The Affine Particle-In-Cell Method

Chenfanfu Jiang† Craig Schroeder† Andrew Selle⋆ Joseph Teran⋆† Alexey Stomakhin⋆

†University of California Los Angeles ⋆Walt Disney Animation Studios

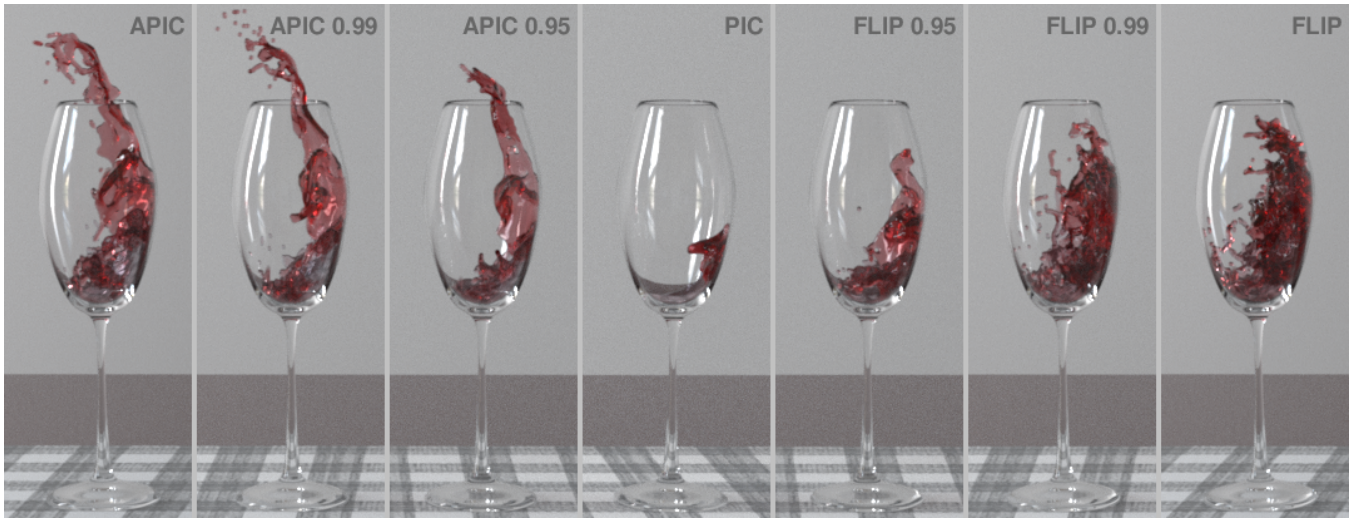

Figure 1: *APIC/PIC blends yield more energetic and more stable behavior than FLIP/PIC blends in a wine pour example. APIC/PIC blends are achieved analogously to FLIP/PIC in that it is a scaling of the particle affine matrices.* ©*Disney.*

Abstract

Hybrid Lagrangian/Eulerian simulation is commonplace in computer graphics for fluids and other materials undergoing large deformation. In these methods, particles are used to resolve transport and topological change, while a background Eulerian grid is used for computing mechanical forces and collision responses. Particle-in-Cell (PIC) techniques, particularly the Fluid Implicit Particle (FLIP) variants have become the norm in computer graphics calculations. While these approaches have proven very powerful, they do suffer from some well known limitations. The original PIC is stable, but highly dissipative, while FLIP, designed to remove this dissipation, is more noisy and at times, unstable. We present a novel technique designed to retain the stability of the original PIC, without suffering from the noise and instability of FLIP. Our primary observation is that the dissipation in the original PIC results from a loss of information when transferring between grid and particle representations. We prevent this loss of information by augmenting each particle with a locally affine, rather than locally constant, description of the velocity. We show that this not only stably removes the dissipation of PIC, but that it also allows for exact conservation of angular momentum across the transfers between particles and grid.

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

Keywords: PIC, FLIP, MPM, fluids, physically-based modeling, coupling

1 Introduction

Simulating natural phenomena for virtual worlds and characters is an important application that remains extremely challenging. An artist's need to manipulate and comprehend physical simulations imposes a significant constraint, all but requiring simulation methods to involve Lagrangian particles. In addition, the need for computational efficiency, topology change and numerical stability has led engineers toward hybrid Lagrangian/Eulerian methods. This is the cause of the ubiquity of incompressible FLIP for simulating liquids in visual effects [Zhu and Bridson 2005]. While such hybridizations solve some problems they also create numerous difficulties. Specifically, while the hybridization allows numerical algorithms to be done in the most appropriate representation, transferring between representations creates error. In this work we show how that error can be minimized with minimal effort.

While our approach will apply to a wide range of continuum phenomena, for simplicity, first consider fluid simulation. Here, pressure and viscosity updates are best done on an Eulerian grid while advection is best done with Lagrangian particles. The first and simplest method of this type is Particle-In-Cell (PIC) [Harlow 1964; Harlow and Welch 1965]. While this method is remarkably effective and simple to implement, it suffers from significant dis-

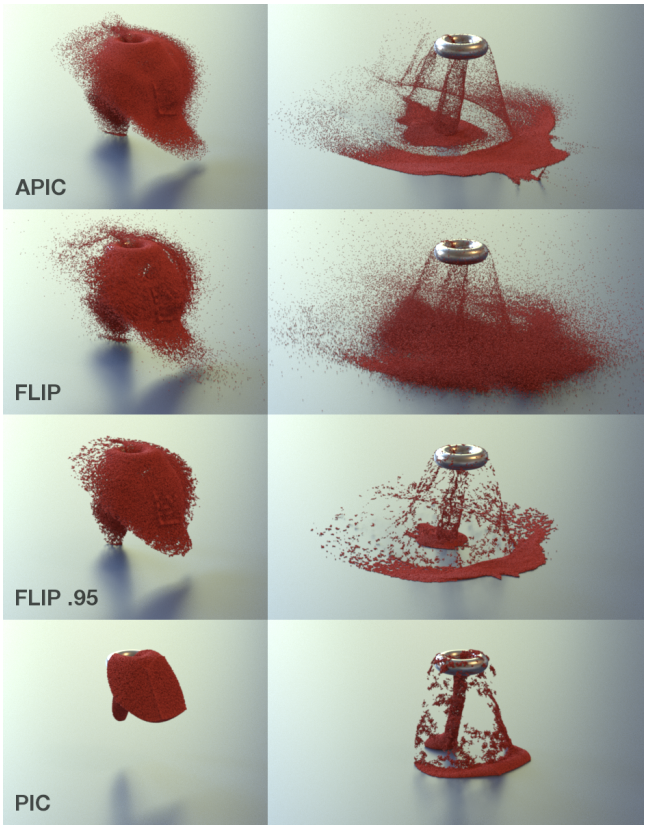

Figure 2: *We compare against FLIP and APIC with a granular material example. Here, a red cube of sand is dropped onto a torus causing it to exhibit interesting flow patterns. APIC appears more like a fine powder—while FLIP suffers from excessive noise and instability. The dissipative nature of PIC causes the sand to clump together, giving it a wet look that also plagues the FLIP/PIC blend.* ©*Disney.*

sipation (viscous appearance) due to frequent particle/grid transfers. Dissipation is addressed in the Fluid-Implicit-Particle (FLIP) method [Brackbill and Ruppel 1986; Brackbill et al. 1988]. The main idea is to transfer increments of velocities and displacements from grid to particles, rather than directly interpolating from the grid. Intuitively, if there is only a small offset, only a small correction will be made, typically reducing the dissipation. Unfortunately, there are other errors besides dissipation inherent to hybrid Lagrangian/Eulerian material representations.

Specifically, the mismatch in particle and grid degrees of freedom leads to a loss of information. Since there are often more particles than grid nodes, some particle modes are not seen by the grid and get no physical response. This is the so-called "ringing instability" (see Figure 4) which was first-discovered in PIC [Brackbill 1988] but is even more-pronounced in FLIP [Love and Sulsky 2006]. Intuitively, this problem is worse in FLIP, because in PIC, particle-to-grid transfer followed by grid-to-particle transfer is a true filtering of the instability. However, while PIC forces all information through the grid, FLIP preserves some particle information which allows the instability to persist and grow unpredictably over multiple time steps. This might lead one to believe that the ringing instability should not exist for PIC, however it does to a lesser extent since movement of particles re-creates the instability in the next time step. In graphics simulations these instabilities lead to practical particle positional artifacts such as noise, clumping and volume

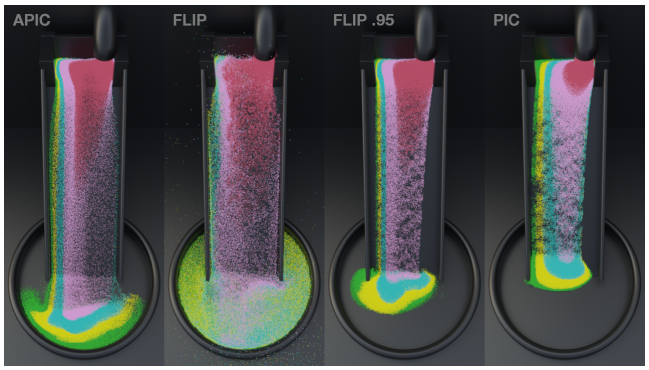

Figure 3: *We compare APIC with PIC, FLIP and FLIP/PIC blends for an MPM simulation of granular materials. Notice PIC and FLIP/PIC blend are stable but exhibit overly viscous behavior, while pure FLIP is unstable and noisy as evidenced by stray particles and excessive mixing. APIC however is both stable and nearly dissipation free.* ©*Disney.*

change.

A particularly problematic artifact of the dissipation with the traditional PIC approach is loss of angular momentum. The standard PIC transfer from grid to particles dissipates a significant amount of angular momentum, which leads to serious rotational artifacts (see Figure 4). Objects in free fall disturbingly stop rotating as if under the action of a viscous fluid drag. While FLIP was developed to reduce the dissipation of PIC, it also greatly improves the angular momentum conservation. However, FLIP will only guarantee exact conservation of angular momentum with the use of a non-diagonal (consistent) mass matrix, which is not possible in practice since the consistent mass matrix can be singular for some configurations of the particles [Love and Sulsky 2006]. This can be remedied in practice by using an effective mass matrix equal to a weighted average of a lumped mass matrix and the consistent mass matrix, and while this does not perfectly conserve angular momentum, it is still a vast improvement over the original PIC [Love and Sulsky 2006].

In graphics Zhu and Bridson [Zhu and Bridson 2005] advocate blending between pure PIC and FLIP to stabilize the simulator. While this does produce more stable behavior, it re-introduces dissipation and may require manual tuning of the blend weights on a case-by-case basis. The problem is particularly bad for thin sheets of fluid. This has been addressed in a number of ways including increasing resolution and adaptivity [Hong et al. 2008b; Hong et al. 2009; Ando and Tsuruno 2011; Ando et al. 2012; Ando et al. 2013]. For similar reasons, Um et al. develop a sub-grid-cell corrective forcing procedure to prevent particle bunching in [Um et al. 2014]. Also, Edwards and Bridson add a regularization term to diminish particle noise not corrected by the grid [Edwards and Bridson 2012].

The current state leaves us with the difficult choice for every simulation we run: (1) bias our simulation toward PIC, effectively avoiding instability at the expense of dissipation, or (2) bias our simulation toward FLIP, getting more lively simulations at the expense of noise and possible unstable behavior. In this paper we present a third option. In particular, we control noise by keeping the pure filter property of PIC but minimizing information loss by enriching each particle with a 3×3 matrix giving locally affine (rather than locally constant) description of the flow. This Affine Particle-In-Cell (APIC) method effectively reduces dissipation, preserves angular momentum and prevents instabilities. Furthermore, we demonstrate that the method is applicable to both incompressible liquids and Material Point Method (MPM) simulations [Sulsky et al. 1995].

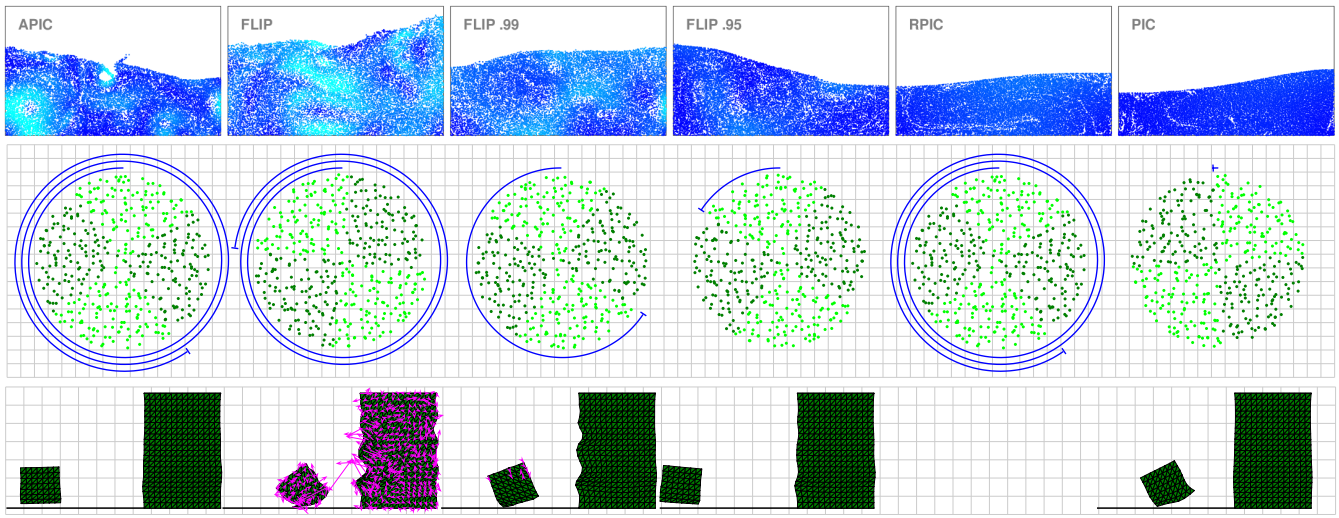

Figure 4: *For illustration, we compare performance with some simple 2D examples. The top row compares the methods in a dam break, free surface test. Note that APIC preserves more vorticity than even pure FLIP, while also remaining less noisy. The second row illustrates the angular momentum conservation properties of the methods. The blue spiral indicates how far the circle has rotated. The bottom row illustrates the ringing instability. Note that for pure FLIP, the velocities are large on particle but zero when transferred to grid.*

2 Previous work

PIC/FLIP: Foster and Metaxas first introduced PIC techniques to computer graphics with liquid simulation [Foster and Metaxas 1996]. Zhu and Bridson popularized the now widely-used linear combination of FLIP and PIC [Zhu and Bridson 2005]. Bridson et al. developed a number of extensions to [Zhu and Bridson 2005], including improved treatment of boundary conditions in irregular domains and coupling with rigid bodies [Batty et al. 2007], viscosity treatment [Batty and Bridson 2008], Discontinuous-Galerkin-based adaptivity [Edwards and Bridson 2014], multiphase flow [Boyd and Bridson 2012] and higher-order accuracy [Edwards and Bridson 2012]. Notably, the approach of Edwards and Bridson in [Edwards and Bridson 2014] is similar to ours in that both can be seen to improve results by using more data per cell. [Cornelis et al. 2014] couple high-resolution FLIP with a low-resolution implicit Smoothed Particle Hydrodynamics (SPH) from [Ihmsen et al. 2013]. Gerszewski and Bargteil use mass-full FLIP with a unilateral incompressibility constraint to resolve large-scale splashing liquids [Gerszewski and Bargteil 2013]. Narain et al. also use FLIP techniques for the simulation of sand dynamics [Narain et al. 2013]. Stomakhin et al. use MPM to simulate snow [Stomakhin et al. 2013] and melting/freezing [Stomakhin et al. 2014].

Figure 5: *Here we compare the methods on a fountain simulation of free-surface flow. The top row shows that while APIC and FLIP are the least dissipative in the initial stages, the FLIP surface is already displaying a noisy leading edge relative to the more smoothly resolved APIC. The bottom row shows that the entire surface of the fountain becomes noisy with FLIP in the later stages.* ©*Disney.*

Level Sets: Many other graphics approaches utilize similar hybrid particle/grid data structures, particularly for resolving free-surface flows. [Enright et al. 2002] use Lagrangian marker particles to improve the accuracy of the level set method for free-surface flows with the Particle Level Set Method (PLS). [Mihalef et al. 2007] take a similar approach but concentrate particles directly on the zero isocontour of the level set. Patkar et al. couple Lagrangian particles with the Particle Level Set Method [Enright et al. 2002] to simulate compressible bubbles in incompressible flow [Patkar et al. 2013]. Kim et al. explore further use of the escaped particles from [Enright et al. 2002] in [Kim et al. 2006]. Song et al. apply the Constrained Interpolation Profile (CIP) [Yabe et al. 2001] approach with [Enright et al. 2002] by allowing particles and grid to store velocity and level set derivative information in [Song et al. 2009].

Hybrid particle/grid: Several works couple SPH with grid-based techniques [Losasso et al. 2008; Hong et al. 2008a; Lee et al. 2009; Gao et al. 2009; Zhu et al. 2010; Raveendran et al. 2011]. Sin et al. couple particles with a Voronoi grid-based pressure projection [Sin et al. 2009]. Feldman et al. simulate explosions with a particle-based advection and grid-based pressure solve [Feldman et al. 2003]. Chentanez et al. couple Lagrangian particles with shallow water and semi-Lagrangian techniques to adapt level of detail in [Chentanez and Muller 2010; Chentanez and Muller 2014] and use particle reseeding for sub-cell detail [Chentanez and Muller 2011]. Muller et al. in [Muller et al. 2015] recently developed an SPH approach that augments particles with a sample of angular momentum. This is similar in spirit to some aspects of our approach. However, because their method is SPH based they do not use a grid and thus they do not use this to define particle-grid transfers. In contrast, we only use this information to improve transfers.

3 Method Outline

A Lagrangian/Eulerian hybrid simulation time step follows a similar pattern regardless of whether one is simulating fluids with incompressible FLIP or solids with MPM. Abstractly, kinematic steps are done on particles and dynamic steps are done on the grid. The exact form of those steps may be different with each phenomenon, and one can see examples of a canonical fluid loop in [Zhu and Bridson 2005] or MPM loop in [Stomakhin et al. 2013]. As such the basic timestep loop for PIC, FLIP and our method is shown

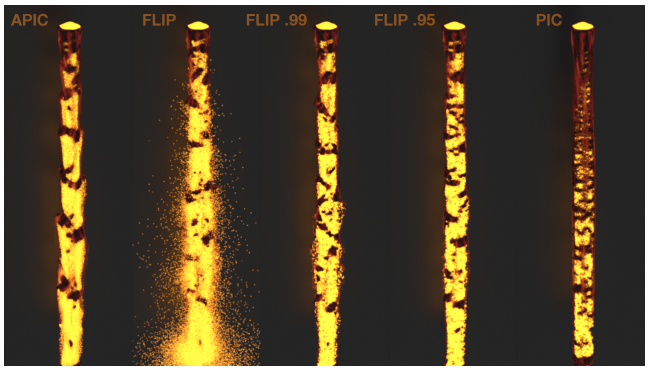

Figure 6: *We compare APIC with FLIP and PIC during a lava in free-fall example. Pure FLIP is unstable which leads to particles exploding out from the interior. FLIP/PIC blends do not suffer from this, but they cannot resolve the detailed flows shown in APIC.* ©*Disney.*

in Figure 7. The only difference between the methods is how the transfer between grid and particles is done. This difference is the focus of our paper.

PIC is the canonical technique for coupling, and its diagram clearly shows that all data flows through the grid, and in fact the transfer from particle to grid is a prefilter to the grid dynamics. By contrast, even though FLIP has the same prefilter when going from particles to the grid, it introduces an additional data path directly from the original particle state. The advantage is less dissipation, but the disadvantage is an unsafe path that can lead to instability. We will show that this path is not the only way to reduce dissipation. In fact, we show that we can obtain low-dissipation simulations while still maintaining the safety of the PIC filtering scheme.

The key observation is that normally a single particle receives data from multiple grid points, but it is typically forced to reduce those influences to a single constant value, leading to loss of information (e.g. dissipation). In Section 5 we develop two approaches for enriching particles to avoid this loss. Rigid Particle-in-cell (RPIC) is introduced in Section 5.2, and it augments each particle with the angular momentum lost in the grid to particle transfer. Unfortunately this is insufficient because shearing modes are still lost. This leads to our final method Affine Particle-in-cell (APIC) in which particles are endowed with a full affine representation of the local grid data, which we discuss in Section 5.3.

With our transfer technique developed, we show how to apply it to fluids in Section 6. In Section 7, we include a novel Lagrangian coupling technique for simulating a wide range of interesting behaviors that were impractical without APIC. Lastly, we demonstrate this method on a range of interesting materials in Section 8.

4 Notation

Before describing out method in detail, we begin by laying out the notation that we use in this paper. Many quantities have subscripts (m_p), and some have superscripts as well ($\mathbf{v}_p^n$). Subscripts p and q are used to refer to particles, and subscripts i and j are used to refer to regular grid indices. When dealing with MAC grids, the subscript a refers to an axis direction ($\mathbf{e}_a$, where $a \in \{1, 2, 3\}$ in 3D), and faces are referred to by cell index and axis (m_{ai}^n). The superscript distinguishes quantities available or computed at the beginning of the time step ($\mathbf{v}_p^n$) from quantities that are computed for use at the beginning of the next time step ($\mathbf{v}_p^{n+1}$). We use lowercase bold for vectors ($\mathbf{v}_p^n$) and uppercase bold for matrices ($\mathbf{B}_p^n$), with the exception of angular momentum, a vector quantity that is normally denoted by $\mathbf{L}_p^n$. The notation we use is summarized in Table 1.

Symbol	location	type	meaning
m_p	p	s	mass
$\mathbf{x}_p^n$	p	v	position
$\mathbf{v}_p^n$	p	v	velocity
$\mathbf{F}_p$	p	m	deformation gradient
$\mathbf{f}_p$	p	v	force
$\boldsymbol{\omega}_p^n$	p	v	angular velocity
$\mathbf{L}_p^n$	p	v	angular momentum
$\mathbf{K}_p^n$	p	m	inertia tensor
$\mathbf{B}_p^n$	p	m	affine state
$\mathbf{C}_p^n$	p	m	velocity derivatives
$\mathbf{c}_{pa}^n$	p	v	velocity derivatives (MAC only)
$\mathbf{D}_p^n$	p	m	inertia-like tensor
m_i^n	n	s	mass
$\mathbf{x}_i$	n	v	position
$\mathbf{v}_i^n$	n	v	velocity
$\tilde{\mathbf{v}}_i^{n+1}$	n	v	intermediate velocity
$\mathbf{f}_i$	n	v	force
w_{ip}^n	n+p	s	weights
∇w_{ip}^n	n+p	v	weight gradients
m_{ai}^n	f	s	mass
$\mathbf{x}_{ai}$	f	v	position
v_{ai}^n	f	s	velocity
$\tilde{v}_{ai}^{n+1}$	f	s	intermediate velocity
w_{aip}^n	f+p	s	weights
∇w_{aip}^n	f+p	v	weight gradients
Φ	g	s	total potential energy
$\mathbf{L}_{tot}^{P,n}$	g	v	total particle angular momentum
$\mathbf{L}_{tot}^{G,n}$	g	v	total grid angular momentum
$N(\mathbf{x})$	g	s	interpolation kernel
Δx	g	s	grid spacing
$\mathbf{e}_a$	g	v	axis vector
$\mathbf{v}^*$	g	m	cross product matrix of $\mathbf{v}$
$\boldsymbol{\epsilon}$	g	t	permutation tensor
$\mathbf{I}$	g	m	identity matrix

Table 1: *Summary of notation used in this paper. Locations are p (particle), n (regular grid node), f (MAC face center), or g (global; does not live at any location in space). Quantities are of type s (scalar), v (vector), m (matrix) or t (rank-3 tensor).*

5 Particle-grid transfers

5.1 PIC

The standard PIC routine stores mass m_p, position $\mathbf{x}_p^n$, and velocity $\mathbf{v}_p^n$. Note that m_p lacks a time n superscript because it is never changed to ensure mass conservation. Each time step begins with a transfer of mass and momentum from particles to a collocated grid according to

$$m_i^n = \sum_p w_{ip}^n m_p, \quad m_i^n \mathbf{v}_i^n = \sum_p w_{ip}^n m_p \mathbf{v}_p^n, \tag{1}$$

where $w_{ip}^n = N(\mathbf{x}_p^n - \mathbf{x}_i)$ are our interpolation weights and $\mathbf{x}_i$ denote the regular Cartesian grid node locations. With mass and momentum on the grid, we apply forces to the velocities in a grid-based update, $\mathbf{v}_i^n \rightarrow \tilde{\mathbf{v}}_i^{n+1}$. Note that we have used $\tilde{\mathbf{v}}_i^{n+1}$ rather than $\mathbf{v}_i^{n+1}$ to distinguish them from $\mathbf{v}_i^n$ at the next time step. Finally, we interpolate the velocity back to particles with

$$\mathbf{v}_p^{n+1} = \sum_i w_{ip}^n \tilde{\mathbf{v}}_i^{n+1}. \tag{2}$$

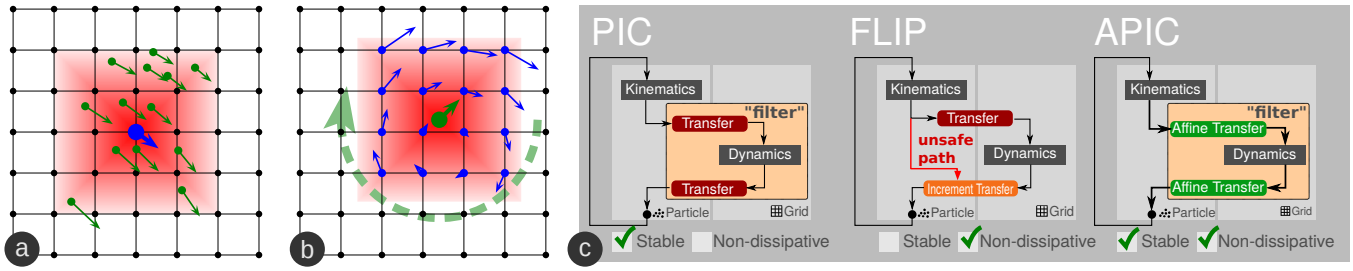

Figure 7: *(a) The traditional view of PIC is of a grid filter, which averages particles to a grid node. (b) An equivalent view is of a filter on particles, which assigns contributions to particles. With this view, particles are naturally fuzzy and have size, which makes it meaningful for particles to have properties like angular momentum. (c) The basic dataflow of various hybrid particle/grid simulation techniques. Both our method and PIC gain stability by using a true filtering transfer.*

A major problem with PIC is that it severely damps rotational motion. We can get some insight into this by considering the angular momentum conservation properties in the transfers. Note that linear momentum is conserved by both transfers. The total angular momentum over the particles and grid at time t^n are given by

$$\mathbf{L}_{tot}^{P,n} = \sum_p \mathbf{x}_p^n \times m_p \mathbf{v}_p^n \qquad \mathbf{L}_{tot}^{G,n} = \sum_i \mathbf{x}_i \times m_i^n \mathbf{v}_i^n. \tag{3}$$

While the transfer from particles to the grid conserves angular momentum, the transfer from the grid back to particles does not. This loss of angular momentum manifests as rotational motion damping (see Figure 4).

5.2 Rigid Particle-In-Cell (RPIC)

In our efforts to reduce the information loss when transferring from particles to grid and vice versa, we first develop modifications to the original PIC transfer designed to facilitate conservation of angular momentum in the grid to particle transfer. Consider the case of a single particle. PIC typically transfers information to multiple grid locations since w_{ip}^n is generally non-zero for a few grid nodes at a time. Thus, even for a single particle of material, its corresponding representation on the grid is capable of storing angular momentum (by virtue of consisting of multiple grid nodes). However, one particle is incapable of representing angular momentum. Therefore, to improve the compatibility of the particle representation with the grid representation, we can additionally store a sample of local angular momentum $\mathbf{L}_p^n$ on each particle. This way, even in the case of one particle, we can prevent the loss of angular momentum when transferring from grid to particle.

The angular momentum that would normally be lost in the transfer from grid to particles is

$$\mathbf{L}_p^{n+1} = \sum_i w_{ip}^n (\mathbf{x}_i - \mathbf{x}_p^n) \times m_p \tilde{\mathbf{v}}_i^{n+1}. \tag{4}$$

With this definition, the total angular momentum on particles becomes

$$\mathbf{L}_{tot}^{P,n} = \sum_p (\mathbf{x}_p^n \times m_p \mathbf{v}_p^n + \mathbf{L}_p^n). \tag{5}$$

Also, with this definition, the transfer from grid to particles trivially conserves angular momentum. Next, we must define the transfer from particles to the grid. If we consider the particles to be rigid bodies with inertia tensors $\mathbf{K}_p^n$, then we can define the angular velocity $\boldsymbol{\omega}_p^n = (\mathbf{K}_p^n)^{-1}\mathbf{L}_p^n$. The rigid body's local velocity at a grid node is $\mathbf{v}_p^n + \boldsymbol{\omega}_p^n \times (\mathbf{x}_i - \mathbf{x}_p^n)$, which suggests the natural transfer

$$m_i^n \mathbf{v}_i^n = \sum_p w_{ip}^n m_p (\mathbf{v}_p^n + ((\mathbf{K}_p^n)^{-1}\mathbf{L}_p^n) \times (\mathbf{x}_i - \mathbf{x}_p^n)), \tag{6}$$

One may imagine this transfer as distributing the masses $w_{ip}^n m_p$ from the rigid body to the grid node i. This suggests using

$$\mathbf{K}_p^n = \sum_j w_{jp}^n m_p (\mathbf{x}_j - \mathbf{x}_p^n)^* (\mathbf{x}_j - \mathbf{x}_p^n)^{*T} \tag{7}$$

for the rigid body's inertia tensor, where $\mathbf{v}^*$ is the cross-product matrix associated with vector $\mathbf{v}$. Performing the transfer from particles to grid in this way conserves angular momentum as shown in a supplementary document.

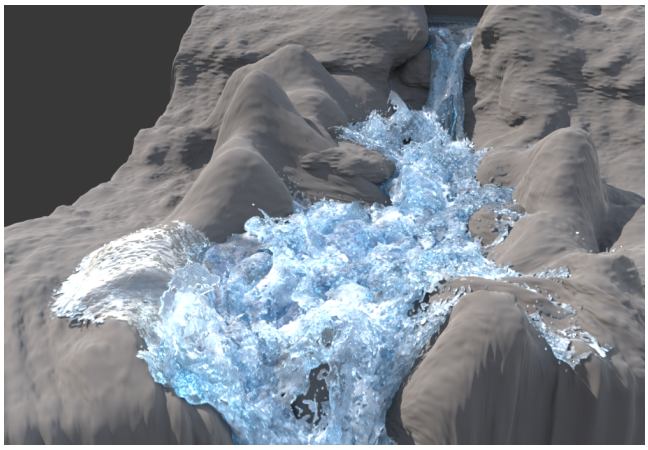

Figure 8: *APIC resolves the complex free-surface dynamics of a rushing river on a rocky terrain. ©Disney.*

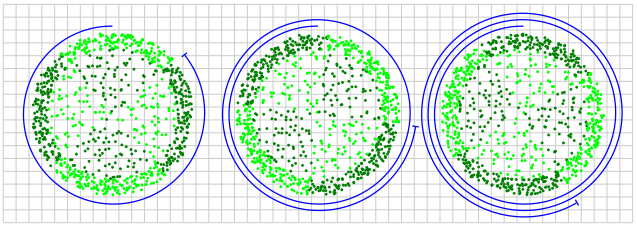

Figure 9: *APIC can handle non-uniform particle distributions. It rotates cleanly even when the particle distribution is higher near the boundary, such as in this example where particle sampling (and mass) is quadrupled in a thin band near the interface.*

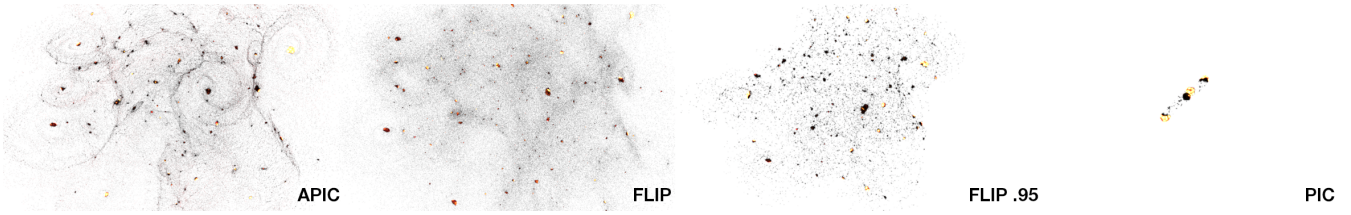

Figure 10: *We compare APIC with FLIP and PIC in a high energy collision example. APIC resolves interesting spiral shedding behavior that the other methods cannot.* ©*Disney.*

5.3 Affine Particle-In-Cell (APIC)

While the piecewise rigid formulation corrects rotational artifacts arising from loss of angular momentum in PIC, it still damps out non-rigid motions such as shearing (see supplemental video). We can extend the idea of enriching our velocity representation to handle shearing modes by idealizing the velocity as locally affine on each particle. This requires the introduction of a matrix $\mathbf{C}_p^n$, and the local velocity represented by a particle at the grid node $\mathbf{x}_i$ is then $\mathbf{v}_p^n + \mathbf{C}_p^n(\mathbf{x}_i - \mathbf{x}_p^n)$. This can be used to define a transfer from particles to grid as in the piecewise rigid case. However, uniquely defining the nine components of $\mathbf{C}_p^n$ from the three components of $\mathbf{L}_p^n$ as in the piecewise rigid case is not possible, which complicates the process of deriving the transfer from grid to particles.

An important feature of the piecewise rigid formulation is that translational and rotational velocity fields are transferred exactly from particles to the grid and vice versa. Rather than explicitly trying to conserve angular momentum in the transfer from grid to particles, we seek to preserve affine velocity fields across both transfers. However, we show that a simple solution derived from the preservation of affine velocity fields also conserves angular momentum in a supplementary document.

The transfer from particles to grid is motivated analogously to the piecewise rigid case and is of the form

$$m_i^n \mathbf{v}_i^n = \sum_p w_{ip}^n m_p (\mathbf{v}_p^n + \mathbf{B}_p^n (\mathbf{D}_p^n)^{-1} (\mathbf{x}_i - \mathbf{x}_p^n)), \tag{8}$$

where $\mathbf{C}_p^n = \mathbf{B}_p^n (\mathbf{D}_p^n)^{-1}$ and $\mathbf{D}_p^n$ is analogous to an inertia tensor. $\mathbf{D}_p^n$ is given by

$$\mathbf{D}_p^n = \sum_i w_{ip}^n (\mathbf{x}_i - \mathbf{x}_p^n)(\mathbf{x}_i - \mathbf{x}_p^n)^T \tag{9}$$

and is derived by preserving affine motion during the transfers. The corresponding transfer from the grid back to particles is

$$\mathbf{B}_p^{n+1} = \sum_i w_{ip}^n \tilde{\mathbf{v}}_i^{n+1} (\mathbf{x}_i - \mathbf{x}_p^n)^T. \tag{10}$$

To discuss the angular momentum conservation properties of the transfer, we must first define angular momentum over the new particle state. A natural definition is the angular momentum on the grid after the transfer in Equation 8. This takes the form

Simulation	Particles	Memory Consumption		PIC/FLIP Blend				
		FLIP Base	Affine Overhead	1.00	0.99	0.95	PIC	APIC
Wine	4.3×10^4	360 MB	1.5 MB	1.2	1.1	1.1	1.0	1.1
River	1.4×10^6	2.2 GB	50 MB	-	1.0	1.1	1.0	1.2
Collision	1.1×10^6	296 MB	40 MB	9.5	3.2	1.5	1.0	5.0
Lava free-fall	1.9×10^6	511 MB	68 MB	2.5	1.1	0.9	1.0	1.3
Water fountain	3.4×10^5	677 MB	12 MB	1.1	1.2	1.5	1.0	1.0
Sand incline	2.9×10^5	78 MB	10 MB	4.0	1.9	1.5	1.0	2.2

Table 2: *Performance statistics for our 3D simulations. The last five columns present runtimes as a multiple of the runtime for the PIC simulation to eliminate the performance impact from our particular simulator. Typically simulation performance is correlated to CFL, thus unstable and lively simulations take longer.*

$$\mathbf{L}_{tot}^{P,n} = \sum_p m_p (\mathbf{x}_p^n \times \mathbf{v}_p^n + (\mathbf{B}_p^n)^T : \boldsymbol{\epsilon}), \tag{11}$$

where $\boldsymbol{\epsilon}$ is the permutation tensor. We take the convention that $\mathbf{A} : \boldsymbol{\epsilon}$ means the same thing as $A_{\alpha\beta}\epsilon_{\alpha\beta\gamma}$. Note that the skew-symmetric component of $\mathbf{B}_p^n$ contains all of the angular momentum information. In this way, it is analogous to $\mathbf{L}_p^n = \mathbf{K}_p^n \boldsymbol{\omega}_p^n$ which combined with $\mathbf{B}_p^n = \mathbf{C}_p^n \mathbf{D}_p^n$ illustrates that $\mathbf{D}_p^n$ is analogous to the inertia tensor. Using this definition, conservation during transfer from particles to grid is automatic. We show that angular momentum is conserved during the transfer from grid to particle in a supplementary document. We note that momentum conservation allows us to resolve rotations correctly; we do not rely on uniform sampling or fortuitous cancellation to achieve this (See Figure 9).

Note that despite these similarities (9) this is not quite the same as (7), and $\mathbf{D}_p^n$ does not strictly speaking have the properties of an inertia tensor. Conveniently, $\mathbf{D}_p^n$ takes on a surprisingly simple form in the case of the quadratic ($\mathbf{D}_p^n = \frac{1}{4}\Delta x^2 \mathbf{I}$) and cubic ($\mathbf{D}_p^n = \frac{1}{3}\Delta x^2 \mathbf{I}$) interpolation stencils commonly used for MPM. Note that for these interpolating stencils, multiplying by $(\mathbf{D}_p^n)^{-1}$ amounts to a constant scaling factor. For trilinear interpolation, a complication arises since $\mathbf{D}_p^n$ may be singular if a particle lies on a grid facet (node, edge or face). However, in the special case of a trilinear stencil, we have the convenient identity $w_{ip}^n (\mathbf{D}_p^n)^{-1} (\mathbf{x}_i - \mathbf{x}_p^n) = \nabla w_{ip}^n$, which allows us to avoid this numerical difficulty entirely since Equation 8 can be readily evaluated without forming $(\mathbf{D}_p^n)^{-1}$.

6 Fluids

To apply these ideas to MAC-based fluid simulations, we formulate a set of transfers between particles and MAC faces. Instead of matrix $\mathbf{B}_p$, we store three vectors per particle denoted by $\mathbf{c}_{pa}$, where a represents an x, y or z face direction. We transfer from particles to faces using

$$m_{ai}^n = \sum_p m_p w_{aip}^n \tag{12}$$

$$m_{ai}^n v_{ai}^n = \sum_p m_p w_{aip}^n (\mathbf{e}_a^T \mathbf{v}_p^n + (\mathbf{c}_{pa}^n)^T (\mathbf{x}_{ai} - \mathbf{x}_p^n)) \tag{13}$$

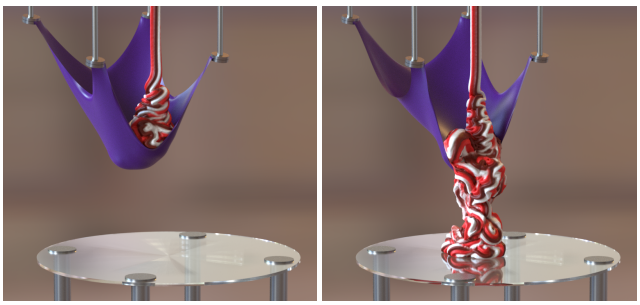

Figure 11: *An APIC coupled simulation of elastoplastic frozen yogurt and elastic cloth where coupling is achieved using our MPM approach with Lagrangian energy-based forces.* ©*Disney.*

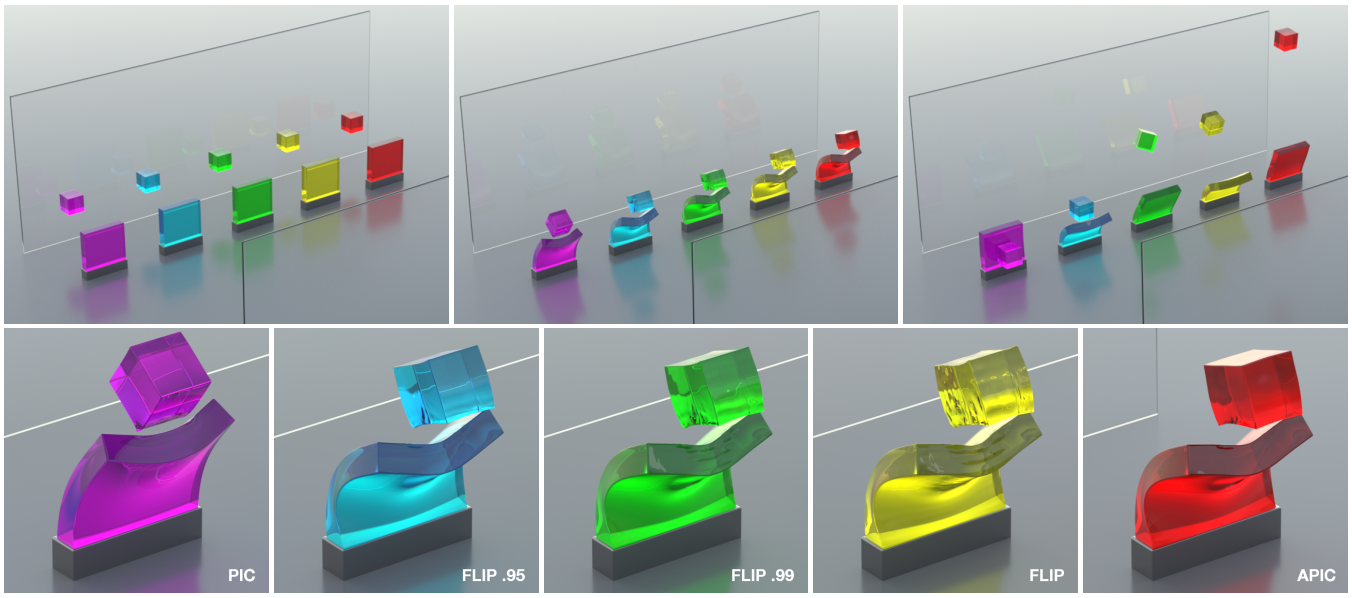

Figure 12: *We compare APIC with FLIP and PIC using the Lagrangian force model from Section 7 and a collision scenario with significant angular momentum. APIC preserves angular momentum better than even pure FLIP, and is at the same time the most stable of the various options. In the bottom row we show that the cube surface remains smooth after collision with APIC relative to the behavior of FLIP.* ©*Disney.*

and from faces to particles using

$$\mathbf{v}_p^{n+1} = \sum_{a,i} w_{aip}^n \tilde{v}_{ai}^{n+1} \mathbf{e}_a \quad \text{and} \quad \mathbf{c}_{pa}^{n+1} = \sum_i \nabla w_{aip}^n \tilde{v}_{ai}^{n+1}. \quad (14)$$

Here, $\mathbf{x}_{ai}$ is the location of a MAC face associated with direction a. The weights are $w_{aip}^n = N(\mathbf{x}_{ai} - \mathbf{x}_p^n)$, where $N(\mathbf{x})$ is chosen to be the trilinear interpolation kernel. m_{ai}^n and v_{ai}^n are the mass and velocity component on the MAC face, and $\mathbf{c}_{pa}^n$ is a vector per axis, notably requiring the same amount of storage as the collocated case. Incompressibility is enforced in the standard way [Bridson 2008].

7 Lagrangian forces

Here we describe a new useful model for MPM force computation that we used in a few of our examples. The discussion in this section is independent of grid-particle transfers and works with even the traditional PIC/FLIP based MPM.

For most of our simulations, we compute forces as in [Stomakhin et al. 2013]. This approach has the advantages of a mesh-free method, such as effortless topology change. For objects that are not intended to undergo topology change, a meshed approach is also available using the ideas in [Sifakis et al. 2007]. In this case, we can use any Lagrangian force model (springs, finite elements, etc.) for which we can write down total potential energy $\Phi(\mathbf{x}_p)$. Corresponding to this Lagrangian force model, we compute forces $\mathbf{f}_p = -\frac{\partial \Phi}{\partial \mathbf{x}_p}$. We also assume that given any vector $\delta \mathbf{u}_q$ on particles we can multiply by force derivatives $\frac{\partial \mathbf{f}_p}{\partial \mathbf{x}_q}$ to obtain $\delta \mathbf{f}_p = \sum_q \frac{\partial \mathbf{f}_p}{\partial \mathbf{x}_q} \delta \mathbf{u}_q$. Since these force-related constructs are purely Lagrangian and are computed in the usual way, we will not elaborate on them here.

Although we have defined our mesh-based forces as Lagrangian forces, we must still apply them through the grid. We must describe how particle positions $\mathbf{x}_p$ relate to our (conceptually) moving grid nodes $\mathbf{x}_i$ so that forces can be evaluated. Then, we must compute $\mathbf{f}_i$ from $\mathbf{f}_p$ and find a means of computing $\delta \mathbf{f}_i$ given $\delta \mathbf{u}_i$. Doing this allows us to use Lagrangian forces as Eulerian forces. Comparing our update rules for $\mathbf{x}_p$ and $\mathbf{x}_i$ we find $\mathbf{x}_p = \sum_i w_{ip}^n \mathbf{x}_i$. Using the chain rule,

$$\mathbf{f}_i = \sum_p w_{ip}^n \mathbf{f}_p \qquad \delta \mathbf{f}_i = \sum_{p,q,j} w_{ip}^n \frac{\partial \mathbf{f}_p}{\partial \mathbf{x}_q} w_{jq}^n \delta \mathbf{u}_j. \quad (15)$$

Although this formula is written with three nested summations, the computation can be done efficiently by computing the summations consecutively. Since these forces are applied to the grid, both the MPM and Lagrangian approaches can be employed in the same simulation. Each particle is labeled as an MPM particle or a meshed particle. Note that the deformation gradient $\mathbf{F}_p^n$ stored on meshed particles is never used, since for those particles this quantity is computed using the mesh. This provides an effective means of coupling MPM with mesh-based approaches, such as shown in Figure 11. This gives the precise surface tracking of Lagrangian techniques coupled with the automatic collision handling of Eulerian grids.

8 Results

Free-surface flow: The most common graphics application of PIC and FLIP is free-surface incompressible flow (Figure 8). We compare APIC with pure FLIP, PIC and FLIP/PIC blends in a number of free-surface calculations. Figure 5 shows a comparison with a fountain example. The top row (earlier time) shows that while APIC and FLIP start as the least dissipative, the FLIP surface is already displaying a noisy leading edge relative to the more smoothly resolved APIC. PIC is the most damped, and any amount of PIC/FLIP blending leads to a damped leading edge position. The bottom row (later time) shows that FLIP simulates a noisy surface which is still visible on FLIP/PIC blends. In Figure 1 we demonstrate the behavior of APIC/PIC blends compared to FLIP/PIC. We note that unlike with FLIP, pure APIC is clearly superior to APIC/PIC blends. Note that APIC produces more smooth and stable wine surfaces than FLIP/PIC blends while simultaneously resolving more energetic splashing behavior than even pure FLIP.

Granular materials: In Figure 3 we compare APIC with FLIP and FLIP/PIC blends. We model the sand as a granular material using the constitutive model from [Stomakhin et al. 2013]. We use an

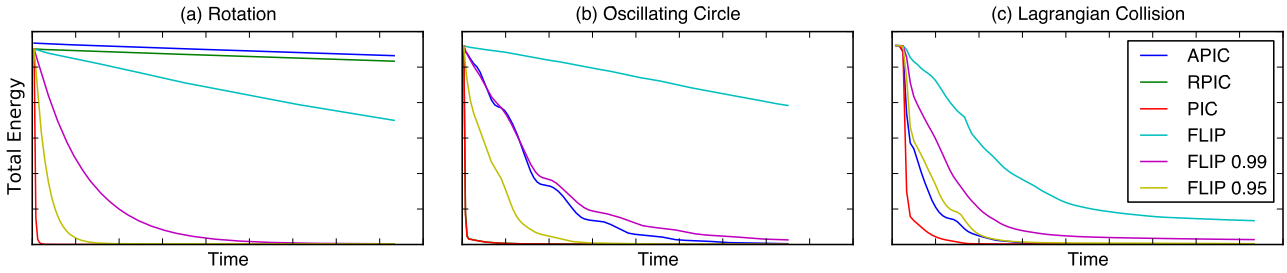

Figure 13: *Total energy is plotted over time for the rotation test (a), an oscillating circle test (b), and the Lagrangian collision test (c). Note that APIC and RPIC are quite effective at resolving rotation, but FLIP is more consistent about retaining energy. FLIP is able to retain kinetic energy by holding it in transfer null modes, where it is immune to dissipation. This is particularly evident in the Lagrangian collision test, where FLIP particles have a significant amount of kinetic energy, even though the objects are not moving. Slight energy increases are observed to occur rarely across the grid update, but energy increase is never observed across the APIC transfers (from grid to particle and back to grid).*

inlet condition at the top of the slide to induce a mixing flow. The dynamics demonstrate the noise of pure FLIP and the dissipation of PIC and FLIP/PIC blends. APIC is able to retain the stability of PIC and FLIP/PIC blends without the excessive dissipation. We show another comparison with granular materials in Figure 2. Here, FLIP again exhibits overly noisy behavior. The dissipation in the PIC method causes the sand to bunch together giving it a wet look. While the FLIP/PIC blend is more stable than pure FLIP, it also suffers from the clumping, wet look of PIC. However, APIC stably resolves the dynamics of a fine powdery sand. In Figure 10 we show that APIC is better able to retain angular momentum without the dispersive behavior of FLIP.

Coupling MPM and Lagrangian: We demonstrate the accuracy and robustness of APIC with a coupling example in Figure 11. Here we use a traditional MPM discretization of the elastoplastic constitutive model from [Bargteil et al. 2007] to simulate frozen yogurt. We couple this with an elastic cloth using the Lagrangian force model outlined in Section 7. The cloth is modeled using a standard mass-spring energy. In Figure 12 we show a comparison using mesh-based cubes with a Lagrangian finite element constitutive model. Here, APIC retains angular momentum and energetic behavior better than PIC, FLIP, and FLIP/PIC blends. FLIP and FLIP/PIC blends produce ringing during the collisions between the cube and the glass plates and flexible block.

Lava: We demonstrate the benefits of APIC for lava flows using the model from [Stomakhin et al. 2014]. In Figure 16 we show interesting lava flows over a rocky terrain. We directly compare APIC with FLIP and FLIP/PIC blends in Figure 6. In this example, a spout

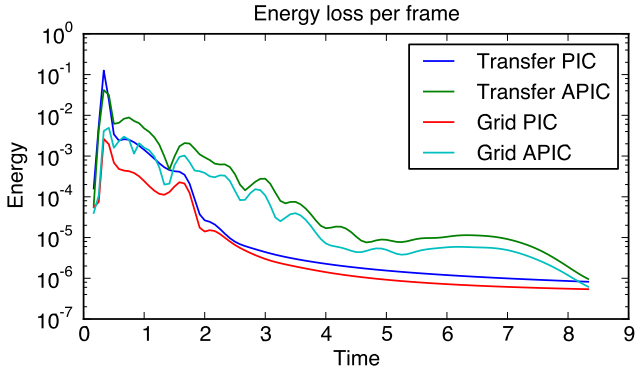

Figure 14: *Energy loss per frame is shown (on a logarithmic scale) for an oscillating circle test, broken down by whether the energy was lost due to the transfer step or due to grid-based sources such as the backward Euler step. Total energy loss for the entire simulation due to transfers is* 82% *for APIC compared to* 95% *for PIC.*

pours lava with cooler, rockier properties near the outer circumference into free-fall. APIC resolves an interesting periodic flow, while pure FLIP goes unstable, with hotter interior particles noisily exploding outwards. FLIP/PIC blends stabilize this behavior, but do not produce flows as detailed as with APIC.

Energy: To explore energy loss, we set up an oscillating circle test with domain $[0,1] \times [0,1]$ and resolution 32×32 in which we create a circle centered at $(0.5, 0.5)$ with radius 0.3 by uniformly sampling four particles per grid cell. The circle has density 2, initial velocity $\langle x - 0.5, 0 \rangle$, and the constitutive model from [Stomakhin et al. 2013] with parameters $E = 0.5$ and $\nu = 0.4$. Results are shown in Figure 13 and Figure 14.

9 Discussion and limitations

Although we eliminate nearly all of the artificial dissipation of pure PIC (see Figure 14 for a detailed comparison), our method neither improves nor exacerbates the ringing instability. This is quite unlike FLIP, which avoids dissipation at the cost of losing stability. However, our method does filter information in the transfer from grid to particle while FLIP does not. For example Figure 13 shows that FLIP does a better job than APIC and RPIC preserving energy in many cases. This could be an advantage for FLIP if stability could be achieved by some other means.

Another minor disadvantage of the approach is the need to store an extra matrix per particle and perform a few extra operations during the transfers. In practice, we have found the extra storage and transfer cost to be negligible as runtime costs are typically dominated by the magnitude of the velocities and our CFL (see Figure 15 for timing breakdown over a typical time step and Table 2 for typical memory usage). PIC tends to be fastest, since it damps out motion and has the smallest velocities. On the other hand, FLIP tends to be slowest due to its instability and consequent larger velocities. In particular, we see that the wine timings in Figure 15 are fairly uniform across all methods, because the maximum velocity is similar. Similarly, unstable FLIP simulations like the high energy collision tend to be very slow.

It is interesting to note that a stable PIC/FLIP blend often contains artifacts that are desirable, especially in the case of liquids and wet sand. FLIP in some ways is akin to forcing functions that are required to make grid-based smoke solvers visually interesting. We assert, however, that if such instabilities are desirable, the artist would prefer to create them in a way they desire rather than have them uncontrollably imposed by the method.

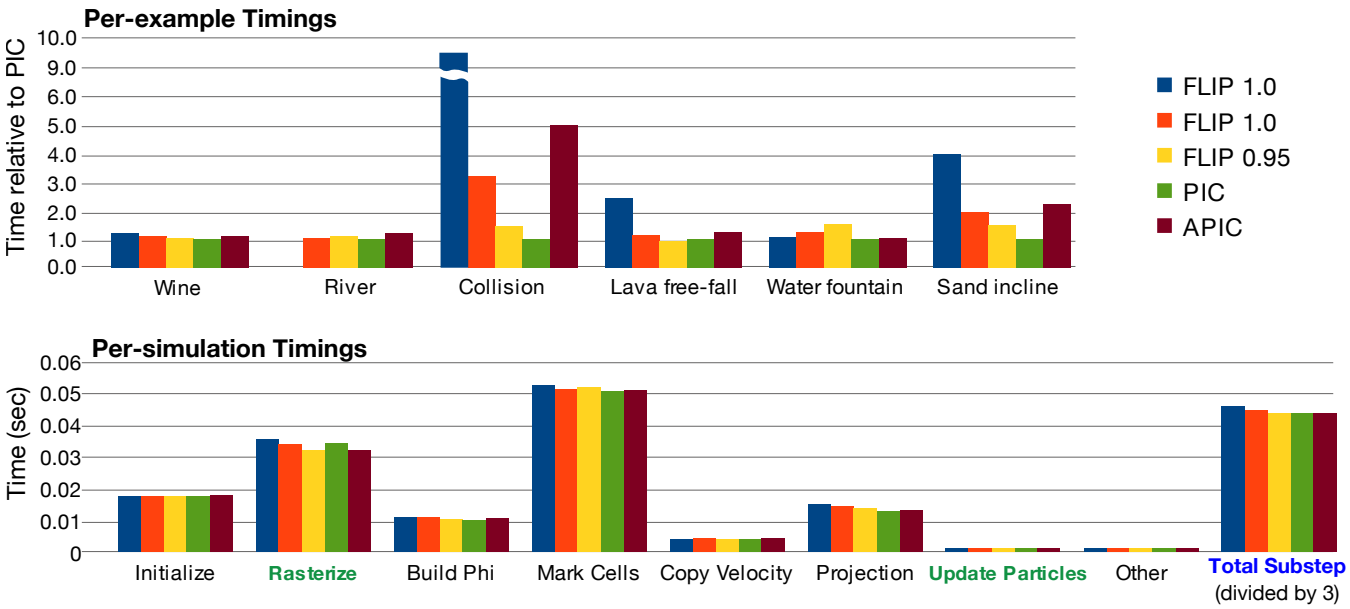

Figure 15: *The top chart compares timings of FLIP and APIC for a number of examples. Timings are calculated relative to a standard PIC simulation. Runtime breakdown for a typical time step for the wine simulation is shown on bottom. Steps changed by APIC are shown with green labels. The total runtime for the time step is shown with a blue label, scaled down by a factor of three to fit the plot range.*

Acknowledgements

UCLA authors were partially supported by NSF CCF-1422795, ONR (N000141110719, N000141210834), Intel STC-Visual Computing Grant (20112360) as well as a gift from Disney Research. We also appreciate the support at the studio from D. Meltzer, R. Sharma, D. Candela, A. Hendrickson, and M. Bryant. Images rendered using Disney's Hyperion Renderer.

References

ANDO, R., AND TSURUNO, R. 2011. A particle-based method for preserving fluid sheets. In *Proc ACM SIGGRAPH/Eurographics Symp Comp Anim*, SCA '11, 7–16.

ANDO, R., THUREY, N., AND TSURUNO, R. 2012. Preserving fluid sheets with adaptively sampled anisotropic particles. *IEEE Trans Vis Comp Graph 18*, 8, 1202–1214.

ANDO, R., THUREY, N., AND WOJTAN, C. 2013. Highly adaptive liquid simulations on tetrahedral meshes. *ACM Trans Graph 32*, 4, 103:1–103:10.

BARGTEIL, A., WOJTAN, C., HODGINS, J., AND TURK, G. 2007. A finite element method for animating large viscoplastic flow. *ACM Trans Graph 26*, 3.

BATTY, C., AND BRIDSON, R. 2008. Accurate viscous free surfaces for buckling, coiling, and rotating liquids. *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, 219–228.

BATTY, C., BERTAILS, F., AND BRIDSON, R. 2007. A fast variational framework for accurate solid-fluid coupling. *ACM Trans Graph 26*, 3.

BOYD, L., AND BRIDSON, R. 2012. Multiflip for energetic two-phase fluid simulation. *ACM Trans Graph 31*, 2, 16:1–16:12.

BRACKBILL, J., AND RUPPEL, H. 1986. Flip: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *J Comp Phys 65*, 314–343.

BRACKBILL, J., KOTHE, D., AND RUPPEL, H. 1988. Flip: A low-dissipation, pic method for fluid flow. *Comp Phys Comm 48*, 25–38.

BRACKBILL, J. 1988. The ringing instability in particle-in-cell calculations of low-speed flow. *J Comp Phys 75*, 2, 469–492.

BRIDSON, R. 2008. *Fluid simulation for computer graphics*. Taylor & Francis.

CHENTANEZ, N., AND MULLER, M. 2010. Real-time simulation of large bodies of water with small scale details. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, SCA '10, 197–206.

CHENTANEZ, N., AND MULLER, M. 2011. Real-time eulerian water simulation using a restricted tall cell grid. *ACM Trans Graph 30*, 4, 82:1–82:10.

CHENTANEZ, N., AND MULLER, M. 2014. Coupling 3d eulerian, height field and particle methods for the simulation of large scale liquid phenomena. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, SCA '14.

CORNELIS, J., IHMSEN, M., PEER, A., AND TESCHNER, M. 2014. Iisph-flip for incompressible fluids. *Comp Graph Forum 33*, 2, 255–262.

EDWARDS, E., AND BRIDSON, R. 2012. A high-order accurate particle-in-cell method. *Int J Numer Meth Eng 90*, 1073–1088.

EDWARDS, E., AND BRIDSON, R. 2014. Detailed water with coarse grids: combining surface meshes and adaptive discontinuous galerkin. *ACM Trans Graph 33*, 4, 136:1–136:9.

ENRIGHT, D., MARSCHNER, S., AND FEDKIW, R. 2002. Animation and rendering of complex water surfaces. *ACM Trans Graph 21*, 3, 736–744.

FELDMAN, B., O'BRIEN, J., AND ARIKAN, O. 2003. Animating suspended particle explosions. *SIGGRAPH '03 22*, 3, 708–715.

FOSTER, N., AND METAXAS, D. 1996. Realistic animation of liquids. *Graph Mod Imag Proc 58*, 471–483.

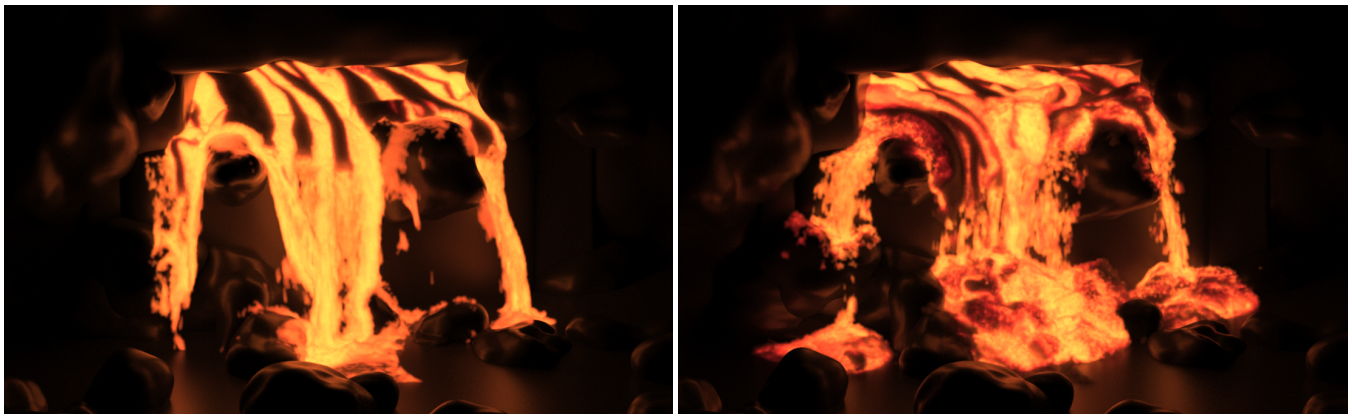

Figure 16: *Here we demonstrate the performance of APIC with an elastoplastic model for lava flow. ©Disney.*

GAO, Y., LI, C., HU, S., AND BARSKY, B. 2009. Simulating gaseous fluids with low and high speeds. *Comp Graph Forum 28*, 28, 1845–1852.

GERSZEWSKI, D., AND BARGTEIL, A. 2013. Physics-based animation of large-scale splashing liquids. *ACM Trans Graph 32*, 6, 185:1–185:6.

HARLOW, F., AND WELCH, E. 1965. Numerical calculation of time dependent viscous flow of fluid with a free surface. *Phys Fluid 8*, 12, 2182–2189.

HARLOW, F. 1964. The particle-in-cell method for numerical solution of problems in fluid dynamics. *Meth Comp Phys 3*, 319–343.

HONG, J., LEE, H., YOON, J., AND KIM, C. 2008. Bubbles alive. *ACM Trans Graph 27*, 3, 48:1–48:4.

HONG, W., HOUSE, D., AND KEYSER, J. 2008. Adaptive particles for incompressible fluid simulation. *Vis Comp 24*, 7, 535–543.

HONG, W., HOUSE, D., AND KEYSER, J. 2009. An adaptive sampling approach to incompressible particle-based fluid. *Theory Pract Comp Graph*, 69–76.

IHMSEN, M., CORNELIS, J., SOLENTHALER, B., HORVATH, C., AND TESCHNER, M. 2013. Implicit incompressible sph. *IEEE Trans Vis Comp Graph 20*, 3, 426–435.

KIM, J., CHA, D., CHANG, B., KOO, B., AND IHM, I. 2006. Practical animation of turbulent splashing water. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, SCA '06, 335–344.

LEE, H., HONG, J., AND KIM, C. 2009. Interchangeable sph and level set method in multiphase fluids. *Vis Comp 25*, 5, 713–718.

LOSASSO, F., TALTON, J., KWATRA, N., AND FEDKIW, R. 2008. Two-way coupled sph and particle level set fluid simulation. *IEEE Trans Vis Comp Graph 14*, 797–804.

LOVE, E., AND SULSKY, D. 2006. An unconditionally stable, energy-momentum consistent implementation of the the material point method. *Comp Meth App Mech Eng 195*, 3903–3925.

MIHALEF, V., METAXAS, D., AND SUSSMAN, M. 2007. Textured liquids based on the marker level set. *Comp Graph Forum*, 457–466.

MULLER, K., FEDOSOV, D., AND GOMPPER, G. 2015. Smoothed dissipative particle dynamics with angular momentum conservation. *J Comp Phys 281*, 301–315.

NARAIN, R., GOLAS, A., AND LIN, M. 2013. Free-flowing granular materials with two-way solid coupling. *ACM Trans Graph 29*, 6, 173:1–173:10.

PATKAR, S., AANJANEYA, M., KARPMAN, D., AND FEDKIW, R. 2013. A hybrid lagrangian-eulerian formulation for bubble generation and dynamics. In *Proc ACM SIGGRAPH/Eurograp Symp Comp Anim*, SCA '13, 105–114.

RAVEENDRAN, K., WOJTAN, C., AND TURK, G. 2011. Hybrid sph. In *Proc 2011 ACM SIGGRAPH/Eurograp Symp Comp Anim*, SCA '11, 33–42.

SIFAKIS, E., SHINAR, T., IRVING, G., AND FEDKIW, R. 2007. Hybrid simulation of deformable solids. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, 81–90.

SIN, F., BARGTEIL, A., AND HODGINS, J. 2009. A point-based method for animating incompressible flow. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, 247–255.

SONG, O., KIM, D., AND KO, H. 2009. Derivative particles for simulating detailed movements of fluids. *IEEE Trans Vis Comp Graph*, 247–255.

STOMAKHIN, A., SCHROEDER, C., CHAI, L., TERAN, J., AND SELLE, A. 2013. A material point method for snow simulation. *ACM Trans Graph 32*, 4, 102:1–102:10.

STOMAKHIN, A., SCHROEDER, C., JIANG, C., CHAI, L., TERAN, J., AND SELLE, A. 2014. Augmented mpm for phase-change and varied materials. *ACM Trans Graph 33*, 4, 138:1–138:11.

SULSKY, D., ZHOU, S., AND SCHREYER, H. 1995. Application of a pic method to solid mechanics. *Comp Phys Comm 87*, 1, 236–252.

UM, K., BAEK, S., AND HAN, J. 2014. Advanced hybrid particle-grid method with sub-grid particle correction. *Comp Graph Forum 33*, 209–218.

YABE, T., XIAO, F., AND UTSUMI, T. 2001. The constrained interpolation profile method for multiphase analysis. *J Comp Phys 169*, 556–593.

ZHU, Y., AND BRIDSON, R. 2005. Animating sand as a fluid. *ACM Trans Graph 24*, 3, 965–972.

ZHU, B., YANG, X., AND FAN, Y. 2010. Creating and preserving vortical details in sph fluid. *Comp Graph Forum 29*, 7, 2207–2214.