An Advection-Reflection Solver for Detail-Preserving Fluid Simulation

JONAS ZEHNDER, Université de Montréal
RAHUL NARAIN, University of Minnesota and Indian Institute of Technology Delhi
BERNHARD THOMASZEWSKI, Université de Montréal

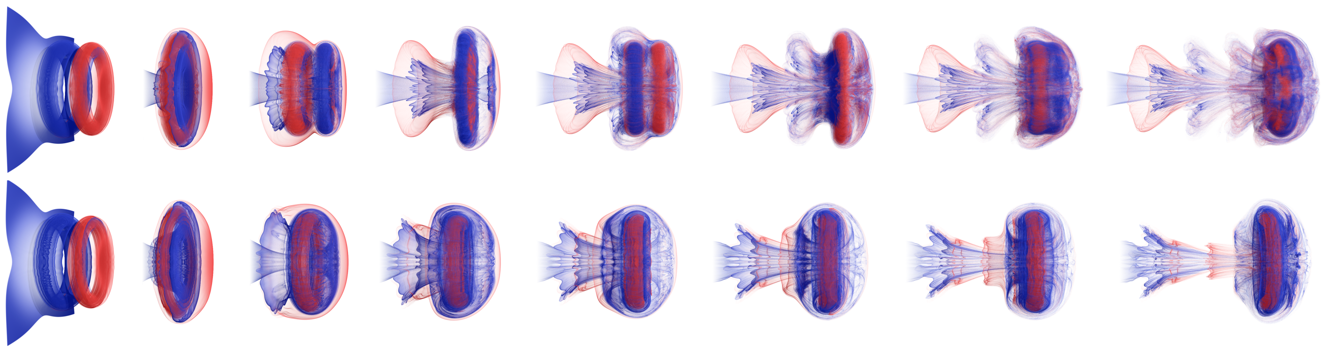


Fig. 1. Our new reflection solver applied to a vortex leap-frogging problem (*top row*). During 10*s* of simulation the two vortex rings move through each other multiple times and stay well separated. By contrast, in a standard advection-projection method with MacCormack advection (*bottom row*), the two vortices merge immediately and never separate afterwards.

Advection-projection methods for fluid animation are widely appreciated for their stability and efficiency. However, the projection step dissipates energy from the system, leading to artificial viscosity and suppression of small-scale details. We propose an alternative approach for detail-preserving fluid animation that is surprisingly simple and effective. We replace the energy-dissipating projection operator applied at the end of a simulation step by an *energy-preserving reflection* operator applied at mid-step. We show that doing so leads to two orders of magnitude reduction in energy loss, which in turn yields vastly improved detail-preservation. We evaluate our reflection solver on a set of 2D and 3D numerical experiments and show that it compares favorably to state-of-the-art methods. Finally, our method integrates seamlessly with existing projection-advection solvers and requires very little additional implementation.

CCS Concepts: • **Computing methodologies → Physical simulation**; *Computer graphics*; *Animation*;

Additional Key Words and Phrases: fluid simulation, advection, reflection, energy conservation

ACM Reference Format:
Jonas Zehnder, Rahul Narain, and Bernhard Thomaszewski. 2018. An Advection-Reflection Solver for Detail-Preserving Fluid Simulation. *ACM Trans. Graph.* 37, 4, Article 85 (August 2018), 8 pages. https://doi.org/10.1145/3197517.3201324

Authors' addresses: Jonas Zehnder, Université de Montréal, jonas.zehnder@umontreal.ca; Rahul Narain, University of Minnesota, Indian Institute of Technology Delhi, narain@cse.iitd.ac.in; Bernhard Thomaszewski, Université de Montréal, bernhard@iro.umontreal.ca.

0730-0301/2018/8-ART85 $15.00
https://doi.org/10.1145/3197517.3201324

1 INTRODUCTION

Advection-projection methods are widely used for fluid animations in computer graphics. Splitting mass transport and conservation into different steps allows for stable and efficient integration, and advances in higher-order advection schemes (e.g. [Selle et al. 2008]) have greatly reduced the well-known numerical diffusion caused by the semi-Lagrangian advection step. However, the splitting of the time integration scheme itself induces numerical dissipation, as kinetic energy is transferred to divergent modes during advection and then lost after projection. This numerical dissipation manifests as rapid decay of large vortices and leads to suppression of small-scale swirling motion. Since visual complexity is a central goal in fluid animation, much effort has been spent on combating numerical dissipation: apart from higher-order advection schemes mentioned above, energy-preserving integration [Mullen et al. 2009], a posteriori correction of the velocity field [Fedkiw et al. 2001; Zhang et al. 2015], and injection of procedurally-generated detail [Kim et al. 2008b] are among the strategies that have been pursued so far.

In this work, we propose an alternative approach to detail-preserving fluid animation that is surprisingly simple and effective: we replace the energy-dissipating projection operator applied at the end of a simulation step by an *energy-preserving reflection* operator applied at mid-step; see Fig 2. We show that doing so leads to an order of magnitude reduction in divergent kinetic energy, which in turn leads to vastly improved preservation of vortices and small-scale detail. This analysis also exposes our method to be first-order structurally symmetric, which motivates an analogy to (and comparison with) symmetric projection methods for structure-preserving integration of constrained mechanical systems [Hairer et al. 2006].

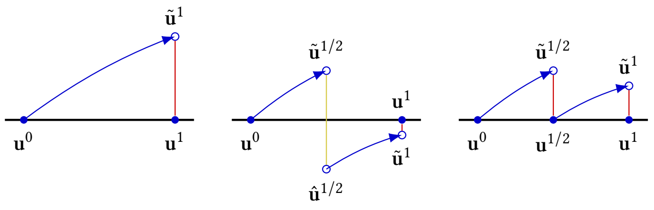


Fig. 2. A geometric interpretation of our method. *Left:* In a standard advection-projection solver, projection to the divergence-free subspace causes kinetic energy loss (red). *Middle:* Our reflection solver uses an energy-preserving *reflection* (yellow) halfway through the advection step, dramatically reducing the energy loss caused by the final projection. Our method has effectively identical computational cost to an advection-projection solver with half the time step *(right)*, but loses less energy.

We show that even the simplest methods from this category are computationally much more expensive and less stable. By contrast, our method preserves the appreciable splitting property of advection-projection methods while offering energy and detail preservation similar to fully-symmetric methods.

Our method integrates seamlessly with existing advection-projection solvers and is agnostic to the choice of advection scheme and pressure discretization. Furthermore, it uses the basic advection and projection steps as primitives, and therefore requires very little additional implementation. We evaluate our reflection solver on an extensive set of 2D and 3D examples and compare its behavior to a number of alternative methods. The results of these comparisons indicate that, for equal computational costs, our method leads to vastly improved energy and vorticity preservation.

2 RELATED WORK

Our review of related works focuses primarily on Eulerian fluid simulation methods, as our approach does not apply to Lagrangian particle-based methods like SPH [Ihmsen et al. 2014]. Also, we will only discuss schemes for solving the core Navier-Stokes equations, omitting the diversity of techniques in graphics for artificially injecting detail such as vorticity confinement [Fedkiw et al. 2001], vortex particles [Selle et al. 2005], and turbulence synthesis [Thuerey et al. 2013].

Most Eulerian methods in graphics follow a Chorin-style advection-projection scheme [Chorin 1968] introduced to graphics by Stam [1999]. The effectiveness of the advection-projection approach stems from three main ingredients: (i) operator splitting decouples the pressure term from the remaining inertial and internal forces, (ii) semi-Lagrangian advection [Robert 1981] permits large time steps unimpeded by the CFL condition, and (iii) staggered grids [Foster and Metaxas 1996; Harlow and Welch 1965] allow for accurate computation of the pressure projection. While this approach is unconditionally stable, it exhibits noticeable energy loss over time even for inviscid flows. Small-scale vortices and turbulent flows tend to decay especially rapidly, leading to a loss of visually interesting detail. Much work in computer graphics has focused on minimizing this numerical dissipation.

As discussed in the introduction, there are two main reasons for the loss of energy in an advection-projection method: the discretization error in the semi-Lagrangian advection step, which manifests as artificial diffusion, and the splitting error caused by decoupling of the advection and projection steps. To reduce the diffusion in semi-Lagrangian advection, Fedkiw et al. [2001] and Kim et al. [2008a] have proposed higher-order interpolation schemes to improve spatial accuracy. Kim et al. [2005, 2007] introduced the BFECC method which performed multiple backward and forward advection steps to correct both spatial and temporal error. This approach was simplified by Selle et al. [2008], who proposed a semi-Lagrangian variation of the MacCormack method and demonstrated second-order accuracy in space and time. Molemaker et al. [2008] proposed the use of the QUICK advection scheme [Leonard 1979] for low-dissipation advection, although it is limited by the CFL condition for stability. In this context, hybrid particle-and-grid methods like FLIP [Zhu and Bridson 2005] and APIC [Jiang et al. 2015] are very attractive as they exhibit little numerical diffusion of this form, because they track the advected quantities on Lagrangian particles which are largely unaffected by the grid interpolation.

In contrast to the improvements in low-diffusion advection schemes, much less attention has been paid to the error introduced by the splitting scheme itself. In graphics, this has been pointed out by Elcott et al. [2007] and Zhang et al. [2015], who noted that semi-Lagrangian advection transfers energy into divergent modes which are then annihilated by the projection step. This is true even for the FLIP and APIC methods, which employ the same advection-projection splitting. Elcott et al. instead adopted the vorticity formulation of the fluid equations, in which the primary variable is the vorticity rather than the velocity field. Other vorticity-based methods in graphics include Angelidis and Neyret [2005]; Park and Kim [2005]; Weißmann and Pinkall [2010]. Since the vorticity representation automatically yields a divergence-free velocity field, such methods do not require a projection step, and consequently do not suffer the associated energy loss. Particularly notable is the method of Mullen et al. [2009], which is time-reversible and offers exact energy preservation through the use of a symplectic integrator, albeit at the cost of requiring the solution to a nonlinear system at each time step. Nevertheless, advection-projection methods continue to enjoy widespread use in industry and academic research, possibly due to their relative simplicity, efficiency, and flexibility in comparison to vorticity methods [Zhang et al. 2015]. Therefore, we believe that advances in energy preservation for advection-projection methods are still highly desirable.

Remaining within the advection-projection framework, Zhang et al. [2015] counteracted the artificial dissipation caused by the projection step by explicitly tracking the lost vorticity and re-injecting it into the fluid. Thus, their work seeks to preserve vorticity but not necessarily the energy of the fluid. In contrast, we propose a simple modification to the splitting scheme which reduces the projection error without requiring explicit tracking and correction. We also provide a simple proof that our scheme preserves kinetic energy to a higher degree than traditional advection-projection methods.

3 THEORY

To set the stage for our theoretical developments, we start with a minimal review of advection projection methods before we introduce our reflection solver. For context and comparison, we subsequently introduce two fully-symmetric projection methods.

For notational convenience, we will refer to the velocity field at the beginning and end of the time step as $\mathbf{u}^0$ and $\mathbf{u}^1$ respectively, instead of $\mathbf{u}^n$ and $\mathbf{u}^{n+1}$. We will also adopt the convention that divergence-free velocity fields are undecorated, e.g. $\mathbf{u}^{1/2}$, while velocity fields with nonzero divergence are denoted $\tilde{\mathbf{u}}$ (before projection) or $\hat{\mathbf{u}}$ (after reflection).

3.1 Advection-Projection Solvers

The starting point for our developments are the inviscid, incompressible Navier-Stokes equations

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \mathbf{f} \tag{1}$$

$$\nabla \cdot \mathbf{u} = \mathbf{0}\,, \tag{2}$$

where $\mathbf{u}$ is the continuous velocity field of the fluid, p is the pressure, ρ the density and $\mathbf{f}$ denote external forces such as buoyancy and gravity. Advection-projection methods discretize the velocities on a Eulerian grid and split the equations into three (or more) different steps:

$$\begin{aligned} \text{Advection} \quad & \tilde{\mathbf{u}}^1 = A(\mathbf{u}^0; \mathbf{u}^0, \Delta t) \\ \text{Forcing} \quad & \tilde{\mathbf{u}}^1 \mathrel{+}= \Delta t\, \mathbf{f} \\ \text{Projection} \quad & \mathbf{u}^1 = P\tilde{\mathbf{u}}^1\,. \end{aligned}$$

In the above expressions, $A(\mathbf{u}; \mathbf{u}^0, \Delta t)$ is an advection operator that implements a semi-Lagrangian discretization of the advection equation, $\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}^0 \cdot \nabla)\mathbf{u} = \mathbf{0}$. Furthermore, P is a projection operator that maps a given velocity field to its closest divergence-free field (under the kinetic energy metric). This operator uses the Helmholtz-Hodge decomposition, which splits any vector field $\mathbf{u} = \mathbf{v} + \mathbf{w}$ into a curl-free part $\mathbf{v}$ and a divergence-free part $\mathbf{w}$. P simply discards the curl-free part, $P\mathbf{u} = \mathbf{w}$, by solving a Poisson problem.

For conciseness, we will neglect the forcing step in the following and focus on the central advection and projection steps. In practice, we apply external forces immediately after each advection.

Lemma 1. *If a vector field $\tilde{\mathbf{u}}$ has divergence $\Theta(\Delta t^k)$, the kinetic energy loss due to projection, $\frac{1}{2}\|\tilde{\mathbf{u}}\|^2 - \frac{1}{2}\|P\tilde{\mathbf{u}}\|^2$, is $\Theta(\Delta t^{2k})$.*

Proof. The pressure projection can be interpreted as decomposing the post-advection velocity $\tilde{\mathbf{u}}$ into two orthogonal components, $\tilde{\mathbf{u}} = P\tilde{\mathbf{u}} + \mathbf{v}$, where $\mathbf{v}$ is the curl-free part. Due to the orthogonality of the Helmholtz-Hodge decomposition, we have $\|\tilde{\mathbf{u}}\|^2 = \|P\tilde{\mathbf{u}}\|^2 + \|\mathbf{v}\|^2$. Therefore, the energy loss $\frac{1}{2}\|\tilde{\mathbf{u}}\|^2 - \frac{1}{2}\|P\tilde{\mathbf{u}}\|^2$ is precisely the kinetic energy of the curl-free part, $\frac{1}{2}\|\mathbf{v}\|^2$. Furthermore, the curl-free part $\mathbf{v}$ is determined by the divergence of $\tilde{\mathbf{u}}$ and depends linearly on it; that is, we can write $\mathbf{v} = H(\nabla \cdot \tilde{\mathbf{u}})$ for some linear operator H.[1] If $\nabla \cdot \tilde{\mathbf{u}} = \Theta(\Delta t^k)$, it can be expressed as $\nabla \cdot \tilde{\mathbf{u}} = \delta\,\Delta t^k + o(\Delta t^k)$ for some scalar field δ. Thus we have $\frac{1}{2}\|\mathbf{v}\|^2 = \frac{1}{2}\|H\delta\|^2\Delta t^{2k} + o(\Delta t^{2k}) = \Theta(\Delta t^{2k})$. □

[1]In particular, $H = \nabla\Delta^{-1}$, where Δ^{-1} denotes the inverse of the Laplace operator with the appropriate problem-defined boundary conditions.

Using this result, we show that an advection-projection solver can at best only preserve energy to first order in time.

Theorem 2. *The kinetic energy loss due to the projection step of the advection-projection method is $\Theta(\Delta t^2)$.*

Proof. Expanding the advection operator into its Taylor series,

$$\tilde{\mathbf{u}}^1 = \mathbf{u}^0 - (\mathbf{u}^0 \cdot \nabla)\mathbf{u}^0\Delta t + O(\Delta t^2). \tag{3}$$

we find that the divergence of the advected velocity field is

$$\nabla \cdot \tilde{\mathbf{u}}^1 = \nabla \cdot \mathbf{u}^0 - \nabla \cdot ((\mathbf{u}^0 \cdot \nabla)\mathbf{u}^0)\Delta t + O(\Delta t^2) \tag{4}$$

$$= \underbrace{-\nabla \cdot ((\mathbf{u}^0 \cdot \nabla)\mathbf{u}^0)}_{\delta(\mathbf{u}^0)}\Delta t + O(\Delta t^2). \tag{5}$$

For a divergence-free velocity field $\mathbf{u}$, it can be shown that the rate of divergence $\delta(\mathbf{u})$ is proportional to the second invariant of the velocity gradient, $\sum_{ij} \frac{\partial u_i}{\partial x_j}\frac{\partial u_j}{\partial x_i}$, and is not in general zero. Therefore, we have $\nabla \cdot \tilde{\mathbf{u}}^1 = \Theta(\Delta t)$, resulting in an energy loss of order $\Theta(\Delta t^2)$. □

3.2 Reflection Solver

We would like to avoid the second-order energy loss during projection while still achieving zero divergence at the end of the time step. The intuition for our approach is that, instead of correcting the velocity at the end of the time step, we can *over-compensate* in the middle: we advect to the middle of the time interval and apply twice the correction needed to obtain a divergence-free field such as to anticipate the divergence incurred during the second half. Geometrically, this operation can be interpreted as symmetrically changing from one side of the divergence-free manifold to the other, hence the term *reflection*; see Fig. 2. Concretely, the reflection solver proceeds as follows:

$$\begin{aligned} \tilde{\mathbf{u}}^{1/2} &= A(\mathbf{u}^0; \mathbf{u}^0, \tfrac{1}{2}\Delta t) \\ \mathbf{u}^{1/2} &= P\tilde{\mathbf{u}}^{1/2} \\ \hat{\mathbf{u}}^{1/2} &= 2\mathbf{u}^{1/2} - \tilde{\mathbf{u}}^{1/2} \\ \tilde{\mathbf{u}}^1 &= A(\hat{\mathbf{u}}^{1/2}; \mathbf{u}^{1/2}, \tfrac{1}{2}\Delta t) \\ \mathbf{u}^1 &= P\tilde{\mathbf{u}}^1\,. \end{aligned}$$

As it is preferable to perform semi-Lagrangian advection using a divergence-free velocity field (otherwise the advection equation does not correspond to a conservation law for the advected quantity), we use the *projected* mid-step velocity $\mathbf{u}^{1/2}$ in the second semi-Lagrangian advection step. Note that the reflection velocity $\hat{\mathbf{u}}^{1/2}$ can also be written directly as $\hat{\mathbf{u}}^{1/2} = R\tilde{\mathbf{u}}^{1/2}$, where $R = 2P - \mathbf{I}$ is the reflection operator. The final projection $\mathbf{u}^1 = P\tilde{\mathbf{u}}^1$ guarantees divergence-free velocity at the end of the time step. However, if the rate of divergence does not vary much across the time step, then $\tilde{\mathbf{u}}^1$ will already be close to divergence-free. Indeed, this intuition is confirmed by the following statement.

Theorem 3. *The kinetic energy loss due to the projection step of the reflection solver is $O(\Delta t^4)$.*

Proof. Using the Taylor series expansion of the advection operator, we approximate the mid-step velocities as

$$\tilde{\mathbf{u}}^{1/2} = \mathbf{u}^0 - (\mathbf{u}^0 \cdot \nabla)\mathbf{u}^0 \Delta t/2 + O(\Delta t^2)\,, \tag{6}$$

$$\hat{\mathbf{u}}^{1/2} = \mathbf{u}^0 - R(\mathbf{u}^0 \cdot \nabla)\mathbf{u}^0 \Delta t/2 + O(\Delta t^2)\,, \tag{7}$$

where we have used the fact that $R\mathbf{u}^0 = \mathbf{u}^0$ since $\mathbf{u}^0$ is already divergence-free. An analogous first-order approximation of the end-of-step velocity *before* projection, $\tilde{\mathbf{u}}^1$, yields

$$\begin{aligned}
\tilde{\mathbf{u}}^1 &= \hat{\mathbf{u}}^{1/2} - (\mathbf{u}^{1/2} \cdot \nabla)\hat{\mathbf{u}}^{1/2}\Delta t/2 + O(\Delta t^2) && (8)\\
&= \big(\mathbf{u}^0 - R(\mathbf{u}^0 \cdot \nabla)\mathbf{u}^0 \Delta t/2\big) && (9)\\
&\quad - \big((\mathbf{u}^0 + O(\Delta t)) \cdot \nabla\big)\big(\mathbf{u}^0 + O(\Delta t)\big)\Delta t/2 + O(\Delta t^2) && (10)\\
&= \mathbf{u}^0 - R(\mathbf{u}^0 \cdot \nabla)\mathbf{u}^0 \Delta t/2 - (\mathbf{u}^0 \cdot \nabla)\mathbf{u}^0 \Delta t/2 + O(\Delta t^2) && (11)\\
&= \mathbf{u}^0 - 2P\big((\mathbf{u}^0 \cdot \nabla)\mathbf{u}^0\big)\Delta t/2 + O(\Delta t^2)\,. && (12)
\end{aligned}$$

From this, it is evident that $\tilde{\mathbf{u}}^1$ is divergence-free to first order, i.e. $\nabla \cdot \tilde{\mathbf{u}}^1 = O(\Delta t^2)$. Therefore, the resulting energy loss is of order $O(\Delta t^4)$. □

The good energy-preserving properties of our reflection solver are further assured by the fact that

Theorem 4. *The reflection operator preserves kinetic energy.*

Proof. The pressure projection $\mathbf{w} = P\mathbf{u}$ can be interpreted as finding $\mathbf{w}$ as the closest point to $\mathbf{u}$ in the space of divergence-free vector fields, under the metric defined by kinetic energy [Batty et al. 2007]. Consequently, P is an *orthogonal* projection with respect to kinetic energy, and $\mathbf{v}$ and $\mathbf{w}$ are orthogonal to each other in the sense that $\langle \mathbf{v}, \mathbf{w}\rangle = \iiint \rho \mathbf{v}\cdot\mathbf{w}\, \mathrm{d}V = 0$. Using the definition of the reflection operator, we have for the reflected velocity field $R\mathbf{u} = 2\mathbf{w} - \mathbf{u} = \mathbf{w} - \mathbf{v}$. Comparing the kinetic energies, we find that

$$\frac{1}{2}\langle R\mathbf{u}, R\mathbf{u}\rangle = \frac{1}{2}(\langle \mathbf{w}, \mathbf{w}\rangle + \langle \mathbf{v}, \mathbf{v}\rangle) = \frac{1}{2}\langle \mathbf{u}, \mathbf{u}\rangle. \tag{13}$$

□

These results also point to the stability of our method, since the reflection operator preserves energy, while the final projection can only reduce it. In conjuction with the stability of semi-Lagrangian advection operations, one would like to argue that the reflection method is unconditionally stable. This argument does not go through smoothly, however, as advection is stable in a slightly different sense: it is monotonic in the field values, not necessarily in energy. However, the same caveat applies to all other advection-projection solvers used in graphics. As such we expect to see unconditional stability in practice, independent of the time step size.

It is worth noting here that the reflection solver does not necessarily improve the *accuracy* of the solution over projection solvers, since it still incurs error due to self-advection using a "frozen" velocity field. For time-varying flows the method remains only first-order accurate in time, as we show in Section 4.3. Instead, the benefit of the reflection approach is a higher degree of energy conservation. This is analogous to how BDF2 and implicit midpoint are both second-order accurate integration schemes, but implicit midpoint has significantly better conservation properties.

3.3 Symmetric Projection Methods

The results of the previous section suggest that our reflection solver is *first-order structurally symmetric*: even though the advection operator itself is not, the structure of the advection-reflection-advection sequence is symmetric. This observation prompted us to draw the analogy to symmetric manifold projection methods, a class of integrators for conservative mechanical systems known for their excellent long-term energy conservation [Hairer et al. 2006]. Indeed, by composing the advection-projection step with its adjoint—and vice-versa—we obtain two immediate candidates for symmetric advection-projection schemes.

We start by defining some key terms, following Hairer et al. [2006]. The *adjoint* of a time-stepping scheme $\Phi(\,\cdot\,, h)$ is the inverse of its time reversal, $\Phi^*(\,\cdot\,, h) = \Phi^{-1}(\,\cdot\,, -h)$. A time-stepping scheme is *symmetric*, or *time-reversible*, if it is equal to its adjoint. Composing any consistent first-order scheme Φ with its adjoint yields a symmetric second-order scheme, $\Psi(\,\cdot\,, h) = \Phi^*(\,\cdot\,, h/2) \circ \Phi(\,\cdot\,, h/2)$.

Strictly speaking, the adjoint of the pressure step does not exist because P as a linear operator is not invertible. In this section, we interpret P as an arbitrary perturbation normal to the divergence-free manifold, defining $P(\mathbf{u}; p) = \mathbf{u} - \nabla p$ for any pressure field p. This operation is self-adjoint: $P^*(\mathbf{u}; p) = P^{-1}(\mathbf{u}; -p) = P(\mathbf{u}; p)$.

APA^*. The first scheme is obtained by composing the advection-projection step, AP, with its adjoint PA^*. Geometrically, we first advect to the middle of the time step to obtain $\tilde{\mathbf{u}}^{1/2}$ and project onto the divergence-zero manifold. To implement PA^*, we solve for a pressure field p used to compute velocity perturbations $\Delta\mathbf{u}_p = -\nabla p$ normal to the manifold, as well as divergence-free end-of-time-step velocities $\mathbf{u}^1$ such that, when advecting $\mathbf{u}^1$ backward in time, we end up at the perturbed mid-step velocities $\hat{\mathbf{u}}^{1/2} = \tilde{\mathbf{u}}^{1/2} - \nabla p$. This sequence of coupled operations translates into a system of nonlinear equations,

$$\left\{ \begin{array}{c} A(\mathbf{u}^1; \mathbf{u}^1, -\frac{1}{2}\Delta t) - (A(\mathbf{u}^0; \mathbf{u}^0, \frac{1}{2}\Delta t) - \nabla p) = 0 \\ \nabla \cdot \mathbf{u}^1 = 0 \end{array} \right\}. \tag{14}$$

in which the projection and perturbation steps combine into a single operation, which is why we mnemonically refer to this scheme as APA^*. We note that this system is similar to the one derived by Mullen et al. [2009]. We solve (14) with Newton's method and, as do Mullen et al., approximate the Jacobian of the advection operator with the identity matrix. When applying block Gaussian elimination to the resulting saddle-point systems, we recover the same Poisson problem encountered in standard advection-projection methods,

$$\nabla^2(\Delta p) = \nabla \cdot \mathbf{r}_{APA^*}\,, \tag{15}$$

albeit with a different right hand side $\mathbf{r}_{APA^*}$ that we omit here for brevity. This pressure solve must be performed once per Newton iteration, providing us with updated perturbations Δp and corresponding velocity updates.

PA^*AP. The second scheme is obtained by composing the adjoint of the advection-projection step with itself, leading to the mnemonic sequence PA^*AP. As a geometric interpretation, we solve for perturbations p and mid-step velocities $\mathbf{u}^{1/2}$ such that 1) advecting from the middle to the end and applying the correction $-\nabla p$ leads to

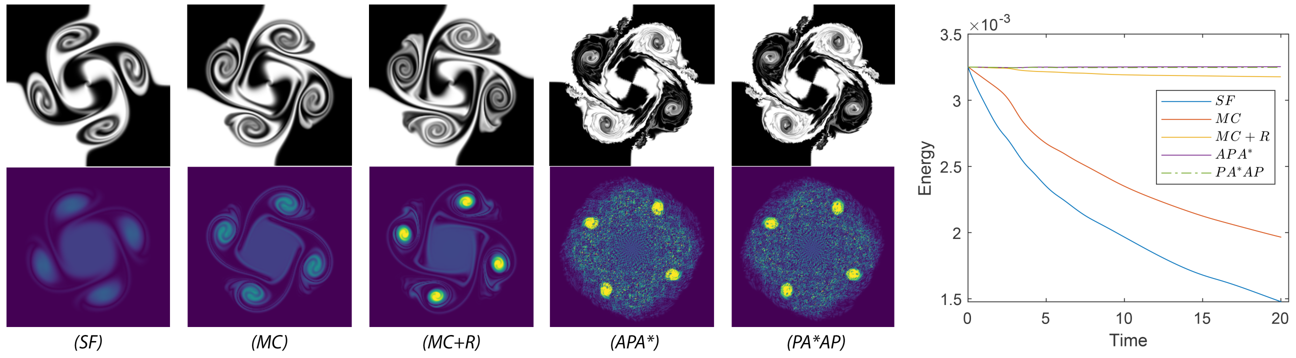


Fig. 3. 2D vortex sheet example: comparison of the density distribution (*top*) and the vorticity magnitude (*bottom*) for a fixed point in time (10*s*) as obtained for the various solvers. *Far right*: kinetic energy as a function of time.

divergence-free velocities, and 2) advecting from the middle backwards leads to the perturbed initial velocities $\mathbf{u}^0 - \nabla p$. Combining these operations together leads to a system of nonlinear equations,

$$\begin{cases} A(\mathbf{u}^{1/2}; \mathbf{u}^{1/2}, -\tfrac{1}{2}\Delta t) - \left(\mathbf{u}^0 - \nabla p\right) = 0 \\ \nabla \cdot \left(A(\mathbf{u}^{1/2}; \mathbf{u}^{1/2}, \tfrac{1}{2}\Delta t) - \nabla p\right) = 0 \end{cases} \tag{16}$$

that we solve for the mid-step velocities $\mathbf{u}^{1/2}$ and the unknown perturbation field p using Newton's method. This scheme is analogous to the symmetric projection method describe by Hairer et al. [2006] and, upon approximation of the Jacobian with the identity matrix and block Gaussian elimination, leads again to a standard pressure solve (the details of which we leave out for conciseness).

Discussion. We provide qualitative and quantitative evaluations for these symmetric projection methods in Section 4 and compare them to our reflection solver. But even before further analysis, we can already expect these methods to have much higher computational cost than our reflection solver: solving the systems to sufficient accuracy requires several Newton iterations, each with one (resp. two) advection steps and a pressure solve. Furthermore, while the use of an approximate Jacobian is justified by the difficulty of computing the full Hessian, it warrants the use of line search for convergence control. However, since the systems of equations do not derive from a minimization problem with associated objective function, we can only monitor the norm of the residual—a poor measure of progress that can even prevent convergence.

4 RESULTS

We investigated the qualitative and quantitative behavior of our reflection solver on a set of 2D and 3D experiments commonly used in the literature. We compare our results to those obtained for conventional advection-projection solvers and report our findings below.

Solvers & Setup. For the 2D experiments, we use our own solver based on a MAC grid discretization. To implement internal obstacles in the flow, we use the method of Batty et al. [2007] and apply corresponding modifications to the matrix of the pressure solves. The 3D examples are based on the Mantaflow library [Thuerey and Pfaff 2016], which we modified slightly to implement our reflection solver.

We compare the following solvers: stable fluids with first-order semi-Lagrangian advection (*SF*) as described by Stam [1999], stable fluids with MacCormack advection (*MC*) [Selle et al. 2008], our reflection solver (*R*), as well as the symmetric projection methods (*APA**) and (*PA*AP*). Statistics on all experiments and solvers—including step size, grid size, and computation time—are listed in Table 1. Since our reflection solver requires two advection operations and two pressure solves per time step, we use twice the step size when comparing to *SF* and *MC*, giving us essentially identical computation time.

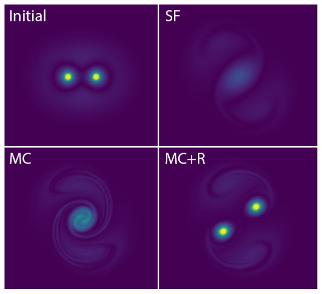
4.1 2D Results

Since two-dimensional flows are generally easier to interpret and compare, we begin our analysis in the 2D setting.

2D Vortex Sheet. We initialize a disc-shaped region in the center of the scene with a rigid rotation as the initial velocity field. After

Table 1. (1) 2D Vortex Sheet, (2) Taylor Vortices, (3) Vortex Shedding, (4) Spiral Maze, (5) Vortex Leap-Frogging, (6) Ink Drop, (7) Smoke Plume, (8) Smoke Plume with Sphere
The time step was doubled for the reflection solver when producing the results to keep cost similar to the projection method. The time step was reduced for PA^*AP and APA^* to satisfy the CFL condition.

	Resolution	Domain size	Δt	Iter. time (s) MC	Iter. time (s) MC+R
1	256×256	1×1	0.025	0.0281	0.055
2	256×256	1×1	0.025	0.0285	0.055
3	512×128	1×0.25	0.0025	0.0322	0.0638
4	256×256	0.75×0.75	0.025	0.0558	0.1107
5	$256 \times 128 \times 128$	$256 \times 128 \times 128$	0.25	5.59	10.43
6	$128 \times 64 \times 64$	$128 \times 64 \times 64$	1	0.27	0.57
7	$128 \times 256 \times 128$	$128 \times 256 \times 128$	1	5.73	10.66
8	$128 \times 256 \times 128$	$128 \times 256 \times 128$	1	7.97	13.48



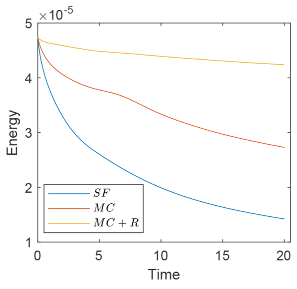


Fig. 4. 2D Taylor vortices. *Left*: initial vorticity magnitude and results for the three solvers after 10*s*. *Right*: kinetic energy as a function of time.

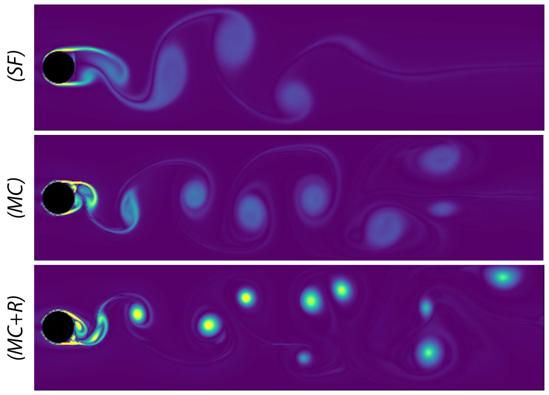


Fig. 5. 2D vortex shedding: comparison of the vorticity magnitude distributions for *SF*, *MC*, and *MC+R* for a fixed point in time (6*s*).

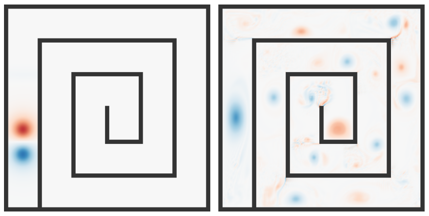


Fig. 6. A vortex in a spiral maze (*left*) correctly advects itself to the center (*right*).

initialization, no additional energy is injected into the system, allowing us to investigate the (long-term) energy conservation properties of the different solvers.

Fig. 3 shows an overview of density and vorticity magnitude fields obtained for the various solvers applied to this problem. While the differences in dynamic behavior are best observed in the accompanying video, it can be seen that our reflection solver *MC+R* shows slightly better detail preservation in the density field than *MC* and much less vorticity diffusion. Moreover, the temporal evolution of kinetic energy shown in Fig. 3 (right) speaks a very clear language: *SF* rapidly dissipates energy, which drops to two thirds of its initial value after roughly 6*s*. *MC* performs better, but still loses one third of its energy after 13*s*. By contrast, our reflection solver preserves energy much better, losing less than 3% over the entire animation (20*s*). For reference, the symmetric projection methods both preserve energy perfectly, but they require roughly 10 times more computation time. What is worse, however, is that both methods lead to visually disturbing artifacts in the density field and very noisy vorticity (see Fig. 3). We have investigated this behavior intensively but could not find a problem with our implementation. We conjecture that the reason for these artifacts lies in the semi-Lagrangian advection operator: the interpolations performed when tracing back through the velocity field act as a low-pass filter, i.e., information is lost due to interpolation. Although the continuity of the flow provides some amount of regularization, the effective inversion of this low-pass filter during A^* is numerically unstable, explaining both the artifacts and the intermittent convergence problems that we observed. Although we intended these symmetric methods to be reference solutions, we found them to be unusable in practice and will therefore not discuss them further.

Taylor Vortices. Another frequently used 2D example is that of two vortices of the same sign placed at a given initial distance. Depending on this initial distance, the analytical solution for this problem will lead to the vortices either merging or separating. Following Mullen et al. [2009], we choose the initial distance slightly larger than the critical value and investigate the behavior of the solvers. For both *SF* and *MC*, the vortices merge shortly after the beginning of the simulation, whereas they separate as predicted by the analytical solution using our reflection solver.

Vortex Shedding. This example investigates the behavior of the different solvers in combination with internal boundaries, leading to dynamic and visually rich vortex shedding. As can be seen from Fig. 5, *MC* initially produces similar behavior in the sense that roughly the same number of vortices is shed during the first 6 seconds. However, whereas our reflection solver approximately preserves the vorticity of the shed vortices, there is a clear decay in vorticity (from left to right) for *MC*. The video also shows that the behavior for *SF* is qualitatively very different and the numerical viscosity is clearly visible.

Spiral Maze. An example introduced by Mullen et al. [2009] contains a pair of vortices in a 2D domain with many boundaries forming a spiral maze. In the absence of dissipation, one of the vortices should advect itself to the center of the maze. As shown in Figure 6, our method produces the expected behavior. Interestingly, unlike the results of Mullen et al., we also observe additional vortex shedding due to flow separation at the convex corners of the maze.

4.2 3D Results

To investigate the behavior of our reflection solver in 3D, we chose a set of examples frequently used in the literature.

Vortex Leap-Frogging. This classical example is initialized with two concentric vortex rings of different radii but equal circulations. In the analytical solution for the completely inviscid, conservative case, the two vortices will produce a *leap-frogging* motion that continues indefinitely. However, reproducing this behavior with Eulerian methods has been a challenging problem due to the tendency for numerical dissipation to diffuse the vortices into each

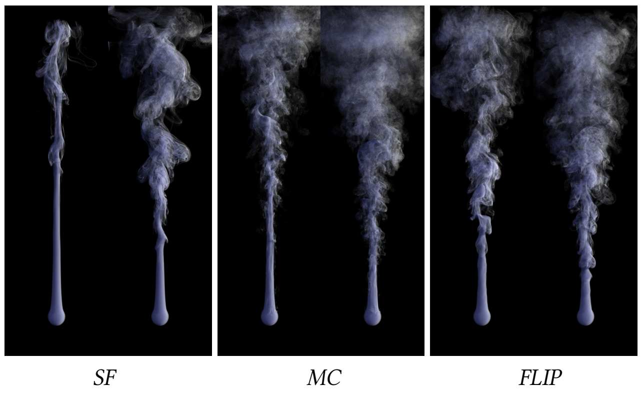


Fig. 7. 3D smoke plume with different advection schemes. In each pair, we compare the same simulation frame computed using a standard projection solver (*left*) and our reflection solver (*right*).

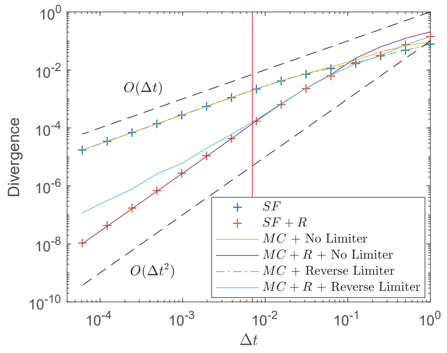


Fig. 8. Norm of the divergence before the projection step for different step sizes and different solvers. *Reverse Limiter* refers to the clamping of the interpolated field value in the MacCormack scheme.

other. As can be seen in Fig. 1, when using the *MC* solver, the vortex rings merge into a single one during the first leap-through motion. By contrast, our solver successfully produces several leap-frogging moves in which the rings remain clearly separated. We refer to the accompanying video for a better illustration of this fascinating phenomenon.

Ink Drop. A spherical density field is initialized with constant horizontal velocity. Using our reflection solver, the velocity gradient at the sphere's interface immediately leads to the formation of a large vortex, leaving behind it a trail of turbulent fluid that develops into increasingly complex patterns. While the *MC* solver can reproduce the vortex formation, the flow has much more viscosity and the trail it leaves behind is devoid of any detail. Please refer to the accompanying video.

Smoke Plume. As another classical example, we simulate a smoke plume by modeling a spherical density source subject to a constant density-proportional buoyancy. The action of buoyancy leads to the formation of a characteristic vortex front, followed by turbulent breakup of the rising smoke column; see Fig. 7. We have compared the standard advection-projection scheme with our advection-reflection solver with three different advection schemes: *SF*, *MC*, and *FLIP* [Zhu and Bridson 2005]. While the overall qualitative behavior is similar for most cases, our reflection solver uniformly yields more pronounced vortical motion and stronger turbulence due to the reduced artificial dissipation. In the accompanying video, we also show an example with a spherical obstacle in the path of the plume.

4.3 Convergence Analysis

In order to validate our theoretical result for the improved order of accuracy of our method, we compared different solvers with different parameters on a 2D smoke plume example. We let the simulation run for 2*s* such that an interesting flow field develops, and compute the divergence before projection at the end of a given time step. Fig. 8 shows a log-log plot of this pre-projection divergence as a function of the step size. It can be seen that, for the *SF* and *MC* methods, the divergence decreases linearly with the step size. For our reflection solver, however, the decrease is indeed quadratic, irrespective of whether it is combined with first order or MacCormack advection. Another observation that we made in this context is that because in this test we only increased the resolution in time, not in space, clamping the interpolated field value in the MacCormack scheme slows the convergence of the norm of the divergence.

To evaluate the accuracy of our method, we performed a convergence test using a 2D Taylor-Green vortex with initial conditions

$$\mathbf{u}_{\text{TG}} = (\sin(2\pi x)\cos(2\pi y), -\cos(2\pi x)\sin(2\pi y)).$$

In the inviscid case, this is a steady flow with $\mathbf{u}$ constant over time. We also simulated an example with initial velocity $\mathbf{u} = \mathbf{u}_{\text{TG}} + (1, 0)$ and periodic boundary conditions, which should result in a pure translation of the vortex. We ran both examples to 1s, and computed the RMS error in velocity with respect to the analytical solution. Figure 9 shows a log-log plot of this error as a function of time step. For the steady flow, our reflection solver exhibits third-order convergence, since energy loss is the only source of error in this case. For the unsteady flow, the error caused by the use of the "frozen" velocity field in self-advection dominates, and we observe similar first-order convergence for both projection and reflection methods.

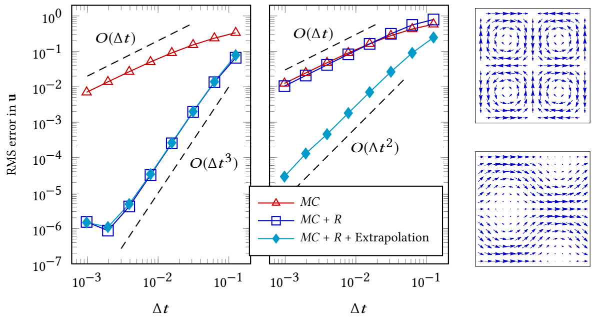


Fig. 9. RMS error in velocity after one second for (*left*) the 2D Taylor-Green vortex, and (*middle*) the Taylor-Green vortex with added translation. "Extrapolation" denotes advection with the velocity field $2\mathbf{u}^{1/2} - \mathbf{u}^0$ in the second half-step. *Right*: Initial velocity fields for both cases are visualized.

However, when performing this test we noticed that convergence appears to be improved to second order by performing the second advection half-step using an "extrapolated" velocity field $2\mathbf{u}^{1/2} - \mathbf{u}^0$, as a first-order approximation of $\mathbf{u}^1$. We leave a detailed investigation of this effect to future work.

5 CONCLUSIONS

We presented a new method for detail-preserving fluid animation that improves on current advection-projection solver: it replaces the energy-dissipating projection applied at the end of a simulation step with an energy-preserving reflection in the middle of the interval.

Compared to existing advection-projection methods, the central advantages of our reflection solver are that *i*) it has provably better energy-conservation properties, and *ii*) it empirically leads to much less vorticity diffusion and preserves more detail for equal computational costs. Furthermore, our method integrates readily with existing grid-based solvers and is very easy to implement.

5.1 Limitations & Future Work

While our reflection solver preserves energy and vorticity very well, it does not do so perfectly. One reason is that the semi-Lagrangian advection step is itself not energy-preserving. Another one is that the projection at the end still removes energy from the system, albeit at a much smaller rate than current advection-projection solvers.

While the improved energy and vorticity behavior generally lead to visually richer animations, our solver does not introduce detail where none should arise according to the true solution. In visual effects, however, it is often desirable to *artificially enhance* results, e.g., by amplifying vorticity or injecting turbulence. Our solver can, in principle, be used on conjunction with such methods and it would be interesting to investigate its behavior in this context. Furthermore, we also like to combine our reflection solver with the recent *advection-enhancing* methods of Zhang et al. [2015] and Chern et al. [2016].

Extensions to simplicial grids are another direction worth exploring. Finally, we would like to apply the reflection solver to other constraint-projection applications such as inextensible cloth [Goldenthal et al. 2007] and volume-preserving solids [Irving et al. 2007] that have so far relied on the step-and-project approach.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their constructive and thoughtful feedback. This work was supported by the Discovery Grants Program of the Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

Alexis Angelidis and Fabrice Neyret. 2005. Simulation of Smoke Based on Vortex Filament Primitives. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '05)*. 87–96. https://doi.org/10.1145/1073368.1073380

Christopher Batty, Florence Bertails, and Robert Bridson. 2007. A Fast Variational Framework for Accurate Solid-fluid Coupling. *ACM Trans. Graph.* 26, 3 (July 2007). https://doi.org/10.1145/1276377.1276502

Albert Chern, Felix Knöppel, Ulrich Pinkall, Peter Schröder, and Steffen Weißmann. 2016. Schrödinger's Smoke. *ACM Trans. Graph.* 35, 4 (July 2016), 77:1–77:13. https://doi.org/10.1145/2897824.2925868

Alexandre Joel Chorin. 1968. Numerical solution of the Navier-Stokes equations. *Math. Comp.* 22 (1968), 745–762. https://doi.org/10.1090/S0025-5718-1968-0242392-2

Sharif Elcott, Yiying Tong, Eva Kanso, Peter Schröder, and Mathieu Desbrun. 2007. Stable, Circulation-preserving, Simplicial Fluids. *ACM Trans. Graph.* 26, 1 (Jan. 2007). https://doi.org/10.1145/1189762.1189766

Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. 2001. Visual Simulation of Smoke. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01)*. 15–22. https://doi.org/10.1145/383259.383260

Nick Foster and Dimitri Metaxas. 1996. Realistic Animation of Liquids. *Graphical Models and Image Processing* 58, 5 (Sept. 1996), 471 – 483. https://doi.org/10.1006/gmip.1996.0039

Rony Goldenthal, David Harmon, Raanan Fattal, Michel Bercovier, and Eitan Grinspun. 2007. Efficient Simulation of Inextensible Cloth. *ACM Trans. Graph.* 26, 3 (July 2007). https://doi.org/10.1145/1276377.1276438

Ernst Hairer, Christian Lubich, and Gerhard Wanner. 2006. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations; 2nd ed.* Springer, Dordrecht. https://cds.cern.ch/record/1250576

Francis H. Harlow and J. Eddie Welch. 1965. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *The Physics of Fluids* 8, 12 (1965), 2182–2189. https://doi.org/10.1063/1.1761178

Markus Ihmsen, Jens Orthmann, Barbara Solenthaler, Andreas Kolb, and Matthias Teschner. 2014. SPH Fluids in Computer Graphics. In *Eurographics 2014 - State of the Art Reports*. https://doi.org/10.2312/egst.20141034

Geoffrey Irving, Craig Schroeder, and Ronald Fedkiw. 2007. Volume Conserving Finite Element Simulations of Deformable Models. *ACM Trans. Graph.* 26, 3 (July 2007). https://doi.org/10.1145/1276377.1276394

Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. 2015. The affine particle-in-cell method. *ACM Transactions on Graphics (TOG)* 34, 4 (July 2015), 51:1–51:10. https://doi.org/10.1145/2766996

ByungMoon Kim, Yingjie Liu, Ignacio Llamas, and Jarek Rossignac. 2005. FlowFixer: Using BFECC for Fluid Simulation. In *Proceedings of the First Eurographics Conference on Natural Phenomena (NPH'05)*. 51–56. https://doi.org/10.2312/NPH/NPH05/051-056

ByungMoon Kim, Yingjie Liu, Ignacio Llamas, and Jarek Rossignac. 2007. Advections with Significantly Reduced Dissipation and Diffusion. *IEEE Transactions on Visualization and Computer Graphics* 13, 1 (Jan. 2007), 135–144. https://doi.org/10.1109/TVCG.2007.3

Doyub Kim, Oh-young Song, and Hyeong-Seok Ko. 2008a. A Semi-Lagrangian CIP Fluid Solver without Dimensional Splitting. *Computer Graphics Forum* 27, 2 (April 2008), 467–475. https://doi.org/10.1111/j.1467-8659.2008.01144.x

Theodore Kim, Nils Thürey, Doug James, and Markus Gross. 2008b. Wavelet Turbulence for Fluid Simulation. *ACM Trans. Graph.* 27, 3 (Aug. 2008), 50:1–50:6. https://doi.org/10.1145/1360612.1360649

B.P. Leonard. 1979. A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Computer Methods in Applied Mechanics and Engineering* 19, 1 (1979), 59 – 98. https://doi.org/10.1016/0045-7825(79)90034-3

Jeroen Molemaker, Jonathan M. Cohen, Sanjit Patel, and Junyong Noh. 2008. Low Viscosity Flow Simulations for Animation. In *SCA '08: Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 9–18. https://doi.org/10.2312/SCA/SCA08/009-018

Patrick Mullen, Keenan Crane, Dmitry Pavlov, Yiying Tong, and Mathieu Desbrun. 2009. Energy-preserving Integrators for Fluid Animation. *ACM Trans. Graph.* 28, 3 (July 2009), 38:1–38:8. https://doi.org/10.1145/1531326.1531344

Sang Il Park and Myoung Jun Kim. 2005. Vortex Fluid for Gaseous Phenomena. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '05)*. 261–270. https://doi.org/10.1145/1073368.1073406

André Robert. 1981. A stable numerical integration scheme for the primitive meteorological equations. *Atmosphere-Ocean* 19, 1 (1981), 35–46. https://doi.org/10.1080/07055900.1981.9649098

Andrew Selle, Ronald Fedkiw, Byungmoon Kim, Yingjie Liu, and Jarek Rossignac. 2008. An Unconditionally Stable MacCormack Method. *J. Sci. Comput.* 35, 2-3 (June 2008), 350–371. https://doi.org/10.1007/s10915-007-9166-4

Andrew Selle, Nick Rasmussen, and Ronald Fedkiw. 2005. A vortex particle method for smoke, water and explosions. *ACM Transactions on Graphics (TOG)* 24, 3 (July 2005), 910–914. https://doi.org/10.1145/1073204.1073282

Jos Stam. 1999. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. 121–128. https://doi.org/10.1145/311535.311548

Nils Thuerey, Theodore Kim, and Tobias Pfaff. 2013. Turbulent fluids. In *ACM SIGGRAPH 2013 Courses*. ACM, 6.

Nils Thuerey and Tobias Pfaff. 2016. Mantaflow. http://mantaflow.com. (2016).

Steffen Weißmann and Ulrich Pinkall. 2010. Filament-based Smoke with Vortex Shedding and Variational Reconnection. *ACM Trans. Graph.* 29, 4 (July 2010), 115:1–115:12. https://doi.org/10.1145/1778765.1778852

Xinxin Zhang, Robert Bridson, and Chen Greif. 2015. Restoring the Missing Vorticity in Advection-projection Fluid Solvers. *ACM Trans. Graph.* 34, 4 (July 2015), 52:1–52:8. https://doi.org/10.1145/2766982

Yongning Zhu and Robert Bridson. 2005. Animating Sand as a Fluid. 24 (July 2005), 965–972. https://doi.org/10.1145/1073204.1073298