Filament-Based Smoke

vorgelegt von
Diplom-Mathematiker
Steffen Weißmann
Neuburg an der Donau

Von der Fakultät II – Mathematik und Naturwissenschaften
der Technischen Universität Berlin
zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften
Dr. rer. nat.

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Martin Skutella
Berichter/Gutachter:	Prof. Dr. Ulrich Pinkall
Berichter/Gutachter:	Prof. Dr. Peter Schröder

Tag der wissenschaftlichen Aussprache: 15. 9. 2010

Berlin 2010

D 83

Abstract

This cumulative dissertation presents a complete model for simulating smoke using polygonal vortex filaments. Based on a Hamiltonian system for the dynamics of smooth vortex filaments, we develop an efficient and robust algorithm that allows simulations in real time. The discrete smoke ring flow allows to use coarse polygonal vortex filaments, while preserving the qualitative behavior of the smooth system. The method handles rigidly moving obstacles as boundary conditions and simulates vortex shedding. Obstacles as well as shed vorticity are also represented as polygonal filaments. Variational vortex reconnection prevents the exponential increase of filament length over time, without significant modification of the fluid velocity field. This allows for simulations over extended periods of time. The algorithm reproduces various real experiments (colliding vortex rings, wakes) that are challenging for classical methods.

Zusammenfassung

Die vorliegende kumulative Dissertation behandelt die Simulation von Rauch mittels polygonaler Wirbelfilamente. Basierend auf einem Hamiltonschen System für glatte Wirbelfilamente wird ein robuster und effizienter Algorithmus entwickelt, mit dem die Bewegung von Rauch in Echtzeit simuliert werden kann. Durch den diskreten Rauchringfluss bleiben die qualitativen Eigenschaften des glatten Systems auch bei grober Diskretisierung erhalten. Die Methode ermöglicht realistische und hochdetaillierte Simulationen, wobei auch starr bewegte Hindernisse als Randbedingung berücksichtigt werden. Dabei wird auch das Ablösen der Grenzschicht ("vortex shedding") simuliert. Die Hindernisse, sowie die abgelösten Wirbel, werden wiederrum durch polygonale Wibelfilamente dargestellt, somit basiert das gesamte Modell ausschließlich auf polygonalen Filamenten. Das exponentiell schnelle Anwachsen der Länge der Filamente wird durch variationale Wiederverbindung ("vortex reconnection") verhindert, ohne dabei die Strömung signifikant zu verändern. Damit sind Simulationen über lange Zeiträume möglich. Mit dem Algorithmus können viele reale Experimente (z.B. kollidierende Wirbelringe, Wirbelschleppen) reproduziert werden, die mit klassischen Verfahren nur schwer zu simulieren sind.

Contents

Introduction

Filament-based simulation of smoke is highly attractive for the creation of special effects in computer graphics, because filaments provide a very compact model for realistic smoke. The main reason is that vorticity usually comes to life as two-dimensional vortex sheets in the boundary layers of obstacles. These vortex sheets curl up into one-dimensional structures, which can be modeled directly by filaments. Simulating the evolution of these filaments is extremely efficient and gives a tremendous amount of detail. From an effect artist's point of view, vortex filaments provide a natural and intuitive primitive for the design of flows.

The vortex filaments determine the fluid velocity field completely, and their time evolution is simply the forward advection with the velocity field they generate. A straightforward implementation of this evolution is however problematic: The velocity field is infinite on the filaments, and the use of polygonal vortex filaments leads to a discretization error (related to the *smoke ring flow*) which cannot be discarded. Vortex stretching in 3D results in an exponential increase of total filament length over time, making simulations over extended periods of time impractical. Also obstacles are difficult to incorporate, because filament-based simulation takes place in unbounded space.

While vortex filaments are frequently used in computational physics for approximating certain vorticity fields by a large number of weak filaments [Cho90, Cho93, AH88, MM91, MM92, Ber06, AH88, Ber09], they have not attracted much attention in computer graphics. The basic idea was introduced to the community by Angelidis and Neyret [AN05] with an unphysical modification of the equation of motion. The resulting method incorporates parameters that have no physical interpretation, which makes it difficult to achieve realistic simulations. Smoke simulations for computer graphics are mostly based on semi-Lagrangian methods on a Eulerian grid [Sta99, FSJ01], allowing only low resolutions at interactive rates.

We develop a robust and efficient numerical method from the discretization of smooth vortex filaments as polygons. The problems mentioned before are addressed by a solid mathematical understructure that incorporates ideas from discrete differential geometry and integrable systems. By the use of the computational power provided by commodity graphics hardware, we achieve real-time performance even for complex scenes.

Summary

In the three consecutive articles of this cumulative dissertation we have developed a model for filament-based smoke simulation that is (a) physically correct and mathematically sound, (b) efficient enough for interactive applications, (c) stable and efficient over long simulation horizons, and (d) includes moving obstacles that shed vorticity.

Ulrich Pinkall, Boris Springborn and Steffen Weißmann. A new doubly-discrete analog of smoke ring flow and the real time simulation of fluid flow. *J. Phys. A: Math. Theor.* 40(42), 2007.

The article describes the doubly-discrete smoke ring evolution for polygons, based on iterated Darboux transforms. The system converges to the smoke ring flow in the limit of smooth curves. The evolution preserves length and area vector of a polygon, which corresponds to the conservation of energy and momentum for the smooth system. The discrete evolution is needed in the numerical algorithm for filament-based fluid simulation to compensate the lack of local interaction that occurs when the smooth vortex filaments are replaced by polygons. The discretization error stems from the fact that a vertex is not affected by its two adjacent edges, while the velocity of a smooth filament incorporates a contribution in binormal direction that is proportional to its curvature. The discrete integrable system is an extension of previous work by Hoffmann [Hof08], allowing for polygons that have non-constant edge lengths. This is crucial since the non-local effects from the Biot-Savart velocity field quickly destroy any arc-length parametrization. In addition, an explicit evaluation of the Biot-Savart formula is given for polygonal filaments. In fact our evaluation of the correct velocity field is computationally more efficient than the unphysical modification in [AN05]. The article also describes how to determine the necessary parameters for the Darboux transforms to match a certain time step. Finally it sketches the implementation of the basic simulation algorithm that allows real-time simulation with a moderate amount of filaments.

Steffen Weißmann and Ulrich Pinkall. Real-time interactive simulation of smoke using discrete integrable vortex filaments. *Proc. Vir. Real., Inter. & Phys. Sim.*, 2009.

The paper describes the implementation of an interactive application to simulate smoke on desktops and also in immersive virtual environments, based on the theory developed in [PSW07]. The paper gives aspects of implementation and pseudo-code for the doubly-discrete smoke ring evolution. The use of vortex filaments is motivated as a model for vorticity fields that develop in real flows. Vorticity is usually created in the boundary layer around obstacles, resulting in vortex sheets that are swept away by the flow. These sheets quickly roll up into one–dimensional structures which can be modeled excellently by a small number of filaments. This insight enables

modeling of very realistic flows. As validation, very realistic and highly resolved jets were simulated. Also a new way for the design of divergence-free background fields by static vortex filaments has been introduced. This allows control of large scale features of the flow.

Steffen Weißmann and Ulrich Pinkall. Filament-based smoke with vortex shedding and variational reconnection. *ACM Trans. Graph. 29(4), Proc. ACM/SIGGRAPH Conf.*, 2010.

In this paper we provide several important enhancements to the method described in [WP09]. Vortex reconnection allows for long time simulations, since it avoids the exponential growth of computational complexity. We have given a new variational reconnection criterion that prevents alternating reconnections which occur when using the criterion introduced by Chorin [Cho90, Cho93]. In addition, our reconnections improve the overall geometry of the filaments. Minimization of the underlying energy can optionally be used to further improve the filament geometries. Moving obstacles as boundary conditions have been included into the algorithm. Obstacles are represented by their surface, given as a triangle mesh. Given such a (possibly moving) obstacle, a vortex sheet on the surface is computed that makes the flow go around the obstacle and creates zero velocity inside. The vortex sheet is computed using a variant of the *panel method* [KP01]. Finally the vortex sheet is discretized as a collection of vortex filaments on the obstacle, making the method consistent in the sense that the whole simulation is purely driven by filaments. Vortex shedding is added to the simulation on top of the obstacle method. Physically, vortex shedding simply denotes the fact that the boundary layer vortex sheet is swept away downstream with the flow. It is well-known that vortex sheets quickly roll up into one–dimensional structures, which can be modeled excellently by filaments. Consequently, we simulate this effect by releasing some of the obstacle filaments into the flow. For validation we have simulated several real experiments. Filament collisions were simulated to validate the physical correctness of the proposed reconnection criterion. Jet simulations with different reconnection parameters demonstrate the long time stability of the method. Our method of vortex shedding was validated by comparing the simulation of a wake behind a sphere with experimental data.

1 A new doubly discrete analogue of smoke ring flow and the real time simulation of fluid flow

Ulrich Pinkall, Boris Springborn, Steffen Weißmann

J. Phys. A: Math. Theor. 40, 2007, pp. 12563–12576.

Modelling incompressible ideal fluids as a finite collection of vortex filaments is important in physics (super-fluidity, models for the onset of turbulence) as well as for numerical algorithms used in computer graphics for the real time simulation of smoke. Here we introduce a time-discrete evolution equation for arbitrary closed polygons in 3-space that is a discretisation of the localised induction approximation of filament motion. This discretisation shares with its continuum limit the property that it is a completely integrable system. We apply this polygon evolution to a significant improvement of the numerical algorithms used in Computer Graphics.

1.1 Introduction

The motion of vortex filaments in an incompressible, inviscid fluid has aroused considerable interest in quite different areas:

Differential geometry. The limiting case of infinitely thin vortex filaments leads to an evolution equation for closed space curves γ,

$$\dot{\gamma} = \gamma' \times \gamma''. \tag{1.1}$$

Equation (1.1) was discovered in the beginning of the 20th century by Levi-Civita and his student Da Rios [SDR06] and is called the *smoke ring flow* or *localised induction approximation*. In 1972 Hasimoto [Has72] discovered that the smoke ring flow is in fact a completely integrable Hamiltonian system equivalent to the non-linear Schrödinger equation. See [Ric91] for more details on the history of the smoke ring equation. Subsequently the smoke ring flow has been studied by differential geometers as a natural evolution equation for space curves [CI05, CGS86, Ive06, LP90]. Also discrete versions of the smoke ring flow in the form of completely integrable evolution equations for polygons with fixed edge length have been developed [Hof00, Hof08, DS95].

Fluid dynamics. As will be explained below, for applications in fluid mechanics a finite thickness of the vortex filaments has to be taken into account. The transition

from infinitely thin filaments to filaments of finite thickness involves the incorporation of long range interactions (governed by the Biot-Savart law) between different filaments and different parts of the same filament into the purely local evolution equation (1.1). The resulting evolution of vortex filaments has been extensively studied both numerically and in the context of explaining the onset of turbulence [Cho91]. Including in addition a small amount of viscosity in the equations leads to striking physical effects like vortex reconnection [KL93, KL03, CKL03] and numerical techniques like "hairpin removal" [Cho90, Cho93].

Computer graphics. Filament-based methods for fluid simulation are becoming important in Computer Graphics for special effects in motion pictures and for real time applications like computer games [AN05, ANSN06]. Here the emphasis is on physical correctness and speed rather than numerical accuracy. Filament methods are ideal for these applications because complicated fluid motions can be created by a graphics designer by modelling the initial positions and strengths of the filaments. Moreover, filament methods work in unbounded space rather than in a bounded box (as is the case for grid-based methods [Sta99]). This is desirable for the simulation of smoke.

The main goal of this paper is to improve the numerical algorithms currently used in Computer Graphics by applying the recent knowledge from Discrete Differential Geometry to the motion of polygonal smoke rings. Our method makes it possible to model thin filaments by polygons with arbitrarily few vertices. For comparison, using current methods to model a circular smoke ring which is thin enough to entrain smoke in a torus shape, it is necessary to use a regular polygon with at least 800 vertices.

In Section 1.2 we will explain the evolution equation for systems of vortex filaments that we will discretise. The resulting equations of motion are still Hamiltonian like the smoke ring flow (1.1). However, since already Poincaré knew that a system of vortex filaments consisting of more than three parallel straight lines (the "n-vortex problem") fails to be an integrable system [AK98, p. 58f], we do not believe that this system is an integrable Hamiltonian system. Nevertheless it is a small perturbation of the integrable system constituted by the limit of infinitely thin filaments. This might be interesting for future investigations along the lines of KAM theory.

In Section 1.3 we consider polygonal vortex filaments. In this case, there is an elementary formula (1.11) for the Biot-Savart integral.

In Section 1.4 we will develop an extension of the known discrete-time smoke ring flow for polygons of constant edge lengths to arbitrary polygons. This is needed because after including the long range Biot-Savart interactions, the lengths of the edges will be no longer constant in time.

In the theory of integrable systems it is known at least since the 1980s that integrable difference equations may be interpreted as Darboux transformations of integrable

differential equations [LB80, Lev81, NQC83]. In the meantime, this seminal discovery has lead to a reversed point of view, where the discrete integrable systems are considered fundamental and the continuous systems appear as smooth limits (see for example [ABS03] and the references therein). In this vein, we will in Section 1.4 define the discrete-time integrable system in terms of iterated Darboux transformations of polygons and show afterwards that the smoke ring flow is obtained as a smooth limit.

In Section 1.5 we will describe our numerical method that very efficiently models the motion of fluids near the smoke ring limit.

1.2 Euler's Equation for Vortex Filaments

Consider an incompressible, inviscid fluid in euclidean 3-space whose velocity field u vanishes at infinity and whose vorticity $\omega = \operatorname{curl} u$ is compactly supported. Then u can be reconstructed from ω by the *Biot-Savart* formula

$$u(x) = -\frac{1}{4\pi} \int_{\mathbb{R}^3} \frac{x-z}{|x-z|^3} \times \omega(z)\, dz. \tag{1.2}$$

The equation of motion can then be written as

$$\dot{\omega} = [\omega, u]. \tag{1.3}$$

Viewed as an evolution equation on the vector space $\mathcal{M}$ of compactly supported divergence-free vector fields on $\mathbb{R}^3$ this is a Hamiltonian system: A symplectic form σ on $\mathcal{M}$ is defined as follows. Let $\omega \in \mathcal{M}$ and $\dot{\omega}, \mathring{\omega} \in T_\omega \mathcal{M}$. Then

$$\sigma_\omega(\dot{\omega}, \mathring{\omega}) = \int_{\mathbb{R}^3} \det(\omega, \dot{\omega}, \mathring{\omega}). \tag{1.4}$$

Let $H : \mathcal{M} \to \mathbb{R}$ be the quadratic function

$$H = \iint \frac{\langle \omega(x), \omega(y) \rangle}{|x-y|}\, dx\, dy, \tag{1.5}$$

where $\langle \cdot, \cdot \rangle$ is the euclidean scalar product on $\mathbb{R}^3$. Then H is the Hamiltonian for the dynamical system (1.3). See [AK98, EM70] for more details on this Hamiltonian description of ideal fluids.

If the vorticity of a fluid is concentrated on a closed curve γ in a delta-function like manner, by Equation (1.2) the resulting velocity field u becomes

$$u(x) = -\frac{\Gamma}{4\pi} \oint \frac{x-\gamma(s)}{\|x-\gamma(s)\|^3} \times \gamma'(s)\, ds. \tag{1.6}$$

Here Γ is the circulation around the filament. The problem with Equation (1.6) is that in order to determine the motion of γ itself, u has to be evaluated on γ, which

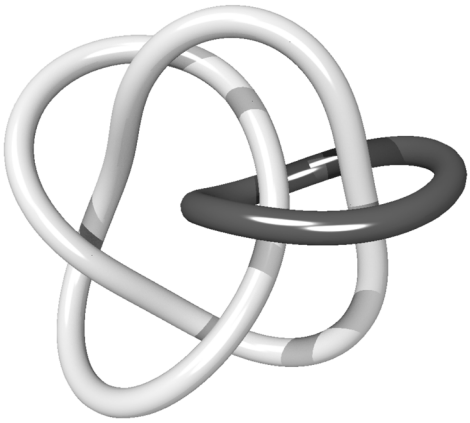


Figure 1.1: *The space of links as the phase space for vortex filaments.*

results in a logarithmically divergent integral. Usually, this problem is handled by considering a vorticity field concentrated in a tube around γ of small but finite radius r. For small r the velocity in this tube is dominated by a term proportional to the localised induction approximation. (See, for example, [Saf92, p. 36f].) Here we want to derive the smoke ring flow by taking the limit $r \to 0$. In order to prevent vortex filaments acquiring infinite speed, one has to scale the circulation Γ down to zero when performing the limit to infinitely thin filaments. This means that the fluid velocity (1.6) goes to zero as well.

The resulting picture is then as follows: The fluid is completely at rest away from the filaments while the filaments just cut through the fluid with finite speed according to the smoke ring flow:

$$\dot{\gamma}_j = K_j \, \gamma_j' \times \gamma_j''. \tag{1.7}$$

Here the constants K_j account for the fact that the circulation of the different filaments γ_j might go to zero at a different rate.

Equation (1.7) can be viewed as a completely integrable Hamiltonian system on the space of weighted links (see Figure 1.1) endowed with the symplectic form

$$\sigma_\gamma(\dot{\gamma}, \mathring{\gamma}) = \sum_j K_j \oint_{\gamma_j} \det(\gamma_j', \dot{\gamma}, \mathring{\gamma}). \tag{1.8}$$

For single curves this symplectic form is due to V. I. Arnold [AK98]. The corresponding Hamiltonian is a weighted sum of the filament lengths

$$H = \sum_j K_j \text{ Length}(\gamma_j).$$

Equation (1.7) can be obtained (using a simple renormalisation of time) as a limit as $a \to 0$ of the following system: Stick with (1.8) as the symplectic form, with K_j replaced by the non-zero circulation Γ_j around γ_j. As a Hamiltonian, use

$$H = \sum_{i,j} \frac{\Gamma_i \Gamma_j}{8\pi} \oint\!\!\oint \frac{\langle \gamma_i'(s), \gamma_j'(\tilde{s}) \rangle}{\sqrt{a^2 + |\gamma_i(s) - \gamma_j(\tilde{s})|^2}} \, ds \, d\tilde{s}.$$

The resulting equation of motion is

$$\dot{\gamma}_i(s) = -\sum_j \frac{\Gamma_j}{4\pi} \oint \frac{\gamma_i(s) - \gamma_j(\tilde{s})}{\sqrt{a^2 + |\gamma_i(s) - \gamma_j(\tilde{s})|^2}^3} \times \gamma_j'(\tilde{s}) \, d\tilde{s}. \tag{1.9}$$

This equation of motion (1.9) can also be derived as follows:

- Smooth the delta-function like vorticity field ω_0 of the link by a suitable convolution kernel and obtain

$$\omega(x) = \frac{3a^2}{4\pi} \int_{\mathbb{R}^3} \frac{\omega_0(y)}{\sqrt{a^2 + |x-y|^2}^5} \, dy.$$

- Compute the corresponding velocity field u with $\operatorname{curl} u = \omega$:

$$u(x) = -\frac{\Gamma}{4\pi} \sum_j \oint \frac{x - \gamma_j(s)}{\sqrt{a^2 + |x - \gamma_j(s)|^2}^3} \times \gamma_j'(s) \, ds. \tag{1.10}$$

- Evaluate u on the filaments to obtain (1.9).

To summarise: We model fluid motion near the filament limit by a Hamiltonian system on the space of weighted links. This system is still Hamiltonian but no longer integrable. Nevertheless it still has all the constants of motion induced by invariance with respect to the euclidean symmetry group. For example the weighted sum of the area vectors

$$A = \sum_j \Gamma_j \oint \gamma_j' \times \gamma_j$$

is one of the preserved quantities. (Compare Theorem 4 of Section 1.4.)

The physical approximation implicit in our model is that we ignore possible deformations of the internal structure of the filaments and reduce everything to the evolution of the filament curves. The finite thickness of the filaments is taken into account by applying a fixed convolution kernel.

1.3 Polygonal Vortex Filaments

In order to develop a numerical method for modelling fluid motion near the filament limit we have to discretise the vortex filaments, i.e. we replace them by polygons. If

γ is a piecewise linear parametrisation of a closed polygon, on each edge we have $\gamma'' = 0$ and we find an explicit anti-derivative for the integrands of equation (1.10):

$$\left(\frac{\langle \gamma, \gamma' \rangle}{\sqrt{a^2 + |\gamma|^2}\ (|\gamma'|^2 a^2 + |\gamma \times \gamma'|^2)} \gamma \times \gamma' \right)' = \frac{\gamma \times \gamma'}{\sqrt{a^2 + |\gamma|^2}^3}. \tag{1.11}$$

Here we have abbreviated $x - \gamma_j(s)$ to γ, $\gamma_j'(s)$ to γ' and the prime is derivation with respect to s.

Inspection of Equation (1.11) reveals the following problem: The two adjacent edges have no influence at all on the velocity of a vertex. This amounts to effectively employing a distance cut-off in order to regularise the singular integral (1.6) for points on γ. It is known [Saf92] that this is roughly equivalent to modelling vortex tubes of thickness equal to the edge length of the polygon. Using this model we would therefore be unable to model thin (and therefore fast) filaments without using excessively many edges for each polygon.

The contribution of local effects behaves like the smoke ring flow and the resulting equation of motion for a vertex γ_i of a polygonal vortex filament γ is then

$$\dot{\gamma}_i = u(\gamma_i) + \lambda \kappa_i b_i, \tag{1.12}$$

where u is given by Equation (1.10) using (1.11), $\kappa_i b_i$ denotes curvature times binormal at γ_i, and λ is constant for fixed a. Since the non-local effects quickly destroy any arc-length parametrisation (i.e. the lengths of the different edges of the polygon) and we do not have an adequate notion of curvature for arbitrary polygons, we can not evaluate (1.12) directly.

On the other hand, for polygons with constant edge lengths it is known that the doubly discrete smoke ring (or Hasimoto) flow [Hof08] captures excellently the qualitative behaviour of the smooth smoke ring flow. In the next section we will discuss a version of this doubly discrete smoke ring flow which works also for polygons with varying edge lengths.

1.4 Darboux Transformation of Polygons

In this section we develop a discrete-time evolution for closed polygons that has the smoke ring flow (1.1) as a limit when the polygon approaches a smooth curve and the time-step goes to zero. This evolution (obtained by iterating so-called Darboux transformations) shares with its continuum limit the property that it is a completely integrable system in the sense that it comes from a Lax pair of quaternionic 2×2-matrices with a spectral parameter. (This system therefore fits into the framework of [BS02].) The constants of the motion of the discrete system converge to constants of the motion of the smooth system in the limit.

Let $\gamma : \mathbb{Z} \to \mathbb{R}^3$ be an immersed polygon in $\mathbb{R}^3$, where *immersed* means that $\gamma_i \neq \gamma_{i+1}$ for all $i \in \mathbb{Z}$, and let $S_i = \gamma_{i+1} - \gamma_i$. If γ is periodic with some period n, then the

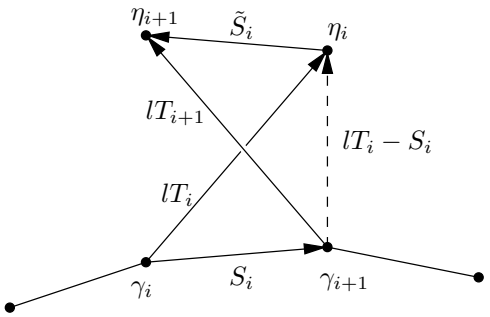


Figure 1.2: *A polygon γ and an edge of its Darboux transform η.*

polygon is *closed* and γ may be interpreted as a function on $\mathbb{Z}/n\mathbb{Z}$. In the following, we identify $\mathbb{R}^3$ with the imaginary quaternions $\operatorname{Im}\mathbb{H} = \{xi + yj + zk \,|\, x, y, z \in \mathbb{R}\}$.

Definition. A polygon η is called a *Darboux transform* of γ with twist parameter $r \in \mathbb{R}$ and distance $l > 0$, if $\|\eta_i - \gamma_i\| = l$ for all $i \in \mathbb{Z}$, and the normalised difference vectors T_i defined by $lT_i = \eta_i - \gamma_i$ satisfy the quaternionic equation

$$T_{i+1} = (-r + lT_i - S_i)T_i(-r + lT_i - S_i)^{-1}. \tag{1.13}$$

The Darboux transformation of polygons and its relationship with the nonlinear Schrödinger equation and smoke ring flow was treated in [Hof08] under the assumption that the polygon γ has constant edge length. To drop this assumption was suggested to us by Tim Hoffmann [Hof05].

Geometrically, Equation (1.13) has the following meaning (see Figure 1.2).

The difference vector T_{i+1} is obtained from T_i by a rotation with axis $lT_i - S_i$. The quadrilateral $\gamma_i\gamma_{i+1}\eta_{i+1}\eta_i$ is therefore a "folded parallelogram". In particular, corresponding edges of γ and η have the same length. The angle of rotation is $2\arctan(\|lT_i - S_i\|/r)$. For $r = 0$ it is π. For $r \to \pm\infty$, it goes to zero and in the limit the Darboux transformation becomes a translation.

Equation (1.13) can be written in the form

$$T_{i+1} = (aT_i + b)(cT_i + d)^{-1}, \tag{1.14}$$

where $a, b, c, d \in \mathbb{H}$ depend on S_i and the parameters l, r. That is, for each $i \in \mathbb{Z}$, T_{i+1} is obtained by applying a quaternionic fractional linear transformation $f_i : \bar{\mathbb{H}} \to \bar{\mathbb{H}}$ to T_i, where $\bar{\mathbb{H}} = \mathbb{H} \cup \{\infty\}$. Indeed, (1.13) is equivalent to

$$T_{i+1} = \big(lT_i - r - S_i\big)\big((r + S_i)T_i + l\big)^{-1}. \tag{1.15}$$

To see this note that $T_i^{-1} = -T_i$ because T_i is a purely imaginary unit quaternion, and hence $T_i(-r + lT_i - S_i)^{-1} = (rT_i + l + S_iT_i)$.

It is convenient to rewrite fractional linear transformations as matrix multiplication. Just as the extended complex plane $\bar{\mathbb{C}} = \mathbb{C} \cup \{\infty\}$ can be identified with the Riemann sphere S^2 and with the complex projective line $\mathbb{C}\mathrm{P}^1$, $\bar{\mathbb{H}} \cong S^4 \cong \mathbb{H}\mathrm{P}^1$. The quaternionic projective line $\mathbb{H}\mathrm{P}^1$ is the set of (quaternionic) 1-dimensional subspaces of the vector space $\mathbb{H}^2$ over $\mathbb{H}$. We consider $\mathbb{H}^2$ as right vector space: the product of a vector $\left(\begin{smallmatrix} p \\ q \end{smallmatrix}\right) \in \mathbb{H}^2$ and a scalar $\lambda \in \mathbb{H}$ is $\left(\begin{smallmatrix} p \\ q \end{smallmatrix}\right)\lambda = \left(\begin{smallmatrix} p\lambda \\ q\lambda \end{smallmatrix}\right)$. A point

$$\begin{bmatrix} p \\ q \end{bmatrix} = \begin{pmatrix} p \\ q \end{pmatrix} \mathbb{H} \in \mathbb{H}\mathrm{P}^1$$

corresponds to the point $pq^{-1} \in \bar{\mathbb{H}}$, and p, q are quaternionic homogeneous coordinates for this point. Now any fractional linear transformation of $\bar{\mathbb{H}}$ can be written as quaternionic 2×2-matrix acting from the left on quaternionic homogeneous coordinates of $\mathbb{H}\mathrm{P}^1$: Writing T_i in quaternionic homogeneous coordinates,

$$T_i = T_i^{(1)} (T_i^{(2)})^{-1},$$

one obtains from (1.15)

$$\begin{pmatrix} T_{i+1}^{(1)} \\ T_{i+1}^{(2)} \end{pmatrix} = U_i(l, r) \begin{pmatrix} T_i^{(1)} \\ T_i^{(2)} \end{pmatrix}, \quad U_i(\lambda, \rho) := \begin{pmatrix} \lambda & -\rho - S_i \\ \rho + S_i & \lambda \end{pmatrix}. \tag{1.16}$$

The following Theorem 1 characterises the Darboux transformations of polygons via a Lax pair of quaternionic 2×2-matrices with spectral parameter. Theorem 2 is a permutability theorem for these Darboux transformations.

Theorem 1 (Lax pair)**.** *Let $S_i = \gamma_{i+1} - \gamma_i$, $|T_i| = 1$, and let $U_i(\lambda, \rho)$ be defined by (1.16) and*

$$\tilde{U}_i(\lambda, \rho) = \begin{pmatrix} \lambda & -\rho - \tilde{S}_i \\ \rho + \tilde{S}_i & \lambda \end{pmatrix},$$

$$V_i(\lambda, \rho) = \begin{pmatrix} \lambda & -\rho + r - lT_i \\ \rho - r + lT_i & \lambda \end{pmatrix}.$$

Then

$$V_{i+1}(\lambda, \rho) U_i(\lambda, \rho) = \tilde{U}_i(\lambda, \rho) V_i(\lambda, \rho) \tag{1.17}$$

for all $\lambda, \rho \in \mathbb{R}$, if and only if S and T satisfy (1.13) and

$$lT_{i+1} + S_i = \tilde{S}_i + lT_i. \tag{1.18}$$

That is, if and only if $\eta = \gamma + lT$ is a Darboux transform of γ with twist parameter r and distance l, and $\tilde{S}_i = \eta_{i+1} - \eta_i$.

Of course (1.17) means that the following diagram commutes:

$$\begin{array}{ccc} \mathbb{H}^2 & \xrightarrow{\tilde{U}_i} & \mathbb{H}^2 \\ {\scriptstyle V_i}\uparrow & & \uparrow{\scriptstyle V_{i+1}} \\ \mathbb{H}^2 & \xrightarrow{U_i} & \mathbb{H}^2 \end{array}$$

Proof. Note that in general for quaternionic 2×2-matrices with $a_+, a, b, \tilde{b} \in \mathbb{H}$ and $\lambda \in \mathbb{R}$ the equality

$$\begin{pmatrix} \lambda & a_+ \\ -a_+ & \lambda \end{pmatrix} \begin{pmatrix} \lambda & b \\ -b & \lambda \end{pmatrix} = \begin{pmatrix} \lambda & \tilde{b} \\ -\tilde{b} & \lambda \end{pmatrix} \begin{pmatrix} \lambda & a \\ -a & \lambda \end{pmatrix}$$

is equivalent to

$$a_+ b = \tilde{b} a \qquad \text{and} \qquad \lambda(a_+ + b) = \lambda(\tilde{b} + a).$$

It follows that (1.17) holds for all $\lambda \in \mathbb{R}$, if and only if (1.18) holds and

$$(-\rho + r - lT_{i+1})(-\rho - S_i) = (-\rho - \tilde{S}_i)(-\rho + r - lT_i).$$

Use (1.18) to eliminate $\tilde{S}_i$ from this equation and gather terms of equal power in ρ on both sides. The coefficients of ρ^2 are both 1, and the coefficients of ρ are obviously equal. What remains is the equation

$$(r - lT_{i+1})(-S_i) = (-S_i - lT_{i+1} + lT_i)(r - lT_i).$$

Solve for T_{i+1} to obtain (1.13). □

Theorem 2 (Permutability)**.** *Suppose $\eta = \gamma + lT$ is a Darboux transform of γ with twist parameter r and distance l, and $\hat{\eta} = \gamma + \lambda\hat{T}$ is a Darboux transform of γ with twist parameter ρ and distance λ, then $\eta + \lambda\tilde{T}$ with*

$$\tilde{T} = \big(\lambda\hat{T} - \rho + r - lT\big)\big((\rho - r + lT)\hat{T} + \lambda\big)^{-1} \tag{1.19}$$

is a Darboux transformation of η with twist parameter ρ and distance λ.

Proof. Note that $\tilde{T}_i$ is obtained by applying the quaternionic fractional linear transformation represented by the matrix $V_i(\lambda, \rho)$ to $\hat{T}_i$. Let us write $\tilde{T}_i = V_i(\lambda, \rho)\hat{T}_i$ for short. Since $\hat{\eta}$ is a Darboux transform of γ with twist parameter ρ and distance λ, Equation (1.16) says that $\hat{T}_{i+1} = U_i(\lambda, \rho)\hat{T}_i$. Now Theorem 1 implies $\tilde{T}_{i+1} = \tilde{U}_i(\lambda, \rho)\tilde{T}_i$ and hence (again by Equation (1.16)), $\eta + \lambda\tilde{T}$ is a Darboux transformation of η with twist parameter ρ and distance λ. □

Even if γ is a closed curve, the curves obtained by iterating (1.13) will in general not close up. However, we will see that any closed curve has generically two closed Darboux transforms.

The fractional linear transformations $f_i : T_i \mapsto T_{i+1}$ that are represented by the matrices $U_i(l, r)$ have the special property that they map the unit sphere $S^2 = \{q \in \operatorname{Im}\mathbb{H} \,|\, q^2 = -1\}$ to itself. This follows directly from (1.13). Hence the restrictions $f_i|_{S^2}$ are Möbius transformations of S^2. In fact, they are orientation preserving Möbius transformations: By continuity, it is enough to check this for a particular value of r and l; and for $r = 0$, $l = 0$ one obtains $T_{i+1} = S_i T_i S_i^{-1}$, which is a 180° rotation with axis S_i.

In order to find for given l, r the closed Darboux transforms of γ, one has to look for choices of the initial unit vector T_0 such that the recursion (1.15) generates a sequence with period n, i.e. $T_0 = T_n$. The composition $f_{n-1} \circ \ldots \circ f_0$, which maps $T_0 \mapsto T_n$, is represented by the monodromy matrix

$$H_{l,r} = U_{n-1}(l, r) \cdots U_2(l, r) U_1(l, r) U_0(l, r).$$

It is itself an orientation-preserving Möbius transformation of the unit sphere S^2 onto itself. For special cases (we will see below that this cannot happen for all l, r) this Möbius-transformation could be the identity, but in general it will have exactly two fixed points (counted with multiplicity).

With each closed curve γ we have thus associated a monodromy map $f_{n-1} \circ \ldots \circ f_0$. T_0 will be a fixed point if and only if $\begin{pmatrix} T_0 \\ 1 \end{pmatrix}$ is an eigenvector of the monodromy matrix $H_{l,r}$. The following theorem is an immediate consequence of Theorem 1.

Theorem 3. *Suppose $\eta = \gamma + lT$ is a closed Darboux transform of γ with distance l and twist parameter r. Then for all λ and ρ, the monodromy matrix $H^\eta_{\lambda,\rho}$ of η is conjugate to the monodromy matrix $H_{\lambda,\rho}$ of γ:*

$$H^\eta_{\lambda,\rho} = V_0(\lambda, \rho) H_{\lambda,\rho} V_0(\lambda, \rho)^{-1}. \tag{1.20}$$

This means that if $\begin{pmatrix} \hat{T}_0 \\ 1 \end{pmatrix}$ is an eigenvector of $H_{\lambda,\rho}$, then $V_0(\lambda, \rho) \begin{pmatrix} \hat{T}_0 \\ 1 \end{pmatrix}$ is an eigenvector of $H^\eta_{\lambda,\rho}$.

Moreover, one can compute all closed Darboux transforms of η without having to solve an eigenvalue problem, even without iterating the f_i. Indeed, by Theorem 2, all closed Darboux transforms of η are given by (1.19).

Theorem 3 implies that apart from the edge lengths there are many other quantities connected with closed polygons that are invariant under Darboux transforms: For each λ, ρ the conjugacy class of the monodromy matrix $H_{\lambda,\rho}$ is invariant. We will show that this implies a nice geometric invariant: The area vector of a closed polygon turns out to be invariant under Darboux transformations (Theorem 4).

To derive the invariance of the area vector from the invariance of the conjugacy class of the monodromy matrix, we equip the set of quaternionic 2×2-matrices of the form

$$\begin{pmatrix} a & -b \\ b & a \end{pmatrix}, \quad a, b \in \mathbb{H} \tag{1.21}$$

with the structure of a $\mathbb{C}$-algebra that is isomorphic to $gl(2,\mathbb{C})$. First note that a quaternionic 2×2-matrix is of the form (1.21) precisely if it commutes with

$$J=\begin{pmatrix} 0 & -1\\ 1 & 0\end{pmatrix}.$$

Define the multiplication of such a matrix with a scalar $\lambda+i\rho\in\mathbb{C}$ by

$$(\lambda+i\rho)A=(\lambda I+\rho J)A, \tag{1.22}$$

where I is the identity matrix.

The complex multiples of the identity are then

$$Z=(\lambda+i\rho)I=\lambda I+\rho J=\begin{pmatrix} \lambda & -\rho\\ \rho & \lambda\end{pmatrix}. \tag{1.23}$$

Thus we can write $U_i(\lambda,\rho)$ and $V_i(\lambda,\rho)$ as

$$U_i(\lambda,\rho)=(\lambda+i\rho)I+J\begin{pmatrix} S_i & 0\\ 0 & S_i\end{pmatrix},$$

$$V_i(\lambda,\rho)=(\lambda+i\rho)I+J\begin{pmatrix} -r+lT_i & 0\\ 0 & -r+lT_i\end{pmatrix}.$$

Remark. This means we can combine λ and ρ into one complex spectral parameter $\lambda+i\rho$.

Equation (1.23) also implies that the trace-free complex matrices in $gl(2,\mathbb{C})$ correspond to those matrices of the form (1.21) with $a,b\in \mathrm{Im}\mathbb{H}$. Further, a matrix of the form (1.21) has $a,b\in\mathrm{Im}\mathbb{H}$ precisely if its square is a matrix of the form (1.23), that is, a (complex) multiple (with multiplication defied by (1.22)) of the identity. Identifying $\mathbb{C}$ with the matrices of the form (1.23) we obtain

$$\frac{1}{2}\operatorname{tr}_{\mathbb{C}}\begin{pmatrix} a & -b\\ b & a\end{pmatrix}=\operatorname{Re}a+(\operatorname{Re}b)J$$

and

$$\det{}_{\mathbb{C}}\begin{pmatrix} a & -b\\ b & a\end{pmatrix}=\frac{1}{2}((\operatorname{tr}A)^2-\operatorname{tr}A^2)=|a^2|-|b^2|+2\langle a,b\rangle J.$$

In particular

$$\det{}_{\mathbb{C}}\begin{pmatrix} l & -r-S\\ r+S & l\end{pmatrix}=l^2-r^2-|S|^2+2lrJ,$$

which vanishes precisely when $r=0, l=\pm|S|$. Using the notation

$$diag(S):=\begin{pmatrix} S & 0\\ 0 & S\end{pmatrix}$$

for $S \in \mathbb{H}$ we can express $H_{\lambda,\rho}$ as

$$H_Z = (Z + J\,diag(S_{n-1})) \cdots (Z + J\,diag(S_0)),$$

with Z given by (1.23). Hence $\det_{\mathbb{C}} H_Z$ is a complex polynomial of degree $2n$ with zeroes precisely at $Z = \pm|S_0|, ..., \pm|S_{n-1}|$. By Theorem 3 this determinant is invariant under Darboux transforms. This just corresponds to the fact that the edge lengths are invariant by construction. Non-trivial further invariants come from the complex polynomial

$$P(Z) = \mathrm{tr}_{\mathbb{C}}\, H_Z$$

of degree n. Let us look at the polynomial coefficients of H_Z itself:

$$H_Z = \sum_{k=1}^{n} Z^k A_{n-k},$$

where

$$A_k = J^k \sum_{n-1 \geq j_1, ..., j_k \geq 0} diag(S_{j_1} \cdots S_{j_k}).$$

In particular,

$$\begin{aligned} A_0 &= I, \\ A_1 &= J^k \textstyle\sum_{k=0}^{n-1} diag(S_k) = 0, \\ A_2 &= -\textstyle\sum_{n-1 \geq i > j \geq 0} diag(S_i S_j). \end{aligned}$$

That is, A_2 is a diagonal matrix with both diagonal entries equal to

$$q = -\sum_{n-1 \geq i > j \geq 0} S_i S_j.$$

The real part of q is

$$\begin{aligned} \mathrm{Re}(q) &= \textstyle\sum_{n-1 \geq i > j \geq 0} \langle S_i, S_j \rangle = \frac{1}{2} \sum_{i \neq j} \langle S_i, S_j \rangle = \frac{1}{2} |\sum_{i=0}^{n-1} S_i|^2 - \frac{1}{2} \sum_{i=0}^{n-1} |S_i|^2 \\ &= -\tfrac{1}{2} \textstyle\sum_{i=0}^{n-1} |S_i|^2. \end{aligned}$$

This is a function of the edge lengths and therefore not interesting. The imaginary part of q is given by

$$\begin{aligned} 2A := \mathrm{Im}(q) &= -\textstyle\sum_{i>j} S_i \times S_j \\ &= \textstyle\sum_{j=1}^{n-1} \left(\sum_{i=1}^{n-1} S_i \right) \times S_j \\ &= \textstyle\sum_{j=1}^{n-1} (\gamma_j - \gamma_0) \times (\gamma_{j+1} - \gamma_j) \\ &= \textstyle\sum_{j=1}^{n-1} (\gamma_j - \gamma_0) \times (\gamma_{j+1} - \gamma_0). \end{aligned}$$

This invariant A is just the area vector. The following proposition (with obvious proof) clarifies its geometrical meaning.

Proposition 1. *Let $a \in \mathbb{R}^3$ be a unit vector, $|a| = 1$, and endow the plane $a^\perp$ with the volume form*

$$\det_{a^\perp}(X, Y) := \det_{\mathbb{R}^3}(a, X, Y).$$

Then the area enclosed by the orthogonal projection $\hat{\gamma}$ of the polygon γ

$$\hat{\gamma}_n = \gamma_n - \langle \gamma_n, a \rangle a$$

is equal to $\langle M, a \rangle$.

This explains the name *area vector*: It encodes all the projected areas.

Theorem 4. *The area vector A is invariant under Darboux transforms.*

Proof. By (1.20), the monodromy matrix of the Darboux transformed curve η,

$$H^\eta_Z = \sum_{k=0}^{n} Z^k A^\eta_{n-k},$$

satisfies

$$H^\eta_Z(Z + J(-rI + l\, diag T_0)) = (Z + J(-rI + l\, diag T_0))H_Z \tag{1.24}$$

Using

$$\begin{aligned} H_Z &= Z^n + Z^{n-2}A_2 + ... + A_0, \\ H^\eta_Z &= Z^n + Z^{n-2}A^\eta_2 + ... + A^\eta_0 \end{aligned}$$

and comparing the Z^{n-2}-coefficients in both sides of (1.24) we obtain $A^\eta_2 = A_2$. □

Finally we consider the continuum limit of smooth curves $\gamma : S^1 \to \mathbb{R}^3$ and indicate why Darboux transforms with small parameters l, r do indeed converge to the smoke ring flow (1.1). The continuum limit of (1.13) is obtained by replacing S by hS and then computing $T' := \frac{d}{dh}\big|_{h=0} T_h$. The resulting differential equation is

$$T' = (TS - ST)(-r + lT)^{-1}$$

or

$$T' = \frac{2}{r^2 + l^2} T \times (lT \times S - rS), \tag{1.25}$$

where $S : \mathbb{R} \to \mathbb{R}^3$ is given by

$$\gamma' = S.$$

One can check that, as expected, the transformed curve $\eta = \gamma + lT$ satisfies

$$|\eta'| = |\gamma'|.$$

The monodromy of the ODE (1.25) is a Möbius transformation of S^2 that generically has exactly two fixed points. Thus, for generic parameters l and r a space curve γ has exactly two closed Darboux transforms.

Assume now that we have for $r = -l$ a family of such closed Darboux transforms η_l that depend analytically on l. Then we reparametrise η_l as

$$\gamma_l(s) := \eta_l(s-l) = \gamma(s-l) + lT_l(s-l). \tag{1.26}$$

Then $\gamma_0 = \gamma$ and comparing coefficients of l in the power series expansion of (1.26) we obtain

$$\frac{\partial}{\partial l}\Big|_{l=0} \gamma_l = 0, \qquad \frac{\partial^2}{\partial l^2}\Big|_{l=0} \gamma_l = \gamma' \times \gamma''.$$

Hence

$$\gamma_l - \gamma_0 = l^2 \gamma' \times \gamma'' + O(l^3).$$

A small time-step Δt of the smoke ring flow is therefore approximated by a Darboux transform with length l given by $l^2 = \Delta t$.

Remark. In order to eliminate the reparametrising effect of the Darboux transforms it is convenient to apply first a Darboux transform with parameters l and $-r$ followed by a reverse Darboux transform with parameters l and r. This will cancel out the (first order in t) tangential shift and leave only the (second order in t) smoke ring evolution (see [Hof00]).

1.5 An algorithm for the real time simulation of fluid flow

Based on the theoretic foundations covered in the previous sections, we have implemented the following algorithm for the simulation of fluid flow. Our aim was to develop an algorithm which is fast enough to generate realistic looking computer animations of fluid motion in real time. Figure 1.3 shows a sample screen shot from a simulation which runs smoothly on standard hardware. We assume the vorticity is concentrated along a few vortex rings, which we represent by closed polygons. Their motion is governed by a mixture of the velocity field induced by the polygonal vortex rings via the smoothed Biot-Savart formula (1.10) of Section 1.3, and Darboux transformations which approximate a time step of the polygonal smoke ring flow as explained in Section 1.4. The rationale behind this scheme is that the velocity field induced by an edge of a polygonal vortex filament is zero on that edge itself. Thus, the adjacent edges do not contribute to the velocity of a vertex. The Darboux transforms make up for this lack of local interaction. The following is a summary description of the algorithm. Details (in particular how we set the parameters r_i and l_i of the Darboux transformation) are given below.

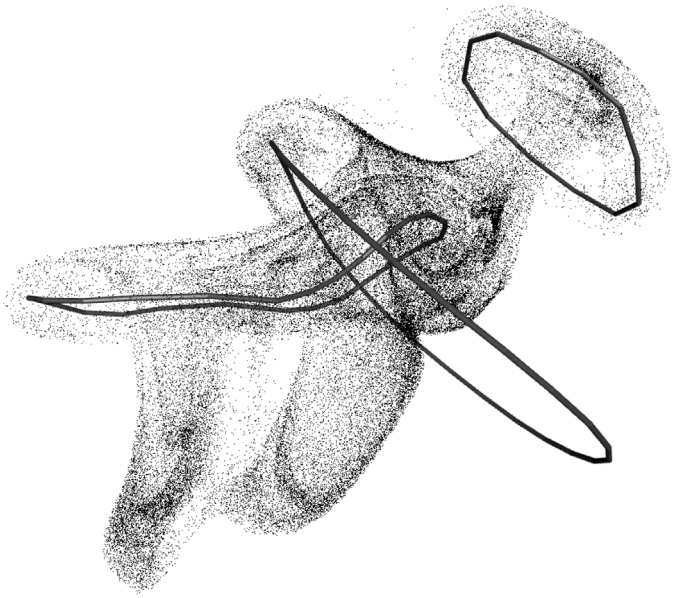


Figure 1.3: 256^2 *fluid particles evolving under the influence of three polygonal vortex filaments.*

input:

- positions γ_{ij} of the jth vertex of the ith polygonal vortex filament γ_i, where $i = 1 \dots m$, $j = 1 \dots n_i$.
- strengths Γ_i and smoothing (thickness) parameters a_i of the vortex filaments.
- positions $p_i \in \mathbb{R}^3$ of advected particles, where $i = 1 \dots k$.
- time-step Δt .

loop:

1 Compute a double Darboux transform η_i with parameters $\mp r_i, l_i$ of each polygon γ_i. $\gamma_{ij} \leftarrow \eta_{ij}$.

2 Solve $\dot{\gamma}_{ij} = u(\gamma_{ij})$ for time-step Δt, where $u(x)$ is the velocity field obtained by the smoothed Biot-Savart formula (1.10).

3 Update the particle positions p_i by solving $\dot{p}_i = u(p_i)$ for time-step Δt.

In Step 1, we determine the parameters l_i and r_i as follows. The amount of smoke ring flow needed to make up for the lack of local interaction depends on the thickness a_i, the number of edges n_i and the total length L_i of γ_i. Since we do not know the correct speed for an arbitrary polygon, we determine the parameters for the test case of a regular n_i-gon with same strength, thickness and total length. We choose the parameters in such a way that for the regular n_i-gon the sum of self-induced velocity

from the Biot-Savart formula (1.10) plus the effect of a double Darboux transform coincides with the analytically known speed U_i for a circle with same length L_i:

$$U_i = \frac{\Gamma_i}{2L_i}\left(\ln\frac{4L_i}{\pi a_i} - 1\right), \tag{1.27}$$

compare [Saf92, p. 212]. We compute the self-induced speed $\tilde{U}_i$ of the n_i-gon by evaluating the smoothed Biot-Savart formula (1.10) at one vertex for all edges of the n_i-gon. This speed is slower than U_i because the adjacent edges have no influence on a vertex, see Section 1.3. Now we choose r_i and l_i such that a double Darboux transformation translates the regular n_i-gon by a distance of $(U_i - \tilde{U}_i)\,\Delta t$. A single Darboux transform of the regular n_i-gon is a translation in binormal direction plus a non-zero rotation about the centre axis. The rotation cancels out for a double Darboux transform and is therefore arbitrary. We choose the rotation angle to be $2\pi/n_i$, which leads to the following formulas for l_i and r_i:

$$l_i = \sqrt{(L_i/n_i)^2 + \sigma_i^2}\,, \qquad r_i = \sigma_i \cot(\pi/n_i)\,,$$

where we have abbreviated $\frac{1}{2}(U_i - \tilde{U}_i)\,\Delta t$ by σ_i.

In Step 2, we use the fourth order Runge-Kutta scheme (RK4) to solve the ordinary the differential equation $\dot{x} = u(x)$ for the time-step Δt. To advect the large number of particles in Step 3 we use second order Runge-Kutta (RK2), where we use the two polygon positions after Step 1 and Step 2 as intermediate values. To improve performance further, this step is computed on the computer's graphics chip (GPGPU).

Evaluating $u(x)$ via Equation (1.10) is unproblematic, because the integral on the right hand side can be solved explicitly for straight line segments; see Equation (1.11) in Section 1.3.

2 Real-time interactive simulation of smoke using discrete integrable vortex filaments

Steffen Weißmann, Ulrich Pinkall

Proc. Vir. Real., Inter. & Phys. Sim. 2009, pp. 1–10.

We present a fluid solver for the real-time interactive simulation of inviscid, ideal fluid flow. The simulation is based on the evolution of discrete vortex filaments, which allow a dramatic increase of detail and performance compared to traditional methods used in Computer Graphics. As a fully Lagrangian method the simulation is not restricted to a fixed domain and does not suffer from numerical dissipation. Vortex filaments arise naturally in real flows and thus provide an excellent building block for modeling realistic smoke. We present a GPU-based implementation which allows the interactive experimentation with 3D fluid flow on desktop computers and also in distributed immersive virtual environments.

2.1 Introduction

Real-time simulation of 3D smoke is an important ingredient for virtual environments in general and computer games in particular. Nevertheless computational challenges have so far prevented the widespread implementation of such simulations. While it is possible to achieve realistic and highly resolved 3D smoke animations in extensive offline simulations, real-time applications are still missing the desirable detail and realism.

We describe a 3D method that is highly efficient while allowing a tremendous amount of detail. The method is based on the simulation of the evolution of vortex filaments. Vorticity originates as 2–dimensional vortex sheets that tend to roll up into complicated 1–dimensional structures. The resulting visual complexity easily exceeds the level of detail that can be achieved with real-time grid based methods, as in Figure 2.1. The use of vortex filaments provides an efficient method to capture the complexity of smoke with very sparse data. The whole fluid velocity field is defined by the vortex filaments and can be used to advect arbitrary marker particles. Besides the application for real-time simulation, the method provides a significant improvement for the workflow of effects artists designing smoke animations. It allows to obtain an immediate preview of the fluid motion. The final animation can then be rendered with an arbitrary number of particles – without affecting the fluid motion at all.

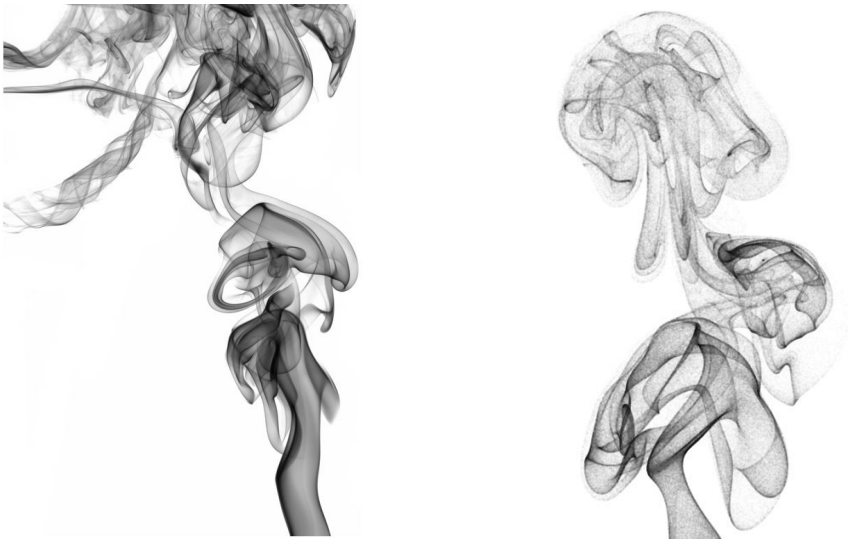


Figure 2.1: *Comparison of a photograph (left, from [Jef]) with our simulation (right). The* 396^2 *particles are rendered as unshaded transparent GL points.*

2.2 Related work and contribution

Real-time 3D smoke simulations have so far been restricted to low resolution grid-based Eulerian methods (mostly based on Stam's *Stable Fluids* [Sta99] with vorticity confinement [FSJ01]) or to algorithms based on 2D reductions [KW05]. Recent progress with GPU techniques allows relatively large 3D grids [LLW04, CLT07, Yan09a, Yan09b] for such methods. Model reduction [TLP06] can significantly speed up such simulations, but it requires extensive precomputations as well as giant storage, which limits its practical applicability to very small grids. Also hybrid methods based on [SRF05] were applied to real-time applications, for instance in [EVG07].

Smoothed Particle Hydrodynamics methods [SF95, DG96, MCG03, BT07] can be used at interactive rates with a respectable number of particles thanks to GPU techniques [ZSP07, YWH$^+$09]. These methods are used with great success for the simulation of water and other liquids. They are however not well-suited for inviscid fluids such as smoke.

The use of vortex filaments as basic primitives for modeling 3D smoke was pioneered by Angelidis and Neyret [AN05, ANSN06]. They however simulated the motion of smoke using a kernel function that differs significantly from the physical system, resulting in an incorrect asymptotic falloff of the velocity field.

Our method is based on the direct discretization of the actual physics. The fluid motion is described by a Hamiltonian system, which guarantees the conservation of both energy and momentum. We are able to explicitly integrate the correct kernel function for the discretized vortex filaments. This yields not only an important

increase of physical realism but at the same time in fact a reduction of the computational costs. Our second contribution is the following: When discretizing the vortex filaments as polygons, their velocity will be significantly underestimated when calculated by the integral kernel alone: Both adjacent edges do not contribute at all to the velocity of a vertex. This effectively results in modeling vortex filaments that have a thickness equal to the edge lengths. This means that excessively many edges would be required to model thin (and therefore fast) filaments. We compensate this discretization artifact by incorporating a discrete evolution equation for polygons (developed in [PSW07]), that excellently captures the locally induced speed. Also this local contribution to the evolution conserves energy and momentum and can be computed efficiently. We are therefore able to model thin vortex filaments using only a small number of edges.

We will describe an interactive, GPU accelerated implementation for use in distributed, immersive virtual environments and on desktop computers.

2.3 Physics of vortex filaments

Irrotational regions will usually be rare given an arbitrary divergence free velocity field. The fluids that we are going to simulate (represented by a small number of vortex filaments) are mostly irrotational, except for thin tubular regions around the filaments. Therefore some explanation is required why it is nevertheless a good idea to model realistic fluids in this way: First, in inviscid fluid, there is no way to increase the volume that contains vorticity. Second, 2–dimensional vortex sheets tend to roll up into 1–dimensional core structures [RVJ00] (see Figure 2.2). This implies that vortex filaments will actually stay filaments, since any flattening of the core structure will result in a rollup. Most important, vorticity usually originates as vortex sheets, for instance at boundary layers or around jets. Thus we can excellently model realistic fluids, using a small number of vortex filaments.

Besides the physical motivation, vortex filaments are a very convenient model: They move along with the motion of the fluid, they do not split, join or vanish. They do not change their strength, so only the geometry of the filaments needs to be tracked. From an artist's point of view, filaments give an intuitive primitive to model fluid motion. Another very important property of vortex filaments is that they ensure that the *vorticity field* is always divergence free. In contrast, vortex particle methods [PK05] only guarantee a divergence free velocity field – the vorticity field will not stay divergence-free when the vortex structures break up.

2.3.1 Mathematical description

The motion of an inviscid, ideal fluid is described by a time-dependent divergence free velocity field u, obeying *Euler's equation*. The vorticity field $\omega = \operatorname{curl} u$ is again a divergence free vector field, and we assume that it is compactly supported. The

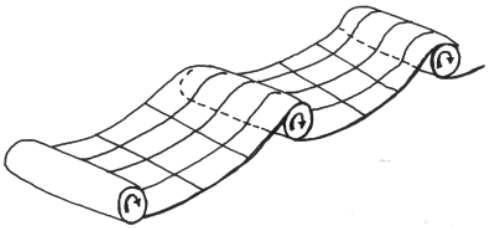


Figure 2.2: *Sketch of the rollup of a vortex sheet to vortex filaments, from [RVJ00].*

vorticity field lines are either closed or end on boundaries, which is not a strict mathematical consequence but based on the observation how vorticity is created in real fluid flow. Euler's equation in vorticity form

$$\dot{\omega} = \operatorname{curl}(\omega \times u) \tag{2.1}$$

states that the vorticity field is forward advected by the velocity field just as the fluid material. The right hand side of Equation (2.1) is the *Lie bracket* $[\omega, u]$ of divergence free vectorfields in $\mathbb{R}^3$.

From the vorticity field ω we can reconstruct the velocity field u using the formula

$$u(x) = -\frac{1}{4\pi} \int_{\mathbb{R}^3} \frac{x - z}{\|x - z\|^3} \times \omega(z)\, dz. \tag{2.2}$$

This is an integral over the whole of $\mathbb{R}^3$, but with the restriction to vorticity fields that are supported on tubular neighborhoods of closed space curves γ_k, Equation (2.2) reduces to a sum of line integrals – the *Biot-Savart law*:

$$u(x) = \sum_k -\frac{\Gamma_k}{4\pi} \oint \frac{x - \gamma_k(s)}{\|x - \gamma_k(s)\|^3} \times \gamma_k'(s)\, ds, \tag{2.3}$$

Here, Γ_k is the *strength* of the filament, it is the flux of vorticity across any cross section of the filament core. The strength is constant along the filament and also constant in time, by Helmholtz's theorems and also as a direct consequence of Equation (2.1).

For the following discussion we will regard one single vortex filament γ, the results can be transferred to a set of filaments by superposition (i.e. summation) of the individual vector fields.

The dynamical system we want to simulate is

$$\dot{\gamma}(s) = u(\gamma(s)) \tag{2.4}$$

where u is given by Equation (2.3). The problem is that the velocity field u diverges logarithmically for points on the γ. Assuming a small but finite tube radius of the vortex filament γ reveals that the filament moves according to the *smoke ring flow*

$$\dot{\gamma}(s) = K\gamma'(s) \times \gamma''(s), \tag{2.5}$$

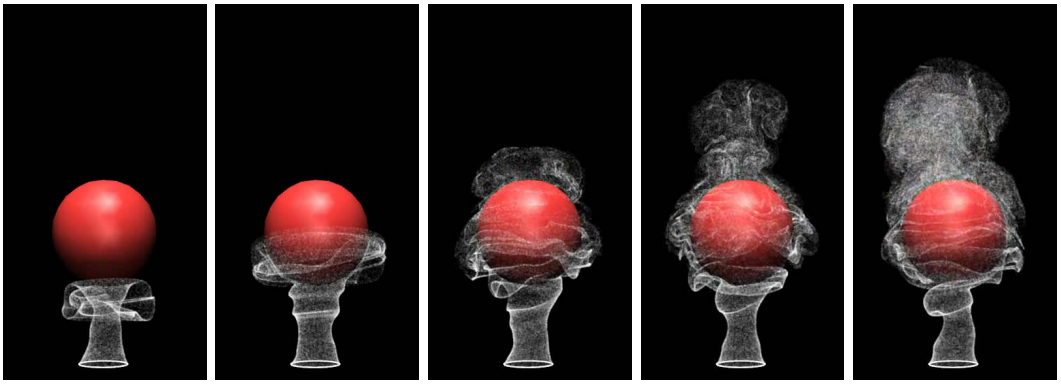


Figure 2.3: *Flow around a sphere using image vorticity.*

where the constant K is proportional to the strength Γ of γ. The smoke ring equation (2.5) is obtained by a renormalization from Equation (2.4), thus it conserves both energy and momentum of the original system. It was discovered at the beginning of the 20th century by Da Rios [SDR06], its integrability was found by Hasimoto [Has72]. A historical review is given in [Ric91].

The correct speed of a smooth vortex filament with small thickness a is obtained by applying a *cut off* to the right-hand-side of Equation (2.4): A small portion around the evaluation point is excluded from the integration domain. This is known [Saf92] to be equivalent to replacing the singular Biot-Savart kernel by the *Rosenhead-Moore* kernel, which gives the velocity field

$$u(x) = -\frac{\Gamma_i}{4\pi} \oint \frac{x - \gamma(s)}{\sqrt{a^2 + \|x - \gamma(s)\|^2}^3} \times \gamma'(s)\, ds, \tag{2.6}$$

where a (the thickness parameter) depends on the core radius of the filament. Another way to obtain Equation (2.6) is to apply a fixed convolution kernel to the singular vorticity field which is concentrated along the filament (see [PSW07]). For $a > 0$ this velocity field has no singularities at all, which makes its evaluation unproblematic.

The singular Biot-Savart formula (2.3) causes two problems to the numerics: The singular behavior at the filaments and the fact that it is hard to integrate analytically – already for circular filaments it results in elliptic functions. Angelidis and Neyret [AN05] have addressed these issues by a modification of the kernel function. To ease analytical integration, they have changed the exponent in the denominator from 3 to 4. The resulting equation of motion is no longer related to the physical system, for instance the asymptotic falloff is different and the system does not scale correctly.

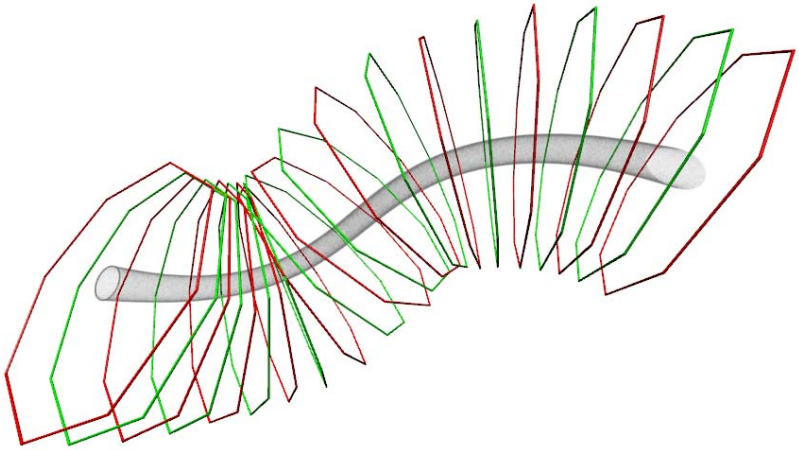


Figure 2.4: *Designing irrotational background velocity fields using static vortex filaments. The particles were injected from a circular curve at the left and advected by the induced velocity field from the static filaments.*

2.3.2 Boundary conditions and background flow

We will not deal with boundary conditions here. Slippy boundaries can be modeled using *image vorticity* [Saf92], for instance an infinite ground plane (as in [AN05]) is obtained by mirroring all filaments at the plane. A sphere can be modeled by inversion of the vortex filaments at the sphere (see Figure 2.3). The resulting velocity field will be tangent to the surface, but in contrast to real boundaries, no vorticity is created. We will not go into details about these special cases for dealing with boundaries. We will describe the incorporation of arbitrary obstacles in future work.

As a background flow, any irrotational field can be added to the velocity field given by Equation (2.6). In this way simple effects of wind or temperature lift can be simulated. It is however not possible to combine this with the image vorticity construction described above, except for the case that the background flow is already tangent to the boundaries. On the other hand it is difficult to design an irrotational background field with certain features. Therefore we suggest the use of *static* vortex filaments, that contribute to the velocity field but do not get advected by the flow: Such filaments generate a velocity field that is irrotational except for some small region around the filament. In this way one can easily design a background field that for instance guides the fluid motion along a certain path, see Figure 2.4. This gives artists a very intuitive tool to control large scale features of the fluid motion. In addition, such background flows can be combined with the simple image vorticity constructions.

2.3.3 Conservation laws

The smooth system we have described is a *Hamiltonian system* (see [EM70, AK98]), preserving the energy

$$E = \sum_{i,j} \frac{\Gamma_i \Gamma_j}{8\pi} \oint \oint \frac{\langle \gamma_i(s), \gamma_j(\tilde{s}) \rangle}{\sqrt{a^2 + \|\gamma_i(s) - \gamma_j(\tilde{s})\|^2}} \, ds \, d\tilde{s}. \tag{2.7}$$

This energy has a nice geometric interpretation: It is the sum of pair-wise fluxes, that the filaments induce to each others. Physically, the energy is very closely related to the kinetic energy $T = \int_{\mathbb{R}^3} \langle u(x), u(x) \rangle dx$, in fact it is the kinetic energy of a fluid with the same vortex filaments, but a slightly different core structure and radius. Another constant of motion is the *hydrodynamic impulse*

$$A = \sum_i \frac{1}{2} \oint \gamma_i(s) \times \gamma_i'(s) \, ds \tag{2.8}$$

whose geometric interpretation (for closed filaments) is the following: For any plane $n^\perp$ with unit normal vector n, the signed area of the orthogonal projection of the filaments onto that plane is $\langle A, n \rangle$. As a reference for Equations (2.7) and (2.8) see [Saf92].

This invariant has an important consequence for the geometry of the filaments (see [Cho91]): Vortex filaments stretch rapidly while evolving under the fluid flow. At the same time, their projected area (for instance onto $A^\perp$) stays constant. This is only possible when the filaments fold, leading to very long and highly curved filaments. While folding, nearby sections of the vortex filaments tend to align anti-parallel, resulting in effectively canceling out each others' contribution to the overall velocity field. A special case is the creation of *hairpins* [Cho90, Cho93]. In slightly viscous fluids, vortex reconnections occur at such anti-parallel aligned filaments.

2.4 Polygonal discretization of smooth filaments

We start with the smooth Hamiltonian system described in Section 2.3.1 and discretize it by replacing the smooth vortex filaments by polygons and advecting the polygon vertices according to the dynamical system (2.4). In Section 2.4.1 we will describe in detail the explicit integration of the velocity field (Equation (2.6)) that is generated by a straight filament edge. The resulting velocity field is obtained by summing up over all edges. For a single step of the simulation we will integrate the ODE (2.4) over the current time step Δt using a standard forward integration scheme.

It turns out that, due to the discretization as polygons, the locally induced velocity of the filaments gets lost. This happens since adjacent edges do not contribute to the velocity at a vertex. The actual locally induced speed is proportional to the smoke ring flow (2.5), which we cannot compute directly for polygons. Instead, we apply

one step of a time-discrete evolution equation (the doubly-discrete smoke ring flow for polygons derived in [PSW07]) that acts as a time integrator of the smoke ring flow for polygons. In Section 2.4.3 we will explain how to compute a single step of the evolution and how to determine the necessary parameters.

2.4.1 Explicit integration of the Biot-Savart law

If γ is a piecewise linear parametrization of a closed polygon, on each edge we have $\gamma'' = 0$ and we find an explicit anti-derivative for the integrands of equation (2.6):

$$\frac{\langle \gamma, \gamma' \rangle}{\sqrt{a^2 + |\gamma|^2}\ (|\gamma'|^2 a^2 + |\gamma \times \gamma'|^2)} \gamma \times \gamma'. \tag{2.9}$$

Here we have abbreviated $x - \gamma(s)$ to γ, $\gamma'(s)$ to γ' and the prime is derivation with respect to s.

More explicitly, regard a single edge with startpoint γ_0 and endpoint γ_1: Then

$$\gamma(s) = \gamma_0 + s(\gamma_1 - \gamma_0),$$

and the cross-product

$$\gamma(s) \times \gamma'(s) = \gamma_1 \times \gamma_0$$

does not depend on s. The generated velocity field at zero is

$$u(0) = \langle \gamma_0, \gamma_1 \rangle \frac{\frac{\|\gamma_0\|^2}{\sqrt{a^2+\|\gamma_0\|^2}} - \frac{\|\gamma_1\|^2}{\sqrt{a^2+\|\gamma_1\|^2}}}{a^2\|\gamma_1 - \gamma_0\|^2 + \|\gamma_1 \times \gamma_0\|^2} \gamma_1 \times \gamma_0, \tag{2.10}$$

any other point x can be evaluated by replacing γ_i by $\gamma_i - x$. This simple formula is actually surprising: Even for circular filaments the integration of the Biot-Savart formula (2.3) requires elliptic functions. Also the modified kernel used in [AN05] turns out to be much more expensive to evaluate, it includes angle-computations that require evaluation of the arctan-function.

2.4.2 Local effects at vertices

The two adjacent edges have no influence at all on the velocity of a vertex. This is roughly equivalent to modeling vortex tubes of thickness equal to the edge length of the polygon. Using this model we would be unable to model thin (and therefore fast) filaments without using excessively many edges for each polygon.

Asymptotic analysis of the velocity field at such a vertex induced by its two adjacent edges (see [Wei06]) reveals the following: The contribution of local effects behaves like the smoke ring flow (2.5) and the resulting equation of motion for a vertex γ_i of a polygonal vortex filament γ is then

$$\dot{\gamma}_i = u(\gamma_i) + \lambda \kappa_i b_i, \tag{2.11}$$

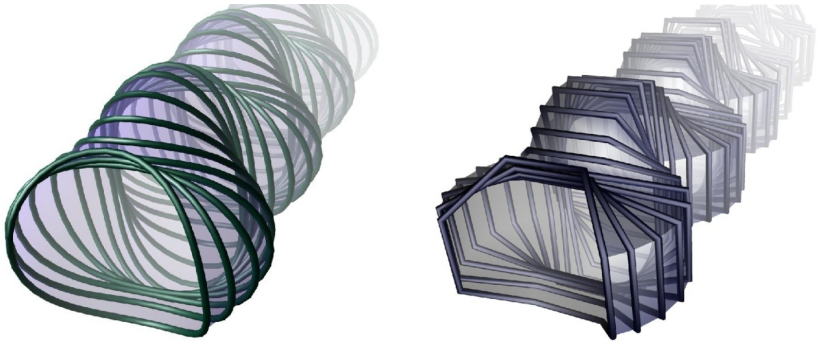


Figure 2.5: *An oval curve evolving under the smoke ring flow and its discrete analog, from [Hof08].*

where u is given by summing up Equation (2.6) for all filaments using (2.10), $\kappa_i b_i$ denotes curvature times binormal at γ_i, and λ is constant depending on a. The mathematical notion of curvature is only available for arc-length parametrized polygons, i.e. all edges have equal length. Since the non-local effects quickly destroy any arc-length parametrization we can not evaluate (2.11) directly.

On the other hand, it is known that the doubly discrete smoke ring (or Hasimoto) flow [Hof00, Hof08, PSW07] captures excellently the qualitative behavior of the smooth smoke ring flow (2.5) for polygons, see Figure 2.5. Thus we apply it as an integrator of the locally induced velocity term κb.

2.4.3 The doubly-discrete smoke ring flow

The doubly-discrete smoke ring flow is a discrete evolution equation for closed polygons. This means, given a closed polygon γ, we can compute a new polygon $\tilde{\gamma}$, that corresponds to the time-evolution of the polygon under the smooth smoke ring flow, see [Hof00, Hof08, PSW07]. One step of this evolution is obtained by computing two successive closed *Darboux transforms*. The Darboux transform is obtained from the initial polygon by elementary geometric constructions (Darboux steps). This will be described in Section 2.4.3.1.

A single step of the evolution depends on two parameters, which determine the corresponding time step of the smooth evolution. In Section 2.4.3.3 we will describe how to determine the required parameters in order to match the current time step.

2.4.3.1 Darboux transforms of polygons

We start with the closed polygon γ, with vertices $\gamma_1, \gamma_2, \ldots, \gamma_n$ and edges $S_1 = \gamma_2 - \gamma_1$, $S_2 = \gamma_3 - \gamma_2, \ldots, S_n = \gamma_1 - \gamma_n$. The Darboux transform η of γ is also a polygon with the same number of edges, and same edge lengths. But η is usually

not closed, thus it has the vertices $\eta_1, \eta_2, \ldots, \eta_n, \tilde{\eta}_1$ and edges $\tilde{S}_1 = \eta_2 - \eta_1, \tilde{S}_2 = \eta_3 - \eta_2, \ldots, \tilde{S}_n = \tilde{\eta}_1 - \eta_n$. The Darboux transform has two real parameters, namely the *distance* parameter $l > 0$ as well as the *twist* parameter r. The polygon η is obtained by the following procedure:

- Choose a start point η_1 of η, with distance l from γ_1. We denote the distance vector $\eta_1 - \gamma_1$ by lT_1, where T_1 is a unit vector, and l the distance.
- Map η_1 to the next point η_2 of η by a single *Darboux step*, see below. The step depends only on the difference vector lT_1 and the first edge S_1 of γ, and it has two important properties:
  - The distance between η_2 and γ_2 is also l:
  $$\eta_2 - \gamma_2 = lT_2.$$
  - The edge lengths of η and γ are equal:
  $$\|S_1\| = \|\tilde{S}_1\|.$$
- Now that we have obtained η_2 we apply a Darboux step to η_2 to obtain η_3 and so on. More precisely, we can replace the indices 1 and 2 by i and $i+1$, which allows us to determine the next vertex η_{i+1} as soon as we have determined its predecessor η_i. Thus we transport the initial start point η_1 all around the polygon γ, and we finally arrive at $\tilde{\eta}_1$.

A single Darboux step is given as a quaternionic equation, so we identify $\mathbb{R}^3$ with the set of imaginary quaternions $\operatorname{Im}\mathbb{H}$. From the current distance vector lT_i and the current edge S_i of γ, we obtain lT_{i+1} as:

$$lT_{i+1} = (-r + lT_i - S_i)lT_i(-r + lT_i - S_i)^{-1}. \tag{2.12}$$

We can apply this equation iteratively to our initial distance vector lT_1 (between γ_1 and the *start point* η_1 of η) to obtain the distance vector $l\tilde{T}_1$ (between γ_1 and the *end point* $\tilde{\eta}_1$ of η), and we will denote this map by f:

$$f\colon lT_1 \mapsto l\tilde{T}_1. \tag{2.13}$$

For the doubly-discrete smoke ring flow we need to determine a *closed* Darboux transform. Therefore we need to determine the start point η_1 in such a way that η is closed, i.e. $\tilde{\eta}_1 = \eta_1$. This is the case when the initial distance vector lT_1 is a fixed point of f. To compute a fixed point we use the *power method*, i.e. we start with some initial vector with length l and iterate f until convergence.

2.4.3.2 A single Darboux step

Now we describe a single *Darboux step* that determines η_{i+1} from η_i. Look at a single quadrilateral with the edges S_i, lT_{i+1}, $-\tilde{S}_i$, $-lT_i$ (Figure 2.6). Since opposite edges

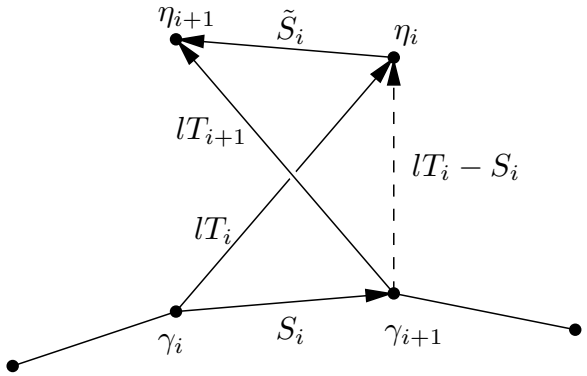


Figure 2.6: *A polygon γ and a single edge $\tilde{S}_i$ of its Darboux transform η.*

have equal length, it is a parallelogram, folded about one of its diagonals. The folding angle about the diagonal between γ_{i+1} and η_i,

$$D_i = \eta_i - \gamma_{i+1} = lT_i - S_i,$$

is required to be

$$\alpha_i = 2 \arctan \|D_i\|/r,$$

where the twist parameter r enters. This rule determines η_{i+1} uniquely. In particular, the newly obtained distance vector lT_{i+1} is obtained by an α_i-degree rotation of the previous distance vector lT_i about the diagonal D_i. By adding lT_{i+1} to γ_{i+1}, we have obtained η_{i+1}.

This construction might be implemented by computing the corresponding rotation matrix and apply it to lT_i, but it is much simpler to implement using quaternions, using Equation (2.12).

2.4.3.3 Parameters for the Darboux transforms

One step of the doubly-discrete smoke ring evolution is obtained by two subsequent Darboux transforms, the first with parameters l and $+r$, the second with parameters l and $-r$. To determine l and r for a polygon γ with n vertices and total length L, we regard a regular, planar n-gon $\tilde{\gamma}$ with the same total length L and compute its self-induced speed:

$$\tilde{U} = \|u(\tilde{\gamma}_i)\|.$$

Note that $u(\gamma_i)$ (given by Equation (2.6)) is the same for each vertex $\tilde{\gamma}_i$ and perpendicular to the n-gons plane.

Now we compare $\tilde{U}$ with the analytically known speed of a smooth circular vortex filament with same length:

$$U = \frac{\Gamma}{2L}\left(\ln \frac{4L}{\pi a} - 1\right). \tag{2.14}$$

The Darboux-transform η of a regular n-gon $\tilde{\gamma}$ is a shifted copy of $\tilde{\gamma}$: It is rotated about the center axis with angle ϕ, and translated along the center axis, with distance d. For the combination of two successive Darboux transforms with parameters l and $\pm r$ the rotations will cancel, and the resulting motion is a pure translation along the center axis, with distance $2d$. The correct speed is obtained when

$$2d = \Delta t\,(U - \tilde{U}).$$

With a choice of the rotation angle $\phi = 2\pi/n$, the parameters turn out to be

$$l = \sqrt{(L/n)^2 + d^2}, \quad r = d\,\cot(\pi/n). \tag{2.15}$$

2.4.4 Summary

We have presented all necessary formulas to compute a single step of the evolution of polygonal vortex filaments. In Equation (2.10) we have presented the explicit formula to evaluate the velocity field induced by the individual edges. We can use this to directly advect the filament vertices for a given time step using a standard forward ODE integration scheme, for instance the Runge-Kutta scheme RK4. But the resulting motion of the discrete filaments does not adequately reproduce the physical system: It underestimates the locally induced speed of the individual filaments. We account for this by applying a single step of the doubly-discrete smoke ring flow evolution to each of the filaments. Therefore one first needs to determine the twist parameter r and the distance parameter l for the given time step, using the Equations (2.15) and (2.14). The step itself is obtained by computing two successive Darboux transforms. This is done by determining a fixed point of the map defined in Equation (2.13), which is itself defined by a sequence of Darboux steps (2.12). The fixed point is obtained by applying the power method.

2.5 Implementation

We will now describe how to implement a fluid simulator using the evolution of discrete vortex filaments. The implementation splits into two independent parts: The evolution of vortex filaments, and the advection of particles. From the evolving filaments, we obtain the current velocity vector field at each time step. We can evaluate this velocity field on the whole of $\mathbb{R}^3$ using the Biot-Savart formula (2.6). We will use this velocity vector field for particle advection, using a standard ODE integration scheme.

In our implementation this second part is implemented using a GLSL shader to perform particle advection on a GPU. The filament simulation is implemented in a Java library. The most expensive part of the filament simulation is the advection step which includes evaluations of the induced velocity field at all filament vertices.

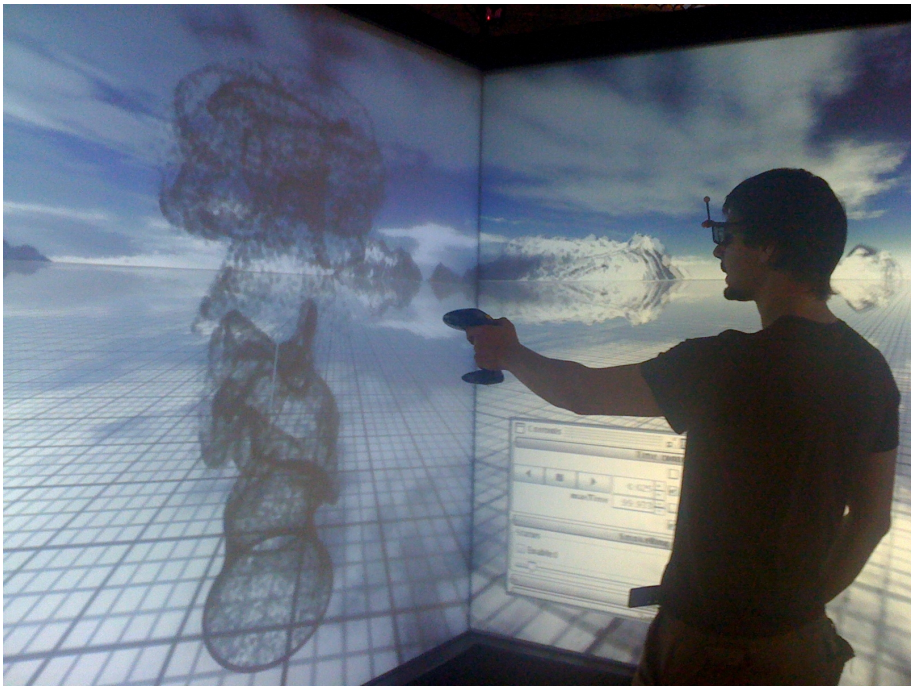


Figure 2.7: *Interactive smoke simulation in an immersive, distributed environment.*

A significant speed up can be achieved when implementing this part on the GPU, but in our experimental implementation we favor the greater flexibility of a CPU implementation.

For particle advection we transfer the whole set of edges to the GPU. This needs to be done in every time step. The particle positions are also stored on the GPU, but they are only transferred during the initialization of the simulation. This allows to advect a large number of particles compared to the number of filament edges, at interactive rates.

The implementation uses jReality [WGB+09], which provides the infrastructure for interaction and rendering. Applications written in jReality will run also in distributed, immersive virtual environments with head-tracking and tracked input devices, see Figure 2.7. Interaction with the 3D scene is usually done with a *pointer device*, which is the mouse pointer in a default desktop setup and the 6DOF tracked wand in an immersive environment.

2.5.1 Filament evolution

Each simulation step of the filament evolution consists of the two steps:

1. Evolve each filament by one step of the doubly-discrete smoke ring evolution, by computing two successive Darboux transforms.

2. Advect the vertices of the filaments according to the evolution Equation (2.4), using a standard ODE integration scheme. The right hand side is obtained by implementing Equation (2.10) and summing up over all filament edges.

2.5.1.1 Implementation of the doubly-discrete smoke ring evolution

To compute one step of the doubly-discrete smoke ring evolution for time step Δt we need to do the following:

1. Determine the length L and number of vertices n of γ.
2. Compute U according to Equation (2.14).
3. Compute $\tilde{U}$. Construct a reference n-gon with same length as γ and evaluate its own induced speed at a vertex.
4. Compute l and r according to Equations (2.15).
5. Compute the Darboux-transform η of γ with parameters l and r.
6. Compute the Darboux-transform $\hat{\gamma}$ of η with parameters l and $-r$.

Now $\tilde{\gamma}$ is the required step of the doubly-discrete smoke ring evolution of γ.

We will now explicitly implement the computation of the Darboux-transform. The following pseudocode assumes that vectors in $\mathbb{R}^3$ are implemented by a class `real3` and quaternions are implemented in a class `quat`. Quaternions are constructed by passing the real part as a `double` and the imaginary part as a `real3`. The two classes have all standard operators implemented, `quat.inverse()` gives the inverse of a quaternion. A polygon is stored as a `vector` containing all polygon vertices.

First, we implement a single Darboux step, as described in Equation (2.12):

```
real3 darboux_step(real3 S_i, real3 lT_i, double r) {
  quat rlT_S = quat(-r, lT_i - S_i);
  quat lT = quat(0, lT_i);
  quat lTnext = rlT_S*lT*rlT_S.inverse();
  return lTnext.imag();
}
```

Then we compute the end vector $l\tilde{T}_1$ for a given start vector lT_1, see Equation (2.13):

```
real3 monodromy(vector<real3> gamma, real3 lT_1, double r) {
  real3 lT = real3(lT_1);
  for (int i=0; i<n; i++) {
    real3 S_i = gamma[i+1]-gamma[i];
    lT = darboux_step(S_i, lT, r);
  }
  return lT;
}
```

Now we determine a start vector lT that leads to a *closed* Darboux transform. This vector is a fixed point that we determine by applying the power method to the previous method:

```
real3 power_method(vector<real3> gamma, double l, double r) {
  real3 lT = real3(0, 0, l);
  for (int i=0; i<MAX_ITERS; i++) {
    real3 lastLT = real3(lT);
    lT = monodromy(gamma, lT, r);
    if ((lT - lastLT).norm() < EPS) return lT;
  } // signal: "did not converge"
}
```

Finally, we construct the closed Darboux transform η using the start vector lT that gives a closed solution:

```
void darboux(vector<real3> gamma,
             vector<real3> eta, double l, double r) {
  real3 lT = power_method(gamma, l, r);
  for (int i=0; i<n; i++) {
    eta[i]=gamma[i]+lT;
    real3 S_i = gamma[i+1]-gamma[i];
    lT = darboux_step(S_i, lT, r);
  }
}
```

Note that in some rare cases the power method might fail to converge (which usually does not happen in our simulations). In this case we just skip the computation of the Darboux transform. To handle these cases also, a more sophisticated method, for instance the periodic QR decomposition [Kre06], needs to be implemented to determine the fixed point.

2.5.2 Particle advection

Particle advection is performed using the second-order Runge-Kutta scheme. This scheme requires the velocity field (and thus the filaments) only at $t = t_i + \Delta t/2$, other schemes would require more filament positions during one time step, resulting in higher traffic between CPU and GPU memory. The advection step of a particle is implemented as a GLSL fragment shader. The fragment shader iterates all vortex filament edges, transforms them such that the evaluation point is at the origin, evaluates the velocity field generated by the current edge, and sums up.

2.5.3 Rendering

Our focus is the real-time interactive simulation algorithm, rather than rendering. For this reason we have chosen the cheapest and fastest way to render the marker particles: As plain, unshaded, highly transparent points.

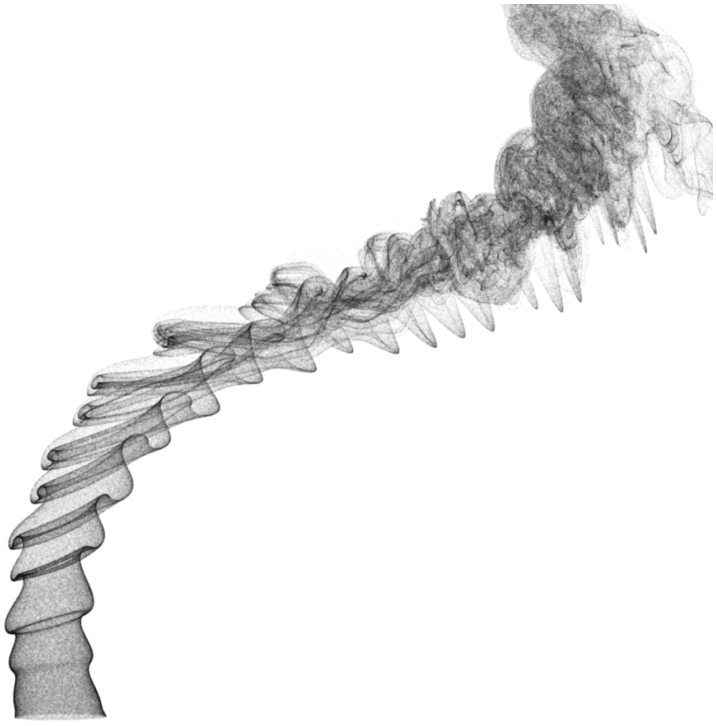


Figure 2.8: *Control of large scale flow features. The image shows the evolution of a jet, guided along a spiral curve using a discretized vortex tube.*

2.5.4 Interaction

Our implementation contains several modeling tools for interactive design of fluid motion. The tools to design an initial setup:

Filament editor: Closed polygonal vortex filaments can be modeled as an initial vorticity configuration. Each closed polygon has an associated strength Γ and a thickness parameter a. Figure 2.1 has been made using three initial filaments.

Vortex tube: Static vortex filaments can be placed along the cross-sections of a tube, which is modeled by its center curve and a given radius. This is used to model a background flow that guides the overall motion of the fluid, see Figures 2.4, 2.8.

Jet emitter: A jet is simulated by frequent emission of circular filaments. This is possible since the originally created vortex sheet rolls up to vortex core structures (Section 2.3). To break symmetry, a random distortion is applied to each filament. This was used in the Figures 2.8 and 2.9, and also below the sphere in Figure 2.3.

For interaction during the simulation one can create new smoke rings using the mouse or a wand in an immersive environment.

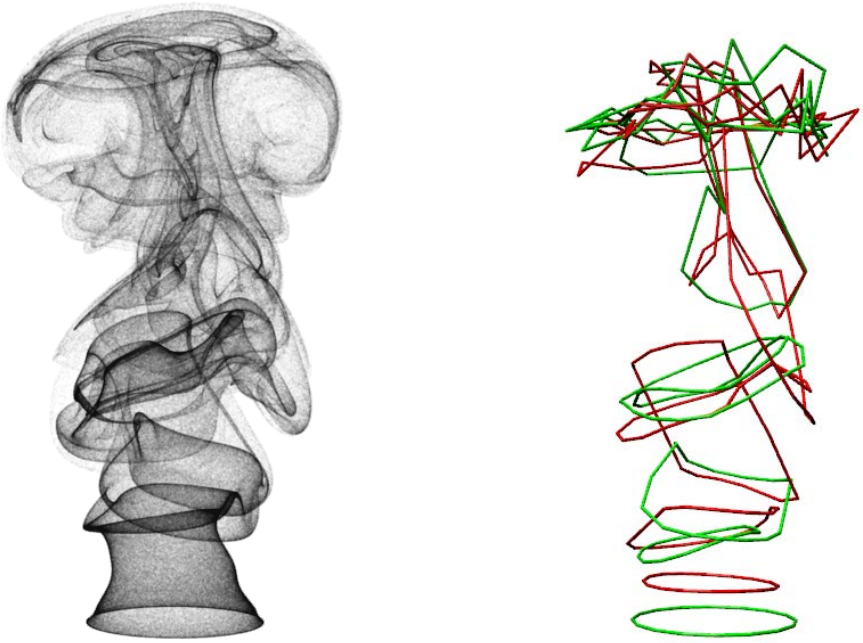


Figure 2.9: *Simulation of a jet. The left picture shows* 1024^2 *particles, injected into the flow from a circular curve at the bottom, where also the filaments are inserted. The right hand side shows the vortex filaments at the same time of the simulation.*

2.6 Results and discussion

The system we have described allows the interactive simulation of 3D smoke on desktops and also in immersive virtual environments. The system is able to simulate realistic and physically accurate fluid motions using a small number of filaments, as shown in the figures. Using the GPU it is possible to advect a large number of marker particles in real time. Table 2.1 shows framerates measured on a Intel Core 2 Extreme CPU X9650 3.00GHz with 4 GB RAM and a nVidia GeForce 8800 Ultra GPU. Note that the frame rate is limited to 200 fps by the update rate of the Java AWT event queue.

Because of the physically accurate Biot-Savart law, our approach is independent of scale; scaling the fluid domain will result in a correctly scaled velocity field. In contrast, the method of Angelidis and Neyret requires adjustments of parameters that are not physically meaningful. By the use of the doubly-discrete smoke ring flow we are able to model even thin filaments with a small number of edges.

The resulting simulations are accurate for a limited amount of time, even when the filaments incur long edges and sharp cusps, see for instance the filaments shown in Figure 2.9. From the discussion about energy and the geometry of the smooth

	# Edges				
# Particles	8	64	128	256	512
4096	>200	>200	180	115	9.4
16384	>200	>200	148	94	9.0
65536	195	126	90	56	8.1
262144	63	42	33	21.8	5.5
1048576	16.8	11.85	9.7	6.2	2.6

Table 2.1: *Performance in fps including simulation and rendering.*

filaments in Section 2.3.3 we conclude that polygons need to be subdivided in order to capture the increasing complexity of the smooth filaments. This will however quickly lead to polygons with excessively many edges and to edge lengths that cause problems to the numerics.

Previously described methods to increase stability for long-time simulations can also be added to our method, although they are not satisfying: An unnatural amount of damping [AN05], and filtering of high frequencies of the filaments [ANSN06]. We believe that in order to implement long-time simulations one has to cope with the increasing complexity of the filaments by handling topology changes due to reconnections.

2.7 Conclusion and outlook

We have presented a method to simulate smoke that uses the physically correct velocity field. Local effects are captured by the doubly-discrete smoke ring flow. Thus the simulation is physically accurate even for a coarse discretization of the filaments. We have implemented the method for desktop and immersive virtual environments. Using GPU techniques, our solution runs in real time.

Current research aims to introduce arbitrary polygonal meshes as boundary conditions, and to handle filament reconnections.

3 Filament-based smoke with vortex shedding and variational reconnection

Steffen Weißmann, Ulrich Pinkall

ACM Trans. Graph. 29(4), Proc. ACM/SIGGRAPH Conf. 2010.

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.

3.1 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 [AN05, ANSN06]. In [PSW07, WP09] 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 Figure 3.1). Our approach gives special

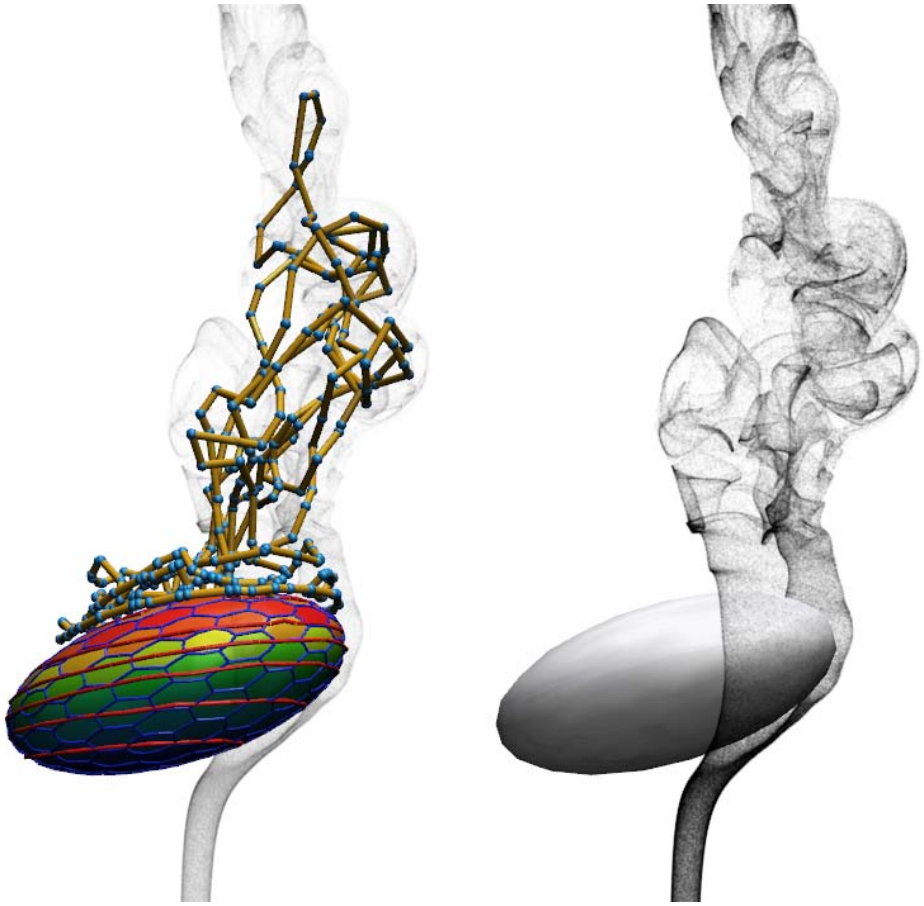


Figure 3.1: *Flow around an ellipsoid with vortex shedding.*

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.

3.1.1 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 Section 3.3 we give a formulation of filament dynamics based on a Hamiltonian system. Compared to [WP09] 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* [Cho90, Cho93], 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 Section 3.5 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 Section 3.4 we show how to include static or moving obstacles into the simulation, by computing *image vorticity* [Saf92] as a vortex sheet on the obstacle. Our method to compute the boundary layer is related to the standard 3D panel method [KP01], 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 Appendix 3.A) is new and also of interest for the numerics of boundary integral equations [SS04].

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 [Lim89]. Head-on collision of two vortex rings can result in a reconnection that leads to many small vortex rings [LN92]. For validation we demonstrate how our method reproduces these real experiments (Figures 3.7, 3.8). We also simulate vortex shedding behind a sphere and demonstrate the long-time robustness of our method using a jet simulation.

3.2 Related work

Much work on 3D smoke simulations is based on Stam's *Stable Fluids* [Sta99], together with *vorticity confinement* [FSJ01]. We will not review the entire literature here and

instead refer to the course notes [MJT08] 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 [SF93, FSJ01, Ney03, SRF05, KLLR05, ETK+07, NSCL08, KTJG08] or even added in a user controlled way [PK05, PTSG09]. Simplicial fluids [ETK+07] preserve circulation (and thus vorticity) by construction. Still there is dissipation of energy which can be avoided using variational integrators for fluids [MCP+09]. 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 [SF95, DG96, MCG03, BT07, YWH+09] or vortex particles. Inviscid flow like smoke is challenging for SPH. Vortex particles [PK05] 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 [SRF05, CCB+08, PTSG09].

The use of vortex filaments as basic primitives for modeling 3D smoke was pioneered by Angelidis and Neyret [AN05]. A more physically based method was introduced in [WP09] 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 [ANSN06] by filtering high frequencies from the filament geometries. A physically motivated approach was introduced by Chorin [Cho90, Cho93]: Hairpin removal and filament reconnection. Chorin's method was also used in [Ber06, MG07, Ber09].

Real-time methods for 3D smoke simulation are mostly based on Semi-Lagrangian methods using GPU techniques [CLT07, Kim08, Yan09a]. Reductions to 2D [RNGF03, KW05] have also been used to improve performance. Model reduction [TLP06] 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: [PK05] compute the boundary layer using the standard 3D panel method [KP01]. [PTSG09] 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.

3.3 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.

3.3.1 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:

$$v_\omega(x) = \frac{1}{4\pi} \int_{\mathbb{R}^3} \omega(z) \times \frac{x - z}{|x - z|^3} \, dz. \tag{3.1}$$

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 γ_i (the *vortex filaments*) in a delta-function like manner: Then the generated velocity field (3.1) 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

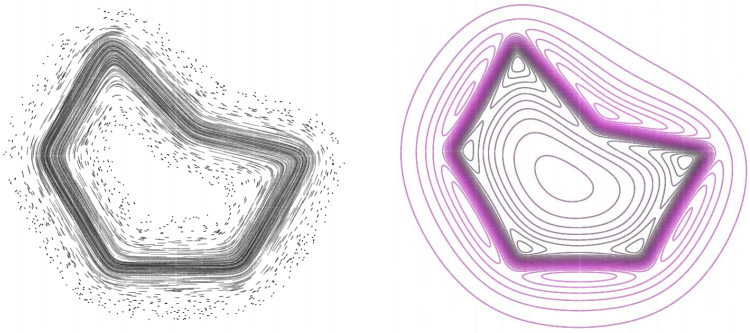


Figure 3.2: *Vorticity and velocity field of a planar polygonal vortex filament γ, smoothed by $S_{0.1}$. Left: Trajectories of $\omega = S_{0.1}\omega_\gamma$. Right: Level lines of $|v|$ of the generated velocity field $v = S_{0.1}v_\gamma$.*

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. Figure 3.2 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 [Sat99, 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

$$(S_a\, v_\gamma)(x) = \sum_i \frac{\Gamma_i}{4\pi} \oint \gamma_i'(\tilde{s}) \times \frac{x - \gamma_i(\tilde{s})}{\sqrt{a^2 + |x - \gamma_i(\tilde{s})|^2}^3}\, d\tilde{s}, \tag{3.2}$$

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

$$\dot{\gamma}_j(s) = \sum_i \frac{\Gamma_i}{4\pi} \oint \gamma_i'(\tilde{s}) \times \frac{\gamma_j(s) - \gamma_i(\tilde{s})}{\sqrt{a^2 + |\gamma_j(s) - \gamma_i(\tilde{s})|^2}^3}\, d\tilde{s} \tag{3.3}$$

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 (3.3) 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 [MW83, PSW07]: 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 [Saf92].

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 (Section 3.4):

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

Here $\mathrm{flux}_\eta(u)$ denotes the normal flux of the vector field u through a disc with boundary η. In Appendix 3.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

$$\begin{aligned} S_a * S_b &\approx S_{\sqrt{a^2+b^2}}, \\ \sqrt{S_a} &\approx S_{\frac{a}{\sqrt{2}}}, \\ \sqrt{S_a} * \sqrt{S_b} &\approx S_{\sqrt{\frac{a^2+b^2}{2}}}. \end{aligned} \tag{3.4}$$

The proposed solution to the mentioned problems is therefore:

- Point particles are advected with the velocity field (3.2) with a^2 replaced by $a_i^2/2$.
- Filaments evolve according to the evolution equation (3.3) 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 $\sqrt{S_{a_i}} * \sqrt{S_{a_j}}$.

3.3.2 Numerical formulation

The basic numerical method for filament-based fluid simulation that we are using was given in [WP09]: 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 (3.3) for filaments:

$$\dot{x}_{jk} = (S_{a_{ij}} v_\gamma)(x_{jk}). \tag{3.5}$$

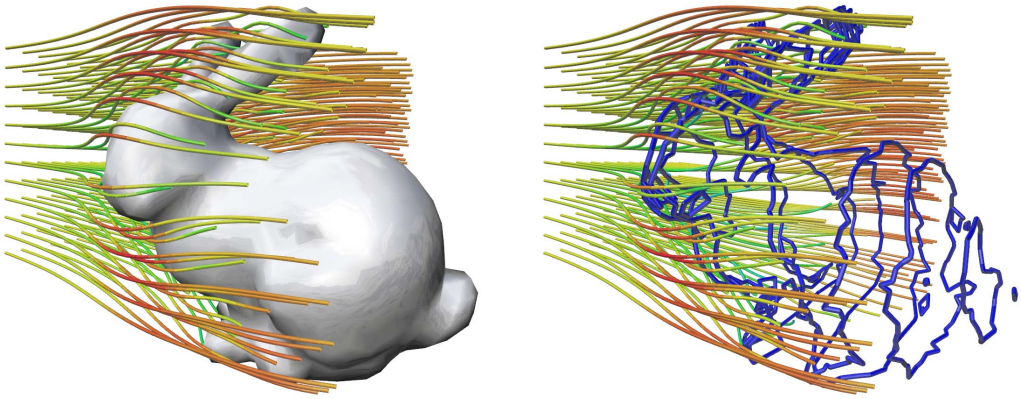


Figure 3.3: *Vortex filaments that force a constant background flow to be tangential to the bunny. Inside the velocity is almost zero. The mesh has* 678 *vertices and* 1352 *facets.*

Here $S_{a_{ij}} v_\gamma$ denotes the velocity field (3.2) 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 [WP09]. The evolution equation for the filament vertices (3.5) 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 [PSW07], the concrete implementation of the method is given in [WP09]. Iteration of these two steps finally gives the evolution of the discrete vortex filaments.

3.4 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 Figure 3.3. 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.

3.4.1 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

$$v_M(p) = \frac{1}{4\pi} \int_M (N \times \operatorname{grad} f)(q) \times \frac{p-q}{|p-q|^3}\, dq. \tag{3.6}$$

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 Equation (3.6) cancels the normal component of v:

$$\langle v(q), N(q)\rangle + \langle v_M(q), N(q)\rangle = 0. \tag{3.7}$$

The vortex sheet on M obtained in this way is called the *image vorticity* [Saf92].

Comparison of the sheet velocity field (3.6) with the Biot-Savart law (3.1) 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 Equation (3.6) 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.

3.4.2 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* [DKT08].

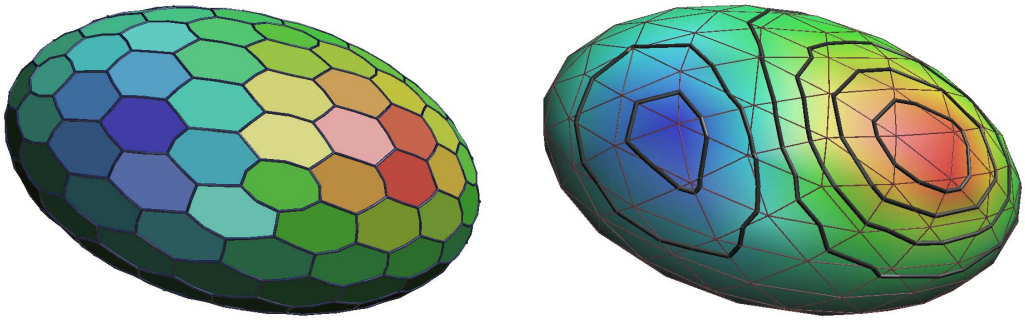


Figure 3.4: *Filament approximation of vortex sheets: The piecewise constant f on the dual mesh gives vorticity on the edge graph (left). f linearly interpolated on the primal triangle mesh yields vorticity along level lines (right).*

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. Equation (3.7) turns into a linear system, stating that the total normal flux through each ϕ_i is zero:

$$\sum_j f_j \underbrace{\operatorname{flux}_{\eta_i}(S_a v_{\eta_j})}_{a_{ij}} = -\underbrace{\operatorname{flux}_{\eta_i}(v)}_{g_i}. \tag{3.8}$$

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 (Appendix 3.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 $\{1, \dots, 1\}$ 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 $\{1, \dots, 1\}$. Fortunately, for the same reason also the right hand of Equation (3.8) 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 Equation (3.8).

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 (Appendix 3.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 Figure 3.4.

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.

3.4.3 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 [GA88, RVJ00]. 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 $\tilde{f}$.

3.5 Variational reconnection

As has been observed by Chorin [Cho90, Cho93], 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 [Cho90, Cho93] 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 $\eta - \gamma$:

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

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

$$F(\eta) = \lambda\, \Gamma\, L(\eta) + d(\eta, \gamma). \tag{3.9}$$

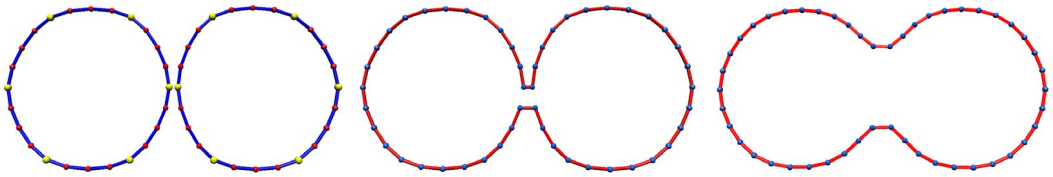
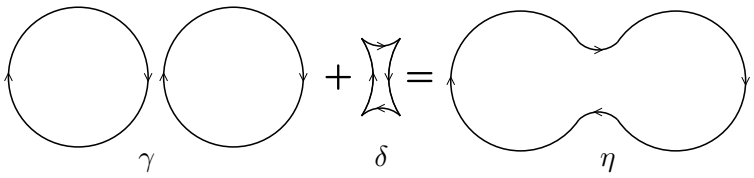


Figure 3.5: *Variational reconnection. Two nearby filaments are reconnected to one single filament. Variation of the reconnected filament according to* grad F *significantly improves the geometry.*

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 (3.9). $\lambda > 0$ 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 (3.9) 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 Figure 3.5. 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

$$L(\delta_t) + L(\delta_b) - L(\delta_l) - L(\delta_r) + \sqrt{\mathrm{flux}_\delta(S_a v_\delta)}/\lambda < 0. \tag{3.10}$$

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 Equation (3.10) 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* [Cho93].

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 Figure 3.6.

3.6 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 (3.2). 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 Section 3.3.2 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 [WP09].

(b) Advect filaments according to the evolution equation (3.11) 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 (3.10).

(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.

Figure 3.6: *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.*

3.6.1 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 Section 3.3.2. 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 $\gamma \cup \gamma_M$:

$$\dot{x}_i = v_B(x_i) + (S_{a_{ij}} v_{\gamma \cup \gamma_M})(x_i). \tag{3.11}$$

Evaluating the right-hand side of Equation (3.11) 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 Equation (3.5).

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 Appendix 3.A). Rigid motion flux is computed as described in Appendix 3.B.

To determine the closed level sets we represent the triangle mesh M using a half-edge data structure [Wei85].

To advect the filament vertices we solve the ODE (3.11) 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 Section 3.7 were computed using a *Dormand-Prince 5/4* solver [DP80].

3.6.2 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 [WGB+09] 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 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 (Equation (3.11)) and the flux evaluations through the obstacle facets (the RHS of Equation (3.8)) are computed simultaneously for different vertices/facets.

3.7 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 [LN92, Lim89] 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 (Figure 3.7) was computed this way. For the head-on collision (Figure 3.8) though we had to use a 40 times smaller time-step because we had to trace particles in a very

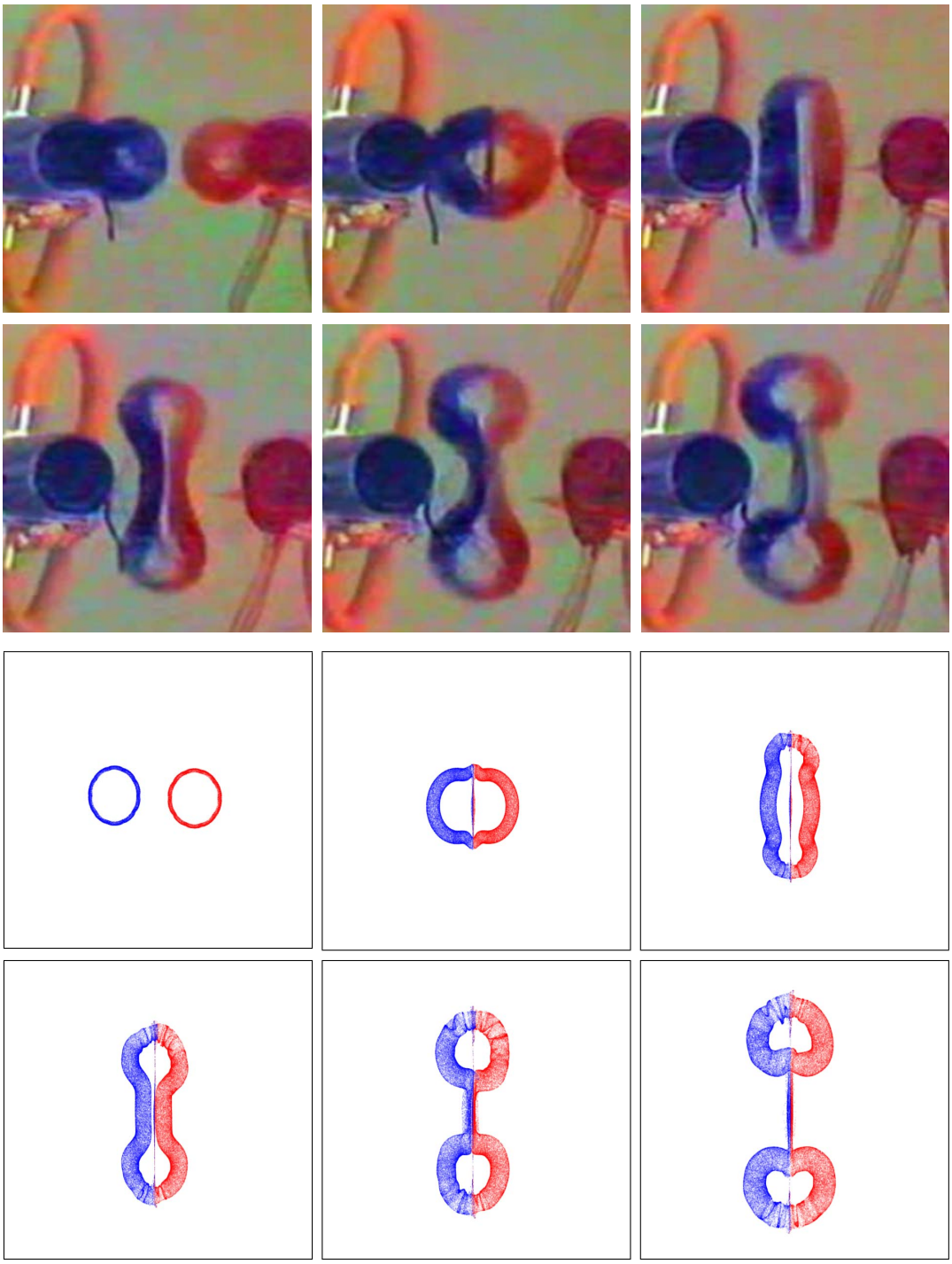


Figure 3.7: *Oblique collision of two vortex rings. Comparison with real experiments by T. T. Lim, with permission.*

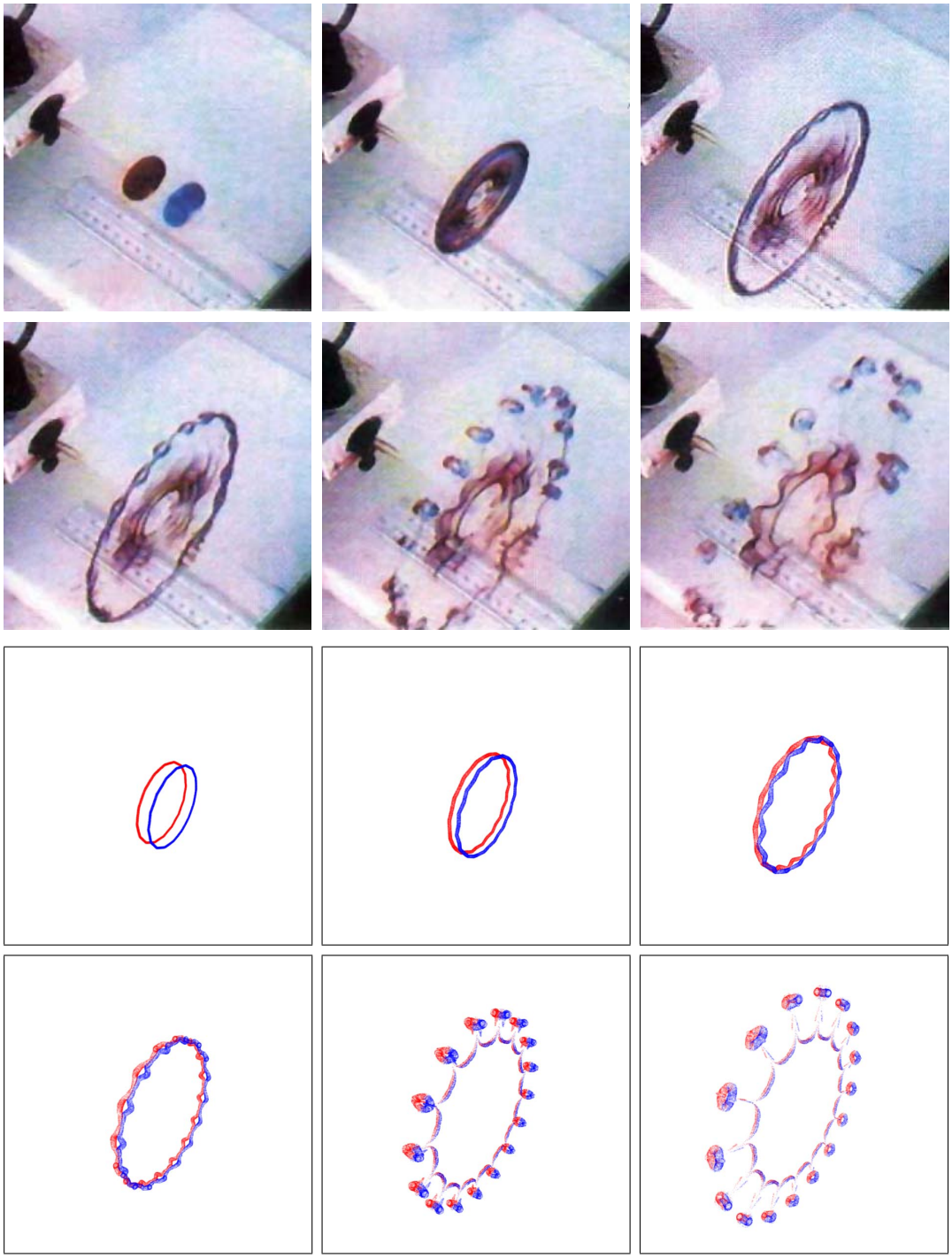


Figure 3.8: *Head-on collision of two vortex rings. Comparison with real experiments by T. T. Lim, with permission.*

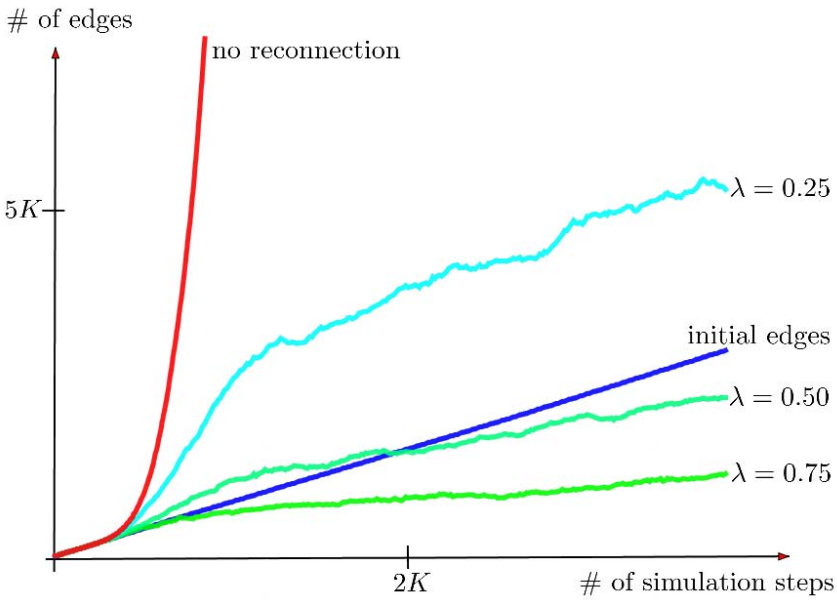


Figure 3.9: *Number of edges used to simulate a jet (Figure 3.6). Circular filaments are added at a fixed rate at the orifice, thus the total number of edges added to the simulation grows linearly (blue). Without reconnection, the number of edges grows exponentially fast due to subdivision (red). With variational reconnection, the number of edges grows slowly (depending on λ) and long time simulations are possible.*

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. Figure 3.6 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 Figure 3.9). Without reconnection, the number of polygon edges increases exponentially 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 Figure 3.1. The wake behind a sphere from our simulation is compared to a photograph of an experiment by Henri Werlé [Dyk82] in Figure 3.10. Different wake patterns (Figure 3.11) 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 (Section 3.6.2): 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.

3.8 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 [BGH03]. 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* [Saf92]. Furthermore, we want to find ways to handle free surfaces using filaments. This would enable us to simulate water with an interface to air.

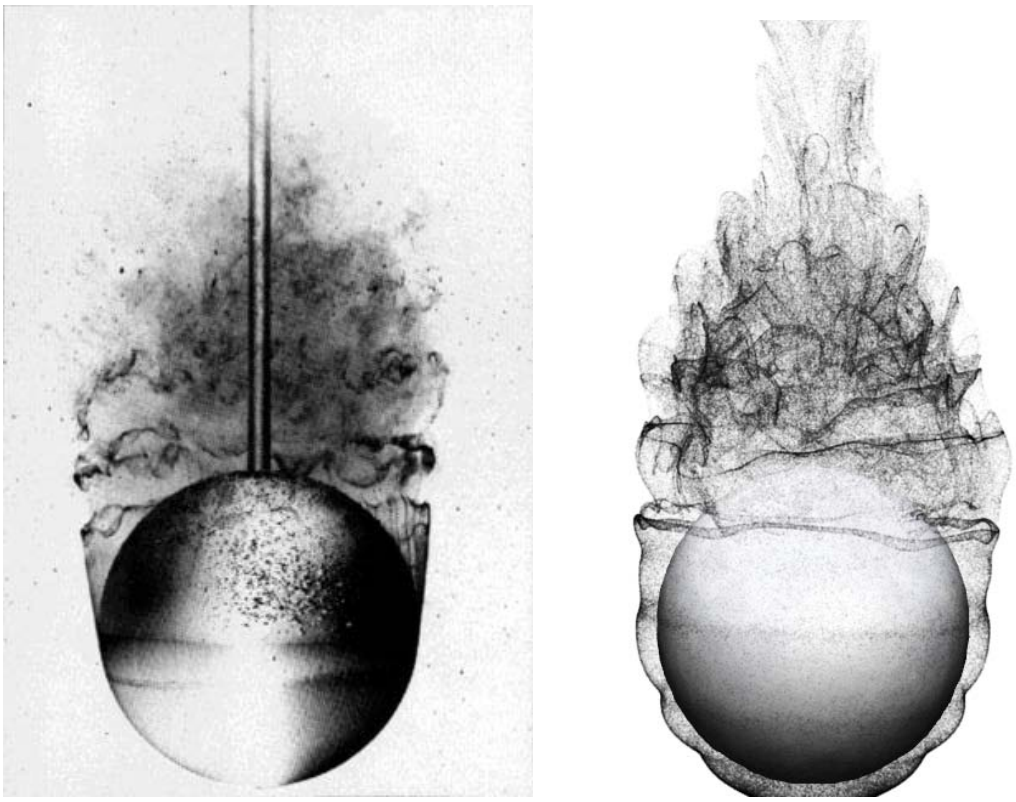


Figure 3.10: *Wake behind a sphere. Comparison of a photo by Henri Werlé (colors inverted) with our simulation. Reproduced with permission of ONERA, The French Aerospace Lab.*

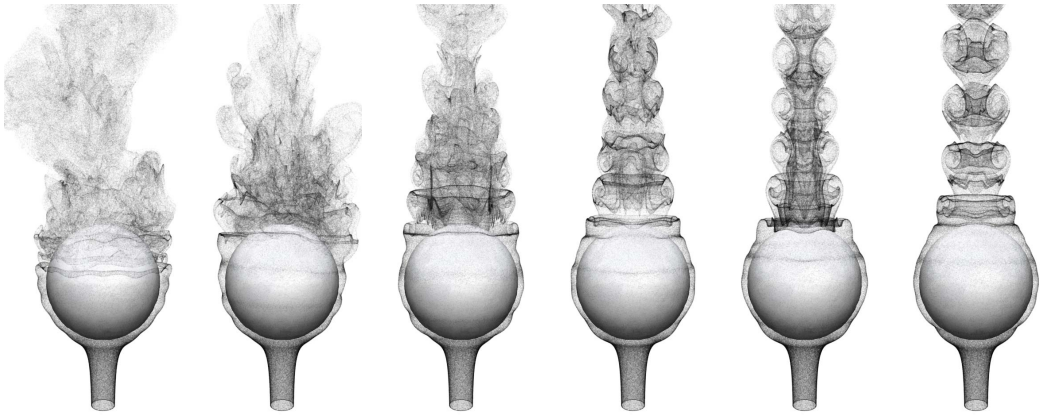


Figure 3.11: *Different wake patterns behind a sphere. The simulations differ only in the rate at which image vorticity filaments are released into the flow. Note the transition to turbulence from right to left.*

3.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. For nearby filaments the velocity field becomes almost singular, here the explicit formula 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 (Equation (3.8)) and in the reconnection criterion (3.10).

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

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

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,

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

and we have to compute

$$f = \int_0^l \int_0^L \frac{ds\,dt}{\sqrt{a^2 + |p + tT - sS|^2}} \tag{3.12}$$

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

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

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 (Equation (3.12)) 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 $\langle p, T\rangle + (t \mp s)$.

3.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

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

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 [Ang04]). 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

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

Directions for future research

Regarding the performance and scalability of the method, further research is required to reduce the quadratic complexity of the filament simulation. The velocity field needs to be evaluated more efficiently, e.g., using a tree code as in [LK01] adapted to filaments. The same holds for obstacles: Many moving obstacles are not possible at interactive rates. Here an hierarchical scheme (e.g., based on hierarchical matrices [BGH03]) is required, based on low-rank approximations for interaction between far away obstacle parts.

To further improve the stability of our method we want to develop a variational integrator that preserves kinetic energy and momentum of the fluid. Concerning extensions of the physical model, we want to include buoyancy effects, deformable obstacles, free surfaces and the two-way coupling between fluid and obstacles.

We also want to derive a relation between the Reynolds number and the parameters that occur in our method to simulate vortex shedding (frequency, offset, circulation). This would allow to determine these parameters directly from the chosen obstacle size, background velocity and fluid material (e.g., water or smoke).

Bibliography

[ABS03] V. E. Adler, A. I. Bobenko, and Yu.B. Suris. Classification of Integrable Equations on Quad-Graphs. The Consistency Approach. *Comm. Math. Phys.*, 233(3):513–543, 2003.

[AH88] J. C. Agui and L. Hesselink. Flow visualization and numerical analysis of a coflowing jet: a three-dimensional approach. *J. Fluid Mech.*, 191:19–45, 1988.

[AK98] V. A. Arnold and B. A. Khesin. *Topological Methods in Hydrodynamics*. Springer, 1998.

[AN05] A. Angelidis and F. Neyret. Simulation of Smoke based on Vortex Filament Primitives. In *Proc. Symp. Comp. Anim.*, pages 87–96, 2005.

[Ang04] A. Angelidis. Hexanions: 6d Space for Twists. Technical report, University of Otago, November 2004.

[ANSN06] A. Angelidis, F. Neyret, K. Singh, and D. Nowrouzezahrai. A Controllable, Fast and Stable Basis for Vortex Based Smoke Simulation. In *Proc. Symp. Comp. Anim.*, pages 25–32, 2006.

[Ber06] P. S. Bernard. Turbulent flow properties of large-scale vortex systems. *Proc. Nat. Acad. Sci. USA*, 103(27):10174–10179, 2006.

[Ber09] P. S. Bernard. Vortex filament simulation of the turbulent coflowing jet. *Phys. Fluids*, 21(2):025107, 2009.

[BGH03] S. Börm, L. Grasedyck, and W. Hackbusch. *Hierarchical Matrices*. Lecture Notes. MPI MIS, 2003.

[BS02] A. I. Bobenko and Yu. B. Suris. Integrable noncommutative equations on quad-graphs. The consistency approach. *Lett. Math. Phys.*, 61(3):241–254, 2002.

[BT07] M. Becker and M. Teschner. Weakly compressible SPH for free surface flows. In *Proc. Symp. Comp. Anim.*, pages 209–217, 2007.

[CCB+08] P. Chatelain, A. Curioni, M. Bergdorf, D. Rossinelli, W. Andreoni, and P. Koumoutsakos. Billion vortex particle direct numerical simulations of aircraft wakes. *Comput. Methods Appl. Mech. Engrg.*, 197:1296–1304, 2008.

[CGS86] J. Cieśliński, P. K. H. Gragert, and A. Sym. Exact Solution to Localized-Induction-Approximation Equation Modeling Smoke Ring Motion. *Phys. Rev. Lett.*, 57(13):1507–1510, 1986.

[Cho90] A. J. Chorin. Hairpin removal in Vortex interactions. *J. Comput. Phys.*, 91(1):1–21, 1990.

[Cho91] A. J. Chorin. *Vorticity and Turbulence*. Springer, 1991.

[Cho93] A. J. Chorin. Hairpin removal in vortex interactions II. *J. Comput. Phys.*, 107(1):1–9, 1993.

[CI05] A. Calini and T. Ivey. Finite-Gap Solutions of the Vortex Filament Equation: Genus One Solutions and Symmetric Solutions. *J. Nonlinear Sci.*, 15(5):321–361, 2005.

[CKL03] P. Chatelain, D. Kivotides, and A. Leonard. Reconnection of Colliding Vortex Rings. *Phys. Rev. Lett.*, 90(5):054501, 2003.

[CLT07] K. Crane, I. Llamas, and S. Tariq. *GPU Gems 3 - Real-Time Simulation and Rendering of 3D Fluids*, chapter 30, pages 633–673. Addison-Wesley, 2007.

[DG96] M. Desbrun and M.-P. Gascuel. Smoothed particles: a new paradigm for animating highly deformable bodies. In *Symp. on Comp. Anim. and Sim.*, pages 61–76. Springer, 1996.

[DKT08] M. Desbrun, E. Kanso, and Y. Tong. Discrete Differential Forms for Computational Modeling. In *Discrete Differential Geometry*, volume 38 of *Oberwolfach Seminars*, pages 287–324. Birkhäuser, 2008.

[DP80] J. R. Dormand and P. J. Prince. A family of embedded Runge-Kutta formulae. *J. Comput. Appl. Math.*, 6(1):19–26, 1980.

[DS95] A. Doliwa and P. M. Santini. Integrable dynamics of a discrete curve and the Ablowitz-Ladik hierarchy. *J. Math. Phys.*, 36(3):1259–1273, 1995.

[Dyk82] M. van Dyke. *An Album of Fluid Motion*. The parabolic Press, 1982.

[EM70] D. Ebin and J. Marsden. Groups of diffeomorphisms and the motion of an incompressible fluid. *Ann. of Math.*, 92:102–163, 1970.

[ETK+07] S. Elcott, Y. Tong, E. Kanso, P. Schröder, and M. Desbrun. Stable, circulation-preserving, simplicial fluids. *ACM Trans. Graph.*, 26(1), 2007.

[EVG07] L. A. Ek, R. Vistnes, and O. E. Gundersen. Animating physically based explosions in real-time. In *Proc. ACM/AFRIGRAPH Conf.*, pages 61–69, 2007.

[FSJ01] R. Fedkiw, J. Stam, and H. W. Jensen. Visual simulation of smoke. In *Proc. ACM/SIGGRAPH Conf.*, pages 15–22, 2001.

[GA88] C. Greengard and C. R. Anderson, editors. *Vortex Methods*, volume 1360 of *Lecture Notes in Mathematics*. Springer, 1988.

[Has72] H. Hasimoto. A soliton on a vortex filament. *J. Fluid Mech.*, 51:477–485, 1972.

[Hof00] T. Hoffmann. *Discrete curves and surfaces*. PhD thesis, Technische Universität Berlin, 2000.

[Hof05] T. Hoffmann, 2005. Personal communication.

[Hof08] T. Hoffmann. Discrete Hashimoto Surfaces and a Doubly Discrete Smoke-Ring Flow. In *Discrete Differential Geometry*, volume 38 of *Oberwolfach Seminars*, pages 95–116. Birkhäuser, 2008.

[Ive06] T. A. Ivey. Geometry and Topology of Finite-Gap Vortex Filaments. In *Geometry, Integrability, and Quantization 2005*, pages 187–202, 2006.

[Jef] G. Jeffery. Coloured Smoke. online. `http://sensitivelight.com/smoke2`, image 14.

[Kim08] T. Kim. Hardware-aware analysis and optimization of stable fluids. In *Proc. I3D Symp.*, pages 99–106, 2008.

[KL93] J. Koplik and H. Levine. Vortex reconnection in superfluid helium. *Phys. Rev. Lett.*, 71(9):1375–1378, 1993.

[KL03] D. Kivotides and A. Leonard. Computational model of vortex reconnection. *Europhys. Lett.*, 63:354–360, 2003.

[KLLR05] B. Kim, Y. Liu, I. Llamas, and J. Rossignac. FlowFixer: Using BFECC for Fluid Simulation. In *Proc. EG Workshop on Natural Phenomena*, 2005.

[KP01] J. Katz and A. Plotkin. *Low-Speed Aerodynamics*. Cambridge University Press, second edition, 2001.

[Kre06] D. Kressner. The periodic QR algorithm is a disguised QR algorithm. *Linear Algebra and Appl.*, 417(2-3):423–433, 2006.

[KTJG08] T. Kim, N. Thürey, D. James, and M. Gross. Wavelet turbulence for fluid simulation. In *Proc. ACM/SIGGRAPH Conf.*, 2008.

[KW05] J. Krüger and R. Westermann. GPU Simulation and Rendering of Volumetric Effects for Computer Games and Virtual Environments. *Comp. Graph. Forum*, 24(3), 2005.

[LB80] D. Levi and R. Benguria. Backlund Transformations and Nonlinear Differential Difference Equations. *Proc. Nat. Acad. Sci. U.S.A.*, 77(9):5025–5027, 1980.

[Lev81] D. Levi. Nonlinear differential difference equations as Backlund transformations. *J. Phys. A*, 14(5):1083–1098, 1981.

[Lim89] T. T. Lim. An erimental study of a vortex ring interacting with an inclined wall. *Exp. Fluids*, 7(7):453–463, 1989.

[LK01] K. Lindsay and R. Krasny. A particle method and adaptive treecode for vortex sheet motion in three-dimensional flow. *J. Comput. Phys.*, 172(2):879–907, 2001.

[LLW04] Y. Liu, X. Liu, and E. Wu. Real-time 3D fluid simulation on GPU with complex obstacles. In *Proc. Comp. Graph. and App. Conf.*, pages 247–256, Oct. 2004.

[LN92] T. T. Lim and T. B. Nickels. Instability and reconnection in the head-on collision of two vortex rings. *Nature*, 357:225–227, 1992.

[LP90] J. Langer and R. Perline. The Hasimoto transformation and integrable flows on curves. *Appl. Math. Lett.*, 3:61–64, 1990.

[MCG03] M. Müller, D. Charypar, and M. Gross. Particle-based fluid simulation for interactive applications. In *Proc. Symp. Comp. Anim.*, pages 154–159, 2003.

[MCP+09] P. Mullen, K. Crane, D. Pavlov, Y. Tong, and M. Desbrun. Energy-preserving Integrators for Fluid Animation. *Proc. ACM/SIGGRAPH Conf.*, 28(3), 2009.

[MG07] Y. M. Marzouk and A. F. Ghoniem. Vorticity structure and evolution in a transverse jet. *J. Fluid Mech.*, 575:267–305, 2007.

[MJT08] J. Müller, M.and Stam, D. James, and N. Thürey. Real time physics: class notes. In *ACM/SIGGRAPH classes*, 2008.

[MM91] J. E. Martin and E. Meiburg. Numerical investigation of three-dimensionally evolving jets subject to axisymmetric and azimuthal perturbations. *J. Fluid Mech.*, 230:271–318, 1991.

[MM92] J. E. Martin and E. Meiburg. Numerical investigation of three-dimensionally evolving jets under helical perturbations. *J. Fluid Mech.*, 243:457–487, 1992.

[MW83] J. Marsden and A. Weinstein. Coadjoint orbits, vortices, and Clebsch variables for incompressible fluids. *Physica*, 7D:305–323, 1983.

[Ney03] F. Neyret. Advected Textures. In *Proc. Symp. Comp. Anim.*, 2003.

[NQC83] F. W. Nijhoff, G. R. W. Quispel, and H. W. Capel. Direct linearization of nonlinear difference-difference equations. *Phys. Lett. A*, 97(4):125–128, 1983.

[NSCL08] R. Narain, J. Sewall, M. Carlson, and M. C. Lin. Fast animation of turbulence using energy transport and procedural synthesis. *ACM Trans. Graph.*, 27(5):1–8, 2008.

[PK05] S. I. Park and M. J. Kim. Vortex fluid for gaseous phenomena. In *Proc. Symp. Comp. Anim.*, pages 261–270, 2005.

[PSW07] U. Pinkall, B. Springborn, and S. Weißmann. 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, 2007.

[PTSG09] T. Pfaff, N. Thuerey, A. Selle, and M. Gross. Synthetic turbulence using artificial boundary layers. In *Proc. ACM/SIGGRAPH Asia Conf.*, 2009.

[Ric91] R. L. Ricca. Rediscovery of Da Rios Equations. *Nature*, 352:561–562, 1991.

[RNGF03] N. Rasmussen, D. Q. Nguyen, W. Geiger, and R. Fedkiw. Smoke simulation for large scale phenomena. *ACM Trans. Graph.*, 22(3):703–707, 2003.

[RVJ00] S. H. L. Rockliff, P. R. Voke, and N. Jacqueline. Three-Dimensional Vortices of a Spatially Developing Plane Jet. *International Journal of Fluid Dynamics*, 4, 2000.

[Saf92] P. G. Saffman. *Vortex Dynamics*. Cambridge University Press, 1992.

[Sat99] K. Sato. *Lévy Processes and Infinitely Divisible Distributions*. Cambridge University Press, 1999.

[SDR06] L. Sante Da Rios. Sul moto d'un liquido indefinito con un filetto vorticoso di forma qualunque. *Rend. Circ. Mat. Palermo*, 22:117–135, 1906.

[SF93] J. Stam and E. Fiume. Turbulent wind fields for gaseous phenomena. In *Proc. ACM/SIGGRAPH Conf.*, pages 369–376, 1993.

[SF95] J. Stam and E. Fiume. Depicting fire and other gaseous phenomena using diffusion processes. In *Proc. ACM/SIGGRAPH Conf.*, pages 129–136, 1995.

[SRF05] A. Selle, N. Rasmussen, and R. Fedkiw. A vortex particle method for smoke, water and explosions. *ACM Trans. Graph.*, 24(3):910–914, 2005.

[SS04] S. Sauter and C. Schwab. *Randelementmethoden*. Vieweg+Teubner, 2004.

[Sta99] J. Stam. Stable fluids. In *Proc. ACM/SIGGRAPH Conf.*, pages 121–128, 1999.

[TLP06] A. Treuille, A. Lewis, and Z. Popović. Model reduction for real-time fluids. *ACM Trans. Graph.*, 25(3):826–834, 2006.

[Wei85] K. Weiler. Edge-Based Data Structures for Solid Modeling in Curved-Surface Environments. *IEEE Comp. Graph. Appl.*, 5(1):21–40, 1985.

[Wei06] S. Weißmann. Real-time simulation of fluid flow. Master's thesis, TU Berlin, 2006.

[WGB+09] S. Weißmann, C. Gunn, P. Brinkmann, T. Hoffmann, and U. Pinkall. jReality. In *Proc. ACM/MM Conf.*, pages 927–928, 2009.

[WP09] S. Weißmann and U. Pinkall. Real-time interactive simulation of smoke using discrete integrable vortex filaments. In *Proc. Vir. Real., Inter. & Phys. Sim.*, pages 1–10, 2009.

[Yan09a] Q. Yang. Real-time simulation of 3D smoke based on Navier-Stokes equation. *W. Trans. Comp.*, 8(1):103–112, 2009.

[Yan09b] Q. Yang. Real-time simulation of 3D smoke on GPU. In *Proc. WSEAS/-CISST Conf.*, pages 130–134. WSEAS, 2009.

[YWH+09] H. Yan, Z. Wang, J. He, X. Chen, C. Wang, and Q. Peng. Real-time fluid simulation with adaptive SPH. *Comp. Anim. and Virtual Worlds*, 20(2–3):417–426, 2009.

[ZSP07] Y. Zhang, B. Solenthaler, and R. Pajarola. GPU accelerated SPH particle simulation and rendering. In *ACM/SIGGRAPH posters*, 2007.