



Effective time step restrictions for explicit MPM simulation

Yunxin Sun¹ and Tamar Shinar²  and Craig Schroeder² 

¹Tongji University, China

²University of California, Riverside

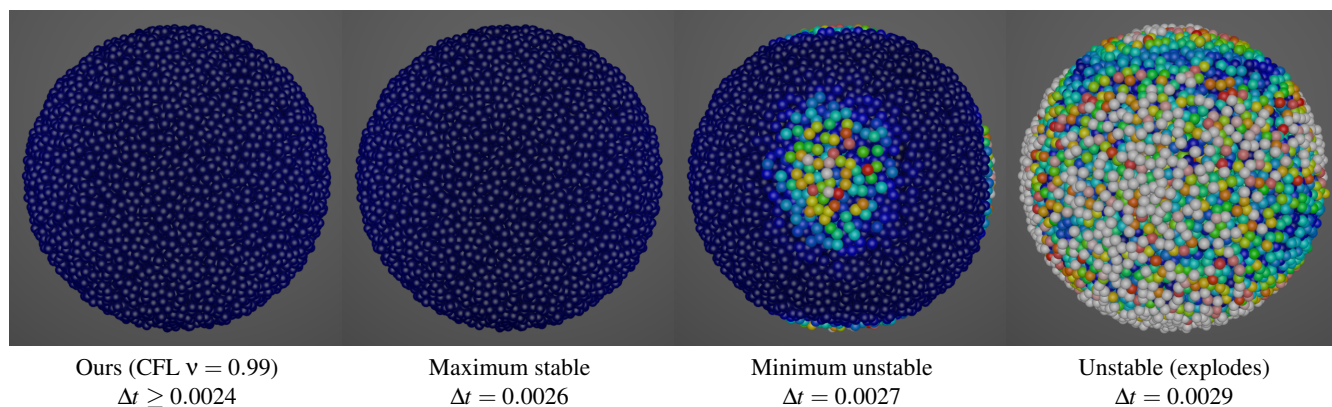


Figure 1: Stability of a perturbed elastic sphere. Using a novel analysis of the nonlinear elastic problem, our method gives a time step restriction which maintains stability throughout the simulation, yet is only $\sim 10\%$ smaller than the empirically determined stability threshold on this test, eliminating the need for costly trial and error. Colors represent velocity magnitude; black is smallest, increasing through the rainbow from blue to red, with white as the largest.

Abstract

Time steps for explicit MPM simulation in computer graphics are often selected by trial and error due to the challenges in automatically selecting stable time step sizes. Our time integration scheme uses time step restrictions that take into account forces, collisions, and even grid-to-particle transfers calculated near the end of the time step. We propose a novel set of time step restrictions that allow a time step to be selected that is stable, efficient to compute, and not too far from optimal. We derive the general solution for the sound speed in nonlinear isotropic hyperelastic materials, which we use to enforce the classical CFL time step restriction. We identify a single-particle instability in explicit MPM integration and propose a corresponding time step restriction in the fluid case. We also propose a reflection-based boundary condition for domain walls that supports separation and accurate Coulomb friction while preventing particles from penetrating the domain walls.

CCS Concepts

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

1. Introduction

Computer graphics has traditionally had a complex relationship with explicit methods and implicit methods. Explicit methods are generally simpler, do not require complex solvers, and tend to be quite efficient when the materials involved are not too stiff. Implicit methods became popular in graphics starting with [BW98], which used backward Euler to take large time steps in stiff cloth simulations. Since then, computer graphics has embraced a combination of explicit and implicit methods, depending on phenomena, stiff-

ness, algorithm complexity, and amenability to an efficient implicit formulation. Recently, the material point method (MPM) [SSC*13] has gained popularity in computer graphics for simulating a variety of materials, with both implicit and explicit formulations proposed.

Choosing effective time step sizes for explicit methods is a long-standing problem [Bra16]. In practice, they are often chosen by trial and error, with the largest size that produces acceptable results then selected and used for reporting efficiency. However, the process of trial and error itself can be costly and produce suboptimal results,

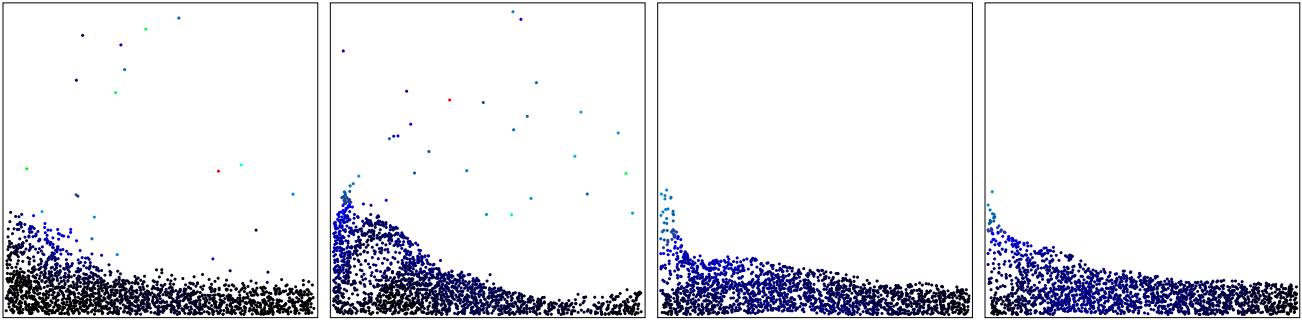


Figure 2: Both the sound speed CFL and single-particle stability criteria are needed to maintain stability in this splashing fluid simulation. (Left) The heuristic time step restrictions on velocity, displacement, and deformation gradient alone are insufficient for stability. (Middle, left) Additionally enforcing the CFL fails to stabilize the simulation. (Middle, right) The simulation is stable using all five stability restrictions. (Right) Using a multiplier of 1.3 for the single-particle time step restriction, the simulation is at the edge of stability.

making it undesirable for artistic tools. Choosing time steps for implicit methods is generally easier due to improved stability; a few time steps per frame (sometimes just one) often suffices. In both explicit and implicit methods, using a fixed time step size is common. The main goal of this paper is to provide a sound basis for adaptively selecting time steps for explicit MPM simulations, thus eliminating the need for trial and error, improving efficiency, and gaining control over the causes of instability.

We generally observe three causes of instability in explicit MPM simulations: (1) Failure to abide by the CFL restriction, which states that the numerical speed of information propagation must be at least as large as the physical speed of information propagation. Though the CFL condition is prevalent in explicit compressible flow simulations, accounting for it has been problematic for solids due to the lack of published wave speeds for common constitutive models. We give a novel derivation and an efficient algorithm to calculate wave speeds for any general hyperelastic isotropic constitutive model. (2) The hybrid nature of MPM introduces instabilities. We identify one such instability associated with isolated particles and derive a time step restriction to avoid it. (3) Postprocessing steps such as collisions can introduce large amounts of energy into a system by changing strain measures such as the deformation gradient. Along with restrictions on velocity and displacement, we restrict the amount of change allowed to the deformation gradient.

The time step restrictions that we introduce are very effective at stabilizing explicit MPM simulations. We have not tuned any of our time step sizes or CFL multipliers, nor introduced damping or any other type of intervention. We note that there is often a gap between time step sizes that produce good results and time step sizes that lead to simulation failure. Time step sizes within this gap lead to ill-behaved simulations; we consider these time steps to be unstable as well.

Additionally, we introduce a novel reflection boundary condition for MPM based on [DSS19], reformulating their method to conserve mass and momentum, and extending it to handle Coulomb friction and separation.

2. Related Work

The material point method was originally devised as an explicit method [SCS94, SZS95]. As hybrid particle-grid methods developed, they were found to suffer from a number of instabilities [Oku72, Bra88, BL98, BK04, SKB08, Gri14]. These were partially alleviated through better interpolation functions [BK04, WG08]. The first implicit methods for MPM were developed significantly later to avoid instabilities and to take larger time steps [GW01, GW03, SK04, LS06]. Nevertheless, explicit methods remain very common within the wider MPM community.

Within computer graphics, MPM was introduced by [SSC*13], which proposed an implicit formulation for simulating snow. Further MPM work in graphics has focused on simulating new phenomena, including foams [YSB*15, RGJ*15], melting and freezing [SSJ*14], sponges [RGJ*15], and sand [DBD16, KGP*16], reducing simulation noise and dissipation through APIC [JSS*15], midpoint rule [JST17], and PolyPIC [FGG*17], and improving integration schemes [GSS*15, WLF*20]. Most MPM methods in graphics are implicit, with [YSB*15, KGP*16, YSC*18, FHHJ18] being notable exceptions. Recent work has introduced the use of a sound speed estimate to predict a stable time step size for asynchronous time integration [FHHJ18]. Their estimate is based on the exact sound speed for linear elasticity at the rest configuration, and thus may differ quite significantly from the physical sound speed, as we show in Section 6. Adaptive time step restrictions were used to improve performance of SPH simulations in [IAGT10], which imposed time step restrictions based on maximum particle velocity and acceleration and the kernel radius. They additionally considered density errors in adjusting the time step size, but did not directly account for the sound speed. As a general rule, time step restrictions depend on a characteristic simulation length scale, which is often the grid Δx for MPM, kernel radius for SPH, and mesh edge lengths for mesh-based Lagrangian methods. All should account for the sound speed when simulated explicitly in order to satisfy the CFL condition.

One of the strengths of MPM is its ability to treat complex materials like sand, snow, or gel, where plasticity often plays an important role. Frictional contacts can also be modeled directly using

plasticity [JGT17, GHF*18, DS19, HGG*19]. The use of plasticity models generally complicates implicit formulations. One approach is to simply lag the plasticity [SSC*13, GPH*18], but this can significantly alter the physical behavior of some models. Alternatively, the plasticity can be included directly within a Newton solver, which produces asymmetrical systems that are generally difficult to solve [KGP*16, DS19], though [TGG*17] has suggested a compromise between the options. Alternative formulations can also avoid the problem [DBD16]. Ultimately, explicit formulations are generally preferred when the problem stiffness is manageable.

Existing object collision treatments for MPM are mostly based on [SSC*13], where collisions between simulated and non-simulated objects are handled by projecting velocities at nodes that are inside a level set. This treatment allows particles to penetrate about a grid cell into objects before completely stopping, but otherwise the treatment is simple and effective. We use it when colliding with objects that are not the domain boundary. We note that other collision treatments also exist, such as an extension for cutting [HFG*18], a penalty collision treatment for implicit integration [DS19], and a special treatment for sand [DBD16]. For domain boundaries, reflection boundary conditions for MPM simulation of fluids were introduced in [DSS19], and give the basis for our reflection boundary conditions. Unlike existing level set treatments for MPM, the reflection boundary conditions prevent particles from penetrating domain boundaries.

3. Algorithm Overview

We first give an overview of our proposed time integration scheme, shown in Algorithm 1. We follow the general scheme of [SSC*13], but use explicit symplectic Euler for the grid-based time integration and APIC for transfers between the particles and grid [JSS*15]. The steps in Algorithm 1 are as follows:

1. **Initial time step selection.** In Line 2, we obtain an initial time step size using the particle velocities and accounting for the ef-

Algorithm 1 Time step.

```

1: procedure TIME_STEP
2:    $\Delta t \leftarrow \text{VELOCITY\_DT}$  ▷ §4.2
3:    $(m_i^n, \mathbf{v}_i^n) \leftarrow \text{PARTICLE\_TO\_GRID}(m_p, \mathbf{v}_p, \mathbf{B}_p)$ 
4:    $\tilde{\mathbf{v}}_i^{n+1} \leftarrow \mathbf{v}_i^n + \Delta t (m_i^n)^{-1} \mathbf{f}_i$ 
5:    $\tilde{\mathbf{v}}_i^{n+1} \leftarrow \text{OBJECT\_COLLISIONS}(\tilde{\mathbf{v}}_i^{n+1})$ 
6:    $\tilde{\mathbf{v}}_i^{n+1} \leftarrow \text{REFLECTION\_BC}(m_i^n, \tilde{\mathbf{v}}_i^{n+1})$  ▷ §5
7:    $\Delta t_x \leftarrow \text{POSITION\_DT}$  ▷ §4.3
8:    $\Delta t_F \leftarrow \text{DEFORMATION\_GRADIENT\_DT}$  ▷ §4.4
9:    $\Delta t_c \leftarrow \text{SOUND\_SPEED\_DT}$  ▷ §4.1
10:   $\Delta t_p \leftarrow \text{SINGLE\_PARTICLE\_DT}$  ▷ §4.5
11:   $\Delta t' \leftarrow \min(\Delta t_x, \Delta t_F, \Delta t_c, \Delta t_p)$ 
12:  if  $\Delta t' < \Delta t$  then
13:     $\tilde{\mathbf{v}}_i^{n+1} \leftarrow \mathbf{v}_i^n + \frac{\Delta t'}{\Delta t} (\tilde{\mathbf{v}}_i^{n+1} - \mathbf{v}_i^n)$ 
14:     $\Delta t \leftarrow \Delta t'$ 
15:   $\tilde{\mathbf{x}}_i^{n+1} = \mathbf{x}_i^n + \Delta t \tilde{\mathbf{v}}_i^{n+1}$ 
16:   $(\mathbf{v}_p^{n+1}, \mathbf{B}_p^{n+1}) \leftarrow \text{GRID\_TO\_PARTICLE}(\tilde{\mathbf{v}}_i^{n+1})$ 
17:  UPDATE\_PLASTICITY
18:   $t \leftarrow t + \Delta t$ 

```

fect of APIC transfers, as described in Section 4.2. Although this estimate is usually too large to maintain stability overall, it suffices for computing forces and processing collisions.

2. **Transfer data from particles to the grid.** Here, we describe Line 3, reviewing some details of MPM [SSC*13] and APIC [JSS*15] for completeness. In APIC, each particle carries a mass m_p , position \mathbf{x}_p , velocity \mathbf{v}_p , deformation gradient \mathbf{F}_p , and affine state \mathbf{B}_p . First, mass is transferred from particles to the grid nodes, indexed by i , using $m_i^n = \sum_p w_{ip}^n m_p$, where the weights are $w_{ip} = N(\mathbf{x}_p - \mathbf{x}_i)$ and $N(\mathbf{x})$ is an interpolation kernel. An inertia-like tensor \mathbf{D}_p is then computed for each particle as $\mathbf{D}_p = \sum_i w_{ip}^n (\mathbf{x}_i^n - \mathbf{x}_p^n) (\mathbf{x}_i^n - \mathbf{x}_p^n)^T = \frac{\Delta x^2}{6-o} \mathbf{I}$, where the last equality applies for the splines of order $o = 2, 3$ that we use for interpolation kernels. Velocity is transferred to the grid as $\mathbf{v}_i^n = \frac{1}{m_i^n} \sum_p w_{ip}^n m_p (\mathbf{v}_p^n + \mathbf{C}_p^n (\mathbf{x}_i^n - \mathbf{x}_p^n))$, where $\mathbf{C}_p^n = \mathbf{B}_p^n \mathbf{D}_p^{-1}$ is the velocity gradient.
3. **Force integration on the grid.** Elastic forces are computed on the grid as $\mathbf{f}_i = -V_p^0 \mathbf{P} \mathbf{F}_p^{nT} \nabla w_{ip}^n$, $\mathbf{P} = \frac{\partial \psi}{\partial \mathbf{F}}(\mathbf{F}_p^n)$, where V_p^0 is the initial particle volume, ψ is the energy density function for the constitutive model, and \mathbf{P} is the first Piola-Kirchhoff stress. Forces are then integrated to update velocities as in Line 4.
4. **Collisions with objects and domain boundaries.** In Line 5, we process grid collisions against collision objects as in [SSC*13]. Line 6 further modifies the grid velocities to account for domain boundaries using our reflection boundary conditions (Section 5).
5. **Selection of a stable time step size.** Following application of forces and collisions, we have sufficient information available to choose a stable time step size for the current time step. In Lines 7-11, we compute time step restrictions based on particle displacements, change to the deformation gradient, elastic wave speeds, and the single-particle instability, respectively. We describe these in detail in Section 4.
6. **Velocity adjustment.** In Line 13, we adjust the velocity to account for the reduced time step size. We note that this is exact for the explicitly integrated forces, but may introduce some inaccuracy for object and domain boundary collisions, e.g., applying collision forces a bit sooner.
7. **Position update on the grid.** In Line 15, the new grid velocities are integrated to get positions at the grid nodes. In practice, this step is incorporated into the particle position update below.
8. **Transfer data from the grid to particles.** Particle velocities and affine states are updated from the grid as $\mathbf{v}_p^{n+1} = \sum_i w_{ip}^n \tilde{\mathbf{v}}_i^{n+1}$ and $\mathbf{B}_p^{n+1} = \sum_i w_{ip}^n \tilde{\mathbf{v}}_i^{n+1} (\mathbf{x}_i^n - \mathbf{x}_p^n)^T$. The deformation gradients are updated as $\mathbf{F}_p^{n+1} = (\mathbf{I} + \Delta t \nabla \mathbf{v}_p) \mathbf{F}_p^n$, where $\nabla \mathbf{v}_p = \sum_i \tilde{\mathbf{v}}_i^{n+1} (\nabla w_{ip}^n)^T$. Particle positions are then updated as $\mathbf{x}_p^{n+1} = \sum_i w_{ip}^n \tilde{\mathbf{x}}_i^{n+1}$. As a practical optimization, we directly compute the equivalent update $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}$ and avoid the need to compute $\tilde{\mathbf{x}}_i^{n+1}$ in Line 15.
9. **Plasticity update.** Line 17 updates the elastic and plastic components of the deformation gradient as in [KGP*16].

Remark on time step size adjustment. Because our time step restrictions account for forces and collisions (Lines 4-6), we do not know our final time step size until after we have updated grid velocities. At this point, if we determine that a smaller time step $\Delta t'$ was needed, we have a number of potential options, including (a)

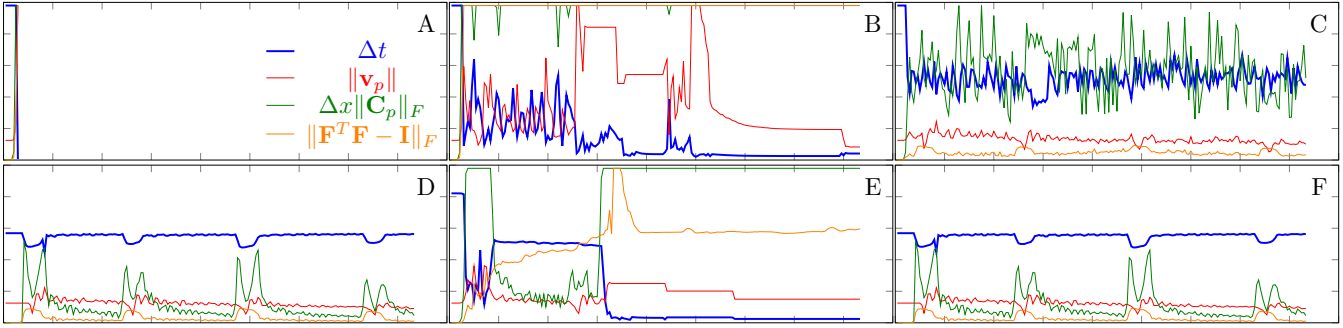


Figure 3: We demonstrate the effect of different combinations of time step restrictions for the colliding spheres example (Fig. 4). Plots show simulation health, as measured through velocity \mathbf{v}_p , velocity gradient \mathbf{C}_p , deformation gradient \mathbf{F} , and minimum time step size Δt within each simulation frame. (A) Using only maximum particle velocity for a time step restriction is unstable; the simulation quickly explodes. (B) Including $\|\mathbf{C}_p\|_F$ in the velocity time step restriction suffices to prevent explosion of the APIC simulation; the large values of $\|\mathbf{C}_p\|_F$ force a very small time step size. Note that the green curve is clamped to the top of the plot for most of the simulation. (C) Additionally controlling the change in \mathbf{F}_p works better and enables a larger time step size. The simulation is still not stable, as particle velocity derivatives vary wildly. (D) Using a proper sound speed CFL ($\nu = 0.9$) produces a stable simulation. Nearly all time steps are sound-speed-limited for this simulation, so this suffices to ensure stability. (E) Using all of our time step restrictions but with a sound speed CFL number of $\nu = 1.3$ makes the simulation unstable. In this case, one of the balls ejects a particle, which subsequently maintains constant, large values in its \mathbf{F}_p and \mathbf{C}_p , resulting in small time steps. (F) Using all of our time step restrictions is stable. The results agree closely with the sound-speed-only simulation, since the time step sizes are mostly the same.

divide and conquer (e.g., divide Δt by 2 and try again), (b) try the time step again with $\Delta t'$, or (c) patch the time step so that it behaves as if the smaller time step size was taken from the beginning. Option (a) would be quite expensive, leading to many more time steps than necessary. Option (b) is at most twice the cost and can be improved significantly by storing and reusing the forces, but we have no guarantee that the time step restrictions would be satisfied, since the new time step would produce different collision forces. Therefore, we pursue option (c). If we treat collision forces as though they were part of \mathbf{f}_i and independent of Δt , then we can simply reapply forces with the new time step. In practice, we do not have \mathbf{f}_i since we applied collisions directly to velocities, but we can compute our effective forces from the initial and final velocities \mathbf{v}_i^n and \mathbf{v}_i^{n+1} , as we do in Line 13.

4. Time Step Size

In this section, we derive five time step restrictions for stable explicit MPM simulation. The criteria for the sound speed and for isolated particles are inherent and thus theoretically required; violating them necessarily leads to unstable simulations under appropriate circumstances. The other three restrictions are heuristic, in the sense that violating them is not intrinsically linked to an instability. There are, however, theoretical justifications for enforcing them, which we note. Due to space considerations, we give abridged derivations and results here and refer the reader to the supplementary document for full details.

4.1. Sound Speed

The most important time step restriction is the classical CFL condition, which mandates that the numerical speed at which infor-

mation travels through a simulation grid must be at least as large as the physical speed [Tor13]. The sound speed plays a fundamental role in compressible flows, where its calculation is a fairly routine exercise [Tor13]. Outside fluid dynamics, we are only aware of results for isotropic linear elasticity at the rest configuration [BH92, FHHJ18] and one-dimensional analysis for Riemann solvers for nonlinear elasticity [BDRT09]. Only the fastest wave speed is needed for properly computing a CFL condition for explicit integrators. In this section, we adapt ideas from the study of compressible flows to give a novel derivation of the wave speeds for fully nonlinear hyperelastic constitutive models in configurations far from the rest state.

We adopt index tensor notation with summation convention. We use comma notation for partial derivatives (e.g., $u_{i,r} = \partial u_i / \partial x_r$), with parentheses indicating the derivative with respect to an entry of the deformation gradient (e.g., $\sigma_{ir,(km)} = \partial \sigma_{ir} / \partial F_{km}$).

Equations of motion and $\mathcal{D} \times \mathcal{D}$ eigenvalue problem. Our equations of motion include conservation of mass and momentum, and an Eulerian form of the MPM update rule for the deformation gradient F , given as

$$\begin{aligned} \rho_t + (\rho u_r)_{,r} &= 0, \\ (\rho u_i)_t + (\rho u_i u_r - \sigma_{ir})_{,r} &= 0, \\ (F_t)_{ij} + F_{i,j,r} u_r - u_{i,r} F_{rj} &= 0, \end{aligned}$$

where ρ is the density, u is the velocity, and σ is the Cauchy stress. We first write our system of equations in the form $U_t + G_r U_r = 0$, with $U = (\rho, \phi, F)^T$, where $\phi = \rho u$. Let $d = 2, 3$ be the number of spatial dimensions, and denote the number of elements in U as $\mathcal{D} = 1 + d + d^2$. The $\mathcal{D} \times \mathcal{D} \times d$ tensor G_r is analogous to a flux Jacobian, and in analogy to sound speed solutions in compress-

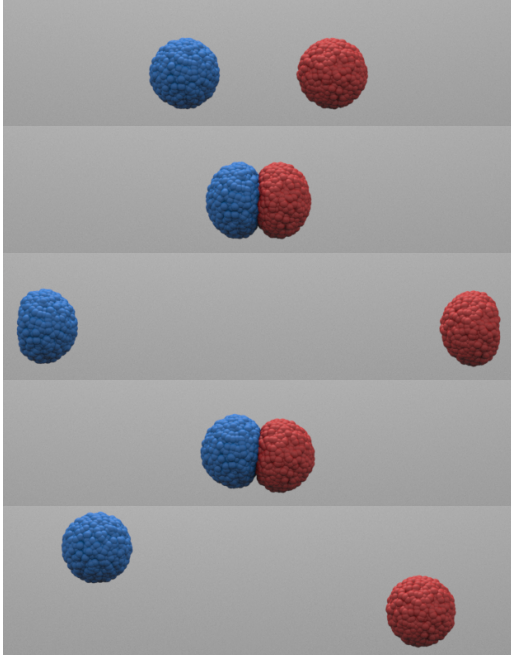


Figure 4: As an illustration of stability, we simulate two spheres alternately colliding with each other and the domain walls.

ible flow [Tor13], we seek the eigenvalues of the $\mathcal{D} \times \mathcal{D}$ matrix $N = G_r n_r$, where n_r is a unit vector. The largest eigenvalue in magnitude of N for any n_r will determine our CFL restriction.

Reduction to a $d \times d$ eigenvalue problem. Using an ansatz based on solution of the one-dimensional case, we find the eigenvalues of N have the form $n_r u_r + c$, where c is the sound speed, and are able to reduce the $\mathcal{D} \times \mathcal{D}$ eigenvalue problem to the smaller $d \times d$ eigenvalue problem $(A_{ik} - \rho c^2 \delta_{ik}) v_k = 0$, with $A_{ik} = J^{-1} P_{in,(km)} F_{rn} F_{sm} n_r n_s$. Recalling that we initially fixed a direction n_r , we can write our full problem as finding κ such that

$$\kappa = \max_{\substack{v_i \\ \|v\|=\|n\|=1}} A_{ik} v_i v_k. \quad (1)$$

Isotropic solutions. We make further progress on solving Eq. (1) by assuming isotropy of the constitutive model. In practice, this covers the significant majority of elasticity models in use in graphics and MPM. Using the SVD of the deformation gradient, $F_{ij} = U_{ik} \Sigma_{kn} V_{jn}$, we can transform $P_{in,(km)} = U_{ir} V_{ns} U_{kt} V_{mu} M_{rstu}$ into the sparse tensor M_{rstu} , whose nonzero entries depend on the derivatives of ψ with respect to the singular values of F [ITF04, SHST12, Sch19]. Expressing v_i and n_j in the basis of the left singular vectors as $p_r = U_{ir} v_i$, $q_o = U_{jo} n_j$, Eq. (1) becomes

$$\kappa = J^{-1} \max_{\substack{p,q \\ \|p\|=\|q\|=1}} M_{rstu} \Sigma_{os} \Sigma_{vu} p_r q_o p_t q_v. \quad (2)$$

Special solutions and sound speed CFL. Eq. (2) has trivial solutions that occur when p_r and q_o are axis vectors (where v_i and n_i

are aligned with the left singular vectors of F). By identifying additional solutions both analytically and through numerical approximation (see supplementary document), we find that these trivial solutions predict the maximum sound speed in the vast majority of cases. In practice, we found the trivial sound speed solutions to be sufficient for our sound speed CFL. Using these solutions, our sound-speed-based CFL condition for MPM can be summarized as

$$\kappa = \frac{1}{J} \max_{a,b} M_{abab} \Sigma_{bb} \Sigma_{bb}, \quad c = \sqrt{\frac{\kappa}{\rho}}, \quad \Delta t = v \frac{\Delta x}{c}, \quad (3)$$

where $v \leq 1$. In terms of the energy density $\hat{\psi}(\sigma_1, \sigma_2, \sigma_3)$,

$$\kappa = \frac{1}{J} \max \left(\max_a \left(\sigma_a^2 \frac{\partial^2 \hat{\psi}}{\partial \sigma_a^2} \right), \max_{a \neq b} \left(\sigma_b^2 \frac{\sigma_a \frac{\partial \hat{\psi}}{\partial \sigma_a} - \sigma_b \frac{\partial \hat{\psi}}{\partial \sigma_b}}{\sigma_a^2 - \sigma_b^2} \right) \right).$$

Note that we use the sound speed c directly, rather than the eigenvalues of N given by $n_r u_r + c$, as we do not need to account for the advective component $n_r u_r$, which is treated in a Lagrangian manner in MPM. For SVD-based constitutive models (such as corotated [MZS*11, SSC*13], fixed corotated [SHST12] and Drucker-Prager with Hencky strain [KGP*16]), Σ and M_{rstu} are readily available, so the CFL computation adds only modest overhead.

4.2. Velocity

Here we describe the velocity-based time step restriction computed by VELOCITY_DT in Algorithm 1, Line 2, which is used to get the initial size for the time step. The velocity transferred from particle p to grid node i is given by $\mathbf{v}_{ip}^n = \mathbf{v}_p^n + \mathbf{B}_p^n \mathbf{D}_p^{-1} (\mathbf{x}_i^n - \mathbf{x}_p^n)$, where $\mathbf{B}_p = \mathbf{0}$ for non-APIC transfers. Let $d = 2, 3$ be the dimension, and let $o = 2, 3$ be the spline order (quadratic or cubic). Using $\mathbf{D}_p^{-1} = \frac{6-o}{\Delta x^2} \mathbf{I}$ [JSS*15], a bound for $\|\mathbf{v}_{ip}^n\|$ over i is given by $\max_i \|\mathbf{v}_{ip}^n\| \leq v$, where $v = \max_p \left(\|\mathbf{v}_p\| + \frac{6\sqrt{d}}{\Delta x} \|\mathbf{B}_p\|_F \right)$. The velocity time step restriction is then given by $\Delta t = \alpha \frac{\Delta x}{v}$, for $0 < \alpha \leq 1$. We choose $\alpha = 1$ for all examples.

4.3. Particle Displacement

We limit the time step so that the displacement of each particle is less than a grid cell. The displacement of a particle over a time step size Δt is given by $\mathbf{x}_p^{n+1} - \mathbf{x}_p^n = \Delta t \sum_i w_{ip}^n \left(\mathbf{v}_i^n + \frac{\Delta t}{m_i} \mathbf{f}_i \right)$. If we scale the time step to be $s \Delta t$, where $0 < s \leq 1$, the particle displacement is $\mathbf{x}_p^{n+1} - \mathbf{x}_p^n = s \Delta t \sum_i w_{ip}^n \mathbf{v}_i^n + s^2 \Delta t \sum_i w_{ip}^n (\tilde{\mathbf{v}}_i^{n+1} - \mathbf{v}_i^n)$. We limit particle displacement by choosing s to enforce $\|\mathbf{x}_p^{n+1} - \mathbf{x}_p^n\|_\infty \leq \alpha \Delta x$. Computing s involves solving a quadratic equation componentwise, which can be avoided in most cases by first testing feasibility of the current estimate s , starting with $s = 1$. POSITION_DT returns $s \Delta t$.

4.4. Deformation Gradient Change

The deformation gradient update over a time step $s \Delta t$ is $\mathbf{F}_p^{n+1} = (\mathbf{I} + \mathbf{A}) \mathbf{F}_p^n$, with

$$\mathbf{A} = s \Delta t \sum_i \mathbf{v}_i^n (\nabla w_{ip}^n)^T + s^2 \Delta t^2 \sum_i (\tilde{\mathbf{v}}_i^{n+1} - \mathbf{v}_i^n) (\nabla w_{ip}^n)^T$$

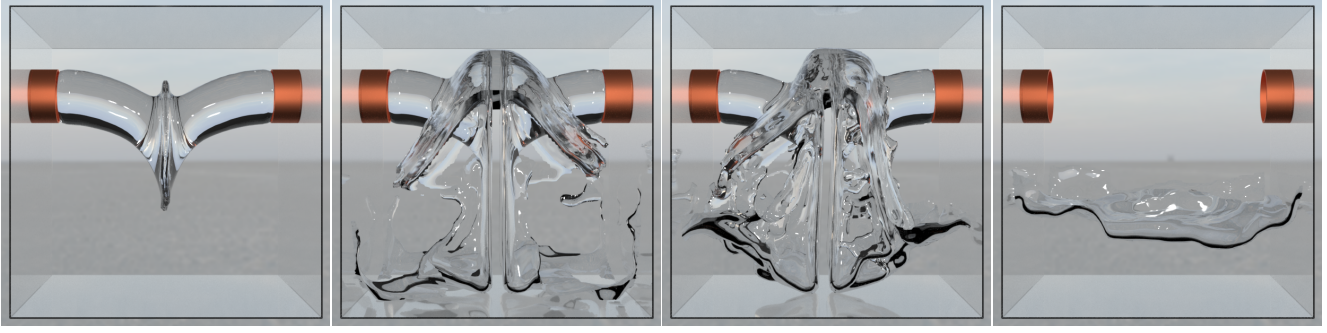


Figure 5: Water pours into a box forming sheets and splashes. The single-particle stability criterion was the most limiting time step restriction in this simulation.

Here, we enforce $\|\mathbf{A}\|_{\max} \leq \delta$, which is a componentwise quadratic as before. In 3D, $0 < \delta < \frac{1}{3}$ implies $\det(\mathbf{I} + \mathbf{A}) > 0$, which guarantees no inversion (i.e., $J_p^n = \det(\mathbf{F}_p^n) > 0$), which is especially important for constitutive models like Neo-Hookean that contain terms like $\ln(J)$ or J^{-1} . We found $\delta = 0.2$ to be a good compromise; values close to $\frac{1}{3}$ should be avoided, and smaller values do not improve stability significantly. In the fluid case, we maintain J_p instead of \mathbf{F}_p^n , update it with $J_p^{n+1} = (\mathbf{I} + \text{tr}(\mathbf{A}))J_p^n$, and enforce $|\text{tr}(\mathbf{A})| \leq \delta$. We note that limiting the change in \mathbf{F}_p during a time step prevents the simulation from taking a large time step size during a collision giving forces a chance to respond to the collision in a stable manner. `DEFORMATION_GRADIENT_DT` returns $s\Delta t$.

4.5. Single-particle Stability

In an MPM simulation, neighboring particles have a stabilizing influence on particle evolution, and thus the ejection and isolation of a particle (for example, due to splashing water) may lead to simulation instability. While this observation is not new [JSS*15, JST17], here we identify for the first time an instability caused by a feedback loop between \mathbf{F} (or J_p for fluids) and forces.

We analyze the single-particle instability for the case of MPM weakly compressible fluid simulation, assuming PIC transfers for simplicity, and show that this instability imposes restrictions on both the choice of basis function and the time step. We note that this is the only stability criterion that we propose that is specific to the underlying spatial discretization.

Feedback between pressure and J . When a particle becomes isolated, the nearby grid data depends only on that particle's data, so that the grid velocity is $\mathbf{v}_i^n = \mathbf{v}_p^n$ and the pressure force is $\mathbf{f}_i = -V_p^0 J_p^n \psi'(J) \nabla w_{ip}^n$ for a single particle p . Integrating forces on the grid, we get $\tilde{\mathbf{v}}_i^{n+1} = \mathbf{v}_i^n + \Delta t \mathbf{f}_i / m_i^n$, which is then used to update J as $J_p^{n+1} = (1 + \Delta t \nabla \cdot \mathbf{v}_p) J_p^n$, where $\nabla \cdot \mathbf{v}_p = \sum_i \tilde{\mathbf{v}}_i^{n+1} \cdot \nabla w_{ip}^n$. Substituting the expressions for $\nabla \cdot \mathbf{v}_p$, $\tilde{\mathbf{v}}_i^{n+1}$, and \mathbf{f}_i into the update rule for J , we get

$$J_p^{n+1} = \left(1 - \Delta t^2 m_p^{-1} V_p^0 J_p^n \psi' \text{tr}(\mathbf{H})\right) J_p^n, \quad (4)$$

where $\mathbf{H} = \sum_i \nabla w_{ip}^n (\nabla w_{ip}^n)^T / w_{ip}^n$. As long as the particle remains isolated, Eq. (4) acts like a fixed-point iteration on J (ignoring mi-

nor changes to \mathbf{H} due to translation), and may diverge if growth is not controlled. If J becomes very large, the particle may inject this stored energy into the simulation when it hits a wall or rejoins the fluid bulk.

Dependence of $\text{tr}(\mathbf{H})$ on interpolation basis. The entries of \mathbf{H} depend on the choice of interpolation basis. We assume $w_{ip}^n = N(\mathbf{x}_i - \mathbf{x}_p)$ is a tensor product basis as was done in [SSC*13], so that in 3D $N(\mathbf{x}_{ijk}) = N(x_i)N(y_j)N(z_k)$. Defining $\hat{N}(\hat{x})$ by $N(x) = \hat{N}(x/\Delta x) = \hat{N}(\hat{x})$, where $\hat{N}(\hat{x})$ is not a function of grid size, we find $\mathbf{H}_{11} = \frac{1}{\Delta x^2} \sum_i \frac{\hat{N}'(\hat{x}_i)^2}{\hat{N}(\hat{x}_i)}$; the other diagonal entries of \mathbf{H} are similar. This imposes a restriction on the choice of interpolation kernel \hat{N} , since we require $\text{tr}(\mathbf{H})$ to be bounded for any location x_i in a grid cell. The linear interpolation kernel fails this stability criterion, but for quadratic and cubic splines, we have the bound $\text{tr}(\mathbf{H}) \leq \frac{Kd}{\Delta x^2}$ where $K = 6$ for quadratic splines and $K = 3.14$ for cubic splines, and $d = 2, 3$ is the dimension.

Bounding the growth of J . With the bound on $\text{tr}(\mathbf{H})$ substituted into Eq. (4), the ratio of change in J is bounded by

$$J_p^{n+1} = (1 - r J_p^n \psi') J_p^n, \quad r = \frac{\Delta t^2 V_p^0 K d}{\Delta x^2 m_p}.$$

We can express Δt in terms of r as

$$\Delta t = \Delta x \sqrt{\frac{m_p r}{V_p^0 K d}} = \Delta x \sqrt{\frac{\rho_p^0 r}{K d}}. \quad (5)$$

Therefore, a bound on r will give a corresponding bound on Δt .

We propose two ways to stabilize a time step with respect to the single-particle instability. The simpler and more restrictive way is to prevent J_p from overshooting 1. Note that $r = 0$ gives $J_p^{n+1} = J_p^n$, whereas to achieve $J_p^{n+1} = 1$, we need $r = \frac{J_p^n - 1}{(J_p^n)^2 \psi'}$. Substituting this expression into Eq. (5) gives us the time step

$$\Delta t = \frac{\Delta x}{J_p^n} \sqrt{\frac{\rho_p^0 (J_p^n - 1)}{K \psi' d}}. \quad (6)$$

Note that there are no real problems when $J_p^n \approx 1$ since by L'Hôpital's rule, $\lim_{J_p^n \rightarrow 1} \Delta t = \Delta x \sqrt{\frac{\rho_p^0}{K \psi' d}}$.

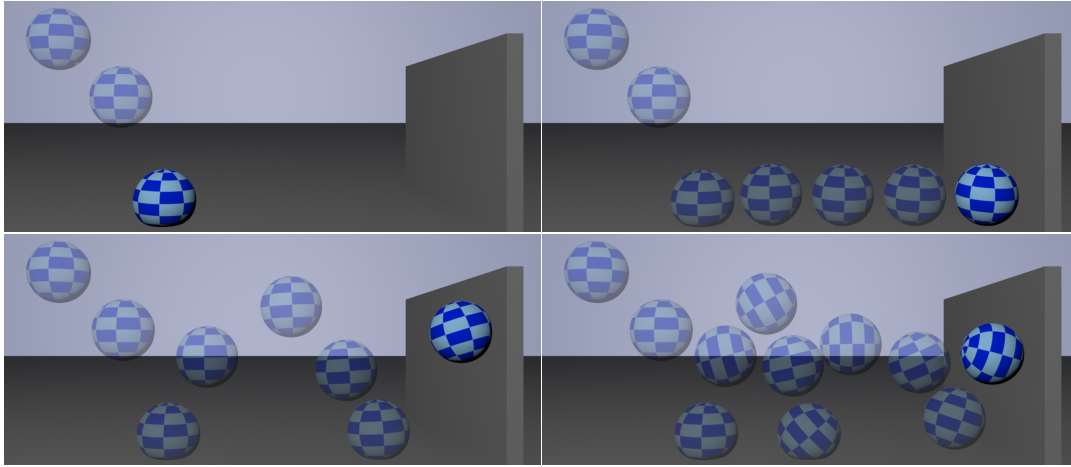


Figure 6: A ball bounces off a ground with different boundary condition types. (Top, left) With sticking collisions, the ball sticks where it collides. (Top, right) With slip collisions, it remains stuck to the ground but slides freely along it. (Bottom, left) Enabling separation, the ball bounces off the ground. (Bottom, right) With the addition of friction, it rotates as well.

Less restrictive bound. The time step restriction in Eq. (6) can be relaxed by up to a factor of $\sqrt{2}$ near $J_p \approx 1$ if we allow J_p to overshoot 1 and instead prevent it from diverging. This requires global information about our constitutive model. For our purposes, we assume that the force's growth is bounded by

$$0 \geq \psi'(J) \geq \lambda(J-1)J^{-2}, \quad 0 < J \leq 1, \quad (7)$$

$$0 \leq \psi'(J) \leq \lambda(J-1), \quad J \geq 1, \quad (8)$$

for some $\lambda > 0$, which must be computed given knowledge of the constitutive model. This growth is fast enough to accommodate constitutive models $\psi(J)$ that contain common terms like $(J-1)^2$, $\ln J$, $\ln^2 J$, or J^{-1} . We select r in order to ensure that $J_2 = (1-rJ\psi')J$ is between J and J^{-1} . Across many time steps, we will have $|\ln(J)| \geq |\ln(J_2)| \geq |\ln(J_3)| \geq |\ln(J_4)| \geq \dots \geq 0 = |\ln(1)|$, which prevents divergence. We find that the bound on J_2 can be satisfied if $r \leq \frac{2}{\lambda(2-J)^2}$ for $0 < J \leq 1$, and $r \leq \frac{J+1}{\lambda J^3}$ for $J \geq 1$. Substituting these expressions into Eq. (5), we can summarize this relaxed time step restriction as

$$\Delta t = \frac{\Delta x}{2-J} \sqrt{\frac{2\rho_p^0}{K\lambda d}}, \quad 0 < J \leq 1,$$

$$\Delta t = \frac{\Delta x}{J} \sqrt{\frac{\rho_p^0(J+1)}{JK\lambda d}}, \quad J \geq 1.$$

Alternative time step restrictions may be derived in the same way for different bounds or directly from individual constitutive models for even more favorable time step sizes. We use this time step restriction for our fluid simulations since both of our fluid constitutive models satisfy the growth bounds (7)-(8).

Isolated particles are very common in fluids as a result of splashing but less frequently encountered with solids; we leave that case for future work. Our treatment of the single-particle instability is dependent on the transfer interpolation functions, the temporal time integration scheme, and the specific properties of the constitutive

model. All of the other time step restrictions are independent of the interpolation functions.

5. Reflection Boundary Conditions for MPM

We propose a novel method for applying collisions or boundary conditions to an MPM simulation at domain walls based on the reflection boundary conditions of [DSS19]. The motivation for [DSS19] is as follows: when touching a mirror, the mirror itself applies the same forces to your hand that a physical reflection of your hand would. Thus, boundary conditions can be enforced by running the simulation as though a reflected copy of each particle existed on the other side of the domain walls.

Reflection boundary conditions were used in [DSS19] to enforce fluid boundary conditions at domain walls, including no-slip, free surface, and free slip conditions. Our method builds on [DSS19] with three improvements: (1) whereas [DSS19] modified mass and momentum, we modify velocity directly; this conserves mass and (where appropriate) momentum. (2) [DSS19] applied mass and momentum modification at various stages in their algorithm, whereas we require only a single velocity correction step at the end of the grid evolution, and (3) [DSS19] did not support friction or separation; we extend our formulation to these cases. Unlike [SSC*13], where particles can seep into objects by a grid cell or so, our method stops particles at domain walls. We give an overview of the motivation and method here and refer the reader to the supplementary document for a full derivation and proofs of important properties.

Reflection notation and boundary types. We use $r(\cdot)$ to refer to reflected quantities: $r(i)$ is the index of the grid node (or cell) obtained by reflecting i across the domain wall, and $r(\mathbf{v}) = \mathbf{A}\mathbf{v} + \mathbf{2b}$ is a reflected velocity with \mathbf{A} a diagonal matrix and \mathbf{b} a vector. The choice of \mathbf{A} and \mathbf{b} determine the type of boundary condition being enforced: $\mathbf{A} = \mathbf{I}, \mathbf{b} = \mathbf{0}$ is a free surface boundary condition (in the computational fluid dynamics sense), $\mathbf{A} = -\mathbf{I}$ enforces the no-slip

boundary condition $\mathbf{v} = \mathbf{b}$, and $\mathbf{A} = \mathbf{I} - 2\mathbf{nn}^T$, $\mathbf{b} = v_n \mathbf{n}$ enforces a slip boundary condition ($\mathbf{n} \cdot \mathbf{v} = v_n$). Note that $\mathbf{n} = \pm \mathbf{e}_a$, where \mathbf{e}_a are the axis vectors, since our domain walls are axis-aligned.

Modified mass, momentum, and forces. For motivation and completeness, we briefly review the reflection boundary conditions presented in [DSS19]. In that work, the boundary conditions are enforced by solving a *modified* grid system, constructed by transferring data from both the regular particles p as well as a virtual, reflected copy $r(p)$ (which is never actually constructed). Below, a hat indicates a modified quantity computed from both real and reflected particles, while no hat indicates a quantity computed from the real particles only. The modified grid mass is given by

$$\hat{m}_i^n = \sum_p w_{ip}^n m_p + \sum_p w_{ir(p)}^n m_{r(p)} = m_i^n + m_{r(i)}^n. \quad (9)$$

That is, the mass from the reflected grid node is simply added to the mass of the grid node. Similarly, the modified grid momentum can be shown to be

$$\hat{m}_i^n \hat{\mathbf{v}}_i^n = m_i^n \mathbf{v}_i^n + m_{r(i)}^n r(\mathbf{v}_{r(i)}^n). \quad (10)$$

Finally, in [DSS19], the addition of the reflected particles also results in modified forces $\hat{\mathbf{f}}_i = \mathbf{f}_i + \mathbf{A}\mathbf{f}_{r(i)}$.

While enforcing the domain boundary conditions, the above modified system does not conserve mass or momentum (along directions where no velocity is enforced and momentum should therefore be conserved, e.g., all directions in the free surface case). Our proposed reflection boundary condition alleviates these issues and facilitates our treatment of friction and separation.

Modified velocities. We observe that using the original grid masses, and modifying only the grid velocities in a manner consistent with Eqs. (9)-(10), we obtain both conservation of mass (trivially), and conservation of momentum in the free surface case ($\mathbf{A} = \mathbf{I}, \mathbf{b} = \mathbf{0}$), since

$$m_i^n \hat{\mathbf{v}}_i^n + m_{r(i)}^n \hat{\mathbf{v}}_{r(i)}^n = m_i^n \mathbf{v}_i^n + m_{r(i)}^n \mathbf{v}_i^n = \hat{m}_i^n \hat{\mathbf{v}}_i^n = m_i^n \mathbf{v}_i^n + m_{r(i)}^n \mathbf{v}_{r(i)}^n.$$

Similarly, for slip boundary conditions, this choice conserves momentum in directions where a specified boundary velocity is not being enforced.

Therefore, our reflection boundary conditions take the form of a velocity correction applied at the end of the grid update (Algo-

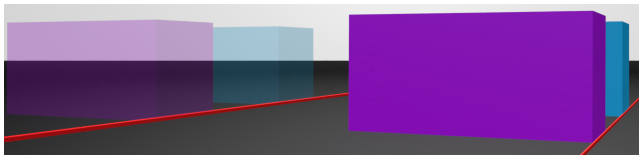


Figure 7: Our reflection boundary conditions produce accurate frictional forces. A purple block slides along the ground with friction and comes to a stop, in agreement with the analytic solution (blue block, red lines).



Figure 8: Sand pours into a pile. Our reflection boundary conditions handle the frictional collisions with the ground.

rithm 1, Line 6). Using Eqs. (9)-(10), this is given by

$$\hat{\mathbf{v}}_i^{n+1} = \alpha_i^n \mathbf{v}_i^{n+1} + \alpha_{r(i)}^n r(\mathbf{v}_{r(i)}^{n+1}), \quad \hat{\mathbf{v}}_{r(i)}^{n+1} = r(\hat{\mathbf{v}}_i^{n+1}), \quad (11)$$

$$\alpha_i^n = \frac{m_i^n}{m_i^n + m_{r(i)}^n}, \quad \alpha_{r(i)}^n = \frac{m_{r(i)}^n}{m_i^n + m_{r(i)}^n}. \quad (12)$$

This correction is only applied near the domain walls where $m_i^n \neq 0$ and $m_{r(i)}^n \neq 0$, limiting it to a thin band around the domain boundaries whose thickness depends on the MPM transfer stencil width.

Frictional contact and separation. We extend the reflection boundary condition in the slip case to handle Coulomb friction and separation. Our frictional contact treatment uses the modified velocities in Eqs. (11)-(12). As a result of enforcing the slip boundary condition by reflecting the velocity, the total tangential momentum was conserved but the normal component of total momentum changed. The difference in the normal component is found to be

$$\begin{aligned} \Delta p_n &= (m_i^n \hat{\mathbf{v}}_i^{n+1} + m_{r(i)}^n \hat{\mathbf{v}}_{r(i)}^{n+1}) \cdot \mathbf{n} - (m_i^n \mathbf{v}_i^{n+1} + m_{r(i)}^n \mathbf{v}_{r(i)}^{n+1}) \cdot \mathbf{n} \\ &= 2m_i^n \alpha_{r(i)}^n (2\mathbf{b} - \mathbf{v}_i^{n+1} - \mathbf{v}_{r(i)}^{n+1}) \cdot \mathbf{n}. \end{aligned}$$

Therefore, we define the separation decision variable $c = (2\mathbf{b} - \mathbf{v}_i^{n+1} - \mathbf{v}_{r(i)}^{n+1}) \cdot \mathbf{n}$. The sign of c indicates whether the momentum exchange in the normal direction due to the reflection boundary condition was a push ($c \geq 0$, friction) or pull ($c < 0$, separation). If $c \geq 0$, we apply friction to the modified velocity by decomposing it into normal and tangential parts, $\hat{\mathbf{v}}_{in}^{n+1} = \mathbf{nn}^T \hat{\mathbf{v}}_i^{n+1}$, $\hat{\mathbf{v}}_{it}^{n+1} = (\mathbf{I} - \mathbf{nn}^T) \hat{\mathbf{v}}_i^{n+1}$, and then scaling the tangential part accord-

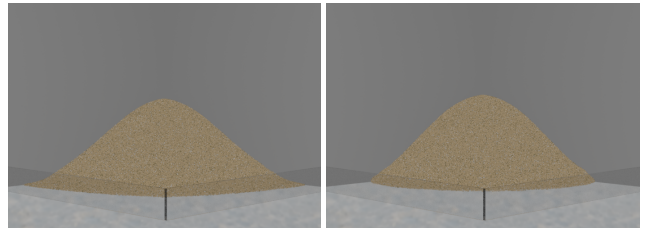


Figure 9: Sand falls into a pile, with low friction (left) and high friction (right) simulated with our reflection boundary conditions.

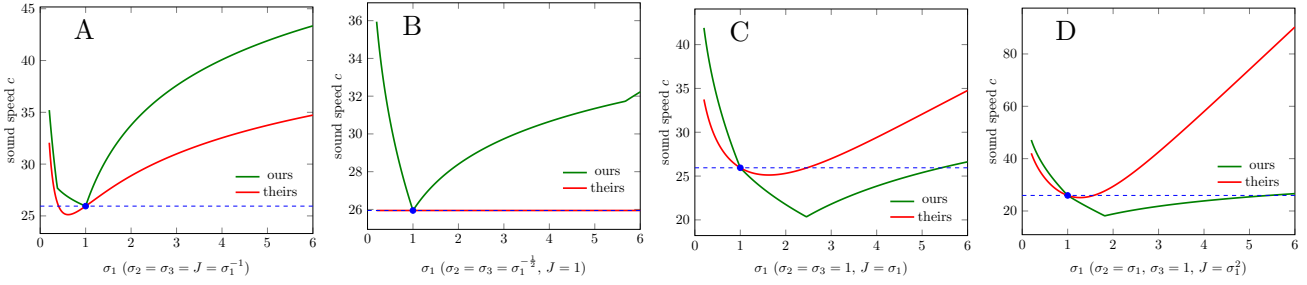


Figure 10: The sound speed estimate used in [FHHJ18] (red) differs significantly from our exact sound speed calculation (green). The deformation gradient is described by its singular values $\sigma_1, \sigma_2, \sigma_3$. Each plot shows the sound speed c along a different path in configuration space, parameterized as a function of σ_1 . (A-D) The estimate of [FHHJ18] is exact only at the rest configuration ($\sigma_1 = \sigma_2 = \sigma_3 = 1$, marked with \bullet). (B) The sound speed is clearly not a function of J alone. The estimate of [FHHJ18] depends only on $J = \sigma_1\sigma_2\sigma_3$, erroneously predicting the rest configuration sound speed whenever $J = 1$, even very far from rest. (C,D) Their model predicts sound speeds both above and below the actual sound speed, and the difference can be quite large.

ing to the Coulomb friction law,

$$\hat{\mathbf{v}}_{i\mu}^{n+1} = \hat{\mathbf{v}}_{i\mu}^{n+1} + \beta \hat{\mathbf{v}}_{it}^{n+1}, \quad \beta = \max\left(1 - \mu \frac{\Delta p_n}{\hat{m}_i^n \|\hat{\mathbf{v}}_{it}^{n+1}\|}, 0\right).$$

The final velocities are $\hat{\mathbf{v}}_{i\mu}^{n+1}$ and $\hat{\mathbf{v}}_{r(i)\mu}^{n+1} = r(\hat{\mathbf{v}}_{i\mu}^{n+1})$.

If $c < 0$, then we must compute separating velocities $\mathbf{v}_{i,sep}^{n+1}$, $\mathbf{v}_{r(i),sep}^{n+1}$ instead. To do this, we combine the original normal velocities and the modified tangential velocities

$$\mathbf{v}_{i,sep}^{n+1} = \mathbf{nn}^T \mathbf{v}_i^{n+1} + \hat{\mathbf{v}}_{it}^{n+1}, \quad \mathbf{v}_{r(i),sep}^{n+1} = \mathbf{nn}^T \mathbf{v}_{r(i)}^{n+1} + \hat{\mathbf{v}}_{it}^{n+1}.$$

This choice conserves momentum, is continuous with the frictional solution at the transition ($c = 0$), and allows free separation in the normal direction by not modifying those velocity components. Detailed derivations for these formulas and proofs of their properties are provided in the accompanying technical document.

6. Results and Discussion

Statistics for our simulations are shown in Table 1. The 3D results were rendered using SideFX/Houdini.

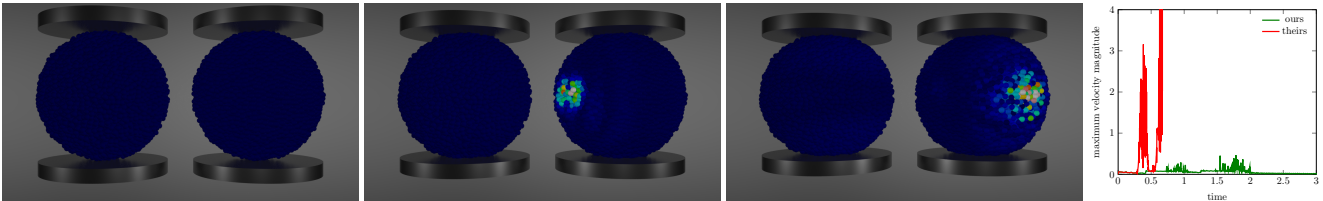


Figure 11: Although the approximate sound speed given in [FHHJ18] is close to the actual sound speed near the rest configuration, it frequently underestimates the actual sound speed, which can lead to instabilities. In this simulation, we compress a sphere between two moving plates using maximum velocity and sound speed to limit time step sizes. Running with their sound speed estimate (right sphere in each image) and a high CFL number ($\nu = 0.99$), we can observe that their estimate is too small, resulting in instability. The simulation is stable when run with the our sound speed and the same CFL number (left sphere in each image).

step sizes) the sound speed. Away from rest, there are many different physical wave speeds corresponding to different directions and wave types, and kinks in the sound speed curves are observed where the maximum wave speed switches between different physical waves. Fig. 10 shows that the physical sound speed is not smooth near the rest configuration (A,B), and changes in the dominant wave speed are not caused solely by changes in the dominant singular value (C). Finally, the estimate of [FHHJ18] implicitly assumes that pressure is purely a function of $J = \det(\mathbf{F})$. This does not apply to popular hyperelasticity models such as Neo-Hookean, whose pressure does not depend only on J , or the corotated model, which has no J dependence at all.

Single-particle instability. In Fig. 2, we show a simple 2D fluid dam break. This simulation becomes unstable when the CFL restriction is not honored or when a particle becomes isolated and the the single-particle stability criterion is not met. The heuristic stability criteria on velocity, particle displacement, and deformation gradient change alone are insufficient for stability. In Fig. 5, we pour water into a box. The isolated particle stability criterion keeps the simulation stable even when particles become isolated as the water splashes.

Combinations of different time step restrictions. Figure 4 shows a simple simulation with two colliding spheres. In Fig. 3, we show the consequences of running the simulation with different combinations of time step restrictions.

Reflection boundary conditions. To demonstrate the effectiveness of the proposed reflection boundary conditions, we run a number of collision tests. Figure 6 shows a ball bouncing off a ground with different types of reflection boundary conditions, demonstrating a variety of behaviors that can be achieved. Figure 7 shows that our friction reflection boundary condition produces accurate friction forces that match the analytic solution. Figure 9 shows a column of sand falling into a pile with different friction coefficients. In Fig. 8, we pour sand from a spout into a pile. Friction with the

Table 1: Simulation statistics. ψ : fixed corotated (FC), Neo-Hookean (NH), Drucker-Prager (DP), $\frac{\lambda}{2}(J-1)^2$ (FL), $\frac{\lambda}{2}\ln^2 J$ (FG, video only). The columns $\Delta t\%$ and $bc\%$ show the percentage of simulation time spent computing time step restrictions and reflection boundary conditions, respectively.

sim	sec/frame	ψ	$\Delta t\%$	$bc\%$	res	particles
1	< 1.3	FC	0-13	2-3	32^3	24k
2	< 1	FL,FG	25-26	< 1	64^2	2k
4,3	< 1	NH	0-10	2-4	64^3	4k
6	1.2 – 1.5	FC	6-7	6-7	32^3	2k
9	300,316	DP	16,17	< 1	128^3	914k
7	28	FC	11	< 1	256×64^2	545k
5	37	FL	17	1	64^3	608k
8	57	DP	18	2	$512^2 \times 192$	131k
12	99	DP	18	0	256×128^2	155k
13	360	FC	9	< 1	240×96^2	1480k

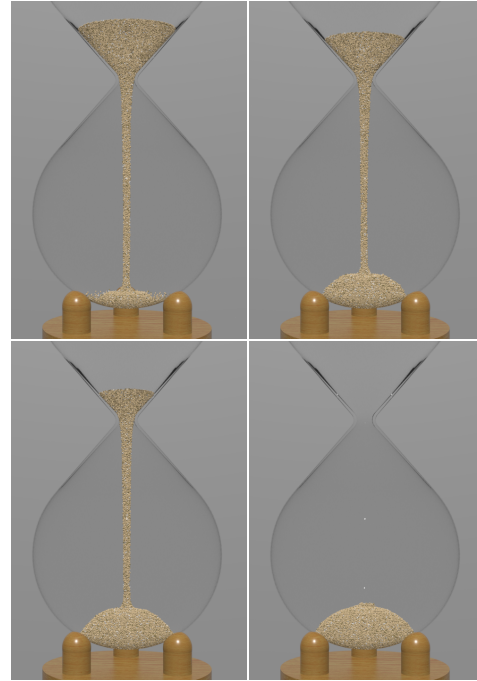


Figure 12: Sand flows through an hourglass showing compatibility of our algorithm and time step restrictions with complex collisions.

ground keeps the sand pile stable. Our robust treatment of particle sources for APIC is detailed in the supplementary document. Our algorithm and time step restrictions also work well with other collisions treatments, such as for the hourglass in Fig. 12. In Figure 13 we drop 36 objects into a box, demonstrating robust simulation even with many collisions.

7. Conclusions and Limitations

The time step restrictions proposed have a number of limitations. In this paper, we assume that forces are due to a hyperelastic isotropic constitutive model, possibly with plasticity. Our algorithm for computing sound speed omits pathological cases, since we found that they occur far too rarely in practice to justify the cost of computing them. Our CFL assumes no damping forces such as viscosity or Rayleigh damping, which would alter the speed of information travel and change the time step restriction. Extending the derivation to include such models would be useful. Although gravity does not cause stability problems, other non-constitutive forces such as penalty springs may, and suitable adjustments to the time step restrictions would be required to keep them stable. Damping models that cannot produce instabilities on their own (scaling down velocity, averaging velocities of nearby grid nodes or particles, blending APIC with RPIC, etc.) may allow larger time steps to be taken than we predict.

We have not analyzed the consequences of using FLIP or FLIP/PIC blends. All of our tests were performed on APIC, though we verified that the single-particle instability also occurs readily under PIC transfers. We have no particular reason to believe

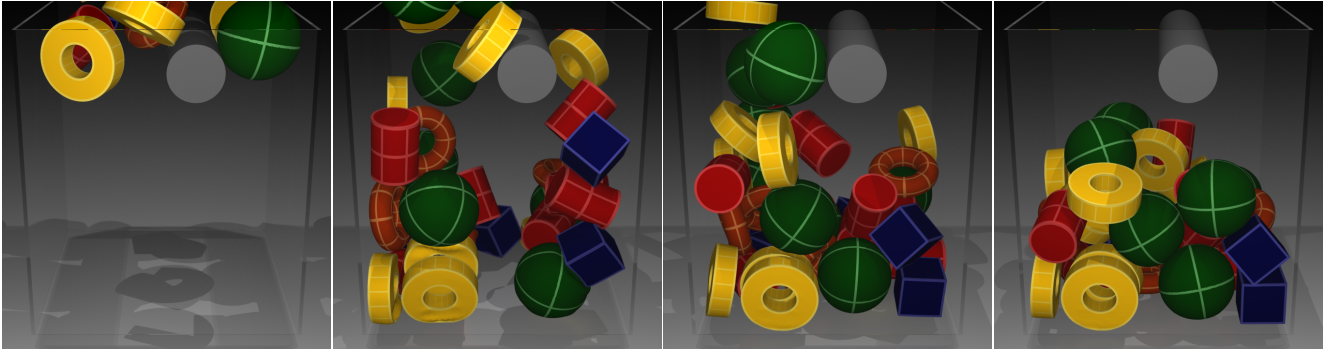


Figure 13: Many stiff, deformable objects fall into box, undergoing complex collisions with each other, the stationary bar, and the box.

that the stability restrictions for sound speed, velocity, particle displacement, or deformation gradient should be different for FLIP, PIC, FLIP/PIC blends, APIC/RPIC blends, MLS, PolyPIC, other choices of basis functions, or other transfer strategies. The single-particle stability analysis for PIC works well for APIC and we expect that it would work well for other PIC-like transfer schemes that interpolate their particle state from the grid, though we note a rather strong dependence on the basis function (quadratic vs. cubic spline in our analysis), suggesting that a new constant should be calculated if a different interpolation basis is used. We suspect that the single-particle stability analysis for FLIP may look quite different due to its tendency to accumulate energy in transfer null modes even for implicit methods.

We derived the single-particle instability case for PIC transfers; we leave a full APIC treatment for future work. We have not formulated an efficient time step restriction for the single-particle instability for non-pressure constitutive models. We have constructed single-particle simulations with general constitutive models that display the instability, confirming its existence. However, we have never observed the instability in a multi-particle simulation, even when particles become isolated. This is quite unlike the fluid case, where the instability is seen frequently. There are many simple heuristic solutions (for example, linearizing the constitutive model, bisection search on the energy density, using the pressure version on the dilational stress component, etc.), but they are either too expensive or do not provide the level of guarantee that we would like. As such, we prefer to propose no solution at present and leave this for future work.

Our time step restrictions do not provide a guarantee of stability, and we have examples of simulations that will fail with these criteria (such as the single-particle solids simulation). In practice, however, it is actually quite difficult for a simulation to explode with our time step restrictions enabled. Very few of the simulations we deliberately made unstable by violating time step restrictions actually exploded; the F_p^n criterion slows the rate of divergence until another restriction (such as C_p^n) drives down the time step size sufficiently to stabilize the simulation. Nevertheless, such simulations are unusable. It is interesting to note that the availability of the C_p^n time step restriction actually makes APIC more stable than the equivalent PIC simulation in some cases. In future work, we would

like to further study the stability of our scheme using techniques such as Von Neumann stability analysis.

Determining the stable time step adds nontrivial computational overhead in our implementation (up to 18% for large simulations), where we have implemented them as separate memory passes for convenience. Of these, the deformation gradient and position time step restrictions account for the significant majority of the cost, since the pass involves grid-to-particle transfers. This overhead is offset in three ways. (1) For larger simulations, the average time step tends to be about 1.5 times the minimum time step. This more than offsets the additional computational cost for most (but not all) of the large simulations. (2) In practice, the time step size that will be selected by trial and error is not equal to the actual minimum required (we tend to reduce the time step by factors of two until we get acceptable results). (3) Finding the stable time step size by trial and error requires the simulation to be run a few times, which is usually not reported for in timing comparisons. For example, for Figure 1, we ran the simulations in a binary search procedure to find the best time step size, which required running the simulation about ten times.

In this paper, we have not considered implicit methods. As general guidance, an implicit method is a better choice if accuracy is not very important, forces are stiff, and plasticity is not being used. More accuracy would require the implicit time steps to be small anyway. Non-stiff forces would allow the explicit solver to take large time step sizes. Handling plasticity accurately in standard implicit methods leads to asymmetrical systems that are expensive to solve. The criteria in this paper make explicit methods more competitive with implicit ones in two ways: they alleviate the need to select a time step size by trial and error (something implicit methods generally do not require), and they allow larger time step sizes since every time step can be run near the stability limit.

References

- [BDRT09] BARTON P. T., DRIKAKIS D., ROMENSKI E., TITAREV V. A.: Exact and approximate solutions of riemann problems in non-linear elasticity. *Journal of Computational Physics* 228, 18 (2009), 7046–7068. 4
- [BH92] BINA C. R., HELFFRICH G. R.: Calculation of elastic properties from thermodynamic equation of state principles. *Annual Review of Earth and Planetary Sciences* 20, 1 (1992), 527–552. 4

- [BK04] BARDENHAGEN S., KOBER E.: The generalized interpolation material point method. *Comp Mod in Eng and Sci* 5, 6 (2004), 477–496. 2
- [BL98] BRACKBILL J., LAPENTA G.: Particle-in-cell magnetohydrodynamics. In *16th Int Conf on the Numer Sim of Plasmas* (1998). 2
- [Bra88] BRACKBILL J.: The ringing instability in particle-in-cell calculations of low-speed flow. *J Comp Phys* 75, 2 (1988), 469–492. 2
- [Bra16] BRANNON R.: The kinematic anomaly in mpm. <https://csmbrannon.net/2016/01/04/the-kinematic-anomaly-in-mpm/>, 2016. Accessed: 2020-01-18. 1
- [BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1998), SIGGRAPH '98, ACM, pp. 43–54. 1
- [DBD16] DAVIET G., BERTAILS-DESCOUBES F.: A semi-implicit material point method for the continuum simulation of granular materials. *ACM Trans Graph* 35, 4 (jul 2016). 2, 3
- [DS19] DING O., SCHROEDER C.: Penalty force for coupling materials with coulomb friction. *IEEE*. 3
- [DSS19] DING O., SHINAR T., SCHROEDER C.: Affine particle in cell method for mac grids and fluid simulation. *Journal of Computational Physics* (2019). 2, 3, 7, 8
- [FGG*17] FU C., GUO Q., GAST T., JIANG C., TERAN J.: A polynomial particle-in-cell method. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 222. 2
- [FHHJ18] FANG Y., HU Y., HU S.-M., JIANG C.: A temporally adaptive material point method with regional time stepping. In *Computer graphics forum* (2018), vol. 37, Wiley Online Library, pp. 195–204. 2, 4, 9, 10
- [GHF*18] GUO Q., HAN X., FU C., GAST T., TAMSTORF R., TERAN J.: A material point method for thin shells with frictional contact. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 147. 3
- [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 Transactions on Graphics (TOG)* 37, 4 (2018), 149. 3
- [Gri14] GRITTON C.: *Ringing Instabilities in Particle Methods*. PhD thesis, The University of Utah, 2014. 2
- [GSS*15] GAST T., SCHROEDER C., STOMAKHIN A., JIANG C., TERAN J.: Optimization integrator for large time steps. *IEEE transactions on visualization and computer graphics* 21, 10 (2015), 1103–1115. 2
- [GW01] GUILKEY J., WEISS J.: An implicit time integration strategy for use with the material point method. In *Proceedings from the First MIT Conference on Computational Fluid and Solid Mechanics* (2001). 2
- [GW03] GUILKEY J. E., WEISS J. A.: Implicit time integration for the material point method: Quantitative and algorithmic comparisons with the finite element method. *Int J Numer Meth Eng* 57, 9 (2003), 1323–1338. 2
- [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 Transactions on Graphics (TOG)* 37, 4 (2018), 150. 3
- [HGG*19] HAN X., GAST T., GUO Q., WANG S., JIANG C., TERAN J.: A hybrid material point method for frictional contact with diverse materials. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 2, 2 (2019), 1–24. 3
- [IAGT10] IHMSEN M., AKINCI N., GISSLER M., TESCHNER M.: Boundary handling and adaptive time-stepping for pcisph. In *Workshop on virtual reality interaction and physical simulation VRIPHYS* (2010), Bender J., Erleben K., Teschner M., (Eds.), vol. 6. 2
- [ITF04] IRVING G., TERAN J., FEDKIW R.: Invertible finite elements for robust simulation of large deformation. In *Proc. Symp. Comp. Anim.* (2004), pp. 131–140. 5
- [JGT17] JIANG C., GAST T., TERAN J.: Anisotropic elastoplasticity for cloth, knit and hair frictional contact. *ACM Transactions on Graphics (SIGGRAPH 2017)* 36, 4 (2017), 152. 3
- [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. 2, 3, 5, 6
- [JST17] JIANG C., SCHROEDER C., TERAN J.: An angular momentum conserving affine-particle-in-cell method. *J Comp Phys* 338 (2017), 137–164. 2, 6
- [KGP*16] KLÁR G., GAST T., PRADHANA A., FU C., SCHROEDER C., JIANG C., TERAN T.: Drucker-prager elastoplasticity for sand animation. *ACM Transactions on Graphics (SIGGRAPH 2016)*. (2016). 2, 3, 5
- [LS06] LOVE E., SULSKY D.: An unconditionally stable, energy-momentum consistent implementation of the the material point method. *Comp Meth App Mech Eng* 195 (2006), 3903–3925. 2
- [MZS*11] MCADAMS A., ZHU Y., SELLE A., EMPEY M., TAMSTORF R., TERAN J., SIFAKIS E.: Efficient elasticity for character skinning with contact and collisions. In *ACM Transactions on Graphics (TOG)* (2011), vol. 30, ACM, p. 37. 5
- [Oku72] OKUDA H.: Nonphysical noises and instabilities in plasma simulation due to a spatial grid. *J Comp Phys* 10, 3 (1972), 475–486. 2
- [RGJ*15] RAM D., GAST T., 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
- [Sch19] SCHROEDER C.: Practical course on computing derivatives in code. In *ACM SIGGRAPH 2019 Courses* (New York, NY, USA, 2019), SIGGRAPH '19, Association for Computing Machinery. URL: <https://doi.org/10.1145/3305366.3328073>, doi: 10.1145/3305366.3328073. 5
- [SCS94] SULSKY D., CHEN Z., SCHREYER H.: A particle method for history-dependent materials. *Comp Meth in App Mech Eng* 118, 1 (1994), 179–196. 2
- [SHST12] STOMAKHIN A., HOWES R., SCHROEDER C., TERAN J.: Energetically consistent invertible elasticity. In *Proc. Symp. Comp. Anim.* (2012), pp. 25–32. 5
- [SK04] SULSKY D., KAUL A.: Implicit dynamics in the material-point method. *Comp Meth in App Mech Eng* 193, 12 (2004), 1137–1170. 2
- [SKB08] STEFFEN M., KIRBY R., BERZINS M.: Analysis and reduction of quadrature errors in the material point method (MPM). *Int J Numer Meth Eng* 76, 6 (2008), 922–948. 2
- [SSC*13] STOMAKHIN A., SCHROEDER C., CHAI L., TERAN J., SELLE A.: A material point method for snow simulation. In *ACM Transactions on Graphics (SIGGRAPH 2013)* (2013), pp. 102:1–10. 1, 2, 3, 5, 6, 7
- [SSJ*14] STOMAKHIN A., SCHROEDER C., JIANG C., CHAI L., TERAN J., SELLE A.: Augmented mpm for phase-change and varied materials. *ACM Transactions on Graphics (SIGGRAPH 2014)* (2014), 1–11. 2
- [SZS95] SULSKY D., ZHOU S., SCHREYER H.: Application of a particle-in-cell method to solid mechanics. *Comp Phys Comm* 87, 1 (1995), 236–252. 2
- [TGK*17] TAMPUBOLON A., GAST T., KLÁR G., FU C., TERAN J., JIANG C., MUSETH K.: Multi-species simulation of porous sand and water mixtures. *ACM Transactions on Graphics (SIGGRAPH 2017)* 36, 4 (2017), 105. 3
- [Tor13] TORO E. F.: *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*. Springer Science & Business Media, 2013. 4, 5

- [WG08] WALLSTEDT P., GUILKEY J.: An evaluation of explicit time integration schemes for use with the generalized interpolation material point method. *J Comp Phys* 227, 22 (2008), 9628–9642. [2](#)
- [WLF*20] WANG X., LI M., FANG Y., ZHANG X., GAO M., TANG M., KAUFMAN D. M., JIANG C.: Hierarchical optimization time integration for cfl-rate mpm stepping. *ACM Transactions on Graphics (TOG)* 39, 3 (2020), 1–16. [2](#)
- [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](#)
- [YSC*18] YUE Y., SMITH B., CHEN P., CHANTHARAYUKHONTHORN M., KAMRIN K., GRINSPUN E.: Hybrid grains: Adaptive coupling of discrete and continuum simulations of granular media. In *SIGGRAPH Asia 2018 Technical Papers* (2018), ACM, p. 283. [2](#)