Revisiting Integration in the Material Point Method: A Scheme for Easier Separation and Less Dissipation

YUN (RAYMOND) FEI, Tencent Game AI Research Center, USA
QI GUO, Tencent Game AI Research Center, USA
RUNDONG WU, Tencent Game AI Research Center, USA
LI HUANG, Tencent Game AI Research Center, P. R. China
MING GAO, Tencent Game AI Research Center, USA

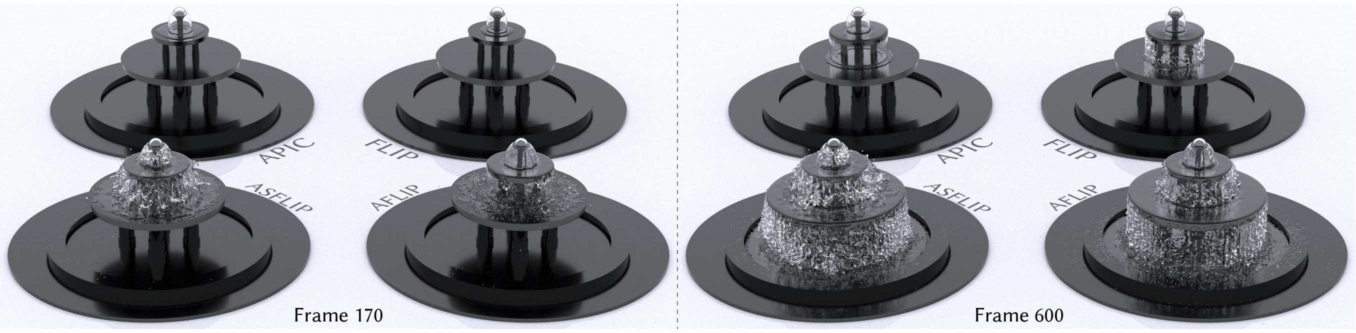

Fig. 1. **Water Spray from a Fountain Nozzle.** Weakly-compressible water is simulated with the material point method. A weak frictional coefficient (0.0125) is applied between the liquid and plates. AFLIP and ASFLIP induce much less dissipation than the traditional methods, so liquid particles are more spread out within the same period. Particles advected by ASFLIP are also more energetic than AFLIP when leaving the top plate since particles are used to assist friction computation. ©2021 Tencent

The material point method (MPM) recently demonstrated its efficacy at simulating many materials and the coupling between them on a massive scale. However, in scenarios containing debris, MPM manifests more dissipation and numerical viscosity than traditional Lagrangian methods. We have two observations from carefully revisiting existing integration methods used in MPM. First, nearby particles would end up with smoothed velocities without recovering momentum for each particle during the particle-grid-particle transfers. Second, most existing integrators assume continuity in the entire domain and advect particles by directly interpolating the positions from deformed nodal positions, which would trap the particles and make them harder to separate. We propose an integration scheme that corrects particle positions at each time step. We demonstrate our method's effectiveness with several large-scale simulations involving brittle materials. Our approach effectively reduces diffusion and unphysical viscosity compared to traditional integrators.

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

Additional Key Words and Phrases: integration, material point method, water, sand, hair, cloth

ACM Reference Format:
Yun (Raymond) Fei, Qi Guo, Rundong Wu, Li Huang, and Ming Gao. 2021. Revisiting Integration in the Material Point Method: A Scheme for Easier Separation and Less Dissipation. *ACM Trans. Graph.* 40, 4, Article 109 (August 2021), 16 pages. https://doi.org/10.1145/3450626.3459678

1 INTRODUCTION

Dynamics of millions of particles or elements are common in real-life: an off-road vehicle accelerates on a beach, stirring up sand particles into the air; a gust of wind blows, loosening one's hair into scattered strands; water spouts from a fountain, splashing on the slates and breaking into shiny droplets. These scenarios are challenging to simulate since the collision and separation need to be resolved correctly.

The **M**aterial **P**oint **M**ethod (**MPM**) [Sulsky et al. 1994] was recently shown to be suitable for digitally reproducing quite a large extent of complex materials and physical phenomena on a massive scale [Jiang et al. 2016]. MPM solves the equations of motion on a uniform or adaptive grid and performs advection with particles. The non-slip contacts are resolved on the grid naturally without paying extra costs like collision detection or resolution, making MPM an effective discretization method for capturing the dynamics of millions of particles or elements.

Nevertheless, MPM is more dissipative than Lagrangian methods that assume particles or elements as discrete (e.g., **D**iscrete

Authors' addresses: Yun (Raymond) Fei, Tencent Game AI Research Center, Los Angeles, USA, raymondfei@tencent.com; Qi Guo, Tencent Game AI Research Center, Los Angeles, USA, kiguo@tencent.com; Rundong Wu, Tencent Game AI Research Center, Los Angeles, USA, rundongwu@tencent.com; Li Huang, Tencent Game AI Research Center, Shenzhen, P. R. China, liiiihuang@tencent.com; Ming Gao, Tencent Game AI Research Center, Los Angeles, USA, mingwiscgao@tencent.com.

0730-0301/2021/8-ART109 $15.00
https://doi.org/10.1145/3450626.3459678

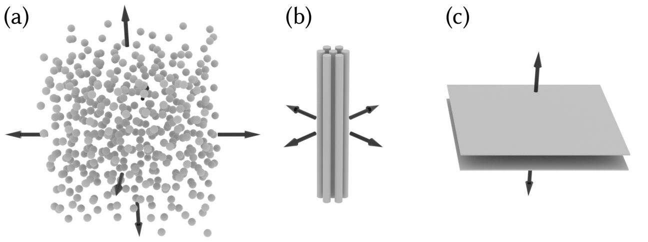

Fig. 2. **Examples of material that is brittle in one or more directions.** (a) dry sand that is brittle in all directions; (b) hair strands that are brittle in the directions perpendicular to their tangential directions; and (c) clothes that are brittle in the directions perpendicular to the clothes' surfaces.

Table 1. **Summary of various integrators.** For clarity, we summarize the features and effects of multiple integrators investigated in this paper. The features include whether the integrator diminishes smoothing of velocity in G2P to have a more energetic simulation, whether it may preserve sub-grid motion to have an easier separation of particles, whether the affine motion is undamped, and whether it correctly handles the boundary.

Integrators	Diminish G2P Smoothing	Preserve Sub-grid Motion	Undamp Affine Motion	Handle Boundary Condition	Easy Separation	Energetic Level
PIC	✗	✗	✗	✓	✗	★
APIC	✗	✗	✓	✓	✗	★★
FLIP	✓	✗	✗	✓	✗	★★
NFLIP	✓	✓	✗	✗	✓	★★
SFLIP	✓	✓	✗	✓	✓	★★
AFLIP	✓	✗	✓	✓	✗	★★★
ASFLIP	✓	✓	✓	✓	✓	★★★

Element **M**ethod, or **DEM** [Yue et al. 2018]), especially when simulating scenarios containing debris. MPM assumes the material to be continuous. Such a hypothesis forbids a piece of material to freely separate from other pieces in its vicinity. In most cases, continuity is desired and required to track the stress inside the material. However, the continuum hypothesis does not hold for separating objects: even if the objects are close to each other, they should have discrete and ballistic movements during separation.

Even if compared with other Lagrangian methods that also adopt the continuum hypothesis (e.g., **S**moothed **P**article **H**ydrodynamics, or **SPH**), MPM may introduce stronger numerical viscosity [Chen et al. 2020; Yang et al. 2017]. Transferring information from particles to the grid would average the momentum at grid nodes even when nearby particles have velocities with opposite directions; on the other hand, if particle velocities are then directly interpolated from nodal velocities, another round of velocity smoothing is performed. Furthermore, to avoid crossing-cell instability [Gao et al. 2017], wide kernels like B-spline quadratic and cubic kernels are generally used in MPM to enforce C^1 continuity, which makes the smoothing effect even more significant.

We notice that integrators adopted in MPM, from early methods like **P**article-**I**n-**C**ell (**PIC**) [Harlow et al. 1955] and **FL**uid-**I**mplicit-**P**articles (**FLIP**) [Brackbill and Ruppel 1986] to the most recently proposed **A**ffine-augmented **PIC** (**APIC**) [Jiang et al. 2015], **M**oving-**L**east-**S**quares (**MLS**) [Hu et al. 2018], and **Poly**nomial **PIC** (PolyPIC) [Fu et al. 2017], play a crucial role in enforcing the continuum hypothesis and deciding the numerical properties of the simulated dynamics.

In this paper, we try to mitigate the numerical viscosity during particle-grid transfers and improve upon cases where the continuum hypothesis no longer holds. We focus on materials that are brittle in one or more directions (Fig. 2) and start by carefully evaluating the integration method proposed by Stomakhin et al. [2013], which we denote as **N**aturally-modified **FLIP** or **NFLIP**[1]. We reveal NFLIP's advantages over previous methods and also some of its critical defects. We then present a **S**eparable **FLIP** method, or **SFLIP**, to break the continuum hypothesis and untrap a particle from the region determined by the deformed grid nodes in its vicinity. As a result, we provide a new option to model the debris with much less dissipation in the MPM framework.

We further note that a simple combination between APIC and FLIP manifests a better trade-off between energy-preservation, stability, and computational costs. As the velocity update of this scheme follows FLIP-style rather than PIC-style, we prefer to denote it as **A**ffine-augmented **FLIP** or **AFLIP** for abbreviation. By keeping track of velocity modes of low and high frequencies separately, AFLIP can preserve intricate structures that would only be reproduced with refined resolution in traditional methods.

To extend the idea of SFLIP to APIC, both the positional and velocity updates of APIC during the grid-to-particle phase need to be modified to achieve easier particle separation, and we denote this novel scheme as **ASFLIP**.

Our contributions in this paper are the investigation of existing integrators and also the development of new integrators for MPM (summarized in Table 1), which include:

- revisiting the continuum hypothesis and the numerical viscosity of various integrators commonly adopted in MPM (§3);
- proposing AFLIP that is capable of generating more energetic dynamics with almost zero extra computational costs (§5.2); and
- presenting a separable integration scheme and applying it to both FLIP (i.e., SFLIP, §4) and AFLIP (i.e., ASFLIP, §5.3) to break the continuum hypothesis among separating particles.

2 RELATED WORKS

The Particle-in-Cell (PIC) and its variants. The PIC method was first developed in the early 1950s [Harlow et al. 1955] and was used primarily in simulating compressible fluid dynamics. However, it is well-known that PIC suffers from excessive numerical dissipation [Jiang et al. 2015]. The vanilla PIC can be extended to the fourth-order accuracy for general transport problems using the moving least-square (MLS) and weighted essentially non-oscillatory (WENO) interpolation [Edwards and Bridson 2012]. Jiang et al. [2015; 2017b] developed the APIC method as an alternative to PIC by replacing the local constant velocity field of particles with the affine velocity field, dedicating to preserving the rotational and shearing information. Based on this method, Fu et al. [2017] developed the PolyPIC method to reduce the information loss when transferring between the background grid and particles. They generalized the

[1] We name the integrators mainly from their velocity update equation in G2P. If a scheme follows PIC-style velocity update, it will be denoted as xPIC, e.g. APIC and ASPIC in §5.1. If an integrator adopts FLIP-style velocity update Eq. (1), it will be named xFLIP, e.g., SFLIP in §4, AFLIP in §5.2 and ASFLIP in §5.3.

locally affine velocity field to a higher-order polynomial representation of the velocity field. Hu et al. [2018] showed that APIC and PolyPIC are consistent with a Galerkin-style Moving Least Squares (MLS) discretization. Recently, Ding et al. [2020] investigated the incompressible fluid simulation with APIC on the MAC grid and applied the Fourier analysis to compare different transfer schemes.

The Fluid-Implicit-Particles (FLIP) and its variants. Brackbill and Ruppel [1986] introduced the FLIP method to address this issue by interpolating the difference between the old and updated grid velocity to particles and achieving almost zero-dissipation for compressible fluid simulation results. Even if the original FLIP method was brought up in early years, it was introduced to graphics by Zhu and Bridson until decades after to animate the sand flow [Zhu and Bridson 2005]. Since then, the FLIP method has grown popular in computer graphics, especially in fluid simulation. Batty et al. [2007] coupled FLIP fluid with solids with complex boundaries. Ando et al. [2012] used adaptively sampled FLIP to preserve the fluid sheets. Boyd and Brison [2012] introduced the multi-FLIP method to simulate the two-phase flow. Cornelis et al. [2014] combined the Implicit Incompressible Smoothed Particle Hydrodynamics (IISPH) [Ihmsen et al. 2013] and FLIP for fluid simulation. Researchers also developed the narrow band FLIP to achieve similar visual results with a reduced number of particles and simulation cost [Ferstl et al. 2016; Sato et al. 2018]. There have been attempts to modify the FLIP solver to mitigate the visual artifacts too. Ando et al. [2012] and Um et al. [2014] addressed the artifacts of spatial particle distribution and relieved them by adding the correction to particle positions. Gerszewski and Bargteil [2013] modified FLIP to simulate large-scale splashing, aiming to avoid artifacts such as numerical surface tension and stickiness. Hammerquist and Nairn [2017] provided XPIC to achieve a balance between PIC and FLIP by reformulating PIC/FLIP transfer into a constrained minimization problem and achieved better visual results with periodical execution of XPIC in a FLIP simulation. In this work, we provide another modification to FLIP, targeting less dissipation and easier particle separation.

The Material Point Method (MPM) and its variants. Originating from FLIP, MPM was developed to simulate the continuum solids [Sulsky et al. 1994, 1995]. Surveys on MPM have been done by Jiang et al. [2016] and Hu et al. [2019]. Among these works, Jiang et al. [2015] extended MPM by embedding Lagrangian forces into the MPM grid (denoted as *Lagrangian MPM* for brevity). The Lagrangian MPM contains three phases: 1) computing Lagrangian forces through non-MPM methods (e.g., finite element method), 2) interpolating Lagrangian forces onto an Eulerian grid, and 3) integrating together with Eulerian forces computed from classical MPM.

A benefit of Lagrangian MPM is that the contacts between codimensional objects can be naturally resolved on the grid by carefully decomposing the particle strain into normal and tangential directions [Jiang et al. 2017a]. However, Lagrangian MPM may suffer from the undesired numerical stickiness due to the unmatched resolution between the grid and mesh [Guo et al. 2018]. For codimensional objects that are densely packed, the numerical stickiness is significant unless the MPM grid's cell size is comparable to the averaged gap size between these objects, which can be either unpractical or very cost-consuming. Alleviating the undesired numerical stickiness in Lagrangian MPM is still an open problem. In this work, we address this issue by modifying the integrator used. Our method applies to materials that are brittle in one or more directions (e.g., sand, clothes, or hair strands) and does not require much modification to the MPM framework.

Modified Lagrangian methods. In Lagrangian MPM, Lagrangian elements help MPM keep track of the direct relationship between particles, while in traditional MPM, such a relationship can only be maintained indirectly via grid nodes. On the other hand, particle-grid hybrid solvers (e.g., MPM [Han et al. 2019], and incompressible Poisson solver [McAdams et al. 2009]) can also help Lagrangian methods for alleviating collision handling. Heuristics are used to determine whether separation is desired and prevent unphysical stickiness due to the continuum assumption. Similar to such heuristics, our method adopts the volume ratio as the criterion.

3 BREAKING THE POSITIONAL TRAP

3.1 Revisit PIC, APIC, and FLIP in the context of MPM

Before introducing our new method, we revisit three existing methods used in MPM for integration, namely, PIC, APIC, and FLIP. In the following discussion, we denote the particle velocities and positions as $\mathbf{v}_p$ and $\mathbf{x}_p$, respectively, and denote the velocities stored on the i-th node as $\mathbf{v}_i$. We also use the superscript to denote the index of time steps, e.g., $\mathbf{v}_p^n$ for the particle velocity at the n-th time step. The time step size is denoted as Δt.

PIC integrates particle velocities and positions through the following steps:

(1) (P2G) for each particle (located at $\mathbf{x}_p^n$), find its neighboring nodes (located at $\mathbf{x}_i^n$) on the grid, and transfer its mass and momentum to these nodes through a continuous interpolation function $w(\mathbf{x}_i^n, \mathbf{x}_p^n)$ (or w_{ip} for abbreviation); then, for each grid node, the accumulated momentum is divided by the accumulated mass to compute nodal velocity $\mathbf{v}_i^n$;
(2) on the grid, calculate forces on nodes by integrating the stress over the neighboring particles, update the nodal velocities, and then resolve the collision with boundaries to get $\mathbf{v}_i^*$;
(3) (G2P) transfer the nodal velocities $\mathbf{v}_i^*$ back to particles through the same function w_{ip} and update the particle positions.

The particle dynamics can be stably integrated with the PIC method but notably suffers from excessive dissipation [Brackbill and Ruppel 1986]. The nodes usually have fewer degrees of freedom than the particles (e.g., one node corresponds to eight or more particles in three dimensions [Jiang et al. 2015]). Thus, even if the particles have some high-frequency velocity modes, they are eliminated after being transferred to the grid [Hammerquist and Nairn 2017].

There is another way to interpret the dissipation in PIC. Since the interpolation function w_{ip} is continuous in the entire domain, a continuous velocity field can be constructed from the nodal velocities over the whole domain. The benefit of having a continuous velocity field is obvious [Brackbill and Ruppel 1986]: it ensures the uniqueness of velocity and allows to define differential operators. However, in specific scenarios, enforcing a continuous velocity field everywhere may bring dissipation and unphysical viscosity. For

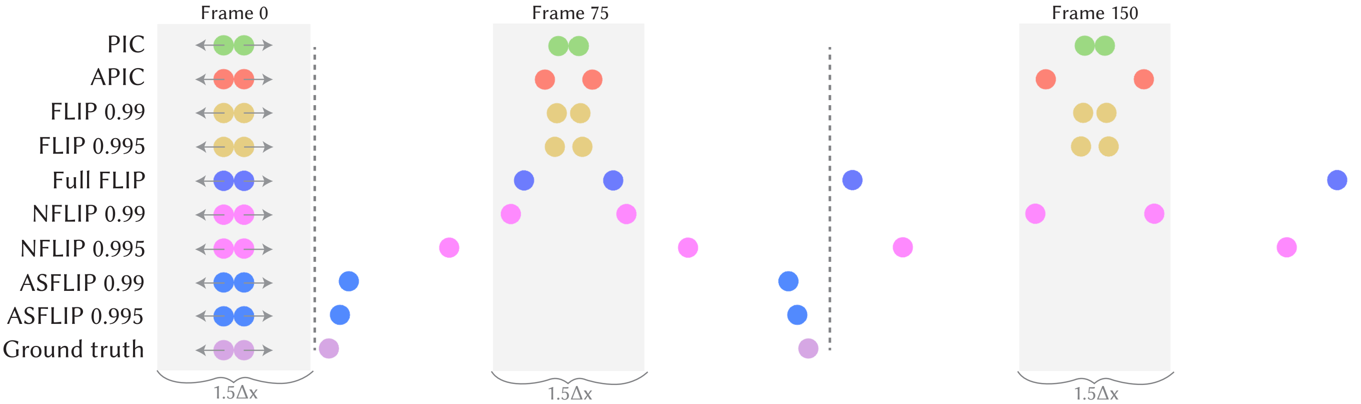

Fig. 3. **Comparison between different advection methods.** Two particles are moving away from each other with velocities of the same magnitudes but opposite directions. The cell size is Δx and the initial distance between them is $0.2\Delta x$. For simplicity, $\mathbf{v}_i^* = \mathbf{v}_i^n$ is assumed: sand or any non-cohesive material can be used, i.e., no force is generated due to the separation, and gravity is turned off. B-spline weighting function is adopted, and the width of the kernel is $1.5\Delta x$ (whose affected region is colored in grey). The ground truth is analytic, i.e., $\mathbf{x}_p = t\mathbf{v}_p$. We denote xFLIP with $\alpha = 0.99$ simply as xFLIP 0.99.

example, when strands inside the same cluster tend to separate from each other, the segments inside the same cell or neighboring cells should have velocities with opposite directions. However, these velocities are smoothed after particle-grid transfers since only continuous velocity is allowed in PIC, and the tendency of separation is depressed. As a consequence, the simulated strands may exhibit excessive viscosity. Similarly, when simulating liquid, some tiny droplets inside the same cell may start with very different velocities. Nevertheless, since PIC enforces a continuous velocity field, it would eliminate the differences in droplets' velocities in a few iterations, and the droplets would not get separated further.

APIC, recently proposed by Jiang et al. [2015], has been widely adopted as an extension to PIC. APIC stores an extra momentum matrix $\mathbf{C}_p^n$ for keeping track of the approximation to local velocity gradient. In G2P, while these two methods share the same update rule of particle velocities and positions, APIC also needs to update $\mathbf{C}_p^n$. And in P2G, the matrix is used to compensate the particle's affine motion w.r.t each neighboring node. APIC produces more energetic simulations by better preserving the affine motion than PIC. The affine matrices from different particles may add velocity increments with different directions, but such increments would still get averaged at grid nodes. Thus, APIC suffers from the same problem: the discontinuous particle velocities are smoothed during a few iterations of P2G and G2P in the scenarios described above, i.e., separating strands and droplets.

FLIP, originally proposed by Brackbill [1986], attempts to mitigate the dissipation in PIC by preserving particle velocities. We first summarize the P2G and G2P transfers of FLIP as:

$$\text{FLIP P2G:} \qquad m_i^n \mathbf{v}_i^n = \sum_p w_{ip} m_p \mathbf{v}_p^n$$

$$\text{FLIP G2P:} \qquad \mathbf{v}_p^{n+1} = \sum_i w_{ip} \mathbf{v}_i^* + \alpha \left(\mathbf{v}_p^n - \sum_i w_{ip} \mathbf{v}_i^n \right) \tag{1}$$

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \sum_i w_{ip} \mathbf{v}_i^* \tag{2}$$

where α is the PIC-FLIP blending ratio [Bridson 2015]. And we name $\alpha \left(\mathbf{v}_p^n - \sum_i w_{ip} \mathbf{v}_i^n \right)$ as the *velocity adjustment* for convenience in later discussion. With $\alpha = 0$, Eq. (1) changes back to PIC, and with $\alpha = 1$ it would turn to *full FLIP*. In the following discussion, we assume FLIP corresponds to $0 < \alpha \leq 1$ to exclude its connection to PIC. Notice FLIP still uses the nodal velocities to integrate the particle positions similar to PIC and APIC.

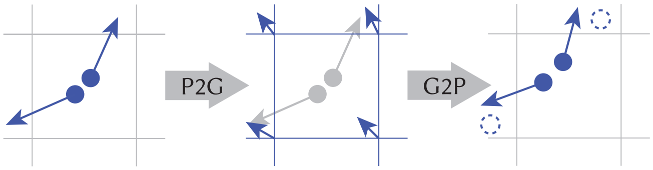

Fig. 4. **Using FLIP to integrate two separating particles.** Left: two particles are separating with large velocities in opposite directions. Middle: the velocities are averaged to nodes after P2G. Right: Because the nodal velocities are used to integrate positions, the particles have not been moved to their deserved positions (dashed circles). Instead, they only slightly separate from each other, even if they should be significantly separated from each other according to their velocities.

The simulation results from FLIP are usually more energetic than those produced with PIC [Brackbill and Ruppel 1986; Bridson 2015] or even APIC [Jiang et al. 2015] in certain scenes (e.g. Fig. 21). The ability of FLIP to retain momentum for each particle owes to the velocity adjustment term and the velocity field is no longer continuous. It becomes a mix between the continuous field defined from nodal velocities and the velocities defined as Dirac delta functions on particles. The particle velocities contain higher frequencies that may get eliminated by P2G and G2P [Hammerquist and Nairn 2017]. The more this term in $\mathbf{v}_p^{n+1}$ is reserved (with a larger α used in Eq. (1)), the more energetic the FLIP simulation would be.

Positional trap. Like PIC and APIC, FLIP still uses the nodal velocity to update particle positions (as shown by Eq. (2)). While FLIP breaks the velocity continuity, it still suffers from dissipation and unphysical viscosity. Fig. 4 shows an example of such dissipation that happens during the separation of two particles.

More concretely, we can expand the right-hand side of the advection equation shared by the three methods and show that the particle positions are interpolated from updated nodal positions in

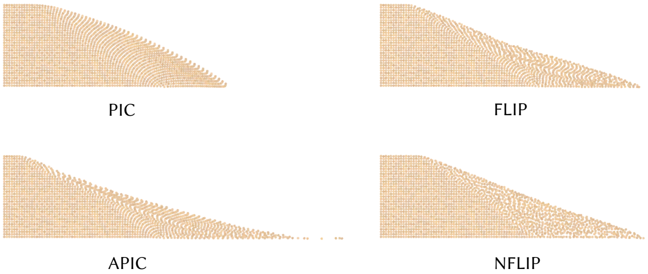

Fig. 5. **The collapse of a sand column.** When initialized on vertical lines, the sand particles could stay in the same (but curved) line due to the positional trap during the collapse when simulated with PIC, FLIP, or APIC. Advecting with NFLIP, on the other hand, would eliminate this artifact.

PIC, APIC, and FLIP, where

$$\begin{aligned}\mathbf{x}_p^{n+1} &= \mathbf{x}_p^n + \Delta t \sum_i w_{ip}\mathbf{v}_i^* = \sum_i w_{ip}\mathbf{x}_i^n + \Delta t \sum_i w_{ip}\mathbf{v}_i^* \\ &= \sum_i w_{ip}\left(\mathbf{x}_i^n + \Delta t \mathbf{v}_i^*\right) = \sum_i w_{ip}\mathbf{x}_i^*\end{aligned}$$

The particles are *trapped* inside the region formed by deformed grid nodes in their vicinity due to the interpolation, even if their velocities indicates a significant separation at the next time step. We name this behavior as the *positional trap*.

3.2 Untrap the Particles

Our first attempt to untrap the particles is to match the position update with the FLIP velocity update. In other words, we directly integrate the velocity adjustment term by Δt and add to particle positions at the next time step:

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \left[\sum_i w_{ip}\mathbf{v}_i^* + \alpha\left(\mathbf{v}_p^n - \sum_i w_{ip}\mathbf{v}_i^n\right)\right]. \tag{3}$$

This method was initially proposed in work by Stomakhin et al. [2013] for snow simulation. We refer to this method as Naturally-modified FLIP or NFLIP, and refer to $\Delta t\alpha\left(\mathbf{v}_p^n - \sum_i w_{ip}\mathbf{v}_i^n\right)$ as the *positional adjustment*. In the following discussion, we propose a detailed analysis of this method.

Similar to the velocity adjustment, the positional adjustment contains high-frequency sub-grid displacement that has not been eliminated during the P2G and G2P process. The positional trap can be eased by adding such information to the integration of positions. In Fig. 4, this positional adjustment would move the particles closer to the locations indicated by the dashed circles, providing the trap-breaking capability.

We compare NFLIP with other methods in Fig. 3, where two particles initially have velocities of the same magnitude but opposite directions. In this setup, since there is no external force to deter particles from separation, particles should keep moving away from each other. However, the particles advected by PIC or a damped FLIP (either $\alpha = 0.99$ or $\alpha = 0.995$) lose individual momenta after a few time steps due to two reasons: 1) velocities with opposite directions are averaged and canceled at grid nodes; 2) particles are not moving fast enough to escape from the weighting kernel's radius.

On the other hand, particles advected by the NFLIP with $\alpha = 0.995$ or full FLIP can move far away from each other since the positional adjustment makes NFLIP move faster, and full FLIP manages to preserve the momenta. We also notice that in NFLIP with $\alpha = 0.99$, the two particles still fail to escape, which implies that the multiplier on the positional adjustment needs to be large enough to have the particles move away in this case.

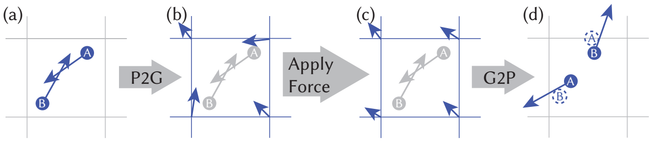

Fig. 6. **Using NFLIP to integrate two colliding particles.** (a) two particles (marked with A and B) are moving closer with large velocities; (b) the velocities are averaged to nodes after P2G; (c) collisional force is applied to the nodal velocity; (d) because the particle velocities are used to integrate positions, the particles penetrate each other and ignore any collisional force applied. Instead, if the nodal velocity is used to integrate the particle positions, the particles would end up in the dashed circles.

For completeness, APIC [Jiang et al. 2015] is also included in the experiment. As it improves upon PIC, the particles can eventually escape since the affine components are preserved. However, the loss of linear momentum makes the particles move much slower than those in full FLIP and NFLIP ($\alpha = 0.995$), indicating dissipation.

As discussed in §4, SFLIP is exactly the same as NFLIP in this scenario since there is no compression at all. Meanwhile, ASFLIP outperforms all other integrators.

In practice, the full FLIP is known to produce noisy and unstable results [Bridson 2015; Hammerquist and Nairn 2017; Jiang et al. 2015]. Therefore, the NFLIP with a large α (e.g., $\alpha = 0.995$) seems to be a better choice (Fig. 5).

However, NFLIP has its own defects. When particles are being compressed, the trap-breaking capability might lead to unphysical penetration (e.g., as shown in Fig. 6, and more penetration cases revealed in the supplemental video), or even velocity *multi-streaming*: particles at the same location may end up with very different velocities [Brackbill and Ruppel 1986]. Nearby particles ignore the collisions between each other since the stress is only resolved on the nodes. Each particle has its displacement, to some extent, independent of the nodal velocities.

We may also discover this issue through analyzing Eq. (3). Imagine two particles in 1D approach each other with velocities $\mathbf{v}_{p_0}^n$ and $\mathbf{v}_{p_1}^n$. After P2G and G2P, the grid-filtered velocities, i.e. $\bar{\mathbf{v}}_{p_0}^n = \sum_i w_{ip_0}\mathbf{v}_i^n$ and $\bar{\mathbf{v}}_{p_1}^n = \sum_i w_{ip_1}\mathbf{v}_i^n$, would have smaller magnitudes than the particle velocities, i.e., $|\bar{\mathbf{v}}_{p_0}^n| < |\mathbf{v}_{p_0}^n|$ and $|\bar{\mathbf{v}}_{p_1}^n| < |\mathbf{v}_{p_1}^n|$. As a result, the two particles advected by NFLIP would be closer to each other at the end of the time step than if they were advected by FLIP. There could be penetration if either the time step Δt or α is large.

Another issue is that NFLIP may not correctly handle the particle-boundary collisions. In MPM, the collision between the simulated material and boundaries is resolved on the nodes instead of the particles[2]. For example, when the grid nodes inside static collision objects are enforced to have zero velocities, particles advected with either FLIP or NFLIP could still have non-zero velocities when they

[2] If the collision is resolved on the particles, it is unclear how the resolved collision would affect the particles' deformation gradient. It is also possible to keep the deformation gradient constant during the collision resolution, but doing so will leave the particles erroneously stack near the boundary [Klár et al. 2016].

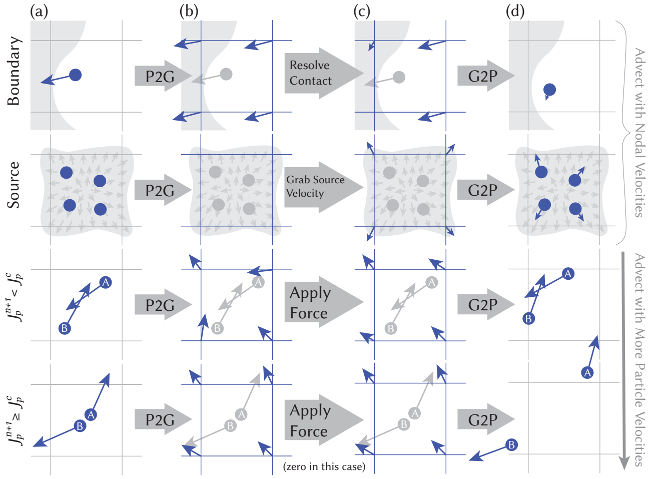

Fig. 7. **Using SFLIP to integrate two particles in different cases. First row:** (a) a particle approaches the boundary; (b) particle velocity is transferred to the nodes; (c) after resolving contacts, velocities on the nodes inside and perpendicular to the boundary are eliminated; (d) SFLIP use the grid velocity to advect particles, so that the collision is correctly handled. **Second row:** (a) new particles are sampled from a material source with zero velocity; (b) particle velocity is transferred to the nodes (zero in this case); (c) source velocity is assigned to the nodes inside it; (d) SFLIP use the grid velocity to advect particles, so that these new particles share the velocity of the source. **Third row:** (a) two particles (marked with A and B) are approaching each other with large and opposite velocities; (b) the velocities are averaged to nodes after P2G; (c) collisional force is applied to the nodal velocity; (d) SFLIP uses nodal velocity to integrate position since $J_p^{n+1} < J_p^c$, which avoids penetration. **Fourth row:** (a) two particles are separating with large velocities in opposite directions; (b) velocities are averaged to nodes after P2G; (c) forces are applied to the nodes (zero since we assume the material has broken); (d) SFLIP uses particle velocity to integrate position, so that the particles may freely separate.

move into the boundary. Furthermore, the positional adjustment of NFLIP could continuously move the particles deep into the boundary, making the penetration much worse than FLIP.

4 A SEPARABLE FLIP SCHEME

To fix all the issues of NFLIP, we propose a **S**eparable **FLIP** scheme, or **SFLIP**, so that the particles can correctly collide and freely separate from each other.

We add a new parameter β_p, namely the trap-breaking ratio, to better control the magnitude of the positional adjustment. With this modification, the equation for the integration of particle positions becomes

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \left[\sum_i w_{ip} \mathbf{v}_i^* + \beta_p \alpha \left(\mathbf{v}_p^n - \sum_i w_{ip} \mathbf{v}_i^n \right) \right] \tag{4}$$

The particle advection should correctly handle the simulation's boundary conditions (e.g., source of particles and collisional boundaries). It should allow the particles to collide when compressed and allow the particles to expand freely when the material reaches its critical failure stress. In a sense, we propose a multi-scale scheme: equations of motion are resolved on a regular grid scale, while particle states determine the separations on a sub-grid scale. The β_p for each particle is computed with the following scheme:

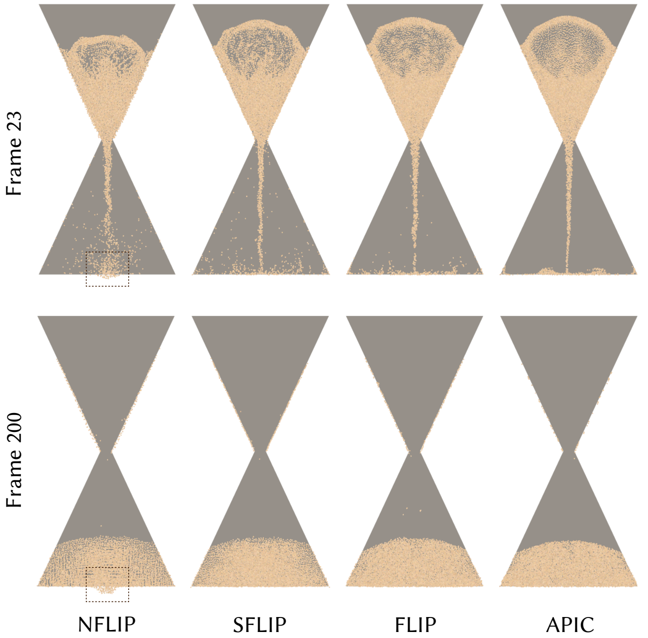

Fig. 8. **Sand pouring into an hourglass.** Simulations by NFLIP and SFLIP behave more energetically (i.e., more debris particles in the open area) than FLIP and APIC. Furthermore, in NFLIP, particles penetrate deep into the ground (marked by a dashed rectangle) while SFLIP correctly handles the boundary condition.

$$\beta_p = \begin{cases} 0 & \text{if } \phi_{\mathbf{x}_p^n + \mathbf{v}_p^n \Delta t} < 0 \text{ and } \nabla \phi_{\mathbf{x}_p^n + \mathbf{v}_p^n \Delta t} \cdot \mathbf{v}_p^n \leq 0 \\ 0 & \text{otherwise if } \varphi_{\mathbf{x}_p^n + \mathbf{v}_p^n \Delta t} < 0 \\ \beta^{\min} & \text{otherwise if } J_p^{E,n+1} < J_p^c \\ \beta^{\max} & \text{otherwise} \end{cases} \tag{5}$$

where J_p^c is critical volume ratio (see below). $\phi_{\mathbf{x}}$ is the signed distance from $\mathbf{x}$ to its nearest boundary and we assume negative values indicate the interior of the boundary; $\nabla \phi_{\mathbf{x}}$, the corresponding gradient at $\mathbf{x}$, gives the normal direction pointing towards outside. Thus each particle that resides inside the collisional object and is not moving outwards will have $\beta_p = 0$. $\varphi_{\mathbf{x}}$ is similar to $\phi_{\mathbf{x}}$ but it is exclusively for material sources (e.g., the fountain nozzle in Fig. 1).

For elastoplastic materials [Bonet and Wood 1997], the deformation gradient is decomposed into the elastic and plastic parts as $\mathbf{F}_p^{n+1} = \mathbf{F}_p^{E,n+1} \mathbf{F}_p^{P,n+1}$, and $J_p^{E,n+1} = \det\left(\mathbf{F}_p^{E,n+1}\right)$ represents the ratio of the elastic volume to initial volume for a particle. For most materials, plastic deformation should not introduce volume change [Bridgman 1949], i.e. $J_p^{P,n+1} = \det\left(\mathbf{F}_p^{P,n+1}\right) = 1$, thus the condition $J_p^{E,n+1} < J_p^c$ is equivalent to $J_p^{n+1} < J_p^c$, which we use in the following discussion for brevity.

Choosing J_p^c. In this work, we focus on materials that are brittle (i.e., with *low ductility*) in all (e.g., sand) or partial directions (e.g., directions perpendicular to a hair strand). These materials cannot remain intact in these directions after the dilational stress makes the

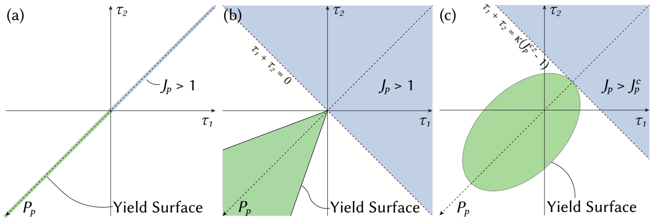

Fig. 9. **The yield criteria and J_p in the principal stress space.** We mark the feasible region in green and the region where $J_p > J_p^c$ (or $J_p > 1$ when $J_p^c = 1$) in blue. Additionally, we draw the axis of pressure P_p in dashed arrow. a) The *weakly-compressible liquid* [Tampubolon et al. 2017] where only the dilational stress is applied (i.e., $\tau_1 = \tau_2$) and the stress is only non-zero when the material is compressive. b) The *Drucker-Prager* yield criterion with Simo's neo-Hookean elasticity [Yue et al. 2018] used. c) The *NACC* yield criterion with Simo's neo-Hookean elasticity [Wolper et al. 2019] used, where κ is the bulk modulus.

material yield. Therefore, we assume these materials break exactly when their dilational stresses reach the yield stress.

We choose J_p^c such that when $J_p^{n+1} \geq J_p^c$ the material yields, or stress-free if it is non-cohesive. In the space of principal Kirchhoff stress $\boldsymbol{\tau} = \{\tau_1, \tau_2, \tau_3\}$ (or $\boldsymbol{\tau} = \{\tau_1, \tau_2\}$ in 2D), J_p^c corresponds to the case where $\boldsymbol{\tau}$ is at the up-right-front tip (or the up-right tip in 2D) of a specific plastic yield criterion (Fig. 9). We take three models used in this paper to illustrate the case of J_p^c, while for other models, similar computational schemes may follow.

(1) For a weakly-compressible liquid[3], the deformation gradient is projected to identity (stress-free) when $J_p > 1$. Therefore, we take $J_p^c = 1$ for this kind of material.
(2) The (non-cohesive) Drucker-Prager [1952] yield criterion has a tip corresponding to $\tau_x = \tau_y = \tau_z = 0$. For Simo's [1988] neo-Hookean elastic model, this state indicates $J_p = 1$. Therefore, we take $J_p^c = 1$ for materials using this yield criterion.
(3) The non-associative Cam-clay (NACC) yield criterion [Wolper et al. 2019] projects the deformation gradient to expansional tip when $P_p \leq -\chi P_{p,0} \equiv -\chi\kappa \sinh\left(\xi \max(-\alpha_p, 0)\right)$, where P_p is the pressure, χ is the cohesion coefficient, κ is the bulk modulus, ξ is the hardening factor, and α_p is a variable used to track hardening. When using Simo's [1988] neo-Hookean elastic model, the pressure can be simply calculated from the derivative of dilational energy W_v as [Stomakhin et al. 2014]:
$$P_p = -\frac{\partial W_v(J_p^c)}{\partial J_p^c} = -\frac{1}{2}\kappa(J_p^c - \frac{1}{J_p^c}) = -\chi P_{p,0}.$$
Solving for J_p^c, we have
$$J_p^c = \frac{\chi P_{p,0}}{\kappa} + \sqrt{\left(\frac{\chi P_{p,0}}{\kappa}\right)^2 + 1}.$$

Material ductilities. Per our assumption that the material has a low ductility (e.g., sand with a Drucker-Prager plastic flow [Yue et al. 2018], or weakly compressible liquid [Tampubolon et al. 2017]), we adopt a simple criterion for the separation of particles. For other materials with different ductilities, such as metal with a von Mises yield condition [Wolper et al. 2019], more sophisticated criterion (e.g., computed from a separation tensor [O'Brien and Hodgins 1999]) may be adopted.

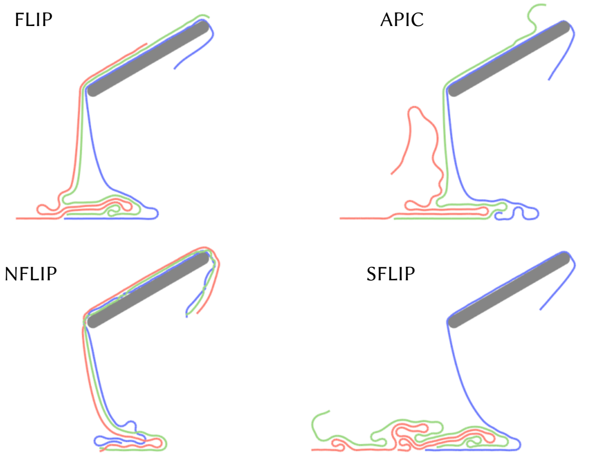

Fig. 10. **Fiber Piling.** Friction between fibers has been turned off. Ideally, the fibers should rapidly slide without delay or penetration. With NFLIP, fibers penetrate each other and result in incorrect behavior. Fibers with SFLIP slide down much faster than FLIP and APIC.

Consistency between deformation gradient and particle motion. The consistency between $\mathbf{F}_p$ and $\mathbf{x}_p$ should be enforced when the continuum hypothesis holds. However, when the particle motion is discrete, i.e., the continuum assumption is no longer applicable, $\mathbf{F}_p$ does not need to match with the change of $\mathbf{x}_p$ anymore. The plasticity models adopted in this work would project $\mathbf{F}_p$ to the upper-right-front tip of the yield surface when J_p reaches J_p^c. In this case, no matter whether our positional adjustment is applied, $\mathbf{F}_p$ keeps to be a constant (e.g., identity for dry sand) and is irrelevant to the particle motion. In other words, our method attempts to address scenarios where continuum assumption should not be applied.

Choosing β^{min} and β^{max}. For particles with $J_p^{n+1} < J_p^c$, their β_p is set to some small value $\beta^{\min}$. And all other particles should have $\beta^{\max}$ to encourage easier separation similar to NFLIP. For the examples proposed in this paper, $\beta^{\min}$ is zero in most cases and 0.05 in the others while $\beta^{\max}$ varies from 0.1 to 1.

When the positional adjustment is diminished ($\beta^{\min}$ is zero or very small), only the grid velocities will be used to integrate the position so that the positional displacement contains most or all of the collisional impulses resolved on the grid to largely suppress particle-particle and particle-boundary penetrations.

We illustrate the four cases of Eq. (5) in Fig. 7, whose rows correspond to the particle-boundary collision, particle-source interaction, particle-particle collision, and otherwise.

Implementation details. For general MPM materials, the deformation gradient is updated as
$$\mathbf{F}_p^{n+1} = (\mathbf{I} + \Delta t \nabla \mathbf{v}_p^{n+1})\mathbf{F}_p^n$$
Thus we simply compute the updated J_p as $J_p^{n+1} = \det(\mathbf{F}_p^{n+1})$.

[3] By weakly-compressible liquid we denote the constitutive model where the Murnaghan [1944] equation of state governs the pressure, i.e., $P_p(J_p) = \frac{\kappa}{\gamma}\left(J_p^{-\gamma} - 1\right)$ (where κ is the bulk modulus and γ is κ's first derivative w.r.t. pressure), and $J_p \leftarrow 1$ once $J_p > 1$. Please refer to the work by Tempubolon et al. [2017] for more details in the context of MPM.

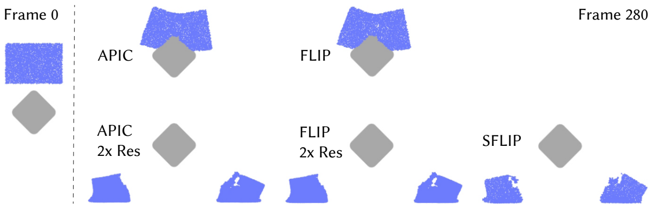

Fig. 11. **Fracture.** A rectangle with the NACC plastic model falls onto a wedge and then collides onto the ground. The rectangle breaks into pieces with SFLIP, FLIP (2× resolution) and APIC (2× resolution).

For MPM materials where the stress depends only on J_p (e.g., weakly-compressible liquid), we provide a more accurate computation of J_p^{n+1} when Δt is large[4],

$$J_p^{n+1} = \exp(\Delta t \nabla \cdot \mathbf{v}_p^{n+1}) J_p^n. \tag{6}$$

For MPM materials where the deformation gradient is stored and updated on the element centroids instead of particles/vertices (e.g., center of a segment in a strand, or centroid of a triangle in a cloth mesh) [Guo et al. 2018; Jiang et al. 2017a], we store an additional J_p on each vertex, and use Eq. (6) to update it.

In the following sections, we show that our new scheme outperforms other state-of-the-art methods, especially for 1) granular materials (e.g. sand [Klár et al. 2016; Yue et al. 2018]) and liquid (we adopt weakly-compressible liquid [Tampubolon et al. 2017]) that should be able to freely expand, 2) cohesive materials with low ductility (e.g. wet clay simulated with NACC [Wolper et al. 2019]) and 3) materials with embedded Lagrangian forces (e.g., clothes, strands [Guo et al. 2018; Jiang et al. 2017a, 2015]) where there is no cohesive force between the manifolds.

4.1 Two dimensional examples

We test SFLIP and compare it with NFLIP, FLIP, and APIC in two dimensions on multiple simulations. Moreover, we further propose a sanity test for measuring the ability to break the positional trap appropriately.

Hourglass. In Fig. 8, integrators with positional adjustment (NFLIP and SFLIP) deliver more energetic dynamics than the other ones. Both NFLIP and SFLIP have many debris particles that are isolated from the bulk (top row of Fig. 8) which continuum methods like MPM can hardly capture with traditional integrators [Yue et al. 2018]. Also, unlike NFLIP, SFLIP does not suffer from the boundary penetration problem.

Fiber piling. In Fig. 10, we simulate piling fibers using the model from Jiang et al. [2017a]. In this example, SFLIP greatly reduces the dissipation when the fibers slide down from the slope. On the other hand, with NFLIP, fibers easily penetrate each other, leading to completely incorrect physics. APIC and FLIP suffer from excessive dissipation, so the fibers separate much more slowly than SFLIP.

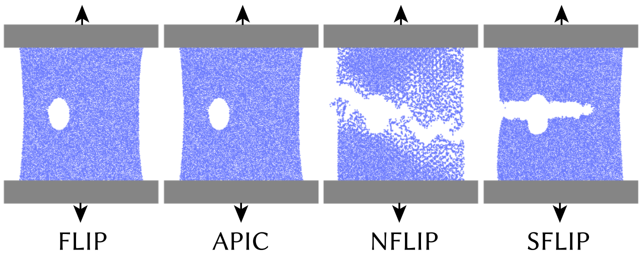

Fig. 12. **Tearing a bulk of wet clay.** A bulk of wet clay with the NACC plastic model is simulated with different integrators. The up and bottom ends of the clay are fixed on two kinematically separating bars. With SFLIP, particles are easier to separate, without the numerical dissipation observed in FLIP and APIC, while severe artifacts appear in the simulation with NFLIP.

Tearing a bulk of wet clay. In Fig. 12, we compare different schemes with a bulk of wet clay using the NACC model, where the material can be cohesive when it has not reached the yield criterion. In this scenario, SFLIP also outperforms other methods, where the bulk is easier to be torn apart without artifacts.

Fracture. In MPM, the Eulerian grid is viewed as a computational scratchpad, while Lagrangian particles act as quadrature points. With MPM, some local region may get fractured due to the lack of quadrature points when such region expands. With SFLIP, particles under stretching would move further along the direction of elongation, and thus the particles separate for a physically valid behavior. As shown in Fig. 11, the rectangle with the NACC plastic model simulated with SFLIP is easier to become fractured than FLIP or APIC. To validate our result, we also simulate it using APIC and FLIP with a 2× refined resolution that makes the separation easier for APIC/FLIP. The fractures are visually similar to the one using SFLIP with the original resolution.

Two rotating squares. To further verify our scheme, we propose a sanity test to measure whether an integration method would separate objects when it is supposed to. As shown in Fig. 13, traditional methods like FLIP and APIC easily fail to separate the two squares, which behave as a single object deforming and rotating around the center of mass during the entire simulation. NFLIP becomes unstable after a few frames due to the unrestrained penetration. On the other hand, SFLIP succeeds in getting the two squares separated as expected.

5 INTEGRATION WITH AFFINE AUGMENTATION

As SFLIP demonstrates its efficacy in several cases featuring the separation of particles, we further investigate the possibility of applying the same idea to APIC. In other words, we expect that combining APIC and the modified position integration may simultaneously preserve affine motion and make particles easier to separate.

5.1 Extension to APIC

The most straightforward way to extend APIC with our new scheme, decribed in §4, is simply to replace the position update in APIC with Eq. (4) and apply Eq. (5) to compute β_p. We denote this first attempt as **A**ffine-augmented **S**eparable **PIC**, or **ASPIC** (in the form

[4] Beginning with the equation governing the deformation gradient [Gonzalez and Stuart 2008], $\frac{D\mathbf{F}_p}{Dt} = (\nabla \mathbf{v}_p^{n+1})\mathbf{F}_p^n$, we have an analytical solution $\mathbf{F}_p^{n+1} = \exp(\Delta t \nabla \mathbf{v}_p^{n+1})\mathbf{F}_p^n$. Computing the determinant for both sides we have $J_p^{n+1} = \det\left[\exp\left(\Delta t \nabla \mathbf{v}_p^{n+1}\right)\right] J_p^n$. By Jacobi's formula [Hall 2015] we have $\det\left[\exp\left(\Delta t \nabla \mathbf{v}_p^{n+1}\right)\right] = \exp\left[\Delta t \mathrm{tr}\left(\nabla \mathbf{v}_p^{n+1}\right)\right] = \exp\left(\Delta t \nabla \cdot \mathbf{v}_p^{n+1}\right)$.

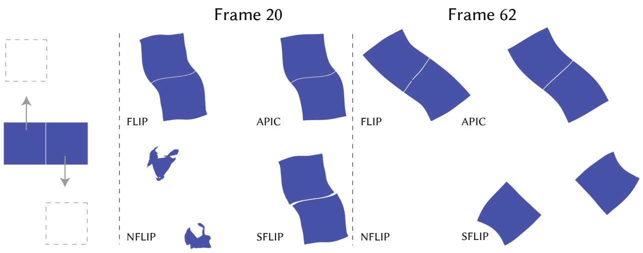

Fig. 13. **Two rotating squares.** Two 2D squares are placed side by side. The left one has an initially upwards velocity, while the right one has an initially downwards velocity. We do not compute stress over the particles' deformation gradient. Instead, we embed a finite element (FE) mesh to each square: the elastic potential energy is computed *only* from the deformation of the FE triangles while Lagrangian forces at mesh vertices are redistributed to the MPM grid nodes (refer to, e.g., §7 of the work by Jiang et al. [2015] for the details). This strategy eliminates the influence of one block's constitutive model on another block. Since there is no stress applied between the two squares, they are supposed to slide frictionlessly and separate in a few time steps (ground truth indicated by dashed squares).

of MLS-MPM [Hu et al. 2018]):

$$\text{ASPIC P2G:}\quad m_i^n \mathbf{v}_i^n = \sum_p w_{ip} m_p \left[\mathbf{v}_p^n + \mathbf{C}_p^n \left(\mathbf{x}_i^n - \mathbf{x}_p^n\right)\right]$$

$$\text{ASPIC G2P:}\quad \mathbf{v}_p^{n+1} = \sum_i w_{ip} \mathbf{v}_i^* \tag{7}$$

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \left[\sum_i w_{ip} \mathbf{v}_i^* + \beta_p \alpha \left(\mathbf{v}_p^n - \sum_i w_{ip} \mathbf{v}_i^n\right)\right] \tag{8}$$

$$\mathbf{C}_p^{n+1} = \sum_i w_{ip} \mathbf{v}_i^* (\mathbf{x}_i^n - \mathbf{x}_p^n)^T (\mathbf{D}_p^n)^{-1}$$

where $\mathbf{D}_p^n = \frac{1}{4}\Delta x^2 \mathbf{I}$ for B-spline quadratic kernel. We test ASPIC with the same sanity check of *two rotating squares*. Unfortunately, ASPIC fails to separate the two squares as illustrated in Fig. 14.

In ASPIC, although the P2G transfer preserves affine momentum better than traditional PIC, the velocity differences in nearby particles are still smoothed at grid nodes. Furthermore, the G2P transfer (Eq. (7)) does not help particles recover their momentum to maintain a discontinuous velocity field. As a result, the positional adjustments are continuous since the particle velocities themselves are continuous. Without discontinuous positional adjustments, the particles would not be able to break the positional trap.

More concretely, assuming zero force applied and the nodal velocity unchanged between P2G and G2P (i.e., $\mathbf{v}_i^* = \mathbf{v}_i^n$), by substituting Eq. (7) into Eq. (8), we have

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \sum_i \left[(1 - \beta_p \alpha) w_{ip}^n \mathbf{v}_i^n + \beta_p \alpha w_{ip}^{n-1} \mathbf{v}_i^{n-1}\right], \tag{9}$$

where $w_{ip}^n = w(\mathbf{x}_i^n, \mathbf{x}_p^n)$ and $w_{ip}^{n-1} = w(\mathbf{x}_i^{n-1}, \mathbf{x}_p^{n-1})$. This equation reveals that the velocity used to advect a particle is simply a weighted mix between the two preceding time steps' continuous nodal velocities. Hence, similar to APIC, ASPIC has no means to break the positional trap.

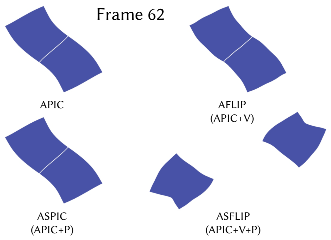

Fig. 14. **Ablation study for two rotating squares.** We investigate whether adding the positional (+P) or the velocity (+V) adjustment to APIC would make the two squares separate from each other. This example shows that the two squares only separate when *both* the positional and velocity adjustments are added.

5.2 Affine-augmented FLIP

The discontinuous velocity modes are the key to break the positional trap. An obvious modification is then to add the velocity adjustment to Eq. (7) so that the particle velocity can be preserved during G2P.

We first discuss whether it makes sense to add the velocity adjustment to APIC (i.e. we ignore the positional adjustment for now), which we denote as Affine-augmented **FLIP**, or **AFLIP** in the following discussion:

$$\text{AFLIP P2G:}\ m_i^n \mathbf{v}_i^n = \sum_p w_{ip} m_p \left[\mathbf{v}_p^n + \mathbf{C}_p^n \left(\mathbf{x}_i^n - \mathbf{x}_p^n\right)\right]$$

$$\text{AFLIP G2P:}\ \mathbf{v}_p^{n+1} = \sum_i w_{ip} \mathbf{v}_i^* + \alpha \left(\mathbf{v}_p^n - \sum_i w_{ip} \mathbf{v}_i^n\right) \tag{10}$$

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \sum_i w_{ip} \mathbf{v}_i^* \tag{11}$$

$$\mathbf{C}_p^{n+1} = \sum_i w_{ip} \mathbf{v}_i^* (\mathbf{x}_i^n - \mathbf{x}_p^n)^T (\mathbf{D}_p^n)^{-1}$$

Notice that AFLIP is equivalent to compensating the affine motion during the P2G step of FLIP.

As discussed in the work of Ding et al. [2020], the velocity adjustment includes movements of almost all frequencies. Therefore, combining the velocity adjustment with the affine momentum tensor brings two benefits:

1. the velocity components of low-frequency (up to affine modes) are guaranteed to be undamped in AFLIP. The FLIP-style velocity update preserves the affine motion during G2P while the affine momentum tensor $\mathbf{C}_p^n$ independently keeps track of a second copy of the velocity gradient. Although it seems to have more information than desired in some instances, this duplication ensures that the affine motion is not diffused. Besides, since this redundant information is low-frequency, it would not lead to instability.
2. visual details can be retained due to the existence of higher-frequency modes.

In the following discussion, we present a practical example to demonstrate these two benefits of AFLIP. A box-shaped liquid is dropped onto a rotating elastic box. Liquid particles get bounced back from the solid and spread out. Due to the high-frequency vibrations between the liquid and solid, detailed structures can be generated as shown in Fig. 15. We use APIC and FLIP with 4×

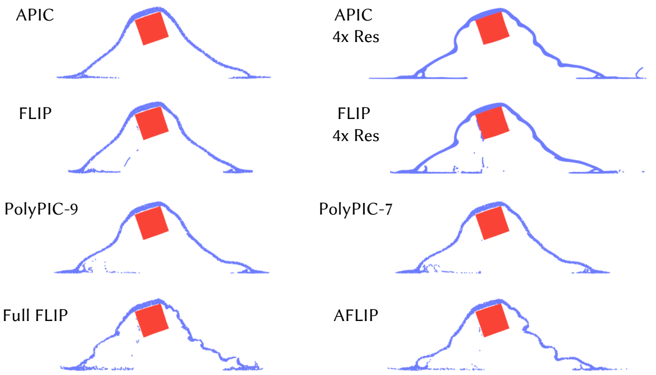

Fig. 15. **Water drops on an elastic square.** Box-shaped weakly-compressible liquid [Tampubolon et al. 2017] is dropped onto a rotating elastic box that is embedded with a FE mesh to prevent possible material damage. AFLIP with $\alpha = 0.99$ reproduces the complex structure that appears in the high-res simulations, and is also immune of the irregular bumpy surfaces in full FLIP. For PolyPIC, we keep the first 4 modes unchanged but scale down higher modes for better stability. For PolyPIC-9, mode 5-6 are scaled by 0.825 and mode 7-9 are scaled by 0.2. For PolyPIC-7, mode 5-7 are scaled with 0.825, but it still explodes after this frame. In either case, PolyPIC fails to reproduce the structure. In addition, the simulation becomes less dissipated for APIC/FLIP when the simulation resolution increases.

resolution as the reference since both can reproduce these intricate structures to some extent.

We compare AFLIP with other integrators, namely APIC, FLIP, and for completeness, PolyPIC [Fu et al. 2017]. Both APIC and FLIP ($\alpha < 1$) with the original resolution cannot reveal the detailed structures. APIC does not preserve any non-affine modes, and the frequency of this vibration is beyond what the affine mode can capture. On the other hand, FLIP with $\alpha = 0.99$ damps out the high-frequency motion in a few frames. A full FLIP integrator ($\alpha = 1$) reproduces the deformation details, though it also suffers from noises that lead to irregular small bumps.

PolyPIC reinterprets the P2G transfer as a weighted minimization problem and introduces a few more non-affine velocity modes. In 2D, there are 9 modes for covering constant, linear, bilinear, and quadratic bases. Nevertheless, increasing modes in PolyPIC (up to all 9 bases) fails to reveal more details; furthermore, we have to scale down the higher-order ones to avoid instability (third row in Fig. 15).

Noise and filtering effect. As far as we are aware, there is no acknowledged definition of noise. In APIC, velocity modes that a grid cannot capture are defined as noise and filtered out during transfers. However, in PolyPIC, some of those filtered modes are extracted and then regarded as helpful information instead of noises. Thus in AFLIP, we prefer to retain the motions with a frequency higher than the affine bases to reproduce the visual details. Notice that such motions should also be damped for stability purposes.

We conclude that both the undamped affine motion (low-frequency) and a dissipated (but existent) non-affine motion (high-frequency) are necessary to produce a visually appealing, energetic simulation. These two conditions exactly correspond to the two benefits of AFLIP. Hence, we believe AFLIP is a reliable alternative to APIC and FLIP for the general simulation purpose.

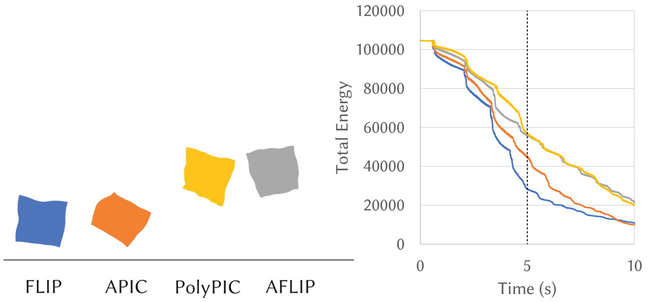

Fig. 16. **A box dropped on the floor.** A finite element (FE) mesh is embedded to provide elasticity. **Left:** A screenshot of the animation. **Right:** Their total energies over time. The dashed line indicates the time of the screenshot on the left.

Furthermore, we demonstrate in Fig. 16 that AFLIP preserves more energy than FLIP and APIC. Its energy preservation capability is comparable with PolyPIC. The latter requires explicitly storing and transferring the high-order terms and brings significant performance overhead [Fu et al. 2017]. AFLIP also allows the user to control the energetic level by merely tuning the value of α. When α reduces to zero, AFLIP turns back to APIC.

5.3 Affine-augmented Separable FLIP

Now we add the positional adjustment into AFLIP for easier particle separation, where we have a new scheme denoted as **A**ffine-augmented **S**eparable **FLIP**, or **ASFLIP**:

$$\text{ASFLIP P2G: } m_i^n \mathbf{v}_i^n = \sum_p w_{ip} m_p \left[\mathbf{v}_p^n + \mathbf{C}_p^n \left(\mathbf{x}_i^n - \mathbf{x}_p^n \right) \right]$$

$$\text{ASFLIP G2P: } \mathbf{v}_p^{n+1} = \sum_i w_{ip} \mathbf{v}_i^* + \alpha \left(\mathbf{v}_p^n - \sum_i w_{ip} \mathbf{v}_i^n \right) \tag{12}$$

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \left[\sum_i w_{ip} \mathbf{v}_i^* + \beta_p \alpha \left(\mathbf{v}_p^n - \sum_i w_{ip} \mathbf{v}_i^n \right) \right] \tag{13}$$

$$\mathbf{C}_p^{n+1} = \sum_i w_{ip} \mathbf{v}_i^* (\mathbf{x}_i^n - \mathbf{x}_p^n)^T (\mathbf{D}_p^n)^{-1}$$

Specifically, AFLIP corresponds to the case where $\beta_p = 0$ constantly. In Fig. 14, ASFLIP outperforms AFLIP by successfully separating two squares. Therefore, we conclude that both the velocity and positional adjustment are necessary to break the positional trap. We further demonstrate the efficacy of ASFLIP with multiple three-dimensional simulations in §6.

6 RESULTS

We have shown a group of didactic 2D examples in the previous sections. We compare the various integrators' effects in larger, more practical, 3D scenarios in the following discussion. A summary of the parameters used for these examples can be found in Table 2.

Similar to the adaptive time-stepping scheme in the work of Fang et al. [2018], we calculate the time step from the sound speed of material and the **C**ourant–**F**riedrichs–**L**ewy (**CFL**) condition.

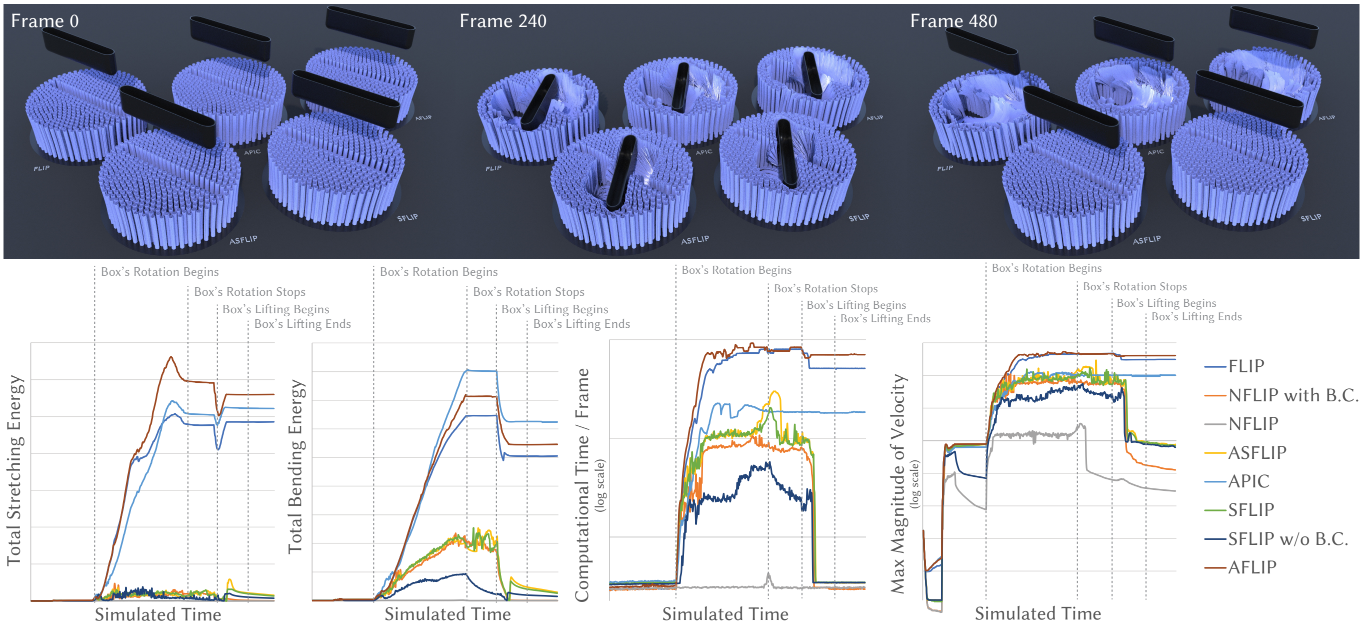

Fig. 17. **Pressed Brush.** A round-box is inserted into a brush (47K strands), rotated, and then pulled out. The averaged gap (≈0.015cm) between the strands in a cluster is much smaller than the averaged distance between strand vertices (0.25cm), to which the cell size is required to be comparable to capture and resolve the contact between strands [Jiang et al. 2017a]. Since the gap between strands is sub-grid scale, the strands simulated with FLIP, APIC, or AFLIP fail to escape the positional trap and stay entangled. On the other hand, the strands simulated with SFLIP or ASFLIP successfully recover to their rest shape due to the correct handling of particle position update. At the bottom, different phases of the box are marked with dashed grey lines. Integrators such as FLIP, APIC, or AFLIP increase the strands' stretching and bending energy dramatically during the box's rotation and do not fall after the box's levitation. When using these simulation methods, the computational costs per frame (1/120s) is much higher because the occasionally released high-speed strand requires a tiny time step. This issue can only be mitigated when using integrators enhanced with our scheme (e.g., SFLIP and ASFLIP). ©2021 Tencent

For the examples with Lagrangian force models, we adopt a splitting scheme similar to Jiang et al. [2017a]. First, we choose an appropriate classic Lagrangian method[5] to compute the momentum change at each vertex or particle per co-dimensional object, which is redistributed to MPM grid nodes. Simo's [1988] neo-Hookean elasticity and Drucker-Prager [1952] plasticity are applied to respond to frictional contacts [Klár et al. 2016; Yue et al. 2018].

We sample 8 ∼ 27 particles in each grid cell for all non-strand and non-cloth examples. We adopt the B-spline quadratic kernel throughout all the examples. We implement MLS-MPM [Hu et al. 2018] following the GPU-based framework by Gao et al. [2018b], and benchmark with an Intel Core i9-9900K CPU and an NVIDIA GeForce RTX 2080Ti GPU. In all experiments, the time consumed by AFLIP per time step is equivalent to that of APIC. On the other hand, our new integration scheme brings 62.2% computational overhead on average (compared with FLIP) for the G2P step and 22.8% overhead for the entire time step. Most overhead comes from the computation of $\phi(\mathbf{x}_p)$ and $\nabla\phi(\mathbf{x}_p)$.

Water Spray from a Fountain Nozzle. In Fig. 1, we compare different integrators simulating weakly-compressible water [Tampubolon et al. 2017] being sprayed out from the nozzle on a fountain. We adopt the source sampling method by Stomakhin and Andrew [2017] to correctly enforce a Dirichlet boundary condition at the nozzle with the following steps:

(1) uniformly sample temporary particles inside a ball at the top of the nozzle with an upwards velocity;
(2) set the grid nodes inside the ball to have the same velocity;
(3) reserve the samples that are outside the ball after advection.

We repeat this process throughout the simulation until a fixed number of particles is reached.

The diameter of the nozzle is about 11cm, and the cell size is 0.25cm. We think it is sufficiently wide that the liquid spout shouldn't be affected by the nozzle boundary. The water simulated with AFLIP and ASFLIP flows significantly faster than the water simulated with other methods. In APIC and FLIP, water particles get damped by particle-grid transfers immediately after they leave the ball. Thus both the compensated affine motion at P2G and the high-frequency preservation at G2P are necessary to have an energetic simulation.

Simulation with AFLIP is less energetic than ASFLIP, mostly when liquid particles leave the top plate. The discrepancy between them is mainly due to the positional adjustment term and the friction handling at the boundary. In our experiment, the latter plays an even more critical role in this particular scenario, even if we apply a tiny frictional coefficient (0.0125) between the liquid and the plates. When a cell overlaps with the boundary (assuming static), nodes inside the boundary would have a damped velocity due to the friction.

[5] We adopt discrete elastic rod for a hair strand [Bergou et al. 2008; Kaldor et al. 2010], finite element method for a volumetric elastic object [Sifakis and Barbic 2012], and discrete shell for a piece of cloth [Bridson et al. 2003; Grinspun et al. 2003]. A Newton-Raphson step [Baraff and Witkin 1998] is used for performance and stability consideration. Our integration scheme and corresponding discussions may also apply to other similar methods (e.g., SuperHelices [Bertails et al. 2006] or mass spring [Selle et al. 2008] for strands).

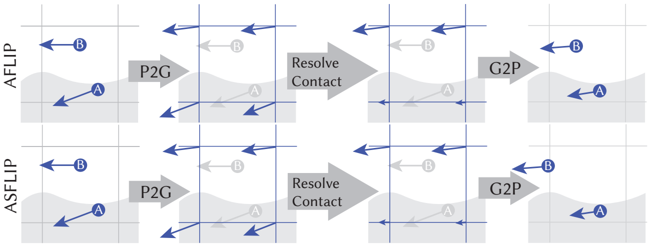

Fig. 18. **AFLIP and ASFLIP at the boundary. Top:** with AFLIP, all particles in the cell overlapping with the boundary (shaded region) would be trapped. **Bottom:** with ASFLIP, only particles that reside in the boundary would be affected by the boundary. As a result, particle B advected by ASFLIP would move faster than if it were advected by AFLIP.

With AFLIP, all particles in that cell would have smaller displacements. On the other hand, with ASFLIP, if particles in that cell have not entered the boundary, they would be affected by neither the boundary collision nor friction (Fig. 18). In other words, the contact handling between the liquid and boundary is more accurate with our new scheme applied.

Pressed Brush. In Fig. 17, we press a round-corner box into a brush (47K strands), rotate, and then pull it out. The cell size is chosen to be comparable to the segment length of a strand, similar to Jiang et al. [2017a]. The gap between strands inside the same cluster is sub-grid scale and is way smaller than the segment length, and it is impractical (w.r.t. memory and computational cost) to sample a single strand (and pick an equivalent cell size) with the gap size. Due to this difficulty (which is common when simulating strands with MPM), APIC, FLIP, or AFLIP fail to recover the brush to the rest shape, and the strands stay entangled. On the other hand, with our correction of particle positions, ASFLIP or SFLIP do not suffer from this visual artifact.

We also present an analysis of the stretching and bending energy stored in the strands. With FLIP, APIC, or AFLIP, the strands keep accumulating these potential energies when entangled. Some of the strands may suddenly get released from the positional trap and get an abnormally high velocity (over 400m/s) due to a large amount of collected potential energy. As a result, a tiny time step needs to be enforced for a stable simulation. Although our scheme brings computational overhead for a single time step, it allows a much larger time step for explicit MPM integration and, hence, tremendously reduces the computational cost over a frame.

Furthermore, we perform an ablation study with this scenario. In Fig. 19, we tweak β_p in Eq. (4) to compare four different cases: 1) NFLIP, 2) NFLIP with only the boundary condition considered, i.e.,

$$\beta_p = \begin{cases} 0 & \text{if } \phi_{\mathbf{x}_p^n+\mathbf{v}_p^n\Delta t} < 0 \text{ and } \nabla\phi_{\mathbf{x}_p^n+\mathbf{v}_p^n\Delta t}\cdot\mathbf{v}_p^n \le 0 \\ \beta^{\max} & \text{otherwise} \end{cases}$$

3) SFLIP without the boundary condition, i.e.,

$$\beta_p = \begin{cases} \beta^{\min} & \text{if } J_p^{n+1} < 1 \\ \beta^{\max} & \text{otherwise} \end{cases}$$

and 4) SFLIP. Extreme penetration can be observed when our boundary condition handling is not applied. On the other hand, without considering J_p^{n+1} during advection, clusters of strands lose their volume when pressed and do not propagate the deformation to nearby strands through collision, leading to the weaker strand-strand collision. This experiment indicates that both J_p^{n+1} and the boundary condition are necessary during integration for correctly resolving both the strand-strand collision and strand-box collision.

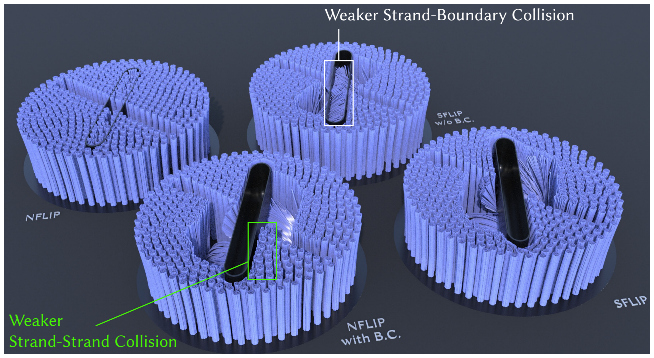
Weaker Strand-Boundary Collision
SFLIP w/o B.C.
NFLIP
SFLIP
Weaker Strand-Strand Collision
NFLIP with B.C.

Fig. 19. **Ablation Study with Pressed Brush.** We compare NFLIP, SFLIP, and the intermediate integrators where only the boundary condition (NFLIP with B.C.) or the compression state (SFLIP w/o B.C.) is considered during integration. NFLIP cannot handle any collision correctly. NFLIP with B.C. can accurately address the boundary conditions but has weaker strand-strand collisions. As a result, the strands deformed by the boundary collision penetrate and overlap their nearby strands (marked with a green box). SFLIP w/o B.C. handles strand-strand collisions, but the strands penetrate the boundary (marked with a white box). SFLIP correctly handles both strand-boundary and strand-strand collisions, where the deformed strands collided with the boundary press other strands nearby, deforming them as well. ©2021 Tencent

Wheel Rolls Up Sand Particles. A rotating wheel slowly moves down into a container full of dry sand particles, rolls up them, and then moves out. Similar to Yue et al. [2018], we apply Simo's [1988] neo-Hookean elastic model and Drucker-Prager [1952] plastic model. Comparing the seven different integrators, we first notice that FLIP behaves much better than APIC in this scenario. While NFLIP spreads more particles to the left, SFLIP has more particles fly upwards since the particles at the container boundary are correctly handled to acquire enough momentum to move towards the sky. Furthermore, without the compensated affine motion, FLIP, NFLIP, and SFLIP, all have particles clumped as if there were cohesive forces to group nearby particles together. On the other hand, with AFLIP and ASFLIP, particles are separated, giving correct dry sand behavior; and ASFLIP drives many more particles to fly up than all other methods as shown in Fig. 21.

Sand Breaks through Elastic Plate. In this example, we drop a column-shaped sand pile onto a thin elastic plate. To introduce fractures that artists can fully control (e.g., where to fracture and how the fractures look like), we adopt a pre-fractured mesh [Molino et al. 2004] by which we mean that the plate is composed of several disconnected meshes.

When the sand particles collide onto the plate, the meshes simulated with APIC and FLIP would be able to resist the impulse from the sand to some extent due to the numerical viscosity. As a result, some sand particles reside on the plate until the end of the simulation, as in Fig. 22. On the other hand, when the integrator is either SFLIP or ASFLIP, meshes in the middle of the plate are stricken

Table 2. **Parameters of Simulations.** N_p denotes the number of particles. The time step size is calculated [Fang et al. 2018] from the mechanical parameters (e.g., bulk modulus κ, poisson ratio ν, and mass density ρ) and the compression state of the material. We adopt centimeter-gram-second units. The lengths are in centimeters, with the mass density in g/cm^3 and the modulus in dyne/cm^2. For sand, we adopt a friction angle of 30°; for hair strands or clothes, we adopt a friction angle of 20°; for the NACC material in Fig. 12, we adopt $\chi = 2$, $\xi = 1$, $\alpha_p = \ln(0.8)$ initially, and a friction angle of 45°; for the NACC material in Fig. 11, we adopt $\chi = 0.999$, $\xi = 1$, $\alpha_p = \ln(0.89)$, and a friction angle of 90°.

Scenario	Δx	α	β_min	β_max	N_p	κ	ν	ρ
Fig. 1	0.25	0.97	0.0	1.0	3.0M (max)	1.0e7	n/a	1.0
Fig. 5	0.2	0.99	n/a	n/a	2.1K	4.08e6	0.3	0.96
Fig. 8	2.0	0.99	0.0	1.0	9.4K	8.33e5	0.3	2.0
Fig. 10	1.0	0.99	0.0	0.5	1.2K	3.00e7 (stretching) 6.00e6 (collision)	0.42	2.0
Fig. 12	0.833	0.99	0.0	0.2	13.8K	1.09e7	0.4	2.0
Fig. 11	2.0	0.99	0.0	1.0	1.7K	1.83e6	0.3	2.0
Fig. 13	6.0	0.99	0.0	0.1	2.3K	8.33e5	0.3	2.0
Fig. 15	2.0	0.99	0.0	0.25	5.0K	3.00e6 (liquid) 8.33e5 (block)	n/a (liquid) 0.3 (block)	1.0 (liquid) 0.5 (block)
Fig. 17	0.125	0.97	0.05	0.2	991.7K	4.52e10 (in-strand) 8.33e3 (collision)	0.39 (in-strand) 0.3 (collision)	1.3
Fig. 21	1.0	0.97	0.0	1.0	1.755M	8.33e6	0.3	1.4
Fig. 22	0.8	0.97	0.0	1.0	707.4K	4.9e7 (board) 4.9e6 (sand)	0.3 (board) 0.4 (sand)	1.1 (board) 2.0 (sand)
Fig. 23	1.0	0.95	0.05	0.2	39.0K	1.02e7	0.202	1.32
Fig. 24	0.375	0.97	0.05	0.2	1.818M	1.28e10 (bangs) 2.56e10 (back hairs) 8.33e3 (collision)	0.37 (in-strand) 0.3 (collision)	1.3

down immediately, and the plate wouldn't have sand particles stick around the fractured hole. Furthermore, ASFLIP produces much more energetic deformations than SFLIP, as expected.

Separating Two Pieces of Sheets. The feature of easier particle separation improves the simulation quality, even for a simple scenario. In Fig. 23, two pieces of clothes are initially packed close together and then pulled apart. Due to the smoothing effect of the particle-grid transfers in FLIP and APIC, the separation of the two pieces has a noticeable latency until the pulling forces become large.

We observe another artifact associated with FLIP and APIC. The surfaces of the two pieces are full of unnatural wrinkles after getting pulled apart. The wrinkles' deformations within each piece of cloth produce Lagrangian forces that would try to smooth out the wrinkles. However, the generated velocity modes are of sub-grid resolution (the same issue is also reported in [Guo et al. 2018]) and cannot be captured by the P2G process. In other words, the momentum change from Lagrangian forces is in the null space of the particle-to-grid transfer and thus is invisible to the grid.

Our position correction can eliminate both issues: in the simulation with SFLIP and ASFLIP, these two sheets easily separate and have smooth surfaces.

Cassandra Chen. We compare different integrators on a short-haired character in a video game production asset (Fig. 24). The hair strands simulated with FLIP or APIC can hardly separate due to the loss of sub-grid motion. On the other hand, the hairs simulated with SFLIP or ASFLIP break the positional trap, separate as usual, and have a flowing visual look.

7 LIMITATION AND FUTURE WORK

Numerical volume gain. As discussed in existing works [Gao et al. 2018a; Yue et al. 2018], in continuum methods like MPM, when individual particles get separated from the bulk material, there will be substantial resistance to prevent them from merging back, leading to numerical volume gain. As our new integrators SFLIP and ASFLIP make particles interact more energetically and separate more easily, the artifact of volume gain can be exacerbated. Fig. 20 shows an example where ASFLIP introduces more severe volume gain than APIC. One possible solution is to convert the state of particles between the continuum bulk and discrete debris [Gao et al. 2018a] by tracking approximated density functions for particles. Nevertheless, completely resolving this issue is still an open problem.

Incompressible fluid. Our computation of β_p relies on the value of J_p^{n+1}. However, for hybrid methods simulating incompressible fluid [Bridson 2015], J_p is always supposed to be one. Even if there could be some deviation of J_p from one due to numerical errors, our scheme is not observed to improve these methods.

Beyond current scheme. Our scheme mostly considers materials with low ductility, where we adopt a simple separation criterion. However, for materials (e.g. metal or pure elastic body) that can be continuously elongated before fracturing, more sophisticated measurements for separation should be provided.

For MPM with Lagrangian forces, in the directions orthogonal to the co-dimensional objects, particles are assumed to freely separate when the material yields. Nevertheless, the extra J_p^{n+1} stored at each particle tracks the volume change for all directions. While our scheme of computing the positional adjustment alleviates numerical viscosity as demonstrated in Fig. 17, Fig. 23 and Fig. 24, a more accurate scheme should only track the volume change in the directions perpendicular to the manifold.

For the same reason, our separation criterion depending on J_p only considers fracture due to expansional pressure. A better criterion that also considers other fracture modes (e.g., fractures due to uniaxial tensile, shearing, or compressive stress), or a separation criterion derived from standard fracture criteria [Broberg 1999], is reserved for future work. Developing a principled theory connecting the fracture mechanics and the hybrid integration scheme would also be a promising research direction.

More accurate contact resolution. Similar to previous work [Guo et al. 2018; Jiang et al. 2017a], our method requires appropriate grid resolution for plausible visual results in scenarios with many frictional contacts. Our method cannot guarantee there would be no penetration if the time step is large. Furthermore, our method requires careful and manual selection of β^max to prevent separating particles from penetrating regions in their vicinity, especially when the time step is large (Fig. 25). We look forward to investigating methods that use continuous collision detection and barrier functions [Li et al. 2020a,b] for resolving this problem.

When particles collide with solids, their displacements depend only on nodal velocities. Furthermore, we adopt a simple contact handling mechanism which may not perfectly resolve frictions. More accurate schemes are worth further exploration.

Coupling with discrete physics. The separation of particles in SFLIP/ASFLIP depends on the high-frequency modes that are not resolvable on the grid. Nevertheless, there is no physics applied at sub-grid scale to ensure these high-frequency modes' mechanical

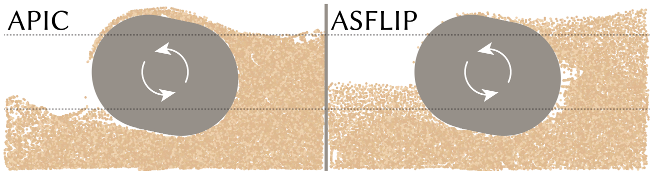

Fig. 20. **Numerical Volume Gain.** A high-speed rotating rock stirred the sand. After the rock stopped spinning, severe volume gain was observed in the simulation by ASFLIP. Two dashed lines are used to assist in comparing the height of the sand pile.

correctness and accuracy. Coupling MPM with discrete physics (e.g., DEM) during advection is a promising research direction.

Comparison with ground truth. In Fig. 3, we compare results from different integrators with the analytic solution. However, for more complicated setups, it would be much more challenging to acquire the ground truth. We reserve a complete comparison as future work.

8 CONCLUSION

We observe that the PIC method filters out information by two rounds of averaging: particle quantities are first averaged to grid nodes during P2G, and then the updated quantities on grid nodes are averaged to particles during G2P. As a result, meaningful information is eliminated during repeated transfers. We propose a modification to the advection step to reduce numerical dissipation.

We target materials with low ductility. When material yields, the underlying particles are supposed to separate from each other without intervention. Based on this premise, we propose using the infinitesimal volume change to determine whether a positional adjustment should be introduced to help break the spatial trap due to interpolation. We apply this idea to both FLIP and APIC and compare our methods with traditional integrators in several practical simulations on a massive scale.

Lastly, we propose AFLIP as a more general-purposed method that can better preserve energy, take almost zero extra computational cost compared to APIC, and retain stability similar to FLIP.

ACKNOWLEDGMENTS

We thank Chendi (Wayne) Wei, Bo (Brandon) Yang, Bing (Binbin) Tang, Jun (Jim) Yin, Dafu (Hyton) Deng, Xiaochun (Willim) Cui and other colleagues in Common R&D Operation System (CROS) of Tencent IEG for their full support, Hua (Toby) Zhang, Jingxiang Li, Cheng Ge and other colleagues in NExT Studios for the modeling and shading of Cassandra Chen, Jiarun (Kelvin) Cai for discussions on Unreal plugin development, Katherian Xia and Sean Hu for proofreading, and Yichen (Peter) Chen for sharing the example setup in Fig. 5. In addition, we would like to thank the anonymous reviewers for their insightful comments.

REFERENCES

Ryoichi Ando, Nils Thurey, and Reiji Tsuruno. 2012. Preserving fluid sheets with adaptively sampled anisotropic particles. *IEEE Transactions on Visualization and Computer Graphics* 18, 8 (2012), 1202–1214.

David Baraff and Andrew Witkin. 1998. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. 43–54.

Christopher Batty, Florence Bertails, and Robert Bridson. 2007. A fast variational framework for accurate solid-fluid coupling. *ACM Trans. on Graph. (TOG)* (2007).

Miklós Bergou, Max Wardetzky, Stephen Robinson, Basile Audoly, and Eitan Grinspun. 2008. Discrete Elastic Rods. *ACM Trans. on Graph. (TOG)* (aug 2008).

Florence Bertails, Basile Audoly, Marie-Paule Cani, Bernard Querleux, Frédéric Leroy, and Jean-Luc Lévêque. 2006. Super-helices for predicting the dynamics of natural hair. *ACM Trans. on Graph. (TOG)* 25, 3 (2006), 1180–1187.

Javier Bonet and Richard D Wood. 1997. *Nonlinear continuum mechanics for finite element analysis*. Cambridge university press.

Landon Boyd and Robert Bridson. 2012. MultiFLIP for energetic two-phase fluid simulation. *ACM Trans. on Graph. (TOG)* 31, 2 (2012), 1–12.

Jeremiah U Brackbill and Hans M Ruppel. 1986. FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational physics* 65, 2 (1986), 314–343.

Percy Williams Bridgman. 1949. *The physics of high pressure*. London: Bells and Sons.

Robert Bridson. 2015. *Fluid simulation for computer graphics*. CRC press.

R Bridson, S Marino, and R Fedkiw. 2003. Simulation of clothing with folds and wrinkles. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. 28–36.

K Bertram Broberg. 1999. *Cracks and fracture*. Elsevier.

Xiao-Song Chen, Chen-Feng Li, Geng-Chen Cao, Yun-Tao Jiang, and Shi-Min Hu. 2020. A moving least square reproducing kernel particle method for unified multiphase continuum simulation. *ACM Trans. on Graph. (TOG)* 39, 6 (2020), 1–15.

Jens Cornelis, Markus Ihmsen, Andreas Peer, and Matthias Teschner. 2014. IISPH-FLIP for incompressible fluids. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 255–262.

Ounan Ding, Tamar Shinar, and Craig Schroeder. 2020. Affine particle in cell method for MAC grids and fluid simulation. *J. Comput. Phys.* 408 (2020), 109311.

Daniel Charles Drucker and William Prager. 1952. Soil mechanics and plastic analysis or limit design. *Quarterly of applied mathematics* 10, 2 (1952), 157–165.

Essex Edwards and Robert Bridson. 2012. A high-order accurate particle-in-cell method. *Internat. J. Numer. Methods Engrg.* 90, 9 (2012), 1073–1088.

Yu Fang, Yuanming Hu, Shi-Min Hu, and Chenfanfu Jiang. 2018. A temporally adaptive material point method with regional time stepping. In *Computer graphics forum*, Vol. 37. Wiley Online Library, 195–204.

Florian Ferstl, Ryoichi Ando, Chris Wojtan, Rüdiger Westermann, and Nils Thuerey. 2016. Narrow band FLIP for liquid simulations. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 225–232.

Chuyuan Fu, Qi Guo, Theodore Gast, Chenfanfu Jiang, and Joseph Teran. 2017. A polynomial particle-in-cell method. *ACM Trans. on Graph. (TOG)* 36, 6 (2017), 1–12.

Ming Gao, Andre Pradhana, Xuchen Han, Qi Guo, Grant Kot, Eftychios Sifakis, and Chenfanfu Jiang. 2018a. Animating fluid sediment mixture in particle-laden flows. *ACM Trans. on Graph. (TOG)* 37, 4 (2018), 1–11.

Ming Gao, Andre Pradhana Tampubolon, Chenfanfu Jiang, and Eftychios Sifakis. 2017. An adaptive generalized interpolation material point method for simulating elastoplastic materials. *ACM Trans. on Graph. (TOG)* 36, 6 (2017), 1–12.

Ming Gao, Xinlei Wang, Kui Wu, Andre Pradhana, Eftychios Sifakis, Cem Yuksel, and Chenfanfu Jiang. 2018b. GPU optimization of material point methods. *ACM Trans. on Graph. (TOG)* 37, 6 (2018), 1–12.

Dan Gerszewski and Adam W Bargteil. 2013. Physics-based animation of large-scale splashing liquids. *ACM Trans. Graph.* 32, 6 (2013), 185–1.

Oscar Gonzalez and Andrew M Stuart. 2008. *A first course in continuum mechanics*. Cambridge University Press.

Eitan Grinspun, Anil N Hirani, Mathieu Desbrun, and Peter Schröder. 2003. Discrete shells. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Citeseer, 62–67.

Qi Guo, Xuchen Han, Chuyuan Fu, Theodore Gast, Rasmus Tamstorf, and Joseph Teran. 2018. A material point method for thin shells with frictional contact. *ACM Trans. on Graph. (TOG)* 37, 4 (2018), 1–15.

Brian Hall. 2015. *Lie groups, Lie algebras, and representations: an elementary introduction*. Vol. 222. Springer.

Chad C Hammerquist and John A Nairn. 2017. A new method for material point method particle updates that reduces noise and enhances stability. *Computer methods in applied mechanics and engineering* 90 (2017), 1073–1088.

X. Han, T. F Gast, Q. Guo, S. Wang, C. Jiang, and J. Teran. 2019. A hybrid material point method for frictional contact with diverse materials. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* (2019).

Francis Harvey Harlow, Martha Evans, and Robert D Richtmyer. 1955. *A machine calculation method for hydrodynamic problems*. Los Alamos Scientific Laboratory of the University of California.

Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. 2018. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Trans. on Graph. (TOG)* (2018).

Yuanming Hu, Xinxin Zhang, Ming Gao, and Chenfanfu Jiang. 2019. On hybrid lagrangian-eulerian simulation methods: practical notes and high-performance aspects. In *ACM SIGGRAPH 2019 Courses*. 1–246.

Markus Ihmsen, Jens Cornelis, Barbara Solenthaler, Christopher Horvath, and Matthias Teschner. 2013. Implicit incompressible SPH. *IEEE transactions on visualization and*

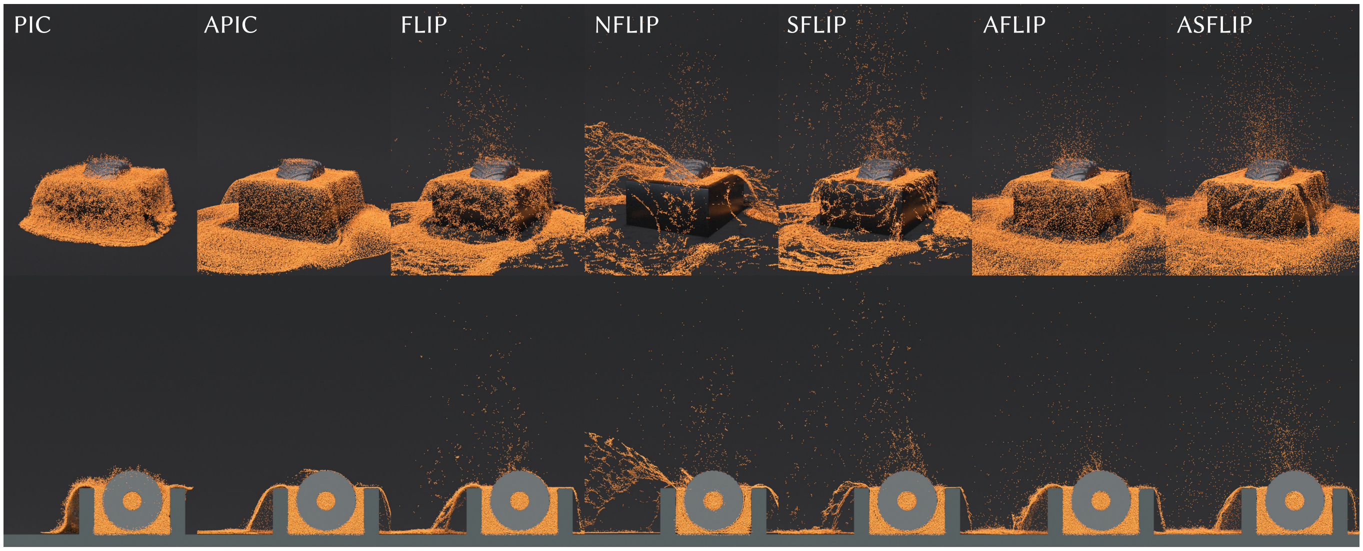

Fig. 21. **Wheel Rolls Up Sand Particles. Top:** Perspective view. **Bottom:** Cutaway view. Simulations with PIC or APIC are significantly more viscous than the others. Without correctly resolving the boundary conditions, NFLIP drives particles to the left side instead of the upper side. With the affine velocity modes, ASFLIP and AFLIP manage to distribute particles without clumping artifact uniformly. Moreover, ASFLIP sprays more particles into the air. ©2021 Tencent

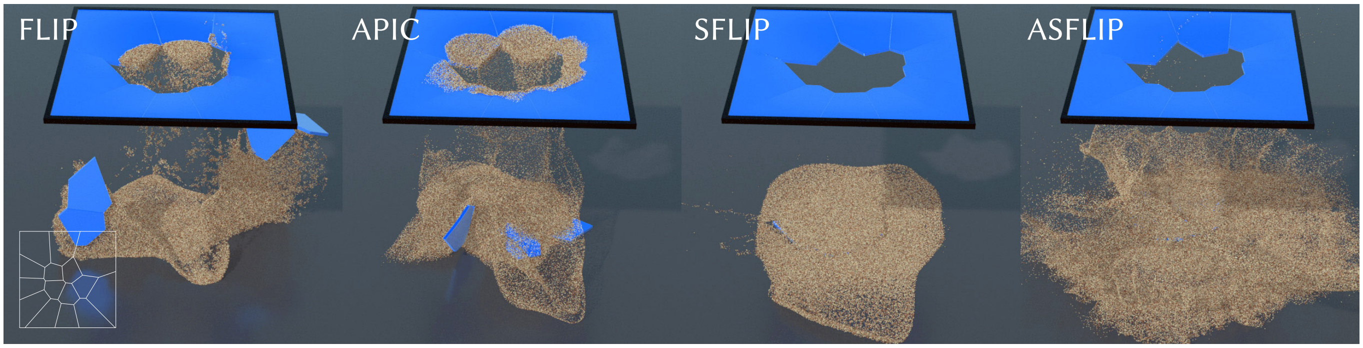

Fig. 22. **Sand Breaks through Elastic Plate.** Sand particles fall onto a non-frictional elastic plate consisting of a few pre-fractured meshes [Molino et al. 2004] (whose pattern is shown in the bottom-left corner). In the simulations with FLIP and APIC, the disconnected meshes exhibit strong viscosity, and the sand particles stick to the plates due to excess dissipation. Instead, with SFLIP or ASFLIP, the sand particles quickly slide. ©2021 Tencent

computer graphics 20, 3 (2013), 426–435.

Chenfanfu Jiang, Theodore Gast, and Joseph Teran. 2017a. Anisotropic elastoplasticity for cloth, knit and hair frictional contact. *ACM Trans. on Graph. (TOG)* (2017).

Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. 2015. The affine particle-in-cell method. *ACM Trans. on Graph. (TOG)* (2015).

Chenfanfu Jiang, Craig Schroeder, and Joseph Teran. 2017b. An angular momentum conserving affine-particle-in-cell method. *J. Comput. Phys.* 338 (2017), 137–164.

Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. 2016. The material point method for simulating continuum materials. In *ACM SIGGRAPH 2016 Courses*. 1–52.

Jonathan M. Kaldor, Doug L. James, and Steve Marschner. 2010. Efficient Yarn-based Cloth with Adaptive Contact Linearization. *ACM Trans. on Graph. (TOG)* 29, 4, Article 105 (July 2010), 10 pages.

Gergely Klár, Theodore Gast, Andre Pradhana, Chuyuan Fu, Craig Schroeder, Chenfanfu Jiang, and Joseph Teran. 2016. Drucker-prager elastoplasticity for sand animation. *ACM Trans. on Graph. (TOG)* 35, 4 (2016), 1–12.

Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M Kaufman. 2020a. Incremental Potential Contact: Intersection-and Inversion-free, Large-Deformation Dynamics. *ACM Trans. on Graph. (TOG)* 39, 4 (2020).

Minchen Li, Danny M Kaufman, and Chenfanfu Jiang. 2020b. Codimensional Incremental Potential Contact. *arXiv preprint arXiv:2012.04457* (2020).

A. McAdams, A. Selle, K. Ward, E. Sifakis, and J. Teran. 2009. Detail preserving continuum simulation of straight hair. *ACM Trans. on Graph. (TOG)* (2009).

Neil Molino, Zhaosheng Bao, and Ron Fedkiw. 2004. A virtual node algorithm for changing mesh topology during simulation. *ACM Trans. on Graph. (TOG)* (2004).

FD Murnaghan. 1944. The compressibility of media under extreme pressures. *Proceedings of the national academy of sciences of the United States of America* (1944).

James F O'Brien and Jessica K Hodgins. 1999. Graphical modeling and animation of brittle fracture. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. 137–146.

Takahiro Sato, Christopher Wojtan, Nils Thuerey, Takeo Igarashi, and Ryoichi Ando. 2018. Extended narrow band FLIP for liquid simulations. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 169–177.

Andrew Selle, Michael Lentine, and Ronald Fedkiw. 2008. A mass spring model for hair simulation. In *ACM SIGGRAPH 2008 papers*. 1–11.

Eftychios Sifakis and Jernej Barbic. 2012. FEM simulation of 3D deformable solids: a practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 courses*. 1–50.

Juan C Simo. 1988. A framework for finite strain elastoplasticity based on maximum plastic dissipation and the multiplicative decomposition: Part I. Continuum formulation. *Computer methods in applied mechanics and engineering* (1988).

A. Stomakhin, C. Schroeder, L. Chai, J. Teran, and A. Selle. 2013. A material point method for snow simulation. *ACM Trans. on Graph. (TOG)* (2013).

Alexey Stomakhin, Craig Schroeder, Chenfanfu Jiang, Lawrence Chai, Joseph Teran, and Andrew Selle. 2014. Augmented MPM for phase-change and varied materials. *ACM Trans. on Graph. (TOG)* 33, 4 (2014), 1–11.

Alexey Stomakhin and Andrew Selle. 2017. Fluxed animated boundary method. *ACM Trans. on Graph. (TOG)* 36, 4 (2017), 1–8.

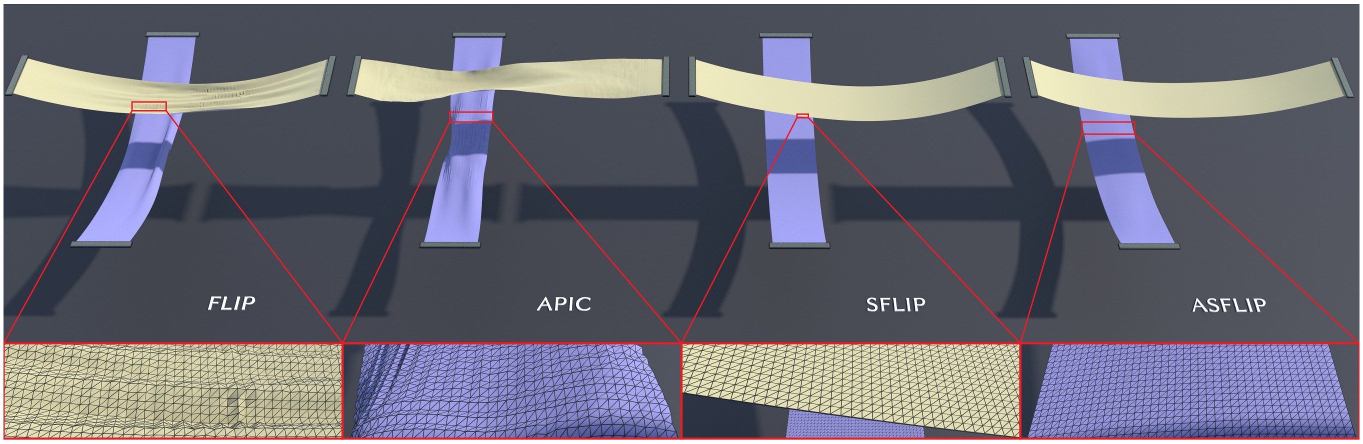

Fig. 23. **Separating Two Pieces of Sheets.** We pull apart two pieces of clothes stacked initially with a gap of less than half of the cell size. Simulations with FLIP and APIC manifest delayed separation due to numerical viscosity and produce resolution-dependent wrinkles reported by Guo et al. [2018]. With our correction, neither of the issues exists anymore. ©2021 Tencent

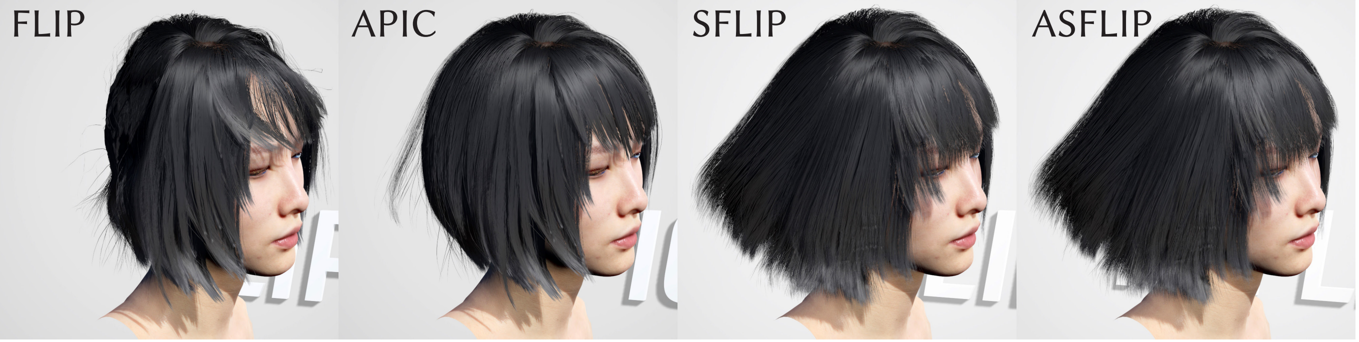

Fig. 24. **Cassandra Chen.** Since the gaps between hairs (93K strands) are much thinner than the cell size, the sub-grid motion has been mostly dissipated by FLIP and APIC. As a result, the hairs stay entangled and look rigid. On the other hand, with SFLIP or ASFLIP, the hairs have a flowing visual look. The head and hairs are modeled and shaded courtesy of NExT Studios. ©2021 Tencent

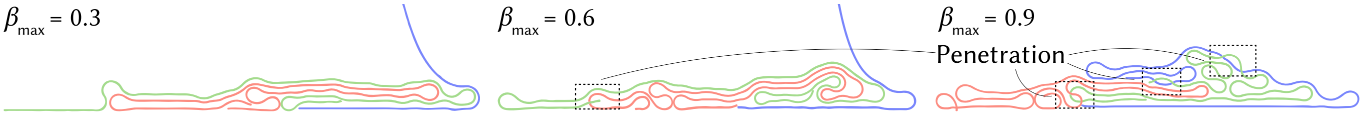

Fig. 25. **Penetration due to large $\beta_{\max}$ and Δt.** We adopt a 3× Δt for the same scenario in Fig. 10 simulated with SFLIP, and compare between using different $\beta_{\max}$. When Δt and $\beta_{\max}$ are both large, severe penetrations may appear (marked with dashed boxes).

Deborah Sulsky, Zhen Chen, and Howard L Schreyer. 1994. A particle method for history-dependent materials. *Computer methods in applied mechanics and engineering* 118, 1-2 (1994), 179–196.

Deborah Sulsky, Shi-Jian Zhou, and Howard L Schreyer. 1995. Application of a particle-in-cell method to solid mechanics. *Computer physics communications* (1995).

Andre Pradhana Tampubolon, Theodore Gast, Gergely Klár, Chuyuan Fu, Joseph Teran, Chenfanfu Jiang, and Ken Museth. 2017. Multi-species simulation of porous sand and water mixtures. *ACM Trans. on Graph. (TOG)* 36, 4 (2017), 1–11.

Kiwon Um, Seungho Baek, and JungHyun Han. 2014. Advanced hybrid particle-grid method with sub-grid particle correction. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 209–218.

Joshuah Wolper, Yu Fang, Minchen Li, Jiecong Lu, Ming Gao, and Chenfanfu Jiang. 2019. CD-MPM: Continuum damage material point methods for dynamic fracture animation. *ACM Trans. on Graph. (TOG)* 38, 4 (2019), 1–15.

Tao Yang, Jian Chang, Ming C Lin, Ralph R Martin, Jian J Zhang, and Shi-Min Hu. 2017. A unified particle system framework for multi-phase, multi-material visual simulations. *ACM Trans. on Graph. (TOG)* 36, 6 (2017), 1–13.

Yonghao Yue, Breannan Smith, Peter Yichen Chen, Maytee Chantharayukhonthorn, Ken Kamrin, and Eitan Grinspun. 2018. Hybrid grains: adaptive coupling of discrete and continuum simulations of granular media. *ACM Trans. on Graph. (TOG)* (2018).

Yongning Zhu and Robert Bridson. 2005. Animating sand as a fluid. *ACM Trans. on Graph. (TOG)* 24, 3 (2005), 965–972.