Eulerian Solid-Fluid Coupling

Yun Teng* David I.W. Levin† Theodore Kim*‡

*University of California, Santa Barbara †Disney Research ‡Pixar Animation Studios

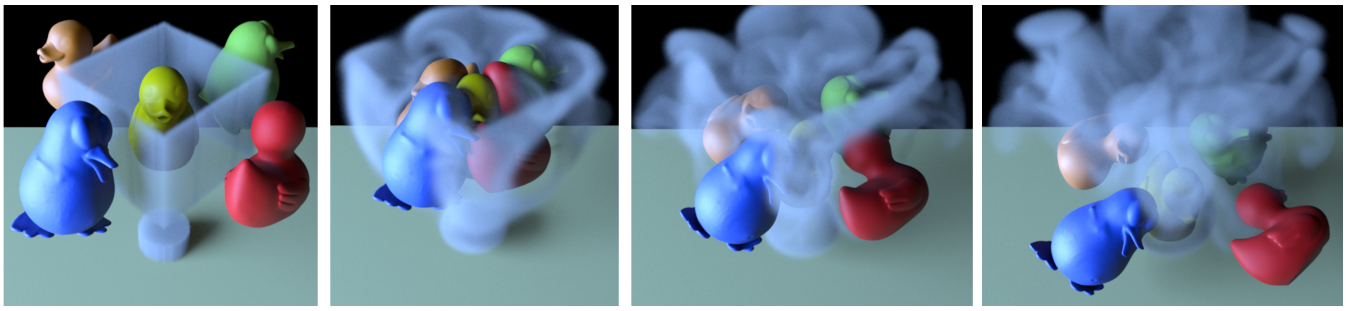


Figure 1: *"Party" scene: Three hyper-elastic and two elasto-plastic objects are squashed into a complex contact configuration, all while fully two-way coupled with the surrounding fluid. All of the objects and the fluid are represented on a* $200 \times 180 \times 200$ *Eulerian grid.*

Abstract

We present a new method that achieves a two-way coupling between deformable solids and an incompressible fluid where the underlying geometric representation is entirely Eulerian. Using the recently developed Eulerian Solids approach [Levin et al. 2011], we are able to simulate multiple solids undergoing complex, frictional contact while simultaneously interacting with a fluid. The complexity of the scenarios we are able to simulate surpasses those that we have seen from any previous method. Eulerian Solids have previously been integrated using explicit schemes, but we develop an implicit scheme that allows large time steps to be taken. The incompressibility condition is satisfied in both the solid and the fluid, which has the added benefit of simplifying collision handling.

1 Introduction

Two-way solid-fluid coupling produces visually and mechanically distinctive behaviors such as a ball pushing smoke and water away while it simultaneously deforms under the fluid's load. The correct handling of this coupling also leads to realistic behavior such as a bowling ball falling more quickly than a feather due to drag forces. Thus, methods for simulating complex scenes that couple solids and fluids can be critical for generating compelling visual effects and accurately simulating the world around us.

Solid simulation, particularly hyperelastic solids, predominantly uses a Lagrangian representation [Irving et al. 2007; Wang et al. 2010; Stomakhin et al. 2012; Sin et al. 2013; Bouaziz et al. 2014], while single-phase fluids such as smoke are often simulated on an Eulerian grid [Mullen et al. 2009; Zhu et al. 2013; Zhang et al. 2015]. As a result, a variety of methods have been developed that attempt to couple these two disparate representations [Chentanez et al. 2006; Robinson-Mosher et al. 2008] using a suite of numerical techniques and geometric operations.

However, the idea of a unified solver, where the underlying geometry is either entirely Lagrangian or entirely Eulerian, is an appealing one. It removes the need to negotiate between different coordinate systems, and promises to simplify both the design and implementation of the overall algorithms. To date, most attempts at such a solver have been Lagrangian [Solenthaler et al. 2007; Akinci et al. 2013; Clausen et al. 2013; Macklin et al. 2014], because SPH-like [Macklin and Müller 2013; Ihmsen et al. 2014] and FLIP-like [Jiang et al. 2015] methods can be used to make the fluid representation Lagrangian as well.

In this work, we take the opposing perspective and explore the coupling of fully Eulerian solids and fluids. This is made possible by the recent work of Levin et al. [2011], which presents a method for simulating hyperelastic solids within an Eulerian framework. By incorporating this method into an Eulerian fluid solver, we are able to resolve complex contact scenarios between multiple solids and a single-phase fluid (Fig. 1). In order to enable large timesteps in the presence of difficult contact configurations, we present a semi-implicit method for stepping the system forward in time. As in the cases of cloth and hair [Sifakis et al. 2008; McAdams et al. 2009], we empirically observe that the incompressibility of the fluid naturally assists in collision handling. The final algorithm does not require the generation or maintenance (i.e. remeshing) of a well-conditioned tetrahedral mesh, and maintains the robustness and ease of use of a purely Eulerian technique. Our method is able to simulate complex collision scenarios between solids and fluids that we have not seen from any previous approach.

Our work makes the following technical contributions:

- A unified, Eulerian framework for simulating fully coupled fluids and deformable solids
- A semi-implicit solver for Eulerian Solids
- A method for satisfying incompressibility for both the solid and fluid regions of the simulation
- A collision resolution scheme for multibody frictional contact

SA '16 Technical Papers, December 05 - 08, 2016, , Macao

ACM 978-1-4503-4514-9/16/12$15.00
DOI: http://dx.doi.org/10.1145/2980179.2980229

2 Related Work

Beginning with the immersed boundary method [Peskin 1972], simulating the coupled motion of solids and fluids has a long history in both graphics and engineering. In graphics, there has been much work coupling fluids to rigid bodies [Takahashi et al. 2002; Carlson et al. 2004; Klingner et al. 2006; Batty et al. 2007], as well as rigid and deformable shells [Guendelman et al. 2005].

Chentanez et al. [2006] modeled the deformable solid as an unstructured tetrahedral mesh and showed how to couple it to an Eulerian fluid, which could be represented as either a regular grid or another unstructured mesh. This approach requires a mesh generation stage, and the specific formulation required an asymmetric system to be solved. This approach has been extended to fully Lagrangian simulations of both the solid and fluid [He et al. 2012; Souli and Benson 2013; Wick 2013], and has incorporated additional phenomena such as phase transitions [Clausen et al. 2013] and porous flow [Lenaerts et al. 2008]. Other methods have further investigated Eulerian fluid discretizations and used sophisticated geometric operations [Robinson-Mosher et al. 2008; Robinson-Mosher et al. 2009] as well as overlapping grids [Baaijens 2001] to couple the grid velocities to a Lagrangian solid. Fast, approximate, position-based methods have also been recently developed for real-time applications [Macklin et al. 2014], which can often need careful parameter tuning to generate realistic results.

Recently, the Material Point Method (MPM) [Stomakhin et al. 2014; Jiang et al. 2015] has become popular for simulating a number of mixed-phase phenomena. It shares some of the same advantages of our approach, as it avoids the need for complex remeshing schemes and geometric conversions. However, as mentioned by Jiang et al. [2016], these schemes are known to have issues representing hyperelastic materials, as artificial plasticity can creep into the simulation. Our scheme naturally handles hyperelastic response, even in the presence of a fluid, and still allows the user to add plasticity if desired.

In order to avoid complicated meshing schemes, simulate elastic objects accurately, and robustly resolve complicated collisions, Levin et al. developed the Eulerian Solids methodology [2011]. With this technique in hand, it is natural to ask whether we can now perform solid-fluid coupling in a purely Eulerian fashion. The closest work to ours in the engineering literature is Kamrin et al. [2012], which showed that a similar "reference map" method can be used to couple deformable, elastic solids to weakly compressible fluids. The approach has also been extended to handle non-frictional contact between two objects [Valkov et al. 2015]. However, this method does not handle incompressible fluids, large time steps or complex contacts between a multitude of objects. Crucially, their efficacy has also only been demonstrated on coarse 2D grids. In contrast, we present a fully 3D Eulerian Solids-based solver that couples an incompressible fluid to multiple deformable objects undergoing frictional contact. By using an implicit time integration scheme, we are able to take large timesteps.

3 Eulerian Solid and Fluid Preliminaries

Notation: We will denote vectors using bold lowercase, e.g. $\mathbf{f}$, and matrices using bold uppercase, e.g. $\mathbf{M}$. Unbolded symbols represent scalars. Departing from the usual fluid simulation notation, we use $\mathbf{u}$ to represent the displacement field of a solid object and instead use $\mathbf{v}$ for velocity. A superscript to the left of a variable is used to distinguish solid objects from fluid. Unlabelled vectors are considered global, i.e. they contain both solid and fluid entries. A superscript $\star$ denotes intermediate states prior to advection, and an overbar, e.g. $\bar{\mathbf{x}}$ denotes the reference configuration of variable $\mathbf{x}$.

Eulerian Solids: In the interest of self-containment, we will give a brief overview of Eulerian Solids, but full details can be found in previous work [Levin et al. 2011; Fan et al. 2013]. The first step of any continuum simulation of materials is to discretize the deformation mapping, $\phi : \bar{\mathbf{x}} \rightarrow \mathbf{x}$ from material space to physical space. Eulerian Solids discretize the physical space as a regular grid and store the material space coordinates as a field to be advected. Rather than store the full material coordinates, we follow Fan et al. [2013], and advect the displacement field $\mathbf{u}$, which improves the robustness of the method under large time steps.

The advected $\mathbf{u}$ is the key to the Eulerian solids approach, as it allows the direct computation of the deformation gradient $\mathbf{F}$ using

$$\mathbf{F} = \left(\mathbf{I} - \frac{\partial \mathbf{u}}{\partial \mathbf{x}}\right)^{-1}. \tag{1}$$

With this $\mathbf{F}$, we can compute the forces inside a solid using any arbitrary constitutive model. Crucially, only displacement fields with zero deformation can yield $\mathbf{F} = \mathbf{I}$. This guarantees that an elastic constitutive model will always generate forces that attempt to return to a zero deformation state, and ensure an accurate hyperelastic simulation.

Eulerian Fluids: For completeness, we also give a brief overview of fluids, but more details can be found in Bridson [2008]. The equations for an inviscid, incompressible fluid are:

$$\begin{aligned} \frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} + \frac{1}{\rho} \nabla p = \mathbf{g}, \\ \nabla \cdot \mathbf{v} = 0. \end{aligned} \tag{2}$$

The first equation is the momentum equation and the second is the incompressibility constraint. Here, $\mathbf{v}$ is the velocity of the fluid, ρ it the density, p the pressure, and $\mathbf{g}$ denotes external forces such as gravity.

Approximating the derivatives using Eulerian finite differences is straightforward, which makes them a popular method for simulating fluids. A basic solver is as follows:

$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} = 0, \tag{3}$$

$$\frac{\partial \mathbf{v}}{\partial t} = \mathbf{g}, \tag{4}$$

$$\frac{\partial \mathbf{v}}{\partial t} + \frac{1}{\rho} \nabla p = 0 \quad \text{such that} \quad \nabla \cdot \mathbf{v} = 0. \tag{5}$$

Eqn. 4 adds the body forces and is often integrated explicitly. Eqn. 5 is usually solved using a Helmholtz-Hodge decomposition that projects out the divergent component of the velocity, and typically requires the solution of a Poisson problem.

The advection step, Eqn. 3, plays a key role in the stability of the simulation. Semi-Lagrangian [Stam 1999] and Fluid-Implicit-Particle (FLIP) [Brackbill and Ruppel 1986; Jiang et al. 2015] methods are two widely used schemes. The former is performed entirely on the Eulerian grid while the latter relies on auxiliary Lagrangian particles.

4 Coupled Solid–Fluid Simulation

4.1 The Continuous Formulation

In this work we focus on the coupled simulation of multiple incompressible, hyper-elastic, and elasto-plastic solids immersed in

an incompressible fluid. This requires us to solve the momentum equation, given by

$$\left.\begin{aligned} \rho\frac{d\mathbf{v}}{dt} &= \nabla\cdot\sigma+\mathbf{f} \\ \nabla\cdot\mathbf{v} &= \mathbf{0} \end{aligned}\right\} \forall \mathbf{x}\in\Omega \qquad {}^{f}\mathbf{v} = {}^{s}\mathbf{v} \quad \forall \mathbf{x}\in\Gamma \tag{6}$$

where Ω denotes a region in world space, $\mathbf{x}$ denotes a point in world space, $\mathbf{v}$ is the velocity of a particle at $\mathbf{x}$, σ is the Cauchy stress and $\mathbf{f}$ are external forces such as gravity. For each $\mathbf{x}$ containing a solid, we compute σ using a standard hyper-elastic or elasto-plastic constitutive model, and for each $\mathbf{x}$ containing a fluid we set $\sigma = \mathbf{0}$. For these cells, the divergence-free condition, $\nabla\cdot\mathbf{v} = \mathbf{0}$, is sufficient. We also enforce a no slip condition along the solid-fluid boundary Γ, where fluid and solid velocities are respectively denoted ${}^{f}\mathbf{v}$ and ${}^{s}\mathbf{v}$. We use the hyper-elastic model of McAdams et al. [2011] for the elastic component of all of our examples.

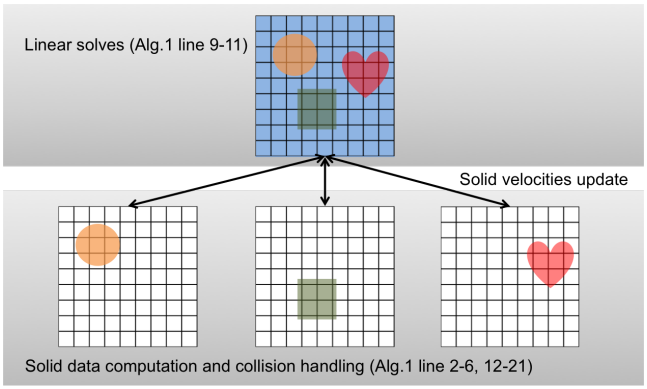


Figure 2: *The high-level structure of our data storage and computation scheme. To assist advection and contact handling, we keep separate velocity fields for each solid object. For efficiency, only values near the solid are updated.*

4.2 Spatial Discretization and Constraints

Our method relies on fixed discretizations of both $\bar{\mathbf{x}}$ and $\mathbf{x}$. In order to solve Eqn. 6, we discretize Ω using regular, hexahedral finite elements. Velocity, displacement and forces are co-located at the grid nodes, while pressure values is stored at grid centers. In order to incorporate equality constraints we rely on a mixed formulation in which incompressibility is applied as a point constraint at the cell center. This can be considered an under-integrated finite element, which is commonly used to prevent locking [Belytschko et al. 2013]. We then compute per-element mass and stiffness matrices, based on whether each cell contains a solid or a fluid, using an eight-point quadrature rule, and then assemble into global $\mathbf{M}$ and $\mathbf{K}$ operators. Our discretized divergence-free constraint is expressed as $\mathbf{J}\mathbf{v} = \mathbf{0}$ where $\mathbf{J}$ is the constant constraint gradient. Note that due to the continuity of the velocity field, the no slip condition on solid-fluid boundaries is implicitly enforced, and no special spatial coupling terms need to be formulated.

To facilitate velocity advection and collision detection, each solid stores a copy of the velocity field, but only values near the solid are ever updated. Fig. 2 shows our high level data storage and computation structure. Our algorithm also requires a discretization of $\bar{\mathbf{x}}$ if plastic deformation is desired. For each plastic solid, we create an auxiliary grid of that solid's reference coordinate system ${}^{l}\bar{\mathbf{x}}$, where $l \in [1, N_s]$ indexes each solid in the simulation.

4.3 Time Integration

We use a splitting scheme to advance our system in time. First, we use implicit integration to compute a divergence-free velocity field for the solid cells, and then perform an advection that resolves collisions. Algorithm 1 gives an overview of our time integration scheme. In the next sections, we describe the key components of our algorithm: a semi-implicit update for Eulerian Solids, pressure projection, and a collision resolution scheme.

Algorithm 1 Eulerian solids and fluids simulation

1: Compute Δt based on CFL condition
2: **for** each solid object, l **do**
3: Compute mass ${}^{l}\mathbf{M}$ and volume fraction ${}^{l}\mathbf{V}$ on the grid
4: Compute material force ${}^{l}\mathbf{f}$ and stiffness matrix ${}^{l}\mathbf{K}$
5: ▷ (§4.4)
6: **end for**
7: Compute fluid mass ${}^{f}\mathbf{M}$
8: Assemble $\mathbf{M}$, $\mathbf{K}$, $\mathbf{C}$ and $\mathbf{f}^{\star}$ ▷ (§4.4)
9: $\mathbf{v}^{\star} = \mathbf{G}^{-1}\mathbf{f}^{\star}$ ▷ (Eqn. 8)
10: Compute pressure $\mathbf{p}$ ▷ (Eqn. 12)
11: Pressure project $\mathbf{v}^{\star}$ to get pre-advection $\mathbf{v}^{n+1}$ ▷ (Eqn. 13)
12: **for** each solid object, l **do**
13: Update solid particle velocities using FLIP from ${}^{l}\mathbf{v}^{n+1}$
14: Add repulsions and frictions to particles in collision
15: ▷ (§4.6)
16: **end for**
17: **for** each solid object, l **do**
18: Rasterize particle velocities to get final velocity ${}^{l}\mathbf{v}^{n+1}$
19: Update displacement ${}^{l}\mathbf{u} = {}^{l}\mathbf{u}^{n} + \Delta t\,{}^{l}\mathbf{v}^{n+1}$
20: Semi-Lagrangian advect ${}^{l}\mathbf{u}$ to get ${}^{l}\mathbf{u}^{n+1}$
21: **end for**
22: Semi-Lagrangian advect fluid velocity ${}^{f}\mathbf{v}^{n+1}$

4.4 Semi-implicit Update

The original Eulerian Solids scheme [Levin et al. 2011] used explicit force integration to compute the velocity field, followed by a first-order finite difference scheme for advection. Both of these design decisions resulted in small time steps. While Fan et al. [2013] introduced Lagrangian modes on top of the Eulerian motion in order to reduce this restriction, we seek to ease it in a way that maintains the convenience of a single spatial discretization. First, we replace the explicit force integration with a semi-implicit scheme that is similar to that of Stomakhin et al. [2013].

We denote the change in velocity at each grid node, due to internal forces $\mathbf{f}_{\text{int}}$, over the time interval $[t, t+\Delta t]$ as:

$$\mathbf{v}^{n+1} = \mathbf{v}^{n} + \Delta t\ \mathbf{f}_{\text{int}}\left(\mathbf{u}^{n+1}\right).$$

A standard Taylor expansion around $\mathbf{x}$ yields,

$$\mathbf{f}(\mathbf{u}^{n+1}) = \mathbf{f}_{\text{int}}(\mathbf{u}^{n} + \Delta t\mathbf{v}^{n+1}) \approx \mathbf{f}^{n}_{\text{int}} + \frac{\partial \mathbf{f}^{n}_{\text{int}}}{\partial \mathbf{x}}\Delta t\ \mathbf{v}^{n+1}, \tag{7}$$

which we can further abbreviate to $\mathbf{f}(\mathbf{u}^{n+1}) = \mathbf{f}^{n}_{\text{int}} + \mathbf{K}\Delta t\mathbf{v}^{n+1}$. By combining this with a first order discretization of acceleration, $\mathbf{a}^{\star} = (\mathbf{v}^{n+1} - \mathbf{v}^{n})/\Delta t$, and the equations of motion for a deformable solid, $\mathbf{M}\mathbf{a}^{\star} + \mathbf{C}\mathbf{v}^{\star} + \mathbf{f}^{\star}_{\text{int}} = \mathbf{f}_{\text{ext}}$, we obtain the semi-implicit update equation:

$$(\mathbf{M} + \Delta t\mathbf{C} + \Delta t^{2}\mathbf{K})\mathbf{v}^{\star} = \mathbf{M}\mathbf{v}^{n} + \Delta t(\mathbf{f}_{\text{ext}} - \mathbf{f}^{n}_{\text{int}}) \tag{8}$$

Here, $\mathbf{C}$ is a Rayleigh damping matrix. The external force term $\mathbf{f}_{\text{ext}}$ includes body, buoyancy and vorticity forces. For fluid-only cells,

$\mathbf{C}$ and $\mathbf{K}$ disappear and only the diagonal mass matrix $\mathbf{M}$ remains. Therefore, the system can be solved efficiently if the simulation domain is dominated by a fluid.

Note that there is no advective term in the stiffness matrix. In a purely Eulerian sense, the force arises from a chain of variables: $\mathbf{f}_{\text{int}}\left(\mathbf{F}\left(\mathbf{u}(\mathbf{x}(t),t)\right)\right)$. The derivative should then be:

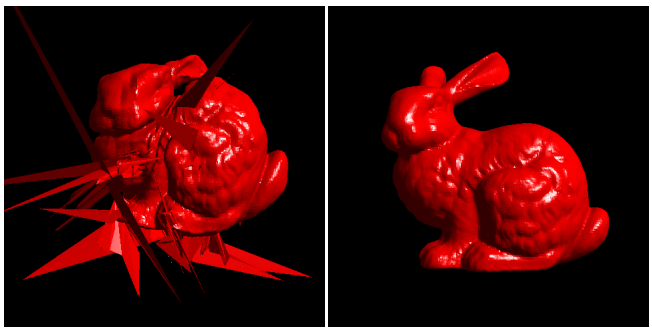
$$\frac{\partial \mathbf{f}_{\text{int}}\left(\mathbf{F}\left(\mathbf{u}(\mathbf{x}(t),t)\right)\right)}{\partial t} = \frac{\partial \mathbf{f}_{\text{int}}}{\partial \mathbf{F}}\frac{\partial \mathbf{F}}{\partial \mathbf{u}}\left(\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial t} + \frac{\partial \mathbf{u}}{\partial t}\right). \quad (9)$$

The $\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial t} = \mathbf{v}\cdot\nabla\mathbf{u}$ appears to introduce an advective term, but we can observe that $\left(\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial t} + \frac{\partial \mathbf{u}}{\partial t}\right) = \frac{D\mathbf{u}}{Dt}$, i.e. the total derivative of $\mathbf{u}$. This then reduces to the Lagrangian case,

$$\frac{\partial \mathbf{f}_{\text{int}}\left(\mathbf{F}\left(\mathbf{u}(t)\right)\right)}{\partial t} = \frac{\partial \mathbf{f}_{\text{int}}}{\partial \mathbf{F}}\frac{\partial \mathbf{F}}{\partial \mathbf{u}}\frac{D\mathbf{u}}{Dt}. \quad (10)$$

Taking a perspective similar to Stomakhin et al. [2013] that the nodes of the Eulerian mesh are fictitiously deforming in a Lagrangian manner, the Lagrangian $\mathbf{K}$ in Eqn. 8 suffices.

Our semi-implicit integration scheme can handle large time steps under severe deformations. In Fig. 3 we initially squished a bunny by half. We did not respect the CFL condition and set $\Delta t = \frac{1}{24}$. Explicit integration blows up almost immediately, while semi-implicit integration correctly returns the bunny to its rest shape.



(a) (b)

Figure 3: *We scale the bunny by half and let it expand. Using $\Delta t = \frac{1}{24}$, explicit integration blew up after 4 frames while our semi-implicit scheme is extremely stable.*

4.5 Incompressibility Constraints

We enforce incompressibility constraints using a primal-dual algorithm. First, we form the Karush-Kuhn-Tucker (KKT) system prescribed by our semi-implicit scheme (§4.4),

$$\begin{bmatrix}\mathbf{G} & \mathbf{J}^T\\ \mathbf{J} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{v}^{n+1}\\ \mathbf{p}\end{bmatrix} = \begin{bmatrix}\mathbf{f}^\star\\ \mathbf{b}\end{bmatrix} \quad (11)$$

where $\mathbf{G} = \mathbf{M} + \Delta t\mathbf{C} + \Delta t^2\mathbf{K}$, $\mathbf{f}^\star = \mathbf{M}\mathbf{v}^n + \Delta t(\mathbf{f}_{\text{ext}} - \mathbf{f}^n_{\text{int}})$ and $\mathbf{b}$ contains the boundary conditions. We first solve $\mathbf{G}\mathbf{v}^\star = \mathbf{f}^\star$ for the unconstrained velocity $\mathbf{v}^\star$ and then solve the dual problem to eliminate the divergent part of the velocity field. We replace $\mathbf{G}^{-1}$ with $\mathbf{M}^{-1}$ in the pressure solve to avoid an expensive matrix inversion:

$$\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\mathbf{p} = \mathbf{J}\mathbf{v}^\star - \mathbf{b}. \quad (12)$$

Finally, we correct $\mathbf{v}^\star$ to get the pre-advection velocity field $\mathbf{v}^{n+1}$:

$$\mathbf{v}^{n+1} = \mathbf{v}^\star - \mathbf{G}^{-1}(\mathbf{J}^T\mathbf{p}). \quad (13)$$

The substitution in Eqn. 12 is can be interpreted in two ways. First, if all the cells contain fluid, the Schur complement in Eqn. 12 naturally yields $\mathbf{M}^{-1}$. Thus, we can interpret this substitution as momentarily approximating the solid cells as fluid. Second, if the solid cells are integrated explicitly, Eqn. 12 again yields $\mathbf{M}^{-1}$, as the material forces still appear on the right hand side. So, we can interpret the substitution as only integrating the *volume* terms implicitly, while treating the solid strain energies explicitly.

We also attempted to solve the KKT system (Eqn. 11) directly, but initial test showed that our primal-dual version ran over $3\times$ faster in 2D. The investigation of more sophisticated solution methods for this problem is left as future work.

Complexity compared to explicit integration: In previous work, [Levin et al. 2011; Fan et al. 2013], two quadratic problems of the same form as Eqn. 11 were solved to determine the time step size. In our formulation, we instead solve three linear systems (Eqs. 8, 12 and 13). In our experiments, we found that the increase in time step size far outweighed the cost of this additional linear solve. Thus, we are able to compute a large, implicit step at a cost that is proportional to a small, explicit step.

4.6 Contact and Collision Response for Solids

As noted in previous work [Sifakis et al. 2008; McAdams et al. 2009], the presence of divergence-free constraints help to maintain a collision-free state. However, some collision handling is still needed to avoid solids from "sticking" if the advection stage introduces overlaps. In order to address this, we apply the repulsion forces of Bridson et al. [2002] during our advection.

For this stage, it is necessary to employ an auxiliary Lagrangian variable. While this seems slightly at odds with the goal of a fully Eulerian simulation, our underlying geometric representation remains Eulerian. Like the Eulerian grid projection stage of the Lagrangian FLIP method [Zhu and Bridson 2005], or the semi-Lagrangian particle traces of grid-based Stable Fluids [Stam 1999], we leverage the advantages of the other coordinate system during time integration without fully commiting to the representation.

Collision resolution begins by copying each solid velocity $\mathbf{v}^{n+1}$ from our spatial grid to the individual solid grids ${}^l\mathbf{v}^{n+1}$, where l indexes each solid in the scene. Next, we instantiate particles for each solid, using 8-16 particles per cell. In order to avoid collisions we check the distance between the initial particles of one solid against all of the other solids. If it is closer than a distance h (typically the grid resolution) the normal velocity of the particle is modified by an impulse r, defined as:

$$r = -\min\left(\Delta t\ k\ d, m\left(\frac{0.1d}{\Delta t} - v_N\right)\right). \quad (14)$$

Here, d is the overlap distance, k is a spring stiffness, m is the mass of the particle and v_N is the relative velocity in the direction of the contact normal. The change of the particle velocity in the normal direction is then defined as $\Delta v_N = r/m$. Friction can also be applied by modifying the relative tangential velocity:

$$\mathbf{v}_T = \max\left(1 - \mu\frac{\Delta v_N}{|\mathbf{v}_T^{\text{pre}}|}, 0\right)\mathbf{v}_T^{\text{pre}}, \quad (15)$$

where $\mathbf{v}_T^{pre}$ is the pre-friction relative tangential velocity. The values of k and μ we used are listed in Table 1.

Computing the contact normal: Each solid object has an embedded surface mesh and a signed distance field ϕ defined in its material domain. The mesh is advected *passively* in the same way as Fan et al. [2013]. When checking for the collision of material

particle s of solid i against solid j, we interpolate the material position field ${}^j\bar{\mathbf{x}}$ and lookup ${}^j\phi$. A repulsion is added if the overlap $d = h - {}^j\phi({}^j\bar{\mathbf{x}}({}^i\mathbf{p}_s)) > 0$. We find the nearest surface element to ${}^j\bar{\mathbf{x}}({}^i\mathbf{p}_s)$ and use its world space normal as the contact normal. If a surface mesh is not available, the gradient of the background volume grid can be used to compute the normal [Levin et al. 2011].

5 Implementation and Results

We solved Equation 8, 12 and 13 using Preconditioned Conjugate Residuals (PCR) with a Jacobi preconditioner because the matrices are semi-definite. Warm starting was used when solving the pressure (Equation 12). We use the Eigen [Guennebaud et al. 2010] library for linear algebra routines. All simulations were run on a 8-core, 3GHz MacPro with 32 GB of RAM using 16 threads. An advantage of Eulerian simulation is that most stages can be embarrassingly parallelized, so OpenMP was used whenever possible. Both our 2D and 3D examples used the co-rotational material from McAdams et al. [2011]. Table 2 shows a performance summary of our 3D examples and Table 1 shows their simulation parameters. We chose a high contact spring stiffness for more bouncy contact and lower value for dampened contact. We used a PCR threshold of 0.02 for the pressure solve (Eqn. 12) in all the examples. For the two velocity solves (Eqs. 8 and 13) we used a PCR threshold of $1e^{-4}$ for Cheb and $1e^{-3}$ for the others.

5.1 Simulating a Single Solid with a Fluid

In all of the following examples, we found that the linear solves, particularly the pressure solve, consumed the largest fraction of the running time (Table 2). The collision assistance provided by the divergence-free constraint can also be seen in the timings, as very little time needs to be spent in collision resolution.

Elasticity validation: Fig. 4(a) shows two jets compressing an elastic circle. External forces, including fluid forces, are then removed and the simulation of the circle continues in isolation. The circle correctly returns to its reference configuration, which is a hallmark of the Eulerian Solids approach (Fig. 4(b)).

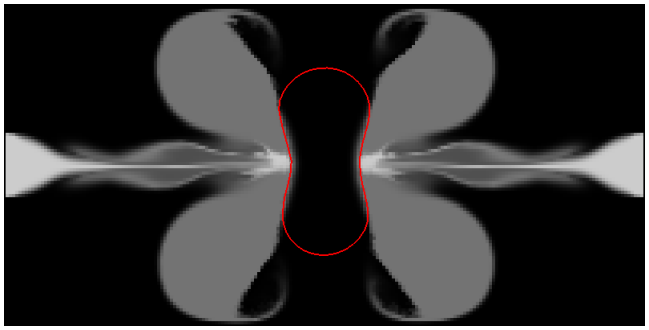
Ball: We dropped a heavy elastic ball to a carpet of smoke. The only external forces applied to the fluids are gravity and vorticity confinement of Fedkiw et al. [2001]. The smoke gets pushed away when the ball hits the flow. As the ball rebounds, a plume of smoke is drawn up by the low-pressure eddy formed in its wake. The simulation is inside a box with Dirichlet boundary conditions, so four vortices form along the four quadrants of the x-z plane (Fig. 5).

Buoyancy: Figure 6 illustrates that solids with different densities behave differently when interacting with the fluid. The heavy bunny falls at the rate of gravity and the light bunny falls much slower due to drag forces. A buoyant jet is used to push the light bunny to the ceiling while the heavy bunny remains on the ground.

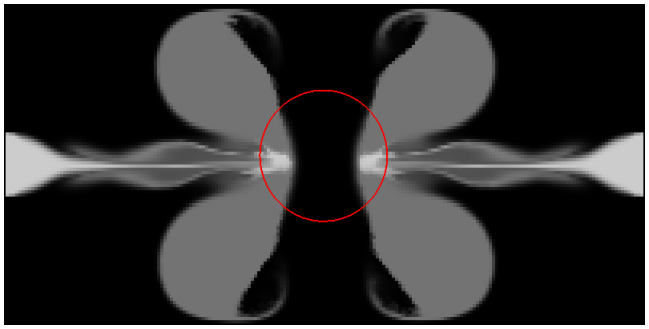
Cheb's Glaucoma Test: For plastic deformation, we use a multiplicative plasticity model [Bargteil et al. 2007]. The plastic deformation gradients are stored in a material space grid, initially set to identity, and updated according to a yield condition. For more details, please refer to §3.7 of Fan et al. [2013]. We take Cheb to an eye exam where a high pressure puff of air is shot at his head. Figure 7(c) and 7(d) shows the comparison between elastic and plastic deformation. In the supplemental video, we show that by varying whether the feet are constrained, different motions are obtained.

5.2 Simulating Multiple Solids with a Fluid

Collision Response: In the supplement video we show two 2D circles moving towards each other. By adding collision response to the solid particles, the circles are able to separate after collision. Without this response, the circles stick.



(a)



(b)

Figure 4: *(a) A elastic circle deforms under the influence of two jets. (b) We removed all external forces and simulated the circle by itself. The circle is able to return to its rest shape.*

Example	Solid-fluid density ratios	k spring stiffness	μ friction coefficient
Ball	1000:1	10000	0.5
Bunny (light)	1.3:1	2000	0.2
Bunny (heavy)	1000:1	2000	0.5
Cheb	1.3:1	2000	0.5
Party	1000:1, 500:1	10000	0.2

Table 1: *Simulation parameters used for each example.*

"Party": We simulate multiple solids with various densities and material parameters interacting with each other and the fluid (Figure 1). Figure 9 shows the time steps used throughout the simulation. We use a CFL number of 0.6 when two solids are less than 2 grid cells apart and 1.8 otherwise. Some small time steps occur after second 6 in order to complete the current render frame, not due to the CFL number. Smoke and side walls are not rendered so that the solid and fluid motions can be seen more clearly. Figure 8 shows the final shape of each solid. Note the extreme deformations such as the plastic duck's wing being compressed to be flush with its body.

Due to the presence of multiple objects, the timing breakdown differs from that of the other examples. More time is spent in advection and constructing $\mathbf{G}$. This is because a $\mathbf{K}$ matrix must be constructed for each solid, and each must also be converted to FLIP particles and then re-rasterized to the grid. Our current implementation parallelizes these operations internally for each solid object; it does not create a separate thread for each solid. Therefore, many opportunities for further accelerating these operations still remain.

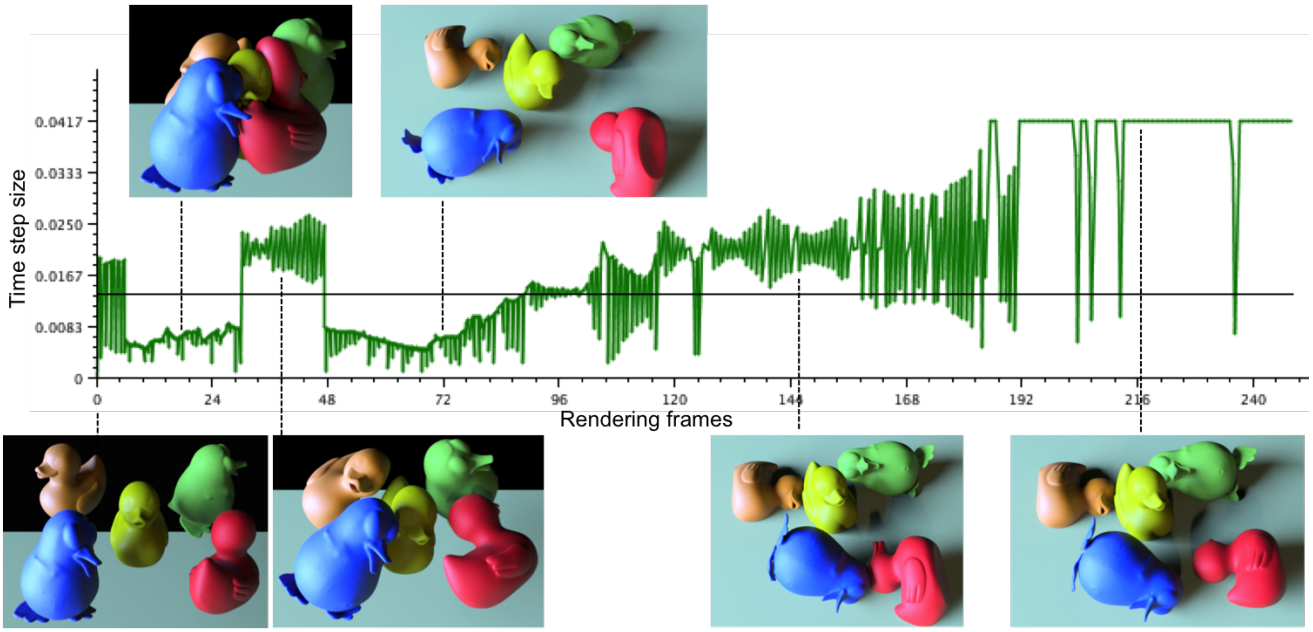


Figure 9: *Simulation timestep sizes as the party progresses. The rendering rate is fixed at $\Delta t = \frac{1}{24}$ throughout. The horizontal black line denote the average timestep during the simulation. Frames from events corresponding to significant changes in the timestep size are shown. From left to right: the solids are given an initial impulse, they collide and form complex contacts, they fall to the floor, bounce on the floor, and finally come to rest.*

Example	Grid Dimensions	Avg. timestep	Min. timestep	Avg. time/frame	Compute $\mathbf{G}$ (line 8)	Velocity solves (lines 9, 11)	Pressure solve (line 10)	Advection (lines 13, 18)	Collision (line 14)
Ball	$120 \times 140 \times 120$	0.025	0.0016	6.54s	0.84s	0.12s	2.83s	1.12s	0.01s
Light Bunny	$144 \times 200 \times 144$	0.0369	0.0064	17.4s	1.32s	5.02s	6.21s	1.72s	0.016s
Heavy Bunny	$144 \times 200 \times 144$	0.0236	0.00058	16.9s	1.35s	2.47s	8.31s	1.68s	0.016s
Cheb	$120 \times 100 \times 80$	0.021	0.0023	6.97s	1.55s	1.10s	1.10s	1.37s	0.01s
Party	$200 \times 180 \times 200$	0.0133	0.00083	54.1s	13.6s	6.72s	12.1s	14.46s	0.25s

Table 2: *For each example, the size of the spatial grid, average / minimum simulation time step sizes and average per-frame simulation times are reported. We also list the computation times of key stages of Algorithm 1. All timings are reported in seconds.*

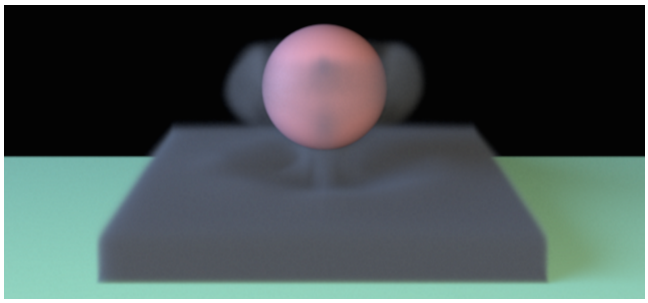


Figure 5: *A plume rises as an elastically deforming ball bounces up from a smoky floor.*

6 Discussion and Future Work

In this paper we have shown how to simulate a two-way coupling between solids and a fluid where the underlying representation is entirely Eulerian. This allows us to generate simulations that feature large deformations and frictional contact, all the while capturing visually interesting fluid effects. We believe our method produces examples that are more complex than previous approaches and avoids the complexities of Lagrangian, mesh-based simulation.

Despite the success of our method, it has several limitations, the most obvious of which is that the fluid is limited to a single phase. Extending this approach include high-quality, fully Eulerian liquid simulations [Heo and Ko 2010] is an interesting direction for future work. As the technique is Eulerian, handling features that are smaller than a single grid cell, e.g. rods and thin shells [Guendelman et al. 2005], remains a challenge. This same difficulty extends to representing detailed fracture patterns though extended finite element approaches [Kaufmann et al. 2009] offer a potential solution.

Furthermore, while we have shown that an implicit integration scheme can be effective when simulating this coupling problem, other schemes that enable even larger timesteps [Lentine et al. 2012], or preserve more structure given the same timestep [Mullen et al. 2009], would be welcome additions to this work. Finally, it remains to be seen whether a combination of Eulerian-on-Lagrangian [Fan et al. 2013] and subspace methods [Kim and Delaney 2013; Liu et al. 2015] could be used to accelerate the overall simulation.

7 Acknowledgements

This work was supported by a National Science Foundation CAREER award (IIS-1253948). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We acknowledge support from the Center for Scientific Computing from the CNSI, MRL: an NSF MRSEC (DMR-

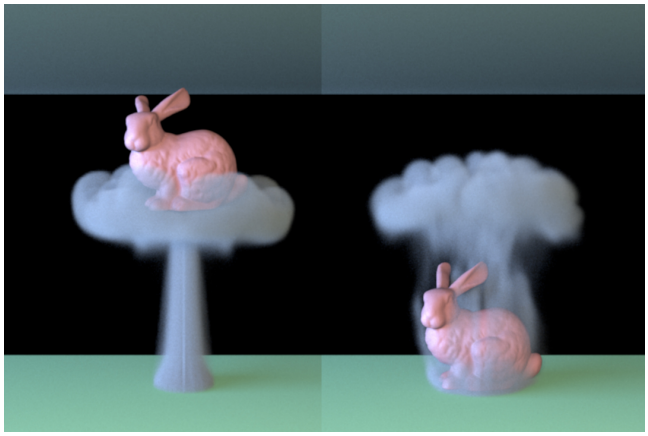


Figure 6: *Different solid densities behave differently under buoyant flow. When the solid-fluid density ratio is 1.3:1 (left), the smoke plumes causes the bunny to rise like a balloon. When the ratio is 1000:1 (right), the bunny drops like a rock.*

1121053) and Hewlett Packard.

References

AKINCI, N., CORNELIS, J., AKINCI, G., AND TESCHNER, M. 2013. Coupling elastic solids with smoothed particle hydrodynamics fluids. *Comp. Anim. and Virtual Worlds*, 195–203.

BAAIJENS, F. P. T. 2001. A fictitious domain mortar element method for fluid/structure interaction. *International Journal for Numerical Methods in Fluids 35*, 7, 743–761.

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

BATTY, C., BERTAILS, F., AND BRIDSON, R. 2007. A fast variational framework for accurate solid-fluid coupling. In *ACM Trans. Graph.*, vol. 26, ACM, 100.

BELYTSCHKO, T., LIU, W. K., MORAN, B., AND ELKHODARY, K. 2013. *Nonlinear finite elements for continua and structures*. John Wiley & Sons.

BOUAZIZ, S., MARTIN, S., LIU, T., KAVAN, L., AND PAULY, M. 2014. Projective dynamics: Fusing constraint projections for fast simulation. *ACM Trans. Graph. 33*, 4 (July), 154:1–154:11.

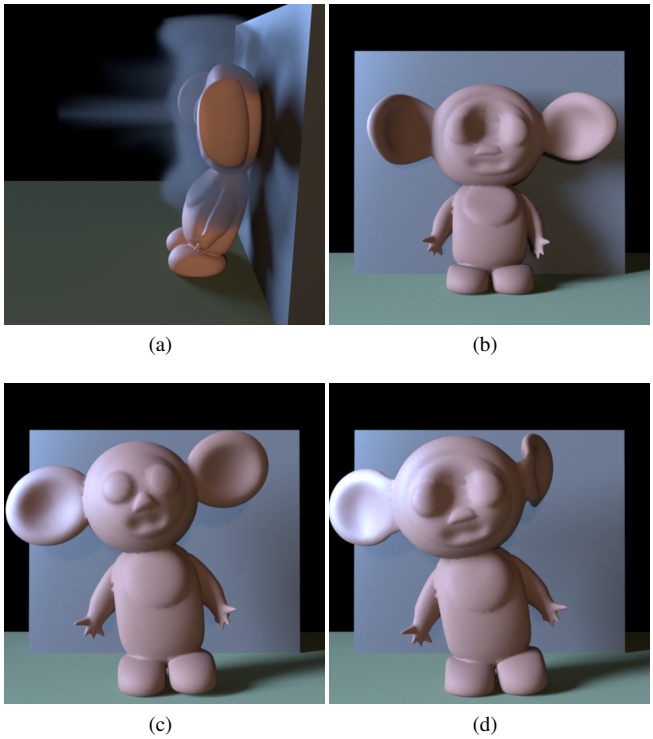
BRACKBILL, J., AND RUPPEL, H. 1986. Flip: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *J. of Comp. Phys. 65*, 2, 314–343.

BRIDSON, R., FEDKIW, R., AND ANDERSON, J. 2002. Robust treatment of collisions, contact and friction for cloth animation. In *ACM Trans. Graph.*, 594–603.

BRIDSON, R. 2008. *Fluid Simulation for Computer Graphics*. AK Peters.

CARLSON, M., MUCHA, P. J., AND TURK, G. 2004. Rigid fluid: Animating the interplay between rigid bodies and fluid. *ACM Trans. Graph. 23*, 3 (Aug.), 377–384.

CHENTANEZ, N., GOKTEKIN, T. G., FELDMAN, B. E., AND O'BRIEN, J. F. 2006. Simultaneous coupling of fluids and deformable bodies. In *ACM SIGGRAPH/Eurographics Symp. Comp. Anim.*, 83–89.



(a) (b) (c) (d)

Figure 7: *(a) Cheb has a high pressure puff of air shot at his head. (b) Same frame, viewed from the front, with smoke removed to make the deformation more visible. (c) Final frame, elastic deformation. (d) Final frame, plastic deformation. Note that the dent in his head persists, as well as the deformations to his ears.*

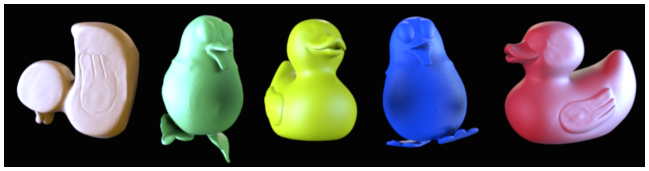


Figure 8: *The final shapes of party members. The duck and penguin on the left are plastic while the other three are elastic.*

CLAUSEN, P., WICKE, M., SHEWCHUK, J. R., AND O'BRIEN, J. F. 2013. Simulating liquids and solid-liquid interactions with lagrangian meshes. *ACM Trans. Graph.*, 17:1–15.

FAN, Y., LITVEN, J., LEVIN, D. I., AND PAI, D. K. 2013. Eulerian-on-lagrangian simulation. *ACM Trans. Graph.*.

FEDKIW, R., STAM, J., AND JENSEN, H. W. 2001. Visual simulation of smoke. In *Proceedings of SIGGRAPH*, 15–22.

GUENDELMAN, E., SELLE, A., LOSASSO, F., AND FEDKIW, R. 2005. Coupling water and smoke to thin deformable and rigid shells. *ACM Trans. Graph. 24*, 3 (July), 973–981.

GUENNEBAUD, G., JACOB, B., ET AL., 2010. Eigen v3. http://eigen.tuxfamily.org.

HE, X., LIU, N., WANG, G., ZHANG, F., LI, S., SHAO, S., AND WANG, H. 2012. Staggered meshless solid-fluid coupling. *ACM Trans. Graph. 31*, 6 (Nov.), 149:1–149:12.

HEO, N., AND KO, H.-S. 2010. Detail-preserving fully-eulerian interface tracking framework. *ACM Trans. Graph. 29*, 6 (Dec.).

IHMSEN, M., ORTHMANN, J., SOLENTHALER, B., KOLB, A., AND TESCHNER, M. 2014. SPH Fluids in Computer Graphics. In *Eurographics State of the Art Reports*.

IRVING, G., SCHROEDER, C., AND FEDKIW, R. 2007. Volume conserving finite element simulations of deformable models. In *ACM Trans. Graph.*, vol. 26.

JIANG, C., SCHROEDER, C., SELLE, A., TERAN, J., AND STOMAKHIN, A. 2015. The affine particle-in-cell method. *ACM Trans. Graph. 34*, 4 (July), 51:1–51:10.

JIANG, C., SCHROEDER, C., TERAN, J., STOMAKHIN, A., AND SELLE, A. 2016. The material point method for simulating continuum materials. In *ACM SIGGRAPH Courses*.

KAMRIN, K., RYCROFT, C. H., AND NAVE, J.-C. 2012. Reference map technique for finite-strain elasticity and fluid–solid interaction. *Journal of the Mechanics and Physics of Solids 60*, 11 (Nov.), 1952–1969.

KAUFMANN, P., MARTIN, S., BOTSCH, M., GRINSPUN, E., AND GROSS, M. 2009. Enrichment textures for detailed cutting of shells. *ACM Trans. Graph. 28*, 3 (July), 50:1–50:10.

KIM, T., AND DELANEY, J. 2013. Subspace fluid re-simulation. *ACM Trans. Graph. 32*, 4 (July), 62:1–62:9.

KLINGNER, B. M., FELDMAN, B. E., CHENTANEZ, N., AND O'BRIEN, J. F. 2006. Fluid animation with dynamic meshes. *ACM Trans. Graph. 25*, 3 (July), 820–825.

LENAERTS, T., ADAMS, B., AND DUTRÉ, P. 2008. Porous flow in particle-based fluid simulations. In *ACM Trans. Graph.*, vol. 27, 49:1–49:8.

LENTINE, M., CONG, M., PATKAR, S., AND FEDKIW, R. 2012. Simulating free surface flow with very large time steps. In *ACM SIGGRAPH/Eurographics Symp. Comp. Anim.*, 107–116.

LEVIN, D. I., LITVEN, J., JONES, G. L., SUEDA, S., AND PAI, D. K. 2011. Eulerian solid simulation with contact. In *ACM Trans. Graph.*, vol. 30.

LIU, B., MASON, G., HODGSON, J., TONG, Y., AND DESBRUN, M. 2015. Model-reduced variational fluid simulation. *ACM Trans. Graph. 34*, 6 (Oct.), 244:1–244:12.

MACKLIN, M., AND MÜLLER, M. 2013. Position based fluids. *ACM Trans. Graph. 32*, 4, 104.

MACKLIN, M., MÜLLER, M., CHENTANEZ, N., AND KIM, T.-Y. 2014. Unified particle physics for real-time applications. *ACM Trans. Graph. 33*, 4 (July), 153:1–153:12.

MCADAMS, A., SELLE, A., WARD, K., SIFAKIS, E., AND TERAN, J. 2009. Detail preserving continuum simulation of straight hair. *ACM Trans. Graph. 28*, 3, 62:1–62:6.

MCADAMS, A., ZHU, Y., SELLE, A., EMPEY, M., TAMSTORF, R., TERAN, J., AND SIFAKIS, E. 2011. Efficient elasticity for character skinning with contact and collisions. *ACM Trans. Graph. 30*, 4 (July), 37:1–37:12.

MULLEN, P., CRANE, K., PAVLOV, D., TONG, Y., AND DESBRUN, M. 2009. Energy-preserving integrators for fluid animation. *ACM Trans. Graph. 28*, 3 (July), 38:1–38:8.

MÜLLER, M., AND CHENTANEZ, N. 2011. Solid simulation with oriented particles. *ACM Trans. Graph. 30*, 4 (July), 92:1–92:10.

PESKIN, C. S. 1972. The immersed boundary method. *Acta Numerica 11* (1), 479–517.

ROBINSON-MOSHER, A., SHINAR, T., GRETARSSON, J., SU, J., AND FEDKIW, R. 2008. Two-way coupling of fluids to rigid and deformable solids and shells. *ACM Trans. Graph. 27*, 3.

ROBINSON-MOSHER, A., ENGLISH, R. E., AND FEDKIW, R. 2009. Accurate tangential velocities for solid fluid coupling. In *ACM SIGGRAPH/Eurographics Symp. Comp. Anim.*, 227–236.

SELLE, A., FEDKIW, R., KIM, B., LIU, Y., AND ROSSIGNAC, J. 2008. An unconditionally stable maccormack method. *J. Sci. Comput. 35*, 2-3 (June), 350–371.

SIFAKIS, E., MARINO, S., AND TERAN, J. 2008. Globally coupled collision handling using volume preserving impulses. In *ACM SIGGRAPH/Eurographics Symp. Comp. Anim.*, 147–153.

SIN, F. S., SCHROEDER, D., AND BARBIC, J. 2013. Vega: Nonlinear fem deformable object simulator. *Computer Graphics Forum 32*, 1, 36–48.

SOLENTHALER, B., SCHLÄFLI, J., AND PAJAROLA, R. 2007. A unified particle model for fluid–solid interactions. *Computer Animation and Virtual Worlds 18*, 1, 69–82.

SOULI, M., AND BENSON, D. J. 2013. *Arbitrary Lagrangian Eulerian and Fluid-Structure Interaction: Numerical Simulation*. John Wiley & Sons.

STAM, J. 1999. Stable fluids. In *Proceedings of SIGGRAPH*, 121–128.

STOMAKHIN, A., HOWES, R., SCHROEDER, C., AND TERAN, J. M. 2012. Energetically consistent invertible elasticity. In *ACM SIGGRAPH/Eurographics Symp. Comp. Anim.*, 25–32.

STOMAKHIN, A., SCHROEDER, C., CHAI, L., TERAN, J., AND SELLE, A. 2013. A material point method for snow simulation. *ACM Transactions on Graphics (TOG) 32*, 4, 102.

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 (July), 138:1–138:11.

TAKAHASHI, T., UEKI, H., KUNIMATSU, A., AND FUJII, H. 2002. The simulation of fluid-rigid body interaction. In *ACM SIGGRAPH Abstracts and Applications*, 266–266.

VALKOV, B., RYCROFT, C. H., AND KAMRIN, K. 2015. Eulerian method for multiphase interactions of soft solid bodies in fluids. *Journal of Applied Mechanics 82*, 4.

WANG, H., O'BRIEN, J., AND RAMAMOORTHI, R. 2010. Multi-resolution isotropic strain limiting. *ACM Transactions on Graphics 29*, 6, 156:1–156:10.

WICK, T. 2013. Coupling of fully eulerian and arbitrary lagrangian–eulerian methods for fluid-structure interaction computations. *Computational Mechanics 52*, 5, 1113–1124.

ZHANG, X., BRIDSON, R., AND GREIF, C. 2015. Restoring the missing vorticity in advection-projection fluid solvers. *ACM Trans. Graph. 34*, 4, 52:1–52:8.

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

ZHU, B., LU, W., CONG, M., KIM, B., AND FEDKIW, R. 2013. A new grid structure for domain extension. *ACM Trans. Graph. 32*, 4, 63:1–63:12.