Importance-Driven Particle Techniques for Flow Visualization

Kai Bürger* Polina Kondratieva†
Computer Graphics and Visualization Group, Technische Universität München
Jens Krüger‡
Scientific Computing and Imaging Institute, University of Utah
Rüdiger Westermann§
Computer Graphics and Visualization Group, Technische Universität München

ABSTRACT

Particle tracing has been established as a powerful visualization technique to show the dynamics of 3D flows. Particle tracing in 3D, however, quickly overextends the viewer due to the massive amount of visual information that is typically produced by this technique. In this paper, we present strategies to reduce this amount at the same time revealing important structures in the flow. As an importance measure, we introduce a simple, yet effective clustering approach for vector fields, and we use scalar flow quantities at different scales in combination with user-defined regions of interest. These measures are used to control the shape, the appearance, and the density of particles in such a way that the user can focus on the dynamics in important regions at the same time preserving context information. We also introduce a new focus for particle tracing, so called anchor lines. Anchor lines are used to analyze local flow features by visualizing how much particles separate over time and how long it takes until they have separated to a fixed distance. It is of particular interest if the finite time Lyapunov exponent - a scalar quantity that measures the rate of separation of infinitesimally close particles in the flow - is used to guide the placement of anchor lines. The effectiveness of our approaches for the visualization of 3D flow fields is validated using synthetic fields as well as real simulation data.

Keywords: Flow visualization, particle tracing, GPU rendering.

Index Terms: I.3.7 [Computing Methodologies]: COMPUTER GRAPHICS—Three-Dimensional Graphics and Realism; I.3.3 [Computing Methodologies]: COMPUTER GRAPHICS—Picture/Image Generation

1 INTRODUCTION AND RELATED WORK

The effective visualization of 3D flow fields is still difficult because no existing technique is able to reveal all relevant information in such fields. Topological methods as introduced by Helman and Hesselink [10], as well as feature-based techniques in general [15, 20], can effectively condense a field to physical meaningful structures, such as topological skeletons and vortices, to name just a few. Such techniques, on the other hand, typically require expensive pre-processing and, in general, do not allow for a direct relation to the original data. Texture-based methods like spot-noise [27], line integral convolution (LIC) [3, 26], and all of the many variants of these techniques [14] generate a dense visualization of the flow that can effectively reveal the directional information in 2D fields. In 3D, however, these techniques introduce perceptual problems as they generate a representation of the entire field. Even by using advanced visualization options like halos [11], transfer functions and clipping geometries [21], and flow-guided importance measures [31], it has been shown difficult to overcome these limitations without loosing relevant information.

As an alternative to the aforementioned flow visualization techniques, particle tracing has positioned itself as a technique to interactively visualize the dynamics of complex 3D flows. For a thorough overview of particle-based and related integration-based techniques for flow visualization let us refer to the report by Post et al. [19]. Particle tracing has the advantage that it does not require any pre-processing and can effectively reveal the dynamics in 3D steady and unsteady flows with respect to speed and direction. Since particle tracing can effectively be used to focus on characteristic trajectories of selected particles in the flow, it is well suited for user-guided visual exploration of flow fields. In particular, due to recent advances in graphics hardware and algorithms, it is now possible to interactively trace and visualize massive amounts of particles in 3D flows [1, 23, 25, 13, 2].

In principle, however, particle tracing suffers similar problems as texture-based techniques in that rendered particle primitives overlap and occlude each other. If large sets of primitives are seeded, the same perceptual problems inherent to dense global techniques arise and important information might be obscured. These limitations can be partially overcome by real-time techniques, i.e. by enabling the user to interactively control the number of seeded particles and their starting positions. Other approaches restrict the visualization to particles moving on or close to specific surfaces in the flow [28, 18]. These techniques effectively restrict the visualization to a focus region, or in the particular case to a focus surface, but by doing so important context information as well as relevant structures not in this region can be lost.

The most inspiration of our work came from previous work on focus+context techniques for scientific visualization as well as feature-based flow visualization. For flow visualization, Fuhrmann and Gröller [6] proposed the combination of a user-controlled focus region and a uniform stream line placement strategy. Within the focus region the flow field is visualized at the highest resolution level, and contextual information is preserved by visualizing a sparse set of primitives outside this region. Löffelmann and Gröller [16] presented a feature-based focus for 3D dynamic systems. By visualizing short stream lines, so called streamlets, only close to a base trajectory in a 3D vector field, occlusion problems could be avoided thus providing a detailed view of particular regions in the field. For 2D flow visualization, Kirby et al. [12] defined a focus by combining visual elements of different size, shape and texture into a multi-layer representation. Doleisch and Hauser [4] presented non-discrete 3D regions of interest including techniques to blend between differently shaped primitives. Mattausch et al. [17] introduced more flexible and interactive focusing strategies as well as multiple options to adaptively modulate the density and appearance of stream lines in 3D flow fields.

*kai.buerger@in.tum.de
†kondrati@in.tum.de
‡mail@jens.krueger.com
§westerma@in.tum.de

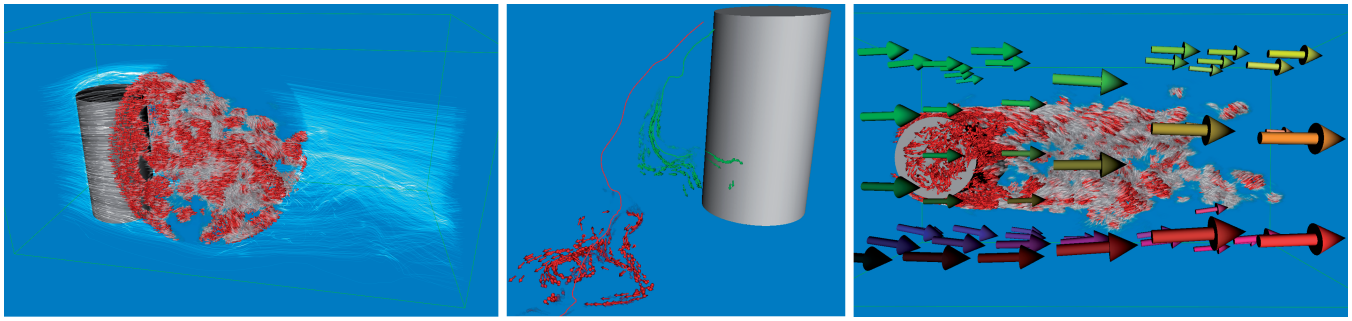

Figure 1: Different particle-based strategies are used to visualize 3D flow fields. (Left) Focus+context visualization using an importance measure based on helicity density and user-defined region of interest. (Middle) Particles seeded in the vicinity of anchor lines show the extend and speed at which particles separate over time. (Right) Cluster arrows are used to show regions of coherent motion.

2 Contribution

In this paper, we propose a number of improvements for particle-based 3D flow visualization. Common to all techniques we present is a significant reduction of the amount of visual information presented to the user. Consequently, these techniques are less prone to perceptual artifacts like occlusions, and they can avoid visual clutter that is introduced by frequent positional changes of large amounts of particles. Relevant structures in the flow are emphasized by integrating user-controlled and feature-based importance measures. The suggested techniques extend previous approaches for particle-based visualization of 3D flows in the following ways (see Figure 1 for a graphical illustration):

- We present techniques to automatically adapt the shape, the appearance, and the density of particle primitives with respect to user-defined and feature-based regions of interest. We also provide means for smooth blends between differently shaped primitives. Thus, the proposed techniques can effectively be used in combination with continuous focus+context and importance measures. Figure 1 (left) demonstrates the possibilities these techniques offer.

- In addition to vorticity and helicity density we consider the finite time Lyapunov exponent as an importance measure. In particular, we employ this measure for the selection of characteristic trajectories in the flow. We call these trajectories anchor lines, and we seed particles close to the starting points of these lines. By only visualizing those particles that leave the anchor, the amount of visual information can be reduced significantly (see middle of Figure 1). Furthermore, we use this approach to allow quantitative statements about the particle movement over time and space.

- We propose a clustering approach to determine regions of coherent motion in the flow, and we use a sparse set of static cluster arrows to emphasize these regions. Figure 1 (right) shows such arrows in combination with feature-based rendering of primitives in focus region. As can be seen, occlusion problems in the visualization of contextual information can be avoided, thus enabling a flexible integration of detail information into selected focus regions.

- All visualization techniques have been integrated into a GPU particle engine. This enables the user to interactively select visualization parameters and rendering modes, and it thus allows for an effective visual analysis of 3D flow structures.

The remainder of this paper is organized as follows. In the next chapter we will discuss the focus+context metaphor underlying our approach, and we will detail the implementation of it. We will then describe the meaning of anchor lines as well as the used particle seeding and rendering strategy. Next, we present the different GPU-based rendering modes we have developed. An analysis of the performance of the proposed techniques on recent GPUs is given in Chapter 6. We conclude the paper with an outline of future research in the field of particle-based flow visualization.

3 Importance-based particle visualization

One important goal in particle-based flow visualization is to reduce the amount of visual information presented to the viewer. This is due to the following observations: Firstly, many interesting flow structures are typically occluded by the primitives rendered in non-interesting or surrounding regions of the flow. Secondly, a large amount of moving particles, often performing rapid directional changes, produces visual clutter that quickly overloads the humans' perceptual system.

While it is easy in general to simply restrict the visualization of particles to user-defined focus regions, this approach typically results in the loss of contextual information necessary to understand the global relationships between flow structures. The focus+context paradigm seeks to combine both aspects into a single visual event by presenting a detailed region in combination with a surrounding context. The visual information used to represent the context region must not occlude details in the focus regions, but at the same time it should indicate characteristic structures in the data. For an overview of focus+context techniques in visualization let us refer to the tutorial of Viola et al. [30].

To use focus+context techniques in particle-based flow visualization, two different strategies have to be pursued: Firstly, the spatial density of visualized particles should be adapted according to the importance classification. Secondly, the appearance of rendered particle primitives should reflect the importance of the region they are traveling through. In the following, we will first describe how to flexibly adjust the shape and the appearance of rendered particle primitives based on their importance.

3.1 Scale-space particles

In the following we assume, that an importance mapping is given, i.e. a function $fc(\vec{x}) \longrightarrow [0,1]$ that can be evaluated at every point $\vec{x}$ in the domain to yield the local importance of the vector field at this point. The larger the value is the higher the importance that is given to this point. Such a function can either be given by a user-defined focus point and an attenuation function defining the decrease in importance from this point, or by an additional importance volume that can be sampled at the respective point position. In addition, both measures can be combined to highlight flow features not only in the focus region. Some possible importance measures directly

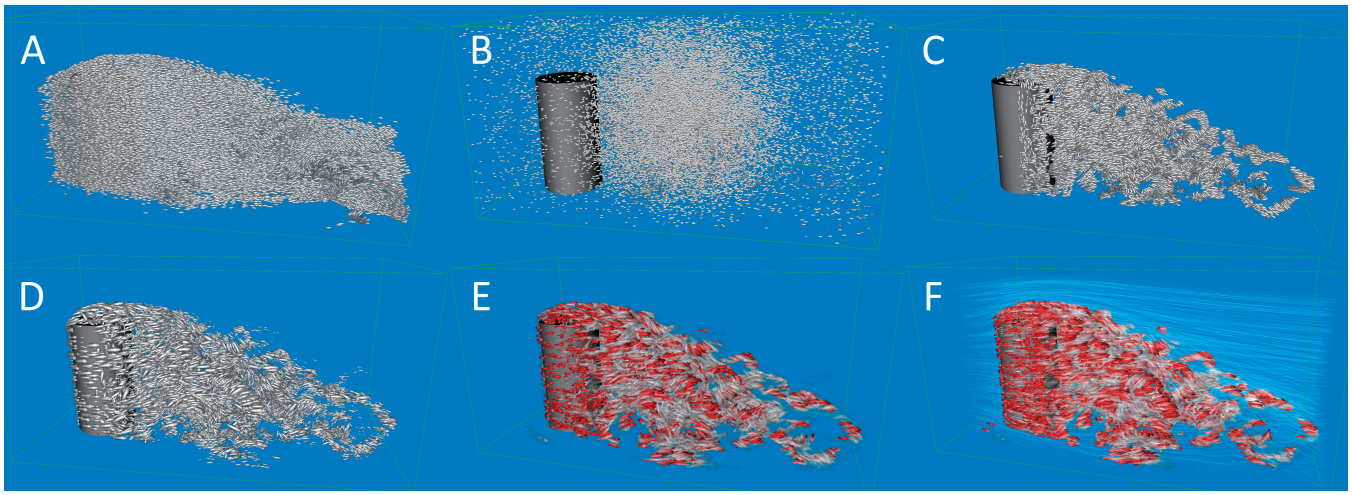

Figure 2: Different approaches for 3D flow visualization using particles are shown. Image A shows one time step of an unsteady flow around a cylinder, visualized by a large amount of particles. In image B, a region of interest has been selected, and with increasing distance to the center of this region the particle density is decreased. Image C shows the effect of importance-driven density adjustment (vorticity was used as importance measure in this example). Particles out of focus regions are removed. In image D, the particle shape is adjusted according to the importance of the region they are traveling through. In this example, the shape is morphed from a small arrow (high importance) to a large ellipsoid (low importance). In image E, in addition to the shape transformation the transparency and color of the particles are transformed. Image F shows the integration of transparent stream lines to sketch the flow structure in less important regions.

derived from the physical properties of the flow will be discussed below.

Once an important mapping is given, the reduction of the amount of information displayed can be achieved by adaptively reducing the number of rendered particle primitives based on local importance. Therefore, we employ a similar approach as used in [32] for the selection of hatches in illustrative volume rendering. Every particle seeded into the flow is assigned a random value in the range of $[0,1]$, and a particle is rendered if the importance at its current position is higher than this random value. By this approach more and more particles are removed with decreasing importance. Figure 2b shows this effect for a user-controlled focus region positioned right behind the cylinder in the flow. As can be perceived, this approach neither emphasizes characteristic structures in the context region nor does it allow a clear distinction between what is in focus and what is not.

To overcome this problem we increase the particle size by a factor inversely proportional to importance. In Figure 2c, where vorticity is used as an importance measure, the resulting visual effect is shown. Although we now obtain a better understanding of the context information, focus and context can still not be clearly distinguished because of the same shape and appearance of the particles being rendered. We thus transform the primitives continuously from a particular shape used to depict the focus region into a shape that indicates the context. In the current example we transform an arrow glyph into an ellipsoid as shown in Figure 2d. Shape morphing allows us to quickly obtain an image of both the focus and the context information, but the visualization still suffers from occlusions due to a few but large particles overlaying the focus region. This problem is finally alleviated by using transparency to fade out particles in the context regions. As can be seen in Figure 2e, we do not remove particles entirely, but we make them highly transparent. The particle color changes from a light shade of grey in the context to saturated red in the focus.

Overall, the following transfer functions are used to adjust the visual attributes of the ith particle:

$$\begin{aligned} show_i &= (rand_i > fc(\vec{x}_i)) \\ size_i &= s + (1 - fc(\vec{x}_i)) \cdot C_s \\ \sigma_i &= (1 - fc(\vec{x}_i)) \cdot C_\sigma \\ color_i &= LUT[\sigma_i] \end{aligned}$$

Here, $show_i$ and $size_i$ are the particles visibility and size, respectively. s is the base size of every particle. $rand_i$ stores a random value in the range [0,1]. σ_i is the particle transparency. User-defined constants C_s and C_σ specify how fast particles are fading out according to decreasing importance. Finally, the color of every particle can be modulated by means of a user-defined color transfer function.

By means of the proposed transfer functions the amount of information that is displayed can effectively be reduced. In addition, with increasing size and transparency of the particles being shown their spatial movements appear increasingly smooth. Thus, visual clutter, as it is typically observed when rendering small and opaque moving primitives, can mostly be avoided.

3.2 Feature-based importance measures

To enable the visualization of characteristic structures in the flow we have integrated a number of different importance measures based on physical properties of the flow. In addition to velocity and vorticity magnitude, the following scalar quantities of the vector field $\mathbf{F} = (F_x, F_y, F_z)^T$ are used:

- *Helicity density* - the extent to which corkscrew-like motion occurs:

$$h = \boldsymbol{\omega} \cdot \mathbf{F} = \omega_x F_x + \omega_y F_y + \omega_z F_z \quad (1)$$

- *Maximum Finite Time Lyapunov Exponent* - the maximum rate of separation of particles in a dynamic system:

$$\sigma_{t_0}^T = \frac{1}{|T|} ln \sqrt{\lambda_{max}(\Delta_{t_0}^T(x))} \quad (2)$$

The finite time Lyapunov exponent depends on the initial positions of the trajectories in space (x) and time (t_0), as well as the time T of integration of these trajectories. λ_{max} is the largest eigenvalue of the finite time Cauchy-Green deformation tensor $\Delta_{t_0}^{T}$, which is derived from the perturbation of two infinitesimally close particles after a fixed time. The FTLE measures the stretching induced by the flow by computing the maximum separation of two infinitesimally close particles when moving with the flow over a fixed time. For a thorough description of the finite time Lyapunov exponent the reader is referred to [9, 24, 22].

All these quantities are hierarchically encoded in a pyramidal data structure. We use a *min-max* and an *average* pyramid of volumes where the first level is the original vector field and each successive volume is reduced about a factor of two in each dimension. Thus, only a small memory overhead is introduced. Each sample in the nth level of the pyramid stores the minimum/maximum or average importance of its eight children in the $n-1$st level of the pyramid. Thus, the pyramid maintains the minimum/maximum or average importance in ever increasing regions of the domain. The kind of pyramid and the level that should be considered as importance measure can be specified by the user.

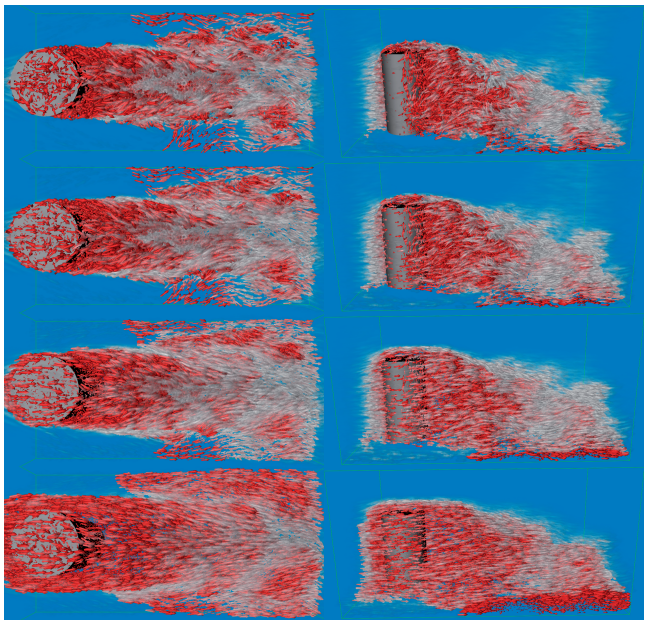

Figure 3: The velocity magnitude at different resolution levels is used as an importance measures (coarser levels from top to bottom). The color coding is from opaque red (low velocity) to transparent grey (high velocity). Different views of the same data set are shown.

By using a feature hierarchy and tri-linear interpolation to reconstruct values form this hierarchy, three different effects can be achieved: First, spurious features can be suppressed by letting the importance be sampled from coarser levels in the pyramid. This is especially useful in the context region to avoid frequent changes of the particle appearance. Second, there is a smooth transition between regions of different importance. Third, continuous regions of interest are supported by smoothly interpolating between different levels in the hierarchy. To show these effects we have conducted experiments with different importance measures, encoded in a multiresolution hierarchy. Figure 3 shows some results indicating the suitability of feature-based importance measures in combination with a user-defined focus region. In particular it can be seen, that even in context regions important features are still emphasized, effectively guiding the visual exploration process.

3.3 Cluster arrows

To further assist the user in the visual analysis of the flow we propose *cluster arrows* as a sparse, and static visualization metaphor. Cluster arrows are geometric primitives that represent regions of constant motion in the flow. The positions at which these primitives are placed are computed in a preprocess using a region growing approach. To find a cluster, i.e. a region in which the velocity directions do not differ more than a given angle, we randomly select a grid point that has not yet been processed, and we inspect the velocities of all of its 26 neighbors in the grid. If none of the velocities in the subtended region diverges more than a given angle from the average of all velocities in that region we continue to grow the cluster until no further expansion is possible. The average velocity of all grid points in the cluster is stored as a representative for the entire region. This process is continued until the entire domain is partitioned into clusters.

During rendering the average velocity stored for each cluster is used to draw an oriented geometric primitive, scaled according to the cluster size. Our system also allows the user to select a minimum and maximum size of the clusters to be visualized. This makes it possible to hide large arrows that would otherwise occlude relevant information, as well as small clusters that would clutter into focus regions in which dynamic particles are shown. The cluster information is also used as an additional importance metric for the rendered particle primitives. Therefore, for every sample point in the grid we store the size of the corresponding cluster, and we fade out rendering primitives with increasing cluster size as demonstrated in Figure 4.

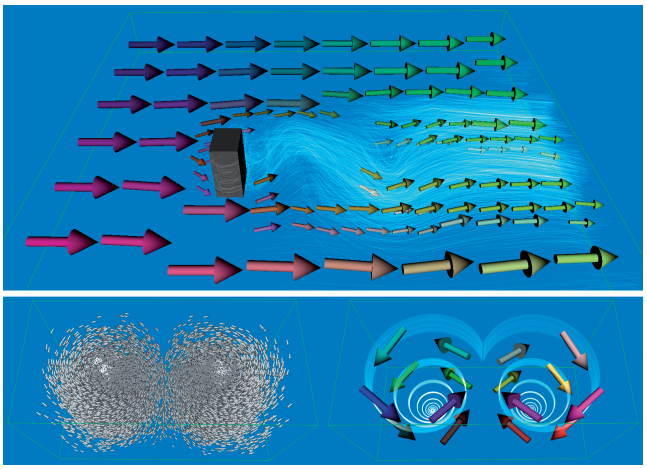

Figure 4: In the top image, cluster arrows and transparent lines are used to indicate coherent and less coherent motion in the flow, respectively. The size of the arrows corresponds to the size of the cluster they represent. In the bottom image (right), the same visualization technique is used. It is compared to a visualization of the same double-vortex flow using arrows as particle primitives (left).

4 Anchor Lines

As can be seen in the images presented so far, importance-driven visualization techniques for 3D flows using scale-space particles can effectively be employed to focus on particular regions and features at the same time maintaining context information. On the other hand, these techniques are problematic, because in the focus region and in regions of high importance the amount of visual

information is still high. To overcome this problem we propose *anchor lines*, a new focus for particle tracing that enables the user to emphasize characteristic information about particle divergence and convergence.

The idea behind anchor lines stems from the observation that one is typically not interested in a detailed visualization of flow regions in which the trajectories of particles do not diverge. Instead, such regions should only be outlined by a few representative primitives. It is of interest, however, to emphasize regions in which trajectories diverge, for instance at saddles, sources or separatrices.

A scalar quantity that can be used to give evidence for the rate of divergence or convergence of neighboring trajectories in a flow is the finite time Lyapunov exponent (FTLE), which has been introduced in subsection 3.2. Most generally, the FTLE is a measure of the average growth of an infinitesimally small error in the vicinity of a point in the domain. In fluid dynamics, the FTLE is used as a measure of the amount of stretching of a fluid element over a fixed time. It allows to locate transport barriers and it has been studied for the analysis of transport and mixing characteristics in multi-dimensional flows.

In the following, we will introduce anchor lines as a means to locally analyze the FTLE measure. In particular, anchor lines can be used to interactively visualize how much particles separate over time and how long it takes until they have separated to a fixed distance. We extend the idea proposed in [16], where short stream lines where placed in the vicinity of characteristic trajectories to show the local flow behavior along these trajectories. To do so, we first define a set of path lines in the vector field - the anchor lines. The user can select these lines by placing their starting points in the domain. Then, additional particles are seeded in close vicinity of these starting points, with the amount of scatter around these points being selected by the user. The particles' transparency is set according to their deviation from the corresponding anchor line, i.e. particles close to the line are faded out while they are rendered more and more opaque once they start to diverge.

Technically speaking, anchor lines are always traced in parallel, and in every integration step the Euclidean distance between a particle and the current point on the line is used as a measure for the deviation. Since a particle trajectory and an anchor line can deviate from each other and again approach each other, it makes sense to consider the maximum deviation of a particle along its path. This means that once the particle deviates more than a specified threshold from the corresponding anchor line it is rendered opaque along the remaining path.

Since high transparency is given to particles that remain close to the anchor line, particles are automatically faded out in regions where there is a high similarity between neighboring vector field values. In such regions only the respective anchor line is shown. Particles in highly heterogeneous regions where the separation is high are emphasized (see Figure 5). From the transparency of a particle it can directly be derived how much this particle separates from the anchor line over time. The time a particle has traveled until it deviates on a fixed distance from the anchor is not directly encoded as a visual attribute, but it can be determined from the animation of particles over time and could also be encoded as an additional attribute like color or size.

In addition to the user-controlled placement of anchor lines, we propose to select the starting points of these lines automatically. In particular, we let points be positioned in the interior of a user-defined probe, but we only accept a point as a starting point if the FTLE at its position is above a certain threshold. It is worth noting here, that the FTLE is pre-computed at every point of the given sampling grid.

The reason for restricting the placement of anchor lines to regions of high FTLE is as follows. While the FTLE characterizes the rate of separation of particles, it does neither indicate into which direction particles separate nor does it tell where the particles separate along a trajectory. Anchor lines placed in regions of high FTLE, on the other hand, are able to answer both questions and can thus be used for an improved analysis of the flow. Figure 6 demonstrates this property.

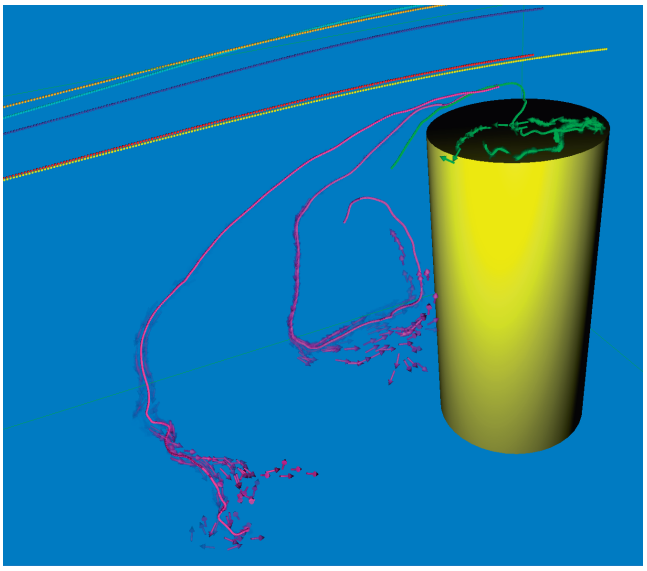

Figure 5: Anchor lines (path lines) and particles seeded close to their starting points are shown. While particles exactly follow some of the lines (blue, red, yellow), along other lines the particles diverge from their anchor lines at different speed (green, purple). To improve the visual perception of the correspondence between anchor lines and particles, every anchor line gets assigned a unique color that is inherited by the particles seeded close to it. Particle transparency is inversely proportional to the separation distance from the anchor line.

The deviation of particles from their anchor lines, and thus their transparency, is computed as follows. Particle positions are stored in a pair of 2D textures, which are alternatively updated in every advection step. In parallel to the particle trajectories the anchor lines are traced. Positions along these lines are stored in an additional 2D texture of size $n \times m$, where n is the number of anchor lines and m is their maximum length. Initially, every particle gets assigned an index i that is used to reference the corresponding anchor line. In the jth advection step every particle locks up the respective position along the ith line and computes the distance between this position and its own position. This value is then used to compute the particles transparency.

5 Rendering Aspects

The proposed techniques for particle-based flow visualization have been integrated into the GPU particle engine presented in [13]. The engine supports the visualization of vector fields given on uniform grids, and it has been further extended to visualize unsteady flows given on such grids [2]. By integrating the proposed importance-based techniques into this engine, these techniques could be validated using time-varying sequences. In this section we describe the necessary extensions of the GPU engine to allow for importance driven particle rendering.

5.1 Particle morphing

Rendering of oriented point sprites of different size, shape and appearance is accomplished by extending the concept of a 2D texture atlas [8, 13]. Such an atlas contains a 2D array of different views of a 3D particle primitive. Views are parameterized with respect to

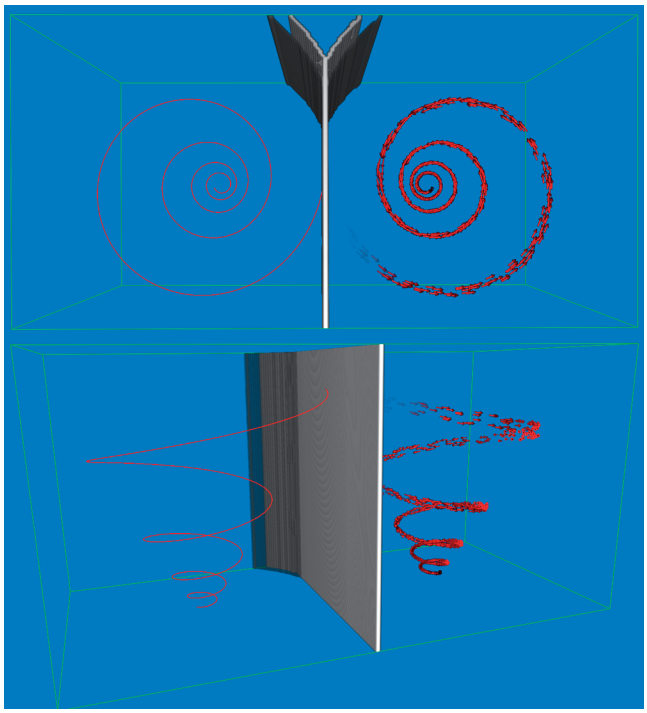

Figure 6: Anchor lines placed in regions of high Lyapunov exponent can effectively emphasize the dual vortex structure of the flow.

scaling and rotation around an axis orthogonal to the viewing axis. To support differently shaped primitives we build multiple of these atlases, each of which contains pre-computed images of a particular primitive. Each atlas is stored in a single slice of a 3D texture.

To continuously morph from one primitive into another one, we interpolate between the respective views of both primitives using 3D texture interpolation. Such an image-based blending between the same view of two different primitives is shown in Figure 7. The color and transparency of the interpolated views can be further modulated using the transfer function described in Chapter 3. It should be mentioned here, that the proposed technique can only be used if the views of two primitives that are morphed into each other are stored in successive 3D texture slices. As in the current approach the morphing is always in a strict order, this requirement does not pose any limitations.

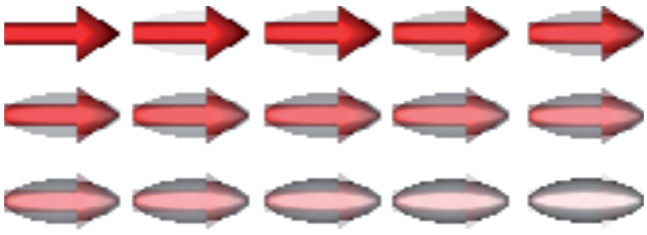

Figure 7: Image-based morphing from an arrow into an ellipsoid.

Although it is obvious that the proposed technique yields different results compared to geometry-based shape morphing and the construction of an atlas using the transformed geometry, our approach did not result in any noticeable artifacts. The reason therefore is, that particles are usually rendered as oriented primitives which show a very similar basic shape. The problem is further alleviated because we first blend between two views then perform the scaling of the result to the adequate size of the primitives. It is clear, on the other hand, that we can easily build separate atlases for arbitrary primitives in-between the given basic shapes and store them in a 3D texture. This will result in even more flexibility to select particular shapes and their appearance in regions that can not clearly be classified in term of importance and unimportance.

5.2 Blending

A critical aspect in the presented particle-based techniques is the use of transparency to visualize individual primitives. If particles are rendered as transparent sprites the order of their rendering becomes important. To guarantee a correct back-to-front or front-to-back order with respect to the viewer we use GPU sorting as proposed in [13]. The sorting algorithm essentially re-organizes the set of particles in such a way that they can be rendered in the order they are stored in local GPU memory.

As for a reasonable number of primitives sorting can quickly become the performance bottleneck we also provide an additional rendering mode that entirely avoids sorting. This mode is inspired by the observation that in a typical interactive exploration session the majority of particles is assigned very high or very low transparency, either manually by focusing on a particular region or automatically by the feature-based criterion proposed.

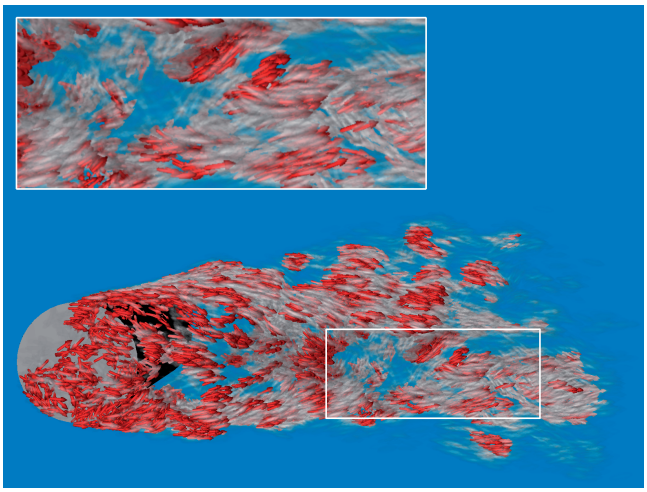

Figure 8: Rendering transparent particle primitives without sorting does not impose serious problems in the current scenario.

The approach is similar to the standard approach used to render opaque and transparent objects in that first all the opaque particles are rendered, and in a second pass the remaining transparent particles are blended into the color buffer. In the first pass, the depth test is enabled and the depth value of a fragment surviving the depth test is written into the depth buffer. In the second pass, writing to the depth buffer is disabled, and transparent particles are rendered in the order they are stored in GPU memory. To avoid brightness saturation as it is typically observed if accumulative blending is used, fragments are blended into the color buffer using alpha-compositing. Figure 8 demonstrates that the proposed rendering of opaque and transparent particles does not impose any noticeable artifacts, even though the visibility is not resolved correctly.

6 Results and Performance Analysis

We have used the proposed GPU techniques for the visualization of a number of real-world and synthetic 3D flow fields on uniform grids:

- *Flow around a box*: Result of a 3D time-dependent simulation of an incompressible turbulent flow around a square cylinder at Re = 22.000. The simulation was performed using a spectro-consistent discretization of the Navier-Stokes equations [29]. The simulation was carried out on a rectilinear grid of size 256×448×64.

- *Flow around a cylinder*: Large eddy simulation of an incompressible unsteady turbulent flow around a wall-mounted finite cylinder at Re = 200.000 [5]. 22 time steps were simulated. The size of the data grid is 256×128×128.

- *Kármán vortex street*: Result of a 3D simulation of an incompressible unsteady flow over an immersed thin cuboid obstacle at Re = 100. The simulation was performed via numerical solution of the Navier-Stokes equations according to [7]. The data set contains 30 time steps, each of which is of size 256×64×64.

- *Double-vortex flow*: A steady axisymmetric flow with two counter rotating vortices, which was computed using the following analytical expression for velocity $\mathbf{F} = (F_x, F_y, F_z)$:

$$\begin{aligned} F_x &= (-y + 0.5) + (0.5 - 2.0 \cdot x)/10.0 \\ F_y &= (2.0 \cdot x - 0.5) + (0.5 - y)/10.0 \\ F_z &= -z/10.0. \end{aligned}$$

  The computed velocity field corresponds to a spiral-like flow along the z-axis with the velocity magnitude decreasing towards the main axis of the spiral. The velocity field was mirrored to obtain the two symmetric vortices.

To validate the effectiveness of the proposed techniques, in Figures 9 and 10 we show additional visualizations of the described data sets using different importance-based visualization methods. With respect to the generated images we should note here, that the benefits of particle-based flow visualization can best be perceived in an animation. In a still image, oriented particles can show the direction of the flow quite clearly, but in contrast to LIC, for example, coherent particle trajectories can hardly be observed.

All of our tests were run on a dual core Core2 Duo 6600 equipped with a NVIDIA Geforce 8800 GTX graphics card with 786 MB local video memory. In terms of performance it can be observed that on recent GPUs the particle advection step only consumes a negligible fraction of the overall time. For instance, in a steady field about 100 millions of particles can be integrated per second using an embedded RK3(2) scheme on our target architecture. Although this rate drops significantly in the unsteady case, where streaming the time steps consumes most of the time, we can still trace about 20 millions of particles per second in a time-varying flow field of size 256×256×256. For more detailed timings the reader is referred to [2].

The performance of the technique thus strongly depends on the number and the size of the rendered particles. In particular, as soon as many large particles are rendered the application quickly becomes raster bound and the overall performance can decrease considerably. On the other hand, as the proposed importance-driven approaches can effectively reduce the amount of rendered particles, in none of our experiments the performance dropped below 100 fps.

7 Conclusion

In this paper, we have presented particle-based visualization techniques for 3D flow fields. These techniques incorporate a number of importance measures to enable an improved visual analysis of the flow. The user controls the appearance of the visualization by a few parameters such as the size and location of a focus region, weights for the context region, and the size, shape and transparency

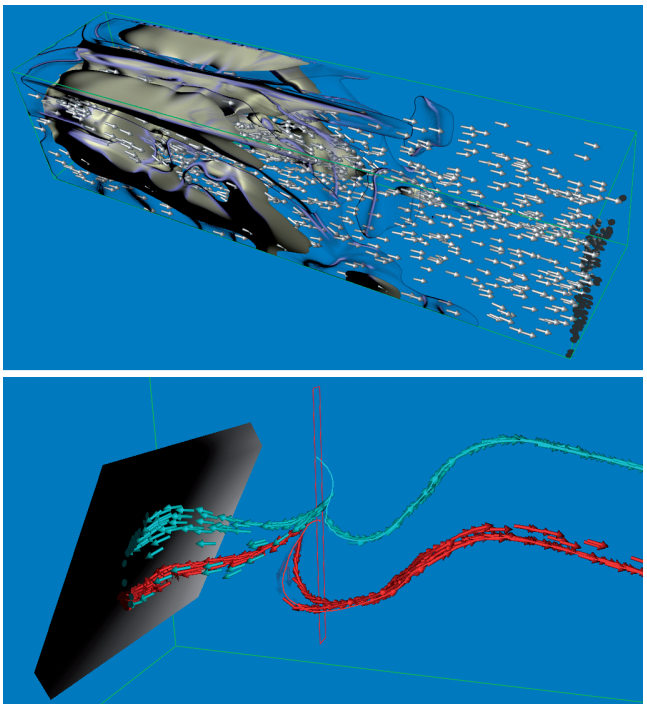

Figure 9: The top image depicts an iso-surface of the FTLE in the Kármán vortex street. The visualization of this data set using anchor lines is shown in the bottom image. Two anchor lines have been placed in the region of high FTLE. From the particle distribution one can see where particles start to separate from their anchor, and the transparency coding shows how fast they separate.

of particles traced through the flow. In addition, feature measures that are directly derived from the flow are considered to adaptively modify the particles visual attributes. In this way, better and faster understanding of complex flow structures is supported. As the proposed techniques run at interactive rates they can provide rapid visual feedback and thus allow for an effective visual exploration of the flow. As the current implementation is restricted to flow fields on uniform sampling grids we will investigate its extension to unstructured grids in the near future. As none of our proposed algorithms inherently depends on the uniform grid structure, we expect this extension to be straightforward, at least from an algorithmic point of view. However, the realization of real-time approaches for GPU-based particle tracing in unstructured grids will leave sufficient room for further research. Another important issue that will be addressed is the integration of topological features into importance-based visualizations. We believe that especially the combination of such features with anchor lines to highlight coherent structures in the flow is an interesting visualization option. Finally let us mention that the proposed techniques can effectively be used for uncertainty visualization. By simply replacing focus by certainty and context by uncertainty the proposed techniques can be used to distinguish between regions containing reliable and non-reliable information. In the future we will investigate in more detail the application of the techniques proposed in this paper for uncertainty visualization.

References

[1] R. W. Bruckschen, F. Kuester, B. Hamann, and K. I. Joy. Real-time out-of-core visualization of particle traces. In D. Breen, A. Heirich, and A. Koning, editors, *IEEE 2001 Symposium on Parallel and Large-Data Visualization and Graphics (PVG2001)*, pages 45–50, 2001.

[2] K. Bürger, J. Schneider, P. Kondratieva, and R. Westermann. Interactive visual exploration of instationary 3D-flows. In *Eurographics / IEEE VGTC Symposium on Visualization (EuroVis), to appear*, 2007.

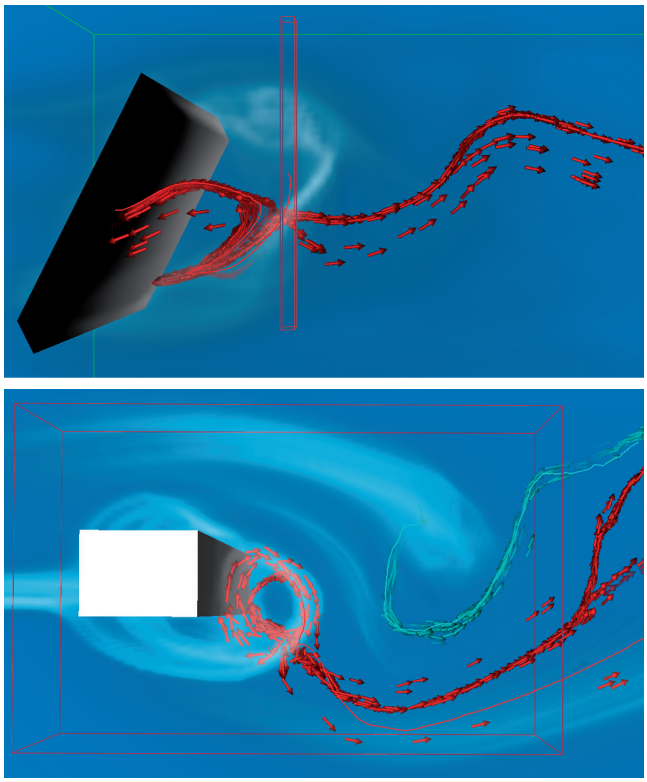

Figure 10: Top: One anchor line seeded in the region of high FTLE in the Kármán vortex street. Bottom: Two anchor lines seeded in the region of high FTLE in the flow around a box. In both images, the distribution of the Lyapunov exponent is visualized using volume rendering.

[3] B. Cabral and L. Leedom. Imaging vector fields using line integral convolution. In *Computer Graphics (Proceedings of SIGGRAPH 93)*, pages 263–270, Aug. 1993.

[4] H. Doleisch and H. Hauser. Smooth brushing for focus+context visualization of simulation data in 3D. In *Proceedings of the 10-th International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media 2002 (WSCG 2002)*, pages 147–154, 2002.

[5] O. Frederich, E. Wassen, and F. Thiele. Flow simulation around a finite cylinder on massively parallel computer architecture. In *International Conference on Parallel Computational Fluid Dynamics*, pages 85–93, 2005.

[6] A. L. Fuhrmann and E. Gröller. Real-time techniques for 3D flow visualization. In D. Ebert, H. Hagen, and H. Rushmeier, editors, *IEEE Visualization '98*, pages 305–312, 1998.

[7] M. Griebel, T. Dornseifer, and T. Neunhoeffer. *Numerical Simulation in Fluid Dynamics: a Practical Introduction*. SIAM, Philadelphia, 1998.

[8] S. Guthe, S. Gumhold, and W. Strasser. Interactive visualization of volumetric vector fields using texture based particles. In *Proceedings of WSCG*, volume 10, pages 33–41, 2002.

[9] G. Haller. Distinguished material surfaces and coherent structures in three-dimensional fluid flows. *Phys. D*, 149(4):248–277, 2001.

[10] J. Helman and L. Hesselink. Representation and display of vector field topology in fluid flow data sets. *Computer*, 22(8):27–36, Aug. 1989.

[11] V. Interrante and C. Grosch. Strategies for effectively visualizing 3D flow with volume LIC. In *IEEE Visualization 97*, pages 421–424, Nov. 1997.

[12] R. M. Kirby, H. Marmanis, and D. H. Laidlaw. Visualizing multivalued data from 2D incompressible flows using concepts from painting. In *IEEE Visualization 99*, pages 333–340, 1999.

[13] 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, Nov. 2005.

[14] 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*, 23(2):203–221, 2004.

[15] R. S. Laramee, H. Hauser, L. Zhao, and F. H. Post. Topology-based flow visualization, the state of the art. In H. Hauser, H. Hagen, and H. Theisel, editors, *Topology-Based Methods in Visualization*, pages 1–20. Springer Verlag, Mathematics and Visualization Series, 2007. (ISBN 978-3-540-70822-3).

[16] H. Löffelmann and M. E. Gröller. Enhancing the visualization of characteristic structures in dynamical systems. In *Proceedings of the 9th Eurographics Workshop on Visualization in Scientific Computing*, pages 35–46, 1998.

[17] O. Mattausch, T. Theußl, H. Hauser, and M. E. Gröller. Strategies for interactive exploration of 3D flow using evenly-spaced illuminated streamlines. In K. Joy, editor, *Proceedings of Spring Conference on Computer Graphics*, pages 213 –222. SCCG, Apr. 2003.

[18] N. Max, R. Crawfis, and C. Grant. Visualizing 3D velocity fields near contour surfaces. In *IEEE Visualization 94*, pages 248–255, 1994.

[19] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch. Feature extraction and visualisation of flow fields. In D. Fellner and R. Scopigno, editors, *Eurographics 2002 State of the Art Reports*, pages 69–100. The Eurographics Association, Saarbrücken Germany, Sept. 2002.

[20] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch. The state of the art in flow visualisation: Feature extraction and tracking. *Computer Graphics Forum*, 22(4):775–792, Dec. 2003.

[21] C. Rezk-Salama, P. Hastreiter, C. Teitzel, and T. Ertl. Interactive exploration of volume line integral convolution based on 3D–texture mapping. In *IEEE Visualization 99*, pages 233–240, 1999.

[22] F. Sadlo and R. Peikert. Efficient visualization of Lagrangian coherent structures by filtered AMR ridge extraction. *To appear: IEEE Transaction on Visualization and Graphics*, 2007.

[23] M. Schirski, A. Gerndt, T. van Reimersdahl, T. Kuhlen, P. Adomeit, O. Lang, S. Pischinger, and C. H. Bischof. Vista flowlib: A framework for interactive visualization and exploration of unsteady flows in virtual environments. In *7th International Workshop on Immersive Projection Technology, 9th Eurographics Workshop on Virtual Enviroments*, pages 77–86. Eurographics Association, 2003.

[24] S. C. Shaddena, F. Lekienb, and J. E. Marsden. Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3–4):271–304, December 2005.

[25] H.-W. Shen, G.-S. Li, and U. D. Bordoloi. Interactive visualization of three-dimensional vector fields with flexible appearance control. *IEEE Transactions on Visualization and Computer Graphics*, 10(4):434–445, July 2004.

[26] D. Stalling and H.-C. Hege. Fast and resolution-independent line integral convolution. In *Computer Graphics (Proceedings of SIGGRAPH 95)*, pages 249–256, Aug. 1995.

[27] J. J. van Wijk. Spot noise-texture synthesis for data visualization. In *Computer Graphics (Proceedings of SIGGRAPH 91)*, volume 25, pages 309–318, 1991.

[28] J. J. van Wijk. Flow visualization with surface particles. *IEEE Computer Graphics and Applications*, 13(4):18–24, July 1993.

[29] R. Verstappen and A. Veldman. Spectro-consistent discretization of Navier-Stokes: a challenge to RANS and LES. *Journal of Engineering Mathematics*, 34(1):163–179, 1998.

[30] I. Viola, M. E. Gröller, K. Bühler, M. Hadwiger, B. Preim, D. Ebert, M. C. Sousa, and D. Stredney. Illustrative visualization. IEEE Visualization Tutorial on Illustrative Visualization, 2005.

[31] D. Weiskopf, T. Schafhitzel, and T. Ertl. Real-time advection and volumetric illumination for the visualization of 3D unsteady flow. In *Eurographics/IEEE VGTC Symposium on Visualization (EuroVis)*, pages 13–20, 2005.

[32] X. Yuan and B. Chen. Illustrating Surfaces in Volume. In *Proceedings of VisSym'04, Joint IEEE/EG Symposium on Visualization (Konstanz, Germany, May 19–21, 2004)*, pages 9–16, 337, 2004.