Virtual Rheoscopic Fluids

Florian Hecht, Peter J. Mucha, and Greg Turk, *Member*, *IEEE*

Abstract—We present a visualization technique for simulated fluid dynamics data that visualizes the gradient of the velocity field in an intuitive way. Our work is inspired by *rheoscopic* particles, which are small, flat particles that, when suspended in fluid, align themselves with the shear of the flow. We adopt the physical principles of real rheoscopic particles and apply them, in model form, to 3D velocity fields. By simulating the behavior and reflectance of these particles, we are able to render 3D simulations in a way that gives insight into the dynamics of the system. The results can be rendered in real time, allowing the user to inspect the simulation from all perspectives. We achieve this by a combination of precomputations and fast ray tracing on the GPU. We demonstrate our method on several different simulations, showing their complex dynamics in the process.

Index Terms—Rheoscopic fluid, flow visualization, tensor field visualization, ellipsoidal particle dynamics.

1 Introduction

Computational Fluid Dynamics (CFD) is a field with many applications in engineering and problems to be addressed in computer science. Many simulations are run before a real experiment is attempted, and the results are used to refine the system for the next simulation or to make sure the expensive experiment is a success. In some cases, the simulation is the final result if a real experiment is impractical or impossible. For both situations, it is crucial that the simulation results are interpreted correctly, which often requires that the simulation data are displayed to the researcher in a way that reveals its important aspects.

In this paper, we present a method for visualizing data from fluid dynamics simulations. We develop a method that takes the velocity field as input and uses the gradient tensor of that field as the basis for the visualization technique. Although our method does not make direct use of the velocity values, the visualizations obtained nevertheless inform about the velocities of regions as the simulation develops, while also adding information about the gradient strength and the boundaries between regions of different velocities, which are usually areas of high interest. All of this information is packed into an intuitive field of intensity values and thereby revealed in the animation. These intensity values can additionally be visually combined with other information that can be encoded as color or texture, e.g., densities, temperatures, etc.

The inspiration for our work comes from real *rheoscopic fluids*, specifically Kalliroscope, one of the more popular rheoscopic fluids, invented by artist Paul Matisse as a technique to display convection currents (http://www.kalliroscope.com). Rheoscopic literally means "flow showing" and this technique has become a well-known tool of experimental fluid dynamics researchers, adopted in many fluids laboratories to visualize flow behavior. Rheoscopic fluids consist of a liquid containing suspended particles. The particles are very small and cannot be seen individually, but they are reflective and their collective reflection can be seen. The particles are typically thin plates that reflect light preferentially in certain directions. For example, Kalliroscope AQ1000 is made from flat plate-like flakes of size $30 \times 6 \times 0.07\ \mu\mathrm{m}^3$ [24]. Because they are so small and have the same density as the liquid, they do not appreciably affect the flow in the liquid, but because of their triaxial dimensions, they will orient themselves with the shear of the flow (a nearly instant process, because of their small size). That is, their orientation depends entirely on the dynamics of the fluid. Since they primarily reflect light along the direction of their shortest axis, the collective reflectance of the shear-aligned particles gives a visual indication of their orientation, and thereby, the flow behavior, as demonstrated in Fig. 1.

Our method adopts the underlying physics of rheoscopic particle motion, reduced to a simplified model that gives computationally fast results by avoiding simulation of the large number of suspended particles. Through precalculation, we reduce most calculations at visualization runtime to those parts that are dependent on the environment and not the previously simulated velocity field. Combining a simple Lambertian reflectance model for the particles with a further approximating simplification of the rotating particle dynamics, we can create real-time visualizations where light position and view point can be changed interactively by the user. We also demonstrate a second approach where we relax this approximating simplification to make the visualization more accurate with regard to the real particle dynamics, at the cost of incorporating the light direction into the precalculation, therefore fixing the light position as a trade-off between interactivity and rendering fidelity. We apply a gradient magnitude measure to attenuate the reflectance from the particles to create smoother results, especially in areas with very low velocities and gradients. We also demonstrate that this magnitude measure itself is useful for flow visualization.

In the following parts of this paper, we take a look at previous work in the field of fluid dynamics visualization.

- *F. Hecht and G. Turk are with the Georgia Institute of Technology, College of Computing, 85 5th Street NW, Atlanta, GA 30032-0760. E-mail: florian.hecht@gmx.de, turk@cc.gatech.edu.*
- *P.J. Mucha is with the Department of Mathematics, University of North Carolina at Chapel Hill, Campus Box #3250, Chapel Hill, NC 27599-3250. E-mail: mucha@unc.edu.*

Manuscript received 12 Sept. 2008; revised 30 Jan. 2009; accepted 27 Mar. 2009; published online 17 Apr. 2009.
Recommended for acceptance by G. Scheuermann.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2008-09-0150.
Digital Object Identifier no. 10.1109/TVCG.2009.46.

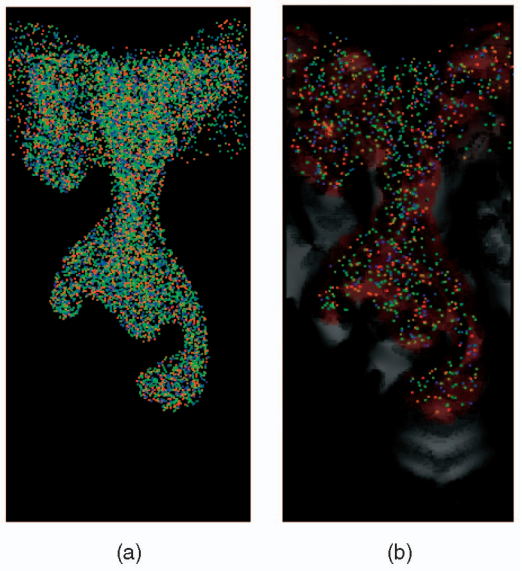

Fig. 1. Particle sedimentation simulation shown with (a) particles only and (b) a subset of the particles together with a rheoscopic lighting model.

After this, we introduce the underlying physical models for rheoscopic particle dynamics. We then describe how we use these principles to rapidly calculate the particle orientations and the resulting reflectances. We then provide implementation details and discuss the example animations our system created. We end with conclusions about this project and possible future extensions.

2 Related Work

In this section, we review techniques for displaying dense, fine-grained characteristics of flows. For many applications, such techniques should be complemented with the identification of high-level features of the flow such as vortex cores and critical points; however, our review of related work will only touch lightly on the visualization of such high-level features. Because our simulated rheoscopic approach is designed as a tool for fluid visualization, we review previous work on flow visualization below. In addition, because the orientation of our virtual particles is guided by a tensor quantity (the velocity gradient), we discuss previous methods for tensor field visualization.

2.1 Flow Visualization

One of the most common methods of displaying a vector field is the vector plot (sometimes called hedgehog illustrations). This is a dense collection of short-field-aligned line segments or arrows that are placed on a regular grid. Noise or jittering of the segment positions can break up visual artifacts due to the grid, as demonstrated by Crawfis and Max and also by Dovey [6], [9].

A different method for vector field visualization is to produce a dense texture. The earliest example of this approach is van Wijk's spot-noise, where many copies of a raster shape such as a circle or square are stretched in the vector field direction and summed together [35]. Cabral and Leedom introduced the line integral convolution (LIC) technique, which integrates a white noise field along a given vector field in order to generate long streaks in the directions of the field [4]. van Wijk makes use of fast texture display and blending in order to produce dense streaks, a technique known as image-based flow visualization (IBFV) [36]. The IBFV technique has also been extended to surfaces by van Wijk and Laramee et al. [37], [21]. Preußer and Rumpf use anisotropic diffusion to stretch noise images in the flow direction [29]. Sanderson et al. create 2D spots that are stretched in the flow direction using a reaction-diffusion process [30]. We recommend the review paper by Laramee et al. for a more detailed coverage of dense texture-based flow visualization techniques [22]. We note that some of these texture-related methods have at least some similarities with Particle Image Velocimetry (PIV), which is reviewed by Grant [11]. This is a lab technique where small particles are added to a flow and imaged by cameras. By analyzing the motion of the particles, fine-grain estimates of the velocity field can be made.

Another technique for flow visualization is to create a dense set of flow-aligned streamlines. Turk and Banks use a density-guided optimization process to produce such a dense collection of streamlines in 2D [34]. Jobard and Lefer greatly accelerate the formation of such streamline collections by seeding new streamlines that are near already-existing streamlines and terminating the new ones when they approach old ones too closely [15]. Several research groups have found streamline seeding strategies for 2D that speed up this process or improve the resulting placement of streamlines [38], [26], [23]. Illuminated streamlines have been created for 3D flows by Mattausch et al. using a seeding method similar to Jobard and Lefer [25]. Ye et al. perform an analysis of 3D vector fields in order to seed 3D streamlines at critical points [40].

2.2 Tensor Field Visualization

Although our work on virtual rheoscopic particles is designed for flow visualization, it depends on the analysis of the velocity gradient tensor. For this reason, we briefly mention some of the approaches that have been taken for visualizing tensor fields.

Several research groups have borrowed the notion of streamlines from vector fields and applied them to tensor fields. Delmarcelle and Hesselink trace what they call *hyperstreamlines* through 3D tensor fields [7]. For diffusion tensor MRI data, Weinstein et al. use advection-diffusion to trace paths through gridded data [39]. Zhukov and Barr use a regularization process to cross over noisy regions in tensor MRI data [43].

Laidlaw et al. display 2D diffusion tensor images using a dense collection of ellipsoids that show the direction and magnitude of the principal directions of the field [20]. Their imaging model is similar to that of many fine brush strokes. Kindlmann et al. use particle repulsion to pack elliptical glyphs in order to visualize tensor fields [19]. Jankun-Kelly and Mehta use oriented superellipsoid glyphs to show 3D tensor fields for liquid crystal data [13]. Kindlmann et al. created elliptical tensor field aligned features in 3D using a reaction-diffusion process [18]. These same authors use an anisotropic shading model for displaying surfaces that are embedded in tensor fields, where the direction of anisotropy is aligned with an eigenvector of the tensor at a given position [17], [18].

Some researchers have generalized the line integral convolution method to tensor fields. Zheng and Pang

perform 2D convolution that is guided by a tensor field in order to produce a dense texture [41]. Hotz et al. overlay two LIC-style textures in order to visualize the two eigenvectors of a tensor field [12].

We end our review of tensor field techniques by noting that there are topological approaches for the visualization of tensor fields. This line of work was begun by Delmarcelle and Hesselink for 2D symmetric tensor fields [8]. We recommend the work of Trcioche et al. [33] and Zheng et al. [42] for more recent developments in this area.

2.3 Previous Methods for Virtual Rheoscopic Visualization

Because of its potential for visual comparison with experiments and its applicability to a wide variety of simulated fluid flow, it is not surprising that previous visualization work has considered virtual reproduction of rheoscopic effects. Gauthier et al. compute the full dynamical motion of triaxial ellipsoids seeded in the flow, including numerical study of the asymptotic trajectories and the transition times to reach them, to simulate laser sheet visualizations of some classical flows of interest [10]. Their resulting images were created by a Monte Carlo approach of randomly simulating the motion of many particles with various initial orientations. A drawback of this approach is that the final images are quite noisy. More recently, Barth and Burns have developed a virtual rheoscopic fluid method that uses a simplified model for particle orientation [2]. Specifically, they replace the particle dynamics by a heuristic definition for the normal to the shear layers, which may not actually correspond to the reflecting surface normal of any real, finite-sized particle and does not address the commonly occurring case of continuously rotating particles (indeed, both unidirectional flow regimes they discuss for intuition correspond to the Jeffery orbit case for axisymmetric particles discussed below). Nevertheless, their method provides a valuable visualization tool, producing images with many of the same visual qualities as real rheoscopic fluids.

Our own work builds on both of these previous methods, combining the best aspects of each. Like Barth and Burns, we use volume rendering to create our final images and a dynamical model of particle orientation that is similar to the work of Gauthier et al.. There are several ways in which our own work departs from these previous methods, however. First, we arrive at a closed-form description of the lighting even for the case of particles that continue to rotate. Thus, our method is faster and less noisy than the Monte Carlo approach of Gauthier et al. and handles the dynamic motion that is not treated by Barth and Burns. Second, we have included an attenuation term that can be used to improve the visual clarity of virtual rheoscopic visualization, and that can also be used as a flow visualization method by itself. Finally, we have applied our new visualization techniques to several flow data sets that are different than the data that were visualized in these previous papers.

3 Rheoscopic Particles

Our rheoscopic visualization method is based on a physical model for the orientations of nonspherical particles that are too small to see, but whose local aggregate orientation reflects light incoming from a particular direction. We discuss the dependence of the particle orientations on the flow in this section, postponing the issues about light reflection to a later section.

Real rheoscopic particles are typically nearly flat plates, with the flat surface much longer in one direction than another. For the purposes of analysis, it is much easier to approximate the particles as being ellipsoids at large aspect ratio. Because of the dimensions of real rheoscopic particles, e.g., Kalliroscope AQ1000 flakes are $30 \times 6 \times 0.07\ \mu\mathrm{m}^3$ [24], we are primarily interested in cases where all three axes of an ellipsoid are different; but we find that these large aspect ratio cases are mathematically connected to those of axisymmetric ellipsoids (two axes of equal length).

While considering the rotation of ellipsoids under shear is not exhaustive of all possible particle behaviors, symmetry considerations do restrict the equations of motion of particles with more general shape. For instance, the rotational response of any particle with a 90-degree rotation symmetry about an axis and a reflection symmetry through a plane including that same axis is characterized by a single parameter and is equivalent to some axisymmetric ellipsoid provided that the particle is not too long along its symmetry axis [3]. A particle with such symmetries relative to an axis along its thin direction thus rotates like some oblate spheroid. Without the 90-degree rotation symmetry, clearly lacking in the Kalliroscope flakes, the situation becomes more complicated. For instance, the rotational response of a particle with reflection symmetries through three mutually perpendicular planes is characterized by three independent parameters, not the two parameters of a general ellipsoid (its aspect ratios, as described below), so the rotation of such a particle is not generally equivalent to some ellipsoid [3]. Nevertheless, our analysis below eventually yields a visualization method rooted in a large aspect ratio limit where neither of these parameters matters. We conjecture (without proof) that this is the same limit obtained at high aspect ratio in the more general three-parameter case of nonellipsoidal particles with such reflection symmetries (such as symmetric thin plates), with the same long-axis and short-axis behaviors we obtain below.

Because of the dimensions of the particles and their density, their Reynolds number is assumed to be very small. We assume that the particles are (sufficiently) neutrally buoyant, neglect Brownian motion, and ignore the modification of the flow due to the ellipsoids themselves because the aggregate orientation of a sufficiently low solid volume fraction collection of particles can be obtained by the study of the effect of the unperturbed flow on a single particle. The translational motion of the particles is therefore primarily a result of direct advection by the prescribed flow. Assuming that the prescribed flow is locally nearly linear on the small scale of the particles, the instantaneous rotation and hence the orientation depends on the gradient of the flow local to the particle. Since we are only interested in the orientation for the reflectance, and further assume that the particle orientations respond quickly compared to the time rate of change of the local gradient (due to advection and changes in the flow), we hereafter ignore particle advection and only focus on the orientation of an ellipsoid in a specified, static velocity gradient; such reasonable approximations additionally avoid tumbling and chaotic motions (e.g., [31], [10]) which would unnecessarily complicate the model calculation of particle orientations here.

Given a velocity field $\mathbf{v}(\mathbf{x}) = (v_x, v_y, v_z)$, in some Cartesian coordinate system, the gradient velocity tensor at a location is defined in that same coordinate system as

$$\mathbf{G} = \begin{Bmatrix} \frac{\partial v_x}{\partial x} & \frac{\partial v_x}{\partial y} & \frac{\partial v_x}{\partial z} \\ \frac{\partial v_y}{\partial x} & \frac{\partial v_y}{\partial y} & \frac{\partial v_y}{\partial z} \\ \frac{\partial v_z}{\partial x} & \frac{\partial v_z}{\partial y} & \frac{\partial v_z}{\partial z} \end{Bmatrix}.$$

To determine the instantaneous rotation of a particle, it is convenient to split the velocity gradient tensor into two parts: the (antisymmetric) vorticity tensor $\mathbf{\Omega} = (\mathbf{G} - \mathbf{G}^T)/2$ and the (symmetric) rate-of-strain tensor $\mathbf{E} = (\mathbf{G} + \mathbf{G}^T)/2$, with $\mathbf{G} = \mathbf{\Omega} + \mathbf{E}$. It is further convenient to work in a right-handed orthonormal reference frame built along the instantaneous axes of the ellipsoid, notated here as $\hat{\mathbf{a}}, \hat{\mathbf{b}}$, and $\hat{\mathbf{c}}$, directed along the axes of lengths a, b, and c, respectively (where we will typically take $a \geq b \geq c$). The actual lengths of the axis are unimportant for the rotational motion of the particle in a linear shear flow, but the aspect ratios $r_1 = b/a, r_2 = c/a$ (typically $r_2 \leq r_1 \leq 1$ per above) determine how $\mathbf{\Omega}$ and $\mathbf{E}$ affect the angular velocity $\boldsymbol{\omega}$ of the particle, as established by Jeffery [14], elucidated in the ellipsoid-axes frame by, e.g., Gauthier et al. [10]:

$$\begin{aligned} \omega_1 &= \frac{b^2 G_{32} - c^2 G_{23}}{b^2 + c^2} = \Omega_{32} + \frac{r_1^2 - r_2^2}{r_1^2 + r_2^2} E_{32}, \\ \omega_2 &= \frac{c^2 G_{13} - a^2 G_{31}}{a^2 + c^2} = \Omega_{13} + \frac{r_2^2 - 1}{r_2^2 + 1} E_{13}, \\ \omega_3 &= \frac{a^2 G_{21} - b^2 G_{12}}{a^2 + b^2} = \Omega_{21} + \frac{1 - r_1^2}{1 + r_1^2} E_{21}. \end{aligned} \tag{1}$$

The above specification of $\boldsymbol{\omega}$ can be used to compute evolving particle orientations by way of simple numerical integration. Specifically, the orientation of a particle can be described by a 3×3 rotation matrix $\mathbf{R}$ describing the rotation from the base (laboratory or simulation) coordinate frame to the ellipsoid-axes frame. This same rotation matrix yields $\mathbf{G}$ in the particle frame from that in the lab frame: $\mathbf{G}_{\text{particle}} = \mathbf{R}^T \mathbf{G}_{\text{lab}} \mathbf{R}$, from which $\mathbf{\Omega}$, $\mathbf{E}$, and finally, $\boldsymbol{\omega}$ are calculated. A simple forward Euler-like integration in the space of rotation matrices is then obtained by calculating a rotation matrix $\tilde{\mathbf{R}}$ as an exponential map of a dt multiple of the antisymmetric matrix representing the appropriate cross product with $\boldsymbol{\omega}$ and updating the new $\mathbf{R}$ to be $\mathbf{R}\tilde{\mathbf{R}}$. Gauthier et al. use similar numerical integrations of particle orientations over time in their visualization calculations. In the present work, we employ a different approach based on direct, though approximate calculation of the long-time statistically steady particle dynamics in a specified gradient field.

3.1 Long-Time Orientations of Axisymmetric Particles

Throughout the remainder of this work, we assume that the velocity gradients in the vicinity of a given particle evolve sufficiently slowly (by changes in the fluid fields and by direct advection of the particles) that the particle orientations can be approximated as being in their statistically steady long-time orientational distributions given an instantaneous, spatially localized velocity gradient $\mathbf{G}$. Such long-time orientations of axisymmetric particles were determined, in a classic paper by Bretherton [3], in terms of the eigenvalues and eigenvectors of a tensor related to $\mathbf{G}$. In the present notation, if we take $b = c$, the optical behavior of the axisymmetric particle is fully specified by the direction $\hat{\mathbf{a}}$. Expressing the single remaining aspect ratio as $r = b/a$ ($r < 1$ corresponding to prolate spheroids, $r > 1$ for oblate spheroids), the instantaneous angular velocity $\boldsymbol{\omega}$ in (1) becomes

$$\boldsymbol{\omega} = \begin{cases} \Omega_{32} \\ \Omega_{13} + \dfrac{r^2 - 1}{r^2 + 1} E_{13}, \\ \Omega_{21} + \dfrac{1 - r^2}{1 + r^2} E_{21}, \end{cases}$$

and the instantaneous rate-of-change of $\hat{\mathbf{a}}$, $\frac{d}{dt}\hat{\mathbf{a}} = \boldsymbol{\omega} \times \hat{\mathbf{a}}$, is

$$\frac{d}{dt}\hat{\mathbf{a}} = \hat{\mathbf{b}}\left(\Omega_{21} + \frac{1 - r^2}{1 + r^2} E_{21}\right) + \hat{\mathbf{c}}\left(\Omega_{31} + \frac{1 - r^2}{1 + r^2} E_{31}\right) \tag{2}$$

because of the symmetry (antisymmetry) of $\mathbf{E}$ ($\mathbf{\Omega}$). Defining

$$\mathbf{\Xi} = \mathbf{\Omega} + \frac{1 - r^2}{1 + r^2}\mathbf{E}, \tag{3}$$

the evolution equation (2) becomes simply

$$\frac{d}{dt}\hat{\mathbf{a}} = \mathbf{\Xi} \cdot \hat{\mathbf{a}} - \Xi_{11}\hat{\mathbf{a}}. \tag{4}$$

Bretherton insightfully observed that a coordinate-free form for the rate-of-change of the a-axis direction could thus be obtained by removing the constraint that it remain unit length. That is, the Ξ_{11} term in (4) is the only frame-dependent quantity in the expression, but it acts only along the a-axis; therefore, removing this term has no effect on the direction of $\mathbf{a}$, only its magnitude, and so one can instead more simply investigate the evolution of a (nonunit) vector $\tilde{\mathbf{a}}$ that evolves according to

$$\frac{d}{dt}\tilde{\mathbf{a}} = \mathbf{\Xi} \cdot \tilde{\mathbf{a}}. \tag{5}$$

Importantly, the above expression is coordinate-free and true in all coordinate frames, including the fixed "lab" frame of the simulation. Moreover, because (5) is a constant-coefficient linear system, the long-time behavior of its solutions is dictated by the dominant eigenvalues and associated eigenvectors of $\mathbf{\Xi}$. That is, the eigenvalues and eigenvectors of $\mathbf{\Xi}$ determine the long-time limiting distribution of orientations and how quickly the particle orientation relaxes to this distribution.

Following, e.g., the review of Bretherton's results in [10], noting that the roots of the (third-order, real coefficient) characteristic polynomial of $\mathbf{\Xi}$ must sum to zero for divergence-free velocity fields, there are either three real eigenvalues or one real eigenvalue with a complex conjugate pair. There are then four cases, considering the number and type of dominant (most positive) eigenvalues of $\mathbf{\Xi}$, as follows:

1. A single dominant real eigenvalue: The corresponding eigenvector indicates the long-time direction of $\hat{\mathbf{a}}$.

2. A dominant complex conjugate pair: The $\hat{\mathbf{a}}$ vector rotates in the plane spanned by the associated eigenvectors.
3. A dominant real eigenvalue of double multiplicity: A given particle evolves toward a fixed orientation, in the plane spanned by the associated eigenvector pair and dependent on its initial orientation or, in the degenerate case, in the direction of the single associated eigenvector.
4. All eigenvalues have zero real part: Either the particle maintains its initial orientation (no nonzero eigenvalues) or undergoes a Jeffery orbit [14], tracing out a nonplanar closed curve that depends on its initial orientation.

3.2 Approximate Orientations of Triaxial (Scalene) Particles

The general triaxial (scalene) particle case is substantially more difficult than the axisymmetric situation elucidated by Bretherton and summarized above. Much of the Bretherton axisymmetric analysis can nevertheless be used to approximately model the triaxial Kalliroscope particle orientations, because of the extreme aspect ratios of the particle axes. With $r_2 \approx 2 \cdot 10^{-3}$ and $r_1 = 0.2$, noting that these aspect ratios always appear squared in (1), we are motivated to study the $r_2 \ll r_1 \ll 1$ limit.

We start our consideration of this extreme aspect ratio limit by rewriting the general triaxial-ellipsoid dynamics of (1) in a form similar to (2), from $\frac{d}{dt}\hat{\mathbf{a}} = \boldsymbol{\omega} \times \hat{\mathbf{a}}$ and $\frac{d}{dt}\hat{\mathbf{c}} = \boldsymbol{\omega} \times \hat{\mathbf{c}}$:

$$\begin{aligned} \tfrac{d}{dt}\hat{\mathbf{a}} &= \hat{\mathbf{b}}\left(G_{21} - \frac{2r_1^2}{1+r_1^2}E_{21}\right) + \hat{\mathbf{c}}\left(G_{31} - \frac{2r_2^2}{1+r_2^2}E_{31}\right), \\ \tfrac{d}{dt}\hat{\mathbf{c}} &= \hat{\mathbf{a}}\left(\frac{2r_2^2}{1+r_2^2}E_{31} - G_{31}\right) + \hat{\mathbf{b}}\left(\frac{2r_2^2}{r_1^2+r_2^2}E_{32} - G_{32}\right) \end{aligned} \tag{6}$$

($\hat{\mathbf{b}} = \hat{\mathbf{c}} \times \hat{\mathbf{a}}$ by orthonormality). Neglecting $O(r^2)$ terms then indicates that the long axis $\hat{\mathbf{a}}$ evolves approximately according to

$$\frac{d}{dt}\hat{\mathbf{a}} \approx G_{21}\hat{\mathbf{b}} + G_{31}\hat{\mathbf{c}}, \tag{7}$$

while the short axis $\hat{\mathbf{c}}$ approximately follows

$$\frac{d}{dt}\hat{\mathbf{c}} \approx -G_{31}\hat{\mathbf{a}} - G_{32}\hat{\mathbf{b}}. \tag{8}$$

Again employing Bretherton's consideration of removing the unit-length constraint, analogous to the development of equations (2)-(5), the directions of the $(\hat{\mathbf{a}}, \hat{\mathbf{c}})$ pair which exactly obey the approximate dynamics in (7) and (8) are identical to the directions of the (nonunit) $(\tilde{\mathbf{a}}, \tilde{\mathbf{c}})$ pair that evolve according to

$$\frac{d}{dt}\tilde{\mathbf{a}} = \mathbf{G} \cdot \tilde{\mathbf{a}}\,, \quad \frac{d}{dt}\tilde{\mathbf{c}} = -\mathbf{G}^T \cdot \tilde{\mathbf{c}}. \tag{9}$$

From these equations, the eigenvector analysis proceeds similarly to the axisymmetric case of the previous section, with the change in detail that the long-time orientational behavior of the long axis $\tilde{\mathbf{a}}$ is determined by the eigenvalues and eigenvectors of the velocity gradient $\mathbf{G}$, while that of the short axis $\tilde{\mathbf{c}}$ follows those of the negative transpose of the gradient $-\mathbf{G}^T$.

These statements of the approximate dynamics capture the intuitively and empirically well-known behaviors of such particles that they align with the principal directions of the local velocity gradient $\mathbf{G}$. That is, the particles "align with the shear layers" where such layers exist, but are allowed to rotate with the fluid motion where appropriate. Importantly, because the eigenvector analysis avoids any direct integration of the approximate differential equations of the dynamics (corresponding to unphysically long and thin particles), the behaviors obtained under the eigenvector analysis correspond to dynamics which are structurally stable for real, finite-sized particles. We additionally remark that these limiting behaviors are the same as those of the high aspect ratio limiting approximations of the axisymmetric prolate and oblate spheroids, respectively, that is, for $r \ll 1$, (3) reduces to $\Xi \approx \boldsymbol{\Omega} + \mathbf{E} = \mathbf{G}$, while for $r \gg 1$, the approximation becomes $\Xi \approx \boldsymbol{\Omega} - \mathbf{E} = -\mathbf{G}^T$ (by the symmetries of $\boldsymbol{\Omega}$ and $\mathbf{E}$). An eigenvector analysis similar to Bretherton's then follows, with $\mathbf{G}$ and $-\mathbf{G}^T$ playing the lead roles. Alternatively, to the same degree of approximation, these can be replaced with Bretherton's evolution tensors (3) at appropriate high aspect ratios.

Because the eigenvalues of $\mathbf{G}$ and $\mathbf{G}^T$ are the same, there are three nongeneric cases (ignoring double eigenvalues and real parts precisely equal to zero, being structurally unstable under the approximate dynamics) for the long-time behavior of $(\tilde{\mathbf{a}}, \tilde{\mathbf{c}})$.

1. Three real (unequal) eigenvalues: The (right) eigenvector of $\mathbf{G}$ associated with the most positive eigenvalue indicates the direction of $\tilde{\mathbf{a}}$ and the eigenvector of $\mathbf{G}^T$ (equivalently, the left eigenvector of $\mathbf{G}$) associated with the most negative eigenvalue indicates the direction of $\tilde{\mathbf{c}}$.
2. A positive-real-part complex conjugate pair: The left eigenvector associated with the negative (real) eigenvalue indicates the direction of $\tilde{\mathbf{c}}$, while the $\tilde{\mathbf{a}}$ vector rotates in the perpendicular plane (spanned by the complex right eigenvectors).
3. A negative-real-part complex conjugate pair: The left eigenvectors associated with the complex negative eigenvalues indicate the plane of rotation of $\tilde{\mathbf{c}}$, while $\tilde{\mathbf{a}}$ points perpendicularly in the direction of the right eigenvector associated with the positive eigenvalue.

Because the reflectance model presented in the next Section relies solely on the orientation of the short axis $\tilde{\mathbf{c}}$, each of the first two cases above lead immediately to the required (approximately) orientation information by a direct calculation of the eigenvectors of $-\mathbf{G}^T$. The rotation of the long axis in the second case does not complicate the calculation of the short-axis direction at this lowest order level of approximation. Returning momentarily to the full dynamics of (6), such rotation of the long axis will lead through the $O(r^2)$ terms to small oscillation in the "fixed" direction of the short axis, as easily verified in numerical solutions of the untruncated orientational dynamics. However, the lowest order remains a reasonably good approximation at the aspect ratios of Kalliroscope particles, and at lowest order, the only (generic) complicated case for the long-time dynamics of $\tilde{\mathbf{c}}$ is the third, rotating-short-axis case, described in more detail below.

3.3 Orientational Distribution in the Rotating-Short-Axis Case

Ignoring the nongeneric cases of double eigenvalues and real parts precisely equal to zero, the only required detail remaining from the above model calculations is the long-time distribution of orientations of $\tilde{\mathbf{c}}$ in the case where this axis is rotating. Physically, if there are many small Kalliroscope particles near a particular point in space (e.g., a grid point of the simulation), then the orientations of those particles at an arbitrary point in time are approximately those drawn randomly from the orientational distribution of a single particle at that point, averaged over long times (as described at the level of the approximate dynamics considered, and assuming sufficient ergodicity from other, ignored interactions that particle-particle orientational correlations are unimportant). That is, we assume that the instantaneous ensemble of local orientations is the same as the temporal distribution of the orientation of a single particle during the periodic motion of the $\tilde{\mathbf{c}}$ vector at long times.

If the angular rate of rotation was uniform, independent of the angle of rotation, then the orientational distribution would be uniformly drawn from the directions spanned by the plane of the associated complex eigenvectors (perpendicular to the calculated long-time $\tilde{\mathbf{a}}$ direction). However, this rotating motion can include very large differences in the rate of rotation during the period, in some cases with the particle direction being almost steady for some time, followed by a rapid half-rotation to being nearly steady in the opposite orientation (equivalently almost steady by symmetry). Therefore, even though we will below consider the visualization effect of simply assuming uniform rotation, we need first determine the correct nonuniform rotational motion.

Starting from the observation that the $\tilde{\mathbf{a}}$ direction of the long axis is in fixed, stable equilibrium when $\tilde{\mathbf{c}}$ is rotating at long times (again, at the lowest order approximation of the rotational dynamics assumed), the velocity gradient in the instantaneous $(\tilde{\mathbf{a}}, \tilde{\mathbf{b}}, \tilde{\mathbf{c}})$ frame has only two nonzero elements off the diagonal, G_{23} and G_{32}. If we fix the frame based on an arbitrarily selected particle orientation from among the periodic motions, declaring $\mathbf{G}^{(0)}$ as the velocity gradient in that frame, the particle orientation at a different point in the period is only a rotation $\mathbf{R}$ by angle $\theta(t)$ around the $\tilde{\mathbf{a}}$-axis. Examining the elements of the rotated velocity gradient $\mathbf{R}^T\mathbf{G}^{(0)}\mathbf{R}$, it follows from (9) that $\theta(t)$ obeys the differential equation

$$\frac{d\theta}{dt} = \left(G_{22}^{(0)} - G_{33}^{(0)}\right)\sin\theta\cos\theta + G_{23}^{(0)}\sin^2\theta - G_{32}^{(0)}\cos^2\theta\,. \tag{10}$$

While integration then yields an implicitly defined $\theta(t)$, the differential equation itself provides sufficient information for the illumination model presented below, because the fractional period of time spent near angle θ is inversely proportional to $d\theta/dt$.

4 Visualization with Virtual Particles

We have created a flow visualization system that uses the reflectance of virtual rheoscopic particles to view 3D flow data. We assume that the velocity field is provided on a regular 3D grid. We volume render this data and use the virtual rheoscopic particles to calculate reflectance values in the volume. Particles in different regions will orient themselves in different directions and this shows up as variations in intensity across the flow. This method can be used to create static images, but it is even more effective for visualizing time-varying flows.

We adopt the simplifying particle model assumptions of the previous section and its focus on the few (generic) cases of long-time particle motion. This model enables immediate calculation of a rheoscopic response directly from the local velocity gradient, without actually simulating any rheoscopic particles. We also use a gradient strength measure to model the uniformity of particle orientations throughout the volume, to model interfering attenuation of the rheoscopic effect. The case of rotating short axes is treated by two models, results from which are compared with one another: 1) a simple uniform-rotation lighting model that enables the fast calculation of the reflectance of rotating particles and 2) a full model of the nonuniform rotation speeds through incorporating the light direction in the eigenvector analysis.

4.1 Bretherton-Like Model Adoption

Where the Bretherton model accurately describes the long-time behavior of an axisymmetric ellipsoid, the "Bretherton-like" model of the previous section achieves a similar, if approximate, description for high-aspect-ratio ellipsoids. Utilization of this model leads to a significant runtime improvement over full dynamical simulation of particle motions (e.g., as used by Gauthier et al.), while maintaining the connection between the physical motions of the microscopic particles and the macroscopic imaging effect. The full simulation method requires evolving and tracking the orientations of many particles, some of which are rotating rapidly, possibly for a large number of frames. In contrast, the simplifying assumptions of the previous section yield the long-time behavior of particles through a simple eigenvector analysis of the velocity gradient, at the cost of the high-aspect-ratio-limit approximation and the assumption that transition times to this limiting behavior are not important in this model, although these could, in principle, also be calculated and utilized, for instance, in modeling coherence and attenuation. Instead, we will later discuss using the gradient magnitude as an attenuation measure to enhance the visualizations.

The enumeration of the three generic cases for triaxial particles in Section 3.2 deliberately correspond only to the first two of the four cases in Bretherton's description of the motion of axisymmetric particles (as in Section 3.1), because the degeneracy of the last two cases correspond to long-time dynamics that depend on the initial orientation of a particle. Handling such cases would require the tracking of the history of particles, which is something to avoid for fast visualization. At the same time, such cases are rare because they require numerically precise equalities between the eigenvalues. Indeed, the impact of not incorporating these cases is minimal, since they represent very specific (often planar) flow situations and account for less than 2 percent of the grid points in our example data sets. Nevertheless, the eigenvector analysis identifies these cases and this information can be used to mark the locations where this

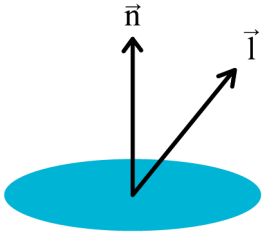

Fig. 2. The reflectance model of the plates (axisymmetric). The normal is only shown on one side, but the plate also reflects on the other side.

happens, which might be interesting information for certain data sets. In the absence of such degeneracies, the eigenvector analysis indicates, for an instantaneously specified velocity gradient, that the short axis is either driven toward a fixed orientation or rotates in a plane. This information is then used to calculate the reflectance in a manner that we will now describe.

4.2 Reflectance Model

We model reflectance from the particles as that from infinitely thin flat plates that reflect on both sides as ideal diffuse (Lambertian) reflectors. This assumption is important for calculating a closed-form reflectance for rotating particles. We take the shortest axis direction of the particle (determined by the calculation of $\tilde{\mathbf{c}}$ in the previous section) to be the unit normal $\vec{n}$ of the reflecting surface in calculating the intensity

$$I_{fixed} = A \cdot |\vec{l} \bullet \vec{n}|,$$

where $\vec{l}$ is the unit vector in the direction to the light source and A is the albedo of the particles (see Fig. 2).

In the case where the short axis of the particle is rotating (Section 3.3), the normal $\vec{n}$ is rotating around an orthogonal direction specified by unit vector $\vec{d}$ (specified by $\tilde{\mathbf{a}}$ in this case). Simplistically replacing the true rotating motion with a uniform rotation speed, we can find a preintegrated solution (similar to [16]) to the reflectance from a whole rotation, and thus, avoid calculating the integral at runtime:

$$I_{rot} = A \cdot \frac{1}{2\pi} \int_0^{2\pi} |\vec{l}^{\,T} \mathbf{R}(\theta, \vec{d}) \vec{n}| d\theta,$$

where $\mathbf{R}(\theta, \vec{d})$ is the rotation matrix that rotates around the axis $\vec{d}$ by the angle θ. If we view this with regard to the light direction like in Fig. 3, we see that the angle α between the light direction $\vec{l}$ and the axis of rotation $\vec{d}$ determines the amount of reflected light. Since the plates reflect on both sides, α can always be taken between 0 and $\frac{\pi}{2}$, and the expression for intensity simplifies to

$$I_{rot} = A \cdot \frac{\sin\alpha}{2\pi} \int_0^{2\pi} |\sin\theta| d\theta = A \cdot \frac{2\sin\alpha}{\pi}.$$

The above expression provides a very simple and efficient way to calculate the reflectance from a rotating particle, given the axis of rotation, under the simplifying assumption that its rotational motion is uniform. Unfortunately, as per Section 3.3, particle rotation is not always at approximately uniform rates, so this can be a

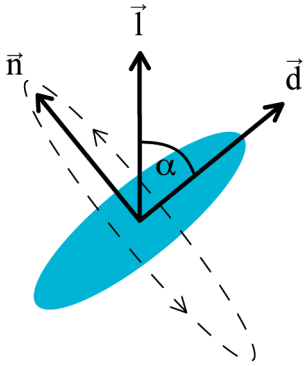

Fig. 3. The reflectance model oriented around the light direction.

poor approximation, but the resulting intensities have often appeared to be good enough approximations in our visualizations. See Fig. 4 for a comparison between the two cases.

4.3 Nonuniform Rotation Speed

In order to treat the rotating-short-axis case with greater accuracy, we can incorporate the nonuniformity of rotation speed into the integral precalculation at the cost of fixing the light source, as such calculation requires again the results from the eigenvector analysis. Specifically, the $d\theta$ integration factor in the intensity expressions of the above section originated as dt (time) factors, the simplifying assumption of uniform rotation yielding the two equivalent for averaging purposes.

The intensity expression accounting for nonuniform rotation is

$$I_{rot} = A \cdot \frac{\sin(\alpha)}{T} \int_0^T |\sin\theta(t)| dt,$$

with T any integer multiple of the half period of the rotation. This integral can be re-expressed as an averaging

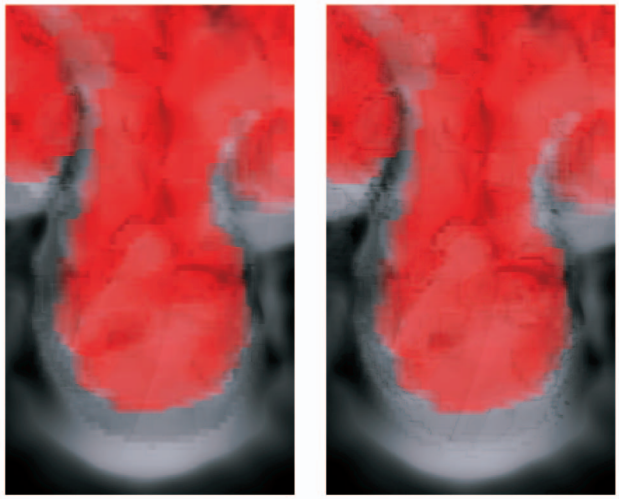

Fig. 4. Detail pictures comparing (a) the uniform and (b) precalculated nonuniform reflectances for rotations, in a frame where 26 percent of the volume is in a rotating case.

integral over θ by the substitution $dt = (\frac{d\theta}{dt})^{-1} d\theta$. The necessary $\frac{d\theta}{dt}$ derivative, appearing in (10) in terms of the velocity gradient in a fixed frame corresponding to the axes of the particle at a specified time, can then be expressed as

$$\frac{d\theta}{dt} = \frac{G^{(0)}_{22} - G^{(0)}_{33}}{2} \sin 2\theta - \frac{G^{(0)}_{23} + G^{(0)}_{32}}{2} \cos 2\theta + \frac{G^{(0)}_{23} - G^{(0)}_{32}}{2},$$

at the cost of specifying the distinguished frame for the $\mathbf{G}^{(0)}$ components as that in which the surface normal is perpendicular to the direction of the light source ($\sin\theta = 0$). The resulting I_{rot} below requires recalculation if the light source is moved.

Because of the arbitrariness of multiplicative factors of angular velocity canceling in the ratio leading to I_{rot}, we find it convenient to define $m_{a,b}(\theta) = 1 + a\sin(2\theta) + b\cos(2\theta)$, with a and b obtained from the velocity gradient in the specified frame as

$$a = \frac{G^{(0)}_{22} - G^{(0)}_{33}}{G^{(0)}_{23} - G^{(0)}_{32}}, \qquad b = -\frac{G^{(0)}_{23} + G^{(0)}_{32}}{G^{(0)}_{23} - G^{(0)}_{32}}.$$

Integrating over a half period at angular velocity $m_{a,b}(\theta)$, the reflected intensity from the nonuniformly rotating particle becomes

$$I_{rot}(\alpha, a, b) = A \cdot \sin(\alpha) \frac{\int_0^\pi \frac{\sin(\theta) d\theta}{m_{a,b}(\theta)}}{\int_0^\pi \frac{d\theta}{m_{a,b}(\theta)}}.$$

Given a specified velocity gradient, the a and b values can be obtained from the above definitions and the integrals in the I_{rot} expression can be evaluated numerically. In practice, the expression can be reasonably approximated ($\sim$10 percent) with the integrals evaluated numerically by a simple trapezoid approximation over a small number of points.

4.4 Gradient Measure

In a real rheoscopic fluid, there would be many particles in each cell of the volume. We make the assumption that if the velocity gradient at a cell is zero (or close to zero), the orientations of the particles would be uniformly distributed over all orientations. In such a situation, there would not be a strong directional component to the particle reflectances. The stronger the gradient becomes, the more particles will have orientations that match the result from the eigenvector analysis. For this reason, we provide the option of using the strength of the velocity gradient to heuristically attenuate the opacity of the virtual rheoscopic particles. In practice, this amounts to changing the opacity on a per-voxel basis.

As a measure of the strength of the gradient, we use

$$\|\xi\| = |\lambda_1|^2 + |\lambda_2|^2 + |\lambda_3|^2,$$

where λ_i are the complex eigenvalues of ξ. A more thorough analysis of the particles' orientational distribution would include effects due to the locally advected changes in fluid velocity and the time scale of relaxation to the long-time orientations, including differences in the real parts of the eigenvalues. We additionally note that we do not directly include the direction of view in modeling the intensity of light scattered toward the viewer here. While the cross-sectional area of an individual particle drops off like $|\vec{n} \cdot \vec{e}|$ as eyed from direction indicated by unit vector $\vec{e}$, the total cross-sectional particle area per unit volume does not decay appreciably if the particle density is sufficiently high, until a sudden drop off when viewed edge on (ignored here). Nevertheless, the view direction plays an important indirect role in the total intensity observed via the ray tracing (described below). If preferred, it is straightforward to further attenuate opacity by this $|\vec{n} \cdot \vec{e}|$ factor, which is the dilute-limit behavior, or by the computationally more convenient $(\vec{n} \cdot \vec{e})^2$ (as in [2]). Thus, while we find the present intensity model and gradient strength opacity heuristic to work well in practice, we are confident that other choices would also be visually useful. Fig. 9 shows examples of the same sedimentation scene visualized with unattenuated rheoscopic reflectances and with opacity attenuation according to the gradient magnitude. On the other hand, we have, on some occasions, found it useful to create visualizations based solely on the gradient magnitude, without any particle orientation analysis, such as for the von Kármán vortex street in Fig. 6.

5 Implementation

5.1 Preprocessing

This section describes the preprocessing that we perform on our velocity field data prior to final image synthesis. This preprocessing step only needs to be performed once and is independent of view position. We perform eigenvector analysis as a preprocessing step and this requires the calculation of (possibly complex) eigenvalues for each cell in the volume. We use a generic method (LAPACK) to calculate the eigenvalues and eigenvectors. A specialized method might speed up the process for runtime evaluation. For each cell, we store one vector that represents either the dominant orientation or the axis of rotation of the particles. We also store scalar values specifying the proper case for each cell (1, 2, or 3, described in Section 3.2), the intensity value for the nonuniform rotating case, the gradient magnitude, and possibly a density value (e.g., fluid density in the internal splash or local particle density in sedimentation). All quantities are stored as 32-bit floating-point values.

If the resolution of the original data is not fine enough, the final images will show voxelization artifacts. The quality of the visualization can be improved by subsampling the original velocity field using trilinear interpolation. This reduces the artifacts but increases the amount of data that has to be preprocessed. We employ this technique with some of our data sets and increase the resolution by a factor of 1.5 to 2.

Our largest data set was the internal splash simulation that is shown in Fig. 7. This data set required about 16 seconds per frame of preprocessing time on a 3.0-GHz Core 2 Duo (E8400) processor. The resolution of this simulation was $120 \times 120 \times 30$ grid points and the resolution along each dimension was increased by a factor of $1.5\times$ for the precalculations, which then becomes $\sim$34 Mbytes per frame. This sequence had 126 frames and this quantity of floating point data does not fit into main memory on a 32-bit system. In such cases when the data are too large to hold in memory, we streamed the frames into main memory, but our implementation is not optimized and is hence currently not fast enough for real-time replay. The vortex street data are also too big for the main memory on our system because of its length of 300 frames and the doubling in resolution. The sedimentation data fit into main

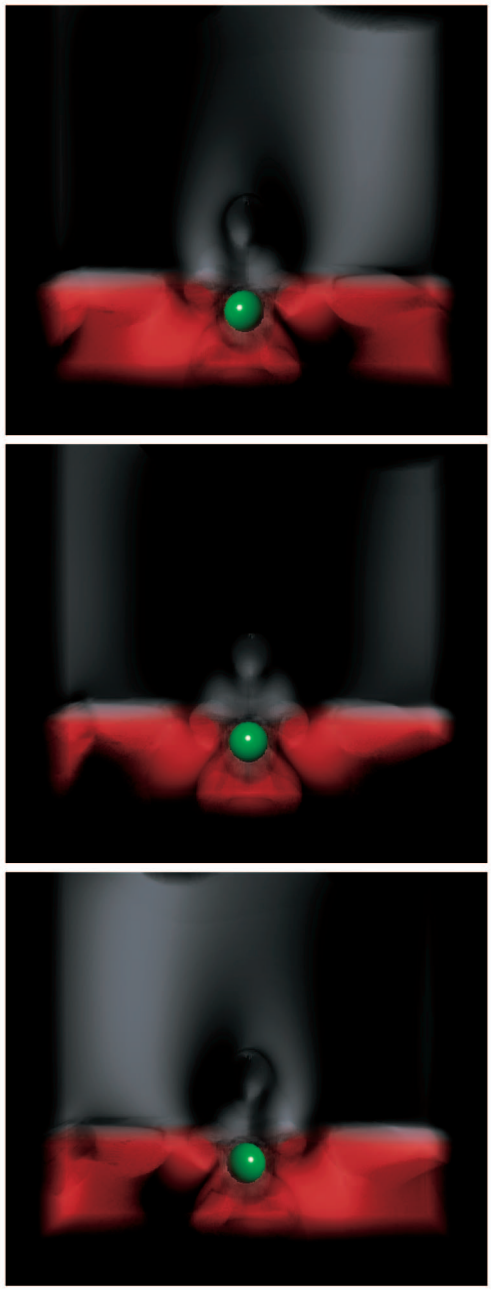

Fig. 5. Internal splash, varying the light position.

memory, as do versions of the previous data sets without the resolution increase, thus allowing real-time playback. See Table 1[1] for more details.

5.2 Rendering

We used an nVidia GeForce 8800 GFX graphics card for rendering and make use of 32-bit floating-point 3D textures to store the precalculated data. The previous generation of graphics cards could only store 16-bit floating-point values in the 3D textures and this gives noticeably poorer quality in the final rendering. Before rendering a new frame, we transfer the precalculated data from main memory to the graphics card.

1. The precalculation time is the number of seconds per frame and does not include the time for storage to hard disk. The data per frame includes the original velocity data, density information, and the precalculated data (dominant or rotation axis, case differentiation, and gradient measure). The average frame rate is for rendering a single frame continuously (no memory transfer from main memory to the graphics card) with the fixed view as shown in most pictures at a resolution of 800×600 pixels. The first rate is for the simple degenerate axis fix and the second one for the complex one. These reported frame rates are from an nVidia GTX 280 (1 GB VRAM), which was used in the final benchmarks.

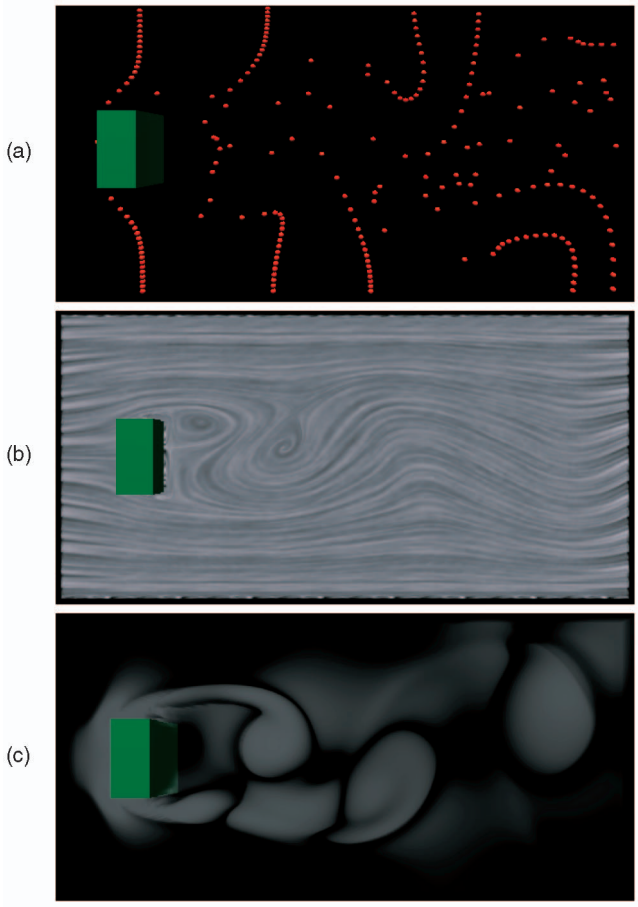

Fig. 6. Vortex street visualizations. (a) Passively advected particles. (b) Use of Line Integral Convolution to visualize the flow. (c) Variation of the opacity of the volume based on the magnitude of the velocity gradient.

The actual rendering is done with a large pixel shader that does ray tracing through the 3D volume textures. Before that, the opaque objects are rendered and the depth buffer is stored in a texture. The volume data are rendered by drawing the sides of a cube and the depth buffer texture is then used to calculate the actual distance a given ray travels through the volume before hitting an object or leaving the volume. The pixel shader then samples the volume textures at equidistant steps along the ray, calculates the rheoscopic reflection for each step, and blends these samples together according to their opacity.

The opacity is either constant over the whole volume or is based on the precalculated gradient magnitude. There is always a base reflection of transparent smoke for each step, onto which the rheoscopic reflection is added. This accounts for any unaligned particles in a cell and helps to give visual definition to the volume even if the rheoscopic particles would not reflect any light. The reflection can also be colored based on the density value to show different phases in the fluid. This technique is used in our internal splash sequences (Figs. 5, 7, and 8). Alternatively, colors could be utilized through multiple light sources of different color, as

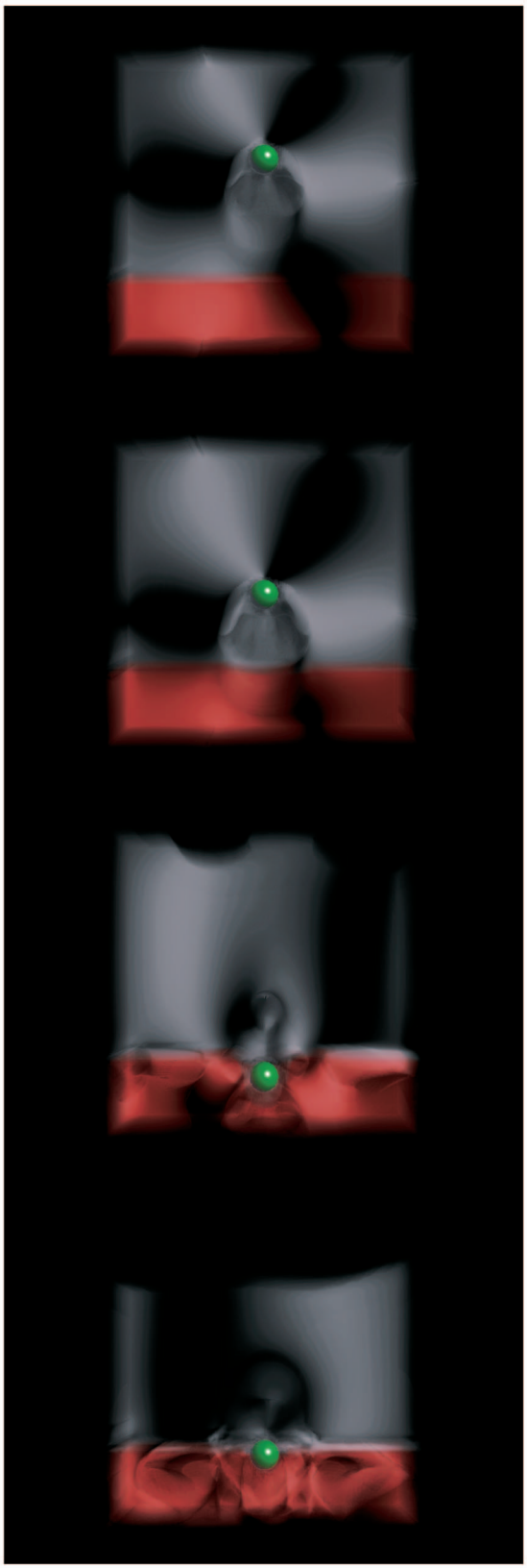

Fig. 7. Simulation of "internal splash." Note the upward plume of fluid that is evident in the last two frames.

introduced experimentally by Thoroddsen and Bauer [32] and used in the visualizations by Barth and Burns [2], by simply repeating the reflectance intensity calculation for each light source and mixing colors appropriately.

There is one issue that must be treated with care, regarding the trilinear sampling of 3D textures, namely, the $\pm$ arbitrariness of the dominant eigenvector, allowing changes in direction by 180 degrees from one cell to the next. To fix this problem, such changes in sign are detected, and then, a special case trilinear interpolation is performed. In this special case, the eight surrounding values are sampled from the texture and each of the dominant directions are flipped so that they point in the same general direction. Then, the regular trilinear interpolation is calculated. This fixes the issue, but requires eight additional samples from the volume texture for each sample that is degenerate. Because of the specific eigenvector calculation used, this case happens infrequently over the whole volume but is usually present somewhere in each frame. If not treated properly, this issue creates visible fissures through the volume. A cheaper way to reduce the issue is to sample at the center of a neighboring cell, when such a sign flip is detected, hence avoiding the trilinear interpolation. This can create slight voxelization artifacts in regions where the axis is changing its orientation rapidly, but requires only one additional sample from the volume texture and we often found it to be an acceptable compromise between image fidelity and rendering speed.

The rendering runs at more then 15 frames per second in views like the ones in the video (800×600 pixel resolution) using the simple flipped axis resolving mechanism. The exact speed depends on the number of samples along each ray and the number of pixels that need to be ray-traced, so it is view-dependent. If the complete sequence fits into main memory, it can be played and viewed interactively; but when the data have to be streamed from hard disk, it is not possible to maintain interactive processing. If the uniform rotation method is used in the preprocessing stage, the light source can be moved interactively as well, otherwise the light source is fixed since the precalculation is done using a particular light position. All pictures in this paper as well as the video have been created with this renderer using the precalculated nonuniform rotation method, except for Fig. 4 and the internal splash sequences where the camera or light source are moving, rendered with the uniform rotation method.

6 Results and Discussion

We have used the virtual rheoscopic method to visualize data from three different CFD simulations. We describe each of these results below.

Our first CFD simulation considered here is the classic von Kármán vortex street, as obtained from a 2D simulation. Fluid enters the simulation grid from the left and flows around an obstacle, in this case a cube. Behind the cube, vortices form and are eventually shed and move downstream toward the left. Fig. 6 shows three different visualizations of this simulation data. The top visualization shows massless particles that have been passively advected through the flow. It is hard to discern the vortices in this still image, but the vortices can be picked out in the video. The middle visualization is using Line Integral Convolution, where a gray-scale noise pattern has been smeared along the direction of flow. The bottom image shows this same fluid simulation using our gradient magnitude measure. The opacity of the cells in the voxel grid is given by the gradient magnitude. This visualization clearly shows the vortices in the flow, demonstrating the value of the gradient magnitude information. In this case, because of the 2D nature of the simulation, a model extension of the flow

TABLE 1
Statistics for the Different Sequences Shown in This Paper

Sequence	Grid	#Frames	Res. factor	Pre-calc. time	Data per frame	Avg. frame rate
Internal Splash	$120 \times 120 \times 30$	126	$1.5\times$	16 sec	34.0 MB	15 / 8 fps
Vortex Street	$80 \times 40 \times 20$	300	$2.0\times$	6 sec	10.8 MB	20 / 12 fps
Sedimentation (Figure 9)	$320 \times 120 \times 9$	126	$1.0\times$	4 sec	10.8 MB	50 / 26 fps

to 3D data with parabolic dependence visualized using our full rheoscopic technique (not shown) yielded images very similar to those using only gradient magnitude. In the fully 3D simulations of other phenomena considered below, the reflectances based on particle orientations capture more complicated details than gradient magnitude alone.

The next CFD simulation that we visualize is the so-called "internal splash" phenomena described by Abaid et al. [1]. When a heavier-than-fluid ball falls through the interface between light and heavy fluid, the ball may temporarily bounce upward. We simulated this phenomena using an Eulerian grid simulation of fluid, using Lagrangian constraints to cause a portion of the grid to mimic a solid object [5], [28]. Fig. 7 shows several frames from such a simulation, with virtual rheoscopic particles used to help visualize the fluid flow. In the lighter fluid, the particles were black-and-white, whereas the particles were colored red in the heavier fluid. In the two upper frames of the figure, the rheoscopic visualization indicates a region of fluid that the ball is pushing in front of itself as it falls. The complexity of the fluid motion when the green ball strikes the light/heavy fluid interface is evident in the lower two frames. Also, in these two frames, there is a visual indication of upward motion of fluid above the ball into the lighter fluid. This motion is even more evident in the accompanying video, as also are the complicated wave motions passing though the heavy fluid after the impact.

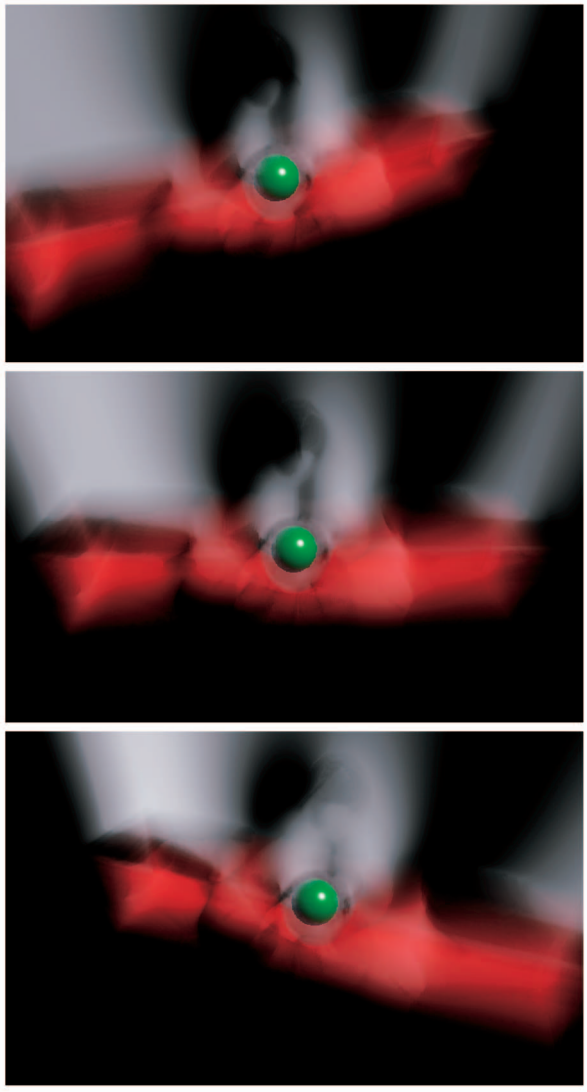

Fig. 8. Internal splash, varying the orientation of the volumetric grid.

Fig. 8 shows the 3D nature of the visualization technique. Just one time step is shown in each of these images, but the voxel grid is rotated to different positions in these images. Fig. 5 shows the effect of moving a virtual light from left to right for a single time step. The virtual rheoscopic particles reflect light differently depending on the position of the virtual light source.

The final CFD simulation that we present demonstrates an instability in particle sedimentation (simulated using dilute-limit interactions as in [27]). Many small, heavy spherical particles are suspended in fluid over a region of clear (particle-free) liquid at the start of the simulation. As the simulation progresses, the spherical particles drop into the clear liquid. The particles' motion is not uniform, and columns of downward moving particles form that are at least superficially similar to the fingering that is generated in a Raleigh-Taylor instability (though in this case, it is a miscible interface). Fig. 9 shows three different visualizations of this simulation data, with fluid velocities evaluated on a uniform grid (Table 1) from the specified particle forces, with centered-difference evaluation of the velocity gradients on the same grid. The top two images are an early and later time step from the simulation showing only particle positions. The motion of the particle-free fluid is not evident from this visualization.

Fig. 9b shows the same two time steps as in the first row, but uses our rheoscopic visualization method in addition to showing the positions of the falling particles. In this visualization, the amount of reflected light in the rheoscopic lighting model is attenuated by the gradient magnitude. In order to better show the rheoscopic lighting, only 1/10 of the original spherical particles are included in this animation. The positions of the omitted particles are indicated by red coloring of the voxels that contain or are very near the falling particles. In these middle frames, the rheoscopic lighting of the fluid gives a clear indication of those portions of the fluid that are in motion. Moreover, the dark and light channels that can be seen in the particle-filled regions indicate that some particle-laden regions are moving faster than others. The light regions on top of the particles indicate columns of particles that are moving upward. In the

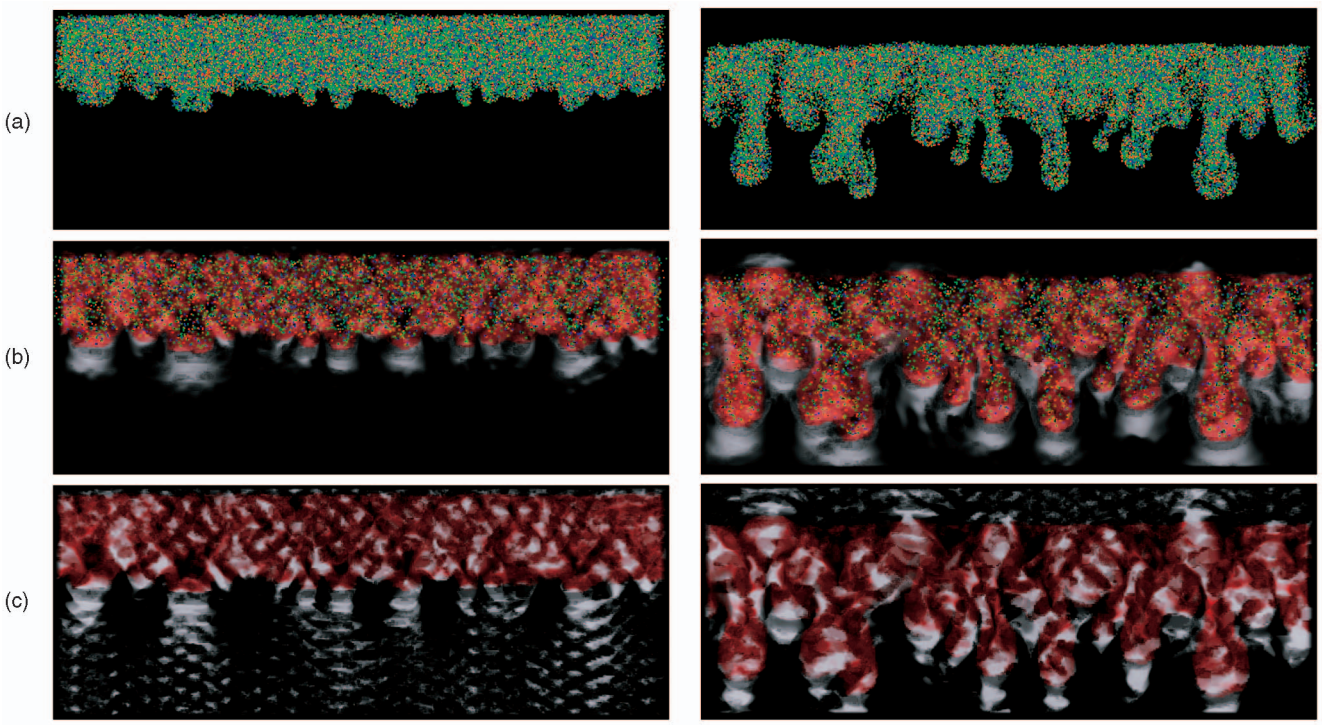

Fig. 9. Particle sedimentation simulation at two time steps. (a) Particles only. (b) Overlays attenuated rheoscopic lighting over the particles. (c) Unattenuated rheoscopic lighting model. The particle colors used in (a) and (b) are purely for ease of tracking individual particles in the video.

particle-only animation of this flow, this variable rate of particle motion can only be seen with difficulty. In the still frames with particles alone, this motion is not evident at all. A close-up of this simulation can also be seen in Fig. 1.

Fig. 9c shows the same simulation data, this time visualized using rheoscopic lighting alone. None of the falling particles are shown and the rheoscopic reflectance is not attenuated by the gradient magnitude. The only indication of the positions of the falling particles is the red color of the region near the particles. The most striking thing about these images are the repeated bands of light and dark both above and below the particles. It is evident from comparing this frame to the middle left frame that these bands diminish in strength as one moves further into the particle-free fluid domain. We were unaware of the presence of these repeated bands from our earlier particle-based visualizations of sedimentation. These bands, with separation comparable to the small out-of-plane dimension of the container (and thus, above the grid scale used for the gradient calculation), appear to correspond to counter-rotating regions which are weakly decaying in strength moving away from the particles. Such behavior is consistent both with the system-size correlations typical of the velocity fluctuation problem in creeping flow sedimentation and with the well-known presence of such counter-rotating flow in the particle-laden region; but only when we viewed this data using the rheoscopic lighting model did this aspect of the simulation data become clear away from the motion of the particles themselves. Also, visible in these two lower images are the slow and fast moving columns of particles, coded by dark and light intensities due to the variation in rheoscopic reflectances.

We have implemented the virtual rheoscopic visualization method of Barth and Burns [2], and include images based on their method for comparison to our own rheoscopic lighting model. Fig. 10 shows a Barth and Burns style of visualization of our particle sedimentation data, which can be compared to our results in Fig. 9. Figs. 11a and 11b show a comparison between our method and Barth and Burns for visualizing the internal splash data. In each case, the qualitative appearances of the data sets appear similar, but the actual features that are highlighted by the two methods differ. In many visualization systems, users are given a broad array of visualization styles to draw upon. Our own rheoscopic lighting model is a new additional visualization style and it provides an alternative to the method of Barth and Burns.

Flow visualization for 3D time-varying data is notoriously difficult. Some of the best methods for visualizing such data include placing particles, glyphs, or streamlines in the flow. Unfortunately, the use of too many such visualization primitives can produce cluttered and confusing images. Our virtual rheoscopic method offers quite a different form of visualization. Our method shows fine flow details, and the images that it produces are relatively uncluttered when compared with most other 3D flow visualization methods. As with any method, our approach has some limitations. Perhaps, the biggest limitation of our virtual rheoscopic method is that the flow direction at a given location cannot be determined from a single still

Fig. 10. Barth and Burns style visualization of the sedimentation data.

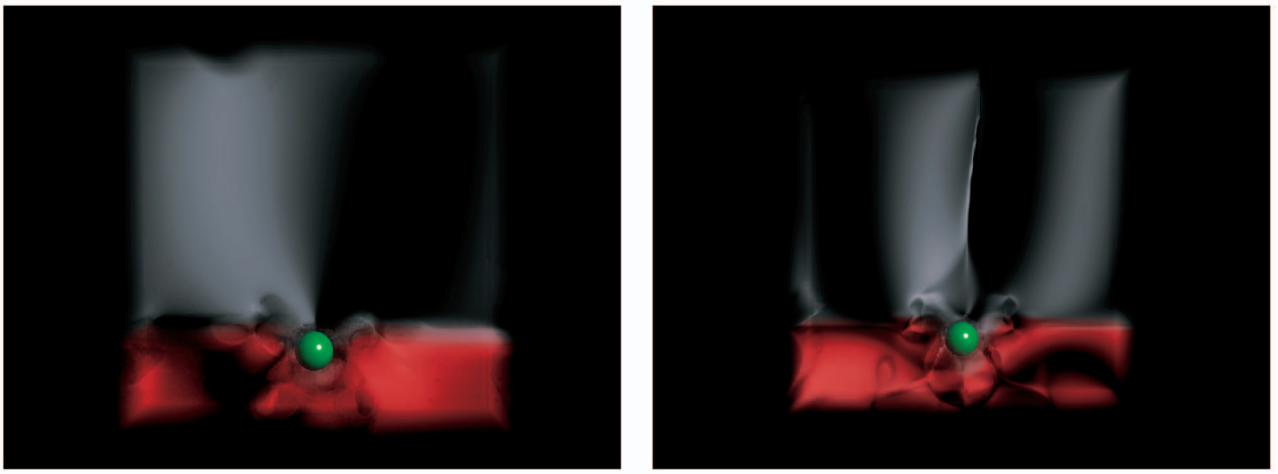

Fig. 11. Internal splash comparison between (a) our rheoscopic lighting model and (b) that of Barth and Burns.

image. When viewing a sequence of images that were created using our method, however, we find that the motion of the fluid is quite easy to see. This can be seen from the accompanying video sequences.

7 Conclusion

We have presented a method of visualizing 3D time-varying flow fields using a virtual rheoscopic lighting model. Inspired by a common fluid dynamics lab tool, our method mimics the reflectance of tiny particles that align themselves according to the velocity gradient of a fluid. Using this approach, we produce real-time visualizations of time-varying 3D flows that are rich in fine details.

There are a number of possible future directions for this work. One possibility is to use our virtual rheoscopic technique together with other flow visualization methods. For example, it may be beneficial to embed oriented glyphs in a virtual rheoscopic simulation. Such a visualization hybrid would show off both fine details (based on the virtual rheoscopic lighting) and also clearly show the direction flow at each glyph. It would also be beneficial to undertake a user's study to see how effective the virtual rheoscopic method is at conveying flow information when compared against other flow visualization techniques. Our current model treats the particles as too small to see except in their aggregate lighting effects. It would be interesting to modify the approach by creating larger visible particles that are oriented as ours, but that are also specifically advected through the flow. Finally, many of the most successful computer techniques for performing flow visualization have been inspired by methods used in real physical experiments. Perhaps, there are still other flow visualization techniques that are used in experimental fluid dynamics labs that can also inspire new computer-based visualization methods.

Acknowledgments

The authors thank Nipun Kwatra, Chris Wojtan, and Mark Carlson for the internal splash simulation code. This research was funded by US National Science Foundation (NSF) grants CCF-0625190 and CCF-0625264 and by a hardware donation from NVIDIA.

References

[1] N. Abaid, D. Adalsteinsson, A. Agyapong, and R.M. McLaughlin, "An Internal Splash: Levitation of Falling Spheres in Stratified Fluids," *Physics of Fluids,* vol. 16, no. 5, pp. 1567-1580, May 2004.

[2] W.L. Barth and C.A. Burns, "Virtual Rheoscopic Fluids for Flow Visualization," *IEEE Trans. Visualization and Computer Graphics,* vol. 13, no. 6, pp. 1751-1758, Nov./Dec. 2007.

[3] F.P. Bretherton, "The Motion of Rigid Particles in a Shear Flow at Low Reynolds Number," *J. Fluid Mechanics,* vol. 14, pp. 284-304, 1962.

[4] B. Cabral and L. Leedom, "Imaging Vector Fields Using Line Integral Convolution," *Proc. ACM SIGGRAPH '93,* pp. 263-270, Aug. 1993.

[5] M. Carlson, P.J. Mucha, and G. Turk, "Rigid Fluid: Animating the Interplay between Rigid Bodies and Fluid," *Proc. ACM SIGGRAPH '04,* pp. 377-384, Aug. 2004.

[6] R. Crawfis and N. Max, "Direct Volume Visualization of Three Dimensional Vector Fields," *Proc. Workshop Volume Visualization,* pp. 55-60, 1992.

[7] T. Delmarcelle and L. Hesselink, "Visualization of Second Order Tensor Fields and Matrix Data," *Proc. IEEE Conf. Visualization,* pp. 316-323, Oct. 1992.

[8] T. Delmarcelle and L. Hesselink, "The Topology of Symmetric, Second-Order Tensor Fields," *Proc. IEEE Conf. Visualization,* pp. 140-147, Oct. 1994.

[9] D. Dovey, "Vector Plots for Irregular Grids," *Proc. IEEE Conf. Visualization,* pp. 248-253, Oct./Nov. 1995.

[10] G. Gauthier, P. Gondret, and M. Rabaud, "Motions of Anisotropic Particles: Application to Visualization of Three-Dimensional Flows," *Physics of Fluids,* vol. 10, pp. 2147-2154, Sept. 1998.

[11] I. Grant, "Particle Image Velocimetry: A Review," *Proc. Inst. of Mechanical Engineers, Part C,* vol. 211, no. 1, pp. 55-76, 1997.

[12] I. Hotz, L. Feng, H. Hagen, B. Hamann, K. Joy, and B. Jeremic, "Physically Based Methods for Tensor Field Visualization," *Proc. IEEE Conf. Visualization,* pp. 123-130, Oct. 2004.

[13] T.J. Jankun-Kelly and K. Mehta, "Superellipsoid-Based Real Symmetric Traceless Tensor Glyphs Motivated by Nematic Liquid Crystal Alignment Visualization," *Proc. IEEE Conf. Visualization,* pp. 1197-1204, Oct./Nov. 2006.

[14] G.B. Jeffery, "The Motion of Ellipsoidal Particles Immersed in a Viscous Fluid," *Proc. Royal Soc. of London, Series A,* vol. 102, pp. 161-179, 1922.

[15] B. Jobard and W. Lefer, "Creating Evenly-Spaced Streamlines of Arbitrary Density," *Proc. Eurographics Workshop Visualization in Scientific Computing,* pp. 43-55, Oct. 1998.

[16] J.T. Kajiya and T.L. Kay, "Rendering Fur with Three Dimensional Textures," *Proc. ACM SIGGRAPH '89,* pp. 271-280, July/Aug. 1989.

[17] G. Kindlmann and D. Weinstein, "Hue-Balls and Lit-Tensors for Direct Volume Rendering of Diffusion Tensor Fields," *Proc. IEEE Conf. Visualization,* pp. 183-189, Oct. 1999.

[18] G. Kindlmann, D. Weinstein, and D. Hart, "Strategies for Direct Volume Rendering of Diffusion Tensor Fields," *IEEE Trans. Visualization and Computer Graphics,* vol. 6, no. 2, pp. 124-138, Apr.-June 2000.
[19] G. Kindlmann, D. Weinstein, and D. Hart, "Strategies for Direct Volume Rendering of Diffusion Tensor Fields," *Proc. IEEE Conf. Visualization,* pp. 1329-1335, Oct./Nov. 2006.
[20] D.H. Laidlaw, E.T. Ahrens, D. Kremers, M.J. Avalos, R.E. Jacobs, and C. Readhead, "Visualizing Diffusion Tensor Images of the Mouse Spinal Cord," *Proc. IEEE Conf. Visualization,* pp. 127-134, Oct. 1998.
[21] R.S. Laramee, B. Jobard, and H. Hauser, "Image Space Based Visualization of Unsteady Flow on Surfaces," *Proc. IEEE Conf. Visualization,* pp. 131-138, Oct. 2003.
[22] R.S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F.H. Post, and D. Weiskopf, "The State of the Art in Flow Visualization: Dense and Texture-Based Techniques," *Computer Graphics Forum,* vol. 23, no. 2, pp. 203-221, 2004.
[23] Z. Liu, R.J. Moorhead, and J. Groner, "An Advanced Evenly-Spaced Streamline Placement Algorithm," *Proc. IEEE Conf. Visualization,* pp. 965-972, Oct./Nov. 2006.
[24] P. Matisse and M. Gorman, "Neutrally Buoyant Anisotropic Particles for Flow Visualization," *Physics of Fluids,* vol. 27, no. 4, pp. 759-760, Apr. 1984.
[25] O. Mattausch, T. Theußl, H. Hauser, and E. Gröller, "Strategies for Interactive Exploration of 3D Flow Using Evenly-Spaced Illuminated Streamlines," *Proc. Spring Conf. Computer Graphics,* pp. 213-222, Apr. 2003.
[26] A. Mebarki, P. Alliez, and O. Devillers, "Farthest Point Seeding for Efficient Placement of Streamlines," *Proc. IEEE Conf. Visualization,* pp. 479-486, Oct. 2005.
[27] P.J. Mucha, S.-Y. Tee, D.A. Weitz, B.I. Shraiman, and M.P. Brenner, "A Model for Velocity Fluctuations in Sedimentation," *J. Fluid Mechanics,* vol. 501, pp. 71-104, 2004.
[28] N.A. Patankar, P. Singh, D.D. Joseph, R. Glowinski, and T.-W. Pan, "A New Formulation of the Distributed Lagrange Multiplier/Fictitious Domain Method for Particulate Flows," *Int'l J. Multiphase Flow,* vol. 26, no. 9, pp. 1509-1524, 2000.
[29] T. Preußer and M. Rumpf, "Anisotropic Nonlinear Diffusion in Flow Visualization," *Proc. IEEE Conf. Visualization,* pp. 325-332, Oct. 1999.
[30] A.R. Sanderson, C.R. Johnson, and R.M. Kirby, "Display of Vector Fields Using a Reaction-Diffusion Model, Visualization Techniques," *Proc. IEEE Conf. Visualization,* pp. 115-122, Oct. 2004.
[31] S. Ömer, "On Flow Visualization Using Reflective Flakes," *J. Fluid Mechanics,* vol. 152, pp. 235-248, 1985.
[32] S.T. Thoroddsen and J.M. Bauer, "Qualitative Flow Visualization Using Colored Lights and Reflective Flakes," *Physics of Fluids,* vol. 11, pp. 1702-1705, 1999.
[33] X. Tricoche, G. Scheuermann, and H. Hagen, "Tensor Topology Tracking: A Visualization Method for Time-Dependent 2D Symmetric Tensor Fields," *Computer Graphics Forum,* vol. 20, no. 3, pp. 461-470, 2001.
[34] G. Turk and D. Banks, "Image-Guided Streamline Placement," *Proc. ACM SIGGRAPH '96,* pp. 453-460, Aug. 1996.
[35] J.J. van Wijk, "Spot Noise: Texture Synthesis for Data Visualization," *Proc. ACM SIGGRAPH '91,* pp. 309-318, July 1991.
[36] J.J. van Wijk, "Image Based Flow Visualization," *Proc. ACM SIGGRAPH '02,* pp. 745-754, July 2002.
[37] J.J. van Wijk, "Image Based Flow Visualization for Curved Surface," *Proc. IEEE Conf. Visualization '03,* pp. 123-130, Oct. 2003.
[38] V. Verma, D.L. Kao, and Alex Pang, "A Flow-Guided Streamline Seeding Strategy," *Proc. IEEE Conf. Visualization,* pp. 415-422, Oct. 2000.
[39] D. Weinstein, G. Kindlmann, and E. Lundberg, "Tensorlines: Advection-Diffusion Based Propagation through Diffusion Tensor Fields," *Proc. IEEE Conf. Visualization,* pp. 249-253, Oct. 1999.
[40] X. Ye, D.L. Kao, and A. Pang, "Strategy for Seeding 3D Streamlines," *Proc. IEEE Conf. Visualization,* pp. 471-478, Oct. 2005.
[41] X. Zheng and A. Pang, "HyperLIC," *Proc. IEEE Conf. Visualization 2003,* pp. 249-256, Oct. 2003.
[42] X. Zheng, B. Parlett, and A. Pang, "Topological Structures of 3D Tensor Fields," *Proc. IEEE Conf. Visualization,* pp. 551-558, Oct. 2005.
[43] L. Zhukov and A. Barr, "Oriented Tensor Reconstruction: Tracing Neural Pathways from Diffusion Tensor MRI," *Proc. IEEE Conf. Visualization,* pp. 387-394, Oct./Nov. 2002.

Florian Hecht received the Diplom in computer science from the University of Karlsruhe, Germany, in 2008, and the MS degree in computer science from the Georgia Institute of Technology in 2007. He began work on his PhD degree in Fall 2009 at the University of California, Berkeley. He is interested in computer graphics, robotics, and computer vision.

Peter J. Mucha received the PhD degree in applied and computational mathematics from Princeton University in 1998. He was an applied mathematics instructor at the Massachussetts Institute of Technology (MIT) for three years, followed by four years as an assistant professor in mathematics at Georgia Institute of Technology. He is currently an associate professor at the University of North Carolina at Chapel Hill, where he is a member of the Department of Mathematics and the Institute for Advanced Materials, Nanoscience & Technology.

Greg Turk received the PhD degree in computer science from the University of North Carolina at Chapel Hill (UNC) in 1992. He was a postdoctoral researcher at Stanford University for two years, followed by two years as a research scientist at UNC Chapel Hill. He is currently a professor at the Georgia Institute of Technology, where he is a member of the School of Interactive Computing and the Graphics, Visualization and Usability Center. His research interests include computer graphics, scientific visualization, and computer vision. In 2008, he was the technical papers chair for ACM SIGGRAPH. He is a member of the IEEE.