, we can weaken the elastic potential energy for all healthy or damaged materials accordingly and capture all the details pertinent to the unrecoverable deformation, necking, damaging, and crack propagation.

- (3) **Efficient parallel framework that takes full advantages of the hierarchical GPU architecture.** To maximize the efficiency of thread-level parallelism and utilization of GPU memory, we first design a thread warp based reduction method that affords threads to make full use of registers and decrease the use of shared memory. In addition, we develop a new prefix summation algorithm, which improves the parallelism of thread groups and reduces the number of thread blocks. Next, we involve the above schemes in the re-initialization stage and phase-field solver to build an efficient GPU-centric framework for our PPF-MPM solver. Based on comprehensive experiments, our novel algorithms achieve better performance compared with GPU MPM [GWW*18].

2. Related Work

Realistic simulation of elastoplastic deformation and fracture has been studied widely in computational physics and computer graphics. However, ductile fracture remains challenging due to the complex theoretical analysis and unstable numerical algorithm. Here we will briefly summarize the most relevant previous works.

Elastoplastic modeling based on MPM. The MPM has a dual view of Eulerian and Lagrangian deformation [SZS95]. After snow simulation from Disney [SSC*13], there have been lots of works that focus on the elastoplastic treatment for MPM. For example, materials like water [TGK*17], sand [KGP*16, TGK*17, GPH*18] and foam [RGJ*15, YSB*15] are almost pure plastic and do not resist shearing, in which the yield criterion is pivotal. However, some elastoplastic materials require a balance between the elastic model and the plastic return mapping, such as snow [SSC*13], rubber [ZZL*17, LGL*19, FLGJ19] and cloth [GHF*18, JGT17]. Therefore, it is crucial to adopt proper constitutive law and yield criterion for the intricate modeling of elastoplastic materials.

Fracture modeling. Fracture simulation has been studied for years in both computational physics and graphics. O'Brien et al. used FEM for both brittle [OH99] and ductile fracture [OBH02] simulations due to its great advantages in dealing with shear and large stress problems [MG04, BHTF07, GMD13]. However, the re-meshing and post-process are indispensable when solving large deformation problems [CYFW14, PNdJO14]. Also, BEM [Ali97, HW16, HW15] was adopted to present the details of crack surface for the brittle fracture phenomena. In addition, researchers also focused on the meshfree approaches such as MLS-based method [MG04], SPH method [CWXQ13, LWQ15], MPM [SSC*13], and level set method [HJST13]. Recently, Wang et al. [WDG*19] applied the softening yield criterion based on MPM and proposed a visualization scheme for rendering the crack surface.

Later, Wolper et al. [WFL*19a] introduced the phase-field fracture method to MPM.

Phase-field method. The phase-field method is popular in recent years. Based on Helmholtz free energy functions, Yang et al. developed a multi-phase method using the Cahn-Hilliard equation to simulate the natural mixing of different liquid mediums and solid-liquid phase changes [YCL*17, YCR*15]. Similarly, the phase-field fracture method based on the energy release of the crack surface naturally avoids topological problems by smoothing the fracture surface into the material. Based on Griffith's theory, the mechanics methods of phase-field are popular because the variational formulation facilitates discretization [BFM00, MHW10]. Lately, Kakouris et al. [KT17] introduced the phase-field method to the MPM framework and solved it by a staggered scheme that is the same as the one in FEM [AKDL16]. Wolper et al. [WFL*19a] also applied this method into graphics and used the conjugate gradient (CG) method to solve the phase-field, which produced realistic results for dynamic fracture of pure elastic materials. However, due to the lack of tight coupling of the phase-field and plasticity, many ductile fracture phenomena cannot be well performed. To overcome the above limitations, our PPF method couples the yield model with the phase-field of material damage.

GPU optimization. With the improvement of computer hardware devices, parallel acceleration based on the CUDA toolkit has become a hot research topic in physically-based simulation, such as GPU SPH [GSSP10, HRZ*19, HZI*20] and GPU MPM [GWW*18]. In GPU MPM, Gao et al. [GWW*18] optimized the grid-particle transfer and particle reordering based on SPGrid, and also provided specific strategies for the numerical calculation, such as the GPU singular value decomposition (SVD) solver. In the latest GPU-based MPM [WQS*20], Wang et al. proposed G2P2G and AoSoA with multiple GPUs, with which the modified MPM workflow has been greatly accelerated. In this case, the time cost of processes like reinitialization cannot be ignored compared with the total time of an MPM step. To facilitate modeling as well as improve the performance of our PPF-MPM workflow, we focus on GPU MPM [GWW*18] and further optimize the reinitialization and phase-field evolution process.

3. Physics Methods

In this section, we will detail the theoretical basis of the PPF method, including governing equations, elastoplasticity, and phase-field theory.

3.1. Continuum Mechanics Background

The deformation theory of MPM is defined in the continuum mechanics by mapping material points from the reference configuration Ω_0 with position \mathbf{X} to the deformed configuration Ω_t with position \mathbf{x} as $\mathbf{x} = \phi(\mathbf{X}, t)$. The deformation gradient \mathbf{F} is the derivative of each component of these two configurations as $\mathbf{F} = \frac{\partial \phi}{\partial \mathbf{X}}(\mathbf{X}, t)$. The governing functions including mass and momentum conservation:

$$\frac{D\rho}{Dt} + \rho \nabla \cdot \mathbf{v} = 0, \quad \rho \frac{D\mathbf{v}}{Dt} = \nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{g}, \quad (1)$$

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

The energy equation, as the material property, is the decisive factor for the elastoplastic deformation. Here we use the superscript e denotes the elastic part of a variable and p for plastic. To model phase-field fracture, we write the energy density function Ψ as

$$\begin{aligned} \Psi &= \Psi^e + \Psi^p, \quad \Psi^e = \Psi^+(\mathbf{b}^e, J^e) + \Psi^-(J^e), \\ \mathbf{F} &= \mathbf{F}^e \mathbf{F}^p, \quad \boldsymbol{\sigma} = \frac{1}{J^e} \frac{\partial \Psi}{\partial \mathbf{F}^e} (\mathbf{F}^e)^T, \quad \mathbf{b} = \mathbf{F}^e (\mathbf{F}^e)^T, \end{aligned} \quad (2)$$

where Ψ^+ is the tensile part of elastic energy density function and Ψ^- is the compression part, \mathbf{b}^e is the elastic left Cauchy-Green deformation tensor, and $J^e = \det(\mathbf{F}^e)$ is determinant of \mathbf{F}^e . In the following, for brevity, we will omit the superscript e for the elastic part of variables.

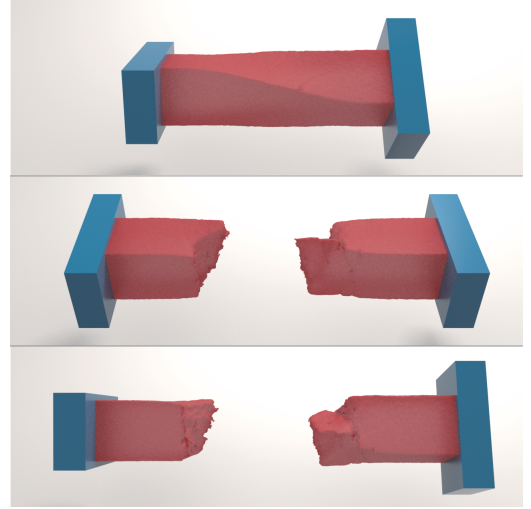


Figure 2: A twisted plastic strip. From top to bottom, the plastic strip is rotated and pulled.

3.2. Elastoplasticity

3.2.1. Hyperelasticity Constitutive Model

In physics, constitutive relation is the relation between force and deformation that expresses the macroscopic properties of materials. The construction of constitutive relations depends on stress tensors and strain tensors [TN04]. In MPM simulations, basic constitutive models include the simplest linear model [DHW*19], and the hyperelastic nonlinear model such as Saint Venant–Kirchhoff model [KGP*16]. Here we adopt the neo-Hookean constitutive model [AKDL16] to describe the change of volume and shape conveniently. Then we split the energy density function into volumetric part and isochoric (deviatoric) part:

$$\begin{aligned} \Psi &= \Psi_{\text{vol}} + \Psi_{\text{dev}}, \\ \Psi_{\text{vol}} &= \frac{\kappa}{2} \left(\frac{J^2 - 1}{2} - \log(J) \right), \quad \Psi_{\text{dev}} = \frac{\mu}{2} (\text{tr}(\bar{\mathbf{b}}) - d), \end{aligned} \quad (3)$$

where $\bar{\mathbf{b}} = J^{-2/d} \mathbf{b}$, d is the simulation dimension, and material parameters are bulk modulus κ and shear modulus μ . Correspondingly,

the stress from the neo-Hookean elastic energy can be written as

$$\boldsymbol{\sigma} = \frac{1}{J} \boldsymbol{\tau}, \quad \boldsymbol{\tau} = \boldsymbol{\tau}_{\text{vol}} + \boldsymbol{\tau}_{\text{dev}} = \frac{\kappa}{2} (J^2 - 1) \mathbf{I} + \mu \text{dev}(\bar{\mathbf{b}}), \quad (4)$$

where $\boldsymbol{\tau}$ is the Kirchhoff stress, $\text{dev}(\mathbf{M}) = \mathbf{M} - \frac{1}{d} \text{tr}(\mathbf{M}) \mathbf{I}$ for any tensor \mathbf{M} , $\boldsymbol{\tau}_{\text{vol}}$ and $\boldsymbol{\tau}_{\text{dev}}$ are respectively volumetric and deviatoric Kirchhoff stress.

Then for the phase-field fracture, we follow the idea of [AMM09, Bor12] and decompose the energy density as

$$\begin{cases} \Psi^+ = \Psi_{\text{dev}} & J < 0, \\ \Psi^- = \Psi_{\text{vol}} & J \geq 0, \end{cases} \quad \begin{cases} \Psi^+ = \Psi_{\text{dev}} + \Psi_{\text{vol}} \\ \Psi^- = 0 \end{cases} \quad J \geq 0. \quad (5)$$

According to the above formulas, we give the elastic potential energy density after degradation as $\hat{\Psi} = g\Psi^+ + \Psi^-$, where g is the degradation function of phase-field. Also, the stress will be degraded accordingly as $\boldsymbol{\tau} = (g \frac{\partial \Psi^+}{\partial \mathbf{F}} + \frac{\partial \Psi^-}{\partial \mathbf{F}}) \mathbf{F}^T$. Then, we record a history maximum positive energy density from the time 0 to T as $\mathcal{H} = \max_{t \in [0, T]} (\Psi_+^{E, n})$ for each material point.

3.2.2. von Mises Plasticity

The plastic treatment is employed to limit elastic potential energy and make the material adapt to deformation, which is necessary for the dynamic process of ductile fracture. Here we will introduce the classic von Mises yield criterion. The yield function is given by

$$y(\boldsymbol{\tau}) = \|\boldsymbol{\tau}_{\text{dev}}^{tr}\|_F - \sqrt{\frac{2}{6-d}} H(\alpha), \quad (6)$$

where y is the yield surface, $\boldsymbol{\tau}_{\text{dev}}^{tr}$ is the trial deviatoric part of Kirchhoff stress, $\|\cdot\|_F$ denotes the Frobenius norm, $H(\alpha)$ is the hardening function and α is the hardening variable. The Lie derivative of the left elastic Cauchy-Green tensor is given by $\mathcal{L}_v \mathbf{b} = -2\gamma \frac{\partial y}{\partial \boldsymbol{\tau}} \mathbf{b}$, where γ is the flow rate. Then we give the evolution function of α as $\dot{\alpha} = \sqrt{\frac{2}{6-d}} \gamma$ [AGDL15, AKDL16]. In the following, we will use an equivalent plastic strain scalar $p = \frac{\alpha}{\alpha_c}$ to represent the accumulation of plastic deformation in a material, where α_c is the critical value of α .

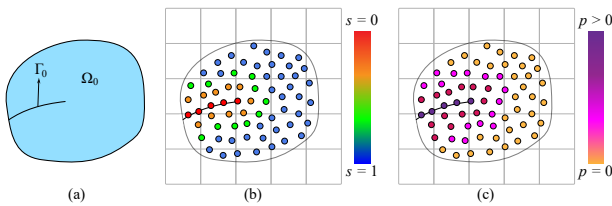


Figure 3: Diagram of fracture surface. From left to right are the fracture diagram, the phase-field distribution diagram, and the plastic strain distribution diagram.

3.3. Phase-Field Theory

Here we emphasize the theoretical basis of the phase-field method for fracture or crack with the variational formulation. Based on Griffith's theory of brittle fracture, the phase-field evolution is solved

as a minimization problem of free energy [FM98], which is given by

$$\mathcal{W} = \int_{\Omega^0} \hat{\Psi} d\mathbf{X} + G_c \int_{\Gamma} d\mathbf{X}, \quad (7)$$

where \mathcal{W} is the total free energy, G_c is the fracture toughness, Γ is the discontinuous crack set [AGDL15, WFL*19a]. The first term in the right hand is the degraded strain energy $\hat{\Psi} = g\Psi^+ + \Psi^-$, and the second is the fracture surface energy \mathcal{W}^s , in which the integral of the fracture surface is replaced by an approximate volume integral [FM98, BFM00] as

$$\mathcal{W}^s(s) = G_c \int_{\Omega^0} \left(\frac{1}{4l} (1-s)^2 + l |\nabla s|^2 \right) d\mathbf{X}, \quad (8)$$

where the phase-field value s represents the material state from healthy ($s = 1$) to damaged ($s < 1$), $s = 0$ denotes the material is completely broken, and the interface depth of the phase-field fracture is controlled by a length-related parameter l . As shown in Fig. 3, the integral along the crack surface (a) is replaced by the volume integral of phase-field in (b), which greatly alleviates the computational difficulties and fits MPM discretization.

Furthermore, in traditional physics literature, the solution for minimizing Eq. (7) is to bind the phase-field evolution and the force calculation into an iterative optimization process [Bor12, AGDL15]. In our paper, to ensure the integrity of the MPM framework, we use the finite difference method to discretize the evolution problem from the derivative of the phase-field (see §4.2 for more details).

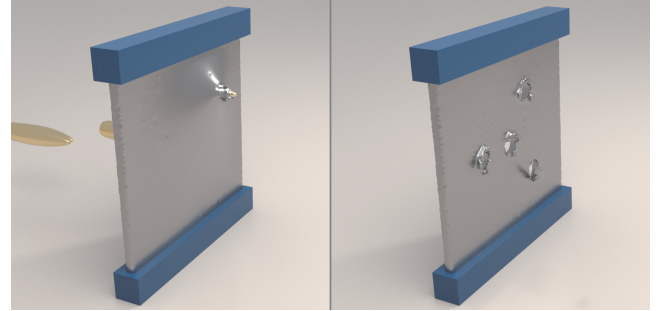


Figure 4: Metal plate. The bullets shot at the metal plate and cause the damage and plastic deformation.

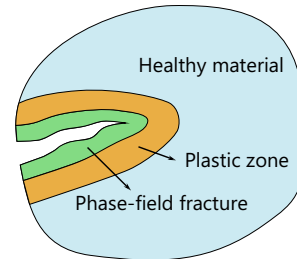


Figure 5: Crack interface. The interface of fracture is in the plastic zone.

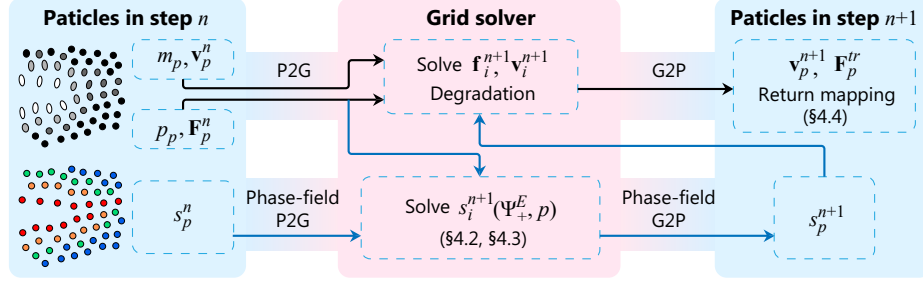


Figure 6: MPM workflow with PPF. After traditional P2G transfer, PPF-MPM solves the phase-field as a subsystem. The solid blue lines denote the additional data flow for PPF.

4. Plastic Phase-Field Method

In this section, we will describe the complete strategy of our proposed PPF method for ductile fracture simulation. First, in §4.1, we will introduce the overall MPM framework with a PPF subsystem. Since our PPF method will not bring additional burden to the phase-field discretization, it can be easily embedded and implemented on the basis of [WFL*19a]. Second, for the evolution of the phase-field in §4.2, we use the finite difference method to obtain an equilibrium equation and derive its weak form [KT17]. Instead of the staggered iteration optimization in computational physics, we construct a large system of linear equations and solve it with the preconditional CG (PCG) method in a few iterations. Third, we will discuss the proposed fracture toughness function in §4.3, which introduces plasticity into the evolution of the phase-field. As a result, the accumulated plastic strain effectively controls the fracture location and sensitivity as shown in Fig. 5. We also fine-tune the degradation function to make the degradation more severe in the case of a violent fracture. Finally, in §4.4, we develop a novel von Mises yield criterion for our PPF method, which applies degradation from the phase-field for the yield surface. That is, when the material is damaged, the yield surface will degrade accordingly for the continuous crack propagation of ductile fracture.

4.1. MPM Workflow and Its Generalization for PPF Method

We start from the classic implementation [SSC*13] and attempt with both implicit and explicit update schemes [HFG*18] for the force computation. Note that we employ the explicit scheme in all our GPU benchmarks. As illustrated in Fig. 6, we provide the full PPF-MPM framework from time t^n to t^{n+1} , where an MPM particle tracks its position \mathbf{x} , mass m , velocity \mathbf{v} , and deformation gradient \mathbf{F} . Following the governing functions in Eq. (1), we interpolate the contributions of particles to the backward Eulerian grid using the APIC [JSS*15] and MLS-MPM [HFG*18] transfer schemes. Note that in the overall document, we use subscripts p for particle attributes and i for grid attributes. Also, to simplify the formula, the notation of the quadratic B-spline interpolating function $N_i(\mathbf{x})$ is simplified as $w_{ip} = N_i(\mathbf{x}_p^n)$, $\nabla w_{ip} = \nabla N_i(\mathbf{x}_p^n)$.

- (1) **Particles to grid (P2G).** For mass and momentum, we use APIC transfer scheme with a velocity-related matrix \mathbf{C} as $m_i^n = \sum_p w_{ip}^n m_p$, $\mathbf{v}_i^n = \frac{1}{m_i^n} \sum_p w_{ip}^n m_p (\mathbf{v}_i^n + \mathbf{C}_p^n (\mathbf{x}_i - \mathbf{x}_p^n))$.

- (2) **Plastic phase-field solver.** Here our phase-field solver is a subsystem as illustrated in Fig. 6. First, the phase-field P2G process transfers phase-field value from particles to the background grid as $s_i^n = \sum_p w_{ip}^n s_p^n / (\sum_p w_{ip}^n)$. Then, we solve phase-field in grid (see more details in §4.2). Finally, we transfer grid value back to particles: $s_p^{n+1} = \max(\min(s_p^n, \sum_i w_{ip}^n s_i^{n+1}), 0)$, and update the degradation function accordingly.
- (3) **Force computation.** We update force and velocity as $\mathbf{f}_i^* = -\sum_p w_{ip}^n V_p \frac{4}{\Delta x^2} \sigma_p(\mathbf{F}_p^*, s_p^{n+1})(\mathbf{x}_i - \mathbf{x}_p^n)$, $\mathbf{v}_i^{n+1} = \mathbf{v}_i^n + \Delta t m_i^{-1} \mathbf{f}_i^*$, where V_p is particle volume, σ_p is Cauchy stress, and $* = n$ for symplectic Euler and $* = n+1$ for backward Euler. At the same time, we update the maximum tensile energy density \mathcal{H} .
- (4) **Grid to particles (G2P).** Update particles states as $\mathbf{v}_p^{n+1} = \sum_i w_{ip}^n \mathbf{v}_i^{n+1}$, $\mathbf{C}_p^{n+1} = \frac{4}{\Delta x^2} \sum_i w_{ip}^n \mathbf{v}_i^{n+1} (\mathbf{x}_i^n - \mathbf{x}_p^n)^T$, $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}$.
- (5) **Deformation gradient update.** The trial elastic deformation gradient is computed as $\mathbf{F}_p^{tr} = (\mathbf{I} + \Delta t \mathbf{C}_p^{n+1}) \mathbf{F}_p^n$. We update $\mathbf{F}_p^{n+1} = \mathbf{F}_p^{tr}$ if no yielding. Otherwise, an additional von Mises return mapping (see §4.4 for more details) is required to project the elastic strain to yield surface, and the fracture toughness can be updated accordingly by Eq. (13). Further details of the algorithm are provided in Appendix A.

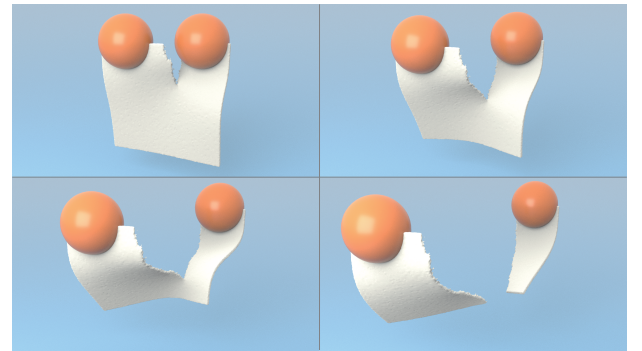


Figure 7: Dough slice. Tearing apart a square piece of raw dough.

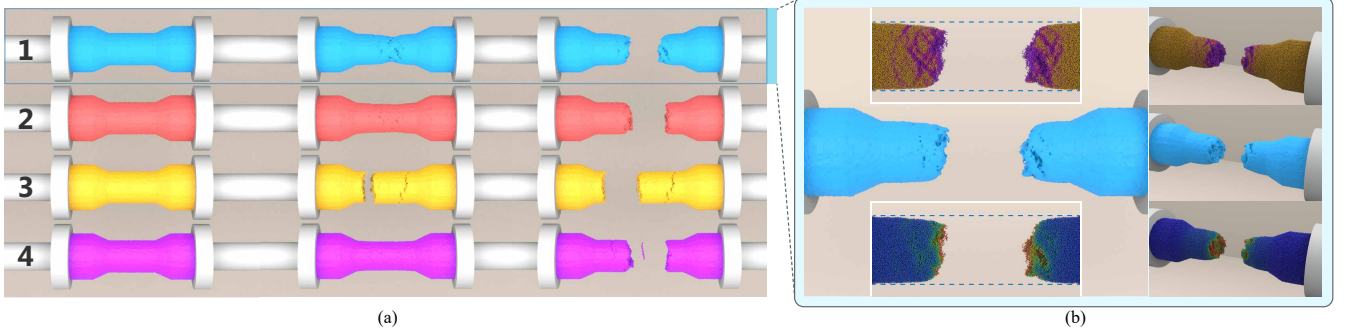


Figure 8: Workpiece fracture. (a) From top to bottom, the workpiece is pulled with our PPF method, phase-field fracture without plasticity [WFL*19a], von Mises plasticity, and the direct combination of phase-field fracture and classic von Mises plasticity in [WFL*19a]. Our method shows necking obviously during ductile fracture. (b) Fracture results of our PPF method, as well as the color distribution of both plastic strain and phase-field. After crack failure, the workpiece keeps localized deformation and damage.

4.2. Phase-field Discretization of PPF

Here we describe how to calculate our phase-field. From the regularized formulation in [BFM00], we write the derivative of the phase-field in Ginzburg-Landau form as

$$\frac{ds}{dt} = -M \frac{\partial \mathcal{W}}{\partial s} = -M \left(2 \left(\frac{\beta}{\beta+p} - \theta \right) s \mathcal{H} - G_c \left(\frac{1-s}{2l} + 2l \nabla^2 s \right) \right), \quad (9)$$

where M is the mobility parameter, β and θ are parameters of degradation function. So Eq. (9) implies that the phase-field evolution is determined by p and \mathcal{H} . In physics, the phase-field fracture method is often used to solve static mechanics problems, and the right hand of Eq. (9) without the parameter M needs to be optimized iteratively until its value is close to zero [AKDL16, BHL*16]. But in the above case, the force needs to be calculated in each iteration, which breaks the traditional MPM framework. To simplify the problem, we follow the Galerkin approximation [Bat06] to discretize phase-field evolution and use M as a parameter to control crack growth. The larger M is, the easier the fracture phase-field propagates [WFL*19a]. Then we give the weak form as

$$\int_V \left(2 \left(\frac{\beta}{\beta+p} - \theta \right) \mathcal{H} + \frac{G_c}{2l} + \frac{1}{M \Delta t} \right) s w dV + \int_V 2l (\nabla s : \nabla w) dV - \int_V \left(\frac{s^n}{M \Delta t} + \frac{G_c}{2l} \right) w dV = 0. \quad (10)$$

More details about the derivation and implementation refer to [KT17, WFL*19b]. Then we discretize Eq. (10) and construct a linear system that has the same dimension with grid nodes N_g as

$$\mathbf{A} \mathbf{s} = \mathbf{e},$$

$$A_{i,j} = \sum_p V_p^n \left(\mathcal{K} q_i(\mathbf{x}_p) q_j(\mathbf{x}_p) + 2l (\nabla q_i(\mathbf{x}_p) \cdot \nabla q_j(\mathbf{x}_p)) \right), \quad (11)$$

$$e_i = \sum_p V_p^n \left(\frac{s_p^n}{M \Delta t} + \frac{G_c}{2l} \right) w_{ip}^n,$$

where \mathbf{A} is the symmetric positive definite coefficient matrix ($N_g \times N_g$), $\mathcal{K} = 2 \left(\frac{\beta}{\beta+p} - \theta \right) \mathcal{H} + \frac{G_c}{2l} + \frac{1}{M \Delta t}$ is a coefficient, $q_i(\mathbf{x}_p)$ is the MLS shape function in grid, \mathbf{s} is the vector ($N_g \times 1$) for phase-field

s_i^{n+1} , and \mathbf{e} is the constant vector ($N_g \times 1$). In general, with our GPU-based PCG solver, the evolution of grid phase-field will converge in a few iterations. We also adopt the mass lumping strategy in [WFL*19a] to reduce the computation cost.

4.3. Phase-field with Accumulated Plastic Strain

In computational physics, the ductile fracture is achieved by modifying and controlling the key parameters of phase-field evolution [BHL*16, DAS*18]. Inspired by the recent works [AKDL16, YK20], we first propose a novel degradation function with the influence of plasticity as

$$\begin{aligned} \hat{\Psi} &= g(s,p) \Psi^+(\mathbf{b}, J) + \Psi^-(J), \\ g(s,p) &= \left(\frac{\beta}{\beta+p} - \theta \right) s^2 + \theta, \end{aligned} \quad (12)$$

where $\theta = 0.01$ is a minimum residual to prevent the artifacts, and β is the relaxation parameter of p . Note that we usually set $\beta = 15$ to make sure that p will promote degradation under violent fracture.

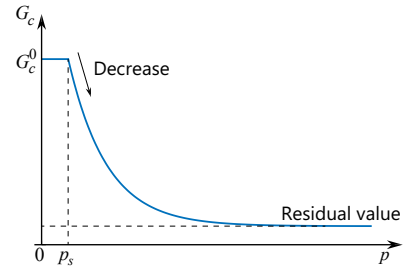


Figure 9: Fracture toughness. The fracture toughness G_c begins to decrease when $p = p_s$. As p increases, G_c continues decreasing until it stops at a residual value.

Then we propose a novel fracture toughness function that replaces the traditional fracture toughness scalar as

$$G_c(p) = \begin{cases} G_c^0 & p \leq p_s \\ G_c^0 \left(\frac{1-c}{\exp(p-p_s)} + c \right) & p > p_s \end{cases}, \quad (13)$$

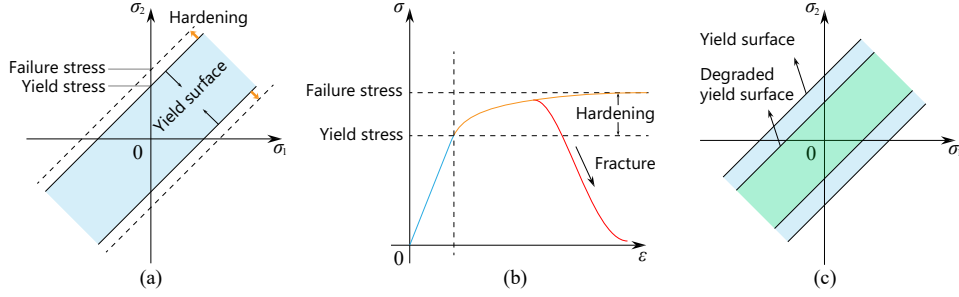


Figure 10: Novel von Mises yield condition. (a) The traditional von Mises yield surface and its hardening process. The yield stress is the initial size of the yield surface and the failure stress is the upper bound. (b) The stress-strain diagram, in which the orange line denotes the normal hardening process and the red line denotes the fracture process. (c) The normal yield surface and degraded one of the damaged material.

where G_c^0 is the initial fracture toughness, p_s is a critical plastic value and c is a residual coefficient. In phase-field evolution, the smaller G_c is, the more likely the material is to break. As shown in Fig. 9, once p reaches the critical value p_s , G_c begins to decrease. As a result, the sensitivity of the phase-field will increase with the accumulation of plastic strain. But to prevent numerical problems, we control the value of G_c is always higher than the fundamental residual value. In general, the smaller p_s is, the greater plastic effects on fracture surface damage is. Meanwhile, the smaller c is, the more serious the final damage state of the fracture surface is. We usually adapt to materials with different fracture toughness by controlling p_s and c . In this way, our phase-field evolution scheme follows the plastic hardening accurately, which leads the localized fracture as shown in Fig. 5.

4.4. Advanced Return Mapping with Degradation

Plastic treatment is another important part of PPF combined with fracture phase-field. Here we adopt a degraded von Mises yield criterion [AKDL16, BHL*16] to keep pace with the evolution of phase-field. The modified yield function and hardening law are given by

$$y(\tau) = \|\tau_{\text{dev}}^{tr}\|_F - g\sqrt{\frac{2}{6-d}}H(\alpha), \quad H(\alpha) = \sigma_f - \frac{\sigma_f - \sigma_y}{\exp(h\alpha)}, \quad (14)$$

where $\tau_{\text{dev}}^{tr} = g(s, p)\mu \text{dev}(\bar{\mathbf{b}}^{tr})$, σ_y is the yield stress, $\sigma_f = \sqrt{\frac{6-d}{2}}\sigma_y$ is the failure stress, and h is the hardening parameter that is usually adjusted according to the yield stress. As shown in Fig. 10 (a), we update the deformation gradient as $\mathbf{F}^{n+1} = \mathbf{F}^{tr}$ when the yield condition satisfies $y(\tau) \leq 0$. Otherwise, we need to project the elastic strain to the yield surface, which then grows in size according to the plastic flow rule, namely hardening. Note that in our PPF method, the modified yield surface will degrade with the function g when the material is damaged (Fig. 10 (c)). In the following, we will describe the projection process. According to the Lie derivative in §3.2.2, the discretization of the deviatoric Kirchhoff stress based on the associative plastic flow rule [Sim92, SM93] is given by

$$\tau_{\text{dev}}^{n+1} - \tau_{\text{dev}}^{tr} = -\frac{2\mu}{d}g(s, p)\Delta\gamma \text{tr}(\bar{\mathbf{b}}^{tr})\frac{\partial y}{\partial \tau}, \quad (15)$$

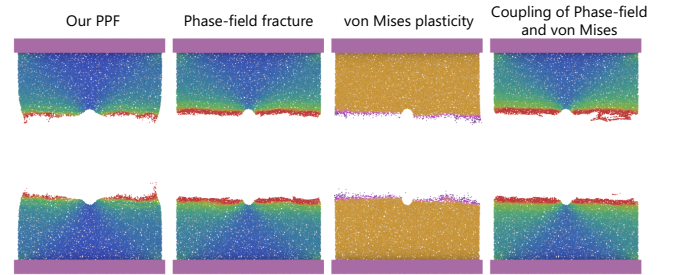


Figure 11: Comparison of different fracture methods. From left to right are our PPF method, phase-field method without plasticity [WFL*19a], von Mises plasticity, and the combination of phase-field fracture and classic von Mises plasticity in [WFL*19a].

where the direction of plastic flow is $\frac{\partial y}{\partial \tau} = \frac{\tau_{\text{dev}}^{n+1}}{\|\tau_{\text{dev}}^{n+1}\|}$, which leads the maximum plastic dissipation for associativity, and $\Delta\gamma$ is flow distance.

To solve Eq. (15) conveniently, we focus on the principal space. The SVD of deformation gradient is $\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}^T$. The Hencky strain and the corresponding Kirchhoff stress are given by

$$\hat{\epsilon}_i = \log(\Sigma_{ii}), \quad \hat{\tau} = \mathbf{C}\hat{\epsilon}, \quad (16)$$

where $\mathbf{C} = 2\mu\mathbf{I} + \lambda\mathbf{1}\mathbf{1}^T$ is the elastic modulus tensor, $\mathbf{1}$ is all ones vector, and $\hat{\mathbf{M}}$ denotes the eigenvalues vector of a matrix \mathbf{M} . So our main task is to solve $\hat{\epsilon}^{n+1}$ now. First we get a trial Kirchhoff stress $\hat{\tau}^{tr}$ from $\hat{\mathbf{F}}^{tr}$. Note that $\hat{\tau}^{n+1}$ and $\hat{\tau}^{tr}$ have the same direction. Combining the plastic flow Eq. (15) and the projected yield condition $y(\hat{\tau}^{n+1}) = 0$, we have

$$\|\hat{\tau}^{tr}\| - g\sqrt{\frac{2}{6-d}}H(\alpha^n + \sqrt{\frac{2}{6-d}}\Delta\gamma) - \frac{2\mu}{d}g\Delta\gamma \text{tr}(\bar{\mathbf{b}}^{tr}) = 0. \quad (17)$$

The above equation can be solved efficiently with the classic Newton method. Once we get $\Delta\gamma$, the plastic variables can be updated immediately as $\alpha^{n+1} = \alpha^n + \sqrt{\frac{2}{6-d}}\Delta\gamma$, $p^{n+1} = \frac{\alpha^{n+1}}{\alpha_c}$. Then we substitute $\Delta\gamma$ into Eq. (15), and the strain can be projected to the yield

surface:

$$\hat{\boldsymbol{\tau}}^{n+1} = \hat{\boldsymbol{\tau}}_{\text{dev}}^{n+1} + \frac{1}{d} \text{tr}(\boldsymbol{\tau}^{tr}) \mathbf{1}, \quad \boldsymbol{\varepsilon}^{n+1} = \mathbf{C}^{-1} \hat{\boldsymbol{\tau}}^{n+1}. \quad (18)$$

Finally the deformation gradient \mathbf{F}^{n+1} can be updated accordingly with Eq. (16). Therefore, in the above plastic treatment, we multiply the yield surface by the degradation function, which changes yield surface adaptively for materials with different damage degrees and prevents the purely elastic failure. In this manner, we achieve the tight coupling of phase-field fracture and plasticity, and produce better visual results for ductile fracture than the simple combination in [WFL*19a].

To summarize, our PPF method combines phase-field and plastic treatment, and tracks the variations of them. When an object deforms continuously, plastic strain accumulates locally, and the sensitivity of the phase-field increases. As a result, the cracks begin to appear, and the material damage occurs. Then our degraded yield surface continues to limit the elasticity of the damaged material until the material failure. Here we compare our method with other fracture approaches in Fig. 8 and Fig. 11. Our PPF method breaks through the limitation of Griffith's theory to accurately predict local fracture, with which we can observe the apparent phenomena of elastoplastic deformation, necking, and crack in ductile fracture.

5. GPU Optimization

In each time step of the PPF-MPM framework, there are many time-consuming processes, such as preprocessing, implicit solver, and phase-field evolution. Recently, Gao et al. [GWW*18] have developed efficient strategies for MPM simulation. First, they designed a warp-level reduction summation algorithm to decrease the write hazard within a SPGrid cell [SABS14, GWW*18], which actually improved the performance of P2G process. Second, the particles in MPM need to be reordered with the histogram sorting algorithm and re-mapped to the corresponding grid space at the reinitialization process of each time step. In these processes, the prefix summation algorithm is vitally important. GPU MPM directly calls the functions in the *trust* library of CUDA, which is efficient compared with other implementations, especially when the array is quite large. Although they have developed suitable solutions, there is still ample room for improvement. Here, we will introduce our novel reduction strategy, prefix summation algorithm, and their generalization in our PPF-MPM framework. We will also provide benchmarks for proving the effectiveness of our methods.

5.1. Efficient Reduction Algorithm based on CUDA Warps

To take the full advantages of memory hierarchical architecture in GPGPU, we design a much more general and efficient parallel reduction algorithm. As shown in Fig. 12 (for the convenience of illustration, we assume the size of a thread block (light blue) is 8, and warp size is 4), in each iteration, the data is loaded from global memory to registers (dark blue blocks) first, then it can be reduced in a warp by CUDA function *shfl_down*. Second, if the number of warp in a thread block is more than one, each reduction result of a CUDA warp will be transferred to shared memory for the reduction on full thread block. Finally, the data is loaded back to global memory for the next iteration until we finish the reduction

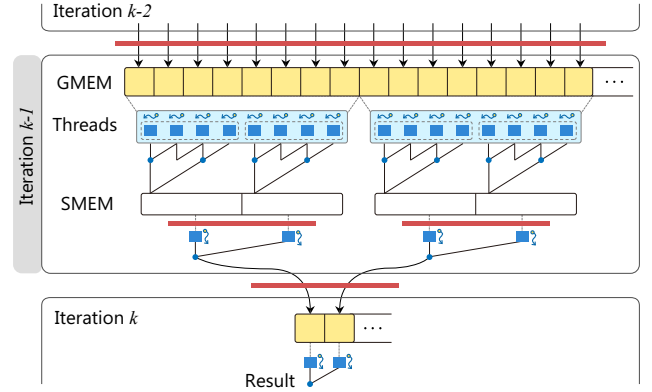


Figure 12: The iteration procedure of our reduction algorithm. Red lines denote the synchronous process, and dark blue blocks are registers. GMEM is the global memory, while SMEM is the shared memory.

Table 1: CFL condition benchmarks. The comparison between our method, the corresponding method in GPU MPM (best and worst case), and the algorithm for finding the maximum value in cublas library of CUDA.

#	GPU MPM		cublas library	Ours	Speedup
	best	worst			
10K	0.050	0.072	0.05621	0.00700	7.14×
100K	0.161	4.040	0.06480	0.01055	15.2×
1M	0.243	46.08	0.07034	0.01772	13.7×
5M	0.285	234.3	0.11194	0.05999	4.75×
10M	0.302	470.0	0.15568	0.10901	2.77×

for the full array. Moreover, we detail the implementation of each iteration in Alg. 1, where W is the CUDA warp (W_{iid} is the thread ID in a CUDA warp, W_n is the warp number, and W_{id} is the warp ID), and T_{id} is the thread ID.

In the algorithm of MPM, to make the calculation accurate and avoid numerical problems, the time step Δt should be calculated under CFL restriction in the reinitialization of each timestep. In this process, we need to find the maximum velocity of the particles in the simulation system, in which the time overhead cannot be ignored especially when the particle number is relatively small. Therefore, we apply our advanced reduction algorithm to the above process and document the time cost of our method and other corresponding algorithms in Table 1. For this process, GPU MPM employs an efficient max value finding algorithm based on CUDA atomic operation, but it lacks stability as their method is quite slow when each value of the array increases with index. Our approach is stable and efficient with the acceleration ratio from 2.77× to 15.2× compared with the best case of the method in GPU MPM. Moreover, our reduction algorithm can be also applied in some other pivotal processes, such as implicit solver and phase-field evolution, in which the PCG solver involves a lot of summation or maximum-finding operations.

Algorithm 1 Reduction Algorithm Based on CUDA Warps

```

1: procedure REDUCTION
2:    $W_{tid} = T_{id} \% 32$ ,  $W_{id} = T_{id} / 32$ ,  $stride = 1$ 
3:    $W_n \leftarrow$  Calculate the number of warp
4:    $temp \leftarrow$  Load from shared memory to registers
5:   while  $stride < 32$  do
6:     Conduct data operations
7:      $stride = 2 \times stride$ 
8:   if  $W_{id} = 0$  then
9:     Transfer data from registers to shared memory
10:  Synchronization
11:  if  $T_{id} \geq W_n$  then
12:    return
13:  if  $W_n > 1$  then
14:     $temp \leftarrow$  Load from shared memory to registers
15:     $stride = 1$ 
16:    while  $stride < W_n$  do
17:      Conduct data operations
18:       $stride = 2 \times stride$ 
19:    if  $T_{id} = 0$  then
20:      Transfer data from registers to global memory
    
```

Table 2: Prefix summation benchmarks. The comparison of time cost (ms) between our method and the corresponding algorithm in GPU MPM.

Particle #	<i>trust</i> library	Ours	Speedup
10K	0.048	0.010	4.8×
100K	0.052	0.018	2.9×
500K	0.057	0.026	2.19×
1M	0.067	0.045	1.49×
2.5M	0.094	0.093	1.01×
3M	0.102	0.109	0.94×

5.2. Optimized Prefix Summation Algorithm

In traditional GPU algorithms, direct thread mapping may cause many threads idle in parallel computing, which will unavoidably decrease the parallelism of thread groups and increase the number of thread blocks. To avoid this situation, we propose a new prefix summation algorithm, which makes each thread correspond to continuous computation tasks, so as to promise the high parallelism of thread groups and reduce the number of thread blocks.

Here we illustrate the data flow of the summation process in Fig. 13 and assume the size of the thread block is 2. First, the data in GMEM1 is loaded into the shared memory of each thread block, and then will be reduced, in which the mapping functions are

$$\begin{aligned}
 L_1 &= T_{id} + G_{id} \times 2^{S-1}, & L_2 &= 2^S \times G_{id} + 2^{S-1} - 1, \\
 \text{s.t. } G_{id} &= \frac{T_{id}}{2^{S-1}} + 1, & L_1 &< 2 \times B_s,
 \end{aligned} \tag{19}$$

where L_1 and L_2 is the location of the target data of a thread in shared memory, S is reduction depth of a thread block, G_{id} is the group ID, and B_s is the block size. Note that our mapping functions map threads to the continuous computation tasks. Then the data is transferred back to GMEM1, and the summation is

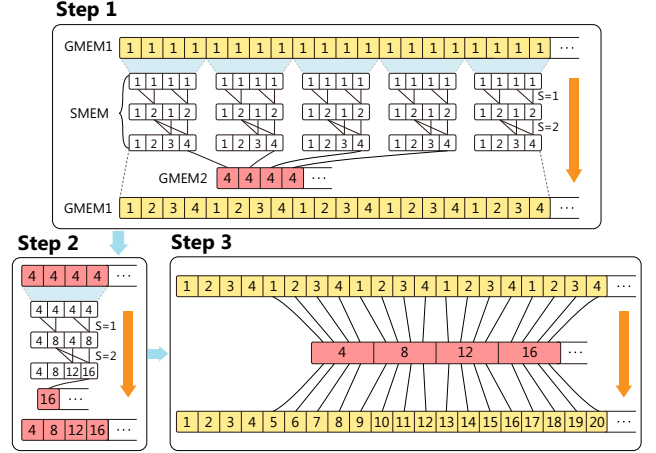


Figure 13: Optimized prefix summation algorithm. The orange arrows denote the merge direction of the data stream. GMEM1 and GMEM2 are two independent parts of global memory.

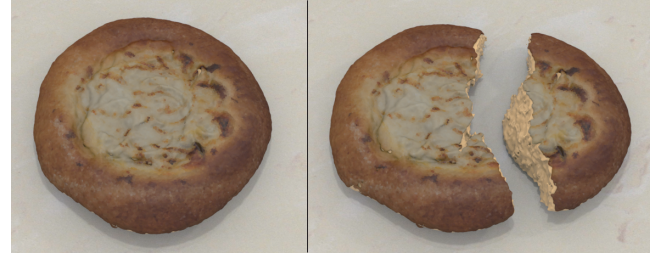


Figure 14: Pancake. A steamed pancake is torn apart.

recorded in GMEM2. Second, we conduct the same operations on GMEM2 to get the offset of each thread block (the offset of the first block is always 0). Finally, we add each data in GMEM1 with the corresponding offset value in GMEM2, so as to get the prefix summation result on the total array.

We give the benchmarks in Table 2, in which we compare our method with *trust* library of CUDA. Note that our method is better when the particle number is less than 2.5 million (most of the simulation scenarios). So in our implementation, we use our method for less than 2.5 million particles and *trust* library for more than 2.5 million particles. Benefiting from the performance improvement from frequent prefix summation operations in PPF MPM, our GPU optimization strategy shows significant speedup in our experiments where the array is usually less than 2.5 million.

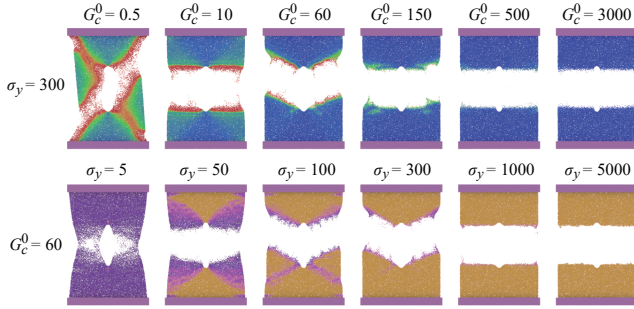
6. Results

We evaluate the effectiveness of our PPF method by a number of ductile fracture simulations. Our demos were performed on a desktop with Intel Core i7-9700K CPU at 3.60 GHz and NVIDIA GTX 2080TI. The parameter settings of the PPF method and the performance statistics are given in Table 3.

First of all, in Fig. 11, we show the comparison between our

Table 3: Parameters and performance. The parameters of each scenario and the computation time on GPU using explicit update for velocity. Scheme denotes the integration scheme on CPU. E is the Young's modulus and ν is the Poisson ratio.

Scene	Particle #	Grid domain	Δt_{step}	Δt_{frame}	Scheme	Material		Phase-field				von Mises			Time (ms)/step		
						E	ν	G_c	M	p_s	c	α_c	σ_y	h	Preprocess	MPM workflow	Total
(Fig. 2) Strip	1.03M	384 ³	1×10^{-4}	1/60	explicit	1000	0.4	20	15	0.01	0.1	0.1	100	2	2.381	4.342	6.723
(Fig. 4) Metal plate	500K	256 ³	3.75×10^{-5}	1/30	explicit	20000	0.3	200	15	0.001	0.02	0.01	14000	1	2.281	2.795	5.076
(Fig. 7) Dough slice	270K	128 ³	1.5×10^{-4}	1/50	implicit	1000	0.3	15	30	0.001	0.05	0.02	200	2	2.103	1.985	4.088
(Fig. 8) Workpiece	449K	256 ³	1×10^{-4}	1/60	explicit	1000	0.4	25	15	0.01	0.1	0.02	120	2	2.233	2.631	4.864
(Fig. 14) Pancake	596K	256 ³	5×10^{-5}	1/100	implicit	1000	0.3	50	15	0.01	0.1	0.02	200	2	2.326	2.868	5.194
(Fig. 17) Wheel	1.30M	384 ³	3.75×10^{-5}	1/120	implicit	1500	0.3	80	20	0.001	0.01	0.008	600	5	2.413	5.368	7.781
(Fig. 18) Rubber table	801K	256 ³	1×10^{-4}	1/100	explicit	15000	0.3	300	20	0.001	0.01	0.01	10500	25	2.362	3.541	5.903

**Figure 15: Results of different parameters for PPF.** By controlling the size of the yield surface σ_y and the initial fracture toughness value G_c^0 , our method shows various artistic effects during ductile fracture.

method, purely elastic phase-field fracture method [WFL*19a], the traditional von Mises model and the simple combination of phase-field fracture and von Mises plasticity [WFL*19a], as well as the visualization result of the phase-field and the accumulation of plastic strain. Then we also provide the 3D comparison in the tensile fracture experiment in Fig. 8. As shown in the right sub-figure, our method demonstrates the fracture toughness and irreversible plastic deformation during damage more realistically. However, the pure phase-field method (workpiece 2) [WFL*19a] only shows the release of fracture energy. After the material is damaged, the whole material returns to its original state. For the classical von Mises yield criterion (workpiece 3), the material will fracture directly when the elastic strain exceeds the threshold. We also conduct the simple combination of phase-field method and plastic treatment (workpiece 4) [WFL*19a], with which we can hardly capture the plastic necking. As a result, our method has apparent advantages in capturing the complete phenomena of ductile fracture.

In Fig. 15, we provide the visual effects of different parameter settings for our PPF method. The larger the G_c value, the less sensitive the phase-field, and the G_c value will decrease with local plastic deformation, so the initial value G_c^0 needs to be moderate. For the yield stress σ_y , we advocate adjustments based on solid material properties. The higher the yield stress, the more difficult plastic accumulation will occur. So we need to adjust it to match the changes in the phase-field, so as to simulate ductile fracture continuously. By the way of parameter control, our method can

be employed for the simulation of a wide range of elastoplastic materials.

Moreover, we present several examples with different patterns of external force. In Fig. 2, we twist and pull a plastic strip, where the plastic deformation of the strip in rotation and necking in pulling cannot recover. Then we demonstrate the tearing effects of flour food in Fig. 7 and Fig. 14, which resemble viscoelastic materials and are highly plastic. While in Fig. 18, an iron ball pierces the table and the local damage caused by the fracture is quite severe. Obviously, our method can produce various visual effects and crack patterns for ductile fracture.

To further validate the robustness of our method, we simulate a high-strength plastic plate penetrated by high-velocity bullets in Fig. 4 and severe material damage occurring under squeezing in Fig. 17. At last, Fig. 1 showcases the ductile fracture of complex multi-material hamburger in the middle sub-figure. All the above examples have proven our method will not encounter numerical fracture under extreme situations with high velocity and pressure.

On the other hand, in order to verify the effectiveness of our GPU optimization strategy, we have already presented two benchmark tests for our strategy in §5. Since our approach accelerates the MPM preprocessing and phase-field evolution, it can be combined with recent acceleration methods for MPM workflow in [WQS*20, GWW*18] to further improve the efficiency. Besides, we benchmark the overall PPF-MPM and traditional MPM in Fig. 16, and our optimization scheme can reach $1.10 \sim 1.56\times$ speedup. Also, for the demos shown in our paper, we document the average time cost of a single time step for each scenario in Table 3. Benefit from the parallel acceleration, our approach works well in general cases and mainstream computational platforms.

Limitations. We developed a novel PPF method and an efficient GPU optimization scheme, with which we could simulate various ductile fracture phenomena and accelerate the entire PPF-MPM on GPU. However, there are still several limitations pertaining to the proposed framework. The phase-field fracture method will cause unexpected damaging results under compression, and the present yield model is not suitable for brittle materials. As a result, our method cannot simulate more violent crack and breakage phenomena for brittle fracture. It would be necessary to design additional constraints for the phase-field evolution rather than simply modifying the underlying physical properties. In addition, the current GPU acceleration scheme is most applicable in the preprocessing of MPM. As the number of particles increases significantly, the

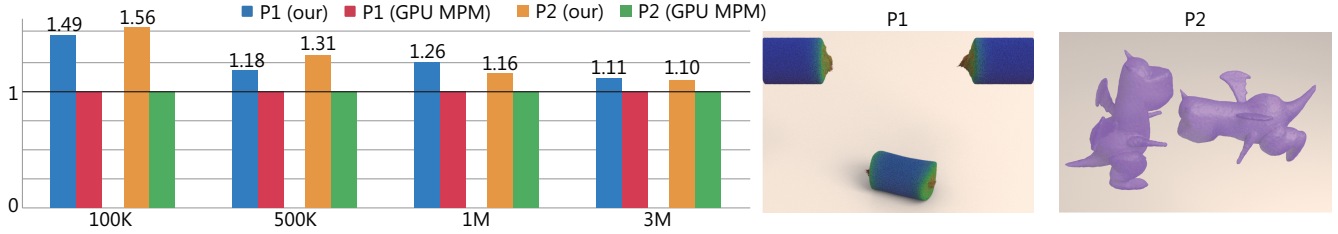


Figure 16: Benchmark experiments. The comparison between our GPU optimization strategy and GPU MPM. We present four groups of experiments with different numbers of particles and the normalized acceleration ratio. In each group, we present two comparative experiments of our PPF-MPM (P1) and traditional MPM with neo-Hookean model (P2).

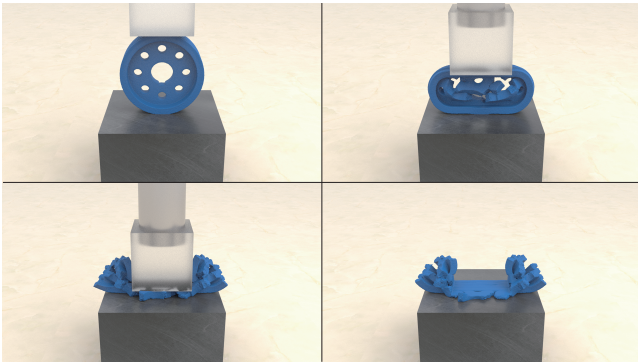


Figure 17: Wheel. A squashed wheel.

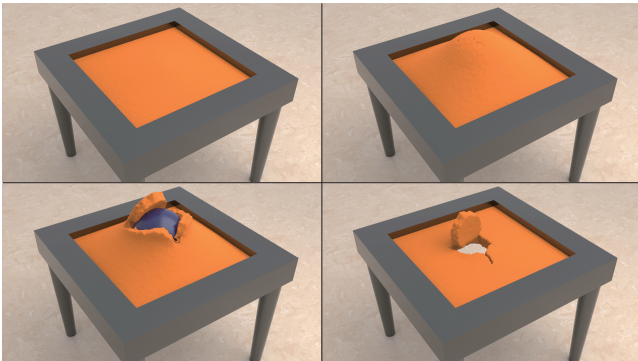


Figure 18: Rubber table. A rising iron ball causes the plastic table top to crack.

preprocessing time will account for reduced percentage of the simulation time, and the enhancing effect of our optimization strategy would be less significant.

7. Conclusions and Future Works

In this paper, we have detailed a novel plastic phase-field method that couples the fracture phase-field theory with von Mises yield criterion for ductile fracture simulation, whose performance was further enhanced by GPU optimization. Our new method involves the

novel hardening-relevant degradation function and fracture toughness function, based on the assumption that the accumulation of localized unrecoverable plastic deformation will increase the sensitivity of the fracture phase-field. We followed the associative flow rule and adopted a novel degraded von Mises yield criterion to accommodate the deformation state of healthy or damaged materials. To integrate the aforementioned physics model in the temporal domain, we employed a PPF subsystem in the MPM framework, which accurately manifests the unique elastoplasticity phenomena during ductile fracture, including necking, crack propagation, and failure. In addition, a GPU optimization scheme that takes the full advantages of GPU hierarchical architecture was adopted to improve our method’s numerical performance and maximize the potential of GPU threads and memory. Our extensive experimental results have confirmed the effectiveness and efficiency of the proposed method.

At present, we restrict that \mathcal{H} can only increase during simulation, but once this limit is broken, the material will heal if the tensile strain of the damaged material decreases. Therefore, by modifying this constraint, more complex dynamic fracture phenomena during intermediate states of the damaged materials could be simulated using the different processes of healing and damage of the materials. Besides, we plan to devote our efforts to the exploration of more meaningful plastic models (e.g., Cam-Clay based models) to achieve better visual realism for more dramatic natural phenomena such as avalanche and landslide, in which the phase-field could help exhibit the small-scale details of collapse and soil flow. The plastic models can be solved using either the associative or non-associative flow rule for many plastically deforming materials. In order to ensure physical accuracy and visual realism, we choose the associative flow rule for our von Mises plasticity, with which the strain can be projected onto the yield surface accurately. However, for complex yield surfaces such as Drucker-Prager and Cam-Clay, the projection optimization using the associative flow rule is quite complex and time-consuming. In this case, we tend to choose the non-associative flow rule to ensure the interaction performance and algorithm robustness, and satisfy the requirements of graphical applications in our future work. Furthermore, AoSoA and G2P2G [WQS*20] can be applied in our GPU framework to further decrease the time consumption of MPM workflow, so as to improve the speedup rate of the entire MPM algorithm.

Acknowledgments

The authors would like to especially thank all reviewers for their sincere and thoughtful suggestions. This paper is partially supported by Natural Science Foundation of China under Grants 61532002, 61672237, 62072183, 62002121 and 61672077, National Science Foundation of USA (IIS-1715985 and IIS-1812606).

Appendix A: Pseudocode

Here we provide the pseudocode of PPF-MPM, including phase-field evolution and plastic processing.

Algorithm 2 PPF-MPM

```

1: procedure MPMP2G
2:   for each grid node do
3:      $m_i^n = \sum_p w_{ip}^n m_p$ 
4:      $\mathbf{v}_i^n = \frac{1}{m_i^n} \sum_p w_{ip}^n m_p (\mathbf{v}_i^n + \mathbf{C}_p^n (\mathbf{x}_i - \mathbf{x}_p^n))$ 
5: procedure PPFPHASEFIELD SOLVER
6:   //Update fracture toughness
7:   for each particle do
8:     if  $p > p_s$  then
9:        $G_c = G_c^0 (\frac{1-c}{\exp(p-p_s)} + c)$ 
10:  //Phase-field P2G
11:  for each grid node do
12:     $s_i^n = \sum_p w_{ip}^n s_p^n / (\sum_p w_{ip}^n)$ 
13:  //Solve phase-field
14:  Construct  $\mathbf{A}_s = \mathbf{e}$  as Eq. (11) and solve it with GPU-based PCG
15:  //Phase-field G2P
16:  for each particle do
17:     $s_p^{n+1} = \max(\min(s_p^n, \sum_i w_{ip}^n s_i^{n+1}), 0)$ 
18:     $g(s, p) = (\frac{\beta}{\beta+p} - \theta) s^2 + \theta$ 
19: procedure FORCECOMPUTATION
20:  if symplectic then
21:     $\mathbf{f}_i^n = -\sum_p w_{ip}^n V_p \frac{4}{\Delta x^2} \sigma_p(\mathbf{F}_p^n, s_p^{n+1})(\mathbf{x}_i - \mathbf{x}_p^n)$ 
22:  else if implicit then
23:     $\mathbf{f}_i^{n+1} = -\sum_p w_{ip}^n V_p \frac{4}{\Delta x^2} \sigma_p(\mathbf{F}_p^{n+1}, s_p^{n+1})(\mathbf{x}_i - \mathbf{x}_p^n)$ 
24:  Update grid velocity
25: procedure MPMG2P
26:  for each particle do
27:     $\mathbf{v}_p^{n+1} = \sum_i w_{ip}^n \mathbf{v}_i^{n+1}$ 
28:     $\mathbf{C}_p^{n+1} = \frac{4}{\Delta x^2} \sum_i w_{ip}^n \mathbf{v}_i^{n+1} (\mathbf{x}_i^n - \mathbf{x}_p^n)^T$ 
29:     $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}$ 
30: procedure DEFORMATIONGRADIENTUPDATE
31:  for each particle do
32:     $\mathbf{F}_p^{tr} = (\mathbf{I} + \Delta t \mathbf{C}_p^{n+1}) \mathbf{F}_p^n$ 
33:  Traditional MPM step with return mapping

```

References

[AGDL15] AMBATI M., GERASIMOV T., DE LORENZIS L.: Phase-field modeling of ductile fracture. *Computational Mechanics* 55, 5 (2015), 1017–1040. 4

Algorithm 3 von Mises Plasticity with Phase-field Degradation

```

1: procedure RETURNMAPPING
2:    $\mathbf{U}, \boldsymbol{\Sigma}, \mathbf{V}^T = \text{SVD}(\mathbf{F}_p^{tr})$ 
3:    $\mathbf{C} = 2\mu \mathbf{I} + \lambda \mathbf{1}\mathbf{1}^T$ 
4:    $\hat{\boldsymbol{\epsilon}}_i = \log(\boldsymbol{\Sigma}_{ii})$  for  $i = 0, 1, 2$ 
5:    $\hat{\boldsymbol{\tau}}^{tr} = \mathbf{C} \hat{\boldsymbol{\epsilon}}$ 
6:    $\hat{\boldsymbol{\tau}}_{\text{dev}}^{tr} = g(s, p) (\hat{\boldsymbol{\tau}}^{tr} - \frac{\text{tr}(\hat{\boldsymbol{\tau}}^{tr})}{d} \mathbf{1}\mathbf{1}^T)$ 
7:    $y = \|\hat{\boldsymbol{\tau}}_{\text{dev}}^{tr}\|_F - g \sqrt{\frac{2}{6-d}} H(\alpha)$ 
8:   if  $y < 0$  then
9:     return  $\mathbf{F}_p^{n+1} = \mathbf{U} \boldsymbol{\Sigma} \mathbf{V}^T$  // Inside the yield surface
10:  Solve Eq. (17) with classic Newton's method.
11:  // Update plastic variables
12:   $\alpha^{n+1} = \alpha^n + \sqrt{\frac{2}{6-d}} \Delta \gamma$ ,  $p^{n+1} = \frac{\alpha^{n+1}}{\alpha_c}$ 
13:   $\hat{\boldsymbol{\tau}}_{\text{dev}}^{n+1} = \hat{\boldsymbol{\tau}}_{\text{dev}}^{tr} - \Delta \gamma \frac{2}{d} g \text{tr}(\hat{\boldsymbol{\tau}}_{\text{dev}}^{tr}) \frac{\hat{\boldsymbol{\tau}}_{\text{dev}}^{tr}}{\|\hat{\boldsymbol{\tau}}_{\text{dev}}^{tr}\|}$ 
14:   $\hat{\boldsymbol{\tau}}^{n+1} = \hat{\boldsymbol{\tau}}_{\text{dev}}^{n+1} + \frac{1}{d} \text{tr}(\hat{\boldsymbol{\tau}}^{tr})$ 
15:   $\boldsymbol{\epsilon}^{n+1} = \mathbf{C}^{-1} \hat{\boldsymbol{\tau}}^{n+1}$ 
16:   $\boldsymbol{\Sigma}_{ii}^{n+1} = \exp(\hat{\boldsymbol{\epsilon}}_i^{n+1})$  for  $i = 0, 1, 2$ 
17:  return  $\mathbf{F}_p^{n+1} = \mathbf{U} \boldsymbol{\Sigma}^{n+1} \mathbf{V}^T$ 

```

[AKDL16] AMBATI M., KRUSE R., DE LORENZIS L.: A phase-field model for ductile fracture at finite strains and its experimental verification. *Computational Mechanics* 57, 1 (2016), 149–167. 3, 4, 6, 7

[Ali97] ALIABADI M. H.: Boundary Element Formulations in Fracture Mechanics. *Applied Mechanics Reviews* 50, 2 (1997), 83–96. 2

[AMM09] AMOR H., MARIGO J.-J., MAURINI C.: Regularized formulation of the variational brittle fracture with unilateral contact: Numerical experiments. *Journal of the Mechanics and Physics of Solids* 57, 8 (2009), 1209–1229. 4

[Bat06] BATHE K.-J.: *Finite element procedures*. Klaus-Jurgen Bathe, 2006. 6

[BFM00] BOURDIN B., FRANCFORT G. A., MARIGO J.-J.: Numerical experiments in revisited brittle fracture. *Journal of the Mechanics and Physics of Solids* 48, 4 (2000), 797–826. 3, 4, 6

[BHL*16] BORDEN M. J., HUGHES T. J., LANDIS C. M., ANVARI A., LEE I. J.: A phase-field formulation for fracture in ductile materials: Finite deformation balance law derivation, plastic degradation, and stress triaxiality effects. *Computer Methods in Applied Mechanics and Engineering* 312 (2016), 130–166. 6, 7

[BHTF07] BAO Z., HONG J., TERAN J., FEDKIW R.: Fracturing rigid materials. *IEEE Trans. Vis. Comput. Graph.* 13, 2 (2007), 370–378. 2

[Bor12] BORDEN M. J.: *Isogeometric analysis of phase-field models for dynamic brittle and ductile fracture*. PhD thesis, 2012. 2, 4

[CWXQ13] CHEN F., WANG C., XIE B., QIN H.: Flexible and rapid animation of brittle fracture using the smoothed particle hydrodynamics formulation. *Comp. Anim. Virt. Worlds* 24, 3-4 (2013), 215–224. 2

[CYFW14] CHEN Z., YAO M., FENG R., WANG H.: Physics-inspired adaptive fracture refinement. *ACM Trans. Graph.* 33, 4 (2014), 113:1–113:7. 2

[DAS*18] DITTMANN M., ALDAKHEEL F., SCHULTE J., WRIGGERS P., HESCH C.: Variational phase-field formulation of non-linear ductile fracture. *Computer Methods in Applied Mechanics and Engineering* 342 (2018), 71–94. 6

[DHW*19] DING M., HAN X., WANG S., GAST T. F., TERAN J. M.: A thermomechanical material point method for baking and cooking. *ACM Trans. Graph.* 38, 6 (2019), 192:1–192:14. 3

- [FLGJ19] FANG Y., LI M., GAO M., JIANG C.: Silly rubber: an implicit material point method for simulating non-equilibrated viscoelastic and elastoplastic solids. *ACM Trans. Graph.* 38, 4 (2019), 118:1–118:13. 2
- [FM98] FRANCFORT G. A., MARIGO J.-J.: Revisiting brittle fracture as an energy minimization problem. *Journal of the Mechanics and Physics of Solids* 46, 8 (1998), 1319–1342. 4
- [GHF*18] GUO Q., HAN X., FU C., GAST T. F., TAMSTORF R., TERAN J.: A material point method for thin shells with frictional contact. *ACM Trans. Graph.* 37, 4 (2018), 147:1–147:15. 2
- [GMD13] GLONDU L., MARCHAL M., DUMONT G.: Real-time simulation of brittle fracture using modal analysis. *IEEE Trans. Vis. Comput. Graph.* 19, 2 (2013), 201–209. 2
- [GPH*18] GAO M., PRADHANA A., HAN X., GUO Q., KOT G., SIFAKIS E., JIANG C.: Animating fluid sediment mixture in particle-laden flows. *ACM Trans. Graph.* 37, 4 (2018), 149:1–149:11. 2
- [GSSP10] GOSWAMI P., SCHLEGEL P., SOLENTHALER B., PAJAROLA R.: Interactive SPH simulation and rendering on the GPU. In *Proceedings of the 2010 Eurographics/ACM SIGGRAPH Symposium on Computer Animation* (2010), Eurographics Association, pp. 55–64. 3
- [GWW*18] GAO M., WANG X., WU K., PRADHANA A., SIFAKIS E., YUKSEL C., JIANG C.: GPU optimization of material point methods. *ACM Trans. Graph.* 37, 6 (2018), 254:1–254:12. 2, 3, 8, 10
- [HFG*18] HU Y., FANG Y., GE Z., QU Z., ZHU Y., PRADHANA A., JIANG C.: A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Trans. Graph.* 37, 4 (2018), 150:1–150:14. 5
- [HJST13] HEGEMANN J., JIANG C., SCHROEDER C., TERAN J. M.: A level set method for ductile fracture. In *The ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (2013), ACM, pp. 193–201. 2
- [HRZ*19] HUANG K., RUAN J., ZHAO Z., LI C., WANG C., QIN H.: A general novel parallel framework for sph-centric algorithms. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 2, 1 (2019), 1–16. 3
- [HW15] HAHN D., WOJTAN C.: High-resolution brittle fracture simulation with boundary elements. *ACM Trans. Graph.* 34, 4 (2015), 151:1–151:12. 2
- [HW16] HAHN D., WOJTAN C.: Fast approximations for boundary element based brittle fracture simulation. *ACM Trans. Graph.* 35, 4 (2016), 104:1–104:11. 2
- [HZI*20] HUANG K., ZHAO Z., LI C., WANG C., QIN H.: Novel hierarchical strategies for sph-centric algorithms on gpgpu. *Graphical Models* (08 2020), 101088. 3
- [JGT17] JIANG C., GAST T. F., TERAN J.: Anisotropic elastoplasticity for cloth, knit and hair frictional contact. *ACM Trans. Graph.* 36, 4 (2017), 152:1–152:14. 2
- [JSS*15] JIANG C., SCHROEDER C., SELLE A., TERAN J., STOMAKHIN A.: The affine particle-in-cell method. *ACM Trans. Graph.* 34, 4 (2015), 51:1–51:10. 5
- [KGP*16] KLÁR G., GAST T. F., PRADHANA A., FU C., SCHROEDER C., JIANG C., TERAN J.: Drucker-prager elastoplasticity for sand animation. *ACM Trans. Graph.* 35, 4 (2016), 103:1–103:12. 2, 3
- [KT17] KAKOURIS E., TRIANTAFYLLOU S. P.: Phase-field material point method for brittle fracture. *International Journal for Numerical Methods in Engineering* 112, 12 (2017), 1750–1776. 2, 3, 5, 6
- [LGL*19] LI M., GAO M., LANGLOIS T. R., JIANG C., KAUFMAN D. M.: Decomposed optimization time integrator for large-step elastodynamics. *ACM Trans. Graph.* 38, 4 (2019), 70:1–70:10. 2
- [LWQ15] LI C., WANG C., QIN H.: Novel adaptive SPH with geometric subdivision for brittle fracture animation of anisotropic materials. *The Visual Computer* 31, 6–8 (2015), 937–946. 2
- [MG04] MÜLLER M., GROSS M. H.: Interactive virtual materials. In *Proceedings of the Graphics Interface Conference* (2004), vol. 62, Canadian Human-Computer Communications Society, pp. 239–246. 2
- [MHW10] MIEHE C., HOFACKER M., WELSCHINGER F.: A phase field model for rate-independent crack propagation: Robust algorithmic implementation based on operator splits. *Computer Methods in Applied Mechanics and Engineering* 199, 45–48 (2010), 2765–2778. 3
- [OBH02] O'BRIEN J. F., BARGTEIL A. W., HODGINS J. K.: Graphical modeling and animation of ductile fracture. *ACM Trans. Graph.* 21, 3 (2002), 291–294. 2
- [OH99] O'BRIEN J. F., HODGINS J. K.: Graphical modeling and animation of brittle fracture. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (1999), ACM, pp. 137–146. 2
- [PNdJO14] PFAFF T., NARAIN R., DE JOYA J. M., O'BRIEN J. F.: Adaptive tearing and cracking of thin sheets. *ACM Trans. Graph.* 33, 4 (2014), 110:1–110:9. 2
- [RGJ*15] RAM D., GAST T. F., JIANG C., SCHROEDER C., STOMAKHIN A., TERAN J., KAVEHPOUR P.: A material point method for viscoelastic fluids, foams and sponges. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (2015), ACM, pp. 157–163. 2
- [SABS14] SETALURI R., AANJANEYA M., BAUER S., SIFAKIS E.: Spgrid: a sparse paged grid structure applied to adaptive smoke simulation. *ACM Trans. Graph.* 33, 6 (2014), 205:1–205:12. 8
- [Sim92] SIMO J. C.: Algorithms for static and dynamic multiplicative plasticity that preserve the classical return mapping schemes of the infinitesimal theory. *Computer Methods in Applied Mechanics and Engineering* 99, 1 (1992), 61–112. 7
- [SM93] SIMO J., MESHKE G.: A new class of algorithms for classical plasticity extended to finite strains. application to geomaterials. *Computational Mechanics* 11, 4 (1993), 253–278. 7
- [SSC*13] STOMAKHIN A., SCHROEDER C., CHAI L., TERAN J., SELLE A.: A material point method for snow simulation. *ACM Trans. Graph.* 32, 4 (2013), 102:1–102:10. 2, 5
- [SZS95] SULSKY D., ZHOU S.-J., SCHREYER H. L.: Application of a particle-in-cell method to solid mechanics. *Computer Physics Communications* 87, 1–2 (1995), 236–252. 2
- [TF88] TERZOPOULOS D., FLEISCHER K. W.: Modeling inelastic deformation: viscoelasticity, plasticity, fracture. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques* (1988), ACM, pp. 269–278. 2
- [TGK*17] TAMPUBOLON A. P., GAST T. F., KLÁR G., FU C., TERAN J., JIANG C., MUSETH K.: Multi-species simulation of porous sand and water mixtures. *ACM Trans. Graph.* 36, 4 (2017), 105:1–105:11. 2
- [TN04] TRUESDELL C., NOLL W.: The non-linear field theories of mechanics. In *The Non-linear Field Theories of Mechanics*. Springer, 2004, pp. 1–579. 3
- [WDG*19] WANG S., DING M., GAST T. F., ZHU L., GAGNIERE S., JIANG C., TERAN J. M.: Simulation and visualization of ductile fracture with the material point method. *Proc. ACM Comput. Graph. Interact. Tech.* 2, 2 (2019), 18:1–18:20. 2
- [WFL*19a] WOLPER J., FANG Y., LI M., LU J., GAO M., JIANG C.: CD-MPM: continuum damage material point methods for dynamic fracture animation. *ACM Trans. Graph.* 38, 4 (2019), 119:1–119:15. 2, 3, 4, 5, 6, 7, 8, 10
- [WFL*19b] WOLPER J., FANG Y., LI M., LU J., GAO M., JIANG C.: CD-MPM: continuum damage material point methods for dynamic fracture animation: Supplemental document. *ACM Trans. Graph.* (2019). 6
- [WQS*20] WANG* X., QIU* Y., SLATTERY S. R., FANG Y., LI M., ZHU S.-C., ZHU Y., TANG M., MANOCHA D., JIANG C.: A massively parallel and scalable multi-gpu material point method. *ACM Trans. Graph.* 39, 4 (2020). 3, 10, 11
- [YCL*17] YANG T., CHANG J., LIN M. C., MARTIN R. R., ZHANG J. J., HU S.: A unified particle system framework for multi-phase, multi-material visual simulations. *ACM Trans. Graph.* 36, 6 (2017), 224:1–224:13. 3

- [YCR*15] YANG T., CHANG J., REN B., LIN M. C., ZHANG J., HU S.: Fast multiple-fluid simulation using helmholtz free energy. *ACM Trans. Graph.* 34, 6 (2015), 201:1–201:11. [3](#)
- [YK20] YIN B., KALISKE M.: A ductile phase-field model based on degrading the fracture toughness: Theory and implementation at small strain. *Computer Methods in Applied Mechanics and Engineering* 366 (2020), 113068. [6](#)
- [YSB*15] YUE Y., SMITH B., BATTY C., ZHENG C., GRINSPUN E.: Continuum foam: A material point method for shear-dependent flows. *ACM Trans. Graph.* 34, 5 (2015), 160:1–160:20. [2](#)
- [ZZL*17] ZHU F., ZHAO J., LI S., TANG Y., WANG G.: Dynamically enriched MPM for invertible elasticity. *Comput. Graph. Forum* 36, 6 (2017), 381–392. [2](#)