Research on smoke simulation with vortex shedding

Rui Tao, Hongxiang Ren*, Delong Wang, Xiangen Bai

1 Navigation College, Dalian Maritime University, Dalian, China, 2 Merchant Marine College, Shanghai Maritime University, Shanghai, China

* dmu_rhx@163.com

Abstract

The Lagrangian vortex method has the advantage of producing highly detailed simulations of fluids such as turbulent smoke. However, this method has two problems: the construction of the velocity field from the vorticity field is inefficient, and handling the boundary condition is difficult. We present a pure Lagrangian vortex method, including a nested grid to accelerate the construction of the velocity field, and a novel boundary treatment method for the vorticity field. Based on a tree structure, the nested grid algorithm considerably improves the efficiency of the velocity computation while producing visual results that are comparable with the original flow. Based on the vortex-generating method, the least square method is used to compute the vorticity strength of the new vortex elements. Further, we consider the mutual influence between the generated vortex particles. We demonstrate our method’s benefits by using a vortex ring and various examples of interaction between the smoke and obstacles.

Introduction

Incompressible flows can be widely observed in our daily life, such as smoke from burning wood and the vortices generated by water flowing through obstacles. As a main research direction in fluid simulation, smoke simulation has drawn substantial attention in industry and academia. The effective use of limited computing resources to obtain a more realistic effect of smoke movement is still challenging.

Various categories of numerical methods were developed to explore different types of complex and changing smoke phenomena. In terms of computing variables, these methods are classified for solving velocity, vorticity, wave function, and so on. In terms of discretization forms, these methods are classified as Lagrangian methods, Eulerian methods, and Lagrangian–Eulerian hybrid methods. Because of their unique advantages, these methods can be used to numerically simulate various flows.

The Lagrangian vortex method has a unique advantage in the simulation of unsteady flows where the vortex structures play a leading role, such as the dynamical evolution of a forced shear layer [1], an impulsively started rigid body [2], and certain interacting coherent vortical structures [3, 4]. First, this method exhibits all the merits of the Lagrangian method. With the
material elements themselves being discrete elements, the Lagrangian vortex method does not have to construct complicated Eulerian meshes for flows with complex geometries. Moreover, it significantly decreases numerical dissipations [5] by avoiding interpolating between grids with the physical quantities stored on material elements. Second, the Lagrangian vortex elements can be placed adaptively during the process of discretizing the vorticity field, with the number of vortex elements placed being proportional to the vorticity strength. In addition, the continuous incompressible velocity field can be built by using the Biot–Saviot(BS) law from the discretized vorticity field. Thus, a continuous solenoidal velocity field that fills the entire computational domain can be reconstructed with relatively fewer discretized vortex elements needed for dynamical evolution [6].

However, unlike the velocity-based Lagrangian method [7, 8], the Lagrangian vortex method has not yet been applied widely in industry and academia for the following reasons. First, the interaction between vortex elements is non-local, leading to higher complexities of high-precision and high-speed algorithms for large-scale simulations [9]. Second, the boundary condition of the flow field is generally given based on the velocity field, making the conversion of the velocity to vorticity boundary conditions necessary [6, 10]. This conversion could be nontrivial, particularly for moving boundaries.

To solve the first of the aforementioned problem, we use a nested grid to group the vortex particles based on their positions and treat the vortex particles in the same group as one super vortex particle in an approximate manner. When calculating the velocity at a given position, we traverse the super vortex particles instead of all the original vortex particles. Thus, we can reduce the number of traversals to decrease the computational cost. We use the vortex–generating method to handle the boundary condition for the second problem. Unlike the momentum form of the Navier–Stokes equation, the vorticity at the boundary cannot be directly set. The plane method [5, 11] in classic airfoil theory is only suitable for simple non-slip boundary conditions. In this study, the generated vortex particles are placed on the boundary to satisfy the flow’s non-through and non-slip boundary conditions. The vorticity of the generated vortex particles is implicitly determined by solving a linear system. Our contributions are summarized as follows:

1. A parallel nested grid method that groups the vortex particles to decrease the number of vortex particles traversed during the velocity computation, thus reducing the computational cost;
2. A novel boundary treatment method based on optimization. The least-square method is used to compute the vorticity strength of the generated vortex particles, satisfying the non-through and non-slip boundary conditions;
3. A pure Lagrangian vortex dynamics framework that simulates complex vortical smoke phenomena with moving obstacles.

Related work

Modeling the dynamic evolution of smoke is a laborious task. Owing to the development of algorithms and hardware, researchers have adopted physics-based methods to simulate smoke. Stable Fluids [12] is a classic method based on the Eulerian method. Although this method produces massive numerical dissipation, it can obtain unconditional numerical stability with large time steps. Subsequently, numerous methods were proposed to reduce the numerical dissipation, such as the BFECC [13], MacCormack [14], advection–reflection [15], “BiMocq2”
and vorticity confinement \([17–19]\) approaches. A common feature of the Eulerian method is that the fluid domain should be discretized in advance. Hence, the Eulerian method is not suitable for smoke simulation in open spaces.

Among the Lagrangian methods, the most widely studied methods are the smooth particle hydrodynamics (SPH) and Lagrangian vortex methods. The advantage of the SPH method is its high computational efficiency. This is because the SPH method does not have to solve the global Euler equation. However, its accuracy relies on the massive distribution of particles, and this is not suitable for smoke simulation in large spaces. To solve this problem, Ren et al. \([20]\) further improved the performance of SPH smoke simulation by solving for the visible smoke without involving the transparent air.

The advantage of the SPH method is its high computational efficiency. This is because the SPH method does not have to solve the global Euler equation. However, its accuracy relies on the massive distribution of particles, and this is not suitable for smoke simulation in large spaces. To solve this problem, Ren et al. \([20]\) further improved the performance of SPH smoke simulation by solving for the visible smoke without involving the transparent air.

The Lagrangian vortex method is the most efficient method for simulating complex flows with sparse data and minimal numerical dissipation. This method discretizes the vorticity field on the Lagrangian vortex element and implements a velocity field by using the BS law without the numerical dissipation that is associated with projection. The flow data can be stored on vortex particles \([9, 21, 22]\), vortex filaments \([3, 23–26]\), and vortex surfaces \([27–29]\). Based on the connectivity of the mesh, the vortex surfaces can be used to track the fluid surfaces. However, this discrete form cannot deal with the topological changes of vortex structures. The vortex filament discretizes the vorticity field on the closed curves, which naturally represents the vortex structure. A fundamental problem with the vortex filament method is the complicated curve repair operation to the topological changes in the filament, making large-scale parallel computing challenging. The vortex particle method, also known as the vortex blob method, generally replaces point vortices with certain vortex cores to remove the singularity in the kernel function. The vortex core has various shapes, such as sphere, ellipsoid, or small vortex sheet. Compared with the vortex filament and vortex surfaces, it has a lower computational overhead and better stability but with decreased accuracy.

One common problem of the Lagrangian vortex method is that the integration operation’s time complexity, from the vorticity field to the velocity field, is \(O(N^2)\). To overcome this problem, some classic solutions were proposed, such as the Fast Multipole Method (FMM) \([30]\), Particle-Particle Particle-Mesh (PPPM) \([9]\), and Octree-Based Method (OBM) \([22]\). The FMM can reduce the integral operation’s time complexity to \(O(N)\) and guarantee sufficient accuracy. However, this algorithm is considerably complex and difficult to deploy on new hardware, making it impossible to be widely used in industries. The PPPM method has a computational time complexity of \(O(N \log N)\) at the expense of partial accuracy and is relatively easy to implement. The OBM method uses a hierarchical tree data structure to group vortex particles, and the computational time complexity is \(O(N \log N)\). However, the tree structure that carries the vortex particles’ spatial distribution information makes the parallelization of the algorithm challenging.

The boundary treatment is notoriously problematic in the Lagrangian vortex method. In computer graphics, researchers generally place vortex layers at the boundary to satisfy the non-slip and non-through boundary conditions. Based on this idea, Chorin \([21]\) proposed a vortex-generating vortex method to deal with the boundary conditions of vortical flows. Park \([31]\) obtained a reasonable vorticity distribution on the solid boundary by solving a system of \(3K_p\) equations, where \(K_p\) is the number of generated vortex elements. Vines \([32]\) reduced the size of the linear system to \(K_p\) by applying only a non-through boundary condition to the normal of the solid, thereby solving the boundary integral problem. Appropriate boundary conditions can reflect the physical mechanism of vorticity generation on the solid. An SPH turbulence model can be used to deal with boundary conditions of vortex filaments \([33]\).
Materials and methods

Vortex dynamics

Vortex particle method. The Navier–Stokes equation can be written in two forms: the velocity ($u$)–pressure ($p$) form and the velocity ($u$)–vorticity ($\omega = \nabla \times u$) form. The former focuses on tracking the change in momentum to solve the pressure, viscous force, or other external forces in the equation. In this study, we use the latter to simulate the smoke by using the following vorticity equation:

$$\nabla \cdot u = 0, \quad (1)$$

$$\frac{D\omega}{Dt} = (\nabla u) \cdot \omega + \nu \nabla^2 \omega, \quad (2)$$

where $\nu$ is the viscosity coefficient. Eq (1) expresses the incompressibility of the fluid. In Eq (2), the pressure term disappears. The left side of Eq (2) shows the change rate of vorticity under the Lagrangian viewpoint, which can be further expressed as $\partial \omega / \partial t + (u \cdot \nabla)\omega$. Furthermore, $(u \cdot \nabla)\omega$ is the advection term. The right side of Eq (2) shows the stretching and diffusion terms, from left to right. The stretching term does not exist in 2D space. In 3D space, as a key element of turbulent motion, it can produce local intensification and reorientation of the vorticity. The diffusion term describes the fluid’s vorticity diffusion due to friction, which can smooth the diversity of fluid movement and slow down the motion of flow. The diffusion term can be solved by employing the particle strength exchange (PSE) method [34], which is not discussed in this paper.

Based on the Lagrangian viewpoint, the vorticity field is discretized on a set of vortex particles; this is consistent with the fact that the non-zero vorticity is generally concentrated in the flow trajectory [35]. The vorticity strength carried by the vortex particles can be computed by using following equation:

$$\Gamma_i = \int_{V_i} \omega dV \approx \omega_i V_i, \quad (3)$$

where $\Gamma_i$, $\omega_i$, and $V_i$ is the vorticity strength, vorticity, and volume of the $i^{th}$ vortex particle, respectively.

Stretching. The methods used for solving the stretching term can be roughly classified into two categories. An explicit method is to discretize the velocity on the grid and compute the Jacobi matrix which is all partial derivatives of all components ($x$, $y$, $z$) of velocity $u = (u, v, w)$. The accuracy of this method is considerably low because of the interpolation. Another approach is to capture the vortex structure’s stretching by calculating the deformation of the geometric structure, such as the vortex ring or vortex sheet in the flow, which cannot be directly applied in the vortex particle method.

We use the vortex segment approach to solve the stretching term. To use vortex segments, which naturally represent the vortices’ geometric structure, we build a bridge between the vortex particles and the vortex segments. Each vortex segment is a small cylinder of length $h$ and constant circulation $\kappa$. The segment has the same direction as the vorticity strength and two endpoints, $x_l$ and $x_r$. A vortex particle of vorticity strength $\Gamma_i$ can be translated to a vortex segment of unit direction $T_i = \frac{x_r - x_l}{\|x_r - x_l\|}$, length $h_i$, and circulation $\kappa_i = \frac{\Gamma_i}{h_i}$.
We first convert the vortex particles into vortex segments and then update the ends of the vortex segments by applying the following equation:

\[
\begin{align*}
    x_{n+l}^{n+1} &= x_{n+l}^n + \Delta t u_{n+l}^n, \\
    x_{n-r}^{n+1} &= x_{n-r}^n + \Delta t u_{n-r}^n,
\end{align*}
\]

(4)

where \( u \) is the ends’ velocity that can be calculated by using Eq (8).

Finally, we translate the vortex segments back into vortex particles with updated vorticity strengths. If we directly set the vortex particle with vorticity strength \( \tilde{\Gamma}_i^{n+1} = \kappa_i (x_{n+i}^{n+1} - x_{n-i}^{n+1}) \), even if the time step is small, the simulation diverges quickly. The reason is that, although the magnitude of velocity is small, the gradient of the velocity field can be large in a turbulent flow. This fact makes the velocity difference between the two ends significantly large, resulting in an unstable simulation. Therefore, we weight \( \tilde{\Gamma}_i^{n+1} \) and \( \Gamma_i^{n+1} \) to stabilize the simulation and obtain the final vorticity strength:

\[
\Gamma_i^{n+1} = (1 - \delta) \Gamma_i^n + \delta \tilde{\Gamma}_i^{n+1},
\]

(5)

where \( \delta \in (0, 1) \) is the interpolation coefficient. Notice that in our vortex segment approach, we smooth the update of the vorticity instead of \( \tilde{\Gamma}_i^{n+1} \) to preserve sharp features as much as possible.

**Velocity computation.** In the Lagrangian vortex method, the velocity field can be obtained from the vorticity field by using the BS law:

\[
\mathbf{u}(\mathbf{p}) = \int_{\mathbb{R}^d} \mathbf{\omega}(\mathbf{x}) \times \frac{\mathbf{p} - \mathbf{x}}{||\mathbf{p} - \mathbf{x}||^2} \, d\mathbf{x},
\]

(6)

where \( \mathbf{p} \) is the inquiring position, and \( d \) is the computational domain’s dimension. In this study, the integration in Eq (6) is equivalent to the summation of all vortex particles

\[
2D: \mathbf{u}(\mathbf{p}) = \sum_{i=1}^{K_v} \frac{(0, 0, \Gamma_i) \times (\mathbf{p} - \mathbf{x}_i)}{2\pi ||\mathbf{p} - \mathbf{x}_i||^2},
\]

(7)

\[
3D: \mathbf{u}(\mathbf{p}) = \sum_{i=1}^{K_v} \frac{\Gamma_i \times (\mathbf{p} - \mathbf{x}_i)}{4\pi ||\mathbf{p} - \mathbf{x}_i||^3},
\]

(8)

where \( K_v \) is the number of vortex particles, and \( \mathbf{x}_i \) and \( \Gamma_i \) are the position and vorticity strength of the \( i^{th} \) vortex particle, respectively. Moreover, \( \Gamma_i \) is a scalar in 2D. The BS law has the following property: the formula’s divergence-free velocity field is not unique. Consequently, the corresponding vorticity field is uniquely determined for a given velocity field. This property allows us to add a harmonic velocity field.

However, the BS law has two problems. One is the expensive numerical integration. We introduce a solution to this problem in Section. Another problem is that vortex particle self-induction can cause a singularity and numerical instability. Researchers have proposed certain methods to modify the BS law to circumvent this problem. Angelidis and Neyret replaced the BS kernel with a radial basis function [36], which is defined smoothly around the centroid. This makes it easier to analyze the integral.

Chorin [21] proposed the concept of a “vortex cluster,” which is a small area with a high concentration of vorticity. The shape of the area can be circular, elliptical, or vortex sheet. The combination of the vortex clusters and different non-singularized kernel functions achieves different smoothing effects. In this study, an artificial smoothing parameter, \( \sigma \), is added to Eqs...
(8) and (9) is our modified BS formula.

\[
\begin{aligned}
\mathbf{u}(\mathbf{p}) &= \sum_{i=1}^{K_v} \frac{\mathbf{\Gamma}_i \times (\mathbf{p} - \mathbf{x}_i)}{K} \\
K &= 4\pi ||\mathbf{p} - \mathbf{x}_i||^3 + \sigma^2.
\end{aligned}
\]

When the inquiring position, \(\mathbf{p}\), is far away from the vortex particles, \(K \approx 4\pi ||\mathbf{p} - \mathbf{x}_i||^3\). When \(\mathbf{p}\) is close to the vortex particles, the properties expressed by Eq (9) are similar to the solid vortex core. Fig 1 shows the velocity profile induced by the modified BS law.

**Fast summation**

A major problem of the BS law is the time-consuming summation. As described by Eq (8), we need to traverse \(K_v\) vortex particles when calculating the velocity of the inquiring position. This means that the time complexity of calculating the velocity of \(K_v\) vortex particles is \(O(K_v^2)\). When a phenomenon requires massive vortex particles to show the details, the computational cost is extremely high.

By using the BS law, we derive an approximate method for reducing the traverse number in summation. According to Eq (9), the velocity is inversely proportional to the square of the distance. This property shows that the distant vortex particles have less influence on the final result, whereas the closer vortex particles have a significant contribution. If the distance between the vortex particles is much shorter than their distance to the inquiring position, the
distance between these vortex particles and the inquiring position can be regarded as approximately equal. By employing the approximation, we can obtain

$$u(p) = \frac{1}{4\pi} \sum_{i=1}^{K_s} \frac{\mathbf{\Gamma}_i \times (p - x_i)}{||p - x||^4} \approx \frac{(p - \bar{x})}{4\pi ||p - \bar{x}||^4} \times \sum_{i=1}^{K_s} \mathbf{\Gamma}_i,$$

(10)

where $K_s$ is the number of vortex particles, $p$ is the inquiring position, and $\bar{x}$ is the weighted average position of the vortex particles.

By applying Eq (10), we find that a group of vortex particles have roughly the same contribution as a super vortex particle towards a distant inquiring position. Therefore, based on their positions, we can group the vortex particles and treat those in the same group as a super vortex particle. This can reduce the computational overhead.

In this study, a nested grid algorithm that is similar to a tree structure is proposed to group vortex particles. The nested grid is a set of grid layers, $G = \{G_i|i = 1, 2, \ldots, K_n\}$, with different resolutions and the same bounding box, where $K_n$ is the number of grid layers. The resolution relationship between grid layers $G^i$ and $G^{i+1}$ is $(K_n^{i+1})^d = (2 \times K_n^i)^d$, where $d$ is the spatial dimension. It means that each cell in grid layer $G^i$ corresponds exactly to the $2^d$ cells in $G^{i+1}$.

The consistency in space is convenient for program implementation. All grid cells are regarded as a super vortex particle, and its vorticity and position are calculated by using Eq (11).

$$\begin{align*}
\mathbf{x}_s &= \frac{\sum_{i=1}^{K_n} ||\mathbf{\Gamma}_i|| \mathbf{x}_i}{\sum_{i=1}^{K_n} ||\mathbf{\Gamma}_i||} \\
\mathbf{\Gamma}_s &= \sum_{i=1}^{K_n} \mathbf{\Gamma}_i,
\end{align*}$$

(11)

where $\mathbf{\Gamma}_i$ and $\mathbf{x}_i$ are the vorticity strength and position of the super vortex particle; and $K_n$, $\mathbf{\Gamma}_n$ and $\mathbf{x}_n$ are the total number, vorticity strength, and position of the $i^{th}$ vortex particles in the cell, respectively.

Based on the spatial consistency of different grid layers, we design the traversal manner of the super vortex particles as follows:

1. Starting from the first grid layer, $G^{i=1}$, traverse all the super vortex particles, $C^{i=1}_n$ in $G^{i=1}$, that is, the grid cells. Find all cells, $C^*_i$, that do not contain the inquiring position, $p$, and directly calculate their induced velocity. Then, find cell $C^*_i$ that contains the inquiring position, $p$, and enter the next grid layer, $G^{i=2}$.

2. Starting from the second grid layer, $G^{i=2}$, traverse only the cell set, $C^*_i$, that corresponds to $C^*_i$ in the upper grid layer. Find the cell set, $C^*_i$, that does not contain the inquiring position, $p$, in $C^*_i$ and compute their induced velocities to accumulate the result with the upper grid layers. Then, find cell $C^*_i$ containing the inquiring position, $p$, enter the next grid layer, $G^{i=3}$, and repeat the second step until the penultimate grid layer.

3. In the last grid layer, $G^{i=K}$, find the cell set corresponding to grid cell $C^*_i$ in the upper grid layer and compute their induced velocity.

We use a 2D example to illustrate the operation of the nested grid. First, we group the vortex particles depending on their distribution in the computation domain, as shown in Fig 2(b). The vortex particles in the same group are regarded as a super vortex particle. Fig 2(c) shows the distribution of super vortex particles in different grid layers for calculating the velocity of $p$. Thus, the time complexity of calculating the velocity of $p$ is reduced from $O(K_n)$ to $O(\log K_n)$. 
For games or real-time applications, performance and effects are more important than accuracy. Although our method loses some accuracy, it substantially improves computational efficiency and can obtain satisfactory visual results.

**Boundary treatment**

We use the vortex-generating method to deal with the boundary conditions for the Lagrangian vortex method. Owing to friction and viscosity, vortices are generated on the surface when the flow passes the obstacle. The vortex-generating method is consistent with this fact. The method can be divided into two steps: (1) compute the position and direction of the generated vortex particles; (2) calculate the magnitude of vorticity strength of the generated vortex particles. From simple to complex, we take the 2D static boundary example to introduce our method and then present the dynamic boundary example. Whether it is a static or dynamic boundary, we follow a principle when dealing with boundary conditions: the obstacle velocity equals the flow velocity.

First, we introduce the method of computing position $x_g$ and direction $t_g$ of the generated vortex particles. As shown in Fig 3, we sample some boundary points that are represented by the gray points with position $x_b$. The number, $K_g$, of boundary sampling points is the same as the number of generated vortex particles. To offset the slip and through velocity of the flow at the boundary sampling point, $t_g$ should be tangent to the velocity, $u_f$, of the flow at the
boundary and the normal, \( n \), of the solid surface. By normalizing the vorticity direction, we can obtain

\[ t_x = \frac{u_x \times n}{\|u_x \times n\|}. \]  

(12)

We place the generated vortex particle near the boundary along vector \( a = \frac{t_x \times n}{\|t_x \times n\|} \), as shown in Fig 3. As a result, \( x_g = x_0 + a \). The placement makes the generated vortex particle’s induced velocity at the boundary opposite to \( u_f \). In our method, \( \alpha \) is set artificially. The larger \( \alpha \) is, the bigger the vorticity strength of the generated vortex particle, and vice versa.

Next, we use the least square method to compute the magnitude, \( \Gamma_g \), of the vorticity strength of the generated vortex particles. Assuming that the velocity of background flow is \( u_\infty \), we consider that the summation of the induced velocity, \( u_g \), of the generated vortex particles to the boundary and \( u_f + u_\infty \) is equal to the velocity, \( u_b = 0 \), of the solid under ideal conditions. Therefore, the problem has been reduced to finding a minimizer of the total velocity.

\[ \Gamma_g = \text{argmin}_g \sum_{i=1}^{K_g} \left\| u_g'(\Gamma_g) + u_f + u_\infty \right\|^2. \]  

(13)

We derive Eq (13) as

\[ A\Gamma_g = BU, \]  

(14)

where \( A = [BB^T]^{-1}, B = \sum_{j=1}^{K_g} b(x_j^0, x_j), b(x_j^0, x_j) = [b_1(x_j^0, x_j), \ldots, b_N(x_j, x_j)]^T \), \( u = u_g + u_f + u_\infty \), and \( b_j(x_j^0, x_j^0) = (t_x \times n) \times (x_j^0 - x_j^0)/(2\pi\|x_j^0 - x_j^0\|^2 + \sigma^2), j = 1, \ldots, K_g \), which is the boundary velocity induced by the generated vortex particles with the unit vorticity strength vector. Eq (14) can be solved as

\[ \Gamma_g = A^{-1}BU, \]  

(15)

where \( A = [\sum_{j=1}^{K_g} b(x_j^0, x_j)b(x_j^0, x_j)^T]^{-1} \).

Different from the static boundary, we need to compute the velocity, \( u_b \), of the solid for the dynamic boundary treatment. We also need to update the boundary sampling points in each time step, which can be computed by using Eq 16. The rigid body’s motion is divided into two parts: the center translation and the rotation relative to the center. The position of the boundary sampling points is found as

\[ x_i(t) = x_i(t) + [x_i(0) - x_i(0)]M_i(t)^T, \]  

(16)

where \( x_i(t) \) and \( M_i(t) \in \mathbb{R}^{d\times d} \) are the center and rotation matrix of the rigid body, respectively. \( d \) is the spatial dimension, and \( [x_i(0) - x_i(0)]M_i(t)^T \) represents the rotation of the rigid body. Then, we take the time derivative of \( x_i(t) \) to determine the velocity, \( u_i \), of the rigid body at the boundary sampling point.

Substituting Eq (16) and \( u_i \) into Eq (15) yields the magnitude of the vorticity strength of the generated vortex particles at the dynamic boundary:

\[ \Gamma_g = A^{-1}BM_iU^*, \]  

(17)

where \( U^* = u_g + u_\infty + u_f - u_b \).
**Time integration scheme**

Our time integration scheme can be summarized as follows. We first update the position of the vortex and tracer particles. In our method, tracer particles are used for visualization. Then, we handle the boundary condition and solve the stretching and diffusion terms. Finally, we construct the velocity field from the vorticity field by using the modified BS law and the background flow. In each time step, we update the states of the flow by using the following steps, as depicted in S1 Fig:

1. Advection. Calculate the induced velocity by using Eq (9) and the advect vortex and tracer particles by using the fourth-order Runge–Kutta method.
2. Boundary treatment. Perform collision detection between the particles and the solid, and compute the position and vorticity strength of the generated vortex particles.
3. Stretching. Use the vortex segment method to update the vorticity strength of the vortex particles.
4. Diffusion. Use the PSE method to solve the diffusion term.
5. Velocity computation. Obtain the bounding box based on the distribution of particles. Then, calculate the velocity of the vortex and tracer particles using our nested grid algorithm.
6. Background flow. Add the velocity of background flow to the particles.

**Results**

**Parameter study**

We conduct two sets of simulations with different parameter values to better understand the impact of the vortex particle’s volume \(V\) and the stretching coefficient \(\delta\) on the simulation. First, we simulate the rising smoke with three different values of \(V\) (as shown in Fig 4). In this example, the computational domain is approximately \(20 \times 20 \times 20\). We emit a vortex circle (consisting of a set of vortex particles) every 5-timesteps, whose normal towards the upper right. From left to right in Fig 4, \(V\) equal to \(64 \nu^3\), \(8 \nu^3\) and \(\nu^3\), respectively. Here, \(\nu\) is \(6.28 \times 10^{-3}\). To ensure the vorticity strength of the three flows are the same, the vortex particles’ number of the vortex circle is 1000, 2000 and 4000 from left to right, respectively.

We observe that the overall motion trends of the three cases are consistent, which shows that the volume of the vortex particle does not change the macroscopic behavior of the flow. In the fine scale, the difference of the vortices with different \(V\) are marked by the red circles.

![Fig 4. Rising smoke with three different values of \(V\). Left: \(V = 64\nu^3\); middle: \(V = 8\nu^3\); right: \(V = \nu^3\). Here, \(\nu = 6.28 \times 10^{-3}\). And the difference of the vortices are marked by the red circles.](https://doi.org/10.1371/journal.pone.0269114.g004)
vortices generated by the vortex particles with big $V$ merge into larger vortices earlier. In contrast, the vortices in the flow with small $V$ maintain longer. It is because the flow with small $V$ has more vortex particles, which samples the vorticity field more uniformly. However, the increase in the number of vortex particles also brings more computational overhead. Therefore, we should balance the computational speed and animation quality according to the actual situation.

Second, we use three different values of $\delta$ in the example of colliding vortex rings (see Fig 5). We set a uniform grid inside the ink and place the vortex particles in the center of every grid. Except for the normal, the vorticity distribution of the two inks is the same, which is calculated by applying Eq (18). We observe that the vortex re-connection phenomenon can be obtained after the collision of rings in all cases, which is consistent with the result reported in [37]. It proves that the numerical damping caused by $\delta$ does not change the physical nature of the simulation. When $\delta = 1$, the vorticity is not suppressed. The vortices after re-connection show irregular motion and instability. In the case with smaller $\delta$, the flow after the collision is more stable. This proves that our stretching approach can stabilize the simulation. In addition, we compare our approach to a MacCormack solver [14], as shown in Fig 6. The simulation with a MacCormack solver can not reproduce such phenomena.

**Nested grid results**

**Vortex ring.** The simulation results of the octree grid [22], our nested grid and the direct summation are qualitatively compared by employing the vortex ring example. In the comparison, the initial conditions are set identically. The vorticity distribution of the vortex ring is approximated according to the empirical formula (18) [38]:

$$\omega(r) = \begin{cases} 
\omega_{\text{max}} \left[ 1 - \exp\left( -\frac{K}{r^2}\exp\left( \frac{1}{r - 1} \right) \right) \right] & \text{if } r^r < 1, \\
0 & \text{else,}
\end{cases}$$  

(18)

where $r^r = r/R$, $R$ is the radius of the cross-section of the vortex ring, $r$ is the distance from the vortex particle to the vortex axis, and $K = 1/2\exp(2)\log(2)$. The vorticity is concentrated in the core of the vortex ring, and the vorticity direction is tangent to the vortex axis and perpendicular to the normal of the vortex ring. The nested grid is a five-layer grid with the configuration of $(K^r)^5 = 64^5$. As depicted in Fig 7, our method’s simulation results are very similar to those of the direct summation method. Although some details are lost on the fluid surface and the
Fig 6. Colliding vortex rings using different methods. The first row: the MacCormack solver; the second row: our method.

https://doi.org/10.1371/journal.pone.0269114.g006

Fig 7. Vortex ring comparison of the octree grid method, the nested grid method and the direct summation method. The first row: the octree grid; the second row: the direct summation; the third row: our nested grid.

https://doi.org/10.1371/journal.pone.0269114.g007
vortex ring’s wake is slightly deformed, our method requires significantly less computation. And compared with the octree grid, our nested grid is visually closer to the original flow. Fig 8 shows the nested grid method’s computation time on the CPU and GPU with the different number of vortex particles. As the number of vortex particles increases, the computational time of our method grows linearly, proving that the time complexity is $O(K_v)$.

Boundary treatment result

Kármán Vortex Street. Fig 9 shows the simulation of classic fluid phenomena, Kármán Vortex Street. In this example, we compare our method with the panel method [31]. At the boundary of the disc, we sample 200 panels with the panel method and 200 boundary points with our boundary treatment method. As shown in Fig 9, some vortex particles that enter the obstacle’s interior are eliminated after advection. The generated vortex particles passing the obstacle produce a repeating pattern of swirling vortices, forming the Kármán vortex street phenomenon. Under the influence of viscosity, the vorticity strength of the vortex particles that are far away from the obstacle decreases slowly, and their velocities gradually become similar to the velocity of the background flow. However, the vortices in the panel method can not maintain for a long time, and finally merge together. This comparison showcase the effectiveness of our boundary treatment method. In addition, we simulate the vortex shedding behind two discs and an airfoil, as shown in Fig 10. We sample 200 boundary points around every disc and 400 boundary points around the airfoil. The airfoil’s angle is 45˚. Compared with the disc, the flow passing around the airfoil presents a more irregular motion. The simulation shows that our method can handle the boundary of the obstacles with complex geometry.
Fig 9. Comparison of Karman vortex street examples using different boundary treatment methods. The first row: the panel method; the second row: our boundary treatment method. Blue and red represent the vortex particles with negative and positive vorticity, respectively. Mineral blue represents the tracer particles.

https://doi.org/10.1371/journal.pone.0269114.g009

Fig 10. 2D boundary treatment results. The first row presents Karman Vortex Street behind two discs; the second row displays vortex shedding behind an airfoil. Blue and red represent the vortex particles with negative and positive vorticity, respectively. Mineral blue represents the tracer particles.

https://doi.org/10.1371/journal.pone.0269114.g010
In addition to the visual results, we provide the computation accuracy of the three cases presented in Fig 11. We define the ratio of velocity elimination as follows:

\[
e = \frac{\sum_{i=1}^{K_g} \left( \| u'_f + u'_\infty - u'_b \| - \| u'_f + u'_\infty - u'_b \| \right)}{\left( \| u'_f + u'_\infty - u'_b \| \right)}
\]

(19)

where \( K_g \) is the number of boundary sampling points, \( u_f \) and \( u_b \) are the velocity of fluid at the boundary sampling points before and after boundary treatment, respectively. \( u_\infty \) is the velocity of the background flow, and \( u_b \) is the velocity of the solid at the boundary sampling points. In the Kármán Vortex Street example, the solid is static and the velocity of the background flow is constant. Therefore, \( e \) can be simplified as

\[
e = \sum_{i=1}^{K_g} \frac{\| \gamma \| - \| \gamma' \|}{\| \gamma \|}.
\]

Fig 11 shows the ratio of the velocity elimination for the three cases. We can observe that our boundary treatment method can cancel out most of the slip and through velocities while dealing with regular obstacles. In the three cases, one disc achieves the highest accuracy, whose elimination ratio reaches approximately 83%. The accuracy of the airfoil is the lowest. We remark that our method is more suitable for dealing with the boundary of obstacles without sharp shapes.

**Vortex shedding behind a static ball.** Fig 12 indicates the realistic wakes behind a static ball in the rising smoke example. In this experiment, we emit a vortex circle every 10 time steps, consisting of 200 vortex particles. The vortex circle carries the tracer particles upward. When the static ball is passed, the vortex particles entering the ball are destroyed, and the collision point with the obstacle is the boundary sampling point. The radius of the ball is 0.5 unit lengths.
Fig 13 demonstrates the realistic wakes behind the static cylinders. We construct the thin smoke to form the words ‘PLOS One’ and let it pass through the cylinders. The radius of the cylinder is 0.8 unit lengths. In this example, the boundary sampling point is computed in advance like the 2D Karman vortex street example. The vortex particles entering the cylinder are destroyed. The simulation results show that our method can produce dispersive thin smoke.

**Vortex shedding behind moving objects.** The first column in Fig 14 depicts the turbulent wake formed by the moving ball passing through the static smoke. Unlike in the 2D Karman vortex street’s sampling manner, we place massive vortex particles regularly in the static smoke and treat the collision points between the vortex particles and the moving ball as the boundary sampling points. The vortex particles entering the obstacle are destroyed. This means that the number of vortex particles is constant in the simulation. The ball’s radius is 0.5 unit lengths, and the ball moves at a speed of 2 unit-lengths/s.
The second column in Fig 14 presents the turbulent wake formed by the moving and rotating bunny passing through the static smoke. In this example, the boundary sampling manner is the same as the moving ball. We use the signed distance field to calculate the bunny’s normal and detect the collision between the vortex particles and the bunny. In addition to the translation speed, we introduce a rotation speed to the bunny to prove that our method can handle complex boundary conditions. Fig 14 shows that the bunny produces more details than the ball. The bunny’s size is equivalent to that of the ball, the translation speed is 2 unit-lengths/s, and the rotation speed is 2 unit radians/s.

Performance

We parallelize most of our method’s steps using OpenMP and implement a GPU version of the nested grid method using CUDA, which is the most computationally expensive part of our method. Our examples are performed on a laptop with a 6-core 2.20 GHz CPU and a NVIDIA GeForce GTX 1060 graphics card. Table 1 lists the timing statistics of various smoke examples.
Conclusions

We presented a pure Lagrangian vortex method to produce rich details for smoke animation. Our method is efficient and suitable for parallelism on a GPU, thereby indicating its potential to be applied in real-time applications. This framework has two essential components. A nested grid is used to group the vortex particles and effectively reduce the computational cost of the BS law. This is validated by using the vortex ring example. Further, we proposed a novel vortex-generating method that supports dynamic boundary conditions for handling complex obstacles. We simulated a series of complex gas phenomena, such as the Kármán Vortex Street, colliding vortex rings, and interaction of smoke and obstacles.

Future work: The nested grid loses some accuracy when computing the velocity. We can improve this by dividing a direct summation area in the finest grid layer, while the other layers remain unchanged. Moreover, the number of generated vortex particles determines the linear equation’s size, making our method challenging in dealing with the interaction between smoke and large-scale obstacles. Reducing the linear equation’s size or splitting the equation into multiple sub-equations could be beneficial. Finally, the two-way fluid rigid body interaction is a prominent research direction for future work.

Supporting information

S1 Video. (MP4)
S1 Fig. Overview of our vortex method. (TIF)

Author Contributions

Funding acquisition: Hongxiang Ren.
Methodology: Rui Tao, Delong Wang.
Software: Rui Tao, Xiangen Bai.
Supervision: Hongxiang Ren.
Visualization: Rui Tao.

Table 1. Performance statistics of smoke simulations.

| Figures            | 2D Scenes                  | Vortex No. | Time(ms) |
|--------------------|----------------------------|------------|----------|
| Bottom in Fig 9    | Kármán Vortex Street       | 20000      | 90       |
| Top in Fig 10      | Flow passing 2 discs        | 40000      | 200      |
| Bottom in Fig 10   | Flow passing a airfoil      | 40000      | 210      |
| Figures            | 3D Scenes                  | Vortex No. | Time(ms) |
| Bottom in Fig 6    | Colliding vortex rings      | 40000      | 70       |
| Bottom in Fig 7    | Vortex ring                | 64000      | 80       |
| Fig 12             | Smoke passing ball         | 80000      | 230      |
| Fig 13             | Smoke passing cylinders     | 120000     | 610      |
| Right in Fig 14    | Ball passing smoke         | 50000      | 175      |
| Left in Fig 14     | Bunny passing smoke        | 50000      | 225      |

https://doi.org/10.1371/journal.pone.0269114.t001
Writing – original draft: Rui Tao.
Writing – review & editing: Delong Wang, Xiangen Bai.

References
1. Ghoniem A. F. and Ng K. K. Numerical study of the dynamics of a forced shear layer. Phys. Fluids, 30:706–721, 1987. https://doi.org/10.1063/1.866321
2. Francois Michel Pepin. Simulation of the flow past an impulsively started cylinder using a discrete vortex method. PhD thesis, California Institute of Technology, 1990.
3. Steffen Weißmann and Ulrich Pinkall. Filament-based smoke with vortex shedding and variational reconnection. In ACM SIGGRAPH 2010 papers, pages 1–12. Association for Computing Machinery, 2010.
4. Padilla M., Chern A., Knöppel F., Pinkall U., and Schröder P. On bubble rings and ink chandeliers. ACM Trans. Graph., 38(4), 2019. https://doi.org/10.1145/3306346.3322962
5. Ploumha ns P and Winckelmans G S. Vortex methods for high-resolution simulations of viscous flow past bluff bodies of general geometry. J. Comput. Phys., 165(2):354–406, 2000. https://doi.org/10.1006/jcph.2000.6614
6. Wu J. Z., Ma H. Y., and Zhou M. D. Vortical Flows. Springer, 2015.
7. Monaghan Joe J. Smoothed particle hydrodynamics. Annu Rev Astron Astr, 30(1):543–574, 1992. https://doi.org/10.1146/annurev.aa.30.090192.002551
8. Solenthaler B. and Pajarola R. Predictive-corrective incompressible sph. ACM Trans. Graph., 28:3, 2009. https://doi.org/10.1145/1531326.1531346
9. Zhang Xinxin and Bridson Robert. A pppm fast summation method for fluids and beyond. ACM Trans. Graph., 33(6):1–11, 2014.
10. Koumoutsakos Petros, Leonard Anthony, and Pepin F. Boundary conditions for viscous vortex methods. J. Comput. Phys., 113(1):52–61, 1994. https://doi.org/10.1016/0022-0396(94)90117-0
11. Ploumha ns Paul, Winckelmans G S, Salomon John K, Leonard Anthony, and Warren MS. Vortex methods for direct numerical simulation of three-dimensional bluff body flows: application to the sphere at re = 300, 500, and 1000. J. Comput. Phys., 178(2):427–463, 2002. https://doi.org/10.1006/jcph.2002.7035
12. Jos Stam. Stable fluids. In Proceedings of the 26th annual conference on Computer graphics and interactive techniques, pages 121–128, 1999.
13. ByungMoon Kim, Yingjie Liu, Ignacio Llamas, and Jaroslaw R Rossignac. Flowfixer: Using bfecc for fluid simulation. Technical report, Georgia Institute of Technology, 2005.
14. Selle Andrew, Fedkiw Ronald, Kim Byungmoon, Liu Yingjie, and Rossignac Jarek. An unconditionally stable maccormack method. J. Sci. Comput., 35(2-3):350–371, 2008. https://doi.org/10.1007/s10915-007-9166-4
15. Zehnder Jonas, Narain Rahul, and Thomaszewski Bernhard. An advection-reflection solver for detail-preserving fluid simulation. ACM Trans. Graph., 37(4):1–8, 2018. https://doi.org/10.1145/3197517.3201324
16. Qu Ziyin, Zhang Xinxin, Gao Ming, Jiang Chenfanfu, and Chen Baoquan. Efficient and conservative fluids using bidirectional mapping. ACM Trans. Graph., 38(4):1–12, 2019.
17. Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. Visual simulation of smoke. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pages 15–22, 2001.
18. He Shengfeng, Wong Hon-Cheng, Pang Wai-Man, and Wong Un-Hong. Real-time smoke simulation with improved turbulence by spatial adaptive vorticity confinement. Comput. Animal. Virt. W., 22(2-3):107–114, 2011. https://doi.org/10.1002/cav.408
19. He Shengfeng and Lau Rynson WH. Synthetic controllable turbulence using robust second vorticity confinement. Comput. Graph. Forum, 32(1):27–35, 2013. https://doi.org/10.1111/j.1467-8659.2012.03228.x
20. Ren Bo, Yan Xiao, Yang Tao, Li Chen-feng, Lin Ming C, and Hu Shi-min. Fast sph simulation for gaseous fluids. Visual Comput., 32(4):523–534, 2016. https://doi.org/10.1007/s00371-015-1086-y
21. Chorin A. J. Numerical study of slightly viscous flow. J. Fluid Mech., 57:785–796, 1973. https://doi.org/10.1017/S0022112073002016
22. Zhanpeng Huang, Guanghong Gong, and Liang Han. Vortex particle smoke simulation with an octree data structure. Comput. Animal. Virt. W., 27(1):14–23, 2016.
23. Liao Xiangyun, Si Weixin, Xu Biao, Yuan Zhiyong, Wang Qiong, and Heng Pheng-Ann. Filament-based realistic turbulent wake synthesis. *Comput. Animal. Virt. W.*, 28(3-4):e1754, 2017. https://doi.org/10.1002/cav.1754

24. Xiong Shiying and Yang Yue. The boundary-constraint method for constructing vortex-surface fields. *J. Comput. Phys.*, 339:31–45, 2017. https://doi.org/10.1016/j.jcp.2017.03.013

25. Xiong Shiying and Yang Yue. Construction of knotted vortex tubes with the writhe-dependent helicity. *Phys. Fluids*, 31(4):047101, 2019. https://doi.org/10.1063/1.5088015

26. Alfred Barnat and Nancy S Pollard. Smoke sheets for graph-structured vortex filaments. In *Proceedings of the 11th ACM SIGGRAPH/Eurographics conference on Computer Animation*, pages 77–86. Citeseer, 2012.

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

28. Pfaff Tobias, Thuerey Nils, and Gross Markus. Lagrangian vortex sheets for animating fluids. *ACM Trans. Graph.*, 31(4):1–8, 2012. https://doi.org/10.1145/2185520.2185608

29. Zhang Xinxin, Bridson Robert, and Greif Chen. Restoring the missing vorticity in advection-projection fluid solvers. *ACM Trans. Graph.*, 34(4):1–8, 2015. https://doi.org/10.1145/2766982

30. Greengard Leslie and Rokhlin Vladimir. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73(2):325–348, 1987. https://doi.org/10.1016/0021-9991(87)90140-9

31. Sang Il Park and Myoung Jun Kim. Vortex fluid for gaseous phenomena. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 261–270, 2005.

32. Vines M., Houston B., Lang J., and Lee W. Vortical inviscid flows with two-way solid-fluid coupling. *IEEE T. Vis. Comput. Gr.*, 20:303–315, 2013. https://doi.org/10.1109/TVCG.2013.95

33. Liao Xiangyun, Si Weixin, Yuan Zhiyong, Sun Hanqiu, Qin Jing, Wang Qiong, et al. Animating wall-bounded turbulent smoke via filament-mesh particle-particle method. *IEEE T. Vis. Comput. Gr.*, 24(3):1260–1273, 2017. https://doi.org/10.1109/TVCG.2017.2665551 PMID: 28186900

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

35. Günther Tobias and Theisel Holger. The state of the art in vortex extraction. *Comput. Graph. Forum*, 37(6):149–173, 2018. https://doi.org/10.1111/cgf.13319

36. Alexis Angelidis and Fabrice Neyret. Simulation of smoke based on vortex filament primitives. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 87–96, 2005.

37. Lim TT and Nickels TB. Instability and reconnection in the head-on collision of two vortex rings. *Nature*, 357(6375):225–227, 1992. https://doi.org/10.1038/357225a0

38. van Rees Wim M, Hussain Fazle, and Koumoutsakos Petros. Vortex tube reconnection at $re = 10^4$. *Phys. Fluids*, 24(7):075105, 2012. https://doi.org/10.1063/1.4731809