Incompressible Flow Simulation on Vortex Segment Clouds

SHIYING XIONG, Dartmouth College
RUI TAO, Dartmouth College and Dalian Maritime University
YAORUI ZHANG, Dartmouth College
FAN FENG, Dartmouth College
BO ZHU, Dartmouth College

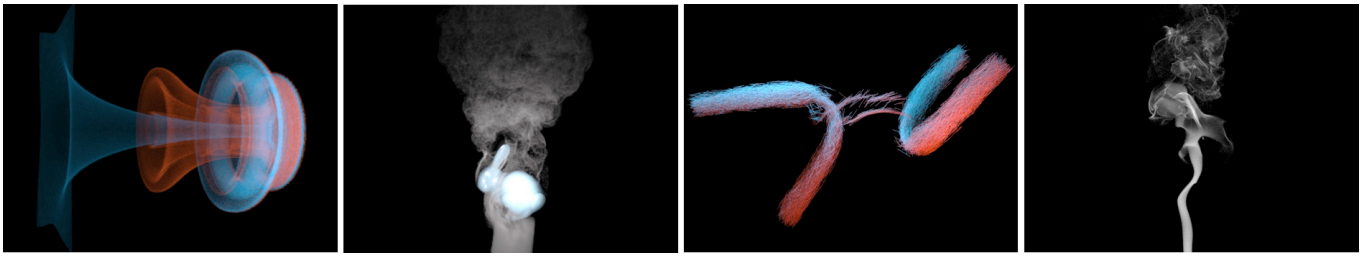

Fig. 1. Various fluid phenomena simulated using our vortex segment method. (Far Left) Leapfrogging vortices. (Middle Left) Turbulent smoke flowing past a rotating bunny. (Middle Right) Reconnected vortex tubes from two intersecting ones. (Far Right) Cigarette smoke.

We propose a novel Lagrangian geometric representation using segment clouds to simulate incompressible fluid exhibiting strong anisotropic vortical features. The central component of our approach is a cloud of discrete segments enhanced by a set of local segment reseeding operations to facilitate both the geometrical evolution and the topological updates of vortical flow. We build a vortex dynamics solver with the support for dynamic solid boundaries based on discrete segment primitives. We demonstrate the efficacy of our approach by simulating a broad range of challenging flow phenomena, such as reconnection of non-closed vortex tubes and vortex shedding behind a rotating object.

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

Additional Key Words and Phrases: Lagrangian fluid simulation, vortex method, segment cloud, vortex reconnection

ACM Reference Format:
Shiying Xiong, Rui Tao, Yaorui Zhang, Fan Feng, and Bo Zhu. 2021. Incompressible Flow Simulation on Vortex Segment Clouds. *ACM Trans. Graph.* 40, 4, Article 98 (August 2021), 12 pages. https://doi.org/10.1145/3450626.3459865

1 INTRODUCTION

What is the most effective discrete geometry representation for vortex dynamics? Researchers in computer graphics and computational physics have sought to answer this question for decades. Various types of discrete data structures and numerical solvers have been proposed toward building robust and accurate computational tools to simulate fluid flow that exhibits strongly anisotropic features and dynamic topological changes (e.g., see [Brochu et al. 2012; Padilla et al. 2019; Weißmann and Pinkall 2010]). A particular example of these visually appealing flow phenomena is vortex tube dynamics [Chern et al. 2017, 2016; Xiong and Yang 2019a]. With strong vorticities concentrating around a narrow region of codimension-2 geometries, the evolution of a vortex tube exhibits a wide scope of flow behaviors, such as stretching [Beardsell et al. 2016a], splitting [Kleckner et al. 2016; Scheeler et al. 2014], reconnection [Kida and Takaoka 1994], and breakdown [Leibovich 1978].

Discrete particles and simplicial meshes are the two most widely recognized geometric representations to simulate vortex dynamics, which can be translated to the categories of vortex particle methods [Cottet and Koumoutsakos 2000] and vortex filament methods [Weißmann and Pinkall 2010]. Without dismissing their wide success and broad impact in visual and scientific fluid simulation, these two approaches might have potential weaknesses. Particle methods are difficult to express local orientation, which makes it challenging to track flows that are particularly thin and anisotropic (typically requiring a prohibitively large number of particles) [Ando et al. 2012]. Mesh methods might either suffer from tedious mesh repair operations, or a prominent increase in the number of elements needed to resolve the increasing complexity of the flow. Hybrid methods such as particle-mesh [van Rees et al. 2012], PIC/FLIP [Ferstl et al. 2016; Zhu and Bridson 2005] and PPPM [Zhang and Bridson 2014], along with many other numerical infrastructures developed in computational physics [Hu et al. 2019], can help alleviate problems with either tracking accuracy or computational efficiency. However, none of these approaches provides a unified and simplified numerical solution to track and evolve temporally vortex structures exhibiting strong *anisotropic* behaviors.

Authors' addresses: Shiying Xiong, Computer Science Department, Dartmouth College, Hanover, NH, USA, shiying.xiong@dartmouth.edu; Rui Tao, Dartmouth College, Computer Science Department, Dalian Maritime University, Dalian, Liaoning, China; Yaorui Zhang, Dartmouth College; Fan Feng, Dartmouth College; Bo Zhu, Dartmouth College.

0730-0301/2021/8-ART98 $15.00
https://doi.org/10.1145/3450626.3459865

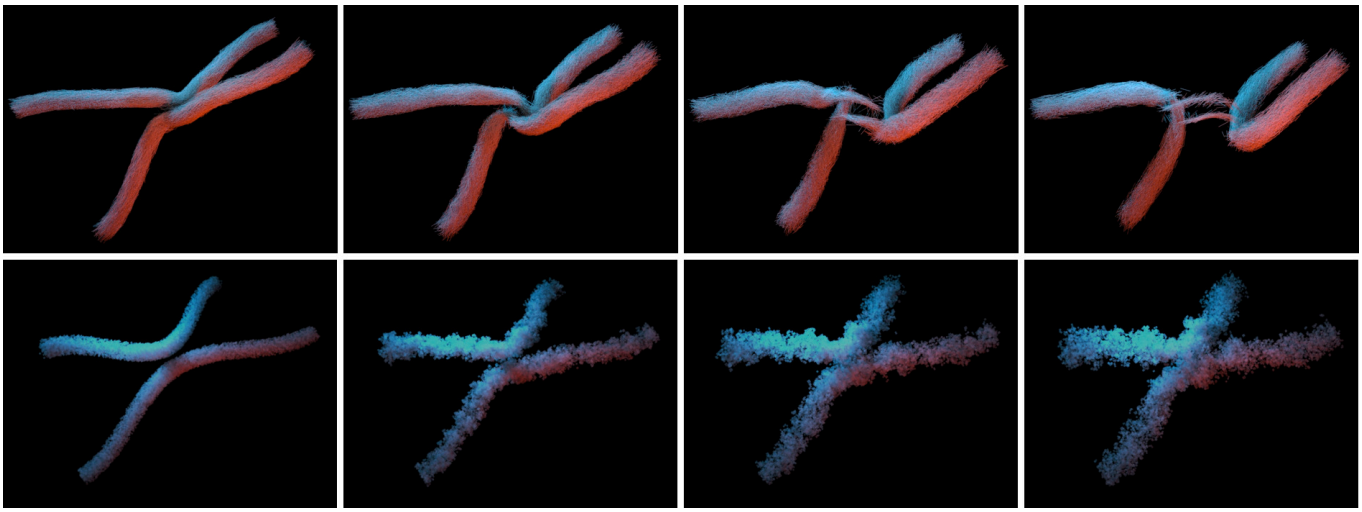

Fig. 2. Comparison of the splitting and reconnection of intersecting vortex tubes with the vortex segment method and the vortex particle method. Top/bottom 4 pictures show frames with vortex segment/particle method at 1, 100, 200 and 300, respectively.

Back in 1990, Chorin described in his pioneering work [Chorin 1990] that *"a physical vortex is approximated by a cloud of tubular vortices."* The vortex segment method he proposed in this work, in which vorticity is carried on a set of segments and evolved by calculating their interactions, was the predecessor of the modern vortex particle method (e.g., [Cottet and Koumoutsakos 2000]). Following this work, Chorin [1993] switched the data representation from discrete segments to a segment mesh, to reduce the redundant vertex storage and hence improve the computation efficiency, which laid the foundation of the modern vortex filament method [Weißmann and Pinkall 2010]. These two pieces of classical work yield an insightful mathematical model that *"an incompressible flow can be approximated by a 'polymeric' model, which consists of an ensemble of stretched, folded, and pinched vortex tubes"* [Chorin 1990], which serves as the motivation for our numerical paradigm design.

Motivated by Chorin's work, we devise a structure-enriched and connectivity-free Lagrangian method to model vortical flow featured by its anisotropic geometry and dynamics. Specifically, we build a generalized particle representation based on segment clouds with each particle consisting of two-point samples. From a computational perspective, discrete segments, or a generalized Lagrangian representation with each particle carrying two-point samples possess a series of inherent computational advantages when modeling anisotropic vortical flows. Numerical merits include ease of modeling local, vortical stretching [Zhang and Bridson 2014], enforcing adaptivity [Fernandez et al. 1996], and to handling topological changes robustly [Weißmann and Pinkall 2009].

To accommodate the various types of anisotropic geometrical and topological evolution on a segment cloud, we build a set of discrete reseeding operations enhanced by each segment's orientation. These reseeding operations consist of merging, splitting, and deleting, which are combined to mimic the conventional particle reseeding procedure. All of these operations leverage the anisotropic and oriented features of the segment primitives. Moreover, these operations are local, parallelizable, and connectivity-free, facilitating a high-performance code implementation to leverage the modern parallel computer architectural intricacies. As demonstrated in our examples, the proposed method accommodates the parallel computation of large-scale vortex phenomena on modern computing hardware, which boosts the capability of the method in solving strongly anisotropic and topologically complicated flows.

On top of the segment cloud discretization, we build a discrete vortex dynamics solver with the support for dynamic solid boundaries based on discrete segment primitives , to enable the simulation of various challenging flow phenomena that are difficult for the previous Lagrangian methods such as reconnection of non-closed vortex tubes and vortex shedding behind a rotating object. We summarize our technical contributions as follows:

- an isotropic Lagrangian geometric representation built on discrete segment clouds to model a broad range of anisotropic flow phenomena;
- a parallelizable segment processing algorithm to facilitate complicated topological changes of vortical flow;
- a vortex dynamics solver based on segment clouds to simulate complex vortical flow phenomena.

2 RELATED WORK

Vortex particle method. Although early works adopted point vortices to numerically simulate the dynamical evolution of 2-D inviscid flow in an unbounded domain [Rosenhead 1931; Takami 1964], the modern vortex method is marked by the vortex blob method proposed by Chorin [1973], which removes the singularity in the kernel function by replacing the point vortex with certain vortex cores. The vortex blobs might be of various shapes, such as an isotropic sphere or a small vortex sheet [Pfaff et al. 2012a]. Various options exist for the vorticity distribution in vortex blob methods, such as the Gaussian distribution [Park and Kim 2005], the Rankine vortex

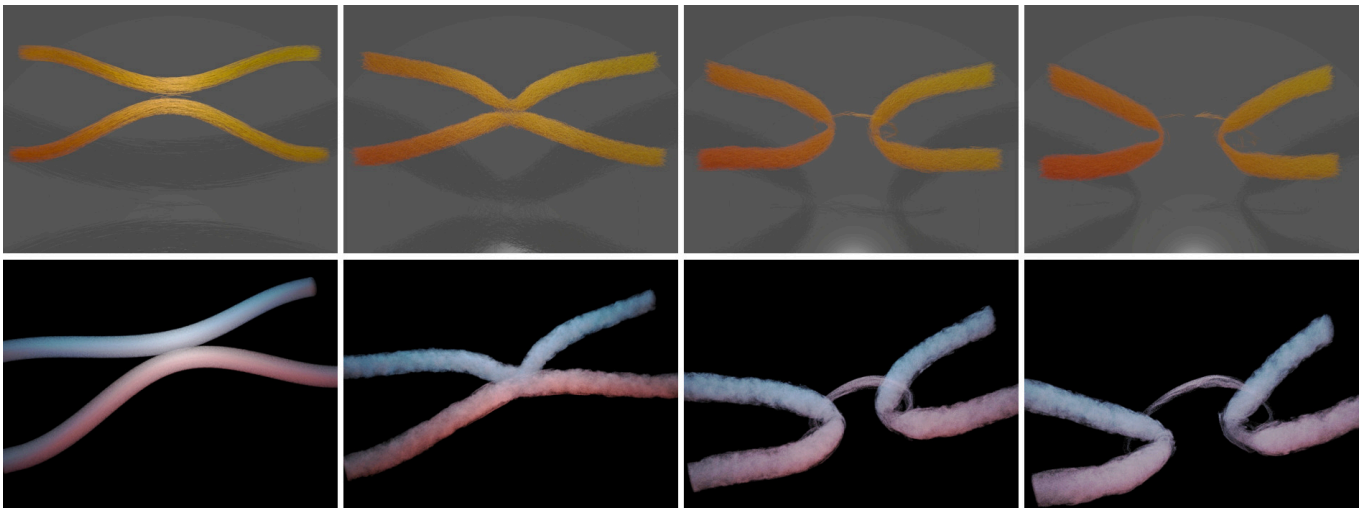

Fig. 3. The splitting and reconnection of quasi-parallel vortex tubes. Left to right columns: frames 1, 100, 300, and 400. The pictures are visualized by vortex segments clouds (top 4 pictures) and tracer particles (bottom 4 pictures).

model [Loiseleux et al. 1998], and the Krasny model [Krasny 1988], etc. There are possible limitations of using the vortex blob method to solve large-scale complex vortical flow. For long-term computational accuracy, a vortex blob method requires that each vortex element overlap its neighboring blobs, which consumes a massive number of vortex elements for computational stability [Hald 1979; Hald and Del Prete 1978]. Besides, the shape of every single blob is different from the common filamentous or tubular structures in the flow field, making it challenging for vortex methods to form coherent structures under a turbulent setting [She et al. 1990]. Finally, a vortex blob method updates the vorticity stretching term with the original, transposed [Choquin and Huberson 1990], or symmetrical [Cottet and Koumoutsakos 2000] form by taking the derivative of the kernel function. The correctness of a numerical stretching relies on the distribution of vortex elements together with the choice of a proper kernel function to ensure numerical precision and adaptability [Angelidis 2017]. In the absence of ambiguity, we refer to the vortex particles/points below as vortex blobs.

Vortex filament method. Vortex filaments are important for 3D turbulence dynamics [Xiong and Yang 2017, 2019b], providing one of the most efficient numerical methods to reproduce the complexity of smoke with sparse discrete primitives [Eberhardt et al. 2017; Weißmann et al. 2014]. The numerical simulation of vortex filaments can be traced back to Hasimoto's study on the local induction approximation (LIA) of some isolated vortex filaments, which validates that a single vortex filament in LIA fits well with the experimental results of the propagation of isolated waves on a twisted structure [Aref and Flinchem 1985; Hasimoto 1972; Hopfinger et al. 1982]. For the first time, Angelidis and Neyret [2005] simulated the flow field with a large number of closed vortex filaments. From a Hamiltonian perspective, Weißmann and Pinkall [2009] proposed a physically conservative model that compensates for the discretization errors inherent to the polygonal vortex filament model. Barnat and Pollard [2012] developed a new set of reconnection criteria to simulate smoke with a filament graph. Then, Padilla et al. [2019] simulated elaborate physical phenomena with the thickness of vortex filaments taken into account, such as the dynamic evolution of an ink drop. A potential limitation of vortex filament methods is the need for tedious mesh repair operations to handle their topological changes, such as splitting and merging [Bernard 2009; Chorin 1990, 1993; Marzouk and Ghoniem 2007]. These operations also make it challenging to establish large-scale parallel processing algorithms. The vortex sheet method, also based on mesh connectivities, is specialized to capture codimension-1 vortex structures evolving in three-dimension space [Brochu et al. 2012; Pfaff et al. 2012b].

Boundary treatment. It is challenging to set a well-posed boundary condition for the vorticity dynamical equations to not only conform to the kinematic relationship between vorticity and velocity fields but also present the physical mechanism of vorticity generated on the boundary layer [Wu et al. 2015]. Lighthill [1963] assumed boundary vorticity can be obtained by counteracting the velocities induced by the vortex sheet that covers the boundary and other vortex structures to satisfy the boundary velocity conditions. Chorin [1973] proposed a vortex-generating method that satisfies the non-slip boundary condition by generating vortex elements near the wall boundary. Though it has an explicit physical meaning and is simple to use, the computational accuracy is relatively low. This method was improved by Vines et al. [2013] and Zhang et al. [2014] by further considering the interaction among the generated vortex elements. Hung and Kinney [1988] proposed a boundary vorticity flux control method that requires solving the boundary pressure gradient precisely, which is not applicable to the Lagrangian vortex method. This method shares similarities with the integral representation method [Wu 1976] and the panel method [Erickson 1990]. Some boundary treatments used in wave propagation simulations, including the equivalent sources method and the fundamental solutions method [Mehra et al. 2013; Schreck et al. 2019], also inspired our boundary treatment.

Fig. 4. Comparison of the simulation of cigarette smoke using vortex segment and particle methods.

Acceleration. The computational complexity of the Lagrangian vortex method is $O(N_v^2)$ with the total number of vortex elements N_v. There are several classic acceleration approaches. Examples commonly used include the octree [Barill et al. 2018; Hu et al. 2020], fast multipole method (FMM) [Angelidis 2017; Greengard and Rokhlin 1987; Koumoutsakos 1993; Pepin 1990], and Particle-Particle Particle-Mesh (PPPM) [Almgren et al. 1994; Zhang and Bridson 2014]. Specifically, an octree structure separates the vortex particles into individual particle groups by subdividing the background grid in a hierarchical way. These particle groups further aggregate into super vortex particles for fast computation, at the expense of losing numerical accuracy if abundant near-field particles are merged. FMM is an elaborately designed method that ensures that the time complexity reduces from $O(N_v^2)$ to $O(N_v)$ while fundamentally maintaining the computational accuracy. However, it is difficult to design an efficient parallel algorithm due to the implementation of abundant calculus operations. Compared with FMM, the implementation of PPPM is simple and its time complexity is $O(N_v \log N_v)$. Though it decreases the computational cost, it loses partial precision at the same time. Moreover, the precision won't be improved by increasing the range of local rectification when the range exceeds a certain threshold value.

3 PHYSICAL MODEL

Considering the incompressible fluid in a domain Ω, the fluid dynamics can be described by

$$\begin{cases} \dfrac{D\boldsymbol{u}}{Dt} = -\dfrac{1}{\rho}\nabla p + \boldsymbol{f}, \\ \nabla \cdot \boldsymbol{u} = 0, \end{cases} \tag{1}$$

with proper initial and boundary conditions. Here $\boldsymbol{u}(\boldsymbol{x}, t)$ is the velocity field, $D/Dt = \partial/\partial t + \boldsymbol{u} \cdot \nabla$ is the material derivative, t denotes the time, p is the pressure, ρ is the density, and $\boldsymbol{f}$ is the body force. If we assume ρ is a constant, taking the curl of (1) yields the governing equation of vorticity $\boldsymbol{\omega} = \nabla \times \boldsymbol{u}$ as

$$\frac{D\boldsymbol{\omega}}{Dt} = (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} + \nabla \times \boldsymbol{f}. \tag{2}$$

We discretize the vorticity field on a set of discrete vortex elements

$$\boldsymbol{\omega}(\boldsymbol{x}, t) = \sum_{j=1}^{N_v} \boldsymbol{\Gamma}_j(t) f_\delta[\boldsymbol{x} - \boldsymbol{x}_j(t)]. \tag{3}$$

Here N_v is the total number of vortex elements, $\boldsymbol{\Gamma}_j$ and $\boldsymbol{x}_j$ are the vorticity strength and the central position of the j^{th} vortex element. We use f_δ as a distribution function that satisfies $\int_\Omega f_\delta(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = 1$,

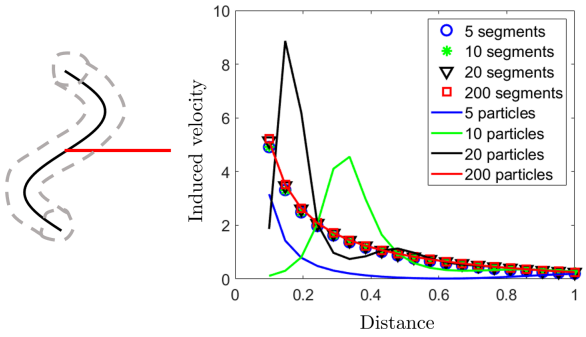

Fig. 5. The induced velocity magnitude of a vortex tube with sine-shape on the central vertical plane. The image on the left is the vortex tube diagram, where the red line denotes the position for calculating the induced velocity; the image on the right shows the induced velocity numerical results using different methods.

describing the distribution of the vorticity around $\boldsymbol{x}_j$. f_δ is designed as an isotropic mollification function in a conventional vortex particle method. Without loss of geometric generalities, the primitives of the vortex elements can be points (the classical vortex particle method), segments, triangles, etc.

Substituting (3) into the Biot–Savart (BS) law, the velocity field for vortex element convection can be obtained as

$$\boldsymbol{u}(\boldsymbol{x}, t) = \boldsymbol{u}_\infty + \frac{\sum_{j=1}^{N_v} \boldsymbol{\Gamma}_j(t) \times \boldsymbol{F}_\delta(\boldsymbol{x}, \boldsymbol{x}_j, t)}{2(N_d - 1)\pi}, \tag{4}$$

with

$$\boldsymbol{F}_\delta(\boldsymbol{x}, \boldsymbol{x}_j, t) = \int_\Omega \frac{(\boldsymbol{x} - \boldsymbol{x}')f_\delta[\boldsymbol{x}' - \boldsymbol{x}_j(t)]}{|\boldsymbol{x} - \boldsymbol{x}'|^{N_d}} \mathrm{d}\Omega', \tag{5}$$

where $\boldsymbol{u}_\infty$ is the background velocity, N_d is the dimension of the computational domain Ω, and $\mathrm{d}\Omega'$ is the volume element at $\boldsymbol{x}'$.

4 DISCRETE VORTEX SEGMENTS

4.1 Geometric Representation

We discretize the vorticity field with a cloud of vortex segments. The information carried on each vortex segment includes the positions of the two endpoints $\boldsymbol{x}_j^{\pm}$, $j = 1, 2, \cdots, N_v$ and the vorticity strength magnitude Γ_j. The segment's midpoint can be calculated as $\boldsymbol{x}_j = (\boldsymbol{x}_j^+ + \boldsymbol{x}_j^-)/2$. The vorticity strength vector on each vortex segment is calculated as $\boldsymbol{\Gamma}_j = \Gamma_j(\boldsymbol{x}_j^+ - \boldsymbol{x}_j^-)/|(\boldsymbol{x}_j^+ - \boldsymbol{x}_j^-)|$. In addition, each vortex segment has a virtual radius $\mathcal{R}$ for numerical regularization. In two-dimensional space, a vortex segment will degenerate to a vortex point $\boldsymbol{x}_j$, in which case the vortex segment method will amount to a vortex particle method.

We remark that a more precisely induced velocity around a vortex tube can be obtained with a cloud of vortex segments compared with vortex particles. We demonstrate this fact by a simple numerical test. Figure 5 plots the induced velocity magnitude of a sinusoid-shaped vortex tube on the central vertical plane. We observed that five vortex segments achieved the same precision as two hundred vortex particles when approximating the surrounding induced velocity field. Using more particles to discretize the vortex tube will result in a

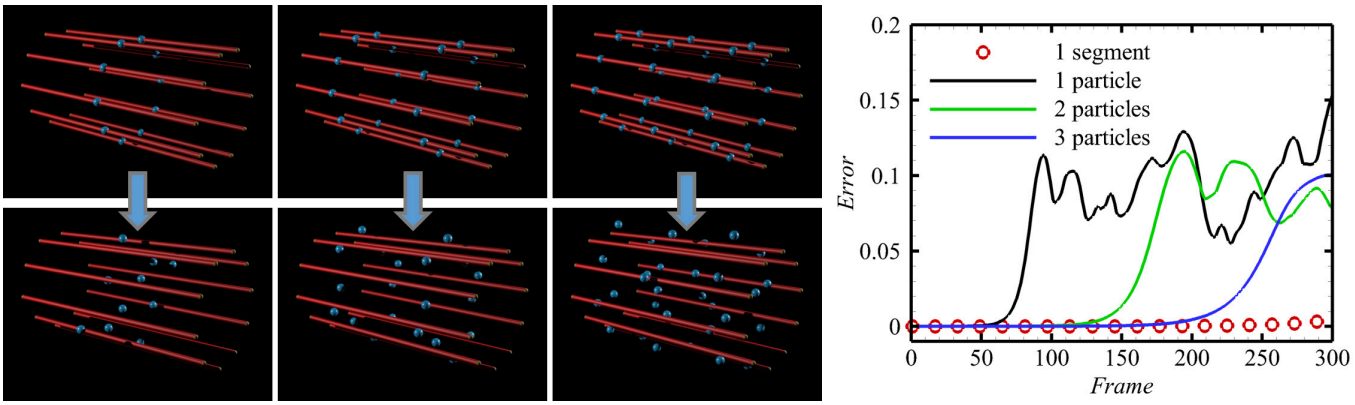

Fig. 6. Left set of pictures: comparisons of the evolution of segment-based vortex tubes and particle-based vortex tubes. From the first to third column: the particle to segment ratio is 1, 2, and 3 respectively. First row: the initial state; second row: state at frame 300. Right picture: an error comparison of segment-based vortex tubes and particle-based vortex tubes.

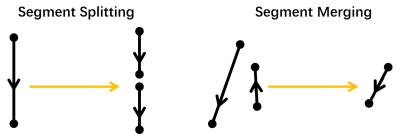

Fig. 7. Splitting and merging of vortex elements

more precise approximation. On the other side, using more segments does not enhance the precision, indicating the expressiveness of segments even with very few discretization elements.

Figure 6 shows a comparison of the evolution of the segment-based and particle-based vortex tubes. We place 1000 sets of particles or segments side by side in the common axis to balance the interactive forces among vortex elements. This operation makes the middlemost set independent of boundary during the calculation, and the simulation can be taken as part of an infinite vortex tube. The particle tube gets more and more chaotic with the evolution; even with an increasing number of particles, the system becomes stable within a certain evolution time but gets messy sooner or later with the evolution processing. The segment-based vortex tube, however, is stable during the whole evolution.

4.2 The BS law on Segment Clouds

In three-dimension space, the position of the j^{th} vortex element is represented by the segment C_j. We have f_δ as a delta function supported on C_j. Substituting f_δ into (5) and (4), we obtain the induced velocity of the j^{th} vortex element with regard to a spatial point $\boldsymbol{x}$ as:

$$\begin{aligned}\boldsymbol{u}_j^{BS}(\boldsymbol{x}) =& \frac{\Gamma_j}{4\pi}\left(\frac{\boldsymbol{x}_j^+ - \boldsymbol{x}}{\|\boldsymbol{x}_j^+ - \boldsymbol{x}\| + \mathcal{R}} - \frac{\boldsymbol{x}_j^- - \boldsymbol{x}}{\|\boldsymbol{x}_j^- - \boldsymbol{x}\| + \mathcal{R}}\right) \\ &\cdot (\boldsymbol{x}_j^+ - \boldsymbol{x}_j^-)\frac{(\boldsymbol{x}_j^- - \boldsymbol{x}) \times (\boldsymbol{x}_j^+ - \boldsymbol{x})}{\|(\boldsymbol{x}_j^- - \boldsymbol{x}) \times (\boldsymbol{x}_j^+ - \boldsymbol{x})\|^2 + \mathcal{R}^2},\end{aligned} \tag{6}$$

where $\mathcal{R}$ is a small positive number for regularization. We refer the readers to Appendix A for a more detailed deduction of (6).

This formula is presented as the form of an analytical expression in [Weißmann and Pinkall 2010], while the discrete form is given in [Padilla et al. 2019]. For 2D cases, $f_\delta(\boldsymbol{x})$ becomes a conventional Dirac delta function $\delta(\boldsymbol{x})$, and the induced velocity becomes

$$\boldsymbol{u}_j^{BS}(\boldsymbol{x}) = \frac{\Gamma_j}{2\pi}\frac{\boldsymbol{e}_z \times (\boldsymbol{x} - \boldsymbol{x}_j)}{\|\boldsymbol{x} - \boldsymbol{x}_j\|^2 + \mathcal{R}^2}, \tag{7}$$

where $\boldsymbol{e}_z$ is the normal direction of the 2D plane. We take the summation of induced velocities of all the vortex elements and the background velocity $\boldsymbol{u}_\infty$ to calculate the velocity at $\boldsymbol{x}$ as

$$\boldsymbol{u}(\boldsymbol{x}) = \sum_{j=1}^{N_v} \boldsymbol{u}_j^{BS}(\boldsymbol{x}) + \boldsymbol{u}_\infty. \tag{8}$$

4.3 Lagrangian Advection and Vortex Stretching

According to the Kelvin's circulation theorem, the vorticity strength of a vortex filament element is conservative during the action of stretching and convection:

$$\frac{\mathrm{d}\Gamma_j}{\mathrm{d}t} = \int_{D_j}\left(\frac{\mathrm{D}\boldsymbol{\omega}}{\mathrm{D}t} - \boldsymbol{\omega}\cdot\nabla\boldsymbol{u}\right)\cdot \mathrm{d}\boldsymbol{S} = 0, \tag{9}$$

where D_j is the cross-section of the vortex segment in the direction of the vorticity. Thus, neither advection nor stretching changes the vorticity strength of the vortex filament. Therefore, naively updating the position of the endpoints of each vortex segment without considering the radius change of the vortex segment

$$\frac{\mathrm{d}\boldsymbol{x}_j^\pm}{\mathrm{d}t} = \boldsymbol{u}(\boldsymbol{x}_j^\pm) \tag{10}$$

can update the vortex element with both advection and stretching. Without considering reseeding, the position and the length of a vortex segment will change during its evolution, but its shape will remain straight.

There is no vorticity stretching in two-dimensional space. Hence, the 2D vortex convection can be simplified as

$$\frac{\mathrm{d}\boldsymbol{x}_j}{\mathrm{d}t} = \boldsymbol{u}(\boldsymbol{x}_j). \tag{11}$$

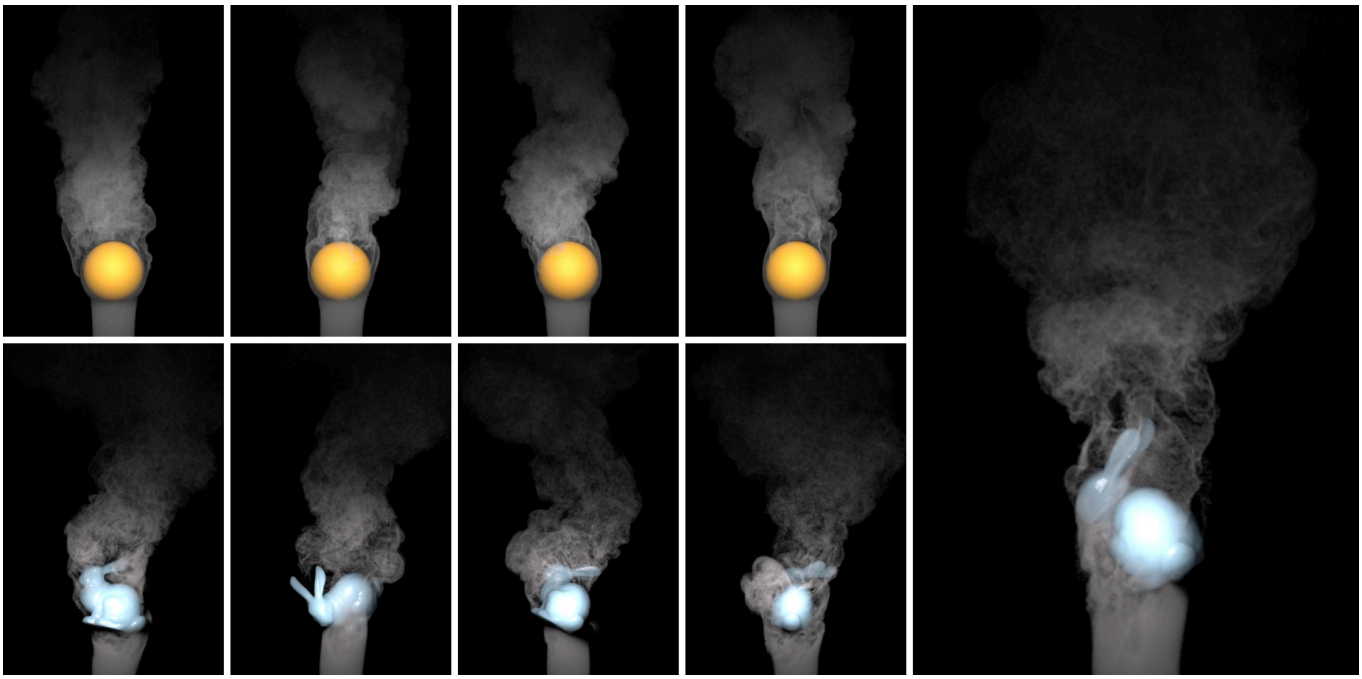

Fig. 8. Turbulence above a static sphere and a rotating bunny. the top-left four pictures: the static sphere frames at 100, 200, 240, and 400, respectively; the bottom-left four pictures: the rotating bunny frames at 83, 93, 163, and 184, respectively; the right picture: the rotating bunny frame at 271.

Similar ideas of processing vortex stretching using an explicit segment representation can also be found in the previous work of [Zhang and Bridson 2014], where virtual segments were created on a background grid in every time step to measure the local stretching effects. Compared with this hybrid representation, our segment cloud method fully leverages the vorticity expressiveness of discrete segments and naturally evolves the system's motion in a pure Lagrangian way.

4.4 Topological Changes with Segments

One of the most salient features of our segment cloud method is its capability of processing local topological changes with simple and parallel segment operations. Motivated by [Chorin 1990] on removing hairpin segments and the various particle reseeding and local re-meshing techniques in computer graphics (e.g., [Ferstl et al. 2016; Wang et al. 2020]), we devise three local segment reseeding operations including segment splitting, merging, and deletion. We showcase that the combination of these three operations can facilitate our simulation system to automatically handle complicated topological changes of vortical fluid such as vortex tube reconnection. At the same time, these segment operations enable our system to always maintain a reasonable number of the segment during the simulation.

Segment splitting. We employ a segment splitting operator (see the left of Figure 7) as splitting a segment into two as they both keep stretching. By setting a max length threshold for a segment, we split it into two new segments with ends when the segment with ends $(\boldsymbol{x}_j^-, \boldsymbol{x}_j^+)$ is greater than the threshold:

$$[\boldsymbol{x}_j^-, (\boldsymbol{x}_j^+ + \boldsymbol{x}_j^-)/2] \text{ and } [(\boldsymbol{x}_j^+ + \boldsymbol{x}_j^-)/2, \boldsymbol{x}_j^+]. \tag{12}$$

Segment merging. We devise a segment merging operation (see the right of Figure 7) to avoid two parallel segments getting too close. We check two criteria before merging a pair of segments. First, we check if the central positions of the two segments are close enough (i.e. the absolute value $|\boldsymbol{x}_i - \boldsymbol{x}_j| < \lambda$ with λ as a threshold). Second, we check if the vorticity directions of the two segments are almost opposing each other. In particular, we check if the angle between the two vorticity vectors (specified by the vorticity magnitude and the segment endpoints) is almost π (i.e. $\boldsymbol{x}_i \cdot \boldsymbol{x}_j / |\boldsymbol{x}_i||\boldsymbol{x}_j| < \cos\theta$ with θ as a given threshold). We will merge two segments if they are both close and their vorticities are pointing to roughly opposite directions. The center, length, and vorticity of the merged segment are updated by the average of the quantities stored on the two original segments as

$$\begin{cases} \boldsymbol{x}_{ij} = \dfrac{\boldsymbol{x}_i + \boldsymbol{x}_j}{2} \\ L_{ij} = \dfrac{|\boldsymbol{x}_i^+ - \boldsymbol{x}_i^-| + |\boldsymbol{x}_j^+ - \boldsymbol{x}_j^-|}{2} \\ \boldsymbol{\Gamma}_{ij} = \dfrac{2[\boldsymbol{\Gamma}_i(\boldsymbol{x}_i^+ - \boldsymbol{x}_i^-) + \boldsymbol{\Gamma}_j(\boldsymbol{x}_j^+ - \boldsymbol{x}_j^-)]}{|\boldsymbol{x}_i^+ - \boldsymbol{x}_i^-| + |\boldsymbol{x}_j^+ - \boldsymbol{x}_j^-|}. \end{cases} \tag{13}$$

Segment deletion. We delete a segment if its vorticity falls below a threshold.

Accommodated by these operations, the realization of disconnection and reconnection of vortex tubes can happen automatically

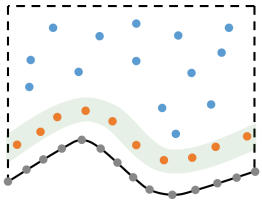

Fig. 9. The schematic diagram of boundary treatment.

while stabilizing the computational algorithm. We demonstrate the efficacy of the combination of these two basic segment operations by simulating the various vortex tube reconnection examples (see Figures 2 and 3) that are infeasible for a pure particle method or potentially complicated for a mesh-based method to process their topological changes.

5 BOUNDARY TREATMENT

Tackling a dynamic solid boundary is challenging for a Lagrangian vortex method. We present a least-squares method motivated by [Chorin 1973] to handle different velocity boundary conditions supporting both static and moving solids. Here we take the 2D situation as an example. As shown in Figure 9, we assume that the black line is the computation boundary $\partial\Omega$, and the blue points are the vortex elements in the flow field Ω. We sample N_b positions $\boldsymbol{b}_i,\ i = 1, 2, \cdots, N_b$, (grey points on the black line) uniformly on the boundary. We generate another N_g vortex elements (orange points)

$$\boldsymbol{g}_\alpha,\ \alpha = 1, 2, \cdots, N_g \tag{14}$$

around the boundary within Ω (the pale-green area).

For the Dirichlet boundary condition

$$\boldsymbol{u}(\boldsymbol{x})|_{\partial\Omega} = \boldsymbol{u}_b(\boldsymbol{x}), \tag{15}$$

along with vortex elements induced velocity within the flow field as $\boldsymbol{u}_d(\boldsymbol{x})$. We calculate the vortex strength $\Gamma_\alpha,\ \alpha = 1, 2, \cdots, N_g$, of the generated vortex elements by approximating their induced velocity regarding the boundary point $\boldsymbol{b}_i$ as $\boldsymbol{u}_b(\boldsymbol{b}_i) - \boldsymbol{u}_d(\boldsymbol{b}_i) - \boldsymbol{u}_\infty$. Therefore, we need to solve the following equation

$$\boldsymbol{K}\boldsymbol{\Gamma} = \boldsymbol{U}, \tag{16}$$

with

$$\begin{cases} \boldsymbol{K} = [\boldsymbol{K}_{i\alpha}],\ \boldsymbol{K}_{i\alpha} = \boldsymbol{u}_\alpha^{BS}(\boldsymbol{b}_i), \\ \boldsymbol{\Gamma} = [\Gamma_\alpha],\ \alpha = 1, 2, \cdots, N_g, \\ \boldsymbol{U} = [\boldsymbol{u}_b(\boldsymbol{b}_i) - \boldsymbol{u}_d(\boldsymbol{b}_i) - \boldsymbol{u}_\infty],\ i = 1, 2, \cdots, N_b, \end{cases} \tag{17}$$

where $\boldsymbol{u}_\alpha^{BS}(\boldsymbol{b}_i)$ is the induced velocity of $\boldsymbol{b}_i$ by the vortex element with the unit vorticity strength at the position $\boldsymbol{g}_\alpha$. Using the least-squares method, (16) can be solved as

$$\boldsymbol{\Gamma} = \left[\boldsymbol{K}^T\boldsymbol{K} + \epsilon_K\boldsymbol{I}(N_g)\right]^{-1}\boldsymbol{K}^T\boldsymbol{U}, \tag{18}$$

where ϵ_K is a positive regularization factor, and $\boldsymbol{I}(N_g)$ is the identity matrix of size N_g. Generally, for the calculation stability, we set $N_b \ll N_g$, where $\boldsymbol{K}^T\boldsymbol{K}$ is full rank.

For the static boundary, there is no need to update $\boldsymbol{K}$, which makes the computational complexity for updating boundary vortex elements as $O(N_gN_b^2 + N_bN_v) \approx O(N_v)$ with a relatively smaller N_b and N_g.

For the moving boundary, we need to update $\boldsymbol{K}$ for every evolutionary time step. However, if the boundary is in rigid body motion, $\left[\boldsymbol{K}^T\boldsymbol{K} + \epsilon_K\boldsymbol{I}(N_g)\right]^{-1}$ in (18) remains constant, which enables the computational complexity to remain relatively low when updating (18) with vortex elements generated near the computational boundary. We take the motion of a rigid body in the flow as an example to illustrate the process of updating (18) presented in Appendix B in detail.

Two cases addressing 2-D boundaries are demonstrated in Appendix C, showing that 2-D boundary treatment is very precise. The boundary treatment is similar with 2-D situations when dealing with 3D flows, except that we need to set the positions of the two vortex element ends $\boldsymbol{g}_\alpha^\pm,\ \alpha = 1, 2, \cdots, N_g$, near the boundary, as well as the need to calculate the vortex element induced velocity by (6) rather than (7) in 2D flows.

6 TEMPORAL EVOLUTION

We initialize each segment's position randomly and its vorticity strength based on a given vorticity distribution. We use the fourth-order Runge–Kutta method for the time integration of the vortex segments.

We carry out the temporal evolution of the vortex segment cloud using the following steps:

(1) Calculation of induced velocity using (8);
(2) Advection of vortex segments using (10) for 3D flows and (11) for 2D flows;
(3) Splitting of vortex segments using (12) for 3D flows;
(4) Merging of vortex segments using (13) for 3D flows;
(5) Generating vortex segments using (14) and computing vorticity strength using (18) for boundary flows;
(6) We delete segments if their vorticity falls below a threshold.

7 NUMERICAL RESULTS

We evaluate the efficacy of our method by several fluid simulation examples, including leapfrogging vortices, vortex tube reconnection, cigarette smoke, and turbulence behind a solid obstacle. We also compare our method with the vortex particle and the vortex filament methods. Detailed experimental settings can be found in the next paragraph. We refer the readers to our supplementary video for all the animations.

Experimental settings. Here we provide the detailed parameter settings we used in our numerical experiments. The default segment length and vortex strength are set to be 0.5 and 0.25, respectively. The split threshold was set to be 2. The distance threshold for segment merging is set to be $\lambda = 1$ and the angle threshold is set to be $\theta = 5\pi/6$. The effects of these parameters were evaluated with a different setting as in Figure 10.

Leapfrogging vortices. As shown in Figure 14, two vortex rings are initialized using the same in-plane vorticity strength and different radii. The interaction between the two rings delivers a leapfrogging

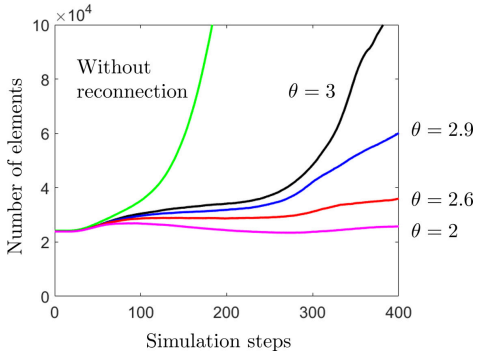

Fig. 10. Number of vortex elements used to simulate the reconnection of quasi-parallel vortex tubes

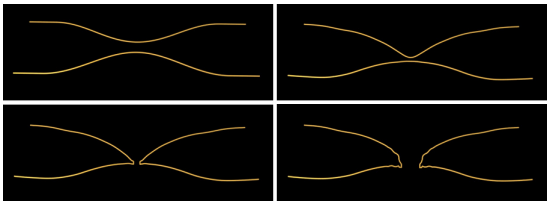

Fig. 11. Splitting and reconnection of quasi-parallel vortex filaments.

motion along the common axis. The bigger ring with smaller self-induced velocity shrinks and accelerates due to their mutual interaction and the smaller ring with larger self-induced velocity widens and decelerates. The rear decrescent ring then passes through the leading enlarged ring. We show that we can capture the leaping dynamics of the vortex rings in a long-term stable fashion without maintaining the segment connectivities as in a conventional vortex filament method.

Vortex tube reconnection. Figures 2 and 3 show the splitting and reconnection of two intersecting and quasi-parallel vortex tubes forming another two separated U-shaped tubes [Beardsell et al. 2016b; van Rees et al. 2012]. This simulation captures the main topological changes and the pinched-off vortex filaments using a mesh-free Lagrangian method, which was not feasible for any previous particle-based methods. In particular, as shown in Figure 2, we compared our results with the ones obtained by a conventional vortex particle method, which failed to capture such highly anisotropic and topologically complicated phenomena. This comparison showcased our method's unique ability in modeling the topological transitions and evolution of complicated vortex flow.

In addition, we compared our method with the conventional vortex filament method [Weißmann and Pinkall 2010]. First, as in Figure 11, we show that a single thread of filaments can capture the dominant motion of vortices yet cannot reproduce small-scale flow details such as the intermediate vortex filaments bridging the two reconnected tubes. Meanwhile, we also compared our method with the vortex filament method with multiple filaments. The simulations in Figure 12 and Figure 3 produces similar vortex evolution, except that some nonphysical deformation occurs near the boundary of

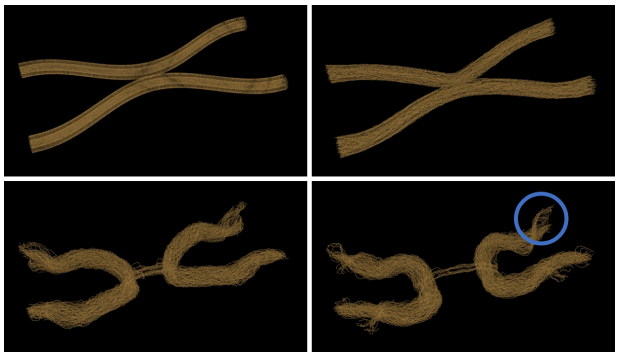

Fig. 12. Splitting and reconnection of quasi-parallel vortex tubes with vortex filament methods showing frames 1, 100, 300, and 400.

the vortex tubes (as the blue circle depicts) produced by the vortex filament method because the vortex filaments are not designed as closed rings. This comparison further weakens the role of the segment connectivities in vortex fluid simulation.

Cigarette smoke. The simulations of rising cigarette smoke in Figure 4 show a comparison between vortex segments and vortex particles. In each time step, a small number of vortex elements in both examples rise from the bottom with an average velocity forming a smoke effect as the vortex elements rising with strength gradually decay to zero. The number of the simulation particles in the computation domain converges to around 400. The small-scale vortical flow details are well-preserved in the simulation produced by the segment method. In contrast, the simulation using the particle method shows less turbulent features due to the insufficient amount of particles being used.

Vortices interacting with solids. Figure 8 demonstrates the upward rising smoke passing a static sphere and a rotating bunny swinging from side to side forming a waving wake flow. Comparisons with the vortex particle method are shown in Figure 13. We can see that our vortex segment method can capture the boundary vortical details effectively and transport these vortices with the advected segments in the flow field. Under a turbulent setting, both the vortex segment method and the vortex particle method can produce visually appealing flow motions.

7.1 Performance

Thanks to the particle nature of the segment cloud method, the implementation is inherently parallelizable. The algorithm is implemented with CUDA and all our examples were run on a GeForce RTX 2080 Ti GPU. Table 1 shows the timing statistics of our examples. We remark that exquisite flow phenomena with rich small-scale structures are demonstrated using our segment cloud method even with only a few segments. With the same number of vortex elements initialized for the three methods , we report the computation cost for particle, segment, and mesh methods with a rough ratio of $1 : 3 : 12$. The proportions of each sub-step in an iteration of different methods are listed in Table 2. The computation cost of the BS law used in the segment method can be reduced by adopting

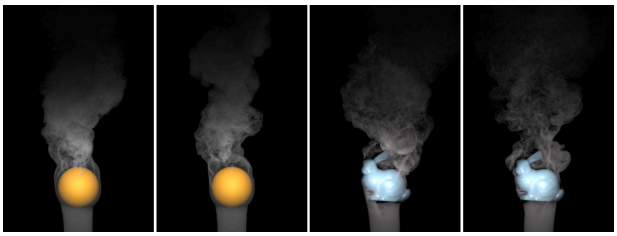

Fig. 13. Comparisons of the simulation of turbulent smoke behind a static sphere and a rotating bunny using vortex segment and particle methods. Segment method: 1, 3; particle method: 2, 4

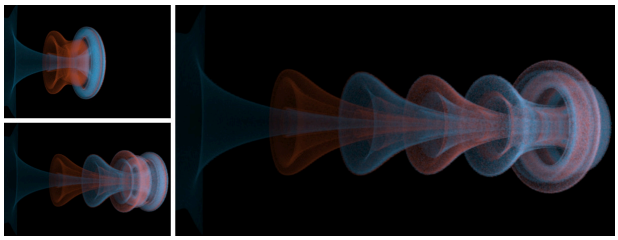

Fig. 14. Leapfrogging vortices with vortex segments showing frames 100 (top left), 200 (bottom left), and 300 (right).

Table 1. Approximate number of vortex elements and computational time of one iteration (one time step contains k iterations for a k^{th} order integrator) in a relatively stable state.

Figures	Description	Segment No.	Time (ms)
Figure 14	Leapfrogging vortices	200	4
Figure 3	Parallel reconnection	30000	200
Figure 2	Intersecting reconnection	30000	200
Figure 4	Cigarette smoke	400	6
Figure 8	Sphere shedding	9000	40
Figure 8	Bunny shedding	9000	40

Table 2. The proportion of each sub-step in an iteration of different methods, where the computation cost of calculating the induced vorticity between two vortex particles with BS law is selected as a time unit. Topology operations include splitting, merging, and deleting.

Methods	BS law	Stretching	Topology
Segment	8.4	0	3.9
Particle	1	3.2	0
Filament	4.2	0	45.1

the one used in the particle method, however, part of the calculation accuracy may be lost at the same time. Besides, the main computation cost for the filament mesh method comes from its non-parallelizable nature for mesh processing. We emphasize that the number of vortex elements can not be guaranteed to stay the same using various methods during the evolution because of the different data structure implementations for points, segments, and filament meshes. In addition, an optional acceleration algorithm is detailed in Appendix D to enable large-scale vortex segment simulation.

8 DISCUSSION

Limitations. The main limitation of the vortex segment method is that we did not consider the varying thickness of each vortex segment, which limits its capability of simulating vortex tubes with varying thicknesses such as the intricate chandeliers formed by ink dropping into fluid [Padilla et al. 2019]. Also, our current discrete model assumes a vortex element is a straight segment, leaving the induced velocity generated by the localized induction approximation of the curved vortex filaments out of consideration. In addition, compared with the traditional filament method, non-physical vortex evolution may occur with the non-divergence-free vorticity field caused by the open-ended segments. Compared with the particle method, a significant increase in computation cost can happen with a more complicated BS law being adopted.

Future work. One of the interesting future directions to explore is to incorporate surface tension into our current framework to simulate free-surface flow such as liquid tubes. We find that it is feasible to update the velocity field with both surface tension and vortex structures by discretizing the flow field into a series of impulse and vorticity elements. Similar to the BS law, the solenoidal velocity induced by impulse elements can also be obtained by a summation formula (see Equation (9) in [Cortez 1996]).

9 CONCLUSIONS

We propose a vortex segment cloud method that combines the flexibility of the traditional vortex particle method and the stability and accuracy of the vortex filament method with discretizing vorticity field as a series of vortex segments of certain lengths to simulate incompressible flows. We develop a full set of parallelizable vortex dynamics solvers with the support of dynamic solid boundaries to endorse the simulation of various challenging flow phenomena and further simulate a series of complex flows, such as the reconnection of vortex tubes, turbulent smokes, and vortex sheddings with boundaries. Our method shares unique advantages in the simulation of anisotropic flows with coherent structures that are ubiquitous in the real world.

ACKNOWLEDGMENTS

We thank Danielle Poole and all the anonymous reviewers for their constructive comments. We acknowledge the funding support from National Science Foundation (1919647). Rui Tao acknowledges the support from the China Scholarship Council visiting research student award. We credit the Houdini educational licenses for the video rendering.

REFERENCES

A. S. Almgren, T. Buttke, and P. Colella. 1994. A fast adaptive vortex method in three dimensions. *J. Comput. Phys.* 113 (1994), 177–200.
R. Ando, N. Thurey, and R. Tsuruno. 2012. Preserving fluid sheets with adaptively sampled anisotropic particles. *IEEE T. Vis. Comput. Gr.* 18 (2012), 1202–1214.
A. Angelidis. 2017. Multi-scale vorticle fluids. *ACM Trans. Graph.* 36 (2017), 1–12.

A. Angelidis and F. Neyret. 2005. Simulation of smoke based on vortex filament primitives. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 87–96.
H. Aref and E. Flinchem. 1985. Dynamics of a vortex filament in a shear flow. *J. Fluid Mech.* 148 (1985), 477–497.
G. Barill, N. G. Dickson, R. Schmidt, D. I. W. Levin, and A. Jacobson. 2018. Fast winding numbers for soups and clouds. *ACM Trans. Graph.* 37 (2018), 43.
A. Barnat and N. S. Pollard. 2012. Smoke sheets for graph-structured vortex filaments. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 77–86.
G. Beardsell, L. Dufresne, and G. Dumas. 2016a. Investigation of the viscous reconnection phenomenon of two vortex tubes through spectral simulations. *Phys. Fluids* 28 (2016), 095103.
G. Beardsell, L. Dufresne, and G. Dumas. 2016b. Investigation of the viscous reconnection phenomenon of two vortex tubes through spectral simulations. *Phys. Fluids* 28 (2016), 095103.
P. S. Bernard. 2009. Vortex filament simulation of the turbulent coflowing jet. *Phys. Fluids* 21 (2009), 025107.
T. Brochu, T. Keeler, and R. Bridson. 2012. Linear-time smoke animation with vortex sheet meshes. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 87–95.
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.
J. P. Choquin and S. Huberson. 1990. Computational experiments on interactions between numerical and physical instabilities. *Int. J. Numer. Meth. Fl.* 11 (1990), 541–553.
A. J. Chorin. 1973. Numerical study of slightly viscous flow. *J. Fluid Mech.* 57 (1973), 785–796.
A. J. Chorin. 1990. Hairpin removal in vortex interactions. *J. Comput. Phys.* 91 (1990), 1–21.
A. J. Chorin. 1993. Hairpin removal in vortex interactions II. *J. Comput. Phys.* 107 (1993), 1–9.
R. Cortez. 1996. An impulse-based approximation of fluid motion due to boundary forces. *J. Comput. Phys.* 123, 2 (1996), 341–353.
G. H. Cottet and P. Koumoutsakos. 2000. *Vortex Methods: Theory and Practice*. Cambridge University Press.
S. Eberhardt, S. Weissmann, U. Pinkall, and N. Thuerey. 2017. Hierarchical vorticity skeletons. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 6.
L. L. Erickson. 1990. *Panel methods: An introduction*. Vol. 2995. National Aeronautics and Space Administration.
V. M. Fernandez, N. J. Zabusky, P. Liu, S. Bhatt, and A. Gerasoulis. 1996. Filament surgery and temporal grid adaptivity extensions to a parallel tree code for simulation and diagnosis in 3D vortex dynamics. In *ESAIM: Proceedings*, Vol. 1. 197–211.
F. Ferstl, R. Ando, C. Wojtan, R. Westermann, and N. Thuerey. 2016. Narrow band fLIP for liquid simulations. In *Computer Graphics Forum*, Vol. 35. 225–232.
L. Greengard and V. Rokhlin. 1987. A fast algorithm for particle simulations. *J. Comput. Phys.* 73 (1987), 325–348.
O. Hald. 1979. Convergence of vortex methods for Euler's equations. II. *SIAM J. Numer. Anal.* 16 (1979), 726–755.
O. Hald and V. M. Del Prete. 1978. Convergence of vortex methods for Euler's equations. *Math. Comput.* 32 (1978), 791–809.
H. Hasimoto. 1972. A soliton on a vortex filament. *J. Fluid Mech.* 854 (1972), 477–485.
E. Hopfinger, F. Browand, and Y. Gagne. 1982. Turbulence and waves in a rotating tank. *J. Fluid Mech.* 125 (1982), 505—-534.
L. Hu, M. Chen, P. X. Liu, and S. Xu. 2020. A vortex method of 3D smoke simulation for virtual surgery. *Comput. Meth. Prog. Bio.* 198 (2020), 105813.
Y. Hu, X. Zhang, M. Gao, and C. Jiang. 2019. On hybrid lagrangian-eulerian simulation methods: practical notes and high-performance aspects. In *ACM SIGGRAPH 2019 Courses*. 1–246.
S. C. Hung and R. B. Kinney. 1988. Unsteady viscous flow over a grooved wall: A comparison of two numerical methods. *Int. J. Numer. Meth. Fl.* 8 (1988), 1403–1437.
S. Kida and M. Takaoka. 1994. Vortex reconnection. *Annu. Rev. Fluid Mech.* 26 (1994), 169–189.
D. Kleckner, L. H. Kauffman, and W. T. M. Irvine. 2016. How superfluid vortex knots untie. *Nat. Phys.* 12 (2016), 650–655.
P. Koumoutsakos. 1993. *Direct numerical simulations of unsteady separated flows using vortex methods*. Ph.D. Dissertation.
R. Krasny. 1988. Numerical simulation of vortex sheet evolution. *Fluid Dyn. Res.* 3 (1988), 93–97.
K. Kuzmina and I. Marchevsky. 2021. Flow simulation around circular cylinder at low Reynolds numbers using vortex particle method. In *Journal of Physics: Conference Series*, Vol. 1715. 012067.
S. Leibovich. 1978. The structure of vortex breakdown. *Ann. Rev. Fluid Mech.* 10 (1978), 221–246.
M. J. Lighthill. 1963. *Introduction: Boundary Layer Theory: Laminar Boundary Layer*. Oxford University Press.
T. Loiseleux, J. M. Chomaz, and P. Huerre. 1998. The effect of swirl on jets and wakes: Linear instability of the Rankine vortex with axial flow. *Phys. Fluids* 10 (1998), 1120–1134.
Y. M. Marzouk and A. F. Ghoniem. 2007. Vorticity structure and evolution in a transverse jet. *J. Fluid Mech.* 575 (2007), 267–305.
A. G. McKenzie. 2007. *HOLA: a high-order Lie advection of discrete differential forms with applications in Fluid Dynamics*. Ph.D. Dissertation. California Institute of Technology.
R. Mehra, N. Raghuvanshi, L. Antani, A. Chandak, S. Curtis, and D. Manocha. 2013. Wave-based sound propagation in large open scenes using an equivalent source formulation. *ACM Trans. Graph.* 32 (2013), 1–13.
M. Padilla, A. Chern, F. Knöppel, U. Pinkall, and P. Schröder. 2019. On bubble rings and ink chandeliers. *ACM Trans. Graph.* 38, 4 (2019).
S. I. Park and M. J. Kim. 2005. Vortex fluid for gaseous phenomena. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 261–270.
F. Pepin. 1990. *Simulation of the flow past an impulsively started cylinder using a discrete vortex method*. Ph.D. Dissertation.
T. Pfaff, N. Thuerey, and M. Gross. 2012a. Lagrangian vortex sheets for animating fluids. *ACM Trans. Graph.* 31 (2012).
T. Pfaff, N. Thuerey, and M. Gross. 2012b. Lagrangian vortex sheets for animating fluids. *ACM Trans. Graph.* 31 (2012), 1–8.
Z. Qu, X. Zhang, M. Gao, C. Jiang, and B. Chen. 2019. Efficient and conservative fluids using bidirectional mapping. *ACM Trans. Graph.* 38, 4 (2019).
L. Rosenhead. 1931. The formation of vortices from a surface of discontinuity. *Proc. Roy. Soc. A* 134 (1931), 170–192.
P. Roushan and X. L. Wu. 2005. Universal wake structures of Kármán vortex streets in two-dimensional flows. *Phys. Fluids* 17 (2005), 073601.
M. W. Scheeler, D. Kleckner, D. Proment, G. L. Kindlmann, and W. T. M. Irvine. 2014. Helicity conservation by flow across scales in reconnecting vortex links and knots. *Proc. Natl. Acad. Sci.* 111 (2014), 15350–15355.
C. Schreck, C. Hafner, and C. Wojtan. 2019. Fundamental solutions for water wave animation. *ACM Trans. Graph.* 38 (2019), 1–14.
Z.-S. She, E. Jackson, and S. A. Orszag. 1990. Intermittent vortex structures in homogeneous isotropic turbulence. *Nature* 344 (1990), 226–228.
H. Takami. 1964. *A numerical experiment with disecrete vortex approximation, with reference to the rolling up of a vortex sheet*. Technical Report. Stanford Univ. Calif.
W. M. van Rees, F. Hussain, and P. Koumoutsakos. 2012. Vortex tube reconnection at $Re = 10^4$. *Phys. Fluids* 24 (2012), 075105.
M. Vines, B. Houston, J. Lang, and W. Lee. 2013. Vortical inviscid flows with two-way solid-fluid coupling. *IEEE T. Vis. Comput. Gr.* 20 (2013), 303–315.
H. Wang, Y. Jin, A. Luo, X. Yang, and B. Zhu. 2020. Codimensional surface tension flow using moving-least-squares particles. *ACM Trans. Graph.* 39, 4 (2020).
S. Weißmann and U. Pinkall. 2009. Real-time interactive simulation of smoke using discrete integrable vortex filaments. *Proc. Vir. Real., Inter. and Phys. Sim.*, 1–10.
S. Weißmann and U. Pinkall. 2010. Filament-based smoke with vortex shedding and variational reconnection. *ACM Trans. Graph.* 29 (2010), 115.
S. Weißmann, U. Pinkall, and P. Schröder. 2014. Smoke Rings from Smoke. *ACM Trans. Graph.* 33 (2014), 140.
J. C. Wu. 1976. Numerical boundary conditions for viscous flow problems. *AIAA J.* 14 (1976), 1042–1049.
J. Z. Wu, H. Y. Ma, and M. D. Zhou. 2015. *Vortical Flows*. Springer.
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. 2019a. Construction of knotted vortex tubes with the writhe-dependent helicity. *Phys. Fluids* 31 (2019), 047101.
S. Xiong and Y. Yang. 2019b. Identifying the tangle of vortex tubes in homogeneous isotropic turbulence. *J. Fluid Mech.* 874 (2019), 952–978.
X. Zhang and R. Bridson. 2014. A PPPM fast summation method for fluids and beyond. *ACM Trans. Graph.* 33 (2014), 1–11.
Y. Zhu and R. Bridson. 2005. Animating sand as a fluid. *ACM Trans. Graph.* 24 (2005), 965–972.

A DERIVATION OF THE DISCRETE BS LAW OF A SINGLE VORTEX SEGMENT

Assuming the length of j^{th} vortex segment as L_j; unit tangential direction as $\boldsymbol{t}_j$; position as C_j. For the vortex segments, f_δ is a delta function supported on C_j. Substituting f_δ into (5) and (4), we get

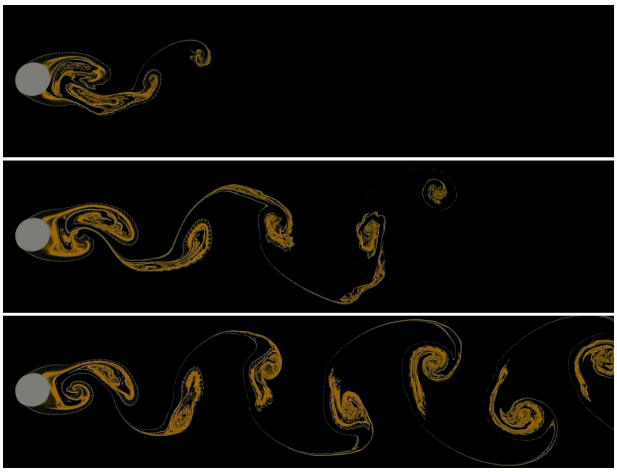

Fig. 15. von Kármán vortex street forming behind a 2-D disc. Up to bottom: frames at 100, 150, and 600..

the induced velocity of the single vortex segment as

$$\boldsymbol{u}_j^{BS} = \frac{\Gamma_j}{4\pi}\int_{C_j}\frac{\boldsymbol{t}_j\times(\boldsymbol{x}-\boldsymbol{x}')}{|\boldsymbol{x}-\boldsymbol{x}'|^3}\mathrm{d}\boldsymbol{l}'. \tag{19}$$

We parameterize the line segment L_j

$$C_j = \{\boldsymbol{x}_j + x\boldsymbol{t}_j | x\in[-L_j/2, L_j/2]\} \tag{20}$$

and assume

$$\boldsymbol{x} = \boldsymbol{x}_j + x_t\boldsymbol{t}_j + x_n\boldsymbol{n}_j, \tag{21}$$

where $x_t = |(\boldsymbol{x}-\boldsymbol{x}_j)\cdot\boldsymbol{t}_j|$ and $x_n = |\boldsymbol{x}-\boldsymbol{x}_j-\boldsymbol{t}_j(\boldsymbol{x}-\boldsymbol{x}_j)\cdot\boldsymbol{t}_j|$ are the tangential and normal components of $\boldsymbol{x}-\boldsymbol{x}_j$ respectively. Substituting (20) and (21) into (19) yields

$$\begin{aligned}
&\boldsymbol{u}_j^{BS}\\
=&\frac{\Gamma_j}{4\pi}\int_{-L_j/2}^{L_j/2}\frac{\boldsymbol{t}_j\times[\boldsymbol{x}_j+x_t\boldsymbol{t}_j+x_n\boldsymbol{n}_j-(\boldsymbol{x}_j+x\boldsymbol{t}_j)]}{|\boldsymbol{x}_j+x_t\boldsymbol{t}_j+x_n\boldsymbol{n}_j-(\boldsymbol{x}_j+x\boldsymbol{t}_j)|^3}\mathrm{d}x\\
=&\frac{\Gamma_j(\boldsymbol{t}_j\times\boldsymbol{n}_j)x_n}{4\pi}\int_{-L_j/2}^{L_j/2}\frac{1}{[(x_t-x)^2+x_n^2]^{1.5}}\mathrm{d}x\\
=&\frac{\Gamma_j(\boldsymbol{t}_j\times\boldsymbol{n}_j)}{4\pi x_n}\left\{\frac{L_j/2+x_t}{[(L_j/2+x_t)^2+x_n^2]^{0.5}}-\frac{x_t-L_j/2}{[(x_t-L_j/2)^2+x_n^2]^{0.5}}\right\},
\end{aligned} \tag{22}$$

which is equivalent to (6) without considering $\mathcal{R}$.

B UPDATE VORTICITY STRENGTH OF GENERATED VORTEX ELEMENTS FOR MOVING BOUNDARY

We take the motion of a rigid body in the flow as an example to detailedly illustrate the updating of (18) during the evolution. The motion of a rigid body in N_d-dimensional space is described as a trajectory $[\boldsymbol{R}(t), \boldsymbol{x}_c(t)]$ in the group of Euclidean motions. Here $\boldsymbol{R}(t)$ with $\boldsymbol{R}(0) = \boldsymbol{I}(N_d)$ is a orthogonal $N_d\times N_d$-matrix that describes the orientation of the body and $\boldsymbol{x}_c(t)$ is the position of the body's "center". Therefore the position of N_b boundary points and

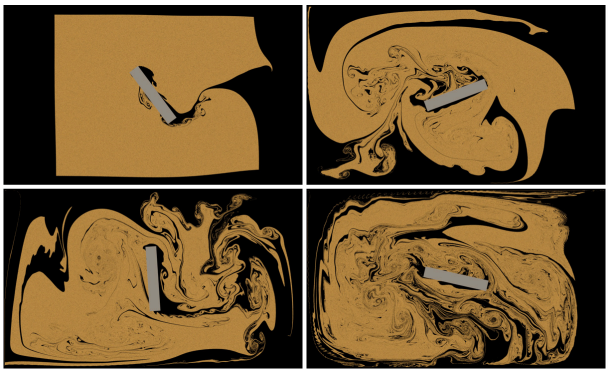

Fig. 16. A paddle mixes smoke in a 2-D box showing frames 50 (up left), 300 (up right), 450 (bottom left), and 700 (bottom right).

N_g generated vortex particles can be updated as

$$\begin{cases}\boldsymbol{b}_i(t) = \boldsymbol{x}_c(t) + \boldsymbol{R}(t)[\boldsymbol{b}_i(0)-\boldsymbol{x}_c(0)],\\ \boldsymbol{g}_\alpha(t) = \boldsymbol{x}_c(t) + \boldsymbol{R}(t)[\boldsymbol{g}_\alpha(0)-\boldsymbol{x}_c(0)],\end{cases} \tag{23}$$

as well as the velocity at $\boldsymbol{b}_i(t)$

$$\boldsymbol{u}_b[\boldsymbol{b}_i(t)] = \frac{\mathrm{d}\boldsymbol{x}_c(t)}{\mathrm{d}t} + \frac{\mathrm{d}\boldsymbol{R}(t)}{\mathrm{d}t}[\boldsymbol{b}_i(0)-\boldsymbol{x}_c(0)]. \tag{24}$$

Substituting (23) into the BS law (7) yields the update of $\boldsymbol{K}_{i\alpha}$ as

$$\boldsymbol{K}_{i\alpha}(t) = \boldsymbol{R}(t)\boldsymbol{K}_{i\alpha}(0). \tag{25}$$

It is notable that though $\boldsymbol{K}$ and $\boldsymbol{U}$ in (17) need to be updated for each time step during the calculation of (18), $\left[\boldsymbol{K}^T\boldsymbol{K}+\epsilon_K\boldsymbol{I}(N_g)\right]^{-1}$ with a relatively high computational complexity in (18) remains unchanged.

C TESTING EXAMPLES FOR BOUNDARY TREATMENT

Kármán Vortex Street. Figure 15 demonstrates an example of a 2-D flow with a constant in-flow velocity passing through a static disk, which is a repeating pattern of swirling vortices, caused by a process known as vortex shedding. For each time step, we repeatedly initialize 50 vortex elements around the static disk, as well as calculate the vorticity strength of generated vortex elements using (18). The velocity of the flow field is the summation of the induced velocity of generated vortex elements and the in-flow velocity. The vortex elements that flow out of the computational domain will not be involved in the numerical calculation anymore. With all these settings, the simulation of Kármán vortex street fits well with the classical experimental and computational results [Kuzmina and Marchevsky 2021; Roushan and Wu 2005]. Thus our boundary condition treatment is effective for scenarios that are sensitive to the influence of the boundary geometry.

A paddle mixing smoke in a 2-D box. Figure 16 shows a rotating and translating paddle shuttling back and forth through a cloud of smoke, disturbing the static smoke to a turbulent state. For this simulation, we repeatedly initialize 150 vortex elements around the moving paddle. We set a certain lifetime for vortex elements in our simulation. Specifically, the vorticity strength of the vortex element decays at a rate of 2% over each time step to 20% of its initial strength

and will be deleted at 280 time-step after its birth. Generally, vortex elements with a longer lifetime make the flow appear less viscous. We comment that in the vortex method simulation in a 2-D closed box since the number of vortex elements cannot be reduced by the merging of line segments, setting the lifetime of vortex elements is necessary to maintain the simulation stability.

D ACCELERATION

We detail the acceleration algorithm for large-scale vortex segment simulations as follows. Equations (6) and (7) show that the induced velocity of j^{th} vortex element to the position $\boldsymbol{x}$ is inversely proportional to the $\|\boldsymbol{x}_n - \boldsymbol{x}\|^{n_d-1}$ with the dimension n_d, possessing which property, we could employ lower calculating accuracy on vortex element that is away from $\boldsymbol{x}$.

We proposed an accelerating method based on multi-nested grids that combines all the merits of the classical methods mentioned in Section 2. As shown in Figure 17, the computational zone is divided into N_G sets of grids with different grid step sizes as $2^{k-1}d_G,\ k = 1, 2, \cdots, N_G$, where d_G is the smallest grid step size. The summation of the vortex strength vector in any specific grid is stored in the center of each grid cell, and we call these grids superparticles. We calculate the induced velocity of inquiring position by using the super particle that corresponds with its grid-scale based on distance.

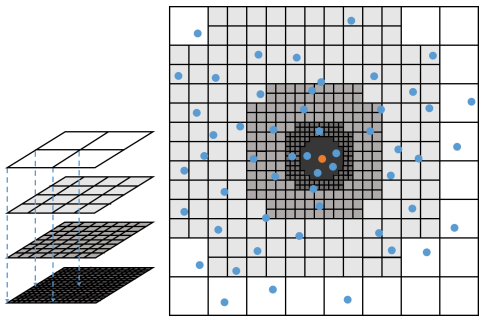

Fig. 17. The schematic diagram of nested grids. The orange dot denotes inquiring position for calculating its induced velocity, the blue dots denote the central positions of vortex elements.

Specifically, for number k^{th} grid set, we define its influence radius as $\mathcal{R}_k,\ k = 1, \cdots, N_G$ with $\mathcal{R}_{N_G} > \mathcal{R}_{N_G-1} > \cdots > \mathcal{R}_1$, as well as spherical-like zone inside the computation space Ω as

$$\Omega_k(\boldsymbol{x}) = \{\boldsymbol{y} | \|\boldsymbol{c}_k(\boldsymbol{x}) - \boldsymbol{c}_k(\boldsymbol{y})\| < \mathcal{R}_k,\ \boldsymbol{y} \in \Omega\}, \tag{26}$$

where $\boldsymbol{c}^k(\boldsymbol{x})$ is the central position of the grid cell in which $\boldsymbol{x}$ located in the k^{th} grid set. Also, we denote $\Omega_{N_G+1} = \Omega$ as the whole computational region.

We calculate the induced velocity at $\boldsymbol{x}$ using the method mentioned in section 4.2 when vortex element is in Ω_1, and using the super particle in k^{th} grid set when vortex element is in the $\Omega_{k+1} \backslash \Omega_k$ (the set difference of Ω_{k+1} and Ω_k), $k = 1, 2, \cdots, N_g$.

The computing scale of this acceleration method is determined by the number of layers the grid is divided as well as the distance $\mathcal{R}_k$ from the center position of various grids. In our simulation, we divide the computational region with scale L^{N_d} as four sets of grids: $2^{(7-k)N_d} k = 1, 2, 3, 4$ with grid steps as $d_k = L/2^{7-k}$ together with influence radiuses as $\mathcal{R}_1 = d_1, \mathcal{R}_2 = 3d_2, \mathcal{R}_3 = 3d_3, \mathcal{R}_4 = 3d_4$. With the above configuration, as shown in figure 18, the computing efficiency can be up to $O(N_v)$.

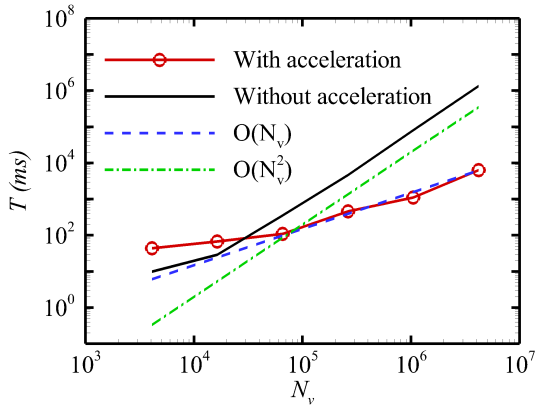

Fig. 18. Performance of the simulation with and without acceleration. The computing time increases approximately linearly/quadratically with the number of vortex elements increasing over time with/without acceleration.

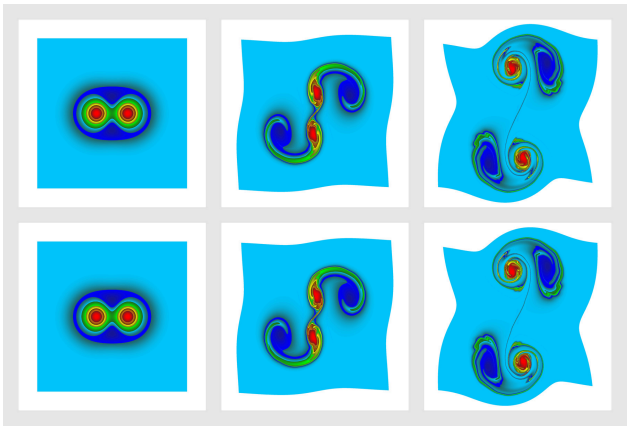

Fig. 19. Taylor vortex at $t = 0, 4, 8$. The flow fields are visualized by 1024^2 tracer particles with their colors rendered by vorticity. First row: without acceleration; second row: with acceleration.

Notably, this acceleration has little impact on accuracy. Figure 19 shows the evolution of 2-D Taylor vortex with and without acceleration respectively, where its initial condition is composed of two vortexes with a distance of 0.8 apart from each other. The vorticity distribution of each vortex is $\omega(\boldsymbol{x}) = (1/0.15 - r/0.027)\exp(0.5 - r/0.18)$, where r is the distance from $\boldsymbol{x}$ to the vortex centre [McKenzie 2007; Qu et al. 2019]. For the numerical simulation, 128^2 vortex elements are distributed symmetrically within a square region with sides of 4. As shown in figure 19, the numerical results of the Taylor vortex fit well with the direct numerical simulation [McKenzie 2007], and barely loses any flow details with acceleration.