DOI: 10.1111/j.1467-8659.2011.01927.x

Eurographics / IEEE Symposium on Visualization 2011 (EuroVis 2011)
H. Hauser, H. Pfister, and J. J. van Wijk
(Guest Editors)

Volume 30 (2011), Number 3

Visual Reconstructability as a Quality Metric for Flow Visualization

Heike Jänicke[1], Thomas Weidner[2], David Chung[3], Robert S. Laramee[3], Peter Townsend[3], and Min Chen[3]

[1]Heidelberg University, Germany [2]Leipzig University, Germany [3]Swansea University, UK

Abstract

We present a novel approach for the evaluation of 2D flow visualizations based on the visual reconstructability of the input vector fields. According to this metric, a visualization has high quality if the underlying data can be reliably reconstructed from the image. This approach provides visualization creators with a cost-effective means to assess the quality of visualization results objectively. We present a vision-based reconstruction system for the three most commonly-used visual representations of vector fields, namely streamlines, arrow glyphs, and line integral convolution. To demonstrate the use of visual reconstructability as a quality metric, we consider a selection of vector fields obtained from numerical simulations, containing typical flow features. We apply the three types of visualization to each dataset, and compare the visualization results based on their visual reconstructability of the original vector field.

1. Introduction

The gerund "visualizing" refers to a process that extracts meaningful information from data, and constructs a visual representation of the information. This process consists of three stages, namely (i) making data displayable by a computer, (ii) transmitting visual representations to human viewers, and (iii) forming a mental picture about the data [BBC*05]. It is thus important to know whether or not the mental picture of the data established by a human viewer is consistent with the original data, and whether or not one specific visualization technique or parameter setting is more effective than another.

Traditionally, we compare visualization techniques or parameter settings by carrying out user studies. As the current state of the art in cognitive science does not allow us to model human perception and cognition algorithmically, user studies are no doubt the most scientific approach for such a comparative evaluation. However, it is not practical to conduct a user study for every individual task. Hence, in the majority of cases, we rely on the visualization creator's judgment, which often reflects personal preference and can be subjective. It is highly desirable to support a visualization process by enabling visualization creators to conduct an objective evaluation using quantitative measurements.

In this work, we propose to make *visual reconstructability* as such a quantitative measurement. When a viewer attempts to make sense of the data being depicted in a visual representation, a variety of perceptual and cognitive actions take place to reconstruct the data. Such actions include but are not limited to the initial reception of the visual representation at the retina, the transfer and reconstruction of visual information when it passes through the visual hierarchy, feature extraction and object identification, the formation of an abstract representation in the memory, and semantic reasoning based on one's knowledge and experience. However, if the visual representation arriving at one's eye already loses information or contains errors, further actions for data reconstruction in one's brain will no doubt suffer. Hence it is important to first measure the quality of a visual representation in terms of visual reconstructability before any perceptual and cognitive processes.

It is unlikely that any single quality metric can determine the overall quality of a visualization. Over-emphasizing aesthetics [FB09] may undermine visual salience [JC10], and vice versa. Over-emphasizing entropy maximization [XLS10] may undermine abstraction quality [CYWR06], and vice versa. However, using a combination of quality metrics will likely help users to arrive a balanced quality judgement. *Visual reconstructability* is a new addition to the currently-rather-small collection of quality metrics for visualization.

A visual representation of a dataset is said to facilitate a

full visual reconstruction if the original dataset can be derived from the visual representation accurately by using a machine-vision system. We utilized commonly-used computer vision algorithms in our implementation, while recognizing that the continuing advances in computer vision will bring better reconstruction results.

We use 2D vector field visualization as an example to demonstrate the concept of visual reconstructability. A 2D vector field can be depicted by several types of visual representations, such as streamlines, arrow glyphs, and line integral convolution (LIC). As illustrated in Fig. 2, errors in reconstruction, regardless by human viewers or machine vision, may be caused by a number of factors, including but not limited to: (a) information loss due to a sparse visual representation of the original vector field, (b) noise introduced by visualization techniques such as LIC, (c) geometric approximation between discrete samples, (d) aliasing due to display resolution. In order to accommodate a broad range of error sources, we carefully selected a number of vector fields obtained from numerical simulation under the guidance of a CFD domain expert. We apply three commonly-used 2D techniques, streamlines, arrow glyphs, and LIC, to these datasets. We develop a reconstruction system for each type of visualization based on established algorithms in the computer vision literature. We then compute the errors between the original and reconstructed datasets, and use the error metric to evaluate the three visualization techniques considered in this work. Our contributions of this work are:

- We propose a new objective approach to evaluate different visualizations, and assess its feasibility using 2D vector field visualization as a test case.
- We develop three quality metrics for streamline, glyph and LIC visualization based on their reconstructability.
- For testing reconstructability quality metrics, we carefully prepared simulation results that exhibit different common features, allowing an objective comparison between different visualization methods and their parameter spaces.

2. Related Work

The traditional way to assess quality information about a visualization technique is to conduct a user study [KHI*03]. Commonly, users have to perform domain specific tasks in these experiments and the results are compared with respect to accuracy and time. In the area of flow visualization, several studies that investigate interaction, object identification and perception have been conducted. Laidlaw et al. [LDM*01, LKJ*05] compare a range of 2D flow visualization methods (vector glyphs, LIC, streamlines). They conclude that visualization methods that show both downstream direction and integral curves are most effective. Forsberg et at. [FCL09] investigate 3D vector field techniques. They conclude that methods which minimize occlusion, clearly depict downstream direction and velocity magnitude, and provided fewer 3D cues were most effective. Ware [War06] confirmed the importance of 3D cues. Joshi and Rheingans [JR08] found that illustratively augmented techniques provide a noticeable improvement in accuracy and speed for time-varying data.

Another way to assess visualization quality is with automatic comparison techniques. Shen et al. [SPU98] distinguish three levels of comparison, namely *data level*, *feature level*, and *image level* comparisons. The data level comparison techniques compare data values, e.g., Pagendarm and Post [PP97], who investigate characteristics of streamlines and streamribbons, e.g., position, to evaluate flow visualizations. Feature level techniques compare extracted features in the data. Verma and Pang [VP04] compare linetype features visualized using streamlines and streamribbons. Image level comparison techniques compare the data as depicted on images. In [SWMJ99] a composite metric for comparing grid-based datasets is proposed. Zhou et al. [ZCW02] evaluate several metrics for image-level comparison between videos of experiments and visualization results.

General quality metrics for visualization cover a wide scope. Cui et al. [CYWR06] consider data abstraction quality. Van Wijk [vW05] suggests to measure the value of visualization based on the notion of profit. Filonik and Baud [FB09] propose a quality metric for measuring aesthetics in information visualization. Jänicke and Chen [JC10] develop a quality metric based on visual salience. Bertini and Santucci [BS06] proposed three generic metrics, size, visual effectiveness and features preservation, and discussed how they might be implemented.

There are very few reports on reconstructing vector fields from flow visualizations, except that Risquet [PR02] reconstructs directional information from texture-based flow visualizations to guide the correction of the underlying vector field. Chen et al. [CCK07] reconstructed a vector field from sampled points on the streamlines in a visualization by interpolating values of these points via Delaunay triangulation. Pineo and Ware [PW08] reconstructed and evaluated glyph-based and texture-based flow visualization using Gabor filters. We enhance the previous works by integrating reconstruction methods for streamlines, glyph-based and texture-based visualization under the same framework for quality evaluation. This enables a qualitative comparison of three widely-used forms of flow visualization based on reconstructability, whilst only two forms were considered in [PW08, LDM*01, LKJ*05]. Instead of artificial vector fields, we employ real CFD simulations.

Pineo and Ware [PW08] first made a case that Gabor filters can represent some functions of the human vision system. While we are cautious about the limitation of machine vision algorithms, we also appreciate the existing advances. Image-based data reconstruction is extensively researched in digital image processing and computer vision. The most fundamental approaches for directional reconstruction are the Marr-Hildreth-operator [MH80], Canny-

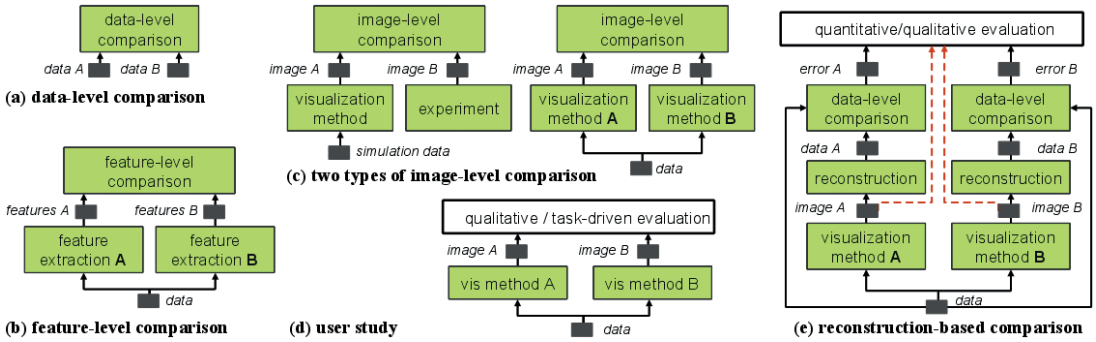


Figure 1: *Comparative visualization models based on (a) differences on the data level, (b) differences in extracted features, (c) different visualizations, (d) user studies, and (e) the reconstructability of the underlying data.*

operator [Can86], Sobel-operator [JÖ2], and Hough transform [DH72]. In texture-based images, directional information can be obtained from spatial frequencies using Gabor filters. Marr and Hildreth described how convolution can be implemented in neural hardware and Palmer [Pal99] details the relationship between human vision and the Gabor function making the Gabor filter a good model for human vision. In our applications we will focus on the directional information conveyed by flow visualizations, and evaluate how well the reconstructed direction matches the underlying data.

3. Motivation

Fig. 1 shows several approaches to comparative evaluation. Figs. 1(a-c) represent the category of automatic comparison methods, including data-level, feature-level and image-level comparison. For comparative evaluation of visualization methods, the image-level methods are the most relevant. The main difficulty of using image-level comparison is that most difference metrics cannot express the difference between two distinct visual representations from the perspective of human perception. For example, a simple image difference between a streamline visualization and a LIC visualization cannot convey anything about the relative merits of the two visualization. Even using those more advanced image difference metrics in [ZCW02] and [SWMJ99] would not provide an adequate solution to this problem.

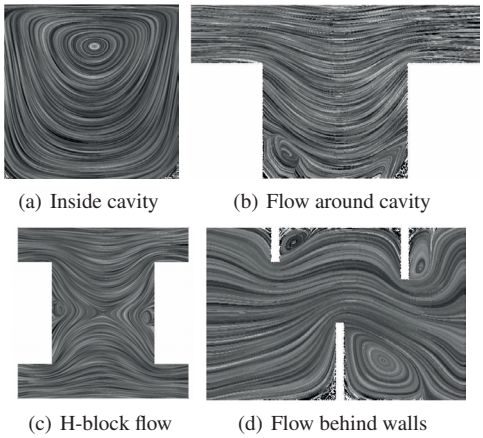
Meanwhile, as illustrated in Fig. 1(d), user studies inherently account for the human perception of a visualization, but are not applicable in a large scale or on a daily basis. In addition, the results of user studies are difficult to disseminate to application users. Despite that a number of user studies were conducted for different flow visualization techniques as mentioned in Section 2, it is rather uncommon for an ordinary user to make a selective decision about visualization techniques and parameter spaces based on the findings of a specific user study. In everyday practice, the decision is normally made based on a qualitative evaluation or personal experience in the best case scenario, or a subjective or circumstantial judgment in the worst.

Figure 2: *Sources of error in flow vis. (left to right): missing information, interpolation, quantization, noise*

This problem motivates us to explore a new approach by combining the concept of quality metrics (e.g., [CYWR06, JC10]) with that of automatic comparison (e.g., [SPU98, VP04, ZCW02]. Fig. 1(e) shows a schematic illustration of our new approach. To compare two visualization methods, a data reconstruction algorithm is applied to the visualization results, with the intention of estimating how much a human observer can see. For example, given two different visual representations *image A* and *image B* for the same vector field *data*, the reconstruction algorithm will result in two vector fields *data A* and *data B* from *image A* and *image B* respectively. A data-level comparison metric can then be applied to the pair of *[data, data A]*, and that of *[data, data B]*, resulting in two error indicators, *error A* and *error B*. From *error A* and *error B*, a user can tell the potential visual reconstructability of the two visualization methods. Coupled with a qualitative judgment and if available other quality metrics, the user can make an informed objective decision about the two methods.

4. Case Study Design

Let D_a and D_b be the two 2D vector fields, and $\vec{a}_{x,y}$ and $\vec{b}_{x,y}$ be the two specific vectors at position (x,y) in D_a and D_b respectively. The difference between $\vec{a}_{x,y}$ and $\vec{b}_{x,y}$ can be decomposed into two aspects, magnitude and direction. It is common for some flow visualization techniques to omit one aspect. For instance, a LIC-based visualization may choose not to depict magnitude, while a streamline visualization may choose not to distinguish between $\vec{a}_{x,y}$ and $-\vec{a}_{x,y}$, relying on the users to work out the flow direction of each streamline based on their semantic understanding of the application.



(a) Inside cavity (b) Flow around cavity

(c) H-block flow (d) Flow behind walls

Figure 3: *Simulated CFD datasets*

When viewing visualization results, especially with the intention of comparing D_a and D_b, a user would typically like to determine if there are differences in terms of:

- major features, such as the number of vortices, saddle points, sources and sinks, and their positions;
- overall flow characteristics, such as velocity, linearity, convergence or divergence;
- detailed local quantities, such as the velocity and direction of the flow at a specific location, and the size of a vortex.

It is generally agreed that the user's cognitive judgement of these differences is based on the user's perception and estimation of the magnitude and direction at the relevant points of the vector fields concerned. As illustrated in Fig. 2, the errors in perception and estimation can be caused by several reasons. It is thus essential for us to test the proposed reconstructability-based quality metrics using vector fields that reflect different levels of these errors, while capturing some typical features and characteristics of a flow. In addition to synthetic vector fields, we also decided to use vector fields resulting from computational fluid dynamics (CFD) simulations, in the hope that they are more representative of real applications than synthetic vector fields.

4.1. Models

Together with a CFD expert, we simulated 5 time-dependent datasets at different levels of complexity using the *IcoFoam* solver in the OpenFOAM®toolbox [Ope10]. Fig. 3 depicts four examples of increasing complexity. Fig. 3(a) models a flow inside a cavity. Fig. 3(b) represents a lid-driven cavity, which is modeled by a closed block with fixed walls on both sides and at the bottom. The lid is specified with an edge moving uniformly and horizontally. With the lid driving across, the flow inside circulates due to the static boundaries, forming a clearly observable vortex. In Fig. 3(c), there are two channels with flow going in opposing directions and a space that connects them together. The field exhibits two vortices and a saddle point. In Fig. 3(d), the flow was simulated in a way such that it would contain 3 different sized vortices. As a vortex may occur when a flow passes over a wall, we constructed a channel that included three walls of different heights. By altering the height of each wall, we can change the size of the vortex forming behind it.

4.2. Visualization Techniques

Flow visualization techniques are commonly divided into four major categories [PVH*02]: *direct, geometric, texture-based, and feature-based techniques*. The division is based on the type of preprocessing and the graphical primitives that are used to depict the data. In the following we will use characteristic examples from the first three groups, which provide a holistic depiction of the dataset.

Direct Visualization – Hedgehogs Hedgehogs are a very basic, fast, and easy to implement technique. A large number of variations of these icons exist based on their shape and positioning. We used an algorithm that draws classical vector icons at the grid position of the vector field.

Geometric Visualization – Streamlines Integral structures, such as streamlines, are computationally more expensive than the direct techniques, but give a better description of large structures and the spatial evolution of the flow. The many varieties of the streamline algorithms differ in the type of line primitive, the integration length, the seed points, and the spacing between the integral structures. We use an adaptive seeding strategy and adaptive distance control.

Texture-based Visualization – LIC The LIC algorithm [CL93, SK98] starts with a texture that is initialized with white noise. In an iterative process the texture is filtered along streamlines using a local kernel. Due to the large number of streamlines that have to be computed, LIC is the computationally most expensive technique, but provides information for each pixel of the displayed texture.

5. Vector Field Reconstruction

For all types of visualization, the reconstruction of the vector field data consists of two steps. First, local information about the direction is extracted from the visualization. Based on the visual metaphor, we use either a reconstruction algorithm for line-based representations, or an algorithm that recovers directional information from a texture. Analogous to findings in computer vision, we rely on two pipelines for these visually very different types of input. Both algorithms do not provide information for all pixels in the image. For line-type features there simply is no information available for some pixels, and in the case of texture-based representations there are pixels where the reconstruction quality is too low. In both cases, a second reconstruction step is necessary that interpolates between existing values. Fig. 4 provides an overview over the two pipelines for line-based and texture-based reconstruction.

5.1. Reconstruction of Texture-based Representations

In texture-based visualizations, such as LIC images, the direction of the vector field is encoded in the local orientation

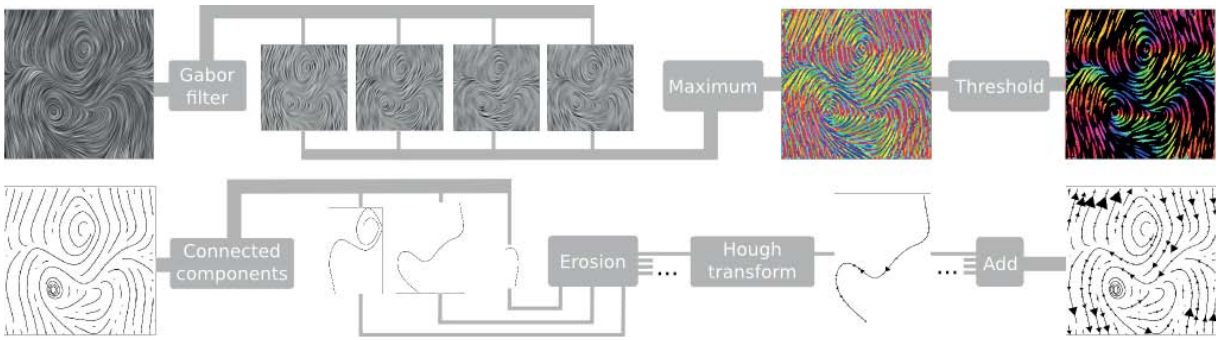


Figure 4: *Reconstruction pipelines: (top) The reconstruction of texture-based visualizations uses Gabor filters to quantify local orientations of the texture. (bottom) The reconstruction of line-type structures is based on the Hough transform.*

of the image texture (see pipeline in Fig. 4). A common technique to extract directional information of textures are Gabor filters from image processing:

$$g(x,y;\lambda,\theta,\sigma) = \exp\left(\quad (x_r^2 + y_r^2)/(2\sigma^2) \right) \cos(2\pi x_r/\lambda)$$
$$x_r = x\cos\theta + y\sin\theta \quad y_r = \quad x\sin\theta + y\cos\theta, \qquad (1)$$

where (x,y) are pixel coordinates, λ represents the wavelength of the cosine factor, θ is the orientation of the Gabor function, and σ the sigma of the Gaussian envelope. The phase shift was set to 0 and the spatial aspect ratio to 1. Gabor filters feature a striped pattern on a small spatial support, and when convolved with a local patch of the textured image, the resulting value tells how well the local structure of the texture agrees with the Gabor filter at a given angle. The frequency and orientation representations of Gabor filters are very similar to the human receptive field [Pal99], and are hence, well suited to model the human perception of texture-based encodings of directed data.

The Gabor filter with given parameter (λ,θ,σ) quantifies the local orientation of the texture in the direction $\theta \quad 90^\circ$. To derive a measure for all possible orientations, the filter has to be applied with multiple angles θ. This is commonly done in a filter bank consisting of Gabor filters with various scales and rotations. To reduce computation costs, we use a single scale that best captures the texture size. The angle θ of the Gabor filter varies between 0° and 180° with an increment of 1°. For our datasets we used $\lambda = 0.066$ and $\sigma = 0.05$, which could capture the data best. Filtering in Fourier space [JÖ2] is used to decrease computational costs.

To derive an estimate of local orientations, the filter image with the highest score has to be found for each pixel. Depending on the local structure of the texture, this can be very difficult as in some areas the filter results are very low for all possible angles. Hence, we apply a threshold to store only high-confidence results. As a rule of thumb, $0.5 \cdot maxval$ can be used, where maxval is the highest value in any of the filtered images. In our analysis we found that roughly 50% of the data can be reliably reconstructed. To derive values in-between the reconstructed data, we use the interpolation method presented in the next section.

5.2. Reconstruction of Line-based Representations

To reconstruct directional information from a line-based visualization, one has to determine the direction of the lines in the image and assign the derived value to the corresponding pixels. The pipeline of the reconstruction procedure is depicted in Fig. 4(bottom). While vector glyphs feature simple straight lines, streamlines are more challenging to reconstruct since they feature curved lines. In order to handle a large variety of line-based visualizations and local defects in their representation, we employ the Hough transform [DH72] to reconstruct directional information from lines.

Unlike many other line reconstruction techniques, the Hough transform does not operate in image space, but in parameter space, which makes it more robust and able to detect many different line types. The parameter space is spanned by the line parameters r and θ, where r is the line's shortest distance to the origin and θ represents the angle between this connecting line and the x-axis. After a pixel-based voting process edges can be reconstructed from local maxima in the parameter space. Further improvements of the original Hough algorithm can be found in [SS01].

To ensure an optimal reconstruction of line segments, the input image is assumed to feature thin, black lines on a white background. To ensure these properties, we apply several preprocessing steps to the input image: 1) First the image is converted to black and white using thresholding. In gray-scale images with luminance range $[0;255]$ we commonly use 240 as threshold to account for small variations in the background. 2) Due to anti-aliasing and parameter settings of the different line-based algorithms, the line representations are often a few pixels wide and have to be thinned. The morphological operator *erosion* is used for this task, which computes the "skeleton" of the structures in the image. We also use morphological operators to derive the direction of the vector glyphs, while filtering the arrow head. In the literature, the standard approach to generate line-based input for the Hough transform is an edge detection, e.g., Canny or Sobel filter, which can easily operate on natural scene images. In our application, however, it is less suited, as edge

detectors give two edges for each line and we would have to estimate the core of the line from the two reconstructed boundaries. Hence, we chose erosion which provides the required information directly. 3) To improve the results of the Hough transform, the original image is separated into connected components that are analyzed separately by storing each connected component in a new image which is Hough transformed. After the preprocessing, the lines in each connected component are reconstructed using the Hough transform. In a final step, all line segments from the individual connected components are stored in a single image. To derive values in-between the reconstructed data, we use the interpolation method presented in Section 5.3.

5.3. Interpolation

The reconstructed data obtained from the algorithms explained in the last two sections features many gaps. In the case of line-based visualizations, data cannot be reconstructed in areas, where there is no line information. Textures on the other side theoretically provide information for each pixel in the image, but in some areas there is high uncertainty in the reconstruction as several directions are equally likely.

Analogous to the processing of visual stimuli in the human visual system [NL82], our algorithm interpolates in-between reconstructed data. Additionally, interpolation ensures that the derived quality metric is meaningful. If data was not interpolated, areas that are difficult to represent, e.g., areas of very turbulent flow, could simply be left out in the visualization and would result in better error values. In our model, we use bilinear interpolation from neighboring reconstructed data points. Therefore, a Delaunay triangulation [dBCvKO08] of the reconstructed data is computed. The input positions P to the algorithm are the set of pixel positions, where directional data could be reconstructed. A triangulation subdivides the plane into triangles such that the positions in P are the vertices of the triangles. In the case of the line-based reconstruction additional constraints for the triangulation have to be added. To ensure correct interpolation along reconstructed lines, these lines have to correspond to edges in the triangulation. In our implementation, we use the Delaunay algorithm of the *cgal*-library [Cga10], which provides a method to set constraints on certain edges that have to be included in the triangulation. We use this feature to retain reconstructed lines as edges in the triangulation.

Techniques commonly used for the interpolation of visual data are nearest neighbor, bilinear and cubic interpolation, which can be either performed on a Delaunay triangulation or a regular grid for spline-based interpolation. We evaluated Delaunay and nearest-neighbor interpolation and found Delaunay interpolation to be superior. Kadyrov and Petrou [KP04] conducted a more extensive study on the different interpolation strategies and found that cubic interpolation performed slightly better than linear interpolation, and both of them much better than nearest neighbor interpolation. Hence, we chose bilinear Delaunay interpolation to estimate values inside the triangles of the triangulation.

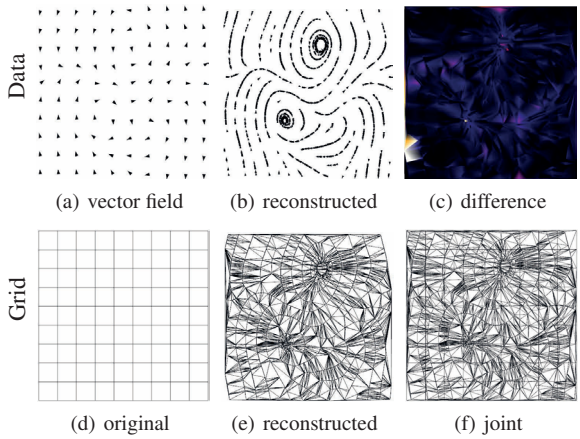


Figure 5: *Error computation: To compare the (a) original data and the (b) reconstructed values, a joint version of the two grids (d) and (e) has to be computed. The joint grid (f) comprises all vertices and edges as given in (d) and (e).*

6. Error Calculation

In the previous section, the reconstruction of directional information from flow visualizations is detailed. The result of this process is a triangulation of the domain with reconstructed values at the vertices of the grid (Fig. 5(b) and 5(e)). To quantify the error between the original and the reconstructed data, the two datasets have to be compared. Therefore, we compute a triangulation of the domain whose set of vertices is the union of the vertices of the original dataset and of the reconstructed positions. The constraints defined in the Delaunay triangulation are carried forward to the new point set. Additional constraints to include the edges in the original field are added as well. The new set of positions along with the combined constraints is triangulated using the Delaunay algorithm (Fig. 5(f)). The values at each vertex of the new triangulation are the normalized dot product of the vectors in the original and the reconstructed dataset (Fig. 5(c)).

In our evaluation we use two different methods to inspect the difference field:
A *visual inspection* of the colormap representation of the difference field helps to identify areas that feature strong deviation between the original and the reconstructed data. In this representation we depict the angle in degree between the original vector and the reconstructed one, which is computed by the arc cosine of the difference field.
To enable an easy-to-compare *quantitative quality measure* of a vector field visualization, we integrate the difference over the entire field. This can be done by summing up the integrated error of all triangles in the triangulation. The final error is the mean error of the entire flow domain.

Both error metrics are complementary to one another. While the numeric measure is easy to compare between multiple visualizations and parameter settings, the visual error representation provides an intuitive depiction of areas with strong deviation.

7. Results and Analysis

In the results section we will focus on two aspects of quantitative evaluation of visualization algorithms. In the first part, we will investigate three techniques and their performance when depicting two different datasets. In the second part, we will compare the individual algorithms with respect to different parameter settings and see how the visualization quality changes with modified parameters.

7.1. Cavity

The first dataset, the simulation of flow inside a cavity, contains a single feature (the central vortex) and a rather uniform structure in the surrounding medium. The grid along with the normalized vector data is depicted in Fig. 6(a). The colormap in Fig. 6(b) shows the orientation of the vectors and is the ground truth to be achieved by the directions reconstructed from the flow visualizations. In Fig. 6(f,i,l) we see this data being visualized using hedgehogs, streamlines and LIC. The basic structure is correctly represented by all three algorithms.

Comparing the colormap depictions of the reconstructed angles in Fig.s 6(g,j,m) to the ground truth in Fig. 6(b), all three visualizations appear quite good. A few outliers are visible in the LIC reconstruction, where the threshold was locally not good enough to filter the defects. The only area of major differences is the center of the vortex. While the color pattern of the hedgehog reconstruction is very similar to the ground truth, the reconstructions for streamlines and LIC look quite different. The error plots in Fig. 6(h,k,n) show the differences more clearly. Major errors in the reconstruction process of the hedgehog image occur in areas of strong curvature, where the error is often above 20°. In contrast, the error induced by streamlines is commonly much lower ($< 15^\circ$) in turbulent areas, which is due to the shape of the streamlines, which follow the direction of the input field closely. Large errors are visible in the lower corners of the image, which is probably due to the sparse sampling in this area. The largest overall error is found in the LIC visualization. Though Gabor filters with a resolution of 1° were used, most areas feature errors around 10°.

In summary, we found that concerning the average error hedgehogs and streamlines perform better than LIC. Streamlines have an advantage in areas of turbulent flow and hedgehogs perform slightly better in areas where the flow direction is constant or very uniform.

7.2. H-block flow

The H-block dataset consists of two linked tubes as depicted in Fig. 7. The fluid in the upper part flows from left to right,

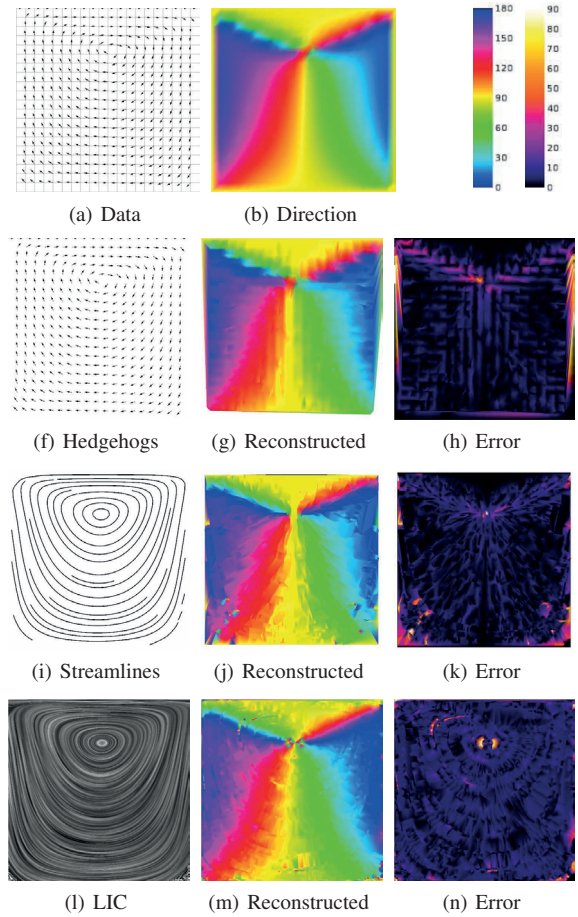
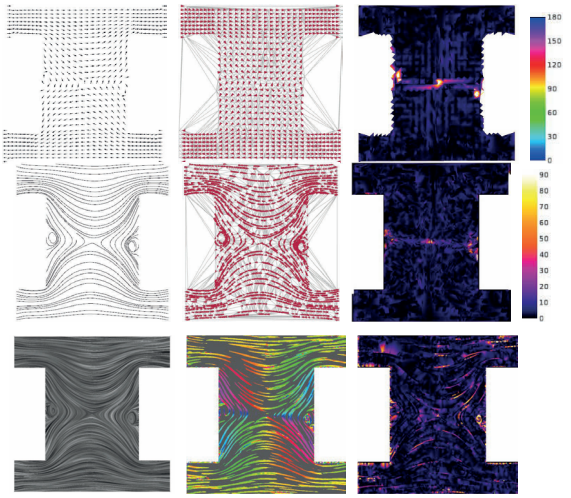


Figure 6: *Reconstruction of the cavity dataset: Fig. (a) shows the original grid along with the normalized vector data. A colormap of the direction of the vector field is given in (b). (f,i,l) Three visualization techniques and (g,j,m) the reconstructed directions. (h,k,n) The difference of the directions of the original and the reconstructed field in degrees.*

the one in the lower part in the opposite direction. These currents in opposite directions result in three features in the central part of the domain: two vortices close to the boundaries and a saddle in the center. Overall the structure of the flow is much more turbulent than the one in the previous example.

The three images in the second column of Fig. 7 depict the reconstructed data. The red triangles in Fig. 7(b) for vector glyphs and streamlines depict the location and direction of reconstructed data and they seem to agree in general very well with the original data. The error plot reveals that hedgehogs do not well represent the data around the saddle point. The third image in the second column represents the data reconstructed from a LIC image. Similar to the previous example, the reconstruction of LIC features highest errors with approximately 10°. Nevertheless, LIC has a very uniform error profile and depicts all features correctly.



(a) Visualization (b) Reconstr. (c) Errors

Figure 7: *Reconstruction of the H-block dataset: For each technique (glyphs, streamlines, LIC), column (a) gives the flow visualization, column (b) the reconstructed data, and column (d) the reconstruction error in degree.*

Summarizing, the second example confirms the findings of the first one. Hedgehogs perform best when depicting areas of straight flow and has major difficulties in representing turbulent areas. This more complex example plays to the strengths of streamlines, that are superior to hedgehogs when it comes to the depiction of turbulent flow. Like in the previous example, LIC features the highest average error, but has a consistent error profile in the entire domain and is hence able to consistently depict a large variety of structures. While vector glyphs and streamlines exhibit largest errors around the saddle point, errors are more widely distributed in the LIC images.

7.3. Comparison of Parameter Settings

In the last section of the results, we want to have a look at the error variability within one technique as induced by different parameter settings. Therefore we use a synthetic dataset that comprises many common flow features in a small area.

Vector Glyphs. Fig. 8(left) depicts the mean error for different parameter settings of the hedgehog algorithm. In this example we varied the density and the size of the vector glyphs. The density is modified by the number of sampling positions in x- and y-direction 9(top). We used four different grid sizes $n \times n$ and 6 scaling factors s, a fraction of the edge length of a grid cell. The mean error varies between 6.4° and 16.9°. The best size of the glyph is $s = 0.4$ regardless of the scale. For smaller glyphs the reliability of the reconstruction algorithm dropped, causing errors due to imprecise reconstruction. When using large glyphs, the error source is two-fold. First, large symbols are not well suited to depict turbulent areas as the direction of the flow changes very quickly and large glyphs do not reflect this property. Second, the finer the grid resolution, the closer the vectors get and the higher the probability that they intersect, especially in dynamic areas, which leads to wrong estimations of the local direction. Comparing the measurements for grid resolution and glyph size separately, we see that the maximum difference between errors within a given grid resolution is commonly around 4°. For different sizes of the glyphs within the same grid, the average error can change by up to 6.4°. While the individual parameter can improve the average error only by a few degrees, their combination can result in 10.5° better performance. Hence, it is important to adequately control the density and the size of vector glyphs in a hedgehog presentation to obtain optimal results.

Streamlines. Fig. 8(center) depicts the results of different parameter settings of the streamline algorithm. The user specified parameter in this algorithm is the distance between streamlines, and an additional parameter that cannot be controlled is the seeding process which includes a random seeding of starting positions. For distances d in the range $[1;15]$ with step size 2, we computed the mean reconstruction error of eight visualizations each with different start configurations based on the random seeding. Hence, for each distance, we resolve an average error over the eight trials and the corresponding standard deviation σ (Fig. 8(center)). Streamline visualizations with a distance of 1 (Fig. 9(center)) feature a mean error of 4.64° and standard deviation 0.01°. When the distance is set to 15, the mean error increases to 12.14° with standard deviation 0.8°. The error increases steadily with increasing distance of the streamlines, and so does the variance. This means that the larger the distance between streamlines, the larger the error and the more the result depends on a good initial seeding.

LIC. In the LIC visualization two parameters can be modified, the length of the filter kernel k and maximum number of hits per pixel n, which indicate how many convolutions per cell are allowed. While a long filter kernel results in long lines in the visualization, the second parameter controls the smoothness of the image but also results in more blurred structures. The length of the filter kernel k was varied between 13 and 100 pixels, and the number of hits per pixel was in the set $[1,3,9]$ (Fig. 9(bottom)). Worst results with respect to reconstructability were achieved with a short kernel and few convolutions per pixel (error = 28.9°) as depicted in Fig. 8(bottom). The smallest error was achieved with long filter kernels and many hits per pixel (error = 9.12°). In general, we can observe a decreasing mean error with increased computational costs, i.e., longer kernels and/or more hits per pixel (Fig. 8(right)). The changes for parameter settings above $(k = 50, n = 3)$, however, were only minor and have to be traded against the computational costs.

7.4. General Findings and Comparisons

All results were computed on a desktop PC on a single Intel core (3.07GHz) with 8GB RAM. In general, the reconstruc-

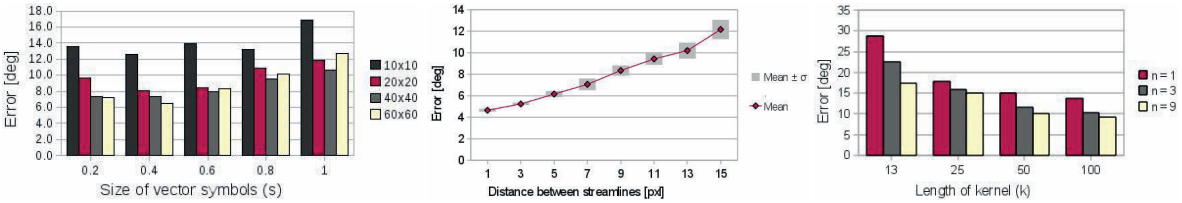


Figure 8: *Errors for the three techniques (left to right: vector glyphs, streamlines, and LIC) and different parameter settings.*

tion of texture-based data is faster than the reconstruction of line-based data, but all images could be reconstructed within less than 30 sec.

The analysis of different flow datasets with different feature settings showed that streamlines perform best when it comes to reconstruction quality. The error of vector glyphs depends a lot on the structure of the flow. LIC visualizations feature a consistent moderate average error. Streamline visualizations proved to be well suited for intricate and very dynamic flow structures, where vector glyphs performed worst. Although LIC features a quite large mean error, this error is very consistent over the entire domain and does not increase when it comes to the depiction of turbulent structures.

The techniques and datasets that we researched are similar to the ones used by Laidlaw et al. [LKJ*05]. Comparing the results of their user study and our automatically evaluated results, we find that the major findings do well agree.

8. Conclusion

In this paper we described an objective quality metric for visualizations based on visual reconstructability. This metric favors visualizations that allow for a precise reconstruction of the underlying data. We developed two pipelines for the reconstruction of line-based and texture-based directional information from flow visualization images. The metric was computed for three common visualization techniques (vector glyphs, streamlines and LIC) and a variety of flow simulations with characteristic features. In two test cases and a comparison of different parameter settings for the same algorithm, we compared and evaluated the different visualizations and settings.

It is a common understanding that the quality of a visualization should be determined by a combination of metrics, while research projects tend to focus on an individual metric. It is also a common understanding that a computational quality metric is not the same as a human vision system, at least for the time being. In many ways, this is similar to measuring the quality of apples, which requires a combination of quality metrics. While the ultimate evaluation is to look, feel and taste apples by humans, we cannot perform this form of evaluation on every apple.

One also need to aware that the accuracy of a reconstructability metric depends on the implementation of the reconstruction algorithm used. At the moment, there is no

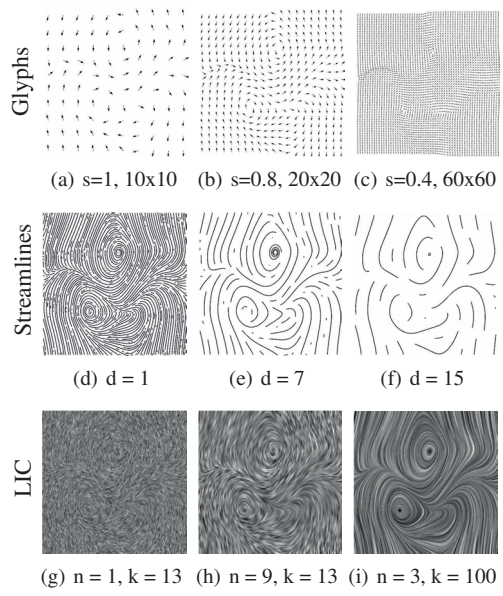


Figure 9: *Visualizations with different parameters.*

obvious mean to determine the ground truth of an "ideal" reconstruction algorithm for analysing the errors of the reconstruction algorithm.

As suggested in [JC10], the development of different quality metrics can enrich future visualization systems by providing users with a collection of metrics for self-evaluation. Together, these metrics can be used on a large-scale and daily basis for quick and intuitive access to visualization quality, and help creators of visualizations understand why certain visualizations are better than others and get ideas on how to improve existing techniques. Of course, the metrics cannot replace user studies, and the development and improvement of such metrics will continue to benefit from user studies.

An interesting direction for future research is to develop an appropriate metric that can reflect human viewers' ability to infer continuity from typical flow visualization. The reconstruction pipeline is suitable for parallelization, which may be interesting in future development. One urgent challenge, which requires a collective effort from the visualization community, is to introduce quality metrics to practical software systems.

References

[BBC*05] BRODLIE K., BROOKE J., CHEN M., CHISNALL D., FEWINGS A., HUGHES C., JOHN N., JONES M., RIDING M., ROARD N.: Visual supercomputing – technologies, applications and challenges. *Comp. Graph. Forum 24*, 4 (2005), 217–245. 1

[BS06] BERTINI E., SANTUCCI G.: Visual quality metrics. In *BELIV '06: Proceedings of the 2006 AVI workshop on Beyond time and errors: novel evaluation methods for information visualization* (2006), ACM. 2

[Can86] CANNY J.: A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell. 8*, 6 (1986), 679–698. 3

[CCK07] CHEN Y., COHEN J., KROLIK J.: Similarity-guided streamline placement with error evaluation. *IEEE TVCG* (2007), 1448–1455. 2

[Cga10] CGAL, Computational Geometry Algorithms Library, last accessed 03/2010. http://www.cgal.org. 6

[CL93] CABRAL B., LEEDOM L. C.: Imaging vector fields using line integral convolution. In *SIGGRAPH '93* (1993), ACM, pp. 263–270. 4

[CYWR06] CUI Q., YANG J., WARD M., RUNDENSTEINER E.: Measuring data abstraction quality in multiresolution visualizations. *IEEE TVCG 12*, 5 (2006), 709–716. 1, 2, 3

[dBCvKO08] DE BERG M., CHEONG O., VAN KREVELD M., OVERMARS M.: *Computational Geometry*. Springer, 2008. 6

[DH72] DUDA R. O., HART P. E.: Use of the hough transformation to detect lines and curves in pictures. *Comm. ACM 15* (1972), 11–15. 3, 5

[FB09] FILONIK D., BAUR D.: Measuring aesthetics for information visualization. In *IV '09: Proc. of the Conf. Info. Vis.* (Washington, DC, USA, 2009), IEEE Computer Society, pp. 579–584. 1, 2

[FCL09] FORSBERG A., CHEN J., LAIDLAW D.: Comparing 3d vector field visualization methods: A user study. *IEEE TVCG 15* (2009), 1219–1226. 2

[JÖ2] JÄHNE B.: *Digital Image Processing*. Springer, 2002. 3, 5

[JC10] JÄNICKE H., CHEN M.: A salience-based quality metric for visualization. *Computer Graphics Forum* (2010), Accepted for Eurographics/IEEE Symposium on Visualization. 1, 2, 3

[JR08] JOSHI A., RHEINGANS P.: Evaluation of illustration-inspired techniques for time-varying data visualization. *Computer Graphics Forum 27*, 3 (2008), 999–1006. 2

[KHI*03] KOSARA R., HEALEY C. G., INTERRANTE V., LAIDLAW D. H., WARE C.: User studies: Why, how, and when? *IEEE Comput. Graph. Appl. 23*, 4 (2003), 20–25. 2

[KP04] KADYROV A., PETROU M.: Reverse engineering the human vision system: A possible explanation for the role of microsaccades. *Pattern Recognition, International Conference on 4* (2004), 64–67. 6

[LDM*01] LAIDLAW D. H., DAVIDSON J. S., MILLER T. S., DA SILVA M., KIRBY R. M., WARREN W. H., TARR M.: Quantitative comparative evaluation of 2d vector field visualization methods. In *VIS '01: Proceedings of the conference on Visualization '01* (Washington, DC, USA, 2001), IEEE Computer Society, pp. 143–150. 2

[LKJ*05] LAIDLAW D. H., KIRBY R. M., JACKSON C. D., DAVIDSON J. S., MILLER T. S., DA SILVA M., WARREN W. H., TARR M. J.: Comparing 2d vector field visualization methods: A user study. *IEEE Transactions on Visualization and Computer Graphics 11*, 1 (2005), 59–70. 2, 9

[MH80] MARR D., HILDRETH E.: Theory of edge detection. *Proceedings of the Royal Society of London. Series B, Biological Sciences 207*, 1167 (1980), 187–217. 2

[NL82] NYMAN G., LAURINEN P.: Reconstruction of spatial information in the human visual system. *Nature 297* (1982), 324–325. 6

[Ope10] OPENFOAM: Open cfd ltd., openfoam (open source cfd toolbox), last accessed 03/2010. http://www.openfoam.com/. 4

[Pal99] PALMER S.: *Vision Science: Photons to Phenomenology*. MIT Press, Cambridge, 1999. 3, 5

[PP97] PAGENDARM H.-G., POST F. H.: Studies in comparative visualization of flow features. In *Scientific Visualization, Overviews, Methodologies, and Techniques* (1997), IEEE Computer Society, pp. 211–227. 2

[PR02] PÉREZ RISQUET C.: *Visualisierung und RÃijckrechnung von zwei-dimensionalen Vektordaten*. PhD thesis, Rostock University, 2002. 2

[PVH*02] POST F. H., VROLIJK B., HAUSER H., LARAMEE R. S., DOLEISCH H.: Feature Extraction and Visualization of Flow Fields. In *Eurographics 2002 State-of-the-Art Reports* (2002), pp. 69–100. 4

[PW08] PINEO D., WARE C.: Neural modeling of flow rendering effectiveness. *ACM Trans. Appl. Percept. 7* (June 2008), 20:1–20:15. 2

[SK98] SHEN H.-W., KAO D. L.: A new line integral convolution algorithm for visualizing time-varying flow fields. *IEEE TVCG 4* (1998), 98–108. 4

[SPU98] SHEN Q., PANG A., USELTON S.: Data level comparison of wind tunnel and computational fluid dynamics data. In *VIS '98: Proc. of the conf. on Vis.* (1998), IEEE Computer Society Press, pp. 415–418. 2, 3

[SS01] SHAPIRO L., STOCKMAN G.: *Computer Vision*. Prentice-Hall, Inc., 2001. 5

[SWMJ99] SAHASRABUDHE N., WEST J. E., MACHIRAJU R., JANUS M.: Structured spatial domain image and data comparison metrics. In *VIS '99: Proc. of the conf. on Vis.* (Los Alamitos, CA, USA, 1999), IEEE Computer Society Press, pp. 97–104. 2, 3

[VP04] VERMA V., PANG A.: Comparative flow visualization. *IEEE TVCG 10*, 6 (2004), 609–624. 2, 3

[vW05] VAN WIJK J.: The value of visualization. In *Proc. IEEE Visualization 2005* (2005), pp. 79–86. 2

[War06] WARE C.: 3d contour perception for flow visualization. In *Applied Perception on Graphics and Visualization (APGV)* (2006), ACM, pp. 101–106. 2

[XLS10] XU L., LEE T.-Y., SHEN H.-W.: An information-theoretic framework for flow visualization. *IEEE TVCG 16*, 6 (2010), 1216–1224. 1

[ZCW02] ZHOU H., CHEN M., WEBSTER M. F.: Comparative evaluation of visualization and experimental results using image comparison metrics. In *VIS '02: Proc. of the conf. on Vis.* (2002), IEEE Computer Society, pp. 315–322. 2, 3