Line Integral Fluids

RUPESH KUMAR, Indian Institute of Technology, Delhi, India
RAHUL NARAIN, Indian Institute of Technology, Delhi, India

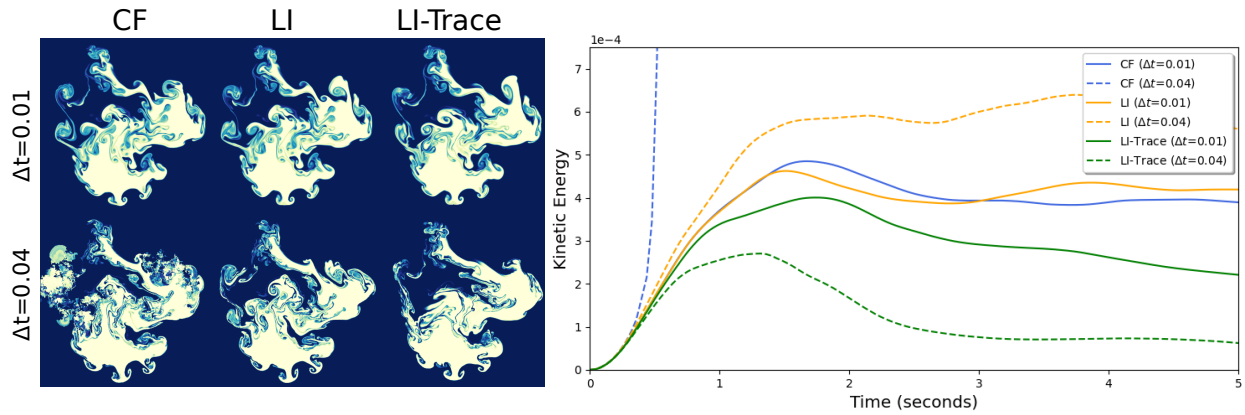

Fig. 1. An ink drop in the shape of the SIGGRAPH logo sinks under the influence of gravity. For a time step of $\Delta t = 0.01$ s, our line integral (LI) method gives similar visual quality as the covector fluids (CF) method of Nabizadeh et al. [2022]. However, at a larger time step of $\Delta t = 0.04$ s, CF rapidly becomes unstable while LI still gives stable results. Our LI-Trace variant adds a simple heuristic to provide greater stability, in this case eliminating the erroneous increase in energy in the basic LI method.

We introduce a new advection scheme for grid-based fluid simulation inspired by the recent covector fluids approach of Nabizadeh et al. [2022]. We directly discretize the integral form of the covector transport equation, which ensures the conservation of line integral over any curve in the discretized fluid domain. Despite a theoretical argument for unconditional stability in 2D, we observe that numerical error in backtracing results in limited stability in practice. To address this issue, we further propose a stabilization heuristic based on the relative change in area of backtraced grid cells. We extend the heuristic to volume ratios in 3D and show that the proposed method is stable even at larger timesteps.

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

1 Introduction

Simulating the visually rich and complex motion of fluids remains an important and challenging task in computer graphics. A standard approach on Eulerian grids is to a semi-Lagrangian scheme for velocity self-advection, followed by a pressure projection to maintain incompressibility. The splitting error introduced by this process causes significant artificial dissipation of circulation around vortices.

Recently, Nabizadeh et al. [2022] introduced a new "covector fluids" (CF) method where the fluid velocity is interpreted as a covector field. Then, advection is performed using the Lie derivative, essentially acting as a pullback of the covector field via the flow map from one time step to the next. They show that this reformulation leads to an advection term that exactly preserves circulation. In practice, after spatial and temporal discretization, circulation preservation may not exactly hold. Moreover, the method is not unconditionally stable, and in practice it becomes unstable at larger time steps.

Authors' Contact Information: Rupesh Kumar, Indian Institute of Technology, Delhi, India; Rahul Narain, Indian Institute of Technology, Delhi, India.

In this work, we propose an alternative discretization of the CF approach that seeks to preserve circulation at a discrete level. Specifically, we perform covector advection by ensuring that the line integrals along dual grid edges are preserved under the flow. This modification further enables an interpretation involving advection of dual edge loops that provides insight into its stability properties. Motivated by this viewpoint, we also propose a simple heuristic that stabilizes the method for large time steps.

Specifically, we make the following two contributions:

- We present a novel discretization of the covector fluids formulation that preserves circulation not just infinitesimally but at the level of grid edges.
- We provide a simple local heuristic that estimates and compensates for the numerical error in backtracing, allowing for stable fluid simulations at large time steps.

The rest of the paper is organised as follows. Existing work related to fluid simulation are described in Section 2. We give a background on covector transport and the covector fluids method in Section 3. Section 4 describes our proposed method. The results of our method are shown in Section 5, and we conclude with a discussion of limitations and future work in Section 6.

2 Related Work

The computer graphics literature has seen an incredible variety of techniques for fluid simulation, including methods using Eulerian grids, Lagrangian methods such as smoothed particle hydrodynamics [Koschier et al. 2022], hybrid particle-in-cell methods [Jiang et al. 2016; Zhu and Bridson 2005a], kinetic solvers using the lattice Boltzmann method e.g. [Li et al. 2023], vorticity-based techniques, and

so on. In this section, we focus only on grid-based methods since those are the most closely related to our work.

The vast majority of grid-based fluid solvers in computer graphics use a splitting scheme, in which the self-advection of the fluid is carried out using semi-Lagrangian advection, after which a pressure projection step makes the velocity field divergence-free. This approach, introduced to graphics by Stam [1999], has the desirable property of being unconditionally stable. However, it suffers from significant artificial dissipation, and a great deal of subsequent work has sought to address this limitation.

One major source of dissipation is the numerical diffusion caused by semi-Lagrangian advection. This can be mitigated by the use of higher-order spatial interpolation [Fedkiw et al. 2001; Losasso et al. 2006]. Error correction schemes such as BFECC [Dupont and Liu 2003; Kim et al. 2005] and the MacCormack method [Selle et al. 2008] attempt to estimate and correct for the diffusion error by performing advection backwards and forwards. Alternatively, one can reduce the error by performing interpolation less frequently, through the use of characteristic mappings that track the fluid motion over multiple time steps [Qu et al. 2019; Sato et al. 2018; Tessendorf and Pelfrey 2011]. Finally, particle-in-cell methods [Fu et al. 2017; Jiang et al. 2015; Zhu and Bridson 2005b] carry information directly on moving particles and thus do not accumulate interpolation error.

On the other hand, the splitting between advection and pressure projection causes artificial dissipation as well, because self-advection of velocity transfers kinetic energy into divergent modes which are then annihilated by the pressure step. Early on, Fedkiw et al. [2001] introduced vorticity confinement to graphics, which adds an artificial centripetal force to counteract the loss of vorticity. Mullen et al. [2009] introduced a time-reversible implicit integrator which exactly preserves energy at the cost of a nonlinear Newton step at each time step. A more efficient advection-reflection scheme was introduced by Zehnder et al. [2018] which applies pressure at the middle of the time step rather than at the end.

Recently, Nabizadeh et al. [2022] introduced a novel formulation of grid-based fluid simulation that eliminates the splitting error between advection and projection. Instead of semi-Lagrangian advection of velocity vectors, their method treats the fluid velocity as a covector and performs advection using the Lie derivative. As our work builds directly on this method, we discuss it in detail in the following section.

3 Background

We consider the motion of an incompressible, inviscid fluid over a d-dimensional domain $M \subseteq \mathbb{R}^d$ (where $d = 2$ or 3), given by the Euler equations

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p, \tag{1}$$

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

Here $\mathbf{u} : M \to \mathbb{R}^d$ is the fluid's velocity field, $p : M \to \mathbb{R}$ is the kinematic pressure, and $\nabla\cdot$ is the divergence operator. Nabizadeh et al. [2022] show that these equations can be solved elegantly by treating the velocity as a covector field, as we discuss below.

3.1 Covector Transport

Consider the motion of the fluid from time 0 to time t. For clarity, we represent quantities at time 0 and t as living on the initial domain M^0 or the current domain M^t respectively, even though the domains are usually equal for a static scene. In particular, let $\Phi^t : M^0 \to M^t$ be the flow map from M^0 to M^t, such that a point $\mathbf{x}^0 \in M^0$ advected by the velocity field for time t arrives at $\mathbf{x}^t = \Phi^t(\mathbf{x}^0) \in M^t$. Also let $\Psi^t = (\Phi^t)^{-1}$ denote the inverse flow map from M^t to M^0.

Thinking of a tangent vector $\mathbf{v}^0 \in T_{\mathbf{x}^0}M^0$ as an infinitesimal displacement of $\mathbf{x}^0$, it is easy to define its counterpart at time t via the property $\Phi^t(\mathbf{x}^0 + \epsilon\mathbf{v}^0) = \mathbf{x}^t + \epsilon\mathbf{v}^t + o(\epsilon)$, implying that

$$\mathbf{v}^t = d\Phi^t(\mathbf{x}^0)\mathbf{v}^0 \tag{3}$$

where $d\Phi^t$ is the Jacobian of Φ^t. Transportation of a covector $\boldsymbol{\xi}^0 \in T^*_{\mathbf{x}}M^0$ is characterized by invariance of its action on vectors, i.e. $\boldsymbol{\xi}^t\mathbf{v}^t = \boldsymbol{\xi}^0\mathbf{v}^0$ for all $\mathbf{v}^0 \in T_{\mathbf{x}^0}M^0$. This leads to the pullback formula

$$\boldsymbol{\xi}^t = (d\Phi^t(\mathbf{x}^0))^{-\mathsf{T}}\boldsymbol{\xi}^0 = (d\Psi^t(\mathbf{x}^t))^{\mathsf{T}}\boldsymbol{\xi}^0. \tag{4}$$

As Nabizadeh et al. [2022] point out, covector transport is equivalent to ensuring that the line integral of $\boldsymbol{\xi}$ along any curve pushed by the flow stays constant over time. That is, given a curve $C^0 : [a, b] \to M^0$ with $C^t = \Phi^t \circ C^0$, we have

$$\int_{C^t} \boldsymbol{\xi}^t(\mathbf{x}^t)d\mathbf{x}^t = \int_{C^0} \boldsymbol{\xi}^0(\mathbf{x}^0)d\mathbf{x}^0. \tag{5}$$

3.2 Covector Fluids

In the covector fluids algorithm [Nabizadeh et al. 2022] the fluid velocity is treated as a covector field, and advection is performed using the pullback from time t to $t + \Delta t$,

$$\mathbf{u}^{t+\Delta t}(\mathbf{x}) = (d\Psi(\mathbf{x}))^{\mathsf{T}}\mathbf{u}^t(\Psi(\mathbf{x})) \tag{6}$$

where the backwards flow map $\Psi : M^{t+\Delta t} \to M^t$ gives the back-traced position at the start of the time step. We note that this is *not* a discretization of the standard advection equation, $\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} = 0$, and covector advection by itself is not an alternative to semi-Lagrangian advection. Rather, it corresponds to a different formulation of the Euler equations,

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + (\nabla \mathbf{u}) \cdot \mathbf{u} = -\nabla \left(p - \tfrac{1}{2}\|\mathbf{u}\|^2\right), \tag{7}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{8}$$

in which the left-hand side of eq. (7) measures the rate of change of the *line integrals* of the velocity covector field, rather than the rate of change of linear momentum.

In the covector fluids algorithm, the advection step takes the right-hand side to be zero and hence updates the velocity field using eq. (6), after which the pressure projection step computes the so-called Lagrangian pressure $\lambda = p - \frac{1}{2}\|\mathbf{u}\|^2$ to restore incompressibility. Since the advection step preserves line integrals of velocity, it preserves circulation over flowing closed loops by design. Remarkably, it can be shown [Nabizadeh et al. 2022] that in this scheme the advection and pressure projection steps commute, thus eliminating the splitting error in the traditional advection-projection approach.

In practice, eq. (6) is enforced only on velocity samples at MAC grid faces, with $d\Psi$ approximated using finite differences. Due to discretization error, the circulation preservation property may not

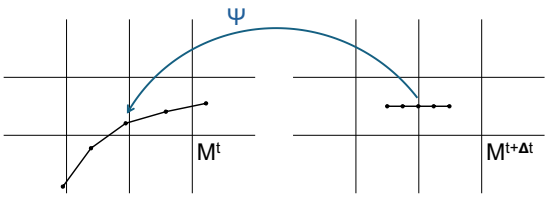

Fig. 2. We perform advection by computing line integrals over the backtraced dual grid edges.

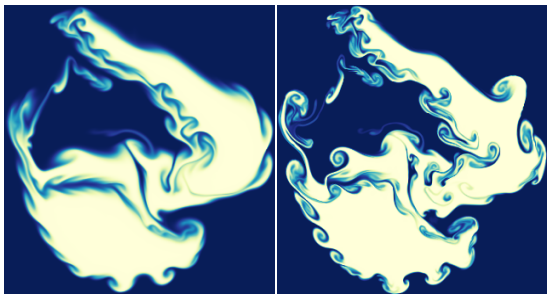

Fig. 3. An ink drop in the shape of the SIGGRAPH logo sinks under the influence of gravity. The left figure uses the LI scheme without BFECC, whereas the right one uses LI along with BFECC.

hold exactly. Furthermore, Nabizadeh et al. [2022] report that the scheme is only conditionally stable; while it allows larger time steps than previous circulation-preserving methods, it does not provide the desirable unconditional stability of semi-Lagrangian methods. In our experiments, we observed the same instability at larger time steps and higher spatial resolutions in both 2D and 3D simulations.

4 Method

In our work, we retain the covector formulation of the Euler equations (7)-(8) and seek a more numerically stable discretization. Our key idea is that the essential property of covector advection is line integral invariance (5), since that is what ensures circulation preservation around vortical structures. Rather than expressing it in differential form via eq. (6) and then discretizing the derivative, we aim to directly impose eq. (5) in the discrete update scheme. In particular, we wish to satisfy

$$\int_C \mathbf{u}^{t+\Delta t}(\mathbf{x}) \cdot d\mathbf{x} = \int_{\Psi(C)} \mathbf{u}^t(\mathbf{x}) \cdot d\mathbf{x} \tag{9}$$

for an appropriate set of curves C defined on the simulation grid.

4.1 Line Integral Advection

Let us consider a velocity field $\mathbf{u}$ discretized on a staggered MAC grid [Fedkiw et al. 2001] with spacing h. Any velocity sample u_i stored at a grid face centered at $\mathbf{x}_i$ can be approximated by a line integral,

$$u_i \approx \frac{1}{h} \int_{C_i} \mathbf{u}(\mathbf{x}) \cdot d\mathbf{x}, \tag{10}$$

where the curve C_i is the line segment joining the two adjacent grid cell centers, i.e. $\mathbf{x}_i - \frac{1}{2}h\mathbf{e}_i$ and $\mathbf{x}_i + \frac{1}{2}h\mathbf{e}_i$ where $\mathbf{e}_i$ is the unit vector along the direction of the velocity component u_i. In particular, when using nearest neighbour interpolation, eq. (10) is an exact equality, although with linear or higher-order interpolation, the integral introduces a small amount of diffusion from adjacent samples. Using eq. (9) and eq. (10), we compute the new velocity using the line integral of velocity over the backtraced line segment,

$$u_i^{t+\Delta t} = \frac{1}{h} \int_{\Psi(C_i)} \mathbf{u}^t(\mathbf{x}^t) \cdot \mathrm{d}\mathbf{x}^t. \tag{11}$$

Thus, we have replaced the finite difference in the implementation of eq. (6) with a more numerically stable integral. In our implementation, the backwards flow map Ψ is precomputed using RK4 time integration and stored on the grid, so sampling $\Psi(C_i)$ does not have any additional cost. We found it sufficient to compute the integral using the trapezoidal rule with $n = 4$ segments. Fig. 2 illustrates the backtracing process used for our line integral advection.

It is interesting to note that the curves C_i on which we preserve the line integral are exactly the edges of the dual grid whose vertices are grid cell centers. Thus, our approach aims to discretely preserve circulation on all loops in the dual grid, i.e. all loops formed by connecting adjacent cell centers of the primal simulation grid. This is in contrast to the covector fluid method, where circulation preservation is only guaranteed in continuous space before discretization.

In practice, we use linear interpolation to reconstruct the velocity field $\mathbf{u}(\mathbf{x})$ from the grid face samples. This results in a small amount of artificial dissipation in eq. (10). We considered addressing this issue either by modifying the interpolation of the old velocities, or by post-filtering the new velocities, in either case with the goal of exactly matching the sampled values to the desired line integrals. However, we found it much easier to suppress such dissipation error simply by using the back-and-forth error compensation and correction (BFECC) scheme [Kim et al. 2005]. Thus, we employ BFECC in our line integral advection, and do not apply any further processing. Finally to reduce the truncation error in timestepping, we employ the midpoint method: we first perform and advection projection step upto $\Delta t/2$ to estimate the midstep flow velocity, and then use it for a full step of advection over Δt (see [Nabizadeh et al. 2022] Algorithm 2). The results in Fig. 3 show that BFECC alleviates the diffusion issue and allows fine-scale vortices to emerge.

4.2 Improving Stability

The following simple argument shows that in principle, fluid simulation using our line integral advection scheme should be unconditionally stable in 2D. Our advection conserves the circulation along any loop of dual grid edges, and since the area of the loop should also be conserved in an incompressible 2D flow, the vorticity on the grid cannot increase or decrease but must simply be moved around by the flow. Fig. 4 illustrates this argument.

However, in practice we found that our method, like the covector fluids algorithm, does become unstable at larger time steps. We found that this is because due to numerical errors in the computation of Ψ, areas are not actually conserved: a loop after backtracing may enclose a larger area, resulting in an increase of vorticity. This

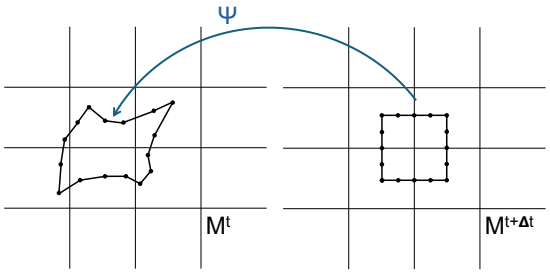

Fig. 4. Line integral advection ensures that the circulation along the dual edge loop is equal to the circulation along the backtraced loop. In an incompressible 2D fluid, the area enclosed by both loops should also be equal, thus the vorticity (i.e. circulation per unit area) should be conserved.

problem is difficult to eliminate: incompressible velocity interpolation [Chang et al. 2022] may help, but correcting Ψ to exactly conserve area would require an expensive global solve over the entire grid.

Instead, we propose a simple local heuristic that significantly improves stability. Consider a loop C consisting of 4 adjacent dual grid edges, enclosing an area $a = h^2$, and let a' be the area enclosed by the backtraced loop $\Psi(C)$. If the area ratio $r = a'/a$ is greater than 1, this indicates a numerical error that may lead to instability. To counteract the change in area, we imagine uniformly scaling down the loop by $1/\sqrt{r}$; consequently, the backtraced edges would become shorter by a factor of $1/\sqrt{r}$, and so would the corresponding line integrals. This motivates our *area ratio heuristic*: for each velocity sample u_i, after we compute the line integral I as the right-hand side of eq. (11), we also compute the area ratios r_1, r_2 for the two adjacent dual edge loops, and set $u_i^{t+\Delta t} = I/\sqrt{\max(r_1, r_2, 1)}$.

We observed that the area ratio heuristic does significantly improve stability, but being an approximation, it does not guarantee unconditional stability either. If a completely stable simulation is desired, we offer a more conservative strategy called the *trace heuristic*, as follows. Let $\mathbf{F} = d\Psi$ be the deformation gradient, i.e. the Jacobian of the backwards flow map, and observe that the area ratio r is essentially its determinant, $\det \mathbf{F}$. In two dimensions, the determinant has a simple upper bound, $\det \mathbf{F} \leq \frac{1}{2} \operatorname{tr}(\mathbf{F}^T\mathbf{F})$; the right-hand side corresponds to the change in the average squared length of the edges of the loop. Therefore, for a more conservative stabilization, we compute r for a loop as the ratio of its average squared edge lengths before and after backtracing instead of the ratio of its enclosed areas. We observed empirically that this approach resulted in stable simulations at large time steps in all of our examples. It can also be shown that since $\mathbf{F} \approx \mathbf{I} - d\mathbf{u}\Delta t$, the two quantities only differ from each other (and from 1) by $O(\Delta t^2)$; thus, this heuristic does not degrade the first-order accuracy of the simulation.

Our stabilization approach can be extended readily to 3D as well, since we can still observe that Ψ incorrectly fails to conserve volume. For any dual grid cell we may compute the volume ratio $r = v'/v$ of the backtraced volume to the current volume, and for any velocity sample we scale the line integrals by $1/\sqrt[3]{\max(r_i, 1)}$ where the maximum is taken over the four adjacent dual grid cells. More conservatively, we derive the trace heuristic from the corresponding inequality, $\det \mathbf{F} \leq \left(\frac{1}{3} \operatorname{tr}(\mathbf{F}^T\mathbf{F})\right)^{3/2}$. Therefore, we compute the average of all 12 squared edge lengths before and after backtracing respectively, say s and s', and define $r = (s'/s)^{3/2}$ for scaling the line integrals. In 3D as well, the trace heuristic provided stability over all time steps and all examples that we tested.

We refer to our algorithm with the trace heuristic as LI-Trace. As we show in section 5, results using LI-Trace provide significantly better stability than the covector fluids scheme, while exhibiting comparable visual quality. Therefore, we consider LI-Trace as our base advection scheme. Implementing LI-Trace is a straightforward process, and it can be easily integrated into the fluid simulation framework presented by Nabizadeh et al. [2022] in place of the covector advection step.

Example	Figure	Size	Resolution	Δt
Taylor vortices	Fig. 5	$2\pi \times 2\pi\ \mathrm{m}^2$	256×256	1/40 s
SIGGRAPH logo	Fig. 6	$0.2 \times 0.25\ \mathrm{m}^2$	420×520	1/100 s
	Fig. 1			1/25 s
Trefoil knot	Fig. 7	$10 \times 5 \times 5\ \mathrm{m}^3$	$256 \times 128 \times 128$	1/48 s
Smoke plume	Fig. 8	$5 \times 10 \times 5\ \mathrm{m}^3$	$128 \times 256 \times 128$	1/192 s
	Fig. 9			1/96 s

Table 1. Resolution and time step for all examples.

5 Results

In this section, we evaluate the accuracy, stability, and visual quality of our method on various examples. Our implementation is a modified version of the codebase provided by Nabizadeh et al. [2022]. All the experiments are run using midpoint timestepping and BFECC, except for the result shown in Fig. 3 (left) which omits BFECC. The experiments are run on a desktop machine with 3.60GHz 4-core Intel Core i7-4790 processor and 16GB memory. The 3D experiments are parallelized using CUDA running on an Nvidia GTX 1070 GPU with 8GB dedicated memory. All the experiments are adapted from Nabizadeh et al. [2022]. For a fair comparison we use the same settings that they provide, except for increasing the time step size. Details of spatial resolution and time step for all examples are given in Table 1.

5.1 2D Examples

We evaluate the accuracy of our method on the classical Taylor vortices example. In Fig. 5 we show that the energy preservation of our line integral scheme is comparable to that of covector fluids. The initial increase in energy in the covector fluids method arises directly from the authors' provided implementation, and we were not able to pinpoint its cause. Enabling the trace heuristic results in some additional dissipation, but the behaviour is qualitatively similar.

In Fig. 6 we show a simulation of an ink drop initialized in the shape of the SIGGRAPH logo surrounded by a less dense fluid. This example shows that our method gives similar visual quality as covector fluids. Despite the additional dissipation caused by LI-Trace,

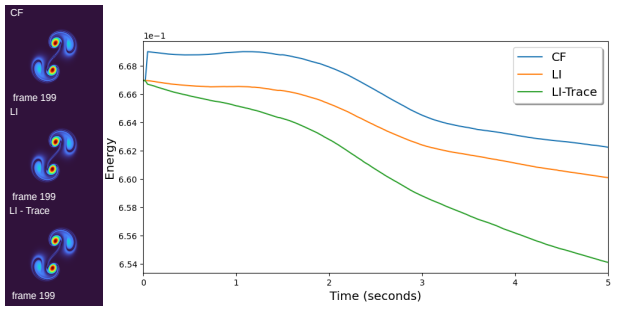

Fig. 5. Taylor vortices with $\Delta t = 0.025$. Our scheme provides results with comparable quality and similar energy conservation as covector fluids. Using the trace heuristic for stabilization causes a moderate loss of energy.

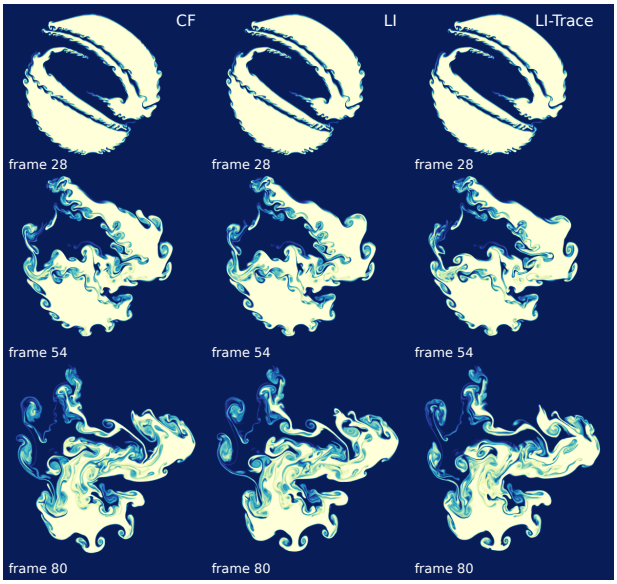

Fig. 6. Evolution of the SIGGRAPH logo example with $\Delta t = 0.01$. The proposed LI and LI-Trace methods show similar visual quality as CF at this small time step size.

Time step (Δt) →	10^{-3}	10^{-2}	10^{-1}
Example\Scheme	**CF/LI/LI-Trace**	**CF/LI/LI-Trace**	**CF/LI/LI-Trace**
Taylor Vortex	✓/✓/✓	✓/✓/✓	✓/✓/✓
Vortex Leapfrogging Pairs	✓/✓/✓	✓/✓/✓	×/✓/✓
2D Ink Drop	✓/✓/✓	✓/✓/✓	×/×/✓
SIGGRAPH Logo	✓/✓/✓	✓/✓/✓	×/×/✓

Table 2. Table showing the stability of the 2D simulations for timesteps of 0.1,0.01 and 0.001 seconds for different examples.

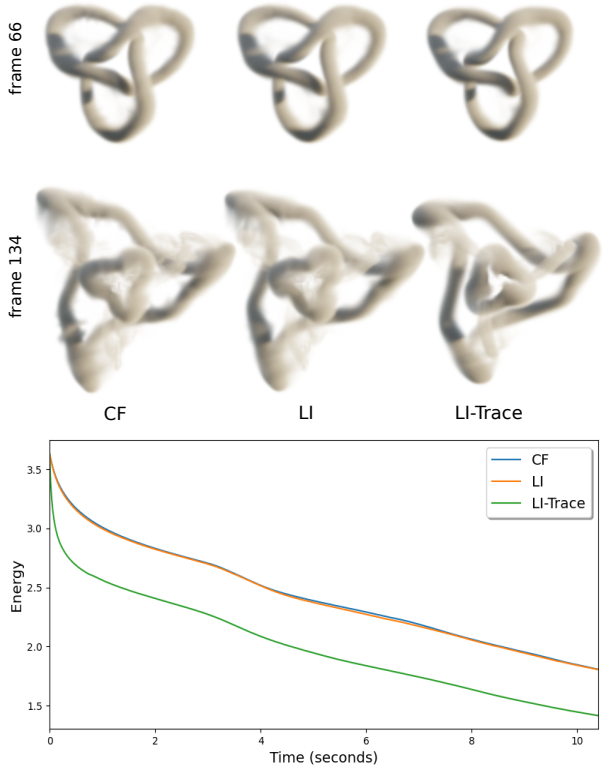

Fig. 7. Trefoil Knot simulation results. The simulations are run with $\Delta t = 1/24$ with 2 substeps per frame. It can be observed that the LI scheme provides same energy preservation as CF. However, LI-Trace loses energy.

the qualitative difference in the visual appearance is unnoticeable in this example.

The key advantage of our method over covector fluids is its stability at larger time steps. In Fig. 1 we ran the SIGGRAPH logo example with two different choices of time step, 0.01 s and 0.04 s. It can be observed that all three methods give similar results at the smaller time step, showing that our methods are consistent with the behaviour of covector fluids. However, at the larger time step, the covector fluids scheme rapidly becomes unstable while our methods remain stable.

We further investigated the stability of the three methods on various examples over a wide range of time steps. The results are shown in Table 2. We observe that as the time step increases, first covector fluids and then the basic line integral method become unstable, whereas with the trace heuristic the line integral method remains stable at the largest time step for all examples.

5.2 3D Examples

To evaluate the accuracy of our method in 3D, we ran the trefoil knot example in which the fluid is initialized with a knotted vorticity distribution. As can be seen in Fig. 7, without trace stabilization, our LI formulation offers identical energy preservation as covector fluids. However, LI-Trace does exhibit some additional dissipation.

For a qualitative evaluation of visual quality, we simulate a plume of buoyant smoke emitted from a spherical source. Fig. 8 shows the results using all three algorithms with a time step of 1/192 s. Both the CF and LI methods yield highly detailed, turbulent plumes, while the addition of the trace heuristic slightly reduces the amount

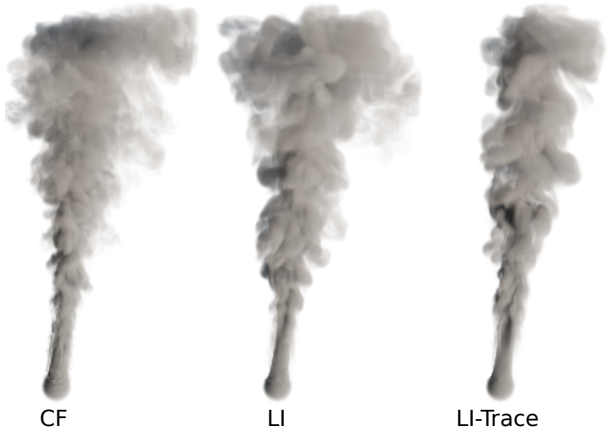

Fig. 8. Smoke plume simulation results. The simulations are run with $\Delta t = 1/24$ with 8 substeps per frame.. We observe that LI scheme provides similar quality as CF, however LI-Trace is more dissipative.

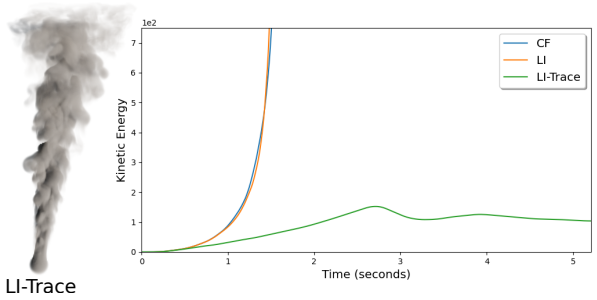

Fig. 9. Smoke plume energy plots for a higher timestep simulation. The simulations are run with $\Delta t = 1/24$ with 4 substeps per frame. We observe that both CF and LI schemes go unstable just by doubling the Δt, whereas the LI-Trace simulation shows stable energy values.

of turbulence. The value of the trace heuristic becomes apparent when we increase the time step by only a factor of 2, as shown in Fig. 9. Here both CF and LI become unstable, while LI-Trace remains stable and continues to provide high-quality results.

6 Conclusion

We have presented a method for fluid simulation based on imposing circulation preservation at the discrete level. Our method exploits the connection between velocity samples in a staggered MAC grid and line integrals of velocity along dual grid edges. We perform advection by backtracing the dual edges to the previous time step and pulling back the velocity line integrals, ensuring circulation preservation along all edge loops in the dual grid. Finally, we introduced a simple local heuristic that counteracts instability arising from numerical error in the backward flow map, and provides stable simulations at large time steps.

6.1 Limitations and Future Work

We have tested our method using the trace heuristic on various examples over a wide range of time steps, and observed that it remains stable in all cases. However, we do not have a formal proof of stability. In addition, it is clear that the trace heuristic is overly conservative and results in noticeable loss of energy in the 3D examples.

In future work, we hope to investigate other stabilization techniques for line integral advection. For example, it would be interesting to see if modifying the backward flow map to exactly conserve area results in unconditional stability in 2D simulations. We also wish explore other local schemes that may exhibit less dissipation than the trace heuristic.

Since our method requires only a backward flow map from the current fluid domain to a reference domain in the past, it should readily fit in with methods based on characteristic mapping [Qu et al. 2019; Sato et al. 2018; Tessendorf and Pelfrey 2011]; one would only need to compute line integrals in the reference domain rather than the previous time step. We are enthusiastic about exploring this combination in the future.

References

Jumyung Chang, Ruben Partono, Vinicius C. Azevedo, and Christopher Batty. 2022. Curl-Flow: Boundary-Respecting Pointwise Incompressible Velocity Interpolation for Grid-Based Fluids. *ACM Trans. Graph.* 41, 6, Article 243 (nov 2022), 21 pages. doi:10.1145/3550454.3555498

Todd F. Dupont and Yingjie Liu. 2003. Back and forth error compensation and correction methods for removing errors induced by uneven gradients of the level set function. *J. Comput. Phys.* 190, 1 (2003), 311–324. doi:10.1016/S0021-9991(03)00276-6

Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. 2001. Visual Simulation of Smoke *(SIGGRAPH '01)*. Association for Computing Machinery, New York, NY, USA, 15–22. doi:10.1145/383259.383260

Chuyuan Fu, Qi Guo, Theodore Gast, Chenfanfu Jiang, and Joseph Teran. 2017. A Polynomial Particle-in-Cell Method. *ACM Trans. Graph.* 36, 6, Article 222 (nov 2017), 12 pages. doi:10.1145/3130800.3130878

Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. 2015. The Affine Particle-in-Cell Method. *ACM Trans. Graph.* 34, 4, Article 51 (jul 2015), 10 pages. doi:10.1145/2766996

Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. 2016. The material point method for simulating continuum materials. In *ACM SIGGRAPH 2016 Courses* (Anaheim, California) *(SIGGRAPH '16)*. Association for Computing Machinery, New York, NY, USA, Article 24, 52 pages. doi:10.1145/2897826.2927348

ByungMoon Kim, Yingjie Liu, Ignacio Llamas, and Jarek Rossignac. 2005. FlowFixer: Using BFECC for Fluid Simulation. In *Eurographics Workshop on Natural Phenomena*, Pierre Poulin and Eric Galin (Eds.). The Eurographics Association. doi:10.2312/NPH/NPH05/051-056

Dan Koschier, Jan Bender, Barbara Solenthaler, and Matthias Teschner. 2022. A Survey on SPH Methods in Computer Graphics. *Computer Graphics Forum* 41, 2 (2022), 737–760. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14508 doi:10.1111/cgf.14508

Wei Li, Tongtong Wang, Zherong Pan, Xifeng Gao, Kui Wu, and Mathieu Desbrun. 2023. High-Order Moment-Encoded Kinetic Simulation of Turbulent Flows. 42, 6, Article 190 (dec 2023), 13 pages. doi:10.1145/3618341

Frank Losasso, Ronald Fedkiw, and Stanley Osher. 2006. Spatially adaptive techniques for level set methods and incompressible flow. *Computers and Fluids* 35, 10 (2006), 995–1010. doi:10.1016/j.compfluid.2005.01.006

Patrick Mullen, Keenan Crane, Dmitry Pavlov, Yiying Tong, and Mathieu Desbrun. 2009. Energy-Preserving Integrators for Fluid Animation. *ACM Trans. Graph.* 28, 3, Article 38 (jul 2009), 8 pages. doi:10.1145/1531326.1531344

Mohammad Sina Nabizadeh, Stephanie Wang, Ravi Ramamoorthi, and Albert Chern. 2022. Covector Fluids. *ACM Trans. Graph.* 41, 4, Article 113 (jul 2022), 16 pages. doi:10.1145/3528223.3530120

Ziyin Qu, Xinxin Zhang, Ming Gao, Chenfanfu Jiang, and Baoquan Chen. 2019. Efficient and Conservative Fluids Using Bidirectional Mapping. *ACM Trans. Graph.* 38, 4, Article 128 (jul 2019), 12 pages. doi:10.1145/3306346.3322945

Takahiro Sato, Christopher Batty, Takeo Igarashi, and Ryoichi Ando. 2018. Spatially adaptive long-term semi-Lagrangian method for accurate velocity advection. *Computational Visual Media* 4, 3 (01 Sep 2018), 223–230. doi:10.1007/s41095-018-0117-9

Andrew Selle, Ronald Fedkiw, ByungMoon Kim, Yingjie Liu, and Jarek Rossignac. 2008. An Unconditionally Stable MacCormack Method. *Journal of Scientific Computing* 35, 2 (01 Jun 2008), 350–371. doi:10.1007/s10915-007-9166-4

Jos Stam. 1999. Stable Fluids. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., USA, 121–128. doi:10.1145/311535.311548

Jerry Tessendorf and Brandon Pelfrey. 2011. The characteristic map for fast and efficient vfx fluid simulations. In *Computer Graphics International Workshop on VFX, Computer Animation, and Stereo Movies. Ottawa, Canada.*

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

Yongning Zhu and Robert Bridson. 2005a. Animating sand as a fluid. *ACM Trans. Graph.* 24, 3 (jul 2005). doi:10.1145/1073204.1073298

Yongning Zhu and Robert Bridson. 2005b. Animating Sand as a Fluid. *ACM Trans. Graph.* 24, 3 (jul 2005), 965–972. doi:10.1145/1073204.1073298