Clebsch Gauge Fluid on Particle Flow Maps

ZHIQI LI, Georgia Institute of Technology, USA
CANDONG LIN, Georgia Institute of Technology, USA
DUOWEN CHEN, Georgia Institute of Technology, USA
XINYI ZHOU, Georgia Institute of Technology, USA
SHIYING XIONG, Zhejiang University, USA
BO ZHU, Georgia Institute of Technology, USA

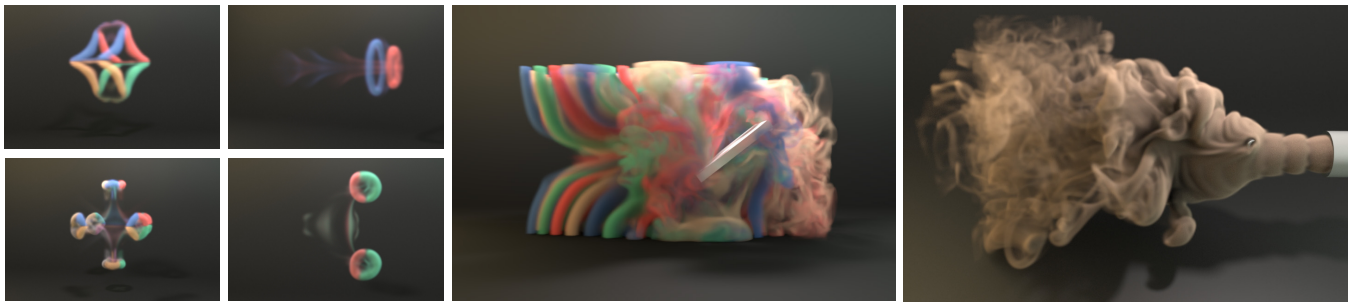


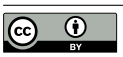
Fig. 1. Our Clebsch wave function-based particle flow map can perfectly preserve vortices even at very low grid resolutions with precise convection. We demonstrate that our method is capable of simulating complex vortex interactions and motion under extremely low resolution (#resolution ≤ 64) (left), handling vortex shedding caused by moving obstacles leading to turbulent smoke (middle), and capturing the vortical structure shedded by the obstacle (right).

We propose a novel gauge fluid solver that evolves Clebsch wave functions on particle flow maps (PFMs). The key insight underlying our work is that particle flow maps exhibit superior performance in transporting point elements—such as Clebsch components—compared to line and surface elements, which were the focus of previous methods relying on impulse and vortex gauge variables for flow maps. Our Clebsch PFM method incorporates three main contributions: a novel gauge transformation enabling accurate transport of wave functions on particle flow maps, an enhanced velocity reconstruction method for coarse grids, and a PFM-based simulation framework designed to better preserve fine-scale flow structures. We validate the Clebsch PFM method through a wide range of benchmark tests and simulation examples, ranging from leapfrogging vortex rings and vortex reconnections to Kelvin–Helmholtz instabilities, demonstrating that our method outperforms its impulse- or vortex-based counterparts on particle flow maps, particularly in preserving and evolving small-scale features.

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

Additional Key Words and Phrases: Gauge Method, Clebsch Fluid, Flow Map Method, Particle Flow Map

Authors' Contact Information: Zhiqi Li, zli3167@gatech.edu, Georgia Institute of Technology, USA; Candong Lin, lincandong@outlook.com, Georgia Institute of Technology, USA; Duowen Chen, dchen322@gatech.edu, Georgia Institute of Technology, USA; Xinyi Zhou, xzhou487@gatech.edu, Georgia Institute of Technology, USA; Shiying Xiong, shiying.xiong@zju.edu.cn, Zhejiang University, USA; Bo Zhu, bo.zhu@gatech.edu, Georgia Institute of Technology, USA.



ACM 1557-7368/2025/8-ART
https://doi.org/10.1145/3731194

ACM Reference Format:
Zhiqi Li, Candong Lin, Duowen Chen, Xinyi Zhou, Shiying Xiong, and Bo Zhu. 2025. Clebsch Gauge Fluid on Particle Flow Maps. *ACM Trans. Graph.* 44, 4 (August 2025), 12 pages. https://doi.org/10.1145/3731194

1 Introduction

The particle flow map (PFM) method [Zhou et al. 2024] represents flow maps using particle trajectories, facilitating long-range vorticity preservation within an Eulerian-Lagrangian framework. The key mechanism enabling PFM to preserve vortical structures is its APIC-style [Jiang et al. 2015] mapping approach, where both the impulse and its gradients are transported on particles through long-rang mapping. In the current literature, all PFM frameworks are coupled with an impulse formulation of the incompressible Navier-Stokes equations (e.g., [Chen et al. 2024; Li et al. 2024a; Zhou et al. 2024]).

One of the key challenges the current particle flow map (PFM) methods are facing lies in addressing the evolution of impulse derivatives on particles, which is critical for ensuring the accuracy of the advection scheme. The current PFM framework focuses solely on the evolution of the mixed high-order term in impulse derivatives with particles, while neglecting the pure high-order terms such as the Hessian of flow maps. This omission is primarily due to the mathematical complexity of expressing Hessian evolution and the practical difficulties associated with implementing it within the particle-grid discretization process.

To further improve the advection accuracy of the particle flow map method and enhance its capability for preserving vorticity, we propose a novel approach to address the difficulties associated with the particle evolution of higher-order derivatives. Instead of directly tackling the computational challenges tied to the geometric nature of impulse derivatives — represented as surface elements (or 1-forms in

differential geometry) with stretching terms in their evolution that make higher-order derivative calculations on particles inherently difficult — we shift our focus to an alternative perspective. If the transported physical quantities are neither line nor surface elements but instead simple point elements (i.e., 0-forms carried on particles), the problem becomes significantly more manageable. Point elements, characterized by having zero material derivative, align naturally with the Lagrangian nature of particles, allowing both their value and derivative evolution to be handled straightforwardly in a PFM framework. This shift in perspective eliminates the complications of geometric stretching terms and provides a more robust framework for improving PFM's advection accuracy.

Motivated by this observation on the nature of particle flow maps, we propose a novel Clebsch gauge variable to reformulate the incompressible flow equations. By adopting this Clebsch representation and transporting it on particles, we aim to leverage its inherent advantages, particularly its point-based nature, which facilitates the handling of long-range flow maps. Introduced by Clebsch in the mid-19th century, a Clebsch representation captures the geometric properties of vorticity and helicity, enabling the velocity field of an incompressible fluid to be described as a combination of scalar functions (see, e.g., [Chern et al. 2016a; Xiong et al. 2022; Yang et al. 2021]). Because Clebsch potentials are inherently point-based scalar quantities, their gradient mappings are simpler than those of surface or line elements. This property makes Clebsch representations particularly compatible with PFM, which is well-suited for advecting point elements. By reformulating the governing equations of fluid motion in terms of Clebsch wave functions, it becomes possible to harness the strengths of PFM while avoiding the challenges associated with higher-order element advection.

Our system comprises three key components: a reformulated Clebsch gauge model that ensures Lagrangian, point-based evolution, enabling accurate advection of wave functions and their gradients within the particle flow map (PFM) framework; a novel velocity reconstruction method from Clebsch wave functions that improves accuracy in vorticity-dominated flows, particularly on coarse grids; and a Clebsch-based simulation framework that outperforms existing gauge methods in accommodating vortical simulations within the PFM pipeline. The efficacy of the Clebsch PFM method has been demonstrated through benchmark tests and simulation scenarios, showcasing its ability to preserve fine-scale flow structures and handle complex vortical dynamics.

The primary contributions of this study are as follows:

(1) **Gauge Transformation for Clebsch Wave Functions**: We develop a novel gauge transformation that aligns the evolution of Clebsch wave functions with the particle flow map (PFM) framework, enabling accurate advection of scalar quantities and their gradients.
(2) **Velocity Reconstruction Scheme**: We introduce a robust method for converting Clebsch wave functions into velocity fields, significantly improving simulation accuracy and fidelity in vorticity-dominated scenarios.
(3) **Clebsch-Based PFM Framework**: By leveraging the point-based nature of Clebsch wave functions, we establish a new PFM framework that achieves state-of-the-art performance in vorticity-preserving simulations.

2 Related Work

Eulerian-Lagrangian Methods. Hybrid methods in fluid simulation combine the strengths of Eulerian and Lagrangian approaches to achieve both flexibility and efficiency. Since the introduction of PIC [Harlow 1962] and FLIP [Brackbill and Ruppel 1986] to the graphics community by Zhu and Bridson [2005], these techniques have been widely adopted for their ability to handle complex fluid behaviors. The material point method (MPM), which builds on PIC and FLIP, has been used to simulate various phenomena, including snow dynamics [Stomakhin et al. 2013], viscoelastic materials [Yue et al. 2015], and magnetized fluids [Sun et al. 2021]. Researchers have also worked to improve particle-to-grid transfers, addressing issues like maintaining accurate motion [Jiang et al. 2015], reducing energy loss [Fei et al. 2021], and conserving volume [Qu et al. 2022]. These advancements make hybrid methods a powerful tool for simulating realistic and complex physical interactions.

Flow Map Methods. Flow maps, which represent the movement of particles over time, are effective in reducing errors in interpolation and advection. First introduced in computational physics [Wiggert and Wylie 1976] and later adopted in computer graphics [Hachisuka 2005], these methods trace particle motion between frames to map temporal changes. Recent work has extended flow maps to velocity fields [Qu et al. 2019; Sato et al. 2018], and their application to covector fluids [Nabizadeh et al. 2022] has proven particularly useful for capturing vorticity-rich phenomena. Advances such as neural buffers [Deng et al. 2023] and particle-based techniques [Li et al. 2024a; Zhou et al. 2024] have further enhanced efficiency, enabling more accurate simulations with reduced memory requirements. More recently, researchers have used flow maps to improve simulations involving dissipative forces [Li et al. 2024b], solid-fluid interactions [Chen et al. 2024], vortex methods [Wang et al. 2024], and two-phase flows [Sun et al. 2024].

Clebsch Gauge Fluid. Clebsch [1859] introduced a vector representation of fluid dynamics in the Eulerian frame, providing a Lagrangian and Hamiltonian description that captures important geometric properties of vorticity fields. Spherical Clebsch maps, proposed by [Chern et al. 2017, 2016a], effectively represent velocity-vorticity fields with non-trivial helicity and serve as a foundation for vortex surface evolution. These maps extend traditional Clebsch potentials, which struggle with knotted fields and vanishing vorticity points, by incorporating multi-component Clebsch variables [Cartes et al. 2007; Zakharov and Kuznetsov 1997]. Combining Clebsch maps with gauge fluid methods [Saye 2016], Yang et al. [2021] successfully simulated surface-tension flows, while Xiong et al. [2022] applied the approach from Tao et al. [2021] to model free-surface vortical flows with knotted velocity fields.

3 Background and Motivation

3.1 Flow Map Foundation

Consider a fluid with velocity field $\mathbf{u}_t(\mathbf{x}) = \mathbf{u}(\mathbf{x}, t)$ at all times $t > 0$. A fluid particle initially located at $\mathbf{x}_s \in \Omega_s$ at time s moves to $\mathbf{x}_r \in \Omega_r$

at time r, where Ω_s and Ω_r are the fluid domains at times s and r, respectively. The positions $\mathbf{x}_s$ and $\mathbf{x}_r$ are related via the forward flow map $\phi_{s\to r} : \Omega_s \to \Omega_r$ and the backward flow map $\psi_{r\to s} : \Omega_r \to \Omega_s$, satisfying $\phi_{s\to r}(\mathbf{x}_s) = \mathbf{x}_r$ and $\psi_{r\to s}(\mathbf{x}_r) = \mathbf{x}_s$. The Jacobian matrices of the flow maps, denoted by $\mathcal{F}_{s\to r}(\mathbf{x}_s) = \partial\phi_{s\to r}(\mathbf{x}_s)/\partial\mathbf{x}_s$ and $\mathcal{T}_{r\to s}(\mathbf{x}_r) = \partial\psi_{r\to s}(\mathbf{x}_r)/\partial\mathbf{x}_r$, satisfy the following evolution equations:

$$\begin{cases} \frac{\partial\phi_{s\to t}(\mathbf{x})}{\partial t} = \mathbf{u}(\phi_{s\to t}(\mathbf{x}), t), & \phi_{s\to s}(\mathbf{x}) = \mathbf{x}, \\ \frac{\partial\mathcal{F}_{s\to t}(\mathbf{x})}{\partial t} = \nabla\mathbf{u}(\phi_{s\to t}(\mathbf{x}), t)\mathcal{F}_{s\to t}(\mathbf{x}), & \mathcal{F}_{s\to s}(\mathbf{x}) = \mathbf{I}, \end{cases} \tag{1}$$

$$\begin{cases} \frac{D\psi_{t\to s}(\mathbf{x})}{Dt} = 0, & \psi_{s\to s}(\mathbf{x}) = \mathbf{x}, \\ \frac{D\mathcal{T}_{t\to s}(\mathbf{x})}{Dt} = -\mathcal{T}_{t\to s}(\mathbf{x})\nabla\mathbf{u}(\mathbf{x}, t), & \mathcal{T}_{s\to s}(\mathbf{x}) = \mathbf{I}. \end{cases} \tag{2}$$

In (2), the presence of advection terms poses challenges for accurate evolution on grids. The PFM method, introduced in recent studies [Chen et al. 2024; Li et al. 2024a,b; Zhou et al. 2024], was developed to address this issue by discretizing flow maps into a collection of particles. Each particle, denoted as $p \in \mathbb{P}$ where $\mathbb{P}$ is the set of all particles, is located at position $\mathbf{x}_p(t)$ at time t. The particle carries Jacobian matrices $\mathcal{F}_p(t) = \mathcal{F}_{s\to t}(\mathbf{x}_p(s))$ and $\mathcal{T}_p(t) = \mathcal{T}_{t\to s}(\mathbf{x}_p(t))$, evolving from its initial position $\mathbf{x}_{p,s}$ at time s. Reformulating (2) for particles yields the following system of equations:

$$\begin{cases} \frac{d\mathbf{x}_p(t)}{dt} = \mathbf{u}(\mathbf{x}_p(t), t), & \mathbf{x}_p(s) = \mathbf{x}_{p,s}, \\ \begin{cases} \frac{d\mathcal{F}_p(t)}{dt} \\ \frac{d\mathcal{T}_p(t)}{dt} \end{cases} = \begin{cases} \nabla\mathbf{u}(\mathbf{x}_p(t), t)\mathcal{F}_p(t), \\ -\mathcal{T}_p(t)\nabla\mathbf{u}(\mathbf{x}_p(t), t), \end{cases} & \mathcal{F}_p(s) = \mathcal{T}_p(s) = \mathbf{I}. \end{cases} \tag{3}$$

Here, the position of the particle itself, $\mathbf{x}_p(t)$, carries the information of the flow maps, denoted as $\phi_p(t) = \phi_{s\to t}(\mathbf{x}_p(s)) = \mathbf{x}_p(t)$ and $\psi_p(t) = \psi_{t\to s}(\mathbf{x}_p(t)) = \mathbf{x}_p(s)$. Since Equation 3 does not contain the advection terms, these equations can be solved accurately by direct time integration, eliminating the need for semi-Lagrangian methods to handle the advection term.

3.2 Quantity Transport on Particle Flow Maps

Flow maps enable precise computation of physical quantities. Instead of using semi-Lagrangian methods to compute the advection of a physical quantity ξ, which require repeated interpolation and lead to accumulated errors, the flow map method [Deng et al. 2023; Li et al. 2023; Zhou et al. 2024] directly maps the initial field ξ_s to the current field, avoiding repeated interpolation as:

$$\xi_t^M(\mathbf{x}) = M_{s\to t}[\xi_s; \psi_{t\to s}, \mathcal{T}_{t\to s}, ...](\mathbf{x}). \tag{4}$$

Here, $M_{s\to t}$ is the mapping operator, acting on the initial field ξ_s and using the flow map to obtain the current value through mapping; the expression after the semicolon represents the flow map information required for mapping. $M_{s\to t}$ takes different forms for different quantities. For example, for the level set ζ_t [Li et al. 2023], $M_{s\to t}$ is given by $\zeta_t^M(\mathbf{x}) = M_{s\to t}[\zeta_s; \psi_{t\to s}](\mathbf{x}) = \zeta_s(\psi_{t\to s}(\mathbf{x}))$.

In PFM, with flow maps represented by particles, physical quantities ξ are also transported on particles as $\xi_p(t)$, with the initial value $\xi_{p,s}$ carried by each particle:

$$\xi_p^M(t) = M_{s\to t}^p[\xi_{p,s}; \psi_p(t), \mathcal{T}_p(t), ...](t). \tag{5}$$

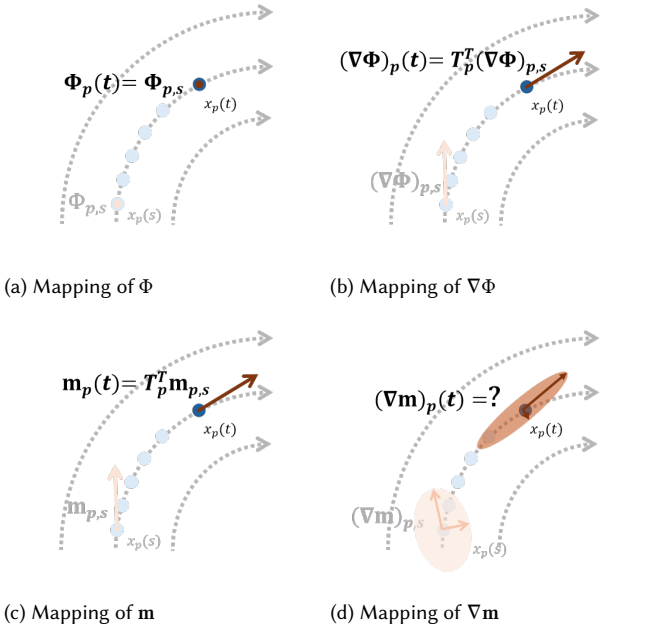


Fig. 2. The comparison between the mapping of impulse m (1-form), gradient of impulse ∇m (complex 2-order tensor), wave function Φ (0-form), and the gradient of wave function $\nabla\Phi$ (1-form). The gradient ∇m cannot be easily mapped, meaning impulse m is unsuitable for convection with the particle flow map, which requires mapping both a quantity and its gradient.

Here, $M_{s\to t}^p$, acting as the mapping operator on particle p, is induced by $M_{s\to t}$ by substituting ξ and the flow map representation in $M_{s\to t}$ with the corresponding quantities carried by the particle. For example, mapping for the level set ζ_t is represented as $\zeta_p^M(t) = M_{s\to t}^p[\zeta_{p,s}; \psi_p(t)](t) = \zeta_{p,s}$.

Since using only particles cannot efficiently and accurately compute gradients of physical quantities, the PFM method primarily relies on the grid for gradient-related computations. Let the grid in PFM be denoted as G with a spacing of Δx, and let the set of cell centers be represented as $\mathbb{G}$. For any field v, we use the subscript g to denote its value at the center $g \in \mathbb{G}$ at position $\mathbf{x}_g$. In PFM, after computing ξ_t^M on particles, ξ_p^M must be interpolated onto the grid for further computations related to the gradient of ξ_t^M. To ensure accurate particle-to-grid (P2G) transfer, PFM requires knowledge of $\nabla\xi_t^M$ on particles, denoted as $(\nabla\xi)_p^M(t)$, and employs an APIC-style P2G formulation for precise interpolation of ξ_t^M [Zhou et al. 2024]:

$$\xi_g^M(t) = \sum_{p\in\mathbb{N}_g} (\xi_p^M(t) + (\nabla\xi)_p^M(t)(\mathbf{x}_g - \mathbf{x}_p))w(\mathbf{x}_g, \mathbf{x}_p), \tag{6}$$

where $w(\mathbf{x}_g, \mathbf{x}_p) = w(\|\mathbf{x}_g - \mathbf{x}_p\|)$ is a radial compactly supported kernel function, and $\mathbb{N}_g$ represents the neighboring particles of grid point g, defined as $\mathbb{N}_g = \{p \mid w(\mathbf{x}_g, \mathbf{x}_p) \geq 0\}$.

The gradient $(\nabla\xi)_p^M(t)$ must also be computed through mapping for accuracy. Since $(\nabla\xi)_t^M$ and ξ_t^M require flow maps of different lengths for $\nabla\xi$ is more sensitive to flow distortion [Zhou et al. 2024],

Algorithm 1 A Typical PFM for Quantity Transport

for each time step t, **do**
  Evolve flow maps;
  Map $\xi_p^M(t)$ from $\xi_{s,p}$; ▷ eq. 5
  Map $(\nabla\xi)_p^M(t)$ from $(\nabla\xi)_{s',p}$; ▷ eq. 7
  Calculate $\xi_g^M(t)$ by P2G $\xi_p^M(t)$ and $(\nabla\xi)_p^M(t)$; ▷ eq. 6
  Perform calculations such as velocity projection on the grid with $\xi_g^M(t)$.

PFM sets a different initial time $s' \geq s$ for mapping $(\nabla\xi)_t^M$ and carries an additional set of Jacobian matrices: $\tilde{\mathcal{F}}_p(t) = \mathcal{F}_{s'\to t}(\mathbf{x}_p(s'))$ and $\tilde{\mathcal{T}}_p(t) = \mathcal{T}_{t\to s'}(\mathbf{x}_p(t))$. With $(\nabla\xi)_{p,s'}^M$ carried, $(\nabla\xi)_p^M(t)$ can also be calculated by mapping.

$$(\nabla\xi)_p^M(t) = M_{s'\to t}^p[(\nabla\xi)_{p,s'}^M; \psi_p(t), \mathcal{T}_p(t), ...](t). \tag{7}$$

Notably, the mapping of $(\nabla\xi)$ and ξ may have different forms. The computational workflow of PFM can be summarized as in Algorithm 1.

3.3 Challenge for Impulse Particle Flow Maps

We aim at simulating incompressible fluid with particle flow map method, following the Euler equations

$$\left(\frac{\partial}{\partial t} + (\mathbf{u}\cdot\nabla)\right)\mathbf{u} = -\frac{1}{\rho}\nabla p, \qquad \nabla\cdot\mathbf{u} = 0, \tag{8}$$

where p and ρ are the pressure and density of the fluid, respectively. Prior work reformulates the Euler equations using the impulse transformation $\mathbf{m} = \mathbf{u} + \nabla\Lambda$ and computes the impulse $\mathbf{m}$ using the particle flow map. The impulse and its gradient are mapped in the following form:

$$\begin{aligned}\mathbf{m}_t(\mathbf{x}) &= M_{s\to t}[\mathbf{m}_s; \psi_{t\to s}, \mathcal{T}_{t\to s}](\mathbf{x}) = \mathcal{T}_{t\to s}^\top(\mathbf{x})\mathbf{m}_s(\psi_{t\to s}(\mathbf{x})),\\ (\nabla\mathbf{m})_t(\mathbf{x}) &= \mathcal{T}_{t\to s}^\top(\mathbf{x})(\nabla\mathbf{m})_{s'}(\psi_{t\to s'}(\mathbf{x}))\tilde{\mathcal{T}}_{t\to s'}(\mathbf{x})\\ &\quad + \mathbf{m}_{s'}(\psi_{t\to s'}(\mathbf{x}))\nabla\tilde{\mathcal{T}}_{t\to s'}(\mathbf{x}).\end{aligned} \tag{9}$$

However, the second term of $(\nabla\mathbf{m})_t(\mathbf{x})$, $\nabla\tilde{\mathcal{T}}_{t\to s'}$, the Hessian matrix of $\psi_{t\to s'}$, is not carried on the particles and cannot be computed by taking the gradient of $\tilde{\mathcal{T}}_{t\to s'}$ on the particles either, as all gradient calculations in PFM rely on the grid. As a result, the original PFM method cannot accurately compute $\nabla\mathbf{m}$ using Equation 9 and instead omits $\mathbf{m}_{s'}(\psi_{t\to s'}(x))\nabla\tilde{\mathcal{T}}_{t\to s'}(x)$, making the mapping of the gradient of the impulse inaccurate. **Therefore, for PFM, finding a new physical quantity that can represent the Euler equations and whose value and gradient can both be accurately mapped remains a significant challenge.**

4 Clebsch Gauge on Particle Flow Maps

4.1 Motivation: Scalar Point as a Better Representation

Intuitively, the reason why the gradient of impulse $\mathbf{m}$ cannot be accurately advected is that $\mathbf{m}$ itself is a 1-form [Nabizadeh et al. 2022], and consequently, its gradient is a second-order tensor which has a complex mapping form as shown in *Equation* 9. Physical quantities related to advection include 0-forms, 1-forms, and 2-forms (also referred to as point elements, surface elements, and line elements [Cortez 1995]). Level sets, impulses, and vortices are typical examples of 0-form, 1-form and 2-form respectively [1]. From Table 1, it is observed that the gradient mapping formulas for 1-forms and 2-forms are quite complex, making them unsuitable for advection using the PFM. However, the 0-form level set cannot independently describe the fluid flow described by the Euler equations. **Therefore, our motivation is to find another quantity q that can describe the motion of the fluid and satisfy the transport equation:**

$$\frac{Dq}{Dt} = 0. \tag{10}$$

For any such quantity, according to Table 1, its gradient ∇q satisfies the advection equation.

$$\frac{D\nabla q}{Dt} + (\nabla\mathbf{u})^\top\nabla q = 0, \tag{11}$$

which has the solution $\nabla q_t(\mathbf{x}) = \mathcal{T}_{t\to s}(\mathbf{x})^\top\nabla q_s(\psi_{t\to s}(\mathbf{x}))$ and can be accurately mapped using the flow map as $\nabla q_p(t) = \mathcal{T}_{t\to s,p}^\top\nabla q_{p,s}$.

We find that Clebsch wave functions can be transformed to satisfy these conditions. In the following sections, we first introduce Clebsch wave functions and their representation of fluid motion (subsection 4.2), then detail a gauge transformation that ensures the transformed wave functions satisfy Equation 10 and Equation 11 (subsection 4.3), along with their computation using PFM (subsection 4.4,4.5).

4.2 Clebsch Wave Functions

For incompressible fluid with motion following Euler equations (Equation 8) , through the spherical Clebsch map [Chern et al. 2017, 2016a; Xiong et al. 2022; Yang et al. 2021], the fluid velocity $\mathbf{u}$ can be encoded into a two-component complex-valued wave function $\Psi = (\Psi_1, \Psi_2)$, where $\Psi_1, \Psi_2 \in \mathbb{C}$. The velocity $\mathbf{u}$ can be calculated from the wave function Ψ using the following formula:

$$\mathbf{u} = \hbar\langle\nabla\Psi, i\Psi\rangle_{\mathbb{R}}, \tag{12}$$

where $\hbar$ is a tunable parameter that controls the quantization of vorticity, $\langle\Upsilon,\Phi\rangle_{\mathbb{R}} = Re(\langle\Upsilon,\Phi\rangle_{\mathbb{C}})$ represents the real part of the inner product $\langle\Upsilon,\Phi\rangle_{\mathbb{C}}$ of the wave functions, and $\langle\Upsilon,\Phi\rangle_{\mathbb{C}} = \bar{\Upsilon}_1\Phi_1 + \bar{\Upsilon}_2\Phi_2$ for any wave functions $\Upsilon = (\Upsilon_1, \Upsilon_2), \Phi = (\Phi_1, \Phi_2)$, with $\Upsilon_i, \Phi_i \in \mathbb{C}, i = 1, 2$. To make the velocity satisfy the incompressibility condition $\nabla\cdot\mathbf{u} = 0$, the wave function must satisfy the normalization constraint

$$\|\Psi\|^2 = \langle\Psi,\Psi\rangle_{\mathbb{R}} = 1, \tag{13}$$

and the solenoidal constraint

$$\langle\Delta\Psi, i\Psi\rangle_{\mathbb{R}} = 0. \tag{14}$$

The velocity-based Euler equations can be transformed into the wave function-based fluid equation [Chern et al. 2016b; Yang et al.

[1] The correspondence between 0-forms and 3-forms and their associated geometric representations—point elements and volume elements—can be ambiguous. While 3-forms are considered point elements from the perspective of integration [Yin et al. 2023], their evolution more closely resembles that of volume elements, whereas the evolution of 0-forms is better regarded as representing point elements [Wu et al. 2007]. In incompressible flows, however, both follow the same evolution due to $\nabla\cdot\mathbf{u} = 0$. Therefore, for simplicity, we uniformly refer to 0-forms as point elements in this paper.

Table 1. A table that includes the evolution equation and mapping formula for all n-forms, where n = 0, 1, 2, along with their gradients. It can be observed that the evolution equation and mapping formula for the 0-form and its gradient are simpler than those for the 1-form and 2-form.

Element	0-form	1-form	2-form
Evolution	$\frac{Dq}{Dt} = 0$	$\frac{D\mathbf{S}}{Dt} + (\nabla \mathbf{u})^\top \mathbf{S} = 0$	$\frac{D\mathbf{l}}{Dt} - (\nabla \mathbf{u})\mathbf{l} = 0$
Mapping	$q_t = q_s \circ \psi_{t\to s}$	$\mathbf{S}_t = \tilde{\mathcal{T}}^\top_{t\to s'}(\mathbf{S}_s \circ \psi_{t\to s})$	$\mathbf{l}_t = \tilde{\mathcal{T}}^{-1}_{t\to s'}(\mathbf{l}_s \circ \psi_{t\to s})$
Gradient Evolution	$\frac{D\nabla q}{Dt} + (\nabla \mathbf{u})^\top \nabla q = 0$	$\frac{D\nabla \mathbf{S}}{Dt} + (\nabla \mathbf{u})^\top \nabla \mathbf{S} + \nabla((\nabla \mathbf{u})^\top \mathbf{S}) = 0$	$\frac{D\nabla \mathbf{l}}{Dt} + (\nabla \mathbf{u})^\top \nabla \mathbf{l} - \nabla((\nabla \mathbf{u})\mathbf{l}) = 0$
Gradient Mapping	$(\nabla q)_t = \tilde{\mathcal{T}}^\top_{t\to s'}((\nabla q)_s \circ \psi_{t\to s})$	$(\nabla \mathbf{S})_t = \tilde{\mathcal{T}}^\top_{t\to s'}(\nabla \mathbf{S} \circ \psi_{t\to s'})\tilde{\mathcal{T}}_{t\to s'}$ $+\nabla \tilde{\mathcal{T}}^\top_{t\to s'}(\mathbf{S} \circ \psi_{t\to s'})$	$(\nabla \mathbf{l})_t = \tilde{\mathcal{T}}^{-1}_{t\to s'}(\nabla \mathbf{l} \circ \psi_{t\to s'})\tilde{\mathcal{T}}_{t\to s'}$ $+\nabla \tilde{\mathcal{T}}^{-1}_{t\to s'}(\mathbf{l} \circ \psi_{t\to s'})$

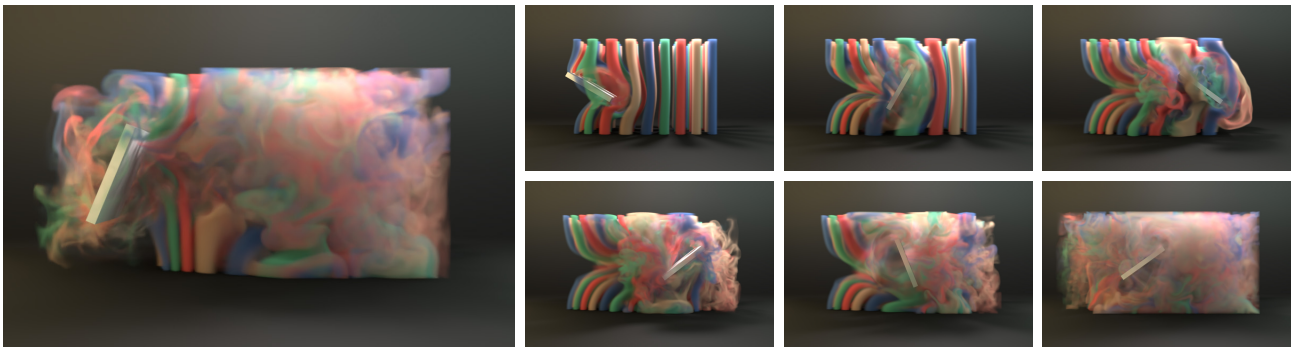


Fig. 3. **Moving Paddle**. When a paddle stirs the air, the velocity difference between the paddle and the surrounding fluid generates complex vortices, which disrupt the smoke columns and transform them into a turbulent mixture.

Fig. 4. **Four-Vortex Collision**. We simulate four colliding vortices as in [Matsuzawa et al. 2022]. The collision causes the vortices to merge into two star-shaped vortices, drifting toward opposite walls while altering their shapes. Our simulation can run at resolutions of 48, 64, and 128.

2021], which serves as the primary equation in our simulation

$$\begin{cases} \frac{D\Psi}{Dt} = -i\frac{1}{\hbar}\left(\frac{p}{\rho} - \frac{|\mathbf{u}|^2}{2}\right)\Psi, \\ \mathbf{u} = \hbar\langle \nabla\Psi, i\Psi\rangle_{\mathbb{R}}, \\ \langle \Delta\Psi, i\Psi\rangle_{\mathbb{R}} = 0. \end{cases} \tag{15}$$

4.3 Gauge Transformation

The form of Equation 15 does not match the desired scalar evolution form in Equation 10. To address this, inspired by [Yang et al. 2021], we apply a gauge transformation to the wave function Ψ to reshape Equation 15 to meet our requirements. However, the gauge transformation in [Yang et al. 2021] does not meet our needs, as it primarily addresses the free surface problem, improving the boundary conditions at the free surface and wave function still obeys the same evolution equation as Equation 15 after the transformation. **We introduce a new gauge transformation on the wave function** Ψ, defined as $\Phi(\mathbf{x}, t) = \Psi(\mathbf{x}, t)e^{i\Gamma_{s\to t}(\mathbf{x})/\hbar}$, where the potential integrator $\Gamma_{s\to t}(\mathbf{x})$ is given by $\Gamma_{s\to t}(\mathbf{x}) = \int_s^t \left(\frac{p}{\rho} - \frac{|\mathbf{u}|^2}{2}\right)(\phi_{s\to\tau}(\psi_{t\to s}(\mathbf{x})), \tau)d\tau$. This transformation converts Equation 15 into the desired form (refer to Appendix A for proof)

$$\begin{cases} \frac{D\Phi(\mathbf{x},t)}{Dt} = 0, \\ \mathbf{u}_m = \hbar\langle \nabla\Phi, i\Phi\rangle_{\mathbb{R}}, \\ \Delta\Gamma_{s\to t} = \nabla \cdot \mathbf{u}_m, \\ \mathbf{u} = \mathbf{u}_m - \nabla\Gamma_{s\to t}. \end{cases} \tag{16}$$

Combined with Equation 11, the evolution equation of $\nabla\Phi$, can be derived as $\frac{D\nabla\Phi}{Dt} + \nabla\mathbf{u}^T\nabla\Phi = 0$. Thus, Φ and $\nabla\Phi$ can be computed using the flow map as:

$$\begin{aligned} \Phi(\mathbf{x}, t) &= \Phi(\psi_{t\to s}(\mathbf{x}), s), \\ \nabla\Phi(\mathbf{x}, t) &= \mathcal{T}_{t\to s}(\mathbf{x})\nabla\Phi(\psi_{t\to s}(\mathbf{x}), s). \end{aligned} \tag{17}$$

4.4 Wave Function-Based Particle Flow Maps

Now, Equation 16 is well-suited for the PFM Algorithm 1, and we discuss how to integrate it into the PFM method. We continue to use particles $p \in \mathbb{P}$ and use a MAC grid with a spacing of Δx for the computation. The wave function is stored at the cell centers $g \in \mathbb{G}$,

Fig. 5. **Ablation Study**. This compares our full method (first row), our method without our enhanced wave function-velocity conversion (second row), and our method without gradient advection (third row). The original conversion method causes distorted rings and axis-aligned artifacts, while omitting gradient advection significantly deteriorates vortex preservation.

while the velocity is stored at the face centers $f \in \mathbb{F}$, where $\mathbb{G}$ and $\mathbb{F}$ represent the sets of cell centers and face centers, respectively. For any $g \in \mathbb{G}$ ($f \in \mathbb{F}$), its position is denoted as $\mathbf{x}_g$ ($\mathbf{x}_f$), and quantities at the cell center g (face center f) are represented using the subscript g (f). The velocity storage follows the MAC grid convention, where the X component of the velocity is stored at face centers in the X-direction, and similarly, the Y and Z components are stored at face centers in the Y and Z directions, respectively. The velocity component at face center f is represented by u_f.

At each substep t, after evolving flow maps, $\Phi_p(t)$ and $(\nabla\Phi)_p(t)$ on the particles are obtained by calculating Equation 17 with the particle flow map method

$$\begin{aligned} \Phi_p(t) &= \Phi_{p,s}, \\ (\nabla\Phi)_p(t) &= \tilde{\mathcal{T}}_p(t)^\top (\nabla\Phi)_{p,s'}, \end{aligned} \tag{18}$$

with initial values $\Phi_{p,s}$ and $(\nabla\Phi)_{p,s'}$ at times s and s', respectively, which are carried by particle $p \in \mathbb{P}$. With $\Phi_p(t)$ and $(\nabla\Phi)_p(t)$, the wave function on the grid $\Phi_g(t)$ can be computed using APIC interpolation Equation 6. Then, $\Phi_g(t)$ is converted to the grid velocity $u_{m,f}$ on each faces by numerically computing $\mathbf{u}_m = \hbar\langle\nabla\Phi, i\Phi\rangle_{\mathbb{R}}$ using the following equation [Chern et al. 2016a]:

$$u_{m,f} = \frac{\hbar}{\Delta x} \arg\langle\Phi_{g_1}, \Phi_{g_2}\rangle, \tag{19}$$

where g_1 and g_2 are the two cell centers adjacent to face f, and the direction from g_1 to g_2 is aligned with the positive normal direction of the face f.

Next, on the grid, the Poisson equation in Equation 16 for $\Gamma_{s\to t}$ is discretized on the grid and solved with a multigrid preconditioned conjugate gradient solver similar to [Zhou et al. 2024], and then the velocity projection is performed to obtain the final velocity at the current substep:

$$u_f = u_{m,f} - \frac{\Gamma_{g_1} - \Gamma_{g_2}}{\Delta x}. \tag{20}$$

Combining the above discussion into Algorithm 1, we can summarize the wave function-based PFM Algorithm 2.

Algorithm 2 Wave Function-Based PFM

1: **for** each time step t, **do**
2: Evolve flow maps;
3: Map $\Phi_p(t)$ from $\Phi_{s,p}$; ▷ eq. 18
4: Map $(\nabla\Phi)_p(t)$ from $(\nabla\Phi)_{s',p}$; ▷ eq. 18
5: Calculate $\Phi_g(t)$ by P2G $\Phi_p(t)$ and $(\nabla\Phi)_p(t)$; ▷ eq. 6
6: Calculate velocity $u_{m,f}$ by wave $\Phi_g(t)$; ▷ eq. 19
7: Calculate u_f by projecting $u_{m,f}$. ▷ eq. 20

Fig. 6. **Eight-Vortex Collision**. In a cubic simulation space with grid sizes of 48/64/128, eight vortices collide, with their initial positions forming the eight vertices of a regular octahedron. After the collision, we accurately simulate the formation of six new vortices, which move along the positive and negative directions of the x-, y-, and z-axes.

4.5 Enhanced Wave Function-Velcity Conversion

Although Equation 12 defines the wave function-velocity relationship, [Chern 2017; Chern et al. 2016a] find direct calculation inaccurate and propose Equation 19 as a more precise method to convert wave function to velocity, based on the following properties [Chern 2017; Chern et al. 2016a]: For any wave function field Φ and any two points $\mathbf{x}_1, \mathbf{x}_2$ close to each other:

$$\hbar \arg\langle\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2)\rangle \approx \int_{\mathbf{x}_1}^{\mathbf{x}_2} \mathbf{u} \cdot d\mathbf{l}, \tag{21}$$

where $\mathbf{l}$ is the straight-line path connecting $\mathbf{x}_1$ and $\mathbf{x}_2$, and $\mathbf{u}$ and Φ satisfies Equation 12. The integral $\int_{\mathbf{x}_1}^{\mathbf{x}_2} \mathbf{u} \cdot d\mathbf{l}$ can be approximated by $u_0 l$, where u_0 is the component of $\mathbf{u}$ along $\overrightarrow{\mathbf{x}_1\mathbf{x}_2}$ at the midpoint of $\mathbf{x}_1$ and $\mathbf{x}_2$, and l is the distance between $\mathbf{x}_1$ and $\mathbf{x}_2$, thus:

$$\hbar \arg\langle\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2)\rangle \approx u_0 l \Rightarrow u_0 = \frac{\hbar}{l} \arg\langle\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2)\rangle. \tag{22}$$

Let $\mathbf{x}_1 = \mathbf{x}_{g_1}$ and $\mathbf{x}_2 = \mathbf{x}_{g_2}$ to obtain Equation 19.

In the above calculation, the accuracy increases as $\mathbf{x}_0$ and $\mathbf{x}_1$ get closer. However, in Equation 19, when converting the wave function to velocity, Φ_g is represented on the grid's cell centers and due to grid limitations, the closest usable points are the adjacent cell centers for a face. Notably, the wave function on the grid, Φ_g, is not the original wave function data, which exists on the particles, including Φ_p and $(\nabla\Phi)_p$. With more particles in the PFM framework than grid centers (typically $|\mathbb{P}|/|\mathbb{G}| \in [8, 16]$ [Zhou et al. 2024]) and first-order gradients, particle data offers a more accurate representation

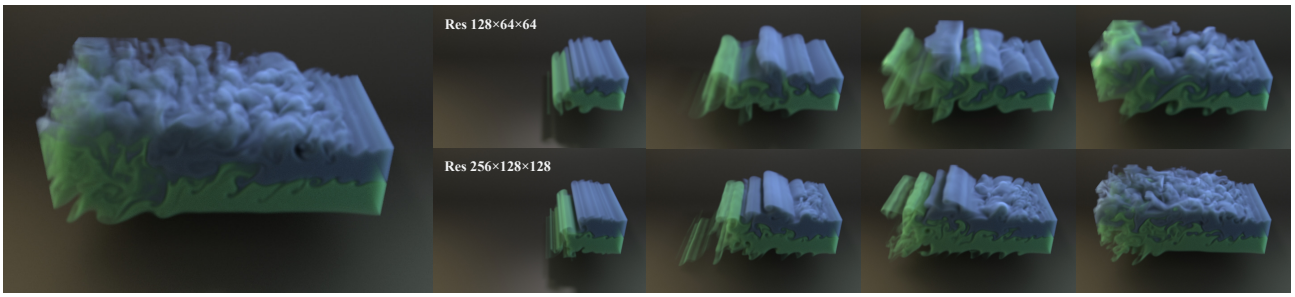


Fig. 8. **KH instability**. When a shear gradient occurs between fluid regions with different velocities, small perturbations grow, forming characteristic vortices. We initialize a four-layer fluid with different velocities across the layers and inject blue and green smoke into the middle two. A fixed-velocity boundary condition with a layered velocity distribution is applied on the right side, while the left side uses an open boundary condition. As shown in the figure above, we successfully captured this phenomenon on grids with resolutions of 128 and 64.

of Φ. Therefore, we can use particle data Φ_p and $(\nabla\Phi)_p$ to obtain values at points closer than adjacent cell centers g_1 and g_2 and apply Equation 20 to obtain velocity.

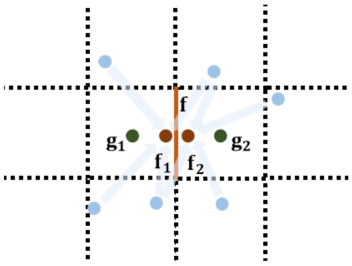


Fig. 7. For the velocity at face center f, we use the wave functions on closer points f_1 and f_2 (dark brown) rather than cell centers g_1 and g_2 (green), obtained by P2G from particles (blue).

Based on this idea, during P2G, we do not directly interpolate the wave function to the grid cell center. Instead, for each face f, we select two points f_1 and f_2 such that f bisects the line connecting f_1 and f_2, with the distance Δx^s between them, where $\Delta x^s < \Delta x$, as shown in Figure 7. Then Φ_{f_1} and Φ_{f_2} are obtained by interpolating the wave function to f_1 and f_2 using Equation 6, and then $u_{m,f}$ are computed as:

$$u_{m,f} = \frac{\hbar}{\Delta x^s} \arg\langle \Phi_{f_1}, \Phi_{f_2} \rangle. \tag{23}$$

This method improves velocity accuracy by using points that are closer together. The experiment in Figure 5 validates that it is more accurate for simulations at lower resolutions.

5 Numerical Method

Some other numerical details are given as follows, and the time integration scheme is provided in Algorithm 3.

Grid-Particle (G2P) Interpolation. The velocity u_f and its gradient at face centers are interpolated onto the particles when evolving the flow map and the same G2P process as in [Zhou et al. 2024] is adopted, i.e., for the k-th component of the velocity $\mathbf{u}$:

$$u_{p,k} = \sum_{f \in \mathbb{N}_{p,k}} u_f w(\mathbf{x}_p, \mathbf{x}_f), \quad \nabla u_{p,k} = \sum_{f \in \mathbb{N}_{p,k}} u_f \nabla w(\mathbf{x}_p, \mathbf{x}_f), \tag{24}$$

where $\mathbb{N}_{p,k}$ represents the set of face centers that are adjacent to particle p and store the k-th velocity component. Similarly, during re-initialization, the wave function and its gradient at the cell centers

Algorithm 3 Wave Function-Based Particle Flow Map

Initialize: initial wave function Φ_g and velocity u_f; $\tilde{\mathcal{T}}_p$ to $\boldsymbol{I}$

for i in total steps **do**
  $j \leftarrow i \pmod{n_V}$;
  $k \leftarrow i \pmod{n_G}$;
  if $j = 0$ **then**
    Set initial time s to now;
    Uniformly distribute fluid particles;
    Normalize and project Φ_g by q; ▷ eq. 26
    Reinitialize $\Phi_{s,p}$ by G2P on Φ_g; ▷ eq. 25
  if $k = 0$ **then**
    Set initial time s' to now;
    Reinitialize $\tilde{\mathcal{T}}_p$ to $\boldsymbol{I}$;
    Reinitialize $(\nabla\Phi)_{s,p}$ by G2P on Φ_g; ▷ eq. 25
  Compute Δt with u_f and the CFL number;
  Estimate midpoint velocity u_f^{mid}; ▷ Alg. 2 in [Zhou et al. 2024]
  Advect $\boldsymbol{x}_p$, $\tilde{\mathcal{T}}_p$ with u_f^{mid} and Δt; ▷ Alg. 3 in [Zhou et al. 2024]
  Calculate mapped wave function Φ_p; ▷ eq. 18
  Calculate mapped gradient of wave function $(\nabla\Phi)_p$; ▷ eq. 18
  Compute Φ_g by P2G with Φ_p and $(\nabla\Phi)_p$; ▷ eq. 6
  Compute $u_{m,f}$ from Φ_g; ▷ eq. 19,23
  Compute $\Gamma_{s\to t}$ by solving Poisson equation and get final velocity u_f by projection. ▷ eq. 20.

are interpolated to particles via the G2P process as:

$$\Phi_p = \sum_{g \in \mathbb{N}_p} \Phi_g w(\mathbf{x}_p, \mathbf{x}_g), \quad (\nabla\Phi)_p = \sum_{g \in \mathbb{N}_p} \Phi_g \nabla w(\mathbf{x}_p, \mathbf{x}_g), \tag{25}$$

where $\mathbb{N}_p$ denotes the set of cell centers adjacent to particle p.

Flow-map Advection. Similar to [Zhou et al. 2024], we use the 4th-order Runge-Kutta method to solve Equation 3 to advect $\mathbf{x}_p(t)$ and $\tilde{\mathcal{T}}_p(t)$. The velocity used to evolve $\mathbf{x}_p(t)$ and $\tilde{\mathcal{T}}_p(t)$ can either be the final velocity $\mathbf{u}_{t'}$ of the previous time step t' or the midpoint velocity $\mathbf{u}_t^{mid}$, which is calculated by Algorithm 2 in [Zhou et al.

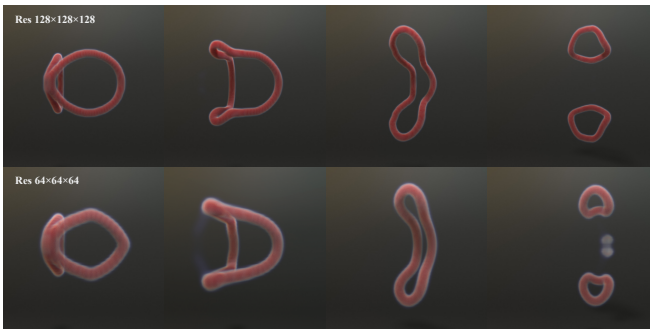


Fig. 9. **Oblique Ring Collision**. Two obliquely positioned vortex rings collide and connect to form a single vortex ring. This vortex is then propelled to the right, undergoing several structural changes before ultimately splitting into two smaller vortex rings. Our method successfully simulates this phenomenon on grids with resolutions of 64 and 128.

2024]. Following [Deng et al. 2023; Zhou et al. 2024], we use the midpoint velocity to evolve $\mathbf{x}_p(t)$ and $\tilde{\mathcal{T}}_p(t)$.

Re-initialization. According to [Zhou et al. 2024], an excessive flow map length leads to distortion and inaccuracies, requiring periodic reinitialization by resetting the initial time to the current time. The flow map length is defined by the number of time steps between two reinitializations. Gradient calculations require shorter flow maps, so we alternate between two initial times: s for the wave function Φ and s' for its gradient $\nabla\Phi$. Every n_G steps, we reset s' to the current time, set $\tilde{\mathcal{T}}_p(s') = I$, and use the G2P process to compute $(\nabla\Phi)_{s',p}$. Every n_V steps ($n_V \mod n_G = 0$), we reset s to the current time, uniformly distribute fluid particles, normalize $\Phi \leftarrow \frac{\Phi}{|\Phi|}$, and project $\Phi \leftarrow \Phi e^{-iq/\hbar}$ for standardization, where q is calculated by solving Poisson equation:

$$\Delta q = \nabla \cdot \mathbf{u}^*, \quad \mathbf{u}^* = \hbar\langle\nabla\Phi, i\Phi\rangle_{\mathbb{R}}. \tag{26}$$

Finally, $\Phi_{s,p}$ is calculated by G2P process.

Boundary conditions. In the simulation, we consider three types of boundary conditions: source, solid boundary and open boundary. For the sources with constant velocity $\mathbf{v}$, we adopt the same method as [Chern et al. 2016a; Yang et al. 2021], where the value of the wave functions in the source region is enforced by assuming a constant velocity within that region, namely $\Psi(\mathbf{x}, t) = (e^{i\langle \mathbf{v}/\hbar, \mathbf{x}-\mathbf{v}t/2\rangle}, \epsilon)$, where $\epsilon = 0.01$ merely to guard against zeros in Ψ_1 during normalization [Chern et al. 2016a]. For the solid boundary, the wave function must be extrapolated into the solid region, as its values may be needed during the G2P process. The extrapolation process follows [Xiong et al. 2022], where for the cell center point g near the boundary in the solid, the wave function is given by

$$\Psi_g = \frac{\sum_{g' \in \mathbb{G}_b} \Psi_{g'} e^{iu_{gg'}\Delta x/\hbar}}{|\sum_{g' \in \mathbb{G}_b} \Psi_{g'} e^{iu_{gg'}\Delta x/\hbar}|}, \tag{27}$$

where $\mathbb{G}_b$ denotes the set of cell center points in the fluid that are adjacent to g, and $u_{gg'}$ represents the solid velocity component at the face-centered point between g and g'. For the open boundary, no special treatment is required; it can be handled in the same way as in velocity-based solvers.

Blending. We found that, in the presence of solids, our method encounters the same issue as [Yang et al. 2021], where advection tends to be stuck behind the solids, possibly due to inaccuracies in the current extrapolation scheme for solid boundary in Equation 27. To enhance the convective flow near the solid boundaries, we blend the computed velocity $\mathbf{u}_m(\mathbf{x}, t)$ with the velocity obtained from the semi-Lagrangian method $\mathbf{u}^A(\mathbf{x}, t)$, i.e., $\tilde{\mathbf{u}}_m = \beta\mathbf{u}^A + (1-\beta)\mathbf{u}_m$, where β is a constant that controls the blending, and replace the original $\mathbf{u}_m$ with the fused velocity $\tilde{\mathbf{u}}_m$.

$\hbar$ Parameters. According to [Chern et al. 2016a; Yang et al. 2021], $\hbar$ determines the strength and scale of vortex filaments that the simulation tends to concentrate on, and it needs to be adjusted based on the grid resolution and initial conditions. In our experiments, we choose values between 0.15 and 1.5 depending on the grid resolution, with detailed settings provided in Table 2.

6 Results and Discussion

Our implementation is built upon Taichi [Hu et al. 2019], with computations performed in double-precision (64-bit) floating point, and all experiments are conducted on a GeForce RTX 4090. The specific details of the settings, such as grid resolutions, parameter choices, etc., as well as timing and other statistical data, are provided in Table 2.

Validation. Using the Clebsch wave function representation, we achieved precise convection through the particle flow map method, which enables vortex preservation even on low-resolution coarse grids. Compared to the previous state-of-the-art impulse-based method (the impulse-based PFM method [Zhou et al. 2024]), the state-of-the-art Clebsch wave function-based method (the Clebsch Gauge Fluid method [Yang et al. 2021]), as well as the classical method (the semi-Lagrangian-based advection-projection method [Stam 1999]), our approach exhibits superior vortex preservation, even on extremely coarse grids. We conduct the following comparative experiments on grids with resolutions below 64 (in our experiments, a resolution of n refers to a grid of size $n \times n \times 2n$ unless otherwise specified): (1) **Leapfrogging Vortices**: We initialize two concentric vortex rings with the same rotation direction and the same radius $r = 0.21$, with their centers separated by a distance $d = 0.47r$. Our approach is able to sustain 6 leaps (see details in Figure 12) on a coarse grid with resolution 64, which clearly outperforms other methods, while incurring no significant time overhead, as shown in Table 2. (2) **Trefoil Vortex Rings**: We simulate trefoil knot vortex. Details and comparison with Clebsch gauge fluid are shown in Figure 14. (3) **Hopf-Linked Vortex Rings**: We initialized two obliquely linked vortex rings, also known as Hopf-linked vortex rings and gave similar evolutions shown in [Yao et al. 2022] (refer to Figure 13 for details and comparison with Clebsch gauge Method). (4) **Four Linked Vortex Rings**: We simulate four obliquely linked vortex rings with comparison performed against Clebsch gauge fluid method (experiment details are shown in Figure 15).

Fig. 10. **Smoke Plume Passing the Bunny**. We simulate a smoke plume passing a solid bunny model on a 128-resolution grid. Numerous vortices are shedded, generating highly turbulent smoke. This case demonstrates our method's capability to handle solid boundaries correctly.

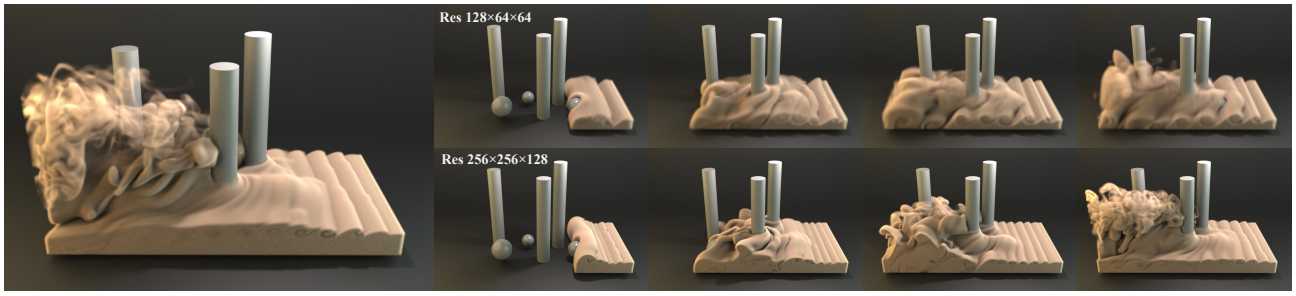


Fig. 11. **Smoke Passing Multiple Obstacles**. A more complex scenario involving solids, consisting of three spherical and three cylindrical obstacles is simulated. A smoke layer is emitted near the ground, and due to variations in flow velocity, KH instability can also be observed. As the smoke passes multiple obstacles, the vortices shed from the solids make the smoke highly turbulent.

Ablation Study. Here, we demonstrate with two experiments that (1) our Enhanced Wave Function-Velocity Conversion improves the accuracy of wave function-based PFM calculations, and (2) we show that advection of the gradient in PFM is essential for accurate advection. Figure 5 details the results from our ablation study.

Example. We present additional examples to demonstrate the robustness and correctness of our method. (1) **Four-Vortex Collision**: In Figure 4, we simulate the collision of four vortices, which collide and reconnect to two-star shaped vortices. Simulation resolutions range from 48 to 128. (2) **Eight-Vortex Collision**: A more complex eight vortex collision scenario is shown in Figure 6. (3) **Oblique Ring Collision**: We simulate oblique vortex collision with different resolutions. We refer readers to Figure 9 for details. (4) **Smoke Plume Passing the Bunny**: In Figure 10, we simulate a smoke plume passing around a bunny-shaped obstacle, which sheds numerous vortices, making the plume highly turbulent. (5) **Kelvin–Helmholtz (KH) Instability**: As shown in Figure 8, we simulate KH instability caused by interactions between four fluid layers with different velocities at resolutions of 128 and 64, generating complex vortices at the layer interfaces. (6) **Smoke Passing Multiple Obstacles**: Figure 11 shows a more complex scene with multiple obstacles, simultaneously capturing vortex shedding and KH instability. (7) **Moving Paddle**: In Figure 3, to demonstrate our method's ability to handle moving solids, we simulate a rotating paddle interacting with smoke pillars, producing a turbulent smoke mixture.

Timing Analysis. As shown in the Leapfrog timing comparison in Table 2, our method introduces only a modest increase in time cost compared to the impulse-based PFM. The PFM approach primarily consists of two stages: advection and projection. Since our projection step is identical to that of the impulse-based PFM, its time cost remains similar (0.14 sec for ours and 0.15 for impulse-based PFM). The additional cost stems from our more complex P2G process for wave function, taking 0.28 sec versus 0.17 sec in the impulse-based method.

Rendering Details. We render all results using Houdini FX 20.5.278 with a voxel-based pipeline. Vortex grids are visualized by mapping their values directly to the material's density in Houdini. For smoke rendering in Figure 13, Figure 14, and Figure 15, we use particle-based tracking and convert particles to a voxel density field via P2G during rendering—an approach that better preserves detail at low resolutions. In other cases, smoke is tracked as a grid-based density field and rendered directly through the material's density channel.

Discussion. In our comparison, we exclude [Chern et al. 2016a] due to its distinct approach. Like other flow map methods [Deng

Fig. 12. **Leapfrogging Vortices**. In a $128 \times 64 \times 64$ resolution, our method successfully maintains separation of the rings through six leaps and continues to sustain them for three leaps even when using a lower resolution $64 \times 32 \times 32$ grid. In comparison, the impulse-based PFM results in merging after just one leap, while the Clebsch gauge method begins to lose vorticity following a single leap.

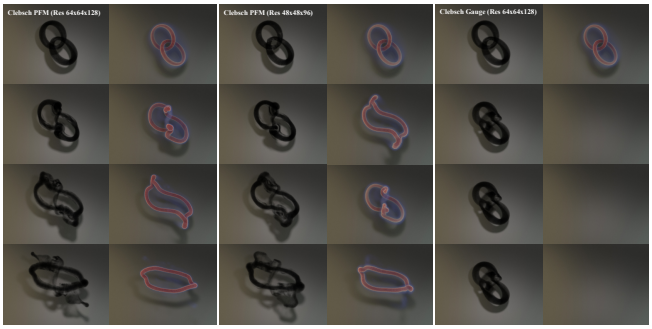


Fig. 13. **Hopf-Linked Vortex Rings**. Due to self-induction, the initially perpendicular rings move closer, deforming to anti-parallel and then reconnect, merging into a larger vortex ring [Yao et al. 2022]. Our method accurately reproduces this phenomenon on both 64- and 48-resolution grids.

et al. 2023; Zhou et al. 2024], we focus on accurately computing the advection term via flow maps to improve advection handling and vortex preservation. While we use the wave function, our equation remains fully equivalent to the Euler equations. In contrast, [Chern et al. 2016a] bypasses advection computation by introducing extra terms, solving a different equation that is not equivalent to the Euler equations. However, we include [Yang et al. 2021] as its equation remains Euler equations-equivalent and is recognized for strong vortex preservation in the literature.

7 Limitations and Future Work

Similar to the method in [Yang et al. 2021], our current Clebsch wave function-based solver relies on blending with the advected velocity computed using the semi-Lagrangian method to introduce numerical viscosity for scenarios involving solids. In the future, we hope to devise a method to add viscosity to the wave function-based PFM solver, similar to the approach in [Li et al. 2024b] for adding viscosity to the impulse-based PFM. Additionally, due to the complexity of the velocity-to-wave function conversion, our solver currently cannot initialize particularly complex scenarios, such as vorticity fields that cannot be represented using vortex filaments. In the future, we expect to use the velocity-to-wave function conversion method proposed in [Chern 2017] to enable more complex initializations.

Table 2. The catalog of all our simulation examples.

Name	Figure	Resolution[1]	$\hbar$	β	Time (sec /substep)[2]	Memory Cost (GB)[3]
Moving Paddle	Figure 3	128	1.0	0.995	1.91	12.69
Four-Vortex Collision	Figure 4	48/64/128	0.1	N/A	0.22/0.38/2.43	2.47/3.79/19.74
Eight-Vortex Collision	Figure 6	48/64/128[4]	0.075	N/A	0.18/0.27/1.25	2.00/2.69/10.88
KH instability	Figure 8	64/128	0.75	0.95	0.73/3.53	2.91/12.69
Oblique Ring Collision	Figure 9	64/128[5]	8	N/A	0.23/2.34	2.69/10.88
Smoke Plume Passing the Bunny	Figure 10	128	1.0	0.995	2.67	12.69
Smoke Passing Multiple Obstacles	Figure 11	64/128	0.5	0.95	0.92/3.78	2.91/12.69
Hopf-Linked Vortex Rings	Figure 13	48/64	0.5	N/A	0.23/0.38	2.47/3.79
Trefoil Knot	Figure 14	48/64[6]	0.5	N/A	0.14/0.23	2.00/2.69
Four Linked Vortex Rings	Figure 15	48/64	0.5	N/A	0.24/0.28	2.47/3.79
Leapfrogging Vortices (Ours)	Figure 12 (2nd row)	64	0.25	N/A	0.42	3.79
Leapfrogging Vortices (Ours, Low Resolution)	Figure 12 (1st row)	32	1.0	N/A	0.13	1.91
Leapfrogging Vortices (Impulse-based PFM)	Figure 12 (3th row)	64	N/A	N/A	0.32	4.10
Leapfrogging Vortices (Clebsch Gauge)	Figure 12 (4th row)	64	0.25	N/A	0.11	1.91
Leapfrogging Vortices (Advectoin-Projection)	Figure 12 (5th row)	64	N/A	N/A	0.09	11.55

[1] Unless otherwise specified, a resolution of n refers to a grid size of $n \times n \times 2n$; for example, 64 indicates a grid of size $64 \times 64 \times 128$. If multiple values are listed separated by "/", it means the experiment was conducted at multiple resolutions.

[2] [3] For experiments conducted at multiple resolutions, we report the time and memory cost for each resolution, listed in order and separated by slashes "/" to correspond to the respective resolutions.

[4] [5] [6] The Eight-Vortex Collision experiment, Oblique Ring Collision experiment and Trefoil Knot experiment are conducted on a grid of size $n \times n \times n$.

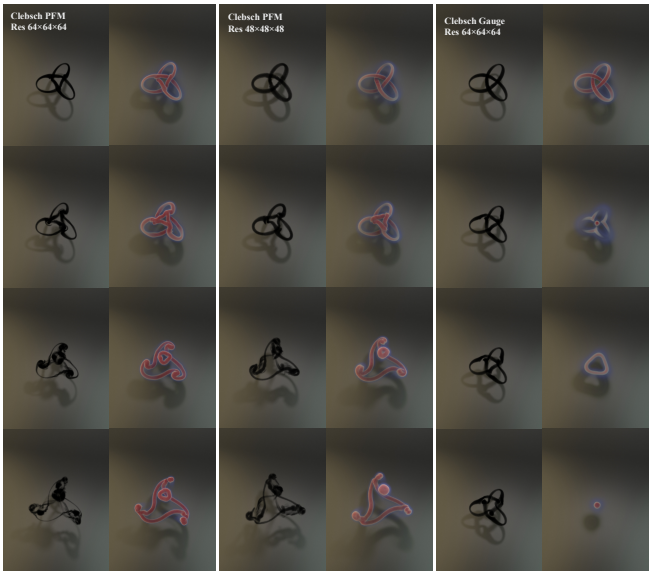


Fig. 14. **Trefoil Knot**. We initialize a trefoil knot vortex filament as in [Kim et al. 2009]. The vortex filament twists, reconnects and decomposes into two vortex rings. Our simulation on 64/48-resolution grids matchs experiments in [Kleckner and Irvine 2013] while Clebsch gauge fluid method fails

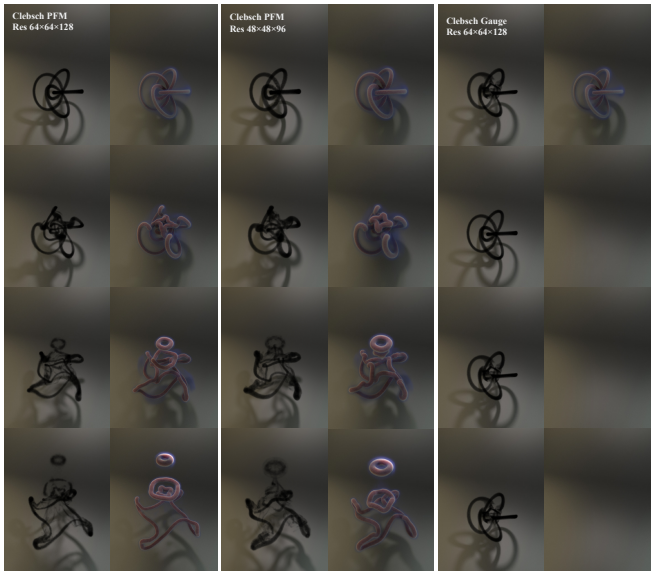


Fig. 15. **Four Linked Vortex Rings**. Four obliquely linked vortex rings evolve, with our method accurately simulating their collision, reconnection, and the formation of four independent rings. Despite one ring disappearing on a 48-sized grid, the other three persist—unachievable by the Clebsch gauge method.

Acknowledgments

We thank the anonymous reviewers for their constructive comments. Georgia Tech authors acknowledge NSF IIS #2433322, ECCS #2318814, CAREER #2420319, IIS #2433307, OISE #2433313, and CNS #1919647 for funding support.

References

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

Duowen Chen, Zhiqi Li, Junwei Zhou, Fan Feng, Tao Du, and Bo Zhu. 2024. Solid-Fluid Interaction on Particle Flow Maps. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–20.

Albert Chern. 2017. *Fluid dynamics with incompressible Schrödinger flow.* California Institute of Technology.

Albert Chern, Felix Knöppel, Ulrich Pinkall, and Peter Schröder. 2017. Inside fluids: Clebsch maps for visualization and processing. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–11.

Albert Chern, Felix Knöppel, Ulrich Pinkall, Peter Schröder, and Steffen Weißmann. 2016a. Schrödinger's smoke. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–13.

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

A. Clebsch. 1859. Ueber die Integration der hydrodynamischen Gleichungen. *Journal für die reine und angewandte Mathematik* 1859, 56 (1859), 1–10. https://doi.org/doi:10.1515/crll.1859.56.1

Ricardo Cortez. 1995. *Impulse-based particle methods for fluid flow.* University of California, Berkeley.

Yitong Deng, Hong-Xing Yu, Diyang Zhang, Jiajun Wu, and Bo Zhu. 2023. Fluid Simulation on Neural Flow Maps. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–21.

Yun Fei, Qi Guo, Rundong Wu, Li Huang, and Ming Gao. 2021. Revisiting integration in the material point method: a scheme for easier separation and less dissipation. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–16.

Toshiya Hachisuka. 2005. Combined Lagrangian-Eulerian approach for accurate advection. In *ACM SIGGRAPH 2005 Posters*. 114–es.

Francis H Harlow. 1962. *The particle-in-cell method for numerical solution of problems in fluid dynamics.* Technical Report. Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

Yuanming Hu, Tzu-Mao Li, Luke Anderson, Jonathan Ragan-Kelley, and Frédo Durand. 2019. Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 201.

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.

Doyub Kim, Oh-young Song, and Hyeong-Seok Ko. 2009. Stretching and wiggling liquids. In *ACM SIGGRAPH Asia 2009 papers*. 1–7.

Dustin Kleckner and William TM Irvine. 2013. Creation and dynamics of knotted vortices. *Nature physics* 9, 4 (2013), 253–258.

Xingqiao Li, Xingyu Ni, Bo Zhu, Bin Wang, and Baoquan Chen. 2023. GARM-LS: A Gradient-Augmented Reference-Map Method for Level-Set Fluid Simulation. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–20.

Zhiqi Li, Barnabás Börcsök, Duowen Chen, Yutong Sun, Bo Zhu, and Greg Turk. 2024a. Lagrangian Covector Fluid with Free Surface. In *ACM SIGGRAPH 2024 (Conference Track)*.

Zhiqi Li, Duowen Chen, Candong Lin, Jinyuan Liu, and Bo Zhu. 2024b. Particle-Laden Fluid on Flow Maps. *arXiv preprint arXiv:2409.06246* (2024).

Takumi Matsuzawa, Noah P Mitchell, Stéphane Perrard, and William TM Irvine. 2022. Video: Turbulence through sustained vortex ring collisions. In *75th Annual Meeting of the APS Division of Fluid Dynamics-Gallery of Fluid Motion (2022). https://api.semanticscholar. org/CorpusID*, Vol. 252974545.

Mohammad Sina Nabizadeh, Stephanie Wang, Ravi Ramamoorthi, and Albert Chern. 2022. Covector fluids. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–16.

Ziyin Qu, Minchen Li, Fernando De Goes, and Chenfanfu Jiang. 2022. The power particle-in-cell method. *ACM Transactions on Graphics* 41, 4 (2022).

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.

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 (2018), 6.

Robert Saye. 2016. Interfacial gauge methods for incompressible fluid dynamics. *Science advances* 2, 6 (2016), e1501869.

Jos Stam. 1999. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. 121–128.

Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. 2013. A material point method for snow simulation. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–10.

Yuchen Sun, Linglai Chen, Weiyuan Zeng, Tao Du, Shiying Xiong, and Bo Zhu. 2024. An Impulse Ghost Fluid Method for Simulating Two-Phase Flows. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–12.

Yuchen Sun, Xingyu Ni, Bo Zhu, Bin Wang, and Baoquan Chen. 2021. A material point method for nonlinearly magnetized materials. *ACM Transactions on Graphics (TOG)* 40, 6 (2021), 1–13.

Rui Tao, Hongxiang Ren, Yunjin Tong, and Shiying Xiong. 2021. Construction and evolution of knotted vortex tubes in incompressible Schrödinger flow. *Physics of Fluids* 33, 7 (2021).

Sinan Wang, Yitong Deng, Molin Deng, Hong-Xing Yu, Junwei Zhou, Duowen Chen, Taku Komura, Jiajun Wu, and Bo Zhu. 2024. An Eulerian Vortex Method on Flow Maps. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–13.

DC Wiggert and EB Wylie. 1976. Numerical predictions of two-dimensional transient groundwater flow by the method of characteristics. *Water Resources Research* 12, 5 (1976), 971–977.

Jie-Zhi Wu, Hui-Yang Ma, and M-D Zhou. 2007. *Vorticity and vortex dynamics.* Springer Science & Business Media.

Shiying Xiong, Zhecheng Wang, Mengdi Wang, and Bo Zhu. 2022. A clebsch method for free-surface vortical flow simulation. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–13.

Shuqi Yang, Shiying Xiong, Yaorui Zhang, Fan Feng, Jinyuan Liu, and Bo Zhu. 2021. Clebsch Gauge Fluid. *ACM Transactions on Graphics (TOG)* 40, 4 (2021).

Jie Yao, Weiyu Shen, Yue Yang, and Fazle Hussain. 2022. Helicity dynamics in viscous vortex links. *Journal of Fluid Mechanics* 944 (2022), A41.

Hang Yin, Mohammad Sina Nabizadeh, Baichuan Wu, Stephanie Wang, and Albert Chern. 2023. Fluid Cohomology. *ACM Trans. Graph.* 42, 4, Article 126 (July 2023), 25 pages. https://doi.org/10.1145/3592402

Yonghao Yue, Breannan Smith, Christopher Batty, Changxi Zheng, and Eitan Grinspun. 2015. Continuum foam: A material point method for shear-dependent flows. *ACM Transactions on Graphics (TOG)* 34, 5 (2015), 1–20.

Junwei Zhou, Duowen Chen, Molin Deng, Yitong Deng, Yuchen Sun, Sinan Wang, Shiying Xiong, and Bo Zhu. 2024. Eulerian-Lagrangian Fluid Simulation on Particle Flow Maps. *ACM Transactions on Graphics (SIGGRAPH 2024)* (2024).

Yongning Zhu and Robert Bridson. 2005. Animating sand as a fluid. *ACM Transactions on Graphics (TOG)* 24, 3 (2005), 965–972.

A Proof of Equation 16

First, we verify the evolution equation of Φ in Equation 16. Let $\Phi = (\Phi_1, \Phi_2)$, $\Psi = (\Psi_1, \Psi_2)$, where $\Phi_k(\mathbf{x}, t), \Psi_k(\mathbf{x}, t) \in C$ for $k = 1, 2$. Then, for $\frac{D\Phi}{Dt} = \left(\frac{D\Phi_1}{Dt}, \frac{D\Phi_2}{Dt}\right)$, considering the k-th component of Φ ($k = 1, 2$), we have

$$
\begin{aligned}
\frac{D\Phi_k}{Dt} &= \frac{D(\Psi_k e^{i\Gamma_{s\to t}/\hbar})}{Dt} \\
&= \Psi_k \frac{De^{i\Gamma_{s\to t}/\hbar}}{Dt} + \frac{D\Psi_k}{Dt} e^{i\Gamma_{s\to t}/\hbar} \\
&= \frac{i}{\hbar}\Psi_k e^{i\Gamma_{s\to t}/\hbar} \frac{D\Gamma_{s\to t}}{Dt} + \frac{D\Psi_k}{Dt} e^{i\Gamma_{s\to t}/\hbar} \overset{①}{\Rightarrow} \\
&= e^{i\Gamma_{s\to t}/\hbar}\left(\frac{i}{\hbar}\Psi_k\left(\frac{p}{\rho} - \frac{|\mathbf{u}|^2}{2}\right) - \frac{i}{\hbar}\Psi_k\left(\frac{p}{\rho} - \frac{|\mathbf{u}|^2}{2}\right)\right) \\
&= 0,
\end{aligned}
\tag{28}
$$

where ① is obtained by substituting $\frac{D\Psi_k}{Dt}$ from Equation 15 and incorporating the definition of $\Gamma_{s\to t}$.

Since the operation $\hbar < \nabla\Psi, i\Psi >_{\mathbb{R}}$ satisfies the property [Chern 2017]: for any scalar field q and wave function Ψ, we have $\hbar < \nabla(\Psi e^{iq/\hbar}), i\Psi e^{iq/\hbar} >_{\mathbb{R}} = \hbar < \nabla\Psi, i\Psi >_{\mathbb{R}} + \nabla q$. Thus, we obtain:

$$
\begin{aligned}
\mathbf{u}_m &= \hbar < \nabla(\Psi e^{i\Gamma_{s\to t}/\hbar}), i(\Psi e^{i\Gamma_{s\to t}/\hbar}) >_{\mathbb{R}} \\
&= \hbar < \nabla\Psi, i\Psi >_{\mathbb{R}} + \Gamma_{s\to t} \\
&= \mathbf{u} + \nabla\Gamma_{s\to t}.
\end{aligned}
\tag{29}
$$

Since $\nabla \cdot \mathbf{u} = 0$, it follows that $\Gamma_{s\to t}$ satisfies $\Delta\Gamma_{s\to t} = \nabla \cdot \mathbf{u}_m$.