# Smoke Surfaces: An Interactive Flow Visualization Technique Inspired by Real-World Flow Experiments

Wolfram von Funck, Tino Weinkauf, Holger Theisel, and Hans-Peter Seidel

**Abstract**—Smoke rendering is a standard technique for flow visualization. Most approaches are based on a volumetric, particle based, or image based representation of the smoke. This paper introduces an alternative representation of smoke structures: as semi-transparent streak surfaces. In order to make streak surface integration fast enough for interactive applications, we avoid expensive adaptive retriangulations by coupling the opacity of the triangles to their shapes. This way, the surface shows a smoke-like look even in rather turbulent areas. Furthermore, we show modifications of the approach to mimic smoke nozzles, wool tufts, and time surfaces. The technique is applied to a number of test data sets.

**Index Terms**—Unsteady flow visualization, streak surfaces, smoke visualization.

## 1 Introduction

Flow visualization is an active field of research. A variety of techniques for the interactive exploration of flow phenomena has been developed. One approach is to resemble well-accepted techniques from experimental flow visualization. Among them, smoke visualization plays an important role: in a real flow of gases, smoke is advected and its temporal behavior gives information about the flow. Often, the smoke is advected from line structures (e.g., from a burning stick), or it is inserted from certain points (smoke nozzles). Another common technique in experimental flow visualization are wool tufts where small yarns are attached to a body and observed during the flow experiment.

To use smoke in (computer aided) flow visualization, it is represented either in a volumetric, a particle based, or an image based way. All these approaches have proven to be useful. In a volumetric approach, the smoke is represented as a density scalar field, making a high resolution or an adaptive grid necessary to capture certain details. Particle based approaches usually need a high number of particles to represent smoke.

In this paper we propose an alternative representation of smoke: a semi-transparent streak surface. This approach is inspired by artistic smoke photographs as shown in figure 1. In this image, smoke was advected from a stick, i.e., a line-like seeding structure. Together with appropriate lighting conditions, expressive and aesthetic photographs of real-world smoke were obtained. In figure 1 the smoke clearly forms a semi-transparent surface structure which can be interpreted as a streak surface. Hence, figure 1 makes us believe that semi-transparent streak surfaces can give expressive visual representations of a flow if we follow the smoke metaphor, i.e., if the surface is rendered to look like smoke.

Stream surface integration is a standard approach in flow visualization. However, the integration of *streak* surfaces in time-dependent flows is fundamentally different, because it requires an adaptive remeshing of the *complete* surface at every time step. Up to now, this prevented streak surfaces from being used in interactive applications.

The main idea of this paper is to use streak surfaces without any adaptive remeshing, i.e., the triangular mesh representing the streak surface has a fixed topology and connectivity. This will lead to large and non-regular triangles e.g. due to diverging flow, but these areas become less visible because of the optical model for smoke that we apply for the rendering. This way, we combine two advantages: the surface looks like a smoke structure, and no time is spent for surface remeshing.

The smoke surface technique obtained this way can be enhanced in several ways. By coloring the mesh vertices, we can visualize time lines and streak lines within the streak surface. The seeding can also be done starting from all vertices of a surface at the same time, leading to semi-transparent time surfaces. Smoke nozzles can be simulated by setting certain parts of the streak surface invisible. Finally, wool tufts can be mimicked by seeding short and narrow streak surfaces close to the obstacles in the flow.

The rest of the paper is organized as follows: section 2 reviews related work. Section 3 describes our approach in detail. Section 4 describes modifications and enhancements. Section 5 applies the approach to a number of test data sets. Section 6 evaluates our approach while conclusions are drawn in section 7.

## 2 Related Work

Smoke is a well-researched subject in both computer graphics and visualization. While several offline techniques for the realistic rendering of smoke exist [16, 15], we restrict the overview here to realtime smoke rendering methods.

Probably the most often used approaches for realtime smoke rendering are based on particles. Here, a (usually large) set of particles is seeded into a flow and advected over time. Surface-particles [34] are represented as points with normals and can be rendered efficiently to visualize flow. Another early method to render particle-based volumetric data is texture splatting [7]. In recent years, point sprites [18] and semi-transparent textured billboards with opacity values proportional to the density value of the particles [12] have become a popular method for particle rendering. In order to reduce artifacts for highly stretched flows, blob particles [27, 1] can be used. They are basically ellipsoid particles that can be stretched and split during animation. By considering the spherical geometry of the particles during fragment processing, spherical billboards [31] eliminate clipping and popping artifacts which occur when the particles flow around other objects in the scene. By using a variant of the depth difference technique, non-photorealistic cartoon rendering of smoke particles is possible [24].

With the increasing power of graphics hardware, realtime volumetric rendering methods of gaseous phenomena are becoming more and more popular. Animated clouds can be rendered realistically using a slice-based volumetric rendering scheme [23]. By shifting complex computations to the GPU, realtime performance is obtained. By utilizing the latest features of current graphics hardware (rendering to 3D textures, geometry shader, stream out), real-time volumetric smoke, fire and water with fluid dynamics have recently been made possible

- *Wolfram von Funck and Hans-Peter Seidel are with MPI Informatik Saarbrücken, E-mail: {wfunck,hpseidel}@mpi-inf.mpg.de.*
- *Tino Weinkauf is with Zuse Institute Berlin, E-mail: weinkauf@zib.de.*
- *Holger Theisel is with University of Magdeburg, E-mail: theisel@isg.cs.uni-magdeburg.de.*

*Manuscript received 31 March 2008; accepted 1 August 2008; posted online 19 October 2008; mailed on 13 October 2008.*
*For information on obtaining reprints of this article, please send e-mailto:tvcg@computer.org.*

Figure 1. Artistic photographs of real smoke. From [3].

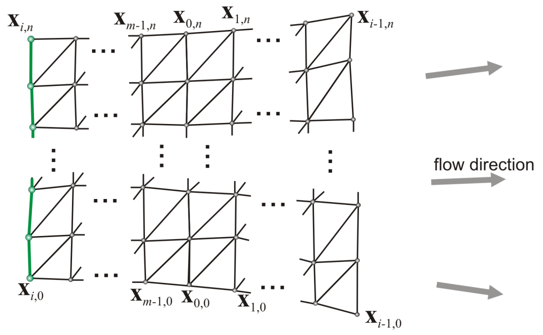


Figure 2. Streak surface at time $t_0 + i \cdot \Delta t$.

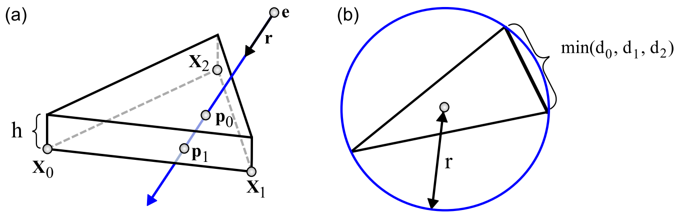


Figure 3. Configurations for computing (a) $\alpha_{density}$ and (b) $\alpha_{shape}$.

[28, 6]. Here, the actual rendering is performed via ray casting on the GPU. Using compensated ray marching [39], it is possible to incorporate dynamic environment lighting into interactive smoke rendering.

In computer graphics, meshes have been used in the context of cloud rendering [9, 30, 2].

In flow visualization, a number of approaches has been developed to create smoke-like images. Flow volumes [20] were introduced as the volumetric counterpart to stream lines. Using volumetric meshes composed of transparently rendered tetrahedra, the technique allows for interactive exploration of vector fields. Particle based methods [17, 4] use a large number of particles to get a smoke impression. Image based smoke visualizations are mentioned in [35]. As an alternative to smoke, dye has been proposed to visualize vector fields [37]. Virtual tufts for flow visualization are described in [25].

**Stream surfaces vs. streak surfaces**
*Stream surfaces* have been extensively used in flow visualization [13, 22, 33, 10, 21]. Starting from a polygonal seeding structure, the surface front line is integrated and adaptively modified. This modification is necessary for two reasons: firstly, converging or diverging properties of the flow may lead to a front line with a too high or too low density of the sample points. In this case, vertices are collapsed, or new vertices are introduced. Secondly, the surface may split into different parts, for example when it flows around different sides of an obstacle. In this case the front has to be split into two parts which are further traced independently. For time-dependent flows, the integration of stream and path surfaces works similar to the steady case.

*Streak surfaces* in time-dependent flows are obtained by repeatedly setting out, tracing, and connecting particles from a seeding curve. There is a fundamental difference to stream surfaces: for streak surfaces, *all* locations of the surface are updated at every time step – and not only a front line. Therefore, the complete surface has to be checked for vertex collapse/split and for surface cut after every advection step. Up to now, this prevented streak surfaces from being used in interactive applications. In fact, we are not aware of any approaches to interactively visualize streak surfaces in unsteady flows.

## 3 Approach

The main idea of our approach is to represent smoke as a triangular mesh of a fixed topology and connectivity. We use an $(m+1)\times(n+1)$ vertex array ($\mathbf{x}_{i,j}$ $i = 0,...,m; j = 0,...,n$) defining a closed surface of cylinder topology, i.e., we assume $\mathbf{x}_{0,j} = \mathbf{x}_{m,j}$ for $j = 0,..,n$. We call the polygon $(\mathbf{x}_{i,0},..,\mathbf{x}_{i,n})$ the $i$-th column, while the polygon $(\mathbf{x}_{0,j},..,\mathbf{x}_{m,j})$ is the $j$-th row. As seeding structure we use a polygon $(\mathbf{s}_0,...,\mathbf{s}_n)$.

For initialization, all vertices are set to the seeding structure, i.e., $\mathbf{x}_{i,j} = \mathbf{s}_j$. Starting at $t_0$, columns of the array are successively released into the flow. The integration of the $i$-th column starts at time $t_0 + i \cdot \Delta t$ for $i = 0,...,m-1$. Note that once a column is released into the flow, it has to be advected in every time step – this is in contrast to stream surfaces, where only the currently last column is integrated. After $m$ time steps, all columns have been consumed at the seeding curve and the complete surface is unfolded into the flow. The seeding continues with the first column, i.e., the vertices of the currently "oldest" column (longest time in the flow) are reset to the seeding structure. This way, a continuous seeding of the streak surface over an unlimited time is possible. Re-allocation is not necessary during the integration: vertices at the end of the streak surface are re-used at the start of it. Figure 2 gives an illustration of the streak surface at time $t_0 + i \cdot \Delta t$.

Note that our system allows to change the location of the seeding polygon interactively. Certain optional parts of the opacity computation (explained in the following section) may make it desirable to choose the step width $\Delta t$ such that the triangles are approximately equilateral shortly after their advection started.

Representing a streak surface with a fixed resolution and connectivity seems to be unsuitable since adaptive schemes have already proven their usefulness for much simpler flow features such as stream surfaces. The fixed resolution will lead to situations where e.g. triangles become rather large due to diverging flow behavior. However, the main goal of this paper is not to extract perfect streak surfaces, but to render smoke based on such surfaces. As we will see in the following section, the optical model for smoke already gives that smoke becomes less visible in areas with diverging behavior. In other words, larger triangles are less visible due to the smoke metaphor and therefore the advantages of the fixed resolution (mainly interactivity and ease of implementation) outweigh its shortcomings in our case of smoke rendering.

### 3.1 Opacity Computation

#### 3.1.1 Optical Model of Smoke

To represent the density of smoke, we assume a triangle $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2$ to have a certain small height $h$, i.e., we consider it to be a rather flat prism evenly filled with smoke. Figure 3a illustrates the configuration. We assume the viewing ray $\mathbf{p}(t) = \mathbf{e} + t\mathbf{r}$ with $\|\mathbf{r}\| = 1$ entering the prism in the point $\mathbf{p}_0 = \mathbf{e} + t_0\mathbf{r}$ and leaving in $\mathbf{p}_1 = \mathbf{e} + t_1\mathbf{r}$. Then the $\alpha$ value describing the absorption is [19]

$$\alpha = 1 - e^{-\int_{t_0}^{t_1} \tau(t)dt} \tag{1}$$

where $\tau(t) = \tau(\mathbf{p}(t))$ is the extinction coefficient at the location $\mathbf{p}(t)$ describing the rate that light is occluded. Since we assume $\tau$ to be constant inside the prism, (1) simplifies to

$$\alpha = 1 - e^{-(t_1 - t_0)\tau}. \tag{2}$$

Let $\gamma$ be the angle between $\mathbf{r}$ and the normal $\mathbf{n}$ of the triangle. Then $(t_1 - t_0) = \frac{h}{\cos\gamma}$ under the assumption that the viewing ray intersects only the spanning triangles of the prism. This gives

$$\alpha(h) = 1 - e^{-\frac{h\,\tau}{\cos\gamma}}. \tag{3}$$

Since $h$ is assumed to be rather small, $\alpha(h)$ can be linearized by a Taylor expansion to

$$\alpha(h) = \alpha(0) + h\,\frac{d\alpha}{dh}(0) = \frac{h\,\tau}{\cos\gamma}. \tag{4}$$

Assuming a particle model for the smoke (i.e., the smoke consists of a number of small absorbing particles, $\tau$ linearly depends on the particle density inside the prism: $\tau = c\,\frac{n_p}{\text{area}(\mathbf{x}_0,\mathbf{x}_1,\mathbf{x}_2)}$ where $c$ is a certain constant and $n_p$ is the number of particles within the prism. This yields

$$\alpha_{density} = \frac{k}{\text{area}(\mathbf{x}_0,\mathbf{x}_1,\mathbf{x}_2)\;\cos\gamma} \tag{5}$$

which describes the $\alpha$ value representing the smoke density inside the prism. The constant $k = h\,c\,n_p$ steers the initial density at seeding time. Note that due to the linearization, $\alpha_{density}$ can be outside the interval $[0,1]$; in this case, it has to be clamped to $[0,1]$.

The physically motivated $\alpha_{density}$ has been used for all smoke visualizations throughout this paper and it steers the main visual appearance of all these images. However, in a few situations $\alpha_{density}$ does not suffice to compensate for the fixed resolution and connectivity of our mesh. Usually these are small areas where the surface flows around some obstacle or becomes too distorted. The perfect solution in these cases would be to increase the resolution, but this also implies to reduce the speed and responsiveness of the application. If interactive frame rates are desired, we have to trade accuracy for speed eventually. In the following section we have identified some situations leading to visual clutter due to a locally too coarse mesh resolution and propose solutions.

### 3.1.2 Optional Opacity Parameters

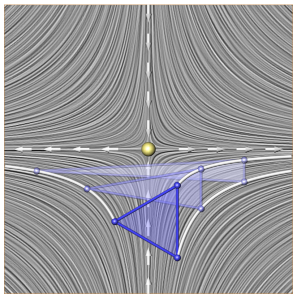
In order to detect cuts or other areas where the triangulation does not describe the smoke surface well,[1] we consider a measure of the local quality of the mesh triangles. Note that the area of a triangle does not suffice to detect surface cuts. Figure 4a shows an example. There, we have a simple linear vector field containing a saddle point toward which a triangle is integrated while a separation takes place: one vertex moves to the left-hand side, while two go to the right-hand side. This is a typical configuration for a cut surface ending in a long thin triangle which should be set invisible. Note that the area of the triangle is almost constant during the integration (see the dotted line in figure 4b).

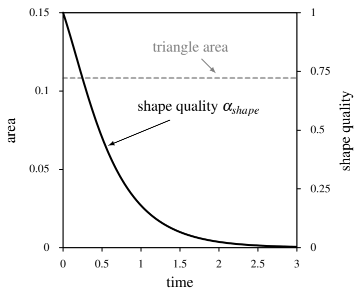
A well-accepted measure of the shape quality of a triangle $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2$ is the ratio of the shortest edge length to the radius $r$ of the circumcircle [26]. The solid line in figure 4b shows the behavior of this measure in the configuration of figure 4a. Since $r = \frac{d_0 d_1 d_2}{2\,\text{area}(\mathbf{x}_0,\mathbf{x}_1,\mathbf{x}_2)}$ with $d_0 = \|\mathbf{x}_2 - \mathbf{x}_1\|$, $d_1 = \|\mathbf{x}_0 - \mathbf{x}_2\|$, $d_2 = \|\mathbf{x}_1 - \mathbf{x}_0\|$, we use this to define the shape quality parameter as

$$\alpha_{shape} = \left( \frac{4\,\text{area}(\mathbf{x}_0,\mathbf{x}_1,\mathbf{x}_2)}{\sqrt{3}\,\max\{d_0 d_1\;,\;d_1 d_2\;,\;d_2 d_0\}} \right)^s. \tag{6}$$

Note that $\alpha_{shape} = 1$ for an equilateral triangle, and that $\alpha_{shape}$ gets smaller the "less equilateral" the triangle is. The positive constant $s$ steers how strong the influence of $\alpha_{shape}$ is relative to $\alpha_{density}$: the

[1]Such unsuitable triangles are also the ones connecting the last column of the streak surface with the (freshly reset) first column.


(a) Integrated triangle. Shown are three instances.



(b) Triangle area and shape quality measure plotted over integration time.

Figure 4. Distortion of a equilateral triangle integrated towards the saddle point in the linear vector field $\mathbf{v} = (x, -y)^T$.

smaller $s$ is, the less influential is $\alpha_{shape}$. Throughout this paper, $s$ was chosen between 0.5 and 1.

In regions of high Mean curvature of the streak surface, the non-adaptive mesh may not be an appropriate representation. In order to make the surface less visible in this case, we introduce $\alpha_{curvature}$ reflecting the curvature. For a vertex $\mathbf{x}_0$, we compute this as

$$\alpha_{curvature} = 1 - b \cdot \max\{|\mathbf{n}_0\,\mathbf{e}_i| : i = 1..valence(\mathbf{x}_0)\} \tag{7}$$

where $\mathbf{n}_0$ is the estimated surface normal at $\mathbf{x}_0$ , $i$ iterates over all vertices $\mathbf{x}_i$ in the 1-ring of $\mathbf{x}_0$, and $\mathbf{e}_i = \frac{\mathbf{x}_i - \mathbf{x}_0}{\|\mathbf{x}_i - \mathbf{x}_0\|}$. If all vertices of the 1-ring of $\mathbf{x}_0$ are approximately in the tangent plane of $\mathbf{x}_0$, $\alpha_{curvature}$ is close to 1. The positive constant $b$ determines how strong a large surface curvature influences $\alpha_{curvature}$. Throughout the paper we have chosen $b = 2$ and clamp $\alpha_{curvature}$ to the interval $[0,1]$.

Smoke tends to disperse and fade over time. To simulate this behavior, we introduce an additional alpha value $\alpha_{fade}$. Given the age $t$ of a vertex, which is the time passed after the vertex was seeded, and a maximum age $t_{max}$, which is the age when a vertex should become invisible, we can compute $\alpha_{fade}$ as

$$\alpha_{fade} = 1 - \frac{t}{t_{max}}. \tag{8}$$

We choose $t_{max}$ as the maximum integration time of a vertex before it is seeded again, i.e., $t_{max} = m\Delta t$.

The final $\alpha$ value including all opacity parameters is computed such that it is not larger than its smallest component:

$$\alpha = \alpha_{density}\;\alpha_{shape}\;\alpha_{curvature}\;\alpha_{fade}. \tag{9}$$

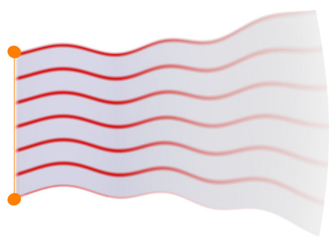
In practice, we define $\alpha$ per-vertex such that we get a piecewise linear interpolation across the surface. To achieve this, we set $\alpha_{density}$ and $\alpha_{shape}$ of a vertex to the minimum value of its adjacent triangles; $\alpha_{curvature}$ and $\alpha_{fade}$ are already defined per-vertex. $\alpha_{density}$, $\alpha_{shape}$, $\alpha_{curvature}$ and $\alpha_{fade}$ have to be clamped to the interval $[0,1]$ before applying (9).

The final $\alpha$ is steered by three degrees of freedom: $k$ for the initial smoke density, $s$ for the influence of the shape parameter, and $b$ for the influence of the curvature.

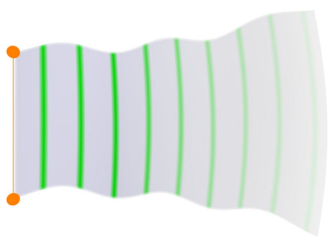
Remark: Formulae (5) and (6) seem to suggest to compute $\alpha_{density}\,\alpha_{shape}$ directly and not separately because area$(\mathbf{x}_0,\mathbf{x}_1,\mathbf{x}_2)$ cancels out. This fails because $\alpha_{density}$ and $\alpha_{shape}$ have to be clamped separately. Only clamping $(\alpha_{density}\,\alpha_{shape})$ for a triangle with small $\gamma$ (i.e., close to a silhouette) can make it visible even though it has a bad shape quality.

## 3.2 Implementation

In our OpenGL implementation, we used *depth peeling* [8] in order to avoid depth-sorting of triangles, which would be computationally expensive and can produce artifacts at overlapping triangles. Depth



(a) Coloring fixed rows in the array reveals streak lines.



(b) Coloring fixed columns in the array reveals time lines.

Figure 5. Smoke surface in a simple vector field which is constantly seeded at the left seeding line. Streak and time lines can easily be shown on the smoke surface using a simple preprocessing step and at no cost for the rendering performance.

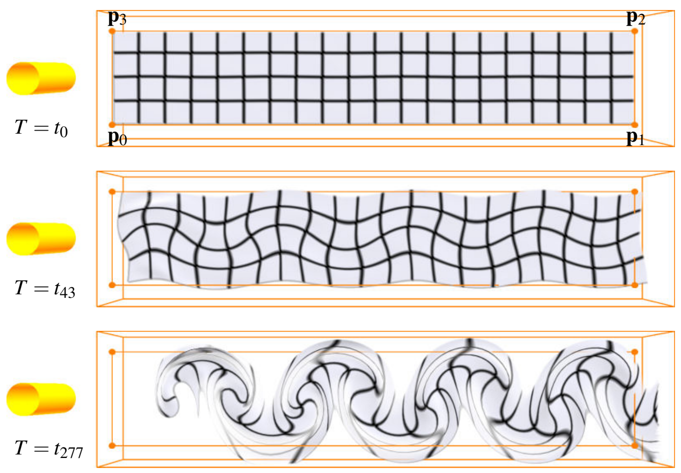


Figure 6. Time surface spanned by the points $\mathbf{p}_0, \ldots, \mathbf{p}_3$ and transported by the flow behind a circular cylinder. A uniform grid becomes visible by coloring some columns and rows of the internal array differently. After some integration steps, this elucidates the distortion introduced by the flow.

peeling basically works by rendering fragments into different layers which are superimposed afterwards to get the final image. We used four layers in our implementation, which seems to be sufficient for smoke rendering. A big advantage of this method is the fact that we can render the surface as a single triangle strip without changing connectivity.

## 4 Enhancements and Modifications

Our approach to rendering smoke surfaces updates the set of vertices by means of advection and computes the alpha values in each frame. Other parts like the connectivity of the surface or rgb-color values remain constant. In this section we propose a number of simple modifications to these constant parts that allow us to achieve a variety of different visualization styles. In particular, we are able to depict streak and time lines, reproduce the visual appearance of smoke nozzles, integrate time surfaces, and mimic an important real-world visualization technique called wool tufts. All these modifications are done in a rather simple preprocessing step. Therefore, they do not influence the rendering performance of our technique.

### 4.1 Streak and Time Lines

A *streak line* is the connection of all particles set out at different times but the same point location. In an experiment, one can observe these structures by constantly releasing dye into the flow from a fixed position. The resulting streak line consists of all particles which have been at this fixed position sometime in the past. Such lines can also be found on our smoke surface under the assumption that the seeding curve remains constant. In this case, the vertices representing a streak line are found in a fixed row of the vertex array: these vertices have been seeded at the same position in space but at different times. Hence, we can depict a streak line by assigning a different color to a row of the array. Figure 5a illustrates this.

A *time line* is the connection of all particles set out at the same time but different locations, i.e., a line which gets advected by the flow. An analogon in the real world is a yarn or wire thrown into a river, which gets transported and deformed by the flow. However, in contrast to the yarn, a time line can get shorter and longer. In our implementation, the vertices representing a time line are found in a fixed column of the vertex array: these vertices have been seeded at the same time along the seeding curve. Hence, we can depict a time line by assigning a different color to a column of the array. Figure 5b illustrates this.

### 4.2 Time Surfaces

Up to now, we seeded particles continuously at a curve – thereby creating a streak surface. We may as well spread out the complete surface in the volume and start the integration of all vertices at once. The surface gets advected and distorted by the flow. In fact, this is a time surface, since all vertices have been seeded at the same time but at different locations. One may think of this as a carpet thrown into a river and transported by the flow.

A problem with this approach is that the surface may rather quickly leave the domain or the visualized smoke simply dissolves after some time. Hence, we have to come up with a scheme for re-injecting smoke. In our implementation, we allow the user to have a small number of time surfaces that can be started and reset interactively.

Time surfaces can aid in understanding the distortion introduced by the flow field. To do so, we color a number of columns and rows in our array such that a uniform grid appears on the surface.[2] Figure 6 shows this for the flow behind a circular cylinder (explained in section 5.1). After some integration steps, the grid lines clearly allow to distinguish between regions of e.g. rotational and laminar behavior. Furthermore, the direction of rotation becomes visible.

### 4.3 Smoke Nozzles

In flow experiments it is common to inject smoke not from a line but from nozzles aligned in a line. Figures 7a-b show this. We can easily achieve this effect by setting the alpha value of every other row of the array to constant zero. This has been done in figure 7c to visualize the flow around an airfoil (described in section 5.3).

However, this means that vertices are advected that will never be seen in the visualization. To avoid this, we may as well break the connectivity between adjacent rows, i.e., we do not triangulate between two rows. In fact, this splits our seeding curve into different parts which may now be placed arbitrarily in the domain.

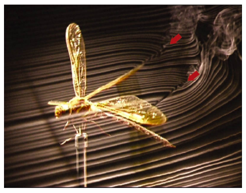
### 4.4 Wool Tufts

In many applications it is of great interest to analyze the flow in the proximity of a boundary, e.g. where the flow might detach from the body of an airfoil or car. Such a flow separation at a boundary often indicates the presence of a recirculation zone which has a negative effect on the drag of the body. Therefore, the design goal of engineers is often to reduce flow detachment. It is commonly visualized in real-world experiments using so-called wool tufts: these are small yarns attached to the body. The different orientations and movements of wool tufts during the experiments allow the experienced viewer to draw conclusions about flow separation.

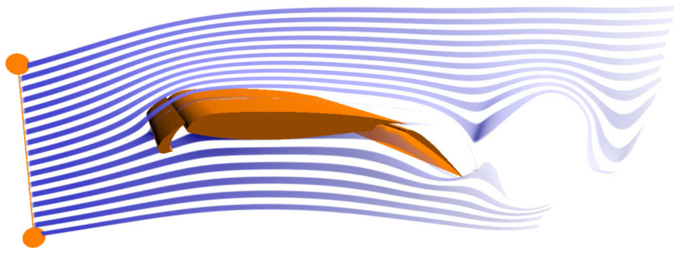
We mimic wool tufts by placing a rather high number of small seeding curves close to the boundary in a flow field. An example of this can be seen in figure 10a where we did this for the flow around an airfoil (described in section 5.3). In our implementation, this can easily be achieved by breaking the connectivity between adjacent rows as described earlier. The result is a set of streak ribbons visualizing the

[2]Note, that all these grid lines are time lines and *not* streak lines.

(a) Flow around an automobile. From [32].

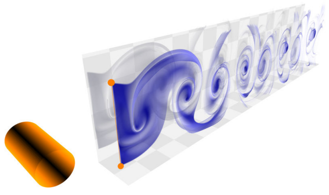


(b) Flow around a dragon fly. From [29].

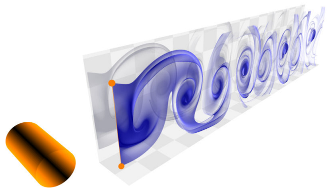


(c) Flow around an airfoil visualized using smoke injected from nozzles aligned along the seeding line at the left.

Figure 7. Injecting smoke from nozzles is a common technique in real-world experiments to yield clearer visualizations. The upper row shows photographs from such setups. We can reproduce this with our system (lower image) either by setting alpha values to constant zero or by breaking the connectivity.



(a) 75k vertices, $\alpha_{density}$ and $\alpha_{shape}$.



(b) 675k vertices, $\alpha_{density}$.

Figure 8. Flow behind a circular cylinder. Smoke surface with different resolutions visualized with shadow to enhance depth perception.

flow in the proximity of the boundary. There are two important differences to real wool tufts. First, real wool tufts have a mass whereas streak ribbons represent the movement of massless particles. Second, our streak ribbons can change their length and therefore they indicate the velocity of the flow not only in terms of direction, but also in terms of magnitude.

## 5 Results

### 5.1 Flow Behind a Circular Cylinder

Figure 6 and 8 demonstrate the results of our method applied to a flow behind a circular cylinder. The data set was derived by Bernd R. Noack (TU Berlin) from a direct numerical Navier Stokes simulation by Gerd Mutschke (FZ Rossendorf). It resolves the so called ‘mode B’ of the 3D cylinder wake at a Reynolds number of 300 and a spanwise wavelength of 1 diameter. The flow exhibits periodic vortex shedding leading to the well known von Kármán vortex street [38]. Figure 6 shows the integration of a time surface including a number of time lines. After some time, the smoke surface clearly shows the distortion introduced by the vortices in the wake of the cylinder. Figure 8 shows the smoke surface advected from a seeding line closely behind the cylinder. To enhance depth perception, the shadow of the smoke was projected to the back wall. Since we work with a triangular mesh, shadow computation is straightforward. Due to the periodic vortex shedding the smoke forms patterns of swirling motion after some integration steps – a clear indication of the von Kármán vortex street.

### 5.2 Flow Behind a Square Cylinder

In figure 9 we visualized the flow around a confined square cylinder. This is a direct numerical Navier Stokes simulation by Simone Camarri and Maria-Vittoria Salvetti (University of Pisa), Marcelo Buffoni (Politecnico of Torino), and Angelo Iollo (University of Bordeaux I) [5] which is publicly available [14]. It is an incompressible solution with a Reynolds number of 200 and the square cylinder has been positioned symmetrically between two parallel walls, where one of them is the wall with the shadow shown in figure 9. The flow has periodic boundary conditions in spanwise direction.

In contrast to the previous cylinder data set (section 5.1), this simulation is initiated from an impulsive start-up and the periodic vortex shedding develops with time. This allows us not only to study this interesting phenomenon, but also to evaluate how our visualization technique performs with increasing unsteadiness of the flow. In order to show the alternating behavior of the vortex shedding, we seeded two smoke surfaces (25000 vertices in total) such that the red one passes above the cylinder and the blue one below.

The flow shows a rather steady behavior for the first time steps and the smoke surface develops almost like an ordinary stream surface. In fact, stream and streak surfaces coincide for steady flows. Once the vortex shedding starts, the flow becomes more unsteady – parts of the smoke are ripped off and transported downstream. Note that all the smoke in figure 9 is internally represented by two smoke surfaces. This example shows that our alpha computation (section 3.1) works reliable even in such challenging situations and produces convincing results. At the end of the transient, the flow develops a von Kármán vortex street with a pronounced three-dimensionality nicely captured by the smoke visualization. We conclude that the visual impression created by smoke surfaces comes closer to real smoke with increasing unsteadiness of a flow.

### 5.3 Flow Around an Airfoil

Figures 7c and 10 show the flow around a Swept-Constant-Chord-Half-model (SCCH) of an airfoil that was simulated by Bert Günther (Technical University Berlin) at a Reynolds number of $10^6$ [11]. The data set exhibits periodic boundary conditions, but note that the airfoil has a sweep angle to the incoming flow direction of 30°. The angle of attack is 6° – thereby conforming to a landing situation. The turbulence was simulated by a combined URANS and DES approach.

The wool tufts visualization of figure 10a allows to describe the underlying physics of this flow. The wool tufts follow the profile on the main element of the airfoil, which means that the flow is still attached to the body. A strong jet of fluid is coming through the gap between the main element and the rear flap as shown by the wool tufts seeded under the body and reaching through the gap. The result of this jet is a detachment of the flow. This can clearly be seen from the wool tufts on the rear flap since they do not follow anymore the general flow direction. Instead, they are directed towards the viewer indicating a strong cross flow section, which is caused by the sweep angle. The wool tufts at the end of the rear flap elucidate the most prominent feature of this flow: a strong vortex created periodically at this position.

In order to increase the lift of such an airfoil, our cooperation partners from the Technical University Berlin try to avoid or minimize both the detachment of the flow at the end of the main element and the periodic vortex shedding at the rear flap. Our wool tufts visualization allows to study both phenomena and has been found useful not at least because of its interactivity. However, the wool tufts visualization shows only the flow in the proximity of the body and does not allow to study the complete characteristics of the flow. Therefore we seeded smoke surfaces close to the regions of flow detachment and vortex creation (figure 10b). This allows us to study the development of these structures away from the body.

Figures 10c-d show the same airfoil, but now a so-called active flow control technique has been applied in order to manipulate the flow structures and achieve a higher lift. This has been done by periodically injecting air at the top of the rear flap (close to the gap). Both visualizations confirm that this excitation led to a better attached flow and less stronger vortex shedding.

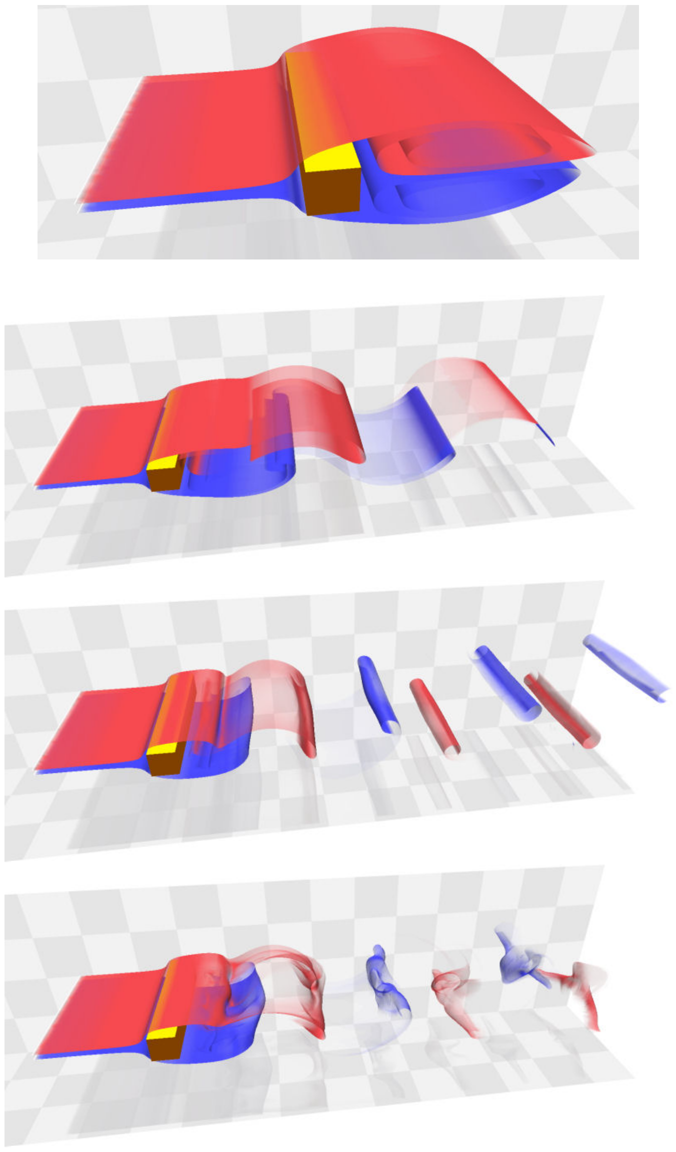


Figure 9. Flow behind a square cylinder. Time is increasing from top to bottom. First the smoke is gathering in a recirculation bubble behind the obstacle. After some time the shedding starts which creates vortices with alternating rotational behavior. Later, the flow develops a pronounced three-dimensionality which perfectly can be observed in the smoke structures.

### 5.4 Ahmed Body

Figure 11 shows the turbulent flow around a bluff body – the so-called Ahmed body. This data set has been computed by Erik Wassen (Technical University Berlin) using a Large-Eddy simulation scheme at a Reynolds number of 500000 based on model length and incoming velocity [36]. Incoming flow is assumed to come from frontal direction. The body stands on a fixed floor with a certain distance to the ground, i.e., a small layer of fluid is passing beneath the model.

The Ahmed body is a generic model for a vehicle, which has been used here in a version with a slanted rear end. The inclination angle of 25° causes a detachment of the flow and the resulting recirculation zone has a rather turbulent behavior over the ramp as it can be observed in figure 11. The wool tufts visualization in figure 11b clearly shows that this recirculation zone affects the vertical rear end as well: the streak ribbons point upwards from their seeding position. In contrast to this, the very last row of ribbons at the vertical rear end points in downstream direction. This indicates the presence of another shear layer created between the recirculation zone and the flow coming from under the body.

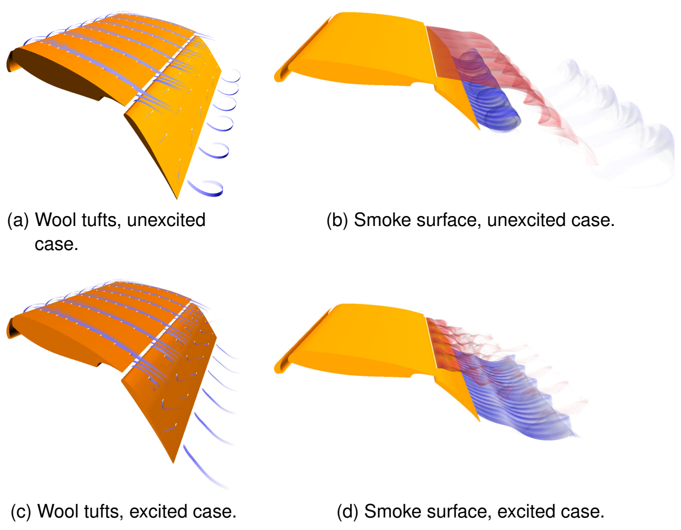

(a) Wool tufts, unexcited case.
(b) Smoke surface, unexcited case.
(c) Wool tufts, excited case.
(d) Smoke surface, excited case.

Figure 10. Unexcited and excited flow around an airfoil. The wool tufts are mimicked using streak ribbons seeded close to the boundary of the airfoil. For example for the unexcited case, their length and orientation shows that the flow detaches from the airfoil and creates a large recirculation zone over the rear flap. Furthermore, the strong vortex created at the bottom of the rear flap becomes obvious. The smoke surfaces are rendered with 40k vertices. They clearly show that the excitation strategy is successful in diminishing the vortex created at the bottom of the rear flap.

Further important structures in this flow are the two vortices created at the upper corners of the ramp. The smoke surface visualizations in figure 11 reveal the cone-like shape of these vortices. It is known that they have a strong impact on the drag of the body. Using our smoke visualization technique we are able to infer a number of important parameters of these vortices: extent, rotation axis, as well as orientation and speed of rotation.

## 6 Evaluation

### 6.1 Performance

Since the technique is based on the integration of triangular meshes, we can achieve an interactive performance of the integration even in a CPU-based implementation. In fact, we used the capabilities of graphics hardware for the semi-transparent rendering of the triangles only. Table 1 shows the results of our performance measurements for smoke surfaces with different resolutions. Note that in our implementation the flow data set is given on a regular grid, making the performance depending only on the size of the streak surface and not of the data set. In fact, all visualizations in this paper and the accompanying movies use a streak surface of 50000 or less vertices, leading to interactive frame rates in an CPU based implementation.

### 6.2 Correctness

In our approach we propose to represent smoke surfaces using meshes with a fixed medium resolution and fixed connectivity in order to maintain interactive frame rates. As already argued in section 3, this is feasible since the physically motivated opacity $\alpha_{density}$ (5) gets lower with increasing triangle area, i.e., larger triangles are less visible due to the smoke metaphor anyway.

However, fixed resolution and connectivity produce artifacts e.g. where the smoke flows around obstacles or when the surface becomes strongly distorted. We addressed these problems by introducing $\alpha_{shape}$ and $\alpha_{curvature}$, which are designed to lower the opacity of the surface at places where it deviates too much from the "real" streak surface. Is

| Vertices | Fps | Rendering | Integration | Alpha + normals |
|---|---|---|---|---|
| 5000 | 85 | 1 ms | 6 ms | 4 ms |
| 10000 | 44 | 1 ms | 13 ms | 9 ms |
| 20000 | 25 | 4 ms | 21 ms | 15 ms |
| 30000 | 18 | 4 ms | 31 ms | 21 ms |
| 40000 | 14 | 4 ms | 39 ms | 29 ms |
| 50000 | 11 | 6 ms | 48 ms | 37 ms |

Table 1. Performance figures of our implementation tested on a 2.6 GHz Opteron CPU with 2 GB RAM and a GeForce 7800 GTX. *Fps* stands for frames per second, *Rendering* gives the rendering time of the semi-transparent surface, *Integration* the time for the integration of all vertices, and *Alpha + normals* the computation time for vertex normals and alpha values.

the result still correct in the sense that all flow features like vortices are shown in the visualization?

To answer this question we produced "ground truth" visualizations using streak surfaces with excessive resolutions, rendered them using $\alpha_{density}$ only and compared the results with their medium-sized counterparts. Figure 8 shows this for the flow behind a circular cylinder. The resolution of the surface in figure 8b is nine times larger than in figure 8a. As it can be seen, both versions faithfully represent the vortices of this flow and show very similar smoke patterns.

Figure 11c shows such a comparison for the Ahmed data set. Although the resolution of the two surfaces differs by a factor of 16, the low-res surface nicely captures all structures of this turbulent flow. In particular, the conical vortices coming from the corners of the ramp have a disrupted smoke appearance on both sides, i.e., this is a pattern of this flow and not due to the additional opacity parameters for the low-res surface. Note that although the Ahmed body itself is symmetrical, the flow around it is not (mainly due to turbulence).

We conclude that our method produces physically correct renderings considering the given resolution when $\alpha_{density}$ is applied only. The optional parameters $\alpha_{shape}$ and $\alpha_{curvature}$ have been carefully designed to reduce visual clutter caused by locally too coarse resolutions while keeping the overall appearance very similar.

### 6.3 Perception

In terms of perception, the question arises: does the visualization really look like smoke? To answer this, we did not do a formal user study. However, informal reactions of a number of people confronted with the visualizations (visualization experts, flow simulation experts, and non-experts) unanimously agreed that they indeed see smoke.

Another question is whether the technique is able to detect relevant features in the flow. Here we refer to the tested applications in section 5. Since part of the data sets were known in advance, we could confirm that smoke surfaces indeed were able to detect relevant features.

### 6.4 Comparison to other smoke visualization techniques

Using particle-based methods, a large number of particles would be necessary to get a detailed, surface-like smoke appearance as in figure 1. Being surface-based, our method is able to get this kind of appearance directly and requires a rather low number of vertices. In contrast to particle-based representations, self-shadowing is straightforward because the surface normals are available in our mesh representation.

While volumetric methods are well-suited for the visualization of thick and diffuse smoke, thin surface-like smoke structures would require high-resolution voxel grids, which comes at the cost of memory usage and speed. In contrast to this, our surface-based approach can visualize surface-like smoke using a rather low mesh resolution.

Smoke surfaces do not intend to replace particle-based or volume-based smoke visualization techniques. However, for some situations we see advantages of smoke surfaces, making them an alternative to previous techniques. Smoke surfaces are simple: less geometric primitives are necessary to obtain expressive visualizations than for particle based or volumetric approaches. Because of this, no specialized graphics hardware is required for integration to obtain interactive frame rates. In general, smoke surfaces are the appropriate smoke representation if the smoke has an approximate surface shape, i.e., the seeding structure is a moving curve.

### 6.5 Limitations

Our method has a number of limitations. Our current implementation handles regular grids only and requests that the complete time-dependent data set is kept in main memory. However, these are not structural problems of smoke surfaces but general challenges for every interactive visualization software.

If the smoke to be visualized is known to be not surface-like but really volumetric (e.g., if the seeding structure is a volume instead of a line structure), then smoke surfaces are not the appropriate method. In these cases, particle based or volumetric approaches will give better results.

## 7 Conclusions

In this paper, we have made the following contributions:

- We introduced a new representation of smoke in a flow: as semi-transparent streak surface.
- For the first time, streak surfaces are used for an interactive visualization of time-dependent flow fields. This was possible by avoiding an expensive adaptive remeshing of the surface.
- By coupling the opacity of the triangle to their area, shapes, and curvatures, we obtain the impression of smoke seeded from line structures. This allows an intuitive and interactive exploration of the flow.
- By slightly changing the setup, we obtained a representation of wool tufts which are well-known from experimental flow visualization.

For future research, the performance can be further increased. Table 1 clearly shows that the current bottle neck is the integration and the computation time for normals and $\alpha$-values. Transforming them to the GPU may further improve performance. Furthermore, the approach should be extended to handle irregular grids, and an out-of-core mechanism should be included to handle data sets which do not fit into main memory.

## References

[1] A. Angelidis and F. Neyret. Simulation of smoke based on vortex filament primitives. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 87–96, 2005.

[2] A. Bouthors, F. Neyret, and S. Lefebvre. Real-time realistic illumination and shading of stratiform clouds. In *Eurographics Workshop on Natural Phenomena*, pages 41–50, 2006.

[3] W. Brennan. Smoke abstractions, 2005. http://www.pbase.com/billyb2/.

[4] K. Bürger, J. Schneider, P. Kondratieva, J. Krüger, and R. Westermann. Interactive visual exploration of instationary 3D-flows. In *Proc. EuroVis '07*, pages 251–258, 2007.

[5] S. Camarri, M.-V. Salvetti, M. Buffoni, and A. Iollo. Simulation of the three-dimensional flow around a square cylinder between parallel walls at moderate reynolds numbers. In *XVII Congresso di Meccanica Teorica ed Applicata*, 2005.

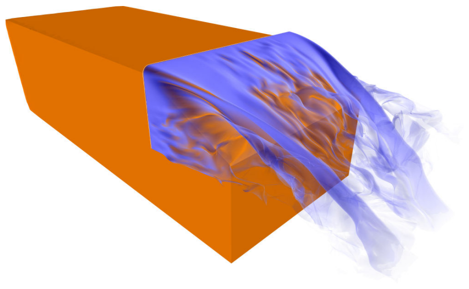
[6] K. Crane, I. Llamas, and S. Tariq. Real-time simulation and rendering of 3d fluids. In *GPU Gems 3*, chapter 30, pages 633–676. Addison-Wesley Professional, 2007.

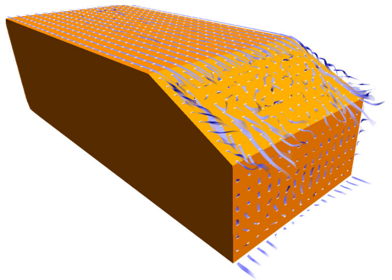
[7] R. A. Crawfis and N. Max. Texture splats for 3D scalar and vector field visualization. In *Proc. IEEE Visualization '93*, pages 261–266, 1993.

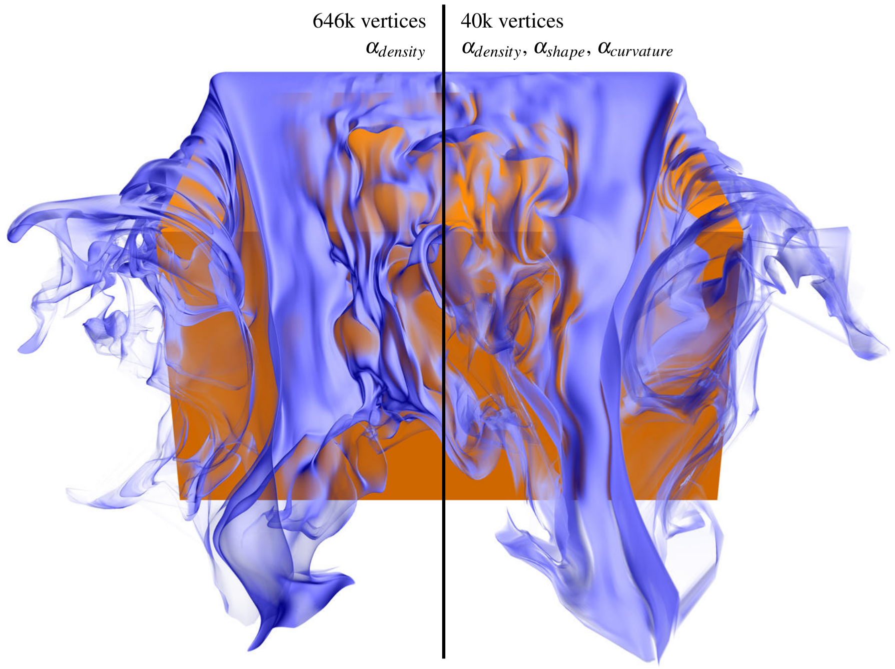
[8] C. Everitt. Interactive-order independent transparency. *NVIDIA*, 2001. http://developer.nvidia.com/object/Interactive_Order_Transparency.html.

[9] G. Y. Gardner. Visual simulation of clouds. *SIGGRAPH Comput. Graph.*, 19(3):297–304, 1985.

[10] C. Garth, X. Tricoche, T. Salzbrunn, T. Bobach, and G. Scheuermann. Surface techniques for vortex visualization. In *VisSym*, pages 155–164, 346, 2004.


(a) Smoke surface with 40k vertices seeded shortly before the ramp and visualized with $\alpha_{density}$, $\alpha_{shape}$, $\alpha_{curvature}$, and $\alpha_{fade}$.


(b) Wool tufts attached to the body. They indicate a recirculation zone at the vertical rear end where they are oriented upwards.



(c) Smoke surface with different resolutions viewed from the back. The high-res surface on the left serves as a ground truth. The low-res version can be rendered interactively and shows the same smoke structures, but needs additional opacity terms to hide distorted triangles.

Figure 11. Flow around the Ahmed body. Both visualizations – wool tufts and smoke surface – reveal the strong vortices created at the corners of the slant as well as the turbulence in the middle of the slant.

[11] B. Günther, F. Thiele, R. Petz, W. Nitsche, J. Sahner, T. Weinkauf, and H.-C. Hege. Control of separation on the flap of a three-element high-lift configuration. In *45th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, U.S.A., January 2007. AIAA-2007-265.

[12] T. Holtkämper. Real-time gaseous phenomena: a phenomenological approach to interactive smoke and steam. In *Proc. AFRIGRAPH '03*, pages 25–30, 2003.

[13] J. Hultquist. Constructing stream surfaces in steady 3D vector fields. In *Proc. IEEE Visualization '92*, pages 171–177, 1992.

[14] International CFD Database, http://cfd.cineca.it/.

[15] H. W. Jensen and P. H. Christensen. Efficient simulation of light transport in scences with participating media using photon maps. In *SIGGRAPH '98*, pages 311–320, New York, NY, USA, 1998. ACM.

[16] J. T. Kajiya and B. P. V. Herzen. Ray tracing volume densities. *SIGGRAPH Comput. Graph.*, 18(3):165–174, 1984.

[17] J. Krüger, P. Kipfer, P. Kondratieva, and R. Westermann. A particle system for interactive visualization of 3D flows. *IEEE Transactions on Visualization and Computer Graphics*, 11(6):744–756, 2005.

[18] J. Krüger and R. Westermann. GPU simulation and rendering of volumetric effects for computer games and virtual environments. *Computer Graphics Forum*, 24(3):685–693, 2005.

[19] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.

[20] N. Max, B. Becker, and R. Crawfis. Flow volumes for interactive vector field visualization. In *Proc. Visualization 93*, pages 19–24, 1993.

[21] T. Schafhitzel, E. Tejada, D. Weiskopf, and T. Ertl. Point-based stream surfaces and path surfaces. In *Graphics Interface*, pages 289–296, 2007.

[22] G. Scheuermann, T. Bobach, H. H. K. Mahrous, B. Hamann, K. Joy, and W. Kollmann. A tetrahedra-based stream surface algorithm. In *Proc. IEEE Visualization 01*, pages 151 – 158, 2001.

[23] J. Schpok, J. Simons, D. S. Ebert, and C. Hansen. A real-time cloud modeling, rendering, and animation system. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 160–166, 2003.

[24] A. Selle, A. Mohr, and S. Chenney. Cartoon rendering of smoke animations. In *NPAR '04: Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, pages 57–60, 2004.

[25] W. Shen and A. Pang. Tuft flow visualization. In *Proc. Second IASTED International Conference on Visualization, Imaging, and Image Processing*, pages 705–710, 2002.

[26] J. R. Shewchuk. What is a good linear element? interpolation, conditioning, and quality measures. In *Eleventh International Meshing Roundtable*, pages 115–126, 2002.

[27] J. Stam and E. Fiume. Turbulent wind fields for gaseous phenomena. In *Proc. SIGGRAPH '93*, pages 369–376, 1993.

[28] S. Tariq and I. Llamas. Real-time volumetric smoke, fire and water with fluid dynamics. *NVIDIA Sessions at SIGGRAPH 2007*, 2007. http://developer.nvidia.com/object/siggraph-2007.html.

[29] A. L. R. Thomas, G. K. Taylor, R. B. Srygley, R. L. Nudds, and R. J. Bomphrey. Dragonfly flight: free-flight and tethered flow visualizations reveal a diverse array of unsteady lift-generating mechanisms, controlled primarily via angle of attack. *J Exp Biol*, 207(24):4299–4323, 2004.

[30] A. Trembilski and A. Broßler. Surface-based efficient cloud visualisation for animation applications. *Journal of WSCG*, 10(2):453–460, 2002.

[31] T. Umenhoffer, L. Szirmay-Kalos, and G. Szijártó. Spherical billboards and their application to rendering explosions. In *GI '06: Proceedings of Graphics Interface 2006*, pages 57–63, 2006.

[32] Use of Drafting in Racing, http://www.aerospaceweb.org/.

[33] J. van Wijk. Implicit stream surfaces. In *Proc. Visualization 93*, pages 245–252, 1993.

[34] J. J. van Wijk. Rendering surface-particles. In *Proc. IEEE Visualization '92*, pages 54–61, 1992.

[35] J. J. van Wijk. Image based flow visualization. In *Proc. ACM SIGGRAPH '02*, pages 745–754, 2002.

[36] E. Wassen and F. Thiele. LES of wake control for a generic fastback vehicle. In *37th AIAA Fluid Dynamics Conference and Exhibit*, Miami, U.S.A, 2007. AIAA-2007-4504.

[37] D. Weiskopf. Dye Advection Without the Blur: A Level-Set Approach for Texture-Based Visualization of Unsteady Flow. *Computer Graphics Forum (Eurographics 2004)*, 23(3):479–488, 2004.

[38] H.-Q. Zhang, U. Fey, B. Noack, M. König, and H. Eckelmann. On the transition of the cylinder wake. *Phys. Fluids*, 7(4):779–795, 1995.

[39] K. Zhou, Z. Ren, S. Lin, H. Bao, B. Guo, and H.-Y. Shum. Real-time smoke rendering using compensated ray marching. *ACM Transactions on Graphics (ACM SIGGRAPH)*, 27(3), 2008.