Clebsch Gauge Fluid

SHUQI YANG, Dartmouth College
SHIYING XIONG*, Dartmouth College
YAORUI ZHANG, Dartmouth College
FAN FENG, Dartmouth College
JINYUAN LIU, Dartmouth College
BO ZHU, Dartmouth College

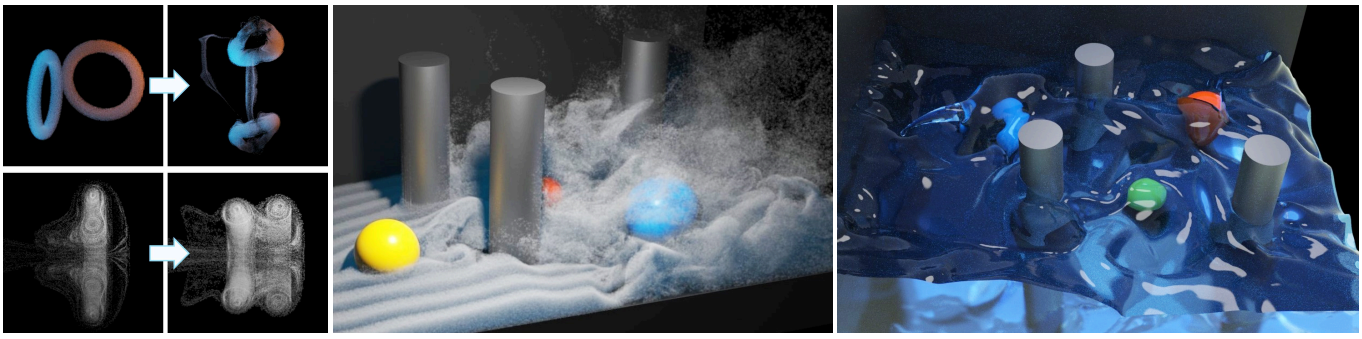


Fig. 1. Our Clebsch gauge method can be used to simulate various fluid scenarios, including complex vortex filaments dynamics (Left: leapfrogging and oblique ring collision), fluids with different obstacles (Middle: smoke simulation), and surface-tension flow with turbulence (Right: flowing water simulation).

We propose a novel gauge fluid solver based on Clebsch wave functions to solve incompressible fluid equations. Our method combines the expressive power of Clebsch wave functions to represent coherent vortical structures and the generality of gauge methods to accommodate a broad array of fluid phenomena. By evolving a transformed wave function as the system's gauge variable enhanced by an additional projection step to enforce pressure jumps on the free boundaries, our method can significantly improve the vorticity generation and preservation ability for a broad range of gaseous and liquid phenomena. Our approach can be easily implemented by modifying a standard grid-based fluid simulator. It can be used to solve various fluid dynamics, including complex vortex filament dynamics, fluids with different obstacles, and surface-tension flow.

CCS Concepts: • **Computing methodologies**; • **Modeling and simulation**;

Additional Key Words and Phrases: Gauge method, Clebsch wave function, vorticity confinement, fluid simulation

*Corresponding author

Authors' addresses: Shuqi Yang, Computer Science Department, Dartmouth College, shuqi.yang.gr@dartmouth.edu; Shiying Xiong, Computer Science Department, Dartmouth College, shiying.xiong@dartmouth.edu; Yaorui Zhang, Computer Science Department, Dartmouth College, yaorui.zhang.gr@dartmouth.edu; Fan Feng, Computer Science Department, Dartmouth College, fan.feng.gr@dartmouth.edu; Jinyuan Liu, Computer Science Department, Dartmouth College, jinyuan.liu.gr@dartmouth.edu; Bo Zhu, Computer Science Department, Dartmouth College, bo.zhu@dartmouth.edu.

0730-0301/2021/8-ART99 $15.00
https://doi.org/10.1145/3450626.3459866

ACM Reference Format:
Shuqi Yang, Shiying Xiong, Yaorui Zhang, Fan Feng, Jinyuan Liu, and Bo Zhu. 2021. Clebsch Gauge Fluid. *ACM Trans. Graph.* 40, 4, Article 99 (August 2021), 11 pages. https://doi.org/10.1145/3450626.3459866

1 INTRODUCTION

Generating and preserving coherent vortical structures are essential for visual fluid simulations. A large variety of previous work in the computer graphics community has been devoted to generate, evolve, and enhance such visually attractive flow structures in a numerical setting. The past efforts, albeit too extensive to conduct a full survey here, can be categorized into three main directions according to the mathematical equations they solved: (i) enhancing the vorticity-preserving capability of a *standard* or *simplified* fluid solver by employing novel numerical techniques such as advection schemes and auxiliary data structures (e.g., see [Selle et al. 2008; Zhu and Bridson 2005]); (ii) solving a *mildly modified* version of fluid equations by adding additional terms such as a vorticity-confinement force (e.g., see [Fedkiw et al. 2001; Foster and Fedkiw 2001]), or solving *a new set of equations* exhibiting visually congruent flow behaviors such as the incompressible Schrödinger's equations [Chern et al. 2016]; (iii) solving a *transformed* version of the standard fluid velocity equations by evolving a new set of variables (e.g., evolving vorticity in a vortex methods [Brochu et al. 2012; Pfaff et al. 2012; Weißmann and Pinkall 2010]), which can be transformed back to the flow velocity at the end of each timestep. Among these three, the third category is of particular interest to computer graphics researchers because of their potential of a transformed variable in better representing the visually important flow features during the system's evolution. Such methods were understood as a type of "gauge methods" in the computational physics

community (see [Saye 2016, 2017a,b] for example), where the researchers have established a broad array of numerical paradigms to solve the different versions of the gauge-transformed Navier-Stokes equations to leverage their particular strengths in capturing and preserving different types of flow features.

Clebsch maps are one of the mathematical representations exhibiting outstanding expressive power in representing the vortical flow, which was introduced to the computer graphics community by Chern et al. [2016] in their pioneering work of adopting a Schrödinger solver for visual fluid simulation. The definition of Clebsch maps dates back to 1859 [Clebsch 1859], which encodes both velocity and vorticity into a bunch of evolving scalar fields. The spherical Clebsch mapping exhibits its unique strengths in conserving coherent vortical structures over the course of incompressible flow evolution because the wave function field fully encodes the "helical" vortex surfaces (e.g., tubes or sheets) during the flow evolution. We refer readers to Section 3.1, Appendix A, and [Chern et al. 2017, 2016] for a detailed mathematical description of the Clebsch wave function.

This paper devises a gauge fluid solver centered around Clebsch wave functions to solve incompressible Euler equations. Our work is motivated by both Chern et al. [2016] and the recent advances made by Robert Saye [2016] on developing interfacial gauge methods. We demonstrate in our method that Clebsch wave functions can be used as a new type of gauge variables to preserve temporally coherent vortical flow structures. By introducing two groups of gauge variables into a grid-based fluid solver, our method extends the application scope of Clebsch wave functions from the numerical Schrödinger equations to a broader horizon of general fluid simulation problems, including examples of smoke, water, and surface-tension flow in particular. Our new gauge-based PDE solver combines computational merits of both the vorticity expressiveness of the Clebsch wave functions and the versatility of the gauge fluid framework. On the implementation side, our method can be programmed by directly modifying a standard grid-based fluid simulator [Fedkiw et al. 2001] that has been well understood and extensively practiced in the computer graphics community.

To summarize, we list our technical contributions as follows:

- We introduced the first Clebsch gauge method to solve the incompressible Euler equations.
- We devised the first Clebsch wave function representation to unify the gaseous and free-surface fluid simulations, with additional support to surface-tension flow.
- We established a new type of gauge framework to capture and evolve coherent vortical structures in incompressible flow.

2 RELATED WORKS

Vorticity-Preserving Fluid Simulation. Stam [1999] introduced an unconditionally stable fluid simulation solver to computer graphics based on semi-Lagrangian advection. Following this pioneering work, different approaches have been developed to alleviate the dissipation and recover the damped flow details. The first category is to design the numerical schemes for advection, such as MacCormack [Selle et al. 2008], BFECC [Kim et al. 2005], PIC/FLIP [Fu et al. 2017; Jiang et al. 2015; Zhu and Bridson 2005], BiMocq [Qu et al. 2019], and energy-conserving [Mullen et al. 2009], to name just a few. In particular, Zhu and Bridson [2005] adapt Fluid-Implicit-Particle (FLIP) method [Brackbill and Ruppel 1986] by interpolating the change of the flow from the grid to reduce diffusion, which opens the door for a series of hybrid particle-grid schemes to conserve flow details (see [Boyd and Bridson 2012; Ding et al. 2020; Fu et al. 2017; Gagniere et al. 2020; Jiang et al. 2015] for example). Another category of methods is to modify the fluid equations, e.g., by adding additional force terms to preserve vorticities. In [Fedkiw et al. 2001], a vorticity-confinement force is derived from the local flow field and added as an artificial term to preserve the vorticity. Foster and Fedkiw [2001] extend this method to fluid simulation by using a level-set to simulate free-surface flow. Kim et al. [2008] use the wavelet decomposition to find missing high-frequency components and synthesize them back to the velocity field. Bridson et al. [2007] generate turbulent velocity fields based on Perlin noise. Recently, Chern et al. [2016] opened another direction for vortical flow simulation by solving Schrödinger's equation exhibiting visually appealing vortical details.

Vortex Methods. The essential idea of gauge methods is to introduce a set of intermediate variables to reformulate the fluid equations. Vortex methods devise the vorticity as the gauge variable and rewrite the fluid equations into their vorticity-velocity form [Brochu et al. 2012; Pfaff et al. 2012; Weißmann and Pinkall 2010]. By advecting vorticity directly, these solvers naturally preserve circulation within the fluid (e.g., see [Cottet et al. 2000]). Vorticity methods usually rely on Lagrangian elements to evolve the flow features, for instance, particles [Cottet et al. 2000; Leonard 1980; Park and Kim 2005], filaments [Angelidis and Neyret 2005; Weißmann and Pinkall 2010], and sheets [Brochu et al. 2012; Stock et al. 2008]. Vorticity can also be used as a primary variable to improve the mesh-based Eulerian simulation [Elcott et al. 2007], which inspires Zhang et al. [2015] to use vorticity directly in existing grid-based solvers. The drawbacks of vorticity modeling lie in the difficulties of geometric managements and boundary treatments of certain types. Many recent approaches are of a hybrid fashion, either by using grids to improve pure Lagrangian methods [Koumoutsakos et al. 2008], or by integrating Lagrangian elements into existing grid-based solvers [Kim et al. 2009; Pfaff et al. 2012; Selle et al. 2005].

Gauge Methods. The general gauge method was introduced back in [Oseledets 1989] by rewriting the Navier-Stokes equations into its Hamiltonian formula. Different choices of the gauge variables lead to different gauge forms of the Navier-Stokes equations, for instance, impulse method in [Summers and Chorin 1996], velicity method in [Buttke 1993], magnetization variable in [Buttke and Chorin 1993], impetus term in [Maddocks and Pego 1995] and continuous projection form in [Liu et al. 2004]. In our work, we define a new type of gauge, the Clebsch gauge, as the primary variable in our grid-based solver. The flexibility of choosing the gauge variable also allows us to treat complicated boundary conditions as well as to enhance the solver's stability and accuracy. Summers [2000] use the impulse formulation to represent boundary viscous flow at no-slip walls. E and Liu [1997] design finite difference schemes in the velocity-impulse formulation with better stability. Donev et al. [2014] introduce a gauge formulation that casts the evolution of a

constrained system as a nonlocal unconstrained system. The most recent progress in this community includes Robert Saye's works [Saye 2016, 2017a,b] which develop a series of gauge methods for multiphase fluid flow problems with large interfacial discontinuity. Based on the discontinuous Galerkin framework, different choices of the gauge with appropriate boundary handling have been proposed for various types of fluid problems. In this work, we combine some of the boundary treatment techniques from Saye's work with our Clebsch gauge formulation.

Clebsch Maps. Clebsch [1859] introduces a vector representation, established as a Lagrangian and Hamiltonian description of fluid in the Eulerian reference frame (also see [Lamb 1932]). Clebsch potentials contain important geometric information of the flow fields, such as closed integral curves of the associated vorticity field are level lines of the vorticity Clebsch potentials [He and Yang 2016; Xiong and Yang 2020], providing an appealing perspective for fluid visualization [Kotiuga 1991; Kuz'min 1983], analysis [Jeong and Hussain 1995] and simulation [Brandenburg 2010; Cartes et al. 2007]. However, original Clebsch maps cannot represent knotted fields with non-vanishing helicity, and they may not exist near points with vanishing vorticity [Graham and Henyey 2000]. Some scholars use multi-component Clebsch variables to describe flow fields with non-zero helicity [Cartes et al. 2007; Graham and Henyey 2000; Zakharov and Kuznetsov 1997]. However, this yields no accessible representations of the vortex lines and surfaces. Chern et al. [2017, 2016] propose spherical Clebsch maps, which could not only represent the velocity-vorticity field with non-trivial helicity but also contain important geometric information of the vorticity field. Given the vorticity spherical Clebsch potentials are exactly the vortex surface fields, they can be used as the initial conditions for the evolution of vortex surfaces in the Lagrangian-like study of vortex dynamics [Xiong and Yang 2017, 2019; Yang and Pullin 2011; Zhao et al. 2018, 2016]. In addition, a series of fluid representation methods can be written in the spherical Clebsch forms such as rational maps [Kedia et al. 2016] and exponential maps [Smiet et al. 2017, 2015]. Our gauge transformation is based on the spherical Clebsch maps.

3 CLEBSCH GAUGE METHOD

In this section, we devise the gauge transformation of the incompressible Euler equations using the Clebsch wave functions defined in the previous section. Without loss of generality, we consider solving an incompressible flow problem in region $\boldsymbol{x} \in \Omega$ with the free surface $\partial\Omega_f$ and the solid wall $\partial\Omega_b$. Our physical model incorporates most of the essential ingredients for an incompressible solver used in computer graphics, including solid boundaries, gravity, interface, and surface tension. Each ingredient can be omitted easily according to a different simulation setting.

3.1 Spherical Clebsch wave functions

Wave function. Mathematically, a spherical Clebsch mapping represents a velocity field $\boldsymbol{u}$ using a normalized wave function composed of two complex numbers:

$$\boldsymbol{\phi} = (\phi_1, \phi_2)^T = (a + bi, c + di)^T, \text{ with } \langle \boldsymbol{\phi}, \boldsymbol{\phi} \rangle_{\mathbb{R}} = 1. \tag{1}$$

Table 1. Notations Table

Notation	Definition
$\boldsymbol{\phi} \in \mathbb{C}^2/\mathbb{R}^4$	Clebsch wave function $\boldsymbol{\phi} = (\phi_1, \phi_2)^T$
$\boldsymbol{\psi} \in \mathbb{C}^2/\mathbb{R}^4$	Gauge wave function $\boldsymbol{\psi} = \boldsymbol{\phi} e^{i\varphi/\hbar}$
$\overline{\boldsymbol{\psi}}$	Conjugate of $\boldsymbol{\psi}$
$\Delta\boldsymbol{\psi}$	Laplacian of $\boldsymbol{\psi}$
$\langle \boldsymbol{\phi}, \boldsymbol{\psi} \rangle_{\mathbb{C}}$	$\overline{\phi}_1\psi_1 + \overline{\phi}_2\psi_2$
$\langle \boldsymbol{\phi}, \boldsymbol{\psi} \rangle_{\mathbb{R}}$	$\mathrm{Re}(\overline{\phi}_1\psi_1 + \overline{\phi}_2\psi_2)$
$\boldsymbol{u} \in \mathbb{R}^d$	Physical velocity
$\boldsymbol{u}_m \in \mathbb{R}^d$	$\boldsymbol{\psi}$-mapped velocity: $\boldsymbol{u}_m = \hbar \langle \nabla\boldsymbol{\psi}, i\boldsymbol{\psi} \rangle_{\mathbb{R}}$
$\boldsymbol{u}_m^* \in \mathbb{R}^d$	Intermediate velocity after blending
$\boldsymbol{u}_q \in \mathbb{R}^d$	Intermediate velocity after surface tension
$q \in \mathbb{R}$	Auxiliary variable for free-surface b.c.
$\varphi \in \mathbb{R}$	Auxiliary variable for incompressibility
$\hbar \in \mathbb{R}$	The parameter for vorticity strength
$\alpha \in \mathbb{R}$	The parameter for $\boldsymbol{\psi}$ partial projection
$\beta \in \mathbb{R}$	The parameter for velocity blend

The mapping from a wave function field $\boldsymbol{\phi}$ to a velocity field $\boldsymbol{u}$ can be expressed using the following mapping operator [Chern et al. 2016]:

$$\boldsymbol{u} = \hbar \langle \nabla\boldsymbol{\phi}, i\boldsymbol{\phi} \rangle_{\mathbb{R}}, \tag{2}$$

with $\langle \boldsymbol{\phi}, \boldsymbol{\psi} \rangle_{\mathbb{R}} = \mathrm{Re}(\overline{\phi}_1\psi_1 + \overline{\phi}_2\psi_2)$ and the constant $\hbar$ as a tunable parameter specifying the quantization of vorticity.

Gauge transformation. For a wave function $\boldsymbol{\phi}$, the gauge transformation is defined as:

$$\boldsymbol{\psi} = \boldsymbol{\phi} e^{i\varphi/\hbar}, \tag{3}$$

where φ is a scalar gauge variable. We showed in Appendix A that this transformation satisfies gauge invariance because both $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ corresponds to the same vorticity field. In addition, the gauge transformation of velocity $\boldsymbol{u} \to \boldsymbol{u} + \nabla\varphi$ amounts to the gauge transformation of the wave function $\boldsymbol{\phi} \to \boldsymbol{\phi} e^{i\varphi/\hbar}$. The incompressibility condition enforced on velocity $\nabla \cdot \boldsymbol{u} = 0$ amounts to $\langle i\boldsymbol{\phi}, \Delta\boldsymbol{\phi} \rangle_{\mathbb{R}} = 0$ enforced on wave functions.

3.2 Euler equations

We write the incompressible Euler equations with gravity, surface tension, and boundary conditions as:

$$\begin{cases} \dfrac{D\boldsymbol{u}}{Dt} = -\nabla\left(\dfrac{p}{\rho} - G\right), \ \boldsymbol{x} \in \Omega, \\ \nabla \cdot \boldsymbol{u} = 0, \ \boldsymbol{x} \in \Omega, \\ \boldsymbol{u} \cdot \boldsymbol{n} = \boldsymbol{u}_\partial \cdot \boldsymbol{n}, \ \boldsymbol{x} \in \partial\Omega_b, \\ p = \gamma\kappa, \ \boldsymbol{x} \in \partial\Omega_f. \end{cases} \tag{4}$$

Here, $D/Dt = \partial/\partial t + \boldsymbol{u} \cdot \nabla$ is the material derivative, $\boldsymbol{u}$ is velocity, p is pressure, ρ is density, $\boldsymbol{u}_\partial$ is the velocity of the solid boundary $\partial\Omega_b$, γ is surface tension, and κ is the interface curvature. The surface tension on the interface is modeled as the pressure jump due to the local curvature. We use $G = G(\boldsymbol{x}) = \boldsymbol{g} \cdot \boldsymbol{x}$ as the gravitational potential. The fluid viscosity is introduced by the numerical scheme

rather than by (4). The four equations in (4) are the momentum conservation equation, the divergence-free condition, the Neumann boundary condition on the solid boundary, and the Dirichlet boundary condition with a pressure jump on the free surface.

3.3 Euler equations with Clebsch wave functions

Next, we show how to rewrite Equation (4) using Clebsch wave functions. Substituting the wave-velocity mapping $\boldsymbol{u} = \hbar \langle \nabla \boldsymbol{\phi}, i\boldsymbol{\phi} \rangle_{\mathbb{R}}$ defined in Equation (2) into Equation (4) yields:

$$\begin{cases} \dfrac{D\boldsymbol{\phi}}{Dt} = -i\dfrac{1}{\hbar}\left(\dfrac{p}{\rho} - G - \dfrac{|\boldsymbol{u}|^2}{2}\right)\boldsymbol{\phi}, \ \boldsymbol{x} \in \Omega, \\ \boldsymbol{u} = \hbar \langle \nabla \boldsymbol{\phi}, i\boldsymbol{\phi} \rangle_{\mathbb{R}}, \ \boldsymbol{x} \in \Omega, \\ \langle i\boldsymbol{\phi}, \Delta \boldsymbol{\phi} \rangle_{\mathbb{R}} = 0, \ \boldsymbol{x} \in \Omega, \\ \hbar \langle \nabla \boldsymbol{\phi}, i\boldsymbol{\phi} \rangle_{\mathbb{R}} \cdot \boldsymbol{n} = \boldsymbol{u}_{\partial} \cdot \boldsymbol{n}, \ \boldsymbol{x} \in \partial\Omega_b, \\ p = \gamma\kappa, \ \boldsymbol{x} \in \partial\Omega_f. \end{cases} \tag{5}$$

The first row in Equation (5) corresponds to a transformed momentum conservation law with wave functions. The second line is the wave-velocity mapping. The third line is the transformed divergence-free condition. The fourth line is the transformed Neumann boundary condition on the solid boundary. And the last line is the same Dirichlet boundary condition as in Equation (4). The first equation in Equation (5) corresponds to Equation (9.13) (with $\epsilon = 0$) in Chern [2017] but it additionally takes into account the interfacial forces. We refer the readers to Section 9.3 in Chern [2017] for a detailed derivation of the first equation in Equation (5).

3.4 Gauge transformation of Equation (5)

We use the gauge transformation of $\boldsymbol{\phi}$ defined in Section 3.1 to derive the gauge form of Equation (5). We introduce a gauge variable φ as an auxiliary scalar field. By setting $\boldsymbol{\psi} = \boldsymbol{\phi}\exp(i\varphi/\hbar)$ and at the same time enforcing φ's Neumann boundary as $\partial_n\varphi = 0$ for $\boldsymbol{x} \in \partial\Omega_b$ and φ's Dirichlet boundary as $\varphi = 0$ for $\boldsymbol{x} \in \partial\Omega_f$ (in order to not change $\boldsymbol{\phi}$'s boundary conditions over the course of the transformation), we can use the gauge wave function $\boldsymbol{\psi}$ to rewrite Equation (5) as:

$$\begin{cases} \dfrac{D\boldsymbol{\psi}}{Dt} = -\dfrac{i}{\hbar}\left[\dfrac{p}{\rho} - \dfrac{|\boldsymbol{u}|^2}{2} - G\right]\boldsymbol{\psi}, \ \boldsymbol{x} \in \Omega, \\ \boldsymbol{u}_m = \hbar \langle \nabla \boldsymbol{\psi}, i\boldsymbol{\psi} \rangle_{\mathbb{R}}, \ \boldsymbol{x} \in \Omega, \\ \Delta\varphi = \nabla \cdot \boldsymbol{u}_m, \ \boldsymbol{x} \in \Omega, \\ \boldsymbol{u} = \boldsymbol{u}_m - \nabla\varphi, \ \boldsymbol{x} \in \Omega, \\ \hbar \langle \nabla \boldsymbol{\psi}, i\boldsymbol{\psi} \rangle_{\mathbb{R}} \cdot \boldsymbol{n} = \boldsymbol{n} \cdot \boldsymbol{u}_{\partial}, \ \boldsymbol{x} \in \partial\Omega_b, \\ \partial_n\varphi = 0, \ \boldsymbol{x} \in \partial\Omega_b, \\ p = \gamma\kappa, \ \boldsymbol{x} \in \partial\Omega_f. \end{cases} \tag{6}$$

The first line in Equation (6) is the gauge transformed momentum conservation law. Lines 2-4 are the gauge transformed incompressibility. We introduce another auxiliary variable $\boldsymbol{u}_m$ as the $\boldsymbol{\psi}$-mapped physical counterpart velocity by employing the wave-velocity mapping defined in Equation (2) on the gauge variable $\boldsymbol{\psi}$. Equations in lines 2-4 present a projection algorithm to project the gauge wave function to its physical counterpart from $\boldsymbol{\psi} \to \boldsymbol{u}_m \to \boldsymbol{u}$. Specifically, the equation in line 3 amounts to the standard Poisson equation we have seen in a conventional projection step. Equations in lines 5-7 present the boundary conditions under the gauge wave function setting similar to the ones defined in Equation (5).

Here we want to put a quick note on the role of $\boldsymbol{u}_m$: it behaves like a gauge variable in a traditional sense (without introducing wave functions) and acts as the physical avatar of the gauge wave function $\boldsymbol{\psi}$. Whenever we need to calculate the interaction between $\boldsymbol{\psi}$ and other quantities in a physical space, we can first convert $\boldsymbol{\psi}$ to $\boldsymbol{u}_m$ and exert the standard operations on $\boldsymbol{u}_m$. In our code implementation, $\boldsymbol{u}_m$ is implemented as a standard vector field in $\mathbb{R}^d$ that can be used in standard operations such as the projection step.

3.5 Gauge form for free surface

To handle the normal stress balance on Ω_f, we introduce a second auxiliary variable q (besides φ) to handle the free-surface boundary conditions. By putting together the terms within the parentheses of the first lines in Equation (5) and Equation (6), we define

$$q = \frac{p}{\rho} - \frac{|\boldsymbol{u}|^2}{2} - G. \tag{7}$$

Thanks to the gauge invariance of q, its Laplacian can be arbitrary as the way q affecting the physical field is via its gradient. Motivated by Saye [2016], we solve a harmonic equation for q to obtain one of its smooth distributions satisfying the given boundary conditions:

$$\begin{cases} \Delta q = 0, \ \boldsymbol{x} \in \Omega, \\ \partial_n q = 0, \ \boldsymbol{x} \in \partial\Omega_b, \\ q - \dfrac{1}{\rho}\gamma\kappa + \left(\dfrac{1}{2}|\boldsymbol{u}|^2 + G\right) = 0, \ \boldsymbol{x} \in \partial\Omega_f. \end{cases} \tag{8}$$

It is worth noting that Equation (8) is a typical Poisson system with Neumann and Dirichlet boundary conditions. Both the surface tension and the body force affect the system via the free-surface boundary conditions (line 3 of Equation (8)), which enforce a strict satisfaction of the free-surface boundary conditions. Solving a harmonic field was a recent numerical technique introduced by Robert Saye in his interfacial gauge method [Saye 2016] (see Equation 7 in his paper) to incorporate interfacial forces into a gauge framework. We leverage this numerical scheme in our interfacial problem under a wave function setting and show that it can facilitate solving the interfacial flow problems with the wave function gauge.

3.6 Clebsch gauge formula

We obtain our final version of the Clebsch gauge formula for incompressible Euler equation by substituting q into Equation (6):

$$\begin{cases} \dfrac{D\boldsymbol{\psi}}{Dt} = -\dfrac{i}{\hbar}q\boldsymbol{\psi}, \\ \boldsymbol{u}_m = \hbar \langle \nabla \boldsymbol{\psi}, i\boldsymbol{\psi} \rangle_{\mathbb{R}}, \\ \Delta q = 0, \\ \Delta\varphi = \nabla \cdot \boldsymbol{u}_m, \\ \boldsymbol{u} = \boldsymbol{u}_m - \nabla\varphi, \end{cases} \quad \boldsymbol{x} \in \Omega, \tag{9}$$

with the solid boundary conditions:

$$\begin{cases} \hbar \langle \nabla \boldsymbol{\psi}, i\boldsymbol{\psi} \rangle_{\mathbb{R}} \cdot \boldsymbol{n} = \boldsymbol{n} \cdot \boldsymbol{u}_{\partial}, \\ \partial_n q = 0, \\ \partial_n\varphi = 0, \end{cases} \quad \boldsymbol{x} \in \partial\Omega_b, \tag{10}$$

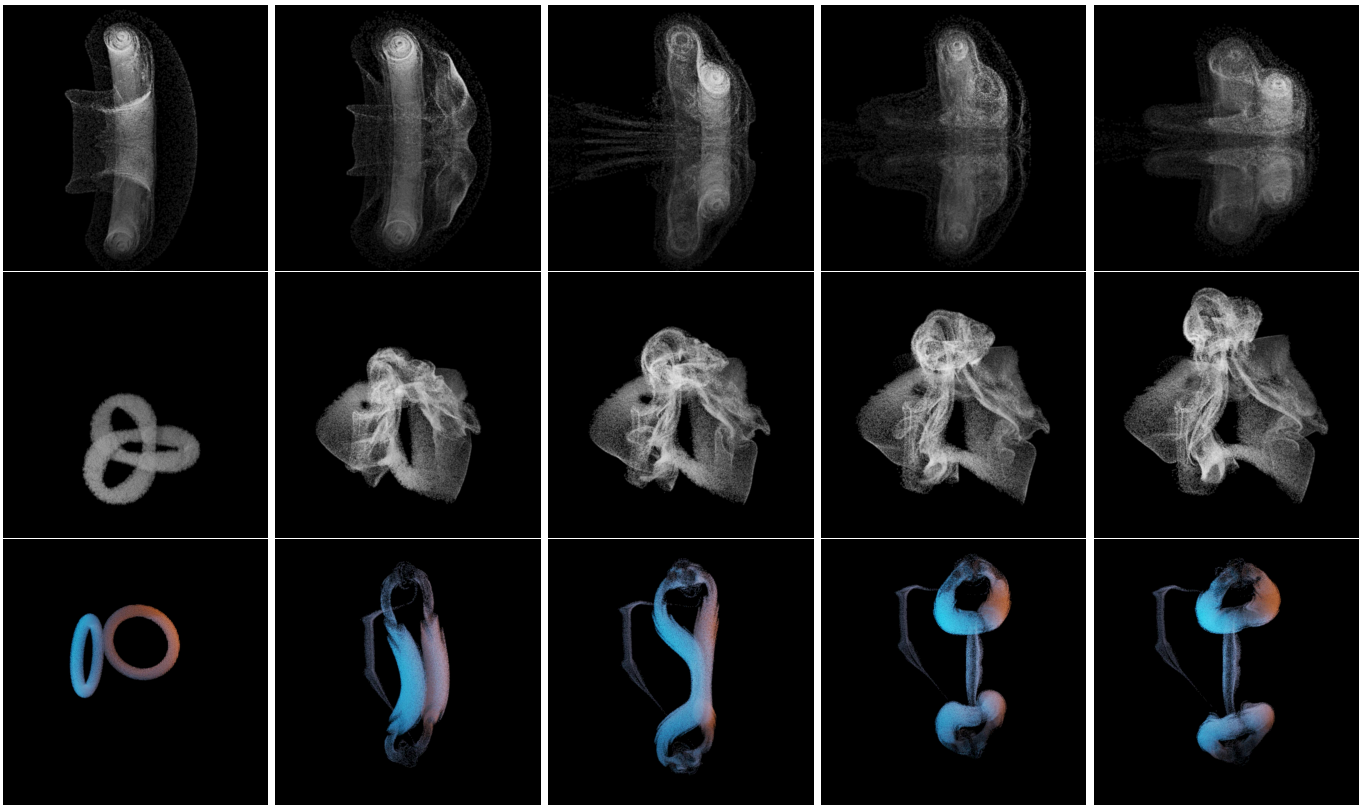


Fig. 2. Top row: Leapfrogging example at frames 100, 120, 200, 300, and 400.; Middle row: Trefoil knot example at frames 1, 200, 250, 300, 350; Bottom row: Oblique ring collision example at frame 1, 200, 260, 300, and 320. All three examples use the grid size of $128 \times 128 \times 128$.

and the free surface boundary conditions:

$$\begin{cases} \varphi = 0, \\ q - \dfrac{1}{\rho}\gamma\kappa + \left(\dfrac{1}{2}|\boldsymbol{u}|^2 + G\right) = 0 \end{cases} \boldsymbol{x} \in \partial\Omega_f. \tag{11}$$

Equations (9), (10), and (11) are the equations that are ready to be discretized in our numerical method. The five equations in Equation (9) can be translated directly into the four steps in the numerical algorithm as gauge wave function advection (line 1), wave-velocity mapping (line 2), solving q (line 3), and divergence-free projection (lines 4-5). The boundary conditions are specified as the Neumann boundary on the solid boundary and the Dirichlet boundary (with jump conditions) on the free surface. If we establish an analog between the auxiliary variable φ and the physical pressure p, these boundary conditions can be understood easily from a standard incompressible Euler solver's perspective. The only exception is the boundary condition on the gauge wave function specified in line 1 of Equation (10).

4 NUMERICAL ALGORITHM

We describe our numerical algorithm to solve the PDEs specified in Equation (9) with the boundary conditions specified in Equation (10) and (11). We take a MAC grid structure to discretize the computational domain $\Omega \in R^d$. The velocity $\boldsymbol{u}$ is a vector field, and therefore we store it on the MAC grid faces. The wave function $\boldsymbol{\psi}$ is a four-component vector field regardless of the problem's dimension, which we store on cell centers. The intermediate variables q and φ are both scalar fields, and we store both of them on cell centers as well. We solve the PDEs on the entire domain for the single-phase incompressible flow (for smoke simulations). We use a level-set function to track the evolution of fluids with free surfaces, following the same implementations in [Kang et al. 2000]. We summarize our time integration scheme in Section 5.

4.1 Advection

We advect the wave function $\boldsymbol{\psi}$ in the advection step. Mathematically, this amounts to solving $\mathrm{D}\boldsymbol{\psi}/\mathrm{D}t = 0$. We conduct the standard semi-Lagrangian advection step for each component of $\boldsymbol{\psi}$ independently.

Boundary conditions for $\boldsymbol{\psi}$. When boundaries exist in a simulation example, the boundary conditions of $\boldsymbol{\psi}$ need to be enforced. We discretize the $\boldsymbol{\psi}$ boundary conditions as fluid *sources*, *solids*, and *free surfaces*.

- For *sources*, we follow the method introduced by [Chern et al. 2016] to enforce the value of the wave functions in the source area by assuming a constant velocity within the source region.
- For *free surfaces*, we conduct an extrapolation step for both $\boldsymbol{u}$ and $\boldsymbol{\psi}$ near the boundary. The extrapolated velocity value at position $\boldsymbol{x}_{nb}$ can be simply calculated using interpolation,

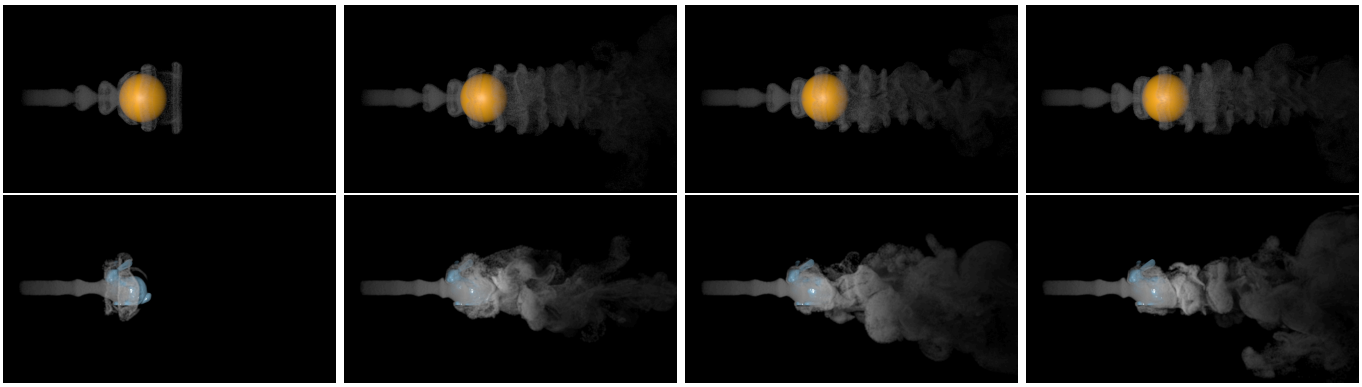


Fig. 3. Results of smoke passing over an obstacle. Top row: A sphere obstacle at frames 32, 128, 256, and 384; Bottom row: A bunny obstacle at frames 16, 64, 128, and 192. Both examples use the grid size of $192 \times 192 \times 384$.

while the extrapolated wave function $\boldsymbol{\psi}_{nb}$ can be calculated using the wave function $\boldsymbol{\psi}_i$ and velocity $\boldsymbol{u}_i$ at its nearest points $\boldsymbol{x}_i$ on the surface as $\boldsymbol{\psi}_{nb} = \boldsymbol{\psi}_i e^{i\boldsymbol{u}_i/\hbar(\boldsymbol{x}_{nb}-\boldsymbol{x}_i)}$.

- For *solid* boundaries, we define no $\boldsymbol{\psi}$ inside the solid area. When the back-traced position is inside a solid area, we can use the same extrapolation method to get the wave function value at that position. Specifically, for a static obstacle, $\boldsymbol{\psi}$ at the intersection point on the solid boundary is used as the back-traced $\boldsymbol{\psi}$ in the obstacle.

Normalizing $\boldsymbol{\psi}$. After the semi-Lagrangian advection step, we normalize the wave function in each grid cell as: $\boldsymbol{\psi} \rightarrow \boldsymbol{\psi}/|\boldsymbol{\psi}|$.

4.2 $\psi \rightarrow \mathbf{u}_m^*$

We convert the advected and normalized wave function to its velocity counterpart by calculating $\boldsymbol{u}_m = \hbar \langle \nabla \boldsymbol{\psi}, i\boldsymbol{\psi} \rangle_{\mathbb{R}}$ on each grid face. On a MAC grid, the velocity on a face $\boldsymbol{u}_m$ is updated using the two wave functions stored on its incident cells (ψ_v, ψ_w) as

$$\boldsymbol{u}_m = \hbar \arg \langle \boldsymbol{\psi}_v, \boldsymbol{\psi}_w \rangle_{\mathbb{C}} . \tag{12}$$

As proven in Appendix D in [Chern et al. 2016], the 1-form velocity calculated by Equation (12) yields analytical precision without accumulating any numerical errors. Equation (12) can also be solved using numerical schemes such as finite-difference on a Cartesian grid. We experimented with both analytical and numerical schemes and observed similar results for the produced $\boldsymbol{u}_m$.

Blending the advected velocity. One problem we noticed for generating $\boldsymbol{u}_m$ from $\boldsymbol{\psi}$ directly is its advection tends to be stuck behind the solids. To enhance the flow convection near solid boundaries, in each time step, we blend the advected velocity with the $\boldsymbol{\psi}-$mapped velocity to improve the advection accuracy. We advect the velocity $\boldsymbol{u}$ at timestep t to obtain an intermediate velocity $\boldsymbol{u}^*$ and further blend it with the $\psi-$mapped velocity $\boldsymbol{u}_m$ by β as: $\boldsymbol{u}_m^* = \beta \boldsymbol{u}_m + (1-\beta)\boldsymbol{u}^*$ with β as a tunable parameter.

4.3 $\psi \rightarrow \psi_q$ and $\mathbf{u}_m^* \rightarrow \mathbf{u}_q$ *(For free-surface flow)*

For the fluid with free surfaces $\partial\Omega_f$, we solve the Laplace equation of the auxiliary variable q:

$$\Delta q = 0 \, , \boldsymbol{x} \in \Omega. \tag{13}$$

We enforce Neumann boundary conditions on the solid boundary and Dirichlet boundary conditions with a jump condition specified by the net effects of the body forces and surface tensions on the free surfaces:

$$\begin{cases} \partial_n q = 0, \ \boldsymbol{x} \in \partial\Omega_b, \\ q - \dfrac{1}{\rho}\gamma\kappa + \left(\dfrac{1}{2}|\boldsymbol{u}|^2 + G\right) = 0, \ \boldsymbol{x} \in \partial\Omega_f. \end{cases} \tag{14}$$

We follow the numerical techniques proposed in [Kang et al. 2000] to solve the Poisson equation with variable coefficients on an irregular domain and jump conditions on the domain boundary. In particular, the curvature κ is calculated as the divergence of the level-set normal on the implicit boundary.

Then we use the calculated $\boldsymbol{q}$ to correct the wave function $\boldsymbol{\psi}$ and its corresponding gauge velocity $\boldsymbol{u}_m^*$ as:

$$\boldsymbol{\psi}_q = \boldsymbol{\psi} e^{-iq\Delta t/\hbar}. \tag{15}$$

The corrected $\boldsymbol{\psi}_q$ is used to calculate a corrected $\boldsymbol{u}_q$. Instead of calculating $\boldsymbol{u}_q$ from $\boldsymbol{\psi}_q$ again, we apply q on $\boldsymbol{u}_m^*$ as $\boldsymbol{u}_q = \boldsymbol{u}_m^* - \nabla q \Delta t$, which gives out the same gauge velocity that has the surface forces handled.

4.4 Divergence-Free Projection

To enforce the divergence-free conditions on velocity $\boldsymbol{u}$, we solve the Poisson equation of φ:

$$\Delta\varphi = \nabla \cdot \boldsymbol{u}_q, \ \boldsymbol{x} \in \Omega, \tag{16}$$

with Neumann boundary conditions on the solid boundary and Dirichlet boundary conditions on the free surface:

$$\begin{cases} \partial_n \varphi = 0, \ \boldsymbol{x} \in \partial\Omega_b, \\ \varphi = 0, \ \boldsymbol{x} \in \partial\Omega_f. \end{cases} \tag{17}$$

Then we use the solved values for φ to project $\boldsymbol{u}_q$ to obtain the physical divergence-free velocity field $\boldsymbol{u}$ for the next time step:

$$\boldsymbol{u} = \boldsymbol{u}_q - \nabla\varphi. \tag{18}$$

Although the divergence-free parts of the velocity mapped from $\boldsymbol{\psi}$ and $\boldsymbol{\psi} e^{(-i\varphi/\hbar)}$ are the same, to prevent the $\boldsymbol{\psi}$-mapped velocity from drastically deviating from the projected incompressible flow

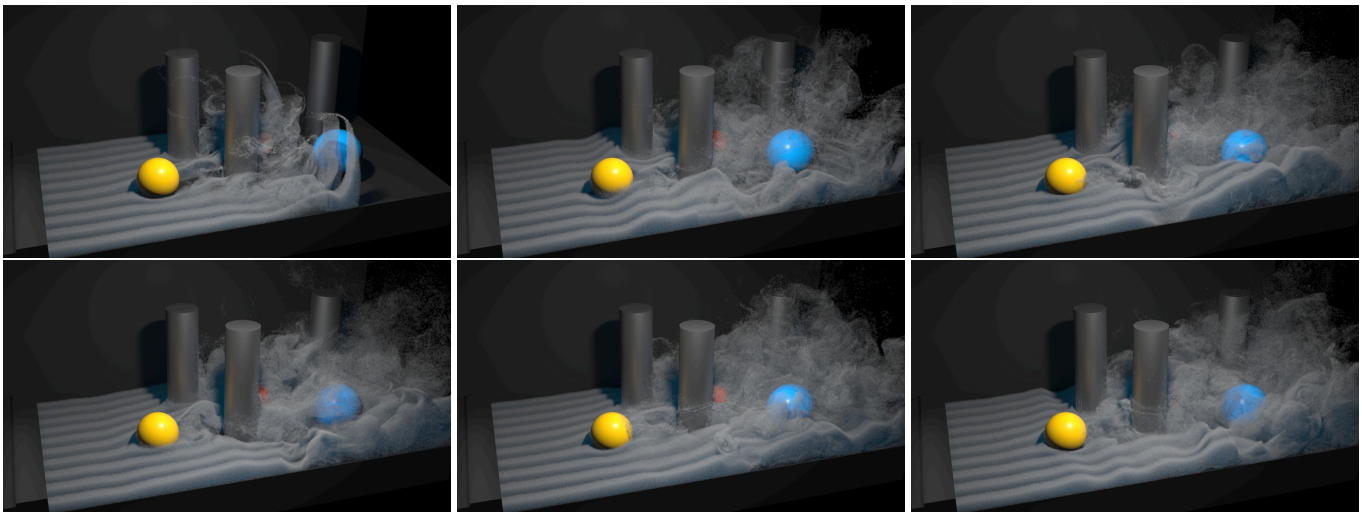


Fig. 4. Results of smoke passing over multiple obstacles at frames 30, 50, 90, 195, 215, and 230. The grid size is $192 \times 192 \times 384$. Arrays of vortices are created behind the obstacles and interact with each other. Coherent vortex structures are preserved and evolved over the course of the entire simulation.

field, we employ an additional projection step on φ to update the gauge variable $\boldsymbol{\psi}$:

$$\boldsymbol{\psi} = \boldsymbol{\psi}_q e^{(-i\varphi/\hbar)\alpha}, \tag{19}$$

with a tunable parameter α ($0 < \alpha \leqslant 1$) to control the portion being projected on $\boldsymbol{\psi}$. We name the step as *soft projection* if φ is only partially applied. We note that the parameters α and β do not change the equations we are solving. We conduct parameter studies in Section 6.4 and summarize their impacts in Section 7.

5 TIME INTEGRATION

Our time integration scheme can be summarized as follows:

(1) *Advect $\boldsymbol{\psi}$:* Advect $\boldsymbol{\psi}$ using $\boldsymbol{u}$.
(2) *Normalize $\boldsymbol{\psi}$:* Normalize $\boldsymbol{\psi}$ after advection.
(3) *Map $\boldsymbol{\psi}$ to $\boldsymbol{u}_m$:* Calculate $\boldsymbol{u}_m$ from $\boldsymbol{\psi}$ using Equation (12).
(4) *Blend $\boldsymbol{u}_m$:* Advect $\boldsymbol{u}$ to obtain $\boldsymbol{u}^*$. Blend $\boldsymbol{u}^*$ into $\boldsymbol{u}_m$ to get $\boldsymbol{u}_m^*$.
(5) *Solve q (free-surface):* If there is a free boundary, solve Equation (13) and (14) to get q and use it to update $\boldsymbol{\psi}_q$ and $\boldsymbol{u}_q$.
(6) *Project φ:* Solve Equation (16) and (17) to get φ. Apply the soft projection step on $\boldsymbol{\psi}_q$, and the projection step on $\boldsymbol{u}_q$ to obtain $\boldsymbol{\psi}$ and $\boldsymbol{u}$ for the next time step.

6 RESULTS

We evaluate the efficacy of our method by a set of fluid simulation examples, including the evolution of vortex tubes, turbulent smoke, and free-surface flows with surface tensions. Detailed settings can be found in Table 2. For smoke simulations, we advect passive particles in the domain for rendering purposes. For liquid simulations, we render the surface extracted from the level-set function. For the timestep, we followed the conventional CFL constraints on velocity. We refer the readers to our supplementary video for all the animations.

6.1 Evolution of Vortex Tubes

We first show our method's ability to keep and evolve coherent vortex structures by testing it in several examples with analytical initialization. To initialize the values for $\boldsymbol{\psi}$, we follow [Chern et al. 2016] to initialize isolated (knotted or unknotted) vortex rings.

Our first example is the leapfrogging vortex rings [Lim 1997]. Two circular vortex rings alternately leapfrog around each other. As shown in Figure 2, the structures of the two vortex rings are still preserved clearly after 5 cycles.

Our second example is the trefoil knot [Kleckner and Irvine 2013]. When the filament crosses itself, our method can correctly reproduce the vortex tube reconnection process that creates two separate vortex rings (one large ring and one small ring that is moving off) and can preserve their structures well in the subsequent evolution, as shown in Figure 2.

Another example of vortex filament dynamics that are difficult to simulate using a standard grid-based solver is the oblique smoke ring collision [Lim 1989]. After the two identical vortex rings approached one another with a colliding angle, the vortex tube reconnection process will start and form two new rings. As shown in Figure 2, our simulation method successfully reproduces this phenomenon.

6.2 Smoke Simulations

We conduct three smoke simulations with source and obstacles to show our method's ability to handle complex sources and solid boundary conditions. As shown in Figure 3, when a high-speed smoke plume passes an obstacle, arrays of vortices are created behind the obstacle, and coherent vortex structures are preserved and evolved over the course of the entire simulation. In the multiple-obstacle example shown in Figure 4, a high-speed source is placed at the bottom of the left wall, and several obstacles are placed in the middle of the scene. Vortices were created behind the objects and can interact with each other.

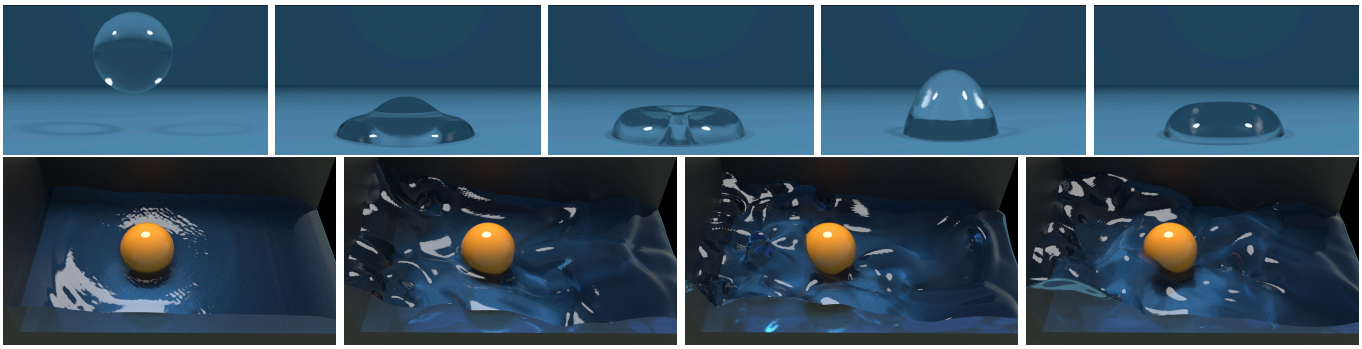
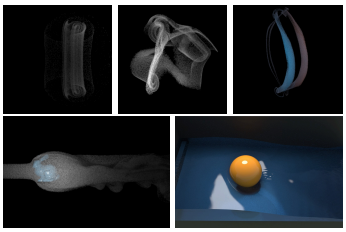


Fig. 5. Top row: A droplet falling to the floor at frames 0, 9, 18, 23, and 35, with a surface tension of 0.2. The grid size for this example is $128 \times 128 \times 42$. Bottom row: A stream flowing over an obstacle at frames 20, 240, 600, and 900, with a surface tension of 0.05. The grid size for this example is $256 \times 128 \times 64$.

6.3 Liquid Simulations

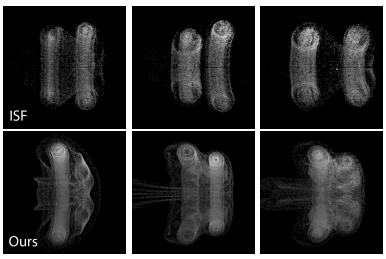
We show three interfacial flow examples where the surface tension force drives the dynamics of the fluids. In Figure 5, we set up a simple example of a surface-tension-driven droplet falling and bouncing on a floor to show our method's ability to handle surface tensions. Next, similar to our smoke examples, we place obstacles and sources for liquid simulation. As shown in Figure 5 and Figure 8, we initialize liquid volume in a water tank with the left, bottom, front and back walls. A source is placed at the bottom of the left wall, and one or multiple obstacles are placed in the middle of the water tank. Our method constantly generates a host of underwater vortices due to the interactions between fluid and obstacles. We solve q to handle surface tension forces in all these examples.

6.4 Validation

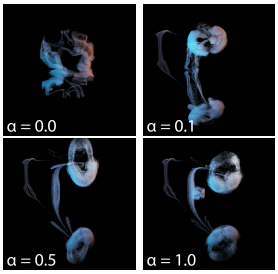


Comparison with the grid-based solver. We compared our Clebsch gauge solver with a standard grid-based fluid solver proposed in [Fedkiw et al. 2001]. For each test case, we ran both methods with the same grid resolution and boundary conditions. The comparisons include leapfrogging (Figure 2), trefoil knot (Figure 2), oblique ring collision (Figure 2), smoke passing a bunny (Figure 3), and a stream passing a sphere (Figure 5). We show the simulations produced by the standard method in the inset figures, and we compiled the comparison animations in the appendix video. As evidenced in these comparisons, a standard grid-based solver cannot preserve coherent vortical structure evolutions such as those captured with our method, be they rhythmic leapfrogging, one rising vortex ring in trefoil knot, pair of vortex rings in oblique ring collision, and an array of interacting vortices behind obstacles or underneath the water surface.

Comparison with Incompressible Schrödinger's Flow. We compared our solver with the Incompressible Schrödinger's Flow (ISF) [Chern et al. 2016]. Since the parameter $\hbar$ has its physical meaning in [Chern et al. 2016], we used a different $\hbar$ value ($\hbar = 0.2$) in this example and keep all the other settings the same. We show the comparion in the inset figures, and we compiled the corresponding animations in the appendix video. Despite the different PDEs being solved, we observed that both approaches could produce flow simulations with clear and coherent vortical structures.



Parameter study of α. To better understand the impact of the parameters α and β, we further conduct two sets of simulations with different parameter values. First, we used four different values of α with the same simulation settings in the ring collision example (see the inset figures). We observed that when $\alpha = 0$, which means no projection is carried out, the vortical structure would not evolve correctly. When α does not equal zero, the simulator can separate the two colliding vortex rings and generate a pair of new rings. Conducting projection (or soft projection) on $\boldsymbol{\psi}$ is necessary to prevent the results from being messy, while the result is not very sensitive to the value of α.



Parameter study of β. Second, in the smoke passing obstacle example with different β values (see the inset figures), we observed that the magnitude of β does not affect the generation of the coherent vortices in the flow field. The main differences show up near the solid boundary: A small β (with the major part of velocity being advected) can enhance the flow convection and vortex shedding near solid boundaries significantly. In contrast, a large β (with the major part of velocity being mapped from ψ) will hinder the vortex shedding around obstacles. If β is too small (e.g., $\beta <= 0.01$), the vortical structures in the flow will be less preserved.

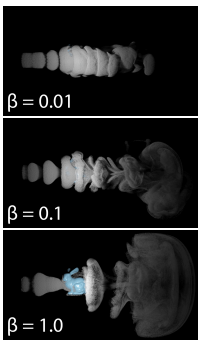


Table 2. Details of the Simulation Examples.

Example Description	α	β	$\hbar$	Domain Size	Resolution	CFL	Frame Rate	Time / Frame (avg.) †
Leapfrogging	.2	1.	.5	$10. \times 10. \times 10.$	$128 \times 128 \times 128$	1.	25	2.9s †3
Trefoil Knot	.05	1.	.1	$5. \times 5. \times 5.$	$128 \times 128 \times 128$	1.	50	2.1s †3
Oblique Ring Collision	.25	1.	.5	$12.5 \times 12.5 \times 12.5$	$128 \times 128 \times 128$	1.	25	1.8s †3
Smoke with Sphere	1.	.02	1.	$20. \times 10. \times 10.$	$384 \times 192 \times 192$	2.	10	33.6s †2
Smoke with Bunny	1.	.02	1.	$20. \times 10. \times 10.$	$384 \times 192 \times 192$	2.	10	22.7s †1
Smoke with Multiple Obstacles	1.	.02	1.	$20. \times 10. \times 10.$	$384 \times 192 \times 192$	2.	10	30.5s †1
Droplet Falling to Ground	1.	.02	.1	$1. \times 1. \times .33$	$128 \times 128 \times 42$	.2	100	2.7s †3
Stream with Sphere	.1	.02	.1	$2. \times 1. \times .5$	$256 \times 128 \times 64$	1.	100	6.8s †1
Stream with Multiple Obstacles	.1	.02	.1	$2. \times 1.5 \times .75$	$256 \times 192 \times 96$	1.	100	23.4s †1

† These experiments were performed on different machines: †1 is a server with a 128-core CPU and a Quadro RTX 8000 GPU; †2 is a desktop with a 16-core CPU and a GTX 1080Ti GPU; †3 is a laptop with a 12-core CPU and an RTX 2070 GPU.

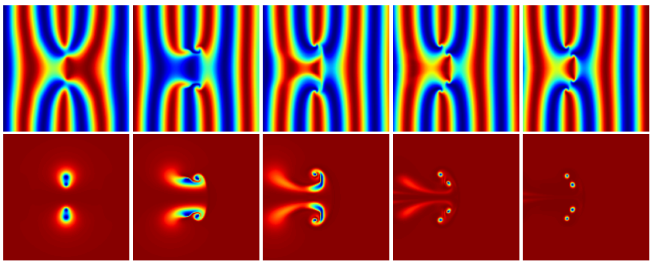


Fig. 6. Top row: The $\mathrm{Re}(\psi_1)$ value of a cross-section of the 3d leapfrogging example at frames 0, 50, 100, 200, and 300; Bottom row: the corresponding s ($= |\psi_1|^2 - |\psi_2|^2$). Values are mapped to colors.

We found a relatively small β is specifically effective for shedding vortices around obstacles, which is very useful for some cases, e.g., when we simulate turbulent smoke interacting with obstacles. We use a large β (e.g., $\beta = 1$) if vortical structures were initialized in the flow domain, such as in those analytical flow examples.

The evolution of ψ. To further investigate the mechanics of our gauge solver, we visualize the spatiotemporal evolution of different intermediate gauge quantities. One of the main motivations to carry out these visualizations is that $\boldsymbol{\psi}$, as a function of the Lagrangian coordinates, may potentially behave badly (e.g., get stirred and mixed up) during the advection. In particular, we visualize the evolution of the real component of the wave function $\mathrm{Re}(\psi_1)$ and the value s ($|\psi_1|^2 - |\psi_2|^2$) in our leapfrogging example (see Figure 6). We did not observe any significant blur over the entire simulation. Further, even if the Clebsch variables get blurred slightly over time, their level-set values (which correspond to the vortex surface) won't be affected drastically.

To further verify the role of advection in stabilizing our system, we also plot the energy spectrum of $\mathrm{Re}(\psi_1)$ and s in Figure 7. No significant aliasing is observed in the plots, which could be that our advection scheme smooths the small-scale energies.

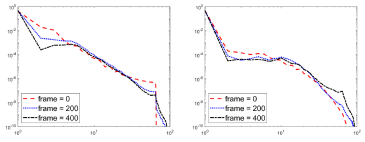


Fig. 7. The energy spectrum of $\mathrm{Re}(\psi_1)$ (left figure) and s (right figure) of the trefoil knot example at frames 0, 200, and 400.

6.5 Performance Analysis

Compared with a standard grid-based solver with one advection step and one projection step, our method only has one additional advection step for $\boldsymbol{\psi}$, one additional step to map $\boldsymbol{u}_m$ from $\boldsymbol{\psi}$, and one (optional) additional projection step if there is a free surface. Because the performance bottleneck of a standard grid-based solver is the projection step, additional advection-style steps add marginal cost to the entire pipeline. To show the performance difference, we use a standard grid-based solver to simulate the leapfrogging example with the same settings. The average time per frame of the grid-based solver is $2.62s$, close to that of our method ($2.9s$). We implemented the projection steps in parallel on GPU by building a multigrid preconditioned conjugate gradient solver to boost the system's performance. Our free-surface solver is approximately twice slow as a standard grid-based solver due to the additional projection step. We want to emphasize that because solving q and φ shares the same projection matrix with the difference on the right-hand side only, the numerical solvers for these two steps can be easily reused, and the time cost for assembling matrices can be reduced. We provide the timing statistics and grid resolution information for all the examples in Table 2.

7 DISCUSSIONS AND CONCLUSIONS

Parameters. Our method uses three parameters, as we showed in Table 1. The values of these parameters can be seen in Table 2. The first parameter $\hbar$ draws from the definition of the Clebsch wave function that controls the strength of the vorticities (the same parameter was used in [Chern et al. 2016]). The second parameter α controls the soft projection scale for the gauge wave function $\boldsymbol{\psi}$. $\alpha = 0$ indicates no projection, which amounts to no reinitialization step in a standard impulse method, and might cause the gradual deviation between the gauge variable and the physical velocity as time evolves. The third parameter β controls the blending ratio between a wave-function-mapped velocity and an advected velocity. We observed the numerical efforts of this parameter in our experiments are to enhance the convective motions behind obstacles of a flow field.

Relation to a grid-based solver. The vast majority of the numerical infrastructures we have developed in our Clebsch gauge solver

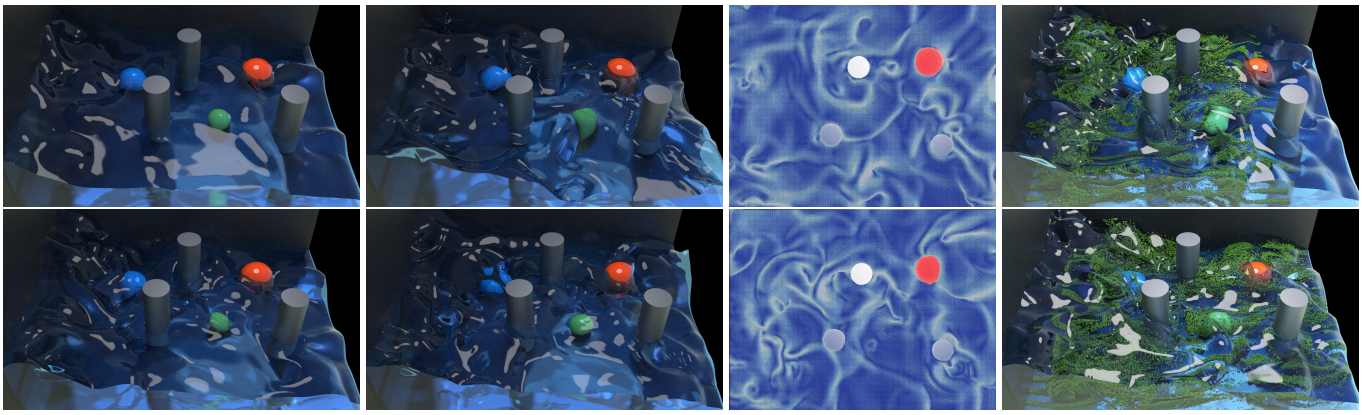


Fig. 8. Left two columns: Results of a stream (with a surface tension of 0.05) flowing over multiple obstacles at frames 150, 240, 440, and 700; The 3rd column: Additional visualizations of the stream surface using the top view; The 4th column: Additional visualizations of the underwater flow using marker particles. The grid size for this example is $256 \times 192 \times 128$. A host of underwater vortices due to the interactions between fluid and obstacles are created.

are based on a standard grid-based solver. In particular, our solver fully reuses the numerical implementation of its advection and projection modules. In this sense, our solver can be understood as an enhanced grid-based solver with its outstanding ability in capturing and preserving (vortical) flow structures. As demonstrated in our series of analytical vortex filament examples, our solver can produce vorticity evolutions with clear and faithful physical structures, which outperforms most of the vorticity confinement techniques used in a purely Eulerian setting.

Relation to incompressible Schrödinger's flow (ISF). We build our Clebsch gauge solver based on the Clebsch wave function proposed by Chern et al. in their ISF work [2016]. Our method's vortex tracking and evolving abilities stem from this fundamental geometric design in [Chern et al. 2016] and share similar visual effects according to our analytical vortex filaments experiments. Our Clebsch gauge solver differs from the ISF method in two aspects: 1) we solve the incompressible fluid equations instead of the Schrödinger's equations; 2) we developed a gauge framework to use the wave function as a gauge variable to evolve the system, which allows us to apply the wave-function technique to solve general fluid simulation problems such as free surface and surface tensions.

Limitation and future work. Our current Clebsch gauge solver relies on the numerical viscosity introduced by the semi-Lagrangian advection to model the viscosity in the system. In the future, we plan to devise more accurate viscous solvers under this gauge framework to accommodate more complicated flow phenomena. Using the additional parameters is another limitation that could be improved in our future work. Also, our current implementation is only for a Cartesian grid. We plan to study the different types of data structures, such as the hybrid Eulerian-Lagrangian framework, to further decouple the gauge and the physical quantity evolution on the level of data structures. Furthermore, we aim to devise more suitable and flexible gauge variables and their numerical solvers to facilitate fluid simulation applications exhibiting complicated evolving flow structures.

ACKNOWLEDGMENTS

We thank all the anonymous reviewers for their constructive comments. We acknowledge the funding support from NSF 1919647. We credit the Houdini education licenses for the video generation.

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*. 87–96.

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

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

Axel Brandenburg. 2010. Magnetic field evolution in simulations with euler potentials. *MON. NOT. R. ASTRON. SOC.* 401 (2010), 347–354.

Robert Bridson, Jim Houriham, and Marcus Nordenstam. 2007. Curl-noise for procedural fluid flow. *ACM Transactions on Graphics (ToG)* 26, 3 (2007).

Tyson Brochu, Todd Keeler, and Robert Bridson. 2012. Linear-time smoke animation with vortex sheet meshes. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Citeseer, 87–95.

Tomas F. Buttke. 1993. *Velicity Methods: Lagrangian Numerical Methods which Preserve the Hamiltonian Structure of Incompressible Fluid Flow.* Springer Netherlands, Dordrecht, 39–57.

Thomas F. Buttke and Alexandre J. Chorin. 1993. Turbulence calculations in magnetization variables. *Applied Numerical Mathematics* 12, 1 (1993), 47 – 54. SPECIAL ISSUE.

C. Cartes, M. D. Bustamante, and M. E. Brachet. 2007. Generalized Eulerian-Lagrangian description of Navier-Stokes dynamics. *Phys. Fluids* 19 (2007), 077101.

A. Chern. 2017. *Fluid Dynamics with Incompressible Schrödinger Flow.* Ph.D. Dissertation. California institute of technology.

A. Chern, F. Knöppel, U. Pinkall, and P. Schröder. 2017. Inside fluids: Clebsch maps for visualization and processing. *ACM Trans. Graph.* 36 (2017), 142.

A. Chern, F. Knöppel, U. Pinkall, P. Schröder, and S. Weißmann. 2016. Schrödinger's smoke. *ACM Trans. Graph.* 35 (2016), 77.

A. Clebsch. 1859. Ueber die Integration der hydrodynamischen Gleichungen. *J. Reine Angew. Math.* 56 (1859), 1–10.

Georges-Henri Cottet, Petros D Koumoutsakos, et al. 2000. *Vortex methods: theory and practice.* Vol. 8. Cambridge university press Cambridge.

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

Aleksandar Donev, Andy Nonaka, Yifei Sun, Thomas Fai, Alejandro Garcia, and John Bell. 2014. Low Mach number fluctuating hydrodynamics of diffusively mixing fluids. *Communications in Applied Mathematics and Computational Science* 9, 1 (May 2014), 47–105.

Weinan E and Jian-Guo Liu. 1997. Finite Difference Schemes for Incompressible Flows in the Velocity–Impulse Density Formulation. *J. Comput. Phys.* 130, 1 (1997), 67 – 76.

Sharif Elcott, Yiying Tong, Eva Kanso, Peter Schröder, and Mathieu Desbrun. 2007. Stable, circulation-preserving, simplicial fluids. *ACM Transactions on Graphics (TOG)* 26, 1 (2007).
Ronald Fedkiw, J. Stam, and H. Jensen. 2001. Visual simulation of smoke. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001).
N. Foster and Ronald Fedkiw. 2001. Practical animation of liquids. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001).
C. Fu, Q. Guo, Theodore F. Gast, Chenfanfu Jiang, and J. Teran. 2017. A polynomial particle-in-cell method. *ACM Transactions on Graphics (TOG)* 36 (2017), 1–12.
S. Gagniere, David Hyde, A. Marquez-Razon, C. Jiang, Z. Ge, X. Han, Q. Guo, and J. Teran. 2020. A Hybrid Lagrangian/Eulerian Collocated Advection and Projection Method for Fluid Simulation. *ArXiv* abs/2003.12227 (2020).
C. R. Graham and F. S. Henyey. 2000. Clebsch representation near points where the vorticity vanishes. *Phys. Fluids* 12 (2000), 744–746.
P. He and Y. Yang. 2016. Construction of initial vortex-surface fields and Clebsch potentials for flows with high-symmetry using first integrals. *Phys. Fluids* 28 (2016), 037101.
H. Hopf. 1931. Über die Abbildungen der Dreidimensionalen Sphäre auf die Kugelfläche. *Math. Ann.* 104 (1931), 637–665.
J. Jeong and F. Hussain. 1995. On the identification of a vortex. *J. Fluid. Mech.* 285 (1995), 69–94.
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 (2015), 1–10.
Myungjoo Kang, Ronald P Fedkiw, and Xu-Dong Liu. 2000. A boundary condition capturing method for multiphase incompressible flow. *Journal of Scientific Computing* 15, 3 (2000), 323–360.
H. Kedia, D. Foster, M. R. Dennis, and W. T. M. Irvine. 2016. Weaving knotted vector fields with tunable helicity. *Phys. Rev. Lett.* 117 (2016), 274501.
ByungMoon Kim, Y. Liu, I. Llamas, and J. Rossignac. 2005. FlowFixer: Using BFECC for Fluid Simulation. In *NPH*.
Doyub Kim, Oh-Young Song, and Hyeongseok Ko. 2009. Stretching and wiggling liquids. *ACM SIGGRAPH Asia 2009 papers* (2009).
Theodore Kim, Nils Thürey, Doug James, and Markus Gross. 2008. Wavelet turbulence for fluid simulation. *ACM Transactions on Graphics (TOG)* 27, 3 (2008), 1–6.
Dustin Kleckner and William TM Irvine. 2013. Creation and dynamics of knotted vortices. *Nat. Phys.* 9, 4 (2013), 253–258.
P. Robert Kotiuga. 1991. Clebsch potentials and the visualization of three-dimensional solenoidal vector fields. *IEEE T. MAGN* 27 (1991), 3986–3989.
Petros Koumoutsakos, Georges-Henri Cottet, and Diego Rossinelli. 2008. Flow simulations using particles-Bridging Computer Graphics and CFD. In *SIGGRAPH 2008-35th International Conference on Computer Graphics and Interactive Techniques*. ACM, 1–73.
G. A. Kuz'min. 1983. Ideal incompressible hydrodynamics in terms of the vortex momentum density. *Phys. Lett. A* 96 (1983), 88–90.
H. Lamb. 1932. *Hydrodynamics* (6 ed.). Cambridge University Press.
A Leonard. 1980. Vortex methods for flow simulation. *J. Comput. Phys.* 37, 3 (1980), 289–335.
TT Lim. 1989. An experimental study of a vortex ring interacting with an inclined wall. *Exp. Fluids* 7, 7 (1989), 453–463.
TT Lim. 1997. A note on the leapfrogging between two coaxial vortex rings at low Reynolds numbers. *Phys. Fluids* 9, 1 (1997), 239–241.
Miao'er Liu, Yu-Xin Ren, and Hanxin Zhang. 2004. A class of fully second order accurate projection methods for solving the incompressible Navier–Stokes equations. *J. Comput. Phys.* 200, 1 (2004), 325 – 346.
John H. Maddocks and Robert L. Pego. 1995. An unconstrained Hamiltonian formulation for incompressible fluid flow. *Comm. Math. Phys.* 170, 1 (1995), 207–217.
Patrick Mullen, Keenan Crane, Dmitry Pavlov, Yiying Tong, and Mathieu Desbrun. 2009. Energy-preserving integrators for fluid animation. *ACM Transactions on Graphics (TOG)* 28, 3 (2009), 1–8.
V I Oseledets. 1989. On a new way of writing the Navier-Stokes equation. The Hamiltonian formalism. *Russian Mathematical Surveys* 44, 3 (jun 1989), 210–211.
S. Park and M. Kim. 2005. Vortex fluid for gaseous phenomena. In *SCA '05*.
Tobias Pfaff, Nils Thuerey, and Markus Gross. 2012. Lagrangian vortex sheets for animating fluids. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–8.
Ziyin Qu, Xinxin Zhang, Ming Gao, Chenfanfu Jiang, and Baoquan Chen. 2019. Efficient and conservative fluids using bidirectional mapping. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–12.
R. Saye. 2016. Interfacial gauge methods for incompressible fluid dynamics. *Sci. Adv.* 2 (2016), e1501869.
Robert Saye. 2017a. Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid–structure interaction, and free surface flow: Part I. *J. Comput. Phys.* 344 (2017), 647 – 682.
Robert Saye. 2017b. Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid–structure interaction, and free surface flow: Part II. *J. Comput. Phys.* 344 (2017), 683 – 723.
A. Selle, Ronald Fedkiw, ByungMoon Kim, Y. Liu, and J. Rossignac. 2008. An Unconditionally Stable MacCormack Method. *Journal of Scientific Computing* 35 (2008), 350–371.
Andrew Selle, Nick Rasmussen, and Ronald Fedkiw. 2005. A vortex particle method for smoke, water and explosions. In *ACM SIGGRAPH 2005 Papers*. 910–914.
C. B. Smiet, S. Candelaresi, and D. Bouwmeester. 2017. Ideal relaxation of the Hopf fibration. *Phys. Plasmas* 24 (2017), 072110.
C. B. Smiet, S. Candelaresi, A. Thompson, J. Swearngin, J.W. Dalhuisen, and D. Bouwmeester. 2015. Self-organizing knotted magnetic structures in plasma. *Phys. Rev. Lett.* 115 (2015), 095001.
J. Stam. 1999. Stable fluids. In *SIGGRAPH '99*.
Mark J Stock, Werner JA Dahm, and Grétar Tryggvason. 2008. Impact of a vortex ring on a density interface using a regularized inviscid vortex sheet method. *J. Comput. Phys.* 227, 21 (2008), 9021–9043.
D.M. Summers. 2000. A Representation of Bounded Viscous Flow Based on Hodge Decomposition of Wall Impulse. *J. Comput. Phys.* 158, 1 (2000), 28 – 50.
D M Summers and A J Chorin. 1996. Numerical vorticity creation based on impulse conservation. *Proceedings of the National Academy of Sciences* 93, 5 (1996), 1881–1885.
S. Weißmann and U. Pinkall. 2010. Filament-based smoke with vortex shedding and variational reconnection. *ACM Trans. Graph.* 29 (2010), 115.
S. Xiong and Y. Yang. 2017. The boundary-constraint method for constructing vortex-surface fields. *J. Comput. Phys.* 339 (2017), 31–45.
S. Xiong and Y. Yang. 2019. Identifying the tangle of vortex tubes in homogeneous isotropic turbulence. *J. Fluid Mech.* 874 (2019), 952–978.
S. Xiong and Y. Yang. 2020. Evolution and helicity analysis of linked vortex tubes in viscous flows. *Sci. Sin-Phys. Mech. Astron.* 50 (2020), 040005.
Y. Yang and D. I. Pullin. 2011. Evolution of vortex-surface fields in viscous Taylor–Green and Kida–Pelz flows. *J. Fluid Mech.* 685 (2011), 146–164.
V. E. Zakharov and E. A. Kuznetsov. 1997. Hamiltonian formalism for nonlinear waves. *Phys.-Usp.* 40 (1997), 1087–1116.
Xinxin Zhang, Robert Bridson, and Chen Greif. 2015. Restoring the missing vorticity in advection-projection fluid solvers. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–8.
Y. Zhao, S. Xiong, Y. Yang, and S. Chen. 2018. Sinuous distortion of vortex surfaces in the lateral growth of turbulent spots. *Phys. Rev. Fluids* 3 (2018), 074701.
Y. Zhao, Y. Yang, and S. Chen. 2016. Vortex reconnection in the late transition in channel flow. *J. Fluid Mech.* 802 (2016), R4.
Yongning Zhu and Robert Bridson. 2005. Animating sand as a fluid. *ACM Transactions on Graphics (TOG)* 24, 3 (2005), 965–972.

A GEOMETRIC DESCRIPTION OF ψ

Like the original Clebsch maps, the wave function in Equation (1) encodes a fundamental geometric description of the vorticity field $\boldsymbol{\omega} = \boldsymbol{\nabla} \times \boldsymbol{u}$. Specifically, Equation (1) can be transformed into vorticity potentials $\boldsymbol{s} = (s_1, s_2, s_3)$ by Hopf map [Hopf 1931]:

$$s_1 = a^2 + b^2 - c^2 - d^2,\ s_2 = 2(bc - ad),\ s_3 = 2(ac + bd), \tag{20}$$

which are exact vortex surface fields owing to the fact that:

$$\boldsymbol{\omega} = \frac{\hbar}{2}(s_1 \boldsymbol{\nabla} s_2 \times \boldsymbol{\nabla} s_3 + s_2 \boldsymbol{\nabla} s_3 \times \boldsymbol{\nabla} s_1 + s_3 \boldsymbol{\nabla} s_1 \times \boldsymbol{\nabla} s_2), \tag{21}$$

and

$$\boldsymbol{\omega} \cdot \boldsymbol{\nabla} s_p = 0,\ p = 1, 2, 3. \tag{22}$$

Gauge transformation of $\boldsymbol{\phi}$. In addition, Equation (20) satisfies the gauge invariance because $\boldsymbol{\phi}$ and its gauge transformation

$$\boldsymbol{\psi} = \boldsymbol{\phi} e^{i\varphi/\hbar} \tag{23}$$

corresponds to the same $\boldsymbol{s}$, thus the same $\boldsymbol{\omega}$, where φ is a scalar gauge function. Moreover, the gauge transformation of the wave function $\boldsymbol{\phi} \rightarrow \boldsymbol{\phi} e^{i\varphi/\hbar}$ corresponds to the gauge transformation of the velocity $\boldsymbol{u} \rightarrow \boldsymbol{u} + \boldsymbol{\nabla}\varphi$, and the incompressible condition of velocity $\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0$ is equivalent to $\langle i\boldsymbol{\phi}, \Delta\boldsymbol{\phi} \rangle_{\mathbb{R}} = 0$.