Global, Geometric, and Feature-Based Techniques for Vector Field Visualization ⋆

Frits H. Post [a] Wim C. de Leeuw [b] I. Ari Sadarjoen [a]
Freek Reinders [a] Theo van Walsum [c]

[a] *Computer Science Department, Delft University of Technology*
[b] *Centre for Mathematics and Computer Science CWI, Amsterdam*
[c] *Image Sciences Institute, Utrecht University*

Abstract

Vector field visualization techniques are subdivided into three categories: global, geometric, and feature-based techniques. We describe each category, and we present some related work and an example in each category from our own recent research. Spot Noise is a texture synthesis technique for global visualization of vector fields on 2D surfaces. Deformable surfaces is a generic technique for extraction and visualization of geometric objects (surfaces or volumes) in 3D data fields. Selective and iconic visualization is an approach that extracts important regions or structures from large data sets, calculates high-level attributes, and visualizes the features using parameterized iconic objects. It is argued that for vector fields a range of visualization techniques are needed to fulfill the needs of the application.

1 Introduction

Visualization plays an important role in the process of computational science: analysis of data sets that result from computer simulations generally requires some visualization technique. By applying visualization techniques to their data sets, scientists and engineers can literally see the results of the computations, and draw conclusions from it. With the continuing rapid increase of computer power, the complexity and size of numerical simulations increases. Due to these developments the need for appropriate visualization techniques has also increased: it is hardly possible to analyze Gbytes of data without proper visualization tools.

⋆ Color pictures available. See `http://www.elsevier.nl/locate/future`

The challenge for researchers in the field of visualization is to develop techniques that allow a scientist or engineer to extract the information from a large data set. Spatial domains of computer simulations have three dimensions, and with dynamic phenomena time is added as a fourth dimension. Several physical quantities are often computed for each grid node and time step, consisting of scalar, vector, and tensor data. Visualization has to deal with this high dimensionality of the data, as the visual display is usually a 2D screen that can display 3D scenes and animations. Especially higher-order data, such as vector and tensor fields, are a great challenge to scientific visualization, as there are no simple ways to directly visualize these high-dimensional data spaces.

In this paper we will focus on visualization of 3D vector fields, mainly resulting from computational fluid dynamics (CFD) simulations. Vector fields (in particular velocity fields) are the main data type resulting from CFD simulations, but often the data sets also contain other types of quantities. A good classification of existing (and future) types of field visualization can be found in [4]. Here we will first present a threefold classification of visualization techniques that has proved useful, and then present some results of our recent research in each category.

- *Global techniques*: qualitative visualization of a complete data set, or a large subset of it, at a low level of abstraction. A global view is most useful in the early stages of exploration to get a general impression; a local view can be used later to focus on interesting details. The mapping of data to a visual representation is direct, without complex conversion or extraction steps. These techniques include color mapping and direct volume rendering for scalar data fields, and hedgehogs (arrow plots) and texture-based visualization for vector fields. As an example of the latter Section 2 describes Spot Noise, a stochastic texture generation technique.
- *Geometric techniques*: extraction and visualization of geometric objects (curves, surfaces, solids) of which the shape is fully and directly related to the data. Usually, a specific geometry extraction step is needed to generate the geometric objects. This type can be considered as an intermediate-level representation, both in locality and abstraction level. Examples are iso-curves and iso-surfaces for scalar fields, and curves and surfaces (streamlines and stream surfaces) for velocity fields. Section 3 presents a new surface extraction technique based on adaptively deformable surfaces.
- *Feature extraction*: large numerical data fields are low-level representations of physical phenomena, and often they contain only a limited amount of relevant information. Thus, it can be useful to extract the high-level information from these data. Features are high-level, abstract entities in a data set, represented by quantitative attributes. In feature extraction, emphasis is on quantification [17], or retrieving high-level quantitative measures for more precise evaluation and comparison. Visualization of features can be done by parameterized iconic objects such as ellipsoids, tubes, or glyphs [19].

The feature attributes are linked to the object's parameters. Examples in this category are vector and tensor field topology, detection of vortices and shock waves in flow fields, and detection of arteries in angiographic images. Section 4 describes the process of selective and iconic visualization as an example of this approach to visualization.

Each of the visualization techniques that is discussed in the following sections will be accompanied by an example, which will show how the technique can be applied effectively.

2 Global techniques: Spot Noise

Global visualization of a flow field presents a number of problems: a continuum has to be represented, structures might occur at various levels of detail and the dimensionality of the data is high: a 3D domain consisting of vector data. Traditionally, color maps, and arrow plots (hedgehogs) are used as global visualization technique. Some drawbacks of arrow plots are cluttering, depth ambiguity, and failure to represent the dynamic nature of a flow. The use of texture for flow visualization overcomes these drawbacks; texture shows the field as a continuum, and the dynamic behavior of the fluid can be shown by animation. Furthermore, a scalar value can be mapped on the texture showing additional data.

In this section, we present the *Spot Noise* technique, a global texture-based visualization technique. We also discuss some extensions of this technique that are focused on interactive display of time-dependent flow data.

2.1 Related work

Several researchers have reported on the use of texture for flow visualization [23,2,3]. In two cases, stochastic texture is used to represent data: Line integral convolution (LIC) [2] and Spot Noise [23]. The visualization principle for both techniques is similar [9]: data is represented by the variation of intensity, and the value of an isolated pixel is essentially random. Both techniques produce textures in which intensity changes are slow in the direction of the flow and fast in the direction perpendicular to the flow, resulting in a striped pattern. The pattern is similar to photographs in which particles are blurred due to a long exposure time.

The difference between the methods is the algorithm used to synthesize the texture. In LIC a random noise image is filtered using a convolution filter

which has the shape of the streamline through the pixel. At each pixel in the texture, a streamline is calculated forward and backward for a certain length. The pixel values from the input image along this streamline are convolved using a one dimensional filter kernel. The output of this convolution is used as the value of the pixel in the texture. The Spot Noise technique is described below. A more in-depth comparison on both techniques can be found in [9].

2.2 Spot Noise

2.2.1 Basic technique

A Spot Noise texture is the sum of a large number of small intensity functions called spots. Because a large number of spots are used, the individual spots can no longer be distinguished, and a texture is perceived instead (Figure 1). Mathematically a texture can be characterized by a scalar function f

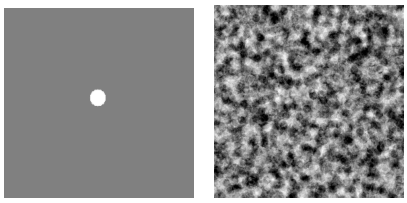


Fig. 1. Principle of Spot Noise: single spot (left) resulting texture (right)

of position $\mathbf{x}$. A Spot Noise texture is defined as;

$$f(\mathbf{x}) = \sum a_i h(\mathbf{x} - \mathbf{x}_i) \tag{1}$$

in which $h(\mathbf{x})$ is the spot function. It is a function everywhere zero except for an area that is small compared to the texture size; usually a disc shape is used. a_i is a random scaling factor with zero mean, $\mathbf{x}_i$ is a random position.

Spot Noise can be used for visualization by locally adapting the shape of the spot. Flow fields can be displayed by scaling spots proportional to the magnitude of the flow and aligning the spot with the direction of the flow. In [10] bent spots were introduced which allow Spot Noise to be used in areas where the curvature of the vector field is high. Instead of using a scaling and rotation only, bent spots are mapped on to a surface generated by advecting streamlines in the flow. This method greatly enhances the quality of the texture at the cost of an increase in the computation needed.

The dynamic behavior of flow can be displayed via an animated sequence of Spot Noise images. A Spot Noise animation of a stationary flow field can be realized by associating a particle with each spot position. The next frame in

the animation sequence is determined by advecting all particles one time step through the flow field.

2.2.2 Zooming

Accurate simulation of flow at high Reynolds numbers requires very high resolution grids, containing millions of cells. At such resolutions the visualization of a slice of data becomes a problem. The difference in size between the smallest details and the whole set is about three orders of magnitude. If a complete slice is shown the smallest detail is smaller than a single pixel on screen. Zooming is essential to inspect data sets of this size. In this section we will show how zooming can be combined with Spot Noise.

The appearance of the final Spot Noise texture is based on the size and distribution of the spots. Choosing small spots will result in a more fine-grained texture which shows flow patterns at a smaller scale. However, choosing a spot size that is too small compared to the resolution of the underlying texture results in aliasing. In [23] it was shown that characteristics of the texture are captured by the power spectrum $P_f\ (\omega)$ of a stochastic function $f(t)$ by:

$$P_f\ (\omega) \ = \ \lim_{T \to \infty} \frac{1}{T} |F_T\ (\omega)|^2 \tag{2}$$

in which $F_T\ (\omega)$ is the Fourier transform of a sample of $f(t)$ with length T. Characteristics of a spot (such as size, shape and direction) directly influence the characteristics of the texture (such as granularity and isotropy).

Since the spatial resolution of the texture is limited, zooming in will result in relatively large texture elements in the image on the screen. We therefore need a multi-resolution scheme which allows both global and detailed views of the data, without compromising on the texture quality on the screen. Our solution is to ensure that the power spectrum of the texture visible on the screen does not change during zooming. This is achieved by two mechanisms. First, the spot size is reduced while zooming in on detail. Second, the region of the data which generates the texture image is reduced while zooming in. The full texture image is dedicated to this region. Also, the size of the spot, which is linked to the surface area of the region of interest is adapted automatically.

2.2.3 Time-dependent Spot Noise

Animated textures for steady flows can be achieved using advection of spots along streamlines. The generation of textures in unsteady flow is more complicated. The pattern in a single Spot Noise texture results from the spatial coherence of pixel values across the texture. Animation is based on temporal

coherence of spot positions. The intensity changes of the texture in successive texture frames must be such that the pixels in the texture seem to move with the flow.

In general, a particle path is calculated by integration:

$$\mathbf{x}(t) = \mathbf{x}(t_0) \; + \; \int_{t_0}^{t} \mathbf{v}(\mathbf{x}(t), t) \; dt \tag{3}$$

where $\mathbf{x}(t)$ is the particle position at time t, $\mathbf{v}(\mathbf{x}(t), t)$ is the velocity of the particle at time t, and $\mathbf{x}(t_0)$ is the initial release position of the particle.

In a stationary flow, where $\mathbf{v}$ is independent of time, temporal coherence is obtained by particle advection and cyclic display. Between successive frames, the spots are advected as particles. The intensity of a spot is modulated between zero at its initial position, a maximum value and zero again at its final position. The length of the life cycle is equal for all spots, but the phase of the life cycle varies randomly over the spots. A small number of frames (typically less than 10) is generated and played back in a closed loop, which gives excellent results in animating sequences of stationary flow [23].

For time dependent flow, where $\mathbf{v}$ varies with time, temporal coherence is again obtained if particle paths are used to advect the spots.

Spots are regarded again as particles, and their positions on the motion path are calculated accordingly. However, in this case cyclic animation is not possible. Also, as spots are advected over time, the distribution of spots will not remain uniform. To maintain a uniform spot distribution, the previously mentioned life cycle mechanism is extended. Spots are advected from a set of initial positions. When a spot reaches the end of its life cycle, it will start a new cycle from its initial position. A life cycle has a typical length of about 15 frames. The distance covered by a spot during this period is sufficiently small to assure a uniform distribution of spots in the texture.

2.3 Application: direct numerical simulation of a turbulent flow.

In [20], methods are discussed for direct numerical simulation of turbulent flow. The goal of this application is to study the evolution of vortex shedding behind a block, and the transition from laminar to turbulent flow. Using Spot Noise the data can be interactively analyzed.

Figure 2 shows a snapshot of a slice of the 3D data set. The flow is from left to right and impinges on a block placed in the field. One can clearly see the

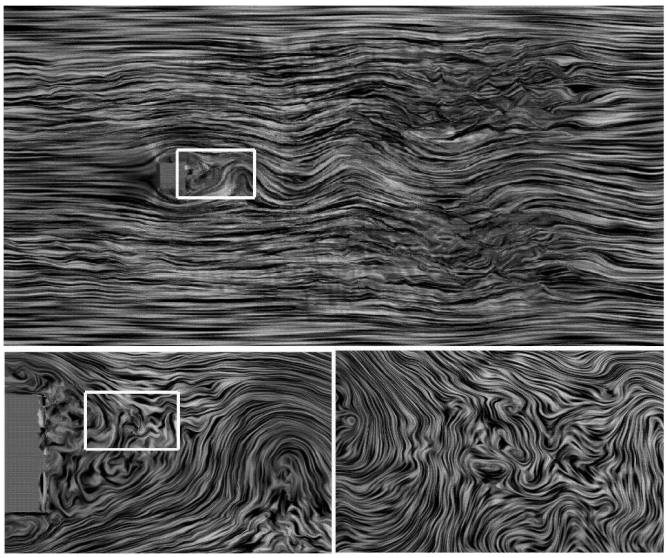


Fig. 2. Top: direct numerical simulation of the wake behind a block, showing vortex shedding and transition from laminar to turbulent flow. Bottom: two detailed views from the upper image.

transition from laminar to turbulent flow behind the block. The data is defined on a rectilinear grid with a resolution of 278×208 cells. Each texture has a resolution of 512×512 pixels and uses 40,000 spots. Bent spots were used because of the turbulent nature of the flow in which strong fluctuations for the flow magnitude and direction occur. The lower part of Figure 2 illustrates the zooming mechanism by two detailed snapshots of the data. The left image is a detail of the wake behind the block showing a region of 128×70 cells. In the right image a detail of 22×14 cells is shown. These details give insight in the generation of vortices in the wake of the block.

3 Geometric techniques: deformable surfaces

Global techniques, as described in the previous section, visualize a large quantity of data simultaneously, and let the scientist extract meaningful information from the presented data. Geometric techniques are based on preprocessing of the data, in order to present relations between (or within) the data more directly. This preprocessing results in geometric objects, such as curves, surfaces, or solids.

3.1 Related work

There exists a myriad of geometric techniques, which can be classified based on the type of fields they work on, or on the type of geometries they produce.

For scalar fields, iso-surfaces are a frequently used technique [11]. For flow fields, there is a large group of algorithms that extract all types of flow curves, such as streamlines, path lines, streak lines, and time lines. Other algorithms extract flow surfaces, such as stream surfaces [7] or shock waves [13]. These are specific techniques that generate specific types of surfaces, and that build geometries in a single step.

The deformable surfaces described below are meant to be more generic, as they work with different types of fields and can extract different types of geometry [16]. For example, using different application-specific criteria, we have extracted iso-surfaces, separation surfaces, and vortices with one single technique. Furthermore, the technique belongs to a class of algorithms that uses iterative curves or surfaces. These curves or surfaces start from an initial shape which is deformed adaptively in many steps, until the curve or surface has reached the desired shape. They are often applied for segmentation of images or 3D data sets scanned using CT/MRI/PET techniques [8,12].

3.2 Deformable surfaces

The process of generating deformable surfaces consists of three stages: region selection, initial surface creation, and surface deformation. These stages are described below.

3.2.1 Region selection

Starting from an input data set, a region of interest is selected using selection techniques, which are described in more detail in Section 4. Basically, a selection criterion selects the nodes from a grid which satisfy user-specified conditions. Next, adjacent selected nodes are grouped into clusters, each representing a coherent selected region that can be treated as a unit. From these clusters, one or more can be chosen to be used as the region of interest.

3.2.2 Initial surface creation

The initial surface should have a shape related to the selected region. This shape might be found by sophisticated shape classification algorithms, such as skeleton extraction. An alternative is to fit an ellipsoid as a rough approximation of the selected region. In this case, statistical attributes of the selected points are calculated: center, variance and covariance [19]. These attributes define an ellipsoid, of which the center position (x, y, z), the lengths (s_1, s_2, s_3) of the main axes, and three angles (α, β, γ) are used to characterize the position, shape type, and orientation of the selected region. The ellipsoid axis

lengths determine the shape of a region to be 1D, 2D, or 3D, where a 1D shape may correspond to a cylindrical tube, a 2D shape to a planar surface, and a 3D shape to a spherical volume. The ellipsoid's position, dimensions, and orientation determine the corresponding parameters of the initial surface. Once the initial surface has been created, it can be deformed to its final shape.

3.2.3 Surface deformation

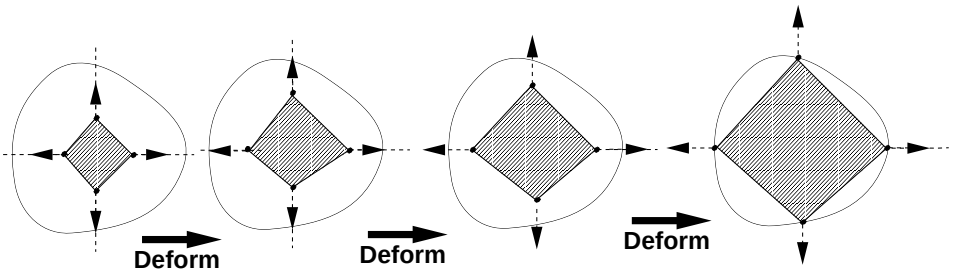
The polygonal surface is locally deformed by displacing the polygon vertices or nodes, using a node displacement criterion to govern the fitting of the surface to the data field. If the displacement direction of the nodes is outward from the center, the object grows, and polygon size increases. Whenever faces become too large they are refined for a better shape approximation.

Node displacement is guided by a displacement criterion, which uses a cost function C based on the physical quantities given in the data set. These quantities may be scalars, vectors, or a combination. The criterion tries to reduce the value of the cost function below a specified tolerance ϵ. In any case, the result is always a displacement step, which has two characteristics: a direction $\mathbf{d}$ and a size λ. The step direction $\mathbf{d}$ may be determined in several ways: by performing a random search in several directions, by following the field gradient, or by following the surface normal. The last method turned out to work best, because it made the surface grow uniformly in all directions. The step size λ may be determined in several ways, which result either in the object growing in multiple small steps (see Figure 3a), or in a single large step (see Figure 3b). Depending on the situation, both methods have their merits.

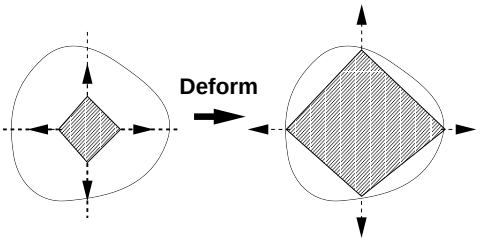
If the faces of a deformable surface become too large, they can be adaptively refined. By adaptive we mean that only those faces f are refined for which some refinement criterion $R(f)$ exceeds a specified threshold θ. The refinement is performed using a scheme based on Miller [12]. Figure 4 illustrates how a triangular face (in grey) is subdivided into four triangles at the edge midpoints, the three adjacent faces (in white) are subdivided into two triangles, to prevent hanging nodes.

3.3 Application: extracting vortex tubes

We have applied the deformable surface method for extracting vortex tubes from numerically simulated flows. These are important in many engineering applications, such as the design of aircraft. Since it is difficult to give a formal definition of vortices, we use the following informal definition: a vortex is a swirling flow pattern which will often behave as a coherent structure. The geometry to be extracted is an elongated tube, in which all fluid rotates in



(a) In the many-step method, surface points are displaced using small steps at a time



(b) In the one-step method, surface points are placed on the target surface in one step

Fig. 3. Many-step and one-step methods for iteration

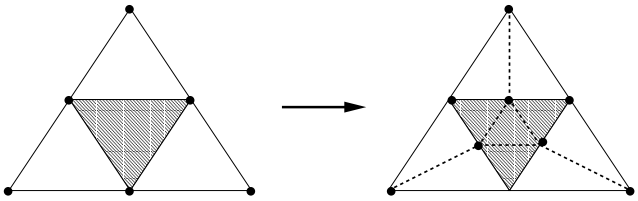


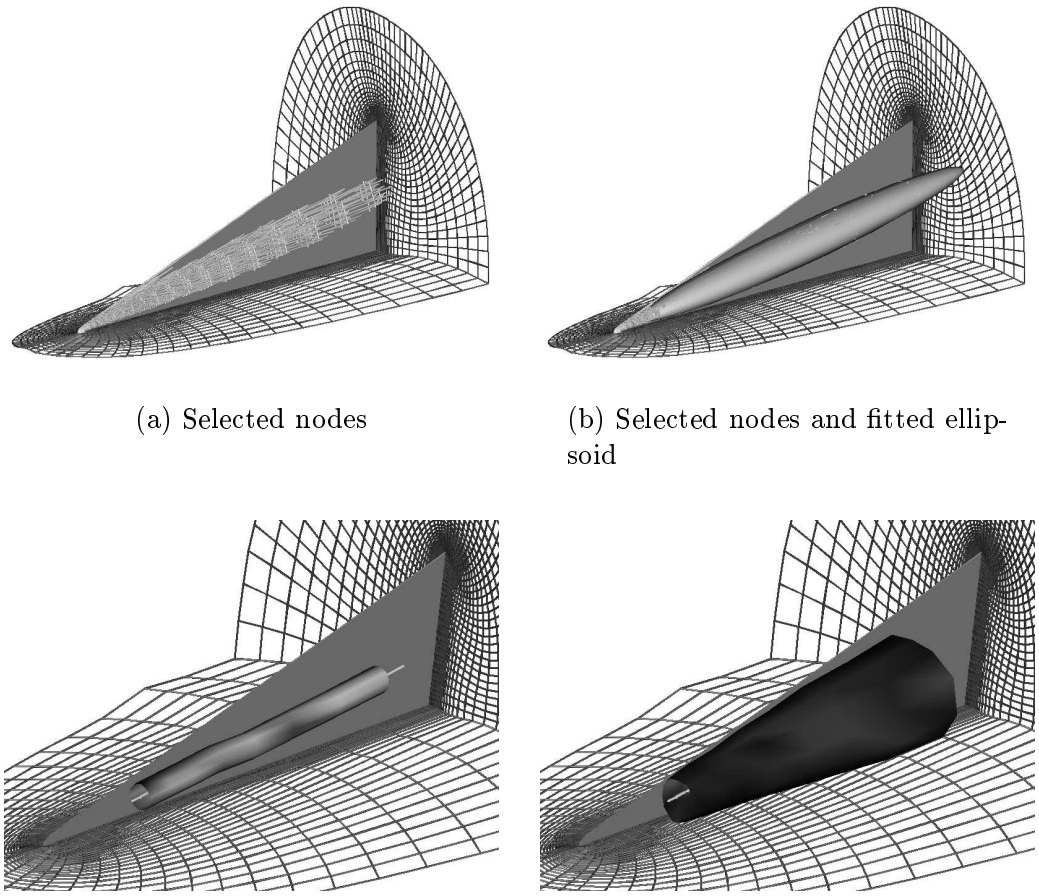
Fig. 4. Face refinement. The grey face is subdivided into 4 faces, the adjacent faces into 2.

one direction. Often, this tube is constructed around a vortex core, which is extracted first. Examples of vortex core and vortex tube algorithms can be found in [1,15,21].

The data set used here is the Delta-Wing data set available from NASA [5]. It models the flow around a delta wing aircraft at a high angle of attack (40 degrees). The original files contain: density, momentum, and stagnation, which are defined on a 56x54x70 structured curvilinear grid. The features in this data set include vortices on one side of the wing, from which we wish to extract one vortex tube using the deformable surface technique.

Region selection criteria are used to find the axis and shape of a vortex feature, attributes which are then used to initialize a deformable surface. Then,

deformation criteria are used to make the surface grow, and to find the outside boundary of the vortex. We used criteria based on vector quantities. Following the scheme in Section 3.2, the following steps were performed:



(a) Selected nodes

(b) Selected nodes and fitted ellipsoid

(c) Initial vortex tube

(d) Final vortex tube

Fig. 5. Extraction of a vortex tube with vector deformation criteria. Color indicates the value of the cost function C (red = high, blue = low).

(1) Region selection: the vortex tube extraction uses the pressure field p given in the data set, as well as additional derived quantities: the vorticity vector field $\boldsymbol{\omega}$ and its magnitude (ω). These were also used in [1,21]. The criteria used for the initialization are low pressure and high vorticity magnitude: $p < c_1 \cdot \min(p)$ and $\omega > c_2 \cdot \max(\omega)$ where c_1 and c_2 are user-definable parameters ($0 < c_1, c_2 < 1$). The selected nodes are shown as cross-marks in Figure 5a.
(2) Initial surface creation: in the selected region, statistical attributes are calculated which define the ellipsoid shown in Figure 5b. The ratios of the ellipsoid axes result in a 1D object, the cylinder shown in Figure 5c.
(3) Surface deformation: the deformation criterion uses the angle α between the local vorticity and the vorticity $\boldsymbol{\omega}_\mathbf{c}$ at the vortex core, as shown in

Figure 6. At the core, this angle is 0 by definition. At a surface node $\mathbf{n}$, the vorticity is $\boldsymbol{\omega}_\mathbf{s}$ and the angle $\alpha = \angle(\boldsymbol{\omega}_\mathbf{s}, \boldsymbol{\omega}_\mathbf{c})$.

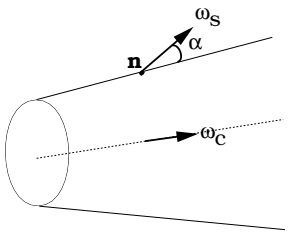


Fig. 6. Vortex-finding with vector criteria

The criterion makes the surface grow in the direction where this angle is 90 degrees, or maximal, by defining the cost function to be minimized as the cosine of the angle: $C(\mathbf{x}_n) = \cos(\angle(\boldsymbol{\omega}_\mathbf{s}, \boldsymbol{\omega}_\mathbf{c}))$.

Figure 5d shows the final vortex tube after 16 iterations with the many-step iteration method. The surface consists of 260 triangles.

4 Feature extraction: selective and iconic visualization

The global techniques leave the extraction of meaningful features to the user. The geometric techniques are an approach to extract and present geometric objects of which the shape is directly related to the data. Both types of visualization leave the extraction of interesting phenomena to the visual inspection by the scientist. Feature extraction techniques, as described in this section, are an approach to extract these interesting phenomena in an algorithmic way, and to obtain quantitative measures of these features.

4.1 Related work

The purpose of feature extraction techniques is to find interesting features in the data more or less automatically, and to determine quantitative characteristics of the features, the so-called attributes. For visualization purposes, the attributes can be mapped onto the parameters of an icon [19], visualizing the feature in an abstract way: instead of 'a visualization with a certain feature', the feature itself is visualized. In this way, feature extraction induces three important advantages: non-relevant data is filtered out, a quantitative measure is obtained, and a huge data reduction is achieved.

Currently, many application-specific feature extraction techniques exist, as the definition what is to be considered an 'interesting feature' differs for each

application. For flow visualization, extraction techniques exist for the visualization of flow field topology [6], vortex tubes [1], and shock waves [14]. All these examples are techniques that extract the occurrence of a particular phenomenon. Below, we will present a more general process of feature extraction. The method is based on the selective visualization method [22], and is best described by a pipeline model (see Figure 7), with the following steps: selection, clustering, attribute calculation, and iconic mapping.

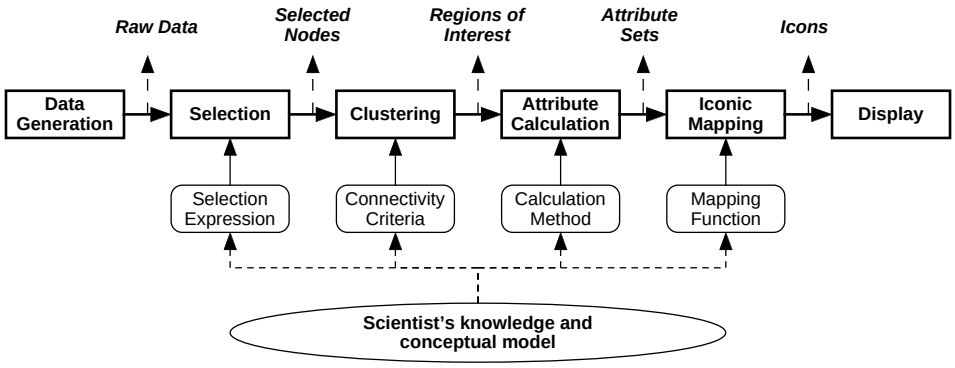


Fig. 7. The selective and iconic visualization pipeline.

4.2 Selective and iconic visualization

4.2.1 Selection

The first step in selective and iconic visualization is the selection of the data points (grid nodes) that belong to the feature. We use the term "feature" for any region or node in the data that is of special interest, and assume that a feature can be distinguished from the rest of the data in some way. Then, a selection of grid nodes can be made that corresponds to the criteria of interest specified by the scientist. The criteria of interest are expressed in a selection expression, which is a mathematical formulation of the underlying physics of the feature. The expression can combine multiple thresholds on raw data or on derived quantities, its evaluation yields TRUE or FALSE, which determines if a node is selected or not. The expression is formulated in a language with operators (boolean, scalar, vector, and matrix operators) and functions (gradient, and statistical functions). With this language almost any selection, depending on the application, can be created based on the data values, and on the node positions.

4.2.2 Clustering

The next step is the clustering of the selected nodes. Individual selected nodes have no other meaning than an indication of the positions where the data

satisfies the criteria of interest. However, our concern are coherent regions consisting of more than one node: regions of interest. Therefore, the task is to find clusters of connected nodes that are part of the selection. The clustering procedure is based on a connectivity criterion, and it assigns a label that identifies the cluster to each selected node. Each cluster is a region of interest, which is an entity instead of a group of individual selected nodes. They are treated as separate entities, and therefore we call them features.

4.2.3 Attribute calculation

Once the regions of interest have been identified, attributes that characterize the feature can be calculated. There are many possible sets of attributes, each resulting from its own calculation method. Many methods are derived from other scientific areas such as computer vision (object fitting methods) and image processing (morphological operators, skeletonization). A generic way to calculate attributes are volume integrals over the nodes of a cluster, which may result in statistical attributes, such as size, center position, spatial distribution, average data value, etc. Another example is given by Silver et al. [18], where ellipsoids are fitted to regions of interest resulting in a measure of the position, size, and orientation of a feature.

4.2.4 Iconic mapping

In order to visualize a feature, the attribute sets can be mapped on a parametric icon [19]. An icon is a geometric object with a parametric shape and visual attributes that can be linked to the attributes of a feature. The relation between the parameters of the icon and the attributes is called the mapping function. The goal of iconic mapping is to visualize essential elements of a feature in an abstract symbolic representation which should relate to the physical concepts and visual languages of the area of application. Like the number of features, the number of different types of icons is unlimited, the design of an icon depends on the specific application area and research problem.

4.3 Application: backward-facing step

An example that illustrates the results of this visualization process, is our visualization of the flow in a backward-facing-step geometry. The inlet is at the boundary near the step (upper left side in Figure 8a), and the outlet is the other end. In such a flow, a region of spiraling flow should occur just behind the step. The goal of the visualization is to visualize streamlines that start at the inlet, then flow through the spiraling region, and finally leave that region again. This could be visualized by manually probing the dataset until

such a spiraling streamline is found. However, this is a tedious and difficult job, as there are only few locations near the inflow boundary from where streamlines show spiraling behavior. The streamlines have to pass through the spiraling flow region, therefore this region is the target of our selection. Using a selection expression based on the normalized helicity density h_n (see Table 1), two spiraling regions of interest are found. Starting forward and backward integrations from positions in these regions, streamlines that go through the spiraling regions are easily found.

Table 1
Selection expressions for selecting nodes with a helicity density larger then 0.66

Selection expression	Helicity density

`; calculate the curl (rotation)`
`rotation = rot(velocity)`
`; calculate helicity density hd`
`hd = dot(rotation, velocity)`
`; normalize helicity density hdn`
`hdn = hd / (len(velocity)*len(rotation))`
`; create the selection select_out`
`select_out = fabs(hdn) > 0.66`
	$$h_n = \frac{\mathbf{v} \cdot (\nabla \times \mathbf{v})}{\lvert\mathbf{v}\rvert\lvert\nabla \times \mathbf{v}\rvert} \quad (4)$$
	with $\mathbf{v}$ velocity, and $\nabla \times \mathbf{v}$ the curl of $\mathbf{v}$.

Figure 8 visualizes the selected nodes by small cross-marks, the two spiraling regions by ellipsoid icons, and two streamlines through the regions by tubes. The radii of the tubes are inversely proportional to the square root of the local velocity magnitude, and the color corresponds to the pressure at the centerline. The example shows two advantages of feature extraction: one, the increase in visualization productivity: manually probing the data set would have taken much longer then the few minutes it took to generate this visualization, and two, it shows the increase in visualization efficiency: only a few geometric objects are needed to show exactly what the scientist is looking for.

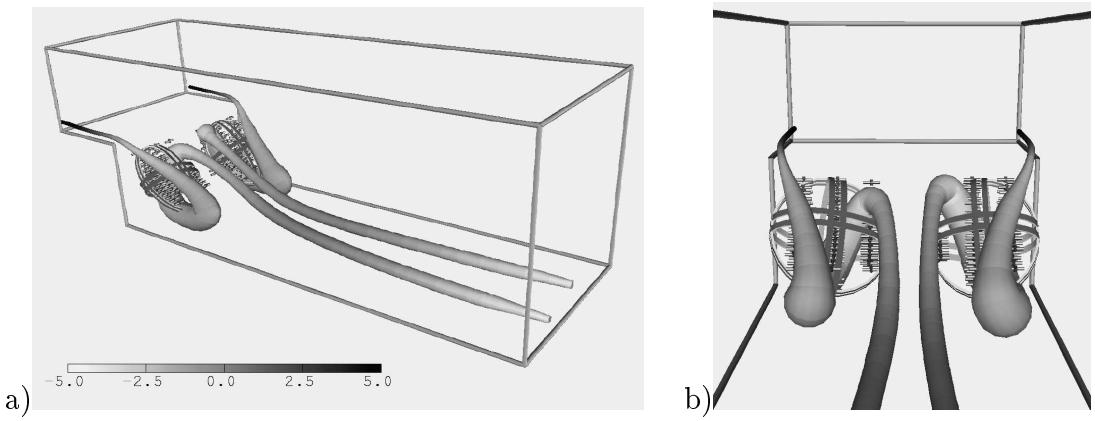


Fig. 8. The backward-facing-step spiraling flow region.

5 Conclusions

We have presented three different techniques for visualization of vector fields. The techniques each cover their own part of the spectrum of flow visualization techniques, but also combine certain aspects in their functionality.

Spot Noise can be used to give a global view of a 2D time-dependent flow field. The multi-scale Spot Noise technique that allows the scientist to zoom in on details is a good example of how the gap between global and local techniques can be bridged. Deformable surfaces can be used to extract amorphous regions of interest and phenomena from data sets. This technique also works at a more local scale than other surface extraction techniques, and uses selection techniques to initialize the deformation process. The selection and iconic visualization approach tries to find interesting regions in the data set, based on boolean expressions, and display attributes of these regions as parameterized icons.

As will be clear from the examples shown, the techniques are not interchangeable: each has its own type of applications for which it is particularly suitable. Global techniques are often close to a natural, intuitive representation, but it is difficult to visualize a solid 3D volume of vector data. Surface-based geometric techniques have the advantage that surfaces are ideally suited for visualization: the human eye is well adapted to perceiving complex shapes from reflected light. Also, color or texture can be used to carry other information, and lighted, colored, and textured surfaces can be efficiently displayed by graphics workstations. Feature-based visualization is more abstract and perhaps less intuitive, but it is closer to the high-level concepts of the scientist; also, a data reduction of up to several orders of magnitude can be achieved.

There are many techniques to choose from for the visualization of a particular CFD data set, and the number is still increasing. Yet, the number of techniques suitable for one particular application is generally rather limited. This is, in our view, characteristic for the field of flow visualization. Despite the many techniques available, vector field visualization is not yet a solved problem. In practice, there is no single universal visualization technique, but the best results will be obtained by sensible combination of different techniques.

The field still has many challenges for visualization research. We expect that 'raw data' direct visualization techniques, and extraction techniques for geometry and features, will be improved and extended. Much effort is needed in the extraction of features from a data set, for example in extending to time tracking and event detection, and comparison of different data sets based on features and quantification. Also, new tools must be developed to incorporate the domain knowledge of the scientist for developing good feature extraction

criteria, combining qualitative exploration and quantitative analysis of data, and interactive steering of simulations. The general goal is to offer the computational scientist a set of tools that covers the whole range of needs for scientific and engineering practice, by closely matching the way many scientists and engineers approach their problems.

References

[1] D.C. Banks and B.A. Singer. A predictor-corrector technique for visualizing unsteady flow. *IEEE Trans. on Visualization and Computer Graphics*, 1(2):151–163, June 1995.

[2] B. Cabral and L. Leedom. Imaging vector fields using line integral convolution. *Computer Graphics (Proc. SIGGRAPH '93)*, 27(4):263–272, August 1993.

[3] R.A. Crawfis and N.L. Max. Texture splats for 3D scalar and vector field visualization. In G.M. Nielson and R.D. Bergeron, editors, *Proceedings Visualization '93*, pages 261–265, Los Alamitos (CA), 1993. IEEE Computer Society Press.

[4] T. Delmarcelle and L. Hesselink. A unified framework for flow visualization. In R. Gallagher, editor, *Computer Visualization*. CRC Press, 1994.

[5] J.A. Ekaterinis and L.B. Schiff. Vortical flows over delta wings and numerical prediction of vortex breakdown. In *AIAA Aerospace Sciences Conference*, number 90-0102 in AIAA Paper, Reno, NV, January 1990. http://science.nas.nasa.gov/Software/DataSets.

[6] J.L. Helman and L. Hesselink. Visualization vector field topology in fluid flows. *IEEE Computer Graphics and Applications*, 11(3):36–46, 1991.

[7] J. Hultquist. Constructing stream surfaces in steady 3D vector fields. In A. E. Kaufman and G.M. Nielson, editors, *Proc. Visualization '92*, pages 171–178. IEEE Computer Society Press, 1992.

[8] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 2:321–331, 1988.

[9] W.C. de Leeuw and R. van Liere. Comparing LIC and spot noise. In *Proceedings Visualization '98*, Los Alamitos (CA), 1998. IEEE Computer Society Press.

[10] W.C. de Leeuw and J.J. van Wijk. Enhanced spot noise for vector field visualization. In G.M. Nielson and D. Silver, editors, *Proc. Visualization '95*, pages 233–239. IEEE Computer Society Press, 1995.

[11] W.E. Lorensen and H.E. Cline. Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4):163–169, July 1987.

[12] James V. Miller, David E. Breen, William E. Lorensen, Robert M. O'Bara, and Michael J. Wozny. Geometrically Deformed Models: A method for extracting closed geometric models from volume data. In *Proc. SIGGRAPH 1991*, pages 217–226. ACM, July 1991.

[13] H.-G. Pagendarm and B. Walter. Feature detection from vector quantities in a numerically simulated hypersonic flow field in combination with experimental flow visualization. In R.D. Bergeron and A.E. Kaufman, editors, *Proceedings Visualization '94*, pages 117–123. IEEE Computer Society Press, Los Alamitos, CA, 1994.

[14] H.G. Pagendarm and B. Seitz. An algorithm for detection and visualization of discontinuities in scientific data fields applied to flow data with shock waves. In P. Palamidese, editor, *Scientific Visualization: Advanced Software Techniques*, pages 161–177. Ellis Horwood Limited, 1993.

[15] M. Roth and R. Peikert. Flow visualization for turbomachinery design. In R Yagel and G.M. Nielson, editors, *Proc. Visualization '96*, pages 381–384. IEEE Computer Society Press, 1996.

[16] I.A. Sadarjoen and F.H. Post. Deformable surface techniques for field visualization. In D. Fellner and L. Szirmay-Kalos, editors, *Eurographics '97*, volume 16 of *Computer Graphics Forum*, pages C109–C116. Blackwell, Oxford, 4–8 Sep 1997.

[17] D. Silver and N.J. Zabusky. Quantifying visualizations for reduced modeling in nonlinear science: Extracting structures from data sets. *Journal of Visual Communication and Image Representation*, 4(1):46–61, March 1993.

[18] D. Silver, N.J. Zabusky, V. Fernandez, Ming Gao, and R. Samtaney. Ellipsoidal quantification of evolving phenomena. In N.M. Patrikalakis, editor, *Scientific Visualization of Natural Phenomena*, pages 573–588. 1993.

[19] T. van Walsum, F.H. Post, D. Silver, and F.J. Post. Feature extraction and iconic visualization. *Trans. on Visualization and Computer Graphics*, 2(2):111–119, 1996.

[20] R.W.C.P. Verstappen and A.E.P. Veldman. Direct numerical simulation at lower costs. *J. of Mathematical Engineering*, (32):143–159, June 1997.

[21] J. Villasenor and A. Vincent. An algorithm for space recognition and time tracking of vorticity tubes in turbulence. *CVGIP: Image Understanding*, 55:27–35, 1992.

[22] T. van Walsum and F.H. Post. Selective visualization of vector fields. In M. Dæhlen and L. Kjelldahl, editors, *Eurographics '94*, volume 13 of *Computer Graphics Forum*, pages C339–C347. Blackwell, Oxford, 12–16 Sep 1994.

[23] J.J. van Wijk. Spot noise – texture synthesis for data visualization. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 309–318, July 1991.