To appear in the ACM SIGGRAPH conference proceedings

Filament-based smoke with vortex shedding and variational reconnection

Steffen Weißmann Ulrich Pinkall
TU Berlin

Abstract

Simulating fluids based on vortex filaments is highly attractive for the creation of special effects because it gives artists full control over the simulation using familiar tools like curve editors or the scripted generation of new vortex filaments over time. Because filaments offer a very compact description of fluid flow, real time applications like games or virtual reality are also possible.

We present a complete model that includes moving obstacles with vortex shedding, all represented as filaments. Due to variational reconnection the long-time behavior of our method is excellent: Energy and momentum stay constant within reasonable bounds and computational complexity does not increase over time.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling

Keywords: fluid simulation, vortex filaments, vortex shedding, vortex reconnection, panel method

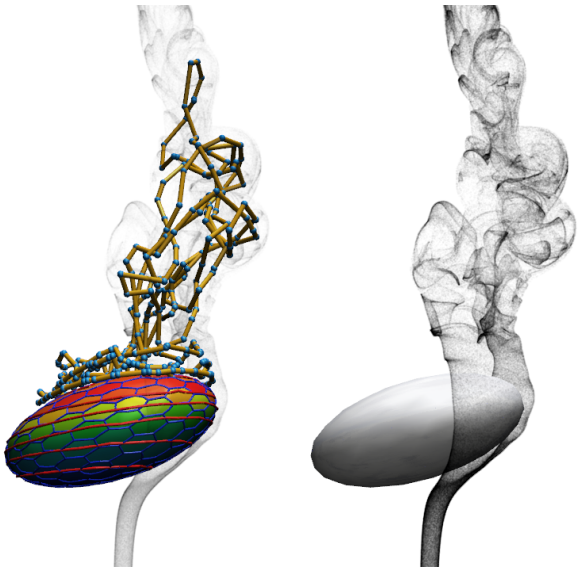


Figure 1: *Flow around an ellipsoid with vortex shedding.*

1 Introduction

Using vortex filaments as basic primitives for modeling fluid flow provides the most efficient method to capture the complexity of smoke with sparse data. This viewpoint was pioneered in [Angelidis and Neyret 2005; Angelidis et al. 2006]. In [Pinkall et al. 2007; Weißmann and Pinkall 2009] substantial improvements of the method were developed that yield an important increase in physical realism as well as a reduction of the computational costs. In particular, discrete differential geometry and integrable systems were used to obtain accurate simulations even with coarse polygonal filaments.

Vortex filaments provide a discretization of fluid dynamics where the vorticity field is concentrated along closed curves. Realistic smoke can be modeled using a small number of vortex filaments. The main reason is that in the real world all vorticity comes to life as two-dimensional vortex sheets in the boundary layers of obstacles. These vortex sheets curl up into complicated one-dimensional structures, the vortex filaments. Simulating these filaments directly provides an extremely efficient way to model the whole flow.

We extend existing methods for long time stability and efficiency, while we also include boundaries with vortex shedding (see Fig. 1). Our approach gives special effects artists full control over the simulation by modeling the geometry of the initial filaments. Obstacles can be handled as triangular meshes. No Eulerian grid is required. Our method is fast enough to simulate non-trivial scenarios in real time, with a level of detail that easily exceeds the resolution that can be achieved by grid-based simulations. In combination with GPU-accelerated particle advection it can be used for computer games and virtual environments.

1.1 Goals and contributions

Our goal is to develop an algorithm that allows vortex filament simulations that (a) are plausible and efficient enough to run in real time, (b) run stably over long simulation times, (c) can handle static and moving obstacles including vortex shedding at boundaries. We will validate our method on various scenarios including the simulation of real experiments.

Hamiltonian formulation for thick vortex filaments: Conservation of energy is essential for the long-time stability and plausibility of physical simulations. In Sec. 3 we give a formulation of filament dynamics based on a Hamiltonian system. Compared to [Weißmann and Pinkall 2009] we allow for filaments with different thickness and work throughout with the correct velocity field. The Hamiltonian of our system is indeed the kinetic energy. It is exactly preserved for smooth filaments.

Variational model of vortex reconnection: Vortex filaments have a strong tendency to develop *hairpins* [Chorin 1990; Chorin 1993], which leads to an exponential increase in time of the number of polygon edges needed in a numerical simulation. Chorin has shown that a simple model of reconnection and hairpin removal leads to accurate and efficient simulations of a vortex ring consisting of many weak closed filaments. This criterion can however lead to alternating reconnections. In Sec. 5 we propose a reconnection criterion based on a variational principle. The key idea is to reduce the filament length while staying as close as possible to the original flow. Our reconnections decrease a certain functional and thus cannot lead to alternating reconnections. At the same time our method improves the overall geometry of the filaments.

Boundary conditions and vortex shedding: Obstacles are important not only to restrict the fluid domain, but also as a source of vorticity (vortex shedding). In Sec. 4 we show how to include static or moving obstacles into the simulation, by computing *image vorticity* [Saffman 1992] as a vortex sheet on the obstacle. Our method to compute the boundary layer is related to the standard 3D panel method [Katz and Plotkin 2001], but our approach reduces the size of the resulting linear system by a factor of two. In addition, we represent the image vorticity as a collection of vortex filaments on the surface of the obstacle. Vortex shedding is modeled by releasing some of these filaments into the flow.

Explicit flux computation: The computation of normal flux induced from a polygonal vortex filament through another polygon is required both for obstacles and reconnection. Its explicit integration (given in App. A) is new and also of interest for the numerics of boundary integral equations [Sauter and Schwab 2004].

Validation: Vortex reconnection can be observed in experiments with real vortex filaments: Two obliquely colliding vortex rings can merge into one big filament that splits again into two separate rings [Lim 1989]. Head-on collision of two vortex rings can result in a reconnection that leads to many small vortex rings [Lim and Nickels 1992]. For validation we demonstrate how our method reproduces these real experiments (Fig. 8). We also simulate vortex shedding behind a sphere and demonstrate the long-time robustness of our method using a jet simulation.

2 Related work

Much work on 3D smoke simulations is based on Stam's *Stable Fluids* [Stam 1999], together with *vorticity confinement* [Fedkiw et al. 2001]. We will not review the entire literature here and instead refer to the course notes [Müller et al. 2008] and the references therein. Despite their ongoing success, grid-based methods share some fundamental problems: The whole fluid domain needs to be discretized, which requires knowledge of the overall behavior of the simulation in advance. Aliasing effects occur due to preferred directions along the grid axes. The large number of grid cells needed restricts the possible resolution, even when using sparse grids. Real-time applications are only possible with coarse grids.

The importance of vorticity for the believability and detailed structure of flows has long been recognized as evidenced by the body of work geared at ensuring that vorticity is not lost [Stam and Fiume 1993; Fedkiw et al. 2001; Neyret 2003; Selle et al. 2005; Kim et al. 2005; Elcott et al. 2007; Narain et al. 2008; Kim et al. 2008] or even added in a user controlled way [Park and Kim 2005; Pfaff et al. 2009]. Simplicial fluids [Elcott et al. 2007] preserve circulation (and thus vorticity) by construction. Still there is dissipation of energy which can be avoided using variational integrators for fluids [Mullen et al. 2009]. Nevertheless it is not possible to accurately resolve the one- or two-dimensional structures of vorticity created at boundary layers even with high-resolution meshes.

Mesh-free methods used in computer graphics are mostly based on SPH [Stam and Fiume 1995; Desbrun and Gascuel 1996; Müller et al. 2003; Becker and Teschner 2007; Yan et al. 2009] or vortex particles. Inviscid flow like smoke is challenging for SPH. Vortex particles [Park and Kim 2005] have difficulties in maintaining the solenoidal property of the vorticity field, which requires the vortex particles to align along closed loops. Vortex particles have also been used with great success in hybrid approaches [Selle et al. 2005; Chatelain et al. 2008; Pfaff et al. 2009].

The use of vortex filaments as basic primitives for modeling 3D smoke was pioneered by Angelidis and Neyret [2005]. A more physically based method was introduced in [Weißmann and Pinkall 2009] in combination with a discrete integrable system that compensates for discretization errors inherent to the polygonal vortex filament model. The excessive increase of filament length makes long time simulations impossible, which has been addressed in [Angelidis et al. 2006] by filtering high frequencies from the filament geometries. A physically motivated approach was introduced by Chorin [Chorin 1990; Chorin 1993]: Hairpin removal and filament reconnection. Chorin's method was also used in [Bernard 2006; Marzouk and Ghoniem 2007; Bernard 2009].

Real-time methods for 3D smoke simulation are mostly based on Semi-Lagrangian methods using GPU techniques [Crane et al. 2007; Kim 2008; Yang 2009]. Reductions to 2D [Rasmussen et al. 2003; Krüger and Westermann 2005] have also been used to improve performance. Model reduction [Treuille et al. 2006] can significantly speed up simulations, but it requires extensive precomputations as well as giant storage.

Vortex shedding from boundary layers has been modeled using vortex particles: [Park and Kim 2005] compute the boundary layer using the standard 3D panel method [Katz and Plotkin 2001]. [Pfaff et al. 2009] determine the boundary layer directly from the laminar flow computed using a grid. Filaments on the other hand are better adapted than vortex particles to the coherent structures that emerge anyway during vortex shedding. Therefore they can yield realistic and detailed results with less computational effort.

3 Filament dynamics

Filament-based fluid simulation makes use of the vorticity formulation of an ideal fluid: The time-dependent velocity field v of the fluid is determined by its vorticity field $\omega = \operatorname{curl} v$. Instead of computing the evolution of v directly, one computes the evolution of ω. Given ω, v can be evaluated at arbitrary points. The vorticity field ω itself is advected by the velocity field v that it generates.

Realistic flows can be approximated very accurately by *vortex filaments*, closed curves along which the vorticity field is concentrated. Instead of having to track the evolution of a vorticity field ω on the whole space, we only need to track the evolution of some closed space curves. Therefore we do not need to discretize the whole fluid domain, but just the space curves defining the vorticity field. As discretizations of the vortex filaments we use polygons.

The basic algorithm of filament-based fluid simulation takes an initial configuration of vortex filaments γ and computes the time evolution of this configuration for time step Δt. Each of these configurations $\gamma|_{t_0}, \gamma|_{t_1}, \ldots$ defines the velocity field $v|_{t_i}$ at time t_i. This sequence is used to advect a set of marker particles.

We emphasize that our smooth filament model is a Hamiltonian system. This fact implies for free the conservation of energy and momentum, which is extremely important for physical realism and long-time robustness of the method.

3.1 Mathematical description

The velocity v of an incompressible fluid in $\mathbb{R}^3$ (which is at rest near infinity) can be uniquely reconstructed from its vorticity field $\omega = \operatorname{curl} v$ by the *Biot-Savart* formula:

$$v_\omega(x) = \frac{1}{4\pi} \int_{\mathbb{R}^3} \omega(z) \times \frac{x - z}{|x - z|^3} \, dz. \tag{1}$$

The Euler equations of ideal fluid motion just say that ω is forward advected by the velocity field it generates.

Let us now look at the situation where the vorticity ω_γ of the fluid is concentrated on a finite collection γ of closed oriented curves

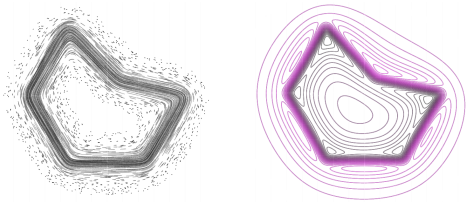


Figure 2: *Vorticity and velocity field of a planar polygonal vortex filament γ, smoothed by $S_{0.1}$. Left: Trajectories of $\omega = S_{0.1}\omega_\gamma$. Right: Level lines of $|v|$ of the generated velocity field $v = S_{0.1}v_\gamma$.*

γ_i (the *vortex filaments*) in a delta-function like manner: Then the generated velocity field (1) reduces to a sum of line integrals along the filaments:

$$v_\gamma(x) = \sum_i \frac{\Gamma_i}{4\pi} \oint \gamma_i'(s) \times \frac{x - \gamma_i(s)}{|x - \gamma_i(s)|^3}\, ds.$$

Here Γ_i denotes the strength of the filament γ_i. It is the flux of vorticity through any cross section of a tube surrounding the filament. Γ_i is constant along the filament, since the vorticity field is divergence free. It is also constant in time by Kelvin's theorem, because it is the circulation of velocity around the filament ($\Gamma_i = \oint \langle v, \eta_i' \rangle$, η_i a small loop around the tube).

A naive application of the general fact that vorticity is advected by the velocity would then say that the filaments γ_i evolve by evaluating the velocity field v_γ on the filament points:

$$\dot\gamma_j(s) = v_\gamma(\gamma_j(s)).$$

This is however problematic since v_γ is infinite on the curves γ_i. Any realistic model therefore has to consider filaments of a small non-zero thickness instead of infinitely sharp ones. We achieve this by spreading out the delta-function like vorticity by a suitable smoothing operator: For any (possibly vector-valued) function f and parameter a define a smoothed version $S_a f$ by the convolution

$$(S_a f)(x) = \frac{3a^2}{4\pi} \int_{\mathbb{R}^3} \frac{f(y)}{\sqrt{a^2 + |x-y|^2}^5}\, dy.$$

Roughly speaking, a measures the distance over which delta functions are spread out by S_a. Fig. 2 illustrates the effect of S_a on a sharp planar polygonal vortex filament. For our discussion it is important to know that the smoothing operator S_a has a square root $\sqrt{S_a}$ that is also given by convolution with a suitable kernel, see [Sato 1999, Eq. (8.23)]. We do not have an explicit expression for $\sqrt{S_a}$ but later we will provide a good approximation.

Applying S_a to the velocity field v_γ we obtain

$$(S_a\, v_\gamma)(x) = \sum_i \frac{\Gamma_i}{4\pi} \oint \gamma_i'(\tilde s) \times \frac{x - \gamma_i(\tilde s)}{\sqrt{a^2 + |x - \gamma_i(\tilde s)|^2}^3}\, d\tilde s, \quad (2)$$

which is known as the *Rosenhead-Moore* formula. It turns out that the corresponding evolution equation for filaments

$$\dot\gamma_j(s) = \sum_i \frac{\Gamma_i}{4\pi} \oint \gamma_i'(\tilde s) \times \frac{\gamma_j(s) - \gamma_i(\tilde s)}{\sqrt{a^2 + |\gamma_j(s) - \gamma_i(\tilde s)|^2}^3}\, d\tilde s \quad (3)$$

is an excellent model for filament-based fluid simulation. It models the evolution of a vorticity field that is obtained from smoothing sharp filaments with $\sqrt{S_a}$. This looks surprising at a first glance (why $\sqrt{S_a}$ instead of S_a?), but in fact the advection of filaments involves two smoothing operations: First the singular vector field generated by the infinitely sharp filaments is smoothed. Then this vector field is averaged around filament points in order to approximate the advection of the smoothed vorticity. The total effect of both $\sqrt{S_a}$ smoothing operations can be combined into smoothing once with S_a.

Note however that a fluid evolving purely under Euler's equation cannot be represented exactly by a filament model: Any initially radially symmetric vorticity around the filaments will deform and lose its symmetry. Thus it can no longer be represented by filaments.

Nevertheless, system (3) evolves almost as under Euler's equation, but with the additional constraint that the vorticity field is always obtained from smoothing sharp filaments with $\sqrt{S_a}$. Even though it is a slight modification of the original system, it still conserves kinetic energy and momentum of the velocity field.

The physical interpretation of our filament model is obtained from inspection of the underlying Hamiltonian system, given in [Marsden and Weinstein 1983; Pinkall et al. 2007]: From the fact that S_a has a square root $\sqrt{S_a}$ given as a convolution (thus being self-adjoint), the Hamiltonian

$$H_a(\gamma) = \int_{\mathbb{R}^3} \langle S_a v_\gamma, v_\gamma \rangle = \int_{\mathbb{R}^3} \langle \sqrt{S_a} v_\gamma, \sqrt{S_a} v_\gamma \rangle$$

turns out to be in fact the kinetic energy $T = \int_{\mathbb{R}^3} |\sqrt{S_a} v_\gamma|^2$ of the velocity field $\sqrt{S_a} v_\gamma$. As a Hamiltonian system, both the kinetic energy T and the hydrodynamic momentum

$$p(\gamma) = \frac{1}{2} \sum_i \Gamma_i \oint \gamma_i \times \gamma_i'$$

are exactly preserved. The hydrodynamic momentum is proportional to the standard definition of momentum $\int_{\mathbb{R}^3} \sqrt{S_a} v_\gamma$, see [Saffman 1992].

H_a can also be computed using only integrals along filaments instead of the whole space:

$$H_a(\gamma) = \sum_{i,j} \frac{\Gamma_i \Gamma_j}{8\pi} \oint\oint \frac{\langle \gamma_i'(s), \gamma_j'(t) \rangle}{\sqrt{a^2 + |\gamma_i(s) - \gamma_j(t)|^2}}\, ds\, dt.$$

This reveals its geometric interpretation, which is particularly important for obstacles (Sec. 4):

$$H_a(\gamma) = \frac{1}{2} \sum_i \Gamma_i \operatorname{flux}_{\gamma_i}(S_a\, v_\gamma).$$

Here $\operatorname{flux}_\eta(u)$ denotes the normal flux of the vector field u through a disc with boundary η. In App. A we show how to compute $H_a(\gamma)$ explicitly in the case of polygonal filaments.

Two problems remain to be solved: First, we do not have a good algorithm to compute $\sqrt{S_a}\, v_\gamma$, which would be the correct velocity field for advecting point particles. If we had used the nicely computable field $S_a\, v_\gamma$ instead, we would have computed the velocity of blurred particles (obtained by smoothing a delta function by $\sqrt{S_a}$). Second, the assumptions that all filaments are spread out by the same amount seems restrictive. One would like to be able to handle filaments with different thickness parameters a_i.

As a solution to both problems we propose to approximate S_a by the convolution $\tilde S_a$ with a suitable heat kernel

$$K_a(x,y) = \frac{1}{(\sqrt{\pi}\lambda a)^3} e^{-|x-y|^2/(\lambda a)^2}.$$

The precise value of the constant λ does not matter for our argument, nor is it important what type of approximation we use. What matters

is only the way K_a depends on a, which is obviously correct for dimensional reasons. Then, using $S_a \approx \tilde{S}_a$ and the well known formula ($*$ denotes the convolution product)

$$\tilde{S}_a * \tilde{S}_b = \tilde{S}_{\sqrt{a^2+b^2}}$$

we arrive at the approximations

$$\begin{aligned} S_a * S_b &\approx S_{\sqrt{a^2+b^2}}, \\ \sqrt{S_a} &\approx S_{\frac{a}{\sqrt{2}}}, \\ \sqrt{S_a} * \sqrt{S_b} &\approx S_{\sqrt{\frac{a^2+b^2}{2}}}. \end{aligned} \tag{4}$$

The proposed solution to the mentioned problems is therefore:

- Point particles are advected with the velocity field (2) with a^2 replaced by $a_i^2/2$.
- Filaments evolve according to the evolution equation (3) with a^2 replaced by $(a_i^2 + a_j^2)/2$.

This modification does not affect the excellent properties of the system. It is still Hamiltonian and the kinetic energy is preserved up to the error made by our approximation for $\sqrt{S_{a_i}} * \sqrt{S_{a_j}}$.

3.2 Numerical formulation

The basic numerical method for filament-based fluid simulation that we are using was given in [Weißmann and Pinkall 2009]: We replace the smooth filaments by polygons, then we advect each polygon vertex x_{jk} (the k-th vertex of the polygon γ_j) according to the evolution equation (3) for filaments:

$$\dot{x}_{jk} = (S_{a_{ij}} v_\gamma)(x_{jk}). \tag{5}$$

Here $S_{a_{ij}} v_\gamma$ denotes the velocity field (2) with a^2 replaced by $(a_i^2 + a_j^2)/2$ as described above. $S_{a_{ij}} v_\gamma$ is given as a sum of integrals over the closed polygons γ_i. Each integral can be evaluated explicitly as a sum over the polygon edges. The explicit formula is given in [Weißmann and Pinkall 2009]. The evolution equation for the filament vertices (5) is a time-independent first order ODE, which can be solved for given time step Δt with an explicit ODE solver. Given an initial configuration of polygonal vortex filaments $\gamma|_{t_0}$, we obtain the next configuration $\gamma|_{t_1}, t_1 = t_0 + \Delta t$ in the time evolution of γ by numerical integration of the initial value problem

$$x|_{t_1} = x|_{t_0} + \int_0^{\Delta t} (S_{a_{ij}} v_\gamma)(x)\, d\tau, \quad x|_{\tau=0} = x|_{t_0}.$$

Here we have used x as short hand for x_{jk}. However this evolution is known to lead to discretization errors that stem from the fact that adjacent edges do not contribute to the velocity field at a vertex. This effect can be compensated using the *doubly-discrete smoke ring flow*, a correction step that is applied before each numerical integration. This step yields an important increase in the physical realism of the simulation, especially for coarse polygons. The mathematical description can be found in [Pinkall et al. 2007], the concrete implementation of the method is given in [Weißmann and Pinkall 2009]. Iteration of these two steps finally gives the evolution of the discrete vortex filaments.

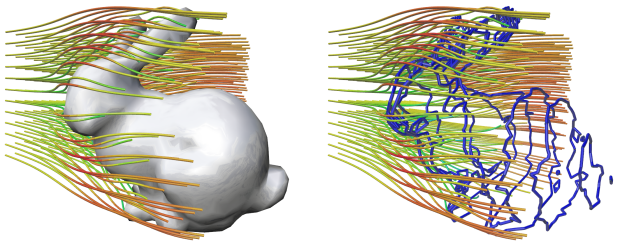


Figure 3: *Vortex filaments that force a constant background flow to be tangential to the bunny. Inside the velocity is almost zero. The mesh has* 678 *vertices and* 1352 *facets.*

4 Obstacles

Filament-based fluid simulation is not confined to any box containing a grid but takes place in unbounded space. This advantage is at the same time a limitation since it is not clear how to treat obstacles (walls, terrain, moving characters ...). Here we explain how to incorporate arbitrary moving obstacles. In fact there is a way to replace obstacles dynamically by certain sets of closed vortex filaments placed on the surface of the obstacles, as shown in Fig. 3. These vortex filaments are given as level sets of a scalar function on the obstacle surface. This makes our approach coherent in the sense that the whole simulation is purely driven by filaments. Moreover, we gain the possibility to simulate vortex shedding by letting some of these filaments diffuse away from the obstacles into the surrounding flow.

4.1 Mathematical model

Consider an obstacle with smooth boundary M and a fluid velocity field v that has a scalar potential near M. For instance, v can include velocity generated by vortex filaments but also background flow and rigid motion of the obstacle. Now we can construct a *vortex sheet* on M in such a way that the velocity field v_M generated by the vortex sheet compensates the normal component of v: The superimposition of v and v_M makes the fluid flow around the obstacle, i.e., $v + v_M$ is tangent to M.

This vortex sheet is defined by a scalar function f on M and the velocity field that it generates is

$$v_M(p) = \frac{1}{4\pi} \int_M (N \times \operatorname{grad} f)(q) \times \frac{p-q}{|p-q|^3}\, dq. \tag{6}$$

Here dq denotes the area element of M. Note that $\operatorname{grad} f$ is tangent to M and therefore $N \times \operatorname{grad} f$ is $\operatorname{grad} f$ rotated by 90° around N.

The function f on M is determined uniquely (up to a constant) by the condition that the normal component of the velocity field v_M given by Eq. (6) cancels the normal component of v:

$$\langle v(q), N(q) \rangle + \langle v_M(q), N(q) \rangle = 0. \tag{7}$$

The vortex sheet on M obtained in this way is called the *image vorticity* [Saffman 1992].

Comparison of the sheet velocity field (6) with the Biot-Savart law (1) reveals that the vorticity ω_M of the sheet is concentrated along the vector field $\kappa = N \times \operatorname{grad} f$ on M. The field lines of κ are precisely the level lines of f, since $\operatorname{grad} f$ is perpendicular to these level lines. From this observation it is intuitively clear that the vortex sheet can be approximated by a choice of certain level lines as filaments with appropriate strength. We will now deduce how to choose these levels and their strengths.

Near a point $q \in M$ where $\operatorname{grad} f$ does not vanish one can introduce coordinates (s, f) where f itself is one of the coordinate functions and s provides an arclength parameter on each level line of f. Then in these coordinates

$$\frac{\partial q}{\partial s} = \frac{N \times \operatorname{grad} f}{|\operatorname{grad} f|}, \quad ds\, df = |\operatorname{grad} f|\, dq$$

and the integral in Eq. (6) can be expressed locally as

$$\int \frac{\partial q}{\partial s} \times (p-q)/|p-q|^3 ds\, df.$$

This local information leads to the following global picture: Let y be a regular value of f. Then the preimage $f^{-1}(y)$ is the union of finitely many closed level lines $\gamma_1, \ldots, \gamma_n$ on M. Let us denote by v_y the velocity field generated by $\gamma_1, \ldots, \gamma_n$ considered as vortex filaments of unit strength. Then v_M can also be computed as

$$v_M = \int_{-\infty}^{\infty} v_y\, dy.$$

This integral can now be conveniently approximated by summing up the values of v_y at equidistant samples with distance $\Gamma > 0$:

$$v_M \approx \sum_{i=-\infty}^{\infty} \Gamma\, v_{i\Gamma} = v_{\gamma_M}.$$

Here $\Gamma\, v_{i\Gamma}$ denotes the velocity field generated by the preimage $f^{-1}(i\Gamma)$ as vortex filaments of strength Γ. Most of the preimages will be empty, but for $i\Gamma$ in the range of f it is a finite set of closed level lines on M. Thus we obtain a finite set of vortex filaments with strength Γ generating a velocity field v_{γ_M} that approximates v_M. In the limit $\Gamma \to 0$ we obtain the original velocity field v_M.

Depending on the value chosen for Γ we either obtain a larger number of weak filaments (for small Γ) or a smaller number of strong filaments (for larger Γ). The quality of the approximation $v_M \approx v_{\gamma_M}$ increases with smaller Γ. In our application Γ needs to be chosen such that the approximation is sufficiently accurate while the number of filaments is small.

4.2 Polygonal discretization

For the discretization we assume that the surface of the obstacle is given as a polygonal mesh M. First we approximate f by a function that it is constant on the facets ϕ_i of M. Consider such a function f with values f_i on the facets: The gradient of f is concentrated in a singular way between adjacent facets, i.e., on the edge graph of the mesh. Thus, by a 90° rotation, the vorticity of the sheet is concentrated along the edges, and the strength of an edge e_{ij} (between the facets ϕ_i and ϕ_j) is the difference $f_j - f_i$. Therefore, the vortex sheet in this case is obtained by superimposing the boundary polygons η_i of the facets ϕ_i as vortex filaments of strength f_i. At each vertex the sum of the strengths satisfies Kirchhoff's law, i.e., incoming intensity equals outgoing intensity. This reflects the fact that the vorticity field is divergence free. Thus we have discretized the vorticity in the boundary layer of M as a "divergence free vector field" concentrated on the edge graph of M. This fits in with the philosophy of *discrete exterior calculus* [Desbrun et al. 2008].

Now we smooth the singular vorticity on the edge graph with thickness parameter a. This corresponds to a non-zero thickness of the boundary layer. Eq. (7) turns into a linear system, stating that the total normal flux through each ϕ_i is zero:

$$\sum_j f_j \underbrace{\operatorname{flux}_{\eta_i}(S_a v_{\eta_j})}_{a_{ij}} = -\underbrace{\operatorname{flux}_{\eta_i}(v)}_{g_i}. \tag{8}$$

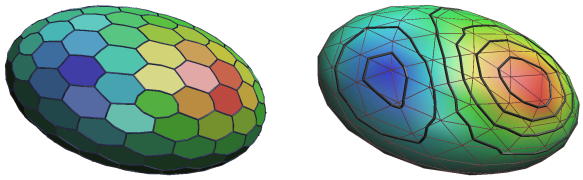


Figure 4: *Filament approximation of vortex sheets: The piecewise constant f on the dual mesh gives vorticity on the edge graph (left). f linearly interpolated on the primal triangle mesh yields vorticity along level lines (right).*

The matrix entry a_{ij} is the normal flux through ϕ_i of the velocity field induced by the boundary polygon η_j as a unit strength vortex filament of thickness a (App. A). Since each edge participates with opposite orientation in two faces, turning on all faces with the same vorticity has no effect at all. So the vector $\{1, \ldots, 1\}$ lies in the kernel of the matrix $A = (a_{ij})$, and in fact it spans the kernel for connected obstacles. By Stokes' theorem the total flux $\sum_j a_{i,j}$ vanishes, so the range of A is the orthogonal complement of $\{1, \ldots, 1\}$. Fortunately, for the same reason also the right hand of Eq. (8) lies in this space, so the linear system has a unique solution.

For a static obstacle the matrix A can be precomputed and prefactored, so the main computational cost during the simulation is the repeated evaluation of the right hand side y in Eq. (8).

For a single rigidly moving obstacle the only change is that we have to add to the right hand side g_i the flux through ϕ_i resulting from the rigid motion (App. B). This adds only little to the total cost. On the other hand, for several obstacles moving independently, the off-diagonal blocks in the matrix A correspond to the interaction of different obstacles. These need to be recomputed continuously.

Note that the flow we generate is not perfectly tangent to the original triangle mesh, but the net flux of the flow through each face vanishes.

The piecewise constant f on the facets ϕ_i determines a vortex sheet whose vorticity is concentrated on the edge graph of M. To approximate such a sheet by filaments as for smooth obstacles, we need a smoothed version $\tilde{f}$ of f. We obtain $\tilde{f}$ by linear interpolation of a function $\hat{f}$ that has values on the vertices of M. This $\hat{f}$ is obtained directly from the dual mesh M^* of M: Compute f as described above for the dual facets ϕ_i^* and take it as a function $\hat{f}$ on the primal vertices. If M is a triangle mesh (which we assume), we can interpolate $\hat{f}$ to a function $\tilde{f}$ that is linear on the triangles. The level sets $\tilde{f}^{-1}(i\Gamma)$ are then polygons lying on M, as shown in Fig. 4.

For convincing results the distance between subsequent levels should match the sheet thickness a. When the distance between subsequent levels is too big, the flow does not follow the obstacle surface properly, wavy motion along the surface occurs. The distance depends on both the velocity field v and the level spacing Γ. Depending on the scenario Γ has to be chosen properly.

4.3 Vortex shedding

Vortex shedding occurs when the boundary layer diffuses away from the obstacle due to viscosity and is subsequently swept along with the fluid. Even in the limit of zero viscosity this effect cannot be discarded. It is well-known that the shed vortex sheets quickly roll up into one-dimensional core structures [Greengard and Anderson 1988; Rockliff et al. 2000]. Thus separation effectively happens in chunks of thick vortex filaments. This is in fact the key reason why filament-based smoke simulation is so strikingly effective in modeling natural phenomena.

The filament discretization of vortex sheets enables us to directly model vortex shedding: By releasing filaments into the flow. After a fixed time delay, one or more vortex filaments of γ_M are added to the set of filaments γ. Before releasing a filament, we apply an offset of length $a/2$ in normal direction of the obstacle.

Our experiments show that the behavior of the simulation is surprisingly indifferent to the particular choice of filaments we choose to release into the flow. This is because the released filaments approximately slide downstream along the obstacle, taking successively the position of other levels we could have chosen. In the beginning of a simulation, the released filaments start to substantially move away from the obstacle only near downstream stagnation points. Later on the influence of the already shed filaments makes newly shed filaments depart from the obstacle at also earlier stages. As long as the released filaments enclose the major downstream stagnation points (local minima of the function $\tilde{f}$) we obtain realistic wakes behind the obstacles. In practice we choose all filaments that are closest to the local minima of $\tilde{f}$.

5 Variational reconnection

As has been observed by Chorin [1990; 1993], filament-based modeling of fluids runs into a fundamental problem: nearby portions of filaments that are approximately anti-parallel attract each other and form long stretches where two oppositely oriented filament pieces nearly coincide with opposite orientation. The effects of these portions of the filaments on the overall flow approximately cancel out, so they could safely be removed from the computation. The problem is made worse by the fact that in the long run these unnecessary double strands of filaments get stretched and convoluted by the flow which leads to an exponential increase in the number of polygon edges needed to capture the geometric complexity of the filaments.

The solution to this problem is to eliminate almost anti-parallel nearby portions of filaments by changing the filament topology. This process is called *vortex reconnection*. In his seminal papers on this subject Chorin [1990; 1993] used many weak filaments to model a fluid with spread-out vorticity. He decides whether to reconnect a pair of polygon edges using a geometric criterion based on the distance between the edges and their directions. He only allows one reconnection per filament and time step since iterative computation of all reconnections is not guaranteed to terminate: Back and forth reconnection is possible.

We propose a variational approach: Given a configuration γ of filaments that need reconnection we determine a new configuration η optimally matching the two following objectives:

- The velocity field generated by η should be as close as possible to the one generated by γ.
- The total length of η should be as small as possible.

Let us look at the situation where two of the filaments γ_i get close to each other in such a way that reconnection is desirable. Then both filaments necessarily must have the same strength Γ, otherwise the construction would lead to a vorticity *graph* (like on the edge graph of an obstacle). Moreover, since filaments with a thickness varying along the filament are not an option for us, we assume that both filaments have the same thickness a. For a configuration with different thicknesses and strengths reconnections are computed for each subset of filaments with same strength and thickness. The reconnection could also take place between different portions of the same filament, so for simplicity we think of the two filaments as a possibly disconnected single filament γ_i. If we change γ_i into a new configuration and keep all the other filaments not participating in the reconnection the same, the difference in the generated velocity field will be just the velocity field generated by $\eta_i - \gamma_i$. Here $-\gamma_i$ denotes the filament γ_i with the reversed orientation. This means that we can ignore the other filaments and assume without loss of generality that there is only one filament (possibly having two connected components). So henceforth we drop the index i. As a measure of closeness $d(\eta, \gamma)$ between the velocity fields generated by the two configurations η and γ we use the L^2-norm of the velocity field generated by $\eta - \gamma$:

$$d^2(\eta, \gamma) = \int_{\mathbb{R}^3} |\sqrt{S_a}\, v_{\eta-\gamma}|^2.$$

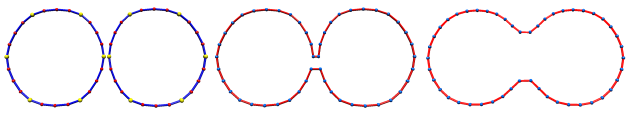


Figure 5: *Variational reconnection. Two nearby filaments are reconnected to one single filament. Variation of the reconnected filament according to* grad F *significantly improves the geometry.*

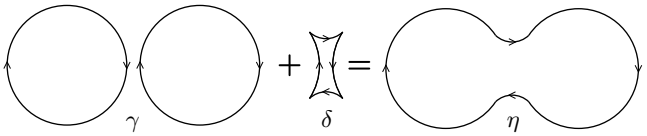
Denoting the length of η by $L(\eta)$ we then want to minimize

$$F(\eta) = \lambda\, \Gamma\, L(\eta) + d(\eta, \gamma). \tag{9}$$

Here λ is a constant parameter that still is to be determined. Setting λ to zero would lead to the original configuration γ as the absolute minimum of the functional (9). $\lambda > 0$ means that we try to reduce the length of η while still trying to remain close to γ. The use of the non-smooth functional d (instead of d^2) comes from the fact that we want to model discontinuous changes. The situation is similar to dry friction, where sliding down a slope only occurs when it is steep enough.

A further use of the functional (9) can be to improve the filament geometry after reconnection by numerically minimizing F to mend the geometric artifacts created by connecting polygon points in a different way. This is shown in Fig. 5. We see this as an optional step in the algorithm that can be omitted when simulation speed is the main issue.

Our main use of the Functional F is that it provides a unified criterion for deciding whether or not reconnection or hairpin removal should be performed between portions of a (possibly disconnected) filament. The goal is to reconnect whenever reconnection results in a lower value of the functional F, i.e., $F(\eta) < F(\gamma)$. The velocity field induced by the difference of the two configurations $\eta - \gamma$ is equivalent to the field induced by a single closed filament δ:



Let us denote the four segments of δ by δ_t, δ_b, δ_l, δ_r (top, bottom, left, right) and compute

$$\begin{aligned} F(\eta) - F(\gamma) = & \\ &= \lambda\Gamma(L(\eta) - L(\gamma)) + \sqrt{\int_{\mathbb{R}^3} |\sqrt{S_a}\, v_\delta|^2} \\ &= \lambda\Gamma(L(\delta_t) + L(\delta_b) - L(\delta_l) - L(\delta_r)) + \Gamma\sqrt{\mathrm{flux}_\delta(S_a v_\delta)}. \end{aligned}$$

Thus we reconnect whenever

$$L(\delta_t) + L(\delta_b) - L(\delta_l) - L(\delta_r) + \sqrt{\mathrm{flux}_\delta(S_a v_\delta)}/\lambda < 0. \tag{10}$$

Figure 6: *Simulation of a jet. Without reconnection we end up with 2560 edges after 600 simulation steps in contrast to 882 edges with reconnection. Note the almost identical structure of the particle evolutions with and without reconnections.*

This criterion provides a unified approach both to reconnection and to hairpin removal. For polygonal filaments this specializes to the following: For two edges $e_l = (x_1, x_2)$, $e_r = (x_3, x_4)$, the corresponding filament δ is the quadrilateral with vertices x_1, x_2, x_3, x_4. We reconnect the two edges when Eq. (10) is satisfied.

In the special case that two adjacent edges (with zero or one intermediate edges) are reconnected, one obtains a degenerated filament consisting of one or two vertices. Such a filament does not generate any velocity and is therefore discarded. The reconnection is then in fact a *hairpin removal* [Chorin 1993].

Note that our criterion implies that reconnection can only occur when the distance between the two edges is smaller than the edge length. Thus, for short edges (shorter than a), one has to consider not only single edges but polygon segments consisting of several consecutive edges. We avoid these additional calculations by maintaining a roughly uniform edge length using adaptive subdivision of the polygons.

Although reconnection does change energy and momentum, our experiments show that in typical situations these changes occur in a way that is sufficiently random to not cause any systematic drift. Visually the results are very similar to simulations without reconnection, see Fig. 6.

6 Implementation

The central task of filament-based fluid simulation is to compute the evolution of an initial configuration of filaments. In each step, the filament configuration defines a vector field on $\mathbb{R}^3$ via the Biot-Savart law (2). The evolution thus gives a time-dependent vector field which is used to advect marker particles. For our implementation we use the basic simulation method described in Sec. 3.2 and extend it with our enhancements for obstacles, vortex shedding and reconnection. Each simulation step of the filament evolution consists of the sub steps:

(a) Apply doubly-discrete smoke ring evolution to compensate discretization errors [Weißmann and Pinkall 2009].

(b) Advect filaments according to the evolution equation (11) for time step Δt.

(c) Add shed vortex filaments from obstacles.

(d) Drop edges that are too short and subdivide long edges.

(e) Reconnect all edge pairs that fulfill the criterion (10).

(f) Drop degenerate filaments (with less than 3 vertices).

Edge removal is realized by collapsing short edges to their center point. For subdivision we use cubic interpolation. To avoid comparing all edge pairs in (e) we use spatial hashing.

6.1 Filament advection with obstacles

Filament advection is a first-order ODE $\dot{x} = F(t, x)$, where t is time and x the vector of filament vertices, compare with Sec. 3.2. In case of a moving obstacle the ODE is *time-dependent*. F is evaluated as follows: t defines the current obstacle pose, the filament vertex vector x defines the current filament set γ. Together with obstacle motion and possibly a background flow v_B the filaments γ determine a set of image vortex filaments γ_M on the obstacle. F is then given as the background flow v_B combined with the velocity field generated by the filaments $\gamma \cup \gamma_M$:

$$\dot{x}_i = v_B(x_i) + (S_{a_{ij}} v_{\gamma \cup \gamma_M})(x_i). \tag{11}$$

Evaluating the right-hand side of Eq. (11) amounts to

(a) compute image vorticity filaments γ_M on the obstacle,

(b) evaluate the background flow v_B and the velocity field induced by $\gamma \cup \gamma_M$ at the vertex positions x_i, see Eq. (5).

The filaments γ_M are obtained as follows:

- For the dual faces of the triangulated obstacle surface M compute the normal flux g from the velocity field induced by the vortex filaments γ, from background flow, and from rigid motion of the obstacle.
- Solve the linear system $A\hat{f} = g$. A is the precomputed obstacle matrix for the dual mesh M^*. To solve the linear system we replace the matrix $A = (a_{ij})$ by $\tilde{A} = (a_{ij} + \epsilon)$, which has full rank and yields the solution for the original system.
- Take $\hat{f}$ as a function on the primal vertices and denote by f its piecewise linear interpolation on the triangles. Determine all level sets $f^{-1}(i\Gamma)$ with $i\Gamma \in [f_{min}, f_{max}]$. Each level set consists of one or more closed polygons lying on M.
- Add each polygon as a vortex filament with strength Γ to the set of image vorticity filaments γ_M.

To compute the normal flux of velocity (required for the entries of the obstacle matrix A as well as for the right hand side g), we use analytical evaluation for nearby filaments and one point quadrature for far away filaments (see App. A). Rigid motion flux is computed as described in App. B.

To determine the closed level sets we represent the triangle mesh M using a half-edge data structure [Weiler 1985].

To advect the filament vertices we solve the ODE (11) for time step Δt using either standard Runge-Kutta with fixed step size (RK2 or RK4) or a more sophisticated explicit solver when high accuracy is required. For instance the colliding vortex filaments shown in Sec. 7 were computed using a *Dormand-Prince 5/4* solver [Dormand and Prince 1980].

6.2 Parallelization

Evaluating the velocity field (and also the flux) becomes expensive for a large number of filament edges. Nevertheless we achieve excellent performance even for complex scenarios via parallelization. Our implementation uses jReality [Weißmann et al. 2009] for rendering and interaction. Particle advection via a GLSL shader is integrated into the rendering pipeline. After each simulation step the current set of filament edges (from γ and γ_M) is transferred to

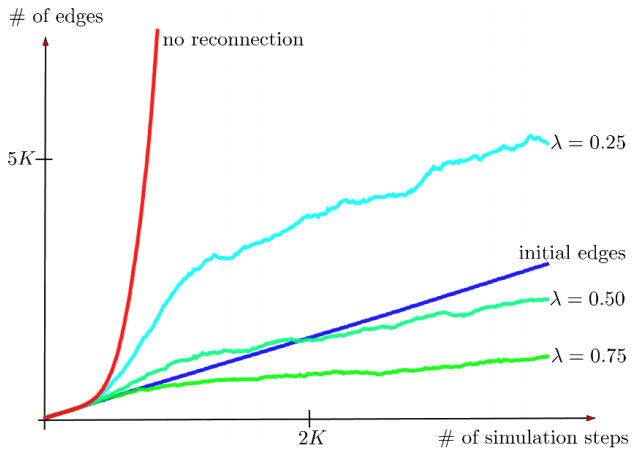


Figure 7: *Number of edges used to simulate a jet (Fig. 6). Circular filaments are added at a fixed rate at the orifice, thus the total number of edges added to the simulation grows linearly (blue). Without reconnection, the number of edges grows exponentially fast due to subdivision (red). With variational reconnection, the number of edges grows slowly (depending on λ) and long time simulations are possible.*

the GPU, and particles are advected for the current time step using the mid-point scheme. CUDA is used to achieve interactive rates for scenarios with many filament edges and obstacles: The evaluations of the velocity field at filament vertices (Eq. (11)) and the flux evaluations through the obstacle facets (the RHS of Eq. (8)) are computed simultaneously for different vertices/facets.

7 Results and Discussion

We have chosen three different scenarios to demonstrate physical accuracy, plausibility, long-time stability and performance of our method.

Colliding vortex rings: We compare physical experiments of colliding vortex rings [Lim and Nickels 1992; Lim 1989] with our simulation. Videos of the real experiments can be found on youtube (head-on, oblique). Both scenarios work out-of-the-box with our simulator, fast enough to run with 65,536 particles at 25 fps. The oblique collision (Fig. 8, top) was computed this way. For the head-on collision (Fig. 8, bottom) though we had to use a 40 times smaller time-step because we had to trace particles in a very thin tube around the filaments. The mid-point scheme used for particle advection had difficulties keeping the particles close to the filaments when using large time-steps. The initial filaments are regular 16-gons, with each edge split into two segments by linear interpolation. This accounts for the 16-fold symmetry of the result. Lim reports that the number of small filaments that emerge from the collision varies from run to run. Both in our simulation and in the experiments slight deviations of the initial filaments from being round determine which frequency will dominate in the end.

Long-time jet simulation: We compare the number of edges required to simulate a jet with and without reconnection. A jet is simulated by repeatedly emitting new circular filaments at a fixed but slightly distorted position, to break symmetry. Fig. 6 shows the the simulations after 600 steps. We add a new filament (consisting of 16 edges) every 20th iteration, i.e., the number of edges added to the simulation grows linear over time (blue graph in Fig. 7). Without reconnection, the number of polygon edges increases exponentially

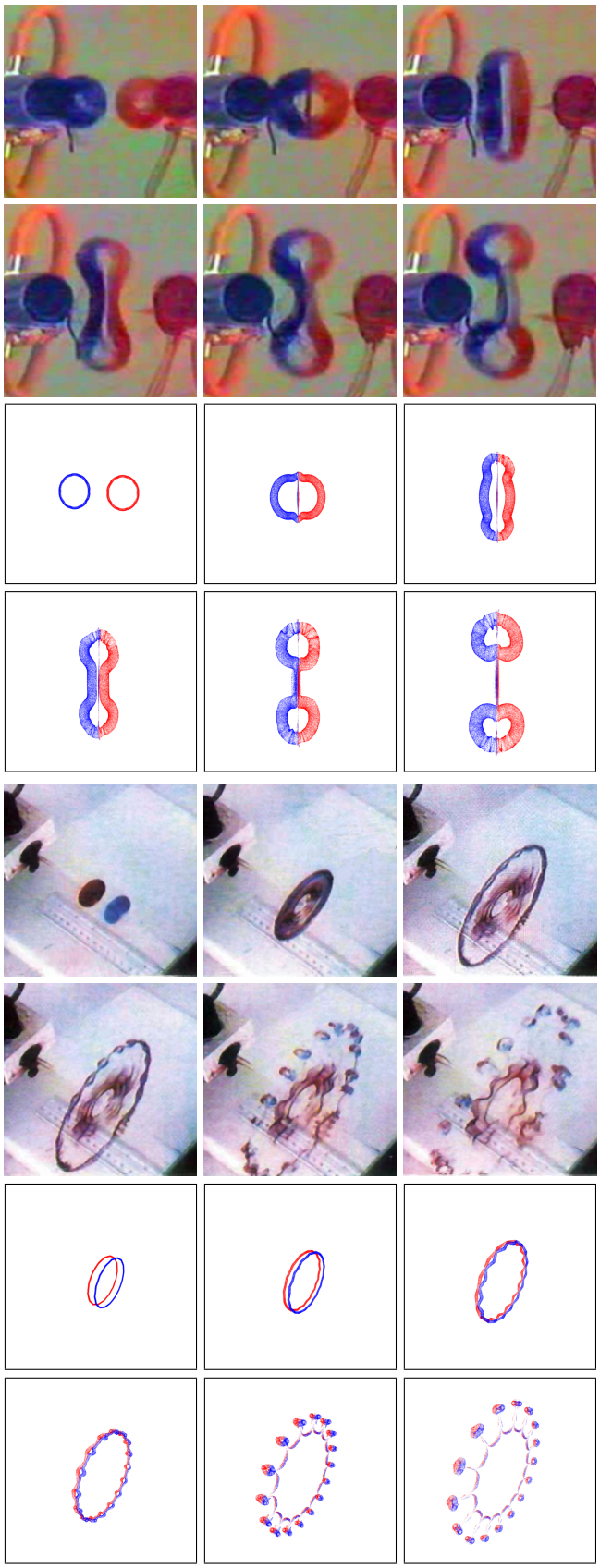


Figure 8: *Oblique and head-on collision of two vortex rings. Comparison with real experiments by T. T. Lim, with permission.*

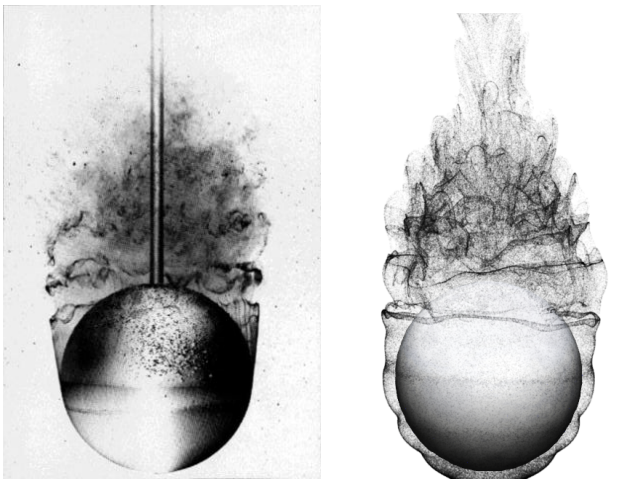


Figure 9: *Wake behind a sphere. Comparison of a photo by Henri Werlé (colors inverted) with our simulation. Reproduced with permission of ONERA, The French Aerospace Lab.*

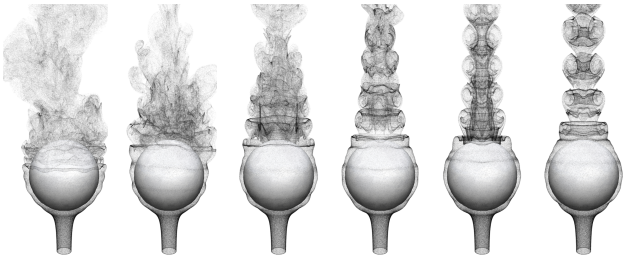


Figure 10: *Different wake patterns behind a sphere. The simulations differ only in the rate at which image vorticity filaments are released into the flow. Note the transition to turbulence from right to left.*

fast due to subdivision (red graph), which makes long-time simulations impractical. With reconnection (green graphs) we can easily achieve sub-linear growth, depending on the particular value of λ. Thus we are able to simulate over arbitrary long simulation times without explosion of computational costs.

Wake behind a sphere: Using our method for vortex shedding we achieve realistic wakes behind objects, as shown for an ellipsoid in Fig. 1. The wake behind a sphere from our simulation is compared to a photograph of an experiment by Henri Werlé [Dyke 1982] in Fig. 9. Different wake patterns (Fig. 10) can be achieved by varying the amount of vorticity shed into the flow.

Performance: The method does not scale nicely with the number of filament edges and marker particles, because each evaluation of the velocity field requires summation over all filament edges. Nevertheless we achieve good performance even for complex scenarios using parallelization (Sec. 6.2): All of the figures included in this paper are frames from simulations running at about 1 fps on our test machine (Core 2 Extreme CPU X9650 3.00GHz, GeForce 8800 Ultra), using 1,048,576 marker particles. Animations that are less optimized for quality but still detailed and plausible (i.e., with less particles) easily run at 20 fps or more. Note in particular that the filament simulations without particles always run at interactive rates. Since the filaments contain all information of the fluid motion, effects designers can work with a real time tool that shows the full information of the fluid motion with a moderate amount of particles.

8 Conclusion and outlook

Our method allows the simulation of 3D fluid flow around obstacles with vortex shedding. It is fast enough to run at interactive rates while achieving resolutions that are challenging for previous methods. Long-time simulations are possible through the use of reconnection, which keeps the number of filament edges low. Our new reconnection criterion is based on a variational principle, which also improves the overall geometry of the filaments. We have validated our implementation by reproducing real experiments of colliding vortex rings and realistic wakes behind objects. We emphasize that the algorithm is sufficiently robust and efficient to be used as an interactive tool for effects artists and in game engines.

Further work is needed to make large scenes tractable: LOD is needed to speed up velocity field evaluations. This will make the method scale much better with the number of marker particles and the number of filament edges. For scenes containing many moving obstacles some hierarchical scheme is required, e.g., hierarchical matrices [Börm et al. 2003]. A model for vorticity creation due to buoyancy is required to simulate the turbulent motion of hot gas. Forces exerted by the fluid on movable obstacles can be simulated via *virtual momentum* [Saffman 1992]. Furthermore, we want to find ways to handle free surfaces using filaments. This would enable us to simulate water with an interface to air.

Acknowledgment This work was supported by the DFG Research Center Matheon. Tee Tai Lim gave permission to use shots from his videos of colliding vortex rings (Fig. 8). ONERA, The French Aerospace Lab, allowed us to reproduce the photo by Henri Werlé (Fig. 9). We are gratefully indebted to Peter Schröder for his support. Special thanks to John M. Sullivan and Boris Springborn for proofreading and discussions.

References

ANGELIDIS, A., AND NEYRET, F. 2005. Simulation of Smoke based on Vortex Filament Primitives. In *Proc. Symp. Comp. Anim.*, 87–96.

ANGELIDIS, A., NEYRET, F., SINGH, K., AND NOWROUZEZAHRAI, D. 2006. A controllable, fast and stable basis for vortex based smoke simulation. In *Proc. Symp. Comp. Anim.*, 25–32.

ANGELIDIS, A. 2004. Hexanions: 6d Space for Twists. Tech. rep., University of Otago, November.

BECKER, M., AND TESCHNER, M. 2007. Weakly compressible SPH for free surface flows. In *Proc. Symp. Comp. Anim.*, 209–217.

BERNARD, P. S. 2006. Turbulent flow properties of large-scale vortex systems. *Proc. Natl. Acad. Sci. USA*.

BERNARD, P. S. 2009. Vortex filament simulation of the turbulent coflowing jet. *Phys. Fluids 21*.

BÖRM, S., GRASEDYCK, L., AND HACKBUSCH, W. 2003. *Hierarchical Matrices*. Lecture Notes. MPI MIS.

CHATELAIN, P., CURIONI, A., BERGDORF, M., ROSSINELLI, D., ANDREONI, W., AND KOUMOUTSAKOS, P. 2008. Billion vortex particle direct numerical simulations of aircraft wakes. *Comput. Methods Appl. Mech. Engrg. 197*, 1296–1304.

CHORIN, A. J. 1990. Hairpin removal in vortex interactions. *J. Comput. Phys. 91*, 1, 1–21.

CHORIN, A. J. 1993. Hairpin Removal in Vortex Interactions II. *J. Comput. Phys. 107*, 1, 1–9.

CRANE, K., LLAMAS, I., AND TARIQ, S. 2007. *GPU Gems 3 - Real-Time Simulation and Rendering of 3D Fluids*. Addison-Wesley, ch. 30, 633–673.

DESBRUN, M., AND GASCUEL, M.-P. 1996. Smoothed particles: a new paradigm for animating highly deformable bodies. In *Symp. on Computer animation and simulation*, Springer, 61–76.

DESBRUN, M., KANSO, E., AND TONG, Y. 2008. Discrete Differential Forms for Computational Modeling. In *Discrete Differential Geometry*, vol. 38 of *Oberwolfach Seminars*. Birkhäuser.

DORMAND, J. R., AND PRINCE, P. J. 1980. A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics 6*, 1, 19–26.

DYKE, M. V. 1982. *An album of fluid motion*. The parabolic Press, Stanford.

ELCOTT, S., TONG, Y., KANSO, E., SCHRÖDER, P., AND DESBRUN, M. 2007. Stable, circulation-preserving, simplicial fluids. *ACM Trans. Graph. 26*, 1.

FEDKIW, R., STAM, J., AND JENSEN, H. W. 2001. Visual simulation of smoke. In *Proc. ACM/SIGGRAPH Conf.*, 15–22.

GREENGARD, C., AND ANDERSON, C. R., Eds. 1988. *Vortex Methods*, vol. 1360 of *Lecture Notes in Mathematics*. Springer.

KATZ, J., AND PLOTKIN, A. 2001. *Low-Speed Aerodynamics*, 2 ed. No. 13 in Cambridge aerospace series. Cambridge Univ. Press.

KIM, B., LIU, Y., LLAMAS, I., AND ROSSIGNAC, J. 2005. FlowFixer: Using BFECC for Fluid Simulation. In *Proc. EG Workshop on Natural Phenomena*, E. Galin and P. Poulin, Eds.

KIM, T., THÜREY, N., JAMES, D., AND GROSS, M. 2008. Wavelet turbulence for fluid simulation. In *Proc. ACM/SIGGRAPH Conf.*, 1–6.

KIM, T. 2008. Hardware-aware analysis and optimization of stable fluids. In *Proc. I3D Symp.*, 99–106.

KRÜGER, J., AND WESTERMANN, R. 2005. GPU Simulation and Rendering of Volumetric Effects for Computer Games and Virtual Environments. *Comp. Graph. Forum 24*, 3.

LIM, T. T., AND NICKELS, T. B. 1992. Instability and reconnection in the head-on collision of two vortex rings. *Nature 357*, 225–227.

LIM, T. T. 1989. An experimental study of a vortex ring interacting with an inclined wall. *Experiments in Fluids 7*, 7 (July), 453–463.

MARSDEN, J., AND WEINSTEIN, A. 1983. Coadjoint orbits, vortices, and Clebsch variables for incompressible fluids. *Physica 7D*, 305–323.

MARZOUK, Y. M., AND GHONIEM, A. F. 2007. Vorticity structure and evolution in a transverse jet. *J. Fluid Mech. 575*, 267–305.

MULLEN, P., CRANE, K., PAVLOV, D., TONG, Y., AND DESBRUN, M. 2009. Energy-preserving Integrators for Fluid Animation. *Proc. ACM/SIGGRAPH Conf. 28*, 3.

MÜLLER, M., CHARYPAR, D., AND GROSS, M. 2003. Particle-based fluid simulation for interactive applications. In *Proc. Symp. Comp. Anim.*, 154–159.

MÜLLER, M., STAM, J., JAMES, D., AND THÜREY, N. 2008. Real time physics: class notes. In *ACM/SIGGRAPH classes*, 1–90.

NARAIN, R., SEWALL, J., CARLSON, M., AND LIN, M. C. 2008. Fast animation of turbulence using energy transport and procedural synthesis. *ACM Trans. Graph. 27*, 5, 1–8.

NEYRET, F. 2003. Advected Textures. In *Proc. Symp. Comp. Anim.*

PARK, S. I., AND KIM, M. J. 2005. Vortex fluid for gaseous phenomena. In *Proc. Symp. Comp. Anim.*, 261–270.

PFAFF, T., THUEREY, N., SELLE, A., AND GROSS, M. 2009. Synthetic turbulence using artificial boundary layers. In *Proc. ACM/SIGGRAPH Asia Conf.*, 1–10.

PINKALL, U., SPRINGBORN, B., AND WEISSMANN, S. 2007. A new doubly discrete analogue of smoke ring flow and the real time simulation of fluid flow. *J. Phys. A: Math. Theor. 40*, 42, 12563–12576.

RASMUSSEN, N., NGUYEN, D. Q., GEIGER, W., AND FEDKIW, R. 2003. Smoke simulation for large scale phenomena. *ACM Trans. Graph. 22*, 3, 703–707.

ROCKLIFF, S. H. L., PETER RYAL VOKE, AND NICOLE JACQUELINE. 2000. Three-Dimensional Vortices of a Spatially Developing Plane Jet. *International Journal of Fluid Dynamics 4*, 1–+.

SAFFMAN, P. G. 1992. *Vortex Dynamics*. Cambridge University Press, Cambridge.

SATO, K. 1999. *Lévy Processes and Infinitely Divisible Distributions*. Cambridge University Press.

SAUTER, S., AND SCHWAB, C. 2004. *Randelementmethoden*. Vieweg+Teubner.

SELLE, A., RASMUSSEN, N., AND FEDKIW, R. 2005. A vortex particle method for smoke, water and explosions. *ACM Trans. Graph. 24*, 3, 910–914.

STAM, J., AND FIUME, E. 1993. Turbulent wind fields for gaseous phenomena. In *Proc. ACM/SIGGRAPH Conf.*, 369–376.

STAM, J., AND FIUME, E. 1995. Depicting fire and other gaseous phenomena using diffusion processes. In *Proc. ACM/SIGGRAPH Conf.*, 129–136.

STAM, J. 1999. Stable fluids. In *Proc. ACM/SIGGRAPH Conf.*, 121–128.

TREUILLE, A., LEWIS, A., AND POPOVIĆ, Z. 2006. Model reduction for real-time fluids. *ACM Trans. Graph. 25*, 3, 826–834.

WEILER, K. 1985. Edge-Based Data Structures for Solid Modeling in Curved-Surface Environments. *Computer Graphics and Applications, IEEE 5*, 1 (Jan.), 21–40.

WEISSMANN, S., AND PINKALL, U. 2009. Real-time interactive simulation of smoke using discrete integrable vortex filaments. In *Proc. Vir. Real., Inter. & Phys. Sim.*, 1–10.

WEISSMANN, S., GUNN, C., BRINKMANN, P., HOFFMANN, T., AND PINKALL, U. 2009. jReality. In *Proc. ACM/MM Conf.*, 927–928.

YAN, H., WANG, Z., HE, J., CHEN, X., WANG, C., AND PENG, Q. 2009. Real-time fluid simulation with adaptive SPH. *Computer Animation and Virtual Worlds 20*, 2–3, 417–426.

YANG, Q. 2009. Real-time simulation of 3D smoke based on Navier-Stokes equation. *W. Trans. on Comp. 8*, 1, 103–112.

A Flux evaluation

For two polygons γ and η we need to compute the normal flux of the velocity field $S_a v_\gamma$ (generated by γ) through a disk D with boundary $\partial D = \eta$:

$$\mathrm{flux}_\eta(S_a v_\gamma) = \int_D \langle S_a v_\gamma, N\rangle \, dA.$$

Here N denotes the surface normal and dA the area element. When the distance between the two filaments η and γ is large we can approximate the flux by a simple one point quadrature (App. A.2). For nearby filaments the velocity field becomes almost singular, here the explicit formula (App. A.1) is needed. This is most significant for the self-flux $\mathrm{flux}_\eta(S_a v_\eta)$, which occurs in the diagonal entries of the obstacle matrix (Eq. (8)) and in the reconnection criterion (10).

A.1 Analytic evaluation

The vector potential ψ_γ of velocity field $S_a v_\gamma$ generated by a single smoke ring γ with unit strength $\Gamma = 1$ is

$$\psi_\gamma(x) = \frac{1}{4\pi} \oint \frac{\gamma'(s)}{\sqrt{a^2 + |x - \gamma(s)|^2}} \, ds,$$

and we can apply Stokes' theorem in order to compute the flux through a disk bounded by η:

$$\mathrm{flux}_\eta(S_a v_\gamma) = \oint_\eta \langle \psi_\gamma, \eta'(t)\rangle \, dt.$$

Let us denote the polygon edges by $l_i S_i = \gamma_{i+1} - \gamma_i$ and $L_j T_j = \eta_{j+1} - \eta_j$, with unit vectors S_i and T_j. Then,

$$\begin{aligned}\mathrm{flux}_\eta(S_a v_\gamma) &= \\ &= \sum_{i,j} \frac{\langle S_i, T_j\rangle}{4\pi} \int_0^{l_i}\!\!\int_0^{L_j} \frac{ds\,dt}{\sqrt{a^2 + |(\eta_i - \gamma_i) + tT_j - sS_i|^2}}\end{aligned}$$

and we have to compute

$$f = \int_0^l \!\! \int_0^L \frac{ds\,dt}{\sqrt{a^2 + |p + tT - sS|^2}} \tag{12}$$

for each pair of edges. We make use of the fact that the quadratic polynomial $|p + tT - sS|^2 - |p|^2$ defines a positive definite quadratic form r on the s,t-plane, given that S and T are not collinear (in this special case we obtain a much simpler formula which we will derive later). Computing the principal axes of r we obtain

$$f = \int_{M(\square)} \frac{1}{|\det G|} \frac{1}{\sqrt{K^2 + r^2}}, \quad \square = [0, L] \times [0, l]$$

where M is the affine map

$$M\colon \begin{pmatrix} s \\ t \end{pmatrix} \mapsto G \begin{pmatrix} s \\ t \end{pmatrix} + H,$$

$$G = \frac{1}{2} \begin{pmatrix} |S-T| & |S-T| \\ |S+T| & -|S+T| \end{pmatrix}, \quad H = -\begin{pmatrix} \langle p, \frac{S-T}{|S-T|}\rangle \\ \langle p, \frac{S+T}{|S+T|}\rangle \end{pmatrix}.$$

Here H is the projection of p onto the span of S and T, $K^2 = a^2 + |p - H|^2$ and $\det G = -\sin\alpha$, where $\alpha \in [0, \pi]$ is the angle between S and T.

We rephrase f as integration of the two-form $du \wedge dv / \sqrt{K^2 + r^2}$ and apply Stokes' Theorem. Let us denote the corners and edges of the parallelogram $M(\square)$ by A_i and $\ell_i B_i = A_{i+1} - A_i$. Then

$$\begin{aligned} f &= \frac{1}{\det G} \int_{M(\square)} \frac{du \wedge dv}{\sqrt{K^2 + r^2}} \\ &= \frac{1}{\det G} \int_{\partial M(\square)} \frac{u\,dv - v\,du}{\sqrt{K^2 + r^2} + K} \\ &= \frac{1}{\det G} \sum_{i=1\ldots4} \int_0^{\ell_i} \frac{\langle A_i + \xi B_i, JB_i\rangle \, d\xi}{\sqrt{K^2 + |A_i + \xi B_i|^2} + K} \\ &= \sum_{i=1\ldots4} \frac{\det(A_i, B_i)}{\det G} \int_0^{\ell_i} \frac{d\xi}{\sqrt{K^2 + |A_i + \xi B_i|^2} + K}. \end{aligned}$$

For small $\det G$ this formula can cause numerical issues that can be controlled by means of the right factorization depending on whether $S \to T$ or $S \to -T$. It remains to find an anti-derivative of

$$\frac{1}{K + \sqrt{K^2 + |A_i + \xi B_i|^2}} = \frac{1}{K + \underbrace{\sqrt{K^2 + P(\xi)}}_{Q(\xi)}}.$$

Rewrite the quadratic polynomial $P(\xi)$ as $(\xi + C)^2 + D^2$, with $C = \langle A, B\rangle$ and $D^2 = |A|^2 - \langle A, B\rangle^2$. For $D > 0$ an anti-derivative is given by

$$\frac{K}{D}\left(\operatorname{atan}\frac{D}{C+\xi} - \operatorname{atan}\frac{DQ(\xi)}{K(C+\xi)}\right) + \log\left(C + \xi + Q(\xi)\right),$$

which reduces in the limit $D \to 0$ to

$$-\frac{C+\xi}{K + Q(\xi)} + \log\left(C + \xi + Q(\xi)\right).$$

Parallel edges: For the case that $S = \pm T$, the contribution of a single edge pair (Eq. (12)) becomes

$$f = \int_0^l \!\! \int_0^L \frac{ds\,dt}{\sqrt{K^2 + \left((t \mp s) + \langle p, T\rangle\right)^2}},$$

with $K^2 = a^2 + |p|^2 - \langle p, T\rangle^2 \geq a^2$. Here an anti-derivative is given by

$$\pm\left(\sqrt{K^2 + X^2} - X \log\left(X + \sqrt{K^2 + X^2}\right)\right),$$

where we have used X as short hand for $\langle p, T\rangle + (t \mp s)$.

A.2 One point quadrature

Given that $S_a v_\gamma$ is generated by a vortex filament that is far away from D, $S_a v_\gamma$ is approximately constant over the whole disk. Let $\eta_1, \ldots, \eta_m$ be the vertices of η. We use the approximation

$$\begin{aligned} c_\eta &= \frac{1}{m} \sum \eta_i, \\ I_\eta &= \frac{1}{2} \sum \eta_i \times \eta_{i+1}, \\ \mathrm{flux}_\eta(S_a v_\gamma) &\approx \langle (S_a v_\gamma)(c_\eta), I_\eta\rangle. \end{aligned}$$

B Flux from rigid motion

We assume that the rigid motion g is a *screw motion* over the current time step Δt. It is determined by an element $\dot{g} = (\theta\omega, h\omega - \theta\omega \times c)$ in the Lie algebra $se(3)$ of the Lie group $SE(3)$ of rigid motions. The flux through the polygon η with vertices η_i is given by

$$\mathrm{flux}_\eta(\dot{g}) = \frac{1}{\Delta t}\left(\langle A_\eta, \theta\omega\rangle + \langle I_\eta, h\omega - \theta\omega \times c\rangle\right),$$

where

$$\begin{aligned} I_\eta &= \frac{1}{2}\sum \eta_i \times \eta_{i+1}, \\ A_\eta &= \sum \frac{\eta_i - \eta_{i+1}}{2}\left(\frac{|\eta_{i+1} - \eta_i|^2}{3} + \langle \eta_i, \eta_{i+1}\rangle\right). \end{aligned}$$

To determine θ, h, ω and $\theta\omega \times c$ we assume that the initial obstacle pose is the identity, and its final pose (after the time step) is given as a rotation matrix R together with a translation vector T (see also [Angelidis 2004]). Then $\theta\omega$ is obtained from

$$R - R^T = 2\sin(\theta)\omega,$$

where the anti-symmetric 3×3-matrices are identified with $\mathbb{R}^3$. We further know that the translation vector is given by $T = c - Rc + h\omega$ with $c \perp \omega$, $Rc \perp \omega$. This yields $h = \langle T, \omega\rangle$.

Denoting a rotation with axis α and rotation angle $|\alpha|$ by R_α we finally obtain

$$-\theta\omega \times c = \frac{\theta}{2\sin\frac{\theta}{2}} R_{-\frac{\theta}{2}\omega}(Rc - c).$$