Computer Aided Geometric Design 30 (2013) 521–528

Contents lists available at SciVerse ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

Interactive comparison of multifield scalar data based on largest contours

Dominic Schneider [a], Christian Heine [a,*], Hamish Carr [b], Gerik Scheuermann [a]

[a] Dept. of Computer Science, University of Leipzig, Leipzig, Germany
[b] School of Computing, University of Leeds, Leeds, United Kingdom

ARTICLE INFO

Article history:
Available online 6 April 2012

Keywords:
Multifield
Comparative visualization
Contour tree
Largest contours
Feature-based visualization

ABSTRACT

We present a feature-based interactive technique for comparing multiple scalar fields. It extracts largest contour features from the topologically simplified contour trees of each field, then measures similarities between features using information theory. These similarities are stored in a weighted graph on which we apply clustering. The clustering identifies coherent features across fields. Finally, the clusters are ranked and presented in a thumbnail gallery providing a quick overview of the whole dataset. We demonstrate the utility of our approach by applying it to a number of complex fluid flow datasets.

1. Introduction

Analyzing multiple fields is a challenging task due to the large amount of data collected and the dependencies between the different variables. An important example is the analysis of fluid flow datasets containing several physical quantities such as velocity, pressure, density, viscosity, as well as derived quantities such as vorticity, λ_2 (Jeong and Hussain, 1995), helicity, to name the most prominent ones. Of special interest in these datasets are the occurring physical phenomena, for example shock fronts, vortices or breakdown bubbles. These phenomena usually extend over several quantities in a specific region forming the physical phenomenon. One major application is to find correlating features within the considered quantities in order to automatically identify the physical phenomena.

Another major application in engineering is the identification and analysis of vortices. More precisely, features from different vortex criteria can be compared regarding their dimensions and relevance. This is important as there is no absolute definition of a vortex yet there exist several vortex region criteria like e.g. helicity, vorticity, λ_2 and locally low pressure. The simplest criterion is vorticity, i.e. the curl of the vector field. It describes the rotational behavior of the fluid, but there can be false positives like shear flow. Another indicator for vortices is locally low pressure, which has the problem that there are numerous reasons for pressure being low and some of them are vortices. A more sophisticated criterion is λ_2. It uses a matrix decomposition of the Jacobian to detect regions of swirling motion. Signed helicity (or just helicity) is the dot product between velocity and vorticity. What makes helicity special is that it takes into account the direction of motion. Hence, vortices are indicated by both strong maxima and minima. Now our method can be used to compare the mentioned vortex region criteria.

A general approach for an interactive comparison of two scalar fields focusing on fluid flow datasets was presented by Schneider et al. (2008). They use largest contour segmentation to identify features in scalar fields. These features are set into relation by encoding spatial overlap in a weighted bipartite graph. For every significant overlap a thumbnail is generated and displayed in a gallery next to the graph. Additionally, the contour trees can be inspected manually, and feature pairs can be selected for showing their contours in a 3-D view. We extend this approach to work with multiple scalar fields revealing phenomena extending over a larger number of variables. We changed the measure of similarity from spatial overlap to

* Corresponding author.
E-mail address: heine@informatik.uni-leipzig.de (C. Heine).

0167-8396/$ – see front matter
http://dx.doi.org/10.1016/j.cagd.2012.03.023

a measure of information-theoretic dependency for two reasons: to have a firmer theoretical footing and to be more flexible with regard to future work. We apply a graph layout that allows to merge the thumbnail and the graph browser for a better utilization of screen space. In comparison to Schneider et al. (2008), our approach can give an overview of the features for the k-fields much quicker than the tedious inspection of $k(k-1)/2$ field pairs.

2. Related work

One of the simplest methods for visualizing multivariate data is to juxtapose visualizations relying on linked views (Wilhelm, 2008). More advanced methods superimpose visualizations by defining multidimensional transfer functions for volume rendering (Kniss et al., 2001), or allow visual queries of individual properties (Gosink et al., 2007).

Sauber et al. (2006) have taken a global approach and use a tree-like graph to show the correlation between scalar fields. However, not all relationships are global; in fact, many are local correlations – especially in fluid flow simulations. Jänicke et al. (2007) looked at local relationships in time-dependent multifield datasets using information theoretic concepts using well-defined neighborhoods, but these neighborhoods are of fixed size and shape. In subsequent work Jänicke et al. (2008) used brushing and linking of a graph structure called attribute cloud to visualize multivariate data.

Linsen et al. (2009) used a multidimensional feature space cluster tree to extract surfaces from multifield particle volume data. The surfaces segment the data with respect to the underlying multivariate function. The surface in object space corresponds to a cluster of the cluster tree in feature space.

Woodring and Shen (2006) allowed for boolean set operations on the data to combine several volumes together into one visualized volume. Akiba and Ma (2007) suggested a trispace approach coupling information visualization with volume rendering to visualize time-dependent, multivariate scalar fields. Lampe et al. (2009) deform the volume to a curve in a neighborhood-preserving fashion. The resulting curves can be analyzed and compared in parallel. Fuchs et al. (2009) used an evolutionary search algorithm to assist hypotheses formalization in multivariate, volumetric data. In recent work Bremer et al. (2010) used different physical quantities to track burning flames in a combustion simulation. An overview of multivariate, multifield visualization techniques can be found in Bürger and Hauser (2007).

3. Contour trees

The contour tree by Boyell and Ruston (1963) is a graph theoretic abstraction of the nesting relationship of all possible contours in a scalar field. Contours are the connected components of an isosurface. The contour tree is the result of shrinking every contour in the field to a single point. This establishes a 1–1 relationship between points on the tree and entire contours in the scalar field. Edges in the contour tree represent a continuous deformation of a contour where connectivity is unchanged for varying isovalue. Critical points where connectivity changes (saddles) appear as inner nodes in the tree; and local extrema appear as leafs in the tree.

Manders et al. (1996) introduced *largest contours*, which are maximal contours that contain only one local extremum, and therefore define features. A segmentation of the domain then follows.

Carr and Snoeyink (2003) developed the flexible isosurface interface where individual contours can be manipulated as single points in the contour tree. The largest contour segmentation is a special case of the flexible isosurface.

Carr et al. (2004) later defined local geometric measures for every contour and used them to simplify the contour tree by removing minor topological features. This allows users to explore individual contours of a dataset interactively. Weber et al. (2007) extended this approach to volume rendering by assigning individual contour tree edges their own transfer function.

4. Feature-based interactive comparison of multifield scalar data

In this section we describe our interactive system for exploration of multiple scalar fields on a simply-connected domain.

For each of the k fields, we compute the contour tree using the algorithm by Carr et al. (2003). We then perform topological simplification on each contour tree driven by volume (Carr et al., 2004) using the same volume threshold. We use the resulting contour trees to compute largest contours, which in the remainder we will view as features. Afterwards we compute a k-partite graph where edge weights store the information-theoretic similarity among features. We cluster this graph using the Chinese Whispers algorithm and rank clusters based on their total volume. Finally, we generate and arrange a thumbnail for each cluster in a gallery and perform a graph layout for each cluster restricted to the thumbnail's area. Each thumbnail contains a preview of the cluster's features when shown in the 3-D view. The user can then start his exploration by selecting and deselecting clusters or single features which are then shown in a 3-D view using semi-transparent surfaces. A first impression of the interface is given in Fig. 1.

4.1. Elementary features

Every area of research has its own definitions of a feature, hence numerous feature detection criteria exist and no absolute definition is available. In this paper we define a feature to be the region contained inside the largest contour of a local extremum (see Schneider et al., 2008).

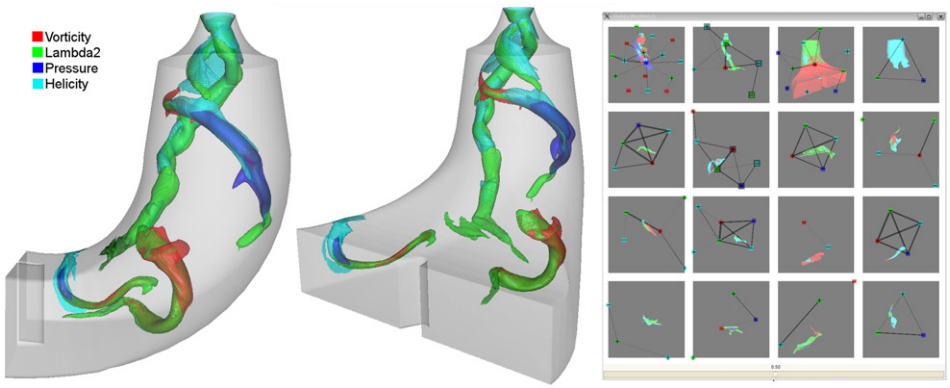


Fig. 1. Overview of the interface: The two leftmost images show the 3-D window from two different perspectives. The right images show the feature browser with the thumbnail gallery and the subgraph for each thumbnail blended together. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

The contour tree is an efficient data structure to compute the largest contour segmentation of a dataset: in it each largest contour is represented by a leaf node together with the incident tree edge. Moreover, the concept of largest contours is generalized since contour trees can be simplified to remove topological noise. Furthermore, the contour tree segments the domain, which means features of the same field do not overlap.

We start by computing the contour tree independently for each scalar field and simplify all contour trees using the same volume threshold. We chose volume rather than persistence or normalized persistence, because it gave the best results for the datasets we considered. Our choice can also be motivated from a theoretical point of view: the remainder of our pipeline only uses the domain-segmenting abilities of the contour tree and this aspect is more dependent on the region each of the contour tree's edges represents than its isovalues. We also chose to use the same threshold for all contour trees as this avoids the burden on users to provide one parameter per field and also because it unifies the minimum feature size.

Finally we create a list of features for each field by traversing the simplified contour trees and adding a new feature for each leaf. We also store the volume of each feature as the volume of the incident contour tree edge. For volume computation in general we use the fast vertex count heuristic, i.e. the volume of a largest contour is set to the number of vertices contained inside it.

4.2. Feature groups

We use mutual information to measure the dependency between elementary features in different scalar fields. For this we define for each feature a random variable, whose outcome is 1 if the feature is present and 0 otherwise. We assume that the given scalar field is the result of a random experiment carried out at each point of the domain. We can then estimate the probability that a random point is inside the feature simply as the volume of the feature divided by the volume of the domain.

Because the contour tree segments the domain we only consider feature dependency for pairs of different fields. To compute these dependencies, we estimate the shared volume between features by vertex counting. From these volumes the probabilities necessary for computing the mutual information are computed by normalizing with respect to the domain volume.

For each pair we compute normalized mutual information (Yao, 2003):

$$sim(X,Y) = \frac{I(X;Y)}{\min(H(X),H(Y))},$$

and store it as edge weights in a graph on the set of features as nodes. The choice of this similarity measure over normalized spatial overlap (e.g. using Jaccard index) has multiple benefits. First, it automatically contains a measure of overlap significance, e.g. two small features which overlap fully have a lower value than two large features which overlap fully. The Jaccard index would have been 1 in both cases. Second, it captures higher level interaction between features such as if the presence of a feature in one field determines the absence of a feature in another field or the presence of a feature at a different location. This is possible because mutual information also takes the complementary events into account. There would be no spatial overlap in the aforementioned situations, but we still might be interested in this relation. However, this property is more interesting for the time-dependent setting. Third, because mutual information always considers all possible outcome combinations of a random variable, it is no longer necessary to define the "inside" and the "outside" of a contour, thus making the measure apply to situations where this choice is not unique. Fourth, mutual information can take the actual scalar values into account and measure the dependency even for different value ranges. This property has been exploited successfully for example in the registration of multimodal medical images by Luan et al. (2008). For us it provides a ground for future extensions of the method.

Initially we chose mutual information, because we wanted a measure of similarity for more than two items. We discarded this approach since the combinatorial explosion of possible feature combinations caused excessive computation times. We argue that where mutual information is based on spatial overlap, higher order combinations must result from lower order combinations. We also note that problems in visual presentation arise: e.g. when showing a 4-combination of features with significant similarity, should significant 3-combinations of these features be shown too? We therefore select pairwise similarity to minimize computation time and visual overload, and because it allows clusters which involve multiple features of the same field.

The feature graph is a weighted k-partite graph, where k denotes the number of scalar fields. Each feature is represented as one node in this graph and the edges store the two partaking features' normalized mutual information. On this graph we apply graph clustering to find strongly dependent groups of features. Our method uses the *Chinese Whispers* algorithm (Biemann, 2006) because it is parameter free, very fast, and very simple to implement. Other implementations may chose to use other clustering algorithms. The result of the clustering are a number of subgraphs where features strongly depend on each other, i.e. they describe a region of a good local correlation between the scalar fields.

4.3. The feature browser

Exploration of the data starts with a thumbnail gallery, which provides a quick overview of all clusters. For a multivariate fluid flow dataset each cluster roughly matches one physical phenomenon (see Section 5).

Each thumbnail represents a single cluster visually by showing the contours of all features in the cluster from one user-provided perspective. This choice shows global location and orientation of clusters: alternatively, perspectives could be chosen automatically, following Schneider et al. (2008). Finally, the order of the thumbnails in the gallery is based on the total volume over all features in the cluster, starting in the upper left corner.

The interactive comparison interface consists of a direct visualization of the clustered subgraphs. Each thumbnail's area in the gallery provides the boundary for a layout of the corresponding cluster. For this layout, we use a simple spring embedder (Fruchterman and Reingold, 1991). The ordering of the subgraphs introduces considerable visual clutter as intercluster edges cross through the visualization. Therefore, we omit the intercluster edges from the drawing. Nodes in the subgraphs represent features and are drawn as a colored glyph. The color of each glyph represents its membership to a certain scalar field. A "plus" or "minus" glyph shows if the feature is a maximum or a minimum, respectively. The glyphs, i.e. the features, can be selected individually and then the surrounding contour is displayed as a transparent surface in a separate 3-D window in the same color as the glyph. A legend in the corner of the 3-D window maps colors to scalar field names.

A slider is located at the bottom of the browser window which seamlessly allows to blend between the two visual representations of the thumbnail gallery and the direct visualization of the feature graph. The advantage of this is that both visualizations can be visible at the same time. Since the subgraphs of the clusters are laid out in the area of the thumbnail the display of both provides a visual association between the thumbnail and the cluster.

4.4. Simplification according to maximum mutual information

For a low simplification threshold (volume) the contour trees have many small extrema and potentially few overlaps. For a high threshold there will only be a few big largest contours and eventually only one largest contour representing the whole dataset. Counter-intuitively, the overall mutual information is zero in that case. This is because the uncertainty is zero which means the joint entropy is zero as well. It follows that the function reaches its global maximum somewhere in between (see Fig. 2).

To find an optimum simplification value, we create one list with all branches of each contour tree. For this set of contour trees we compute the largest contours, their volumes and similarities. This list is sorted by local geometric measure, in our case volume. The branch with the smallest measure is selected and removed from its contour tree. This process is repeated until each contour tree consists of just one branch and we remember the simplification threshold at which the sum of all similarities was a global maximum. Note that each step in this iterative process replaces two or more features of one contour tree by a new feature and thus only the similarities of the new features with the features that are currently present in other fields have to be computed and the total similarity updated.

5. Examples of multifield data exploration

The interactive comparison technique described above can be used in different ways to analyze and explore multifield data. In the following we want to describe two of them in more detail focusing on fluid flow datasets.

5.1. Identifying physical phenomena

The first application is to obtain an automatic overview of all phenomena occurring in the bluntfin dataset. In the feature browser one can examine the phenomena and explore the contribution of the different variables.

Our method performed very well on the bluntfin dataset. In this dataset, flow moves over a plate and around an obstacle commonly known as the blunt fin. The dataset contains two major flow features: a shock front and a vortex below.

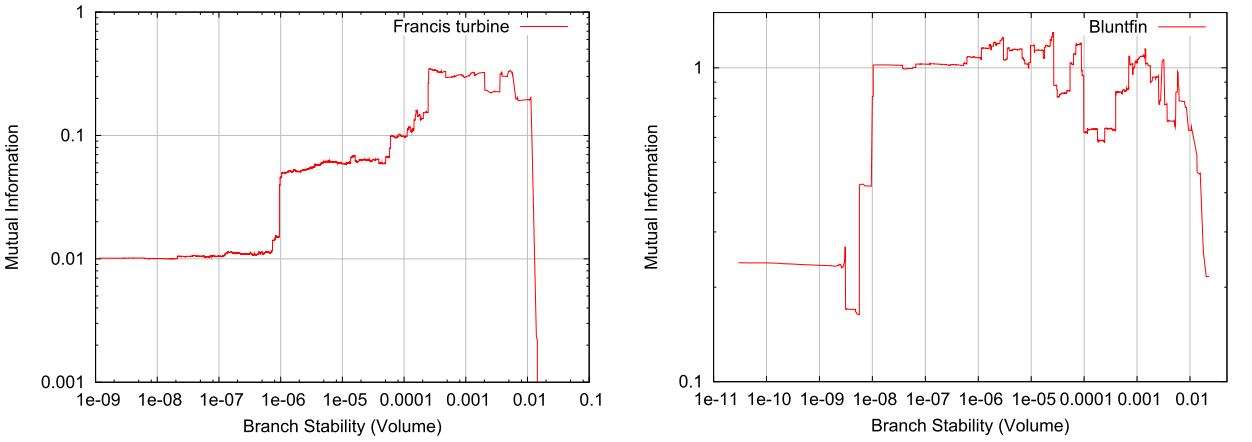
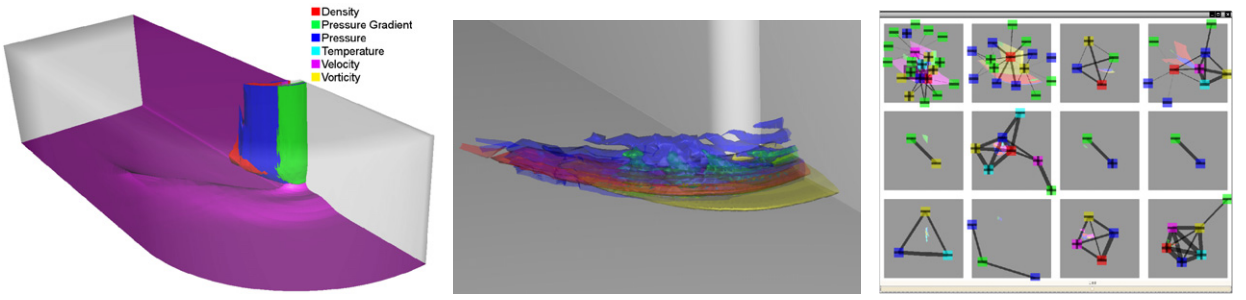


Fig. 2. Simplification diagram for finding the threshold with maximal global mutual information.



(a) Shock front overview (b) Vortex feature close-up (c) Feature browser

Fig. 3. Comparative analysis and visualization of the bluntfin dataset. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

The dataset is defined on 40.960 vertices and consists of 224.874 cells. The time for finding the optimal threshold (see Section 4.4) took 11 min. The optimal relative threshold as a fraction of the whole volume was $2.567 \cdot 10^{-5}$. The time to generate the clustering and the thumbnails was 3 min.

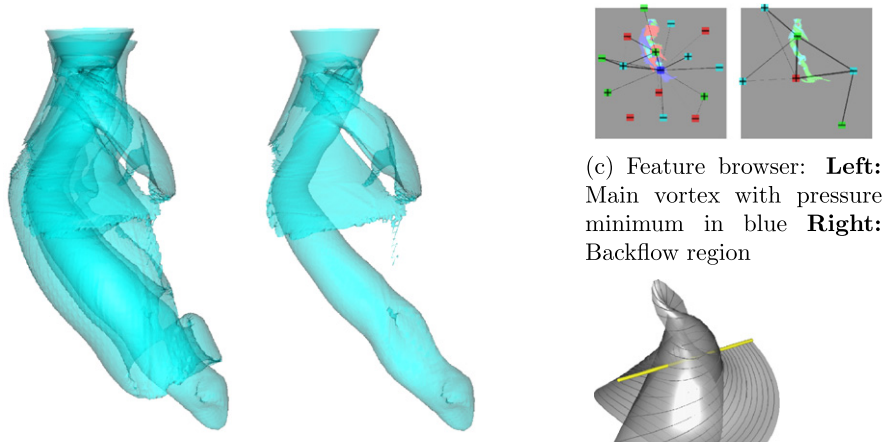
The first thumbnail in the feature browser contains the shock front (see Fig. 3(c)). It consists of a pressure gradient maximum (green), a density maximum (red), a velocity minimum (magenta) and a local pressure maximum (blue) (see Fig. 3(a)). The cluster contains also a temperature maximum (cyan) but this is mainly located at the fin. The fin causes another pressure gradient maximum attached to it. Vorticity features (yellow) do not seem to play a major role as they are rather small and scattered over the shock front.

The third thumbnail contains the vortex right below the shock front in the corner between fin and plate. It is easily identified through the vorticity maximum (yellow). Density (red), pressure gradient (green) and pressure (blue) exhibit a minimum at the same location (see Fig. 3(b)).

5.2. Comparing different vortex criteria

The Francis turbine dataset is a CFD simulation of a water turbine defined on about 1 million vertices with 1 million arbitrarily-shaped hexahedron cells. The hexahedron cells have been converted to 5.7 million tetrahedrons for the contour tree computation. The water enters the turbine on the top through the so-called Francis runner and leaves it at the bottom through two rectangular openings. The dataset is very complex with one large main vortex attached to the runner and several smaller vortices. The time for finding the optimal threshold (see Section 4.4) took 19 h and 38 min. The optimal relative threshold as a fraction of the whole volume was $2.601 \cdot 10^{-4}$. The time to generate the clustering and the thumbnails was 1 h.

The dominating feature in this dataset is a large vortex attached to the runner, i.e. the top of the dataset. It extends almost to the two openings at the bottom. The interesting thing about this vortex is, that it consists of two motions. One swirling region advecting fluid downwards (see Fig. 4(a)) and another inner structure transporting fluid upwards (see Fig. 4(b)) in the direction of the runner. These two structures have strong helicity extrema but with opposite sign in the feature browser (see Fig. 4(c)). The result of that is that the feature is split up into two. This means two clusters



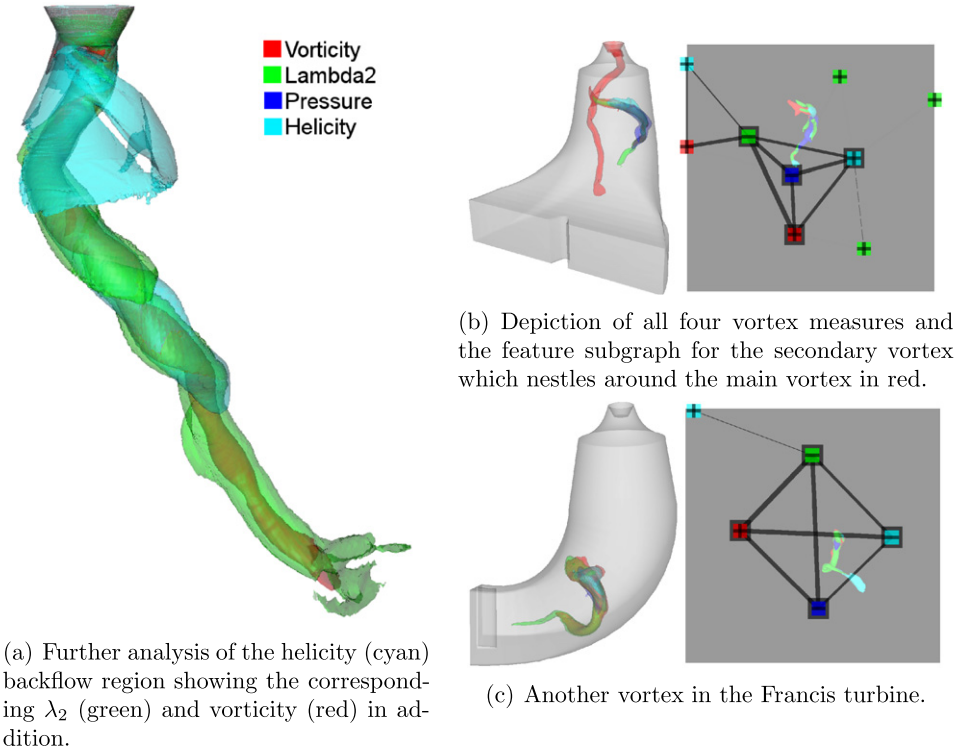
(a) Vortex and backflow region together

(b) Backflow region

(c) Feature browser: **Left:** Main vortex with pressure minimum in blue **Right:** Backflow region

(d) Stream surface

Fig. 4. Analysis of the main vortex with backflow region. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)



(a) Further analysis of the helicity (cyan) backflow region showing the corresponding λ_2 (green) and vorticity (red) in addition.

(b) Depiction of all four vortex measures and the feature subgraph for the secondary vortex which nestles around the main vortex in red.

(c) Another vortex in the Francis turbine.

Fig. 5. Further investigation of features in the Francis turbine dataset. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

appear describing the different aspects of the fluid motion, although both features reside within the large pressure minimum in blue. To back this we integrated a stream surface right through the middle of the vortex and the backflow region (see Fig. 4(d)). As one can see the middle part of the stream surface is moving upwards whereas the outer parts move downwards. This is consistent with the fact that the outer structure contains a tunnel where the fluid is transported upwards (see Fig. 4(a)). The backflow region can now be investigated further with respect to other vortex criteria like vorticity or λ_2 (see Fig. 5(a)).

Together with the main vortex exists another vortex which is entangled with the main vortex and winds around it (see Fig. 5(b)). There exist numerous other vortices in this dataset which can be investigated; some of them are attached to the boundary of the turbine (see Fig. 5(c)).

6. Conclusion and future work

We described a novel interface for the interactive comparison of multiple scalar fields based on features. We introduced a new data structure called the feature graph on which we perform graph clustering employing mutual information as similarity measure between features to find coherent feature groups. We argued that the use of clustering on the basis of binary similarity is preferable to similarity measures of more than two items, because it allows simpler presentation and avoids combinatorial explosion. A gallery shows the feature graph with a layout that emphasizes clusters and a 3-D contour rendering thumbnail for each cluster. It provides an overview over the feature groups and allows for the selection of single features and feature groups to be shown in a main view for further exploration. Thus it enables the user to perform both global and local comparisons of features.

Furthermore, we presented a method to automatically determine the threshold by which all contour trees are simplified topologically. Our algorithm finds the threshold at which the dependency between the different scalar field features reaches its maximum. Using one threshold for all fields limits our choice on simplification measure. Compared to persistence and normalized persistence, volume gave most meaningful results. Physical phenomena can be disproportionately expressed in the different scalar fields, but always occupy similar regions. However, using volume inadvertently sets the level of detail and very small features cannot be studied. Hypervolume (Carr et al., 2004) is an interesting alternative as it combines volume and persistence. Also, setting the threshold for each field separately could improve investigation, but would lead to either a large burden on the user or a combinatorial explosion in the automatic search. An alternative would be a manual per-field adaptation with an automatically determined global initial value. It might even be feasible to split, i.e. unsimplify, features on demand. Optimally, we would like the contour trees to negotiate their simplification automatically among each other, which we will investigate in future work.

The thumbnails in the gallery give an overview by hinting at the position, size and shape of feature groups. In-depth analysis requires to show a select few in a larger interactive 3-D view to avoid occlusions when showing all. The more feature groups there are the smaller each thumbnail and the lower its expressivity becomes. As an alternative it would be interesting to show all feature groups in a 3-D view but move them apart like motor parts in technical illustrative drawings.

As the single features that make up a feature group have to be shown in an overlapping fashion, rendering their contours typically uses semi-transparent surfaces to avoid occlusion problems and help visual comparison (Weigle and Taylor, 2005; Schneider et al., 2008). For more than two surfaces this poses perceptual challenges. An alternative would be to split the feature volumes into core regions and show these. However, it is not clear how many of the k features have to partake in a core structure to be interesting and how to deal with disconnected core structures. Furthermore this overemphasizes commonality at the expense of individuality. In the future we want to investigate how volume rendering can alleviate the perceptual challenges of multiple transparent surfaces.

Apart from the points mentioned above, we also want to add the actual isovalue distributions into the similarity measure and combine the ideas of clustering and higher order similarity measures. Finally, we want to extend this work to time-dependent data.

Acknowledgements

We thank Ronald Peikert and VA Tech for the Francis turbine dataset and NASA for the Bluntfin dataset. This work was partially supported by grant DFG SCHE 663/3-8.

References

Akiba, H., Ma, K.L., 2007. A tri-space visualization interface for analyzing time-varying multivariate volume data. In: Proceedings of Eurographics/IEEE VGTC Symposium on Visualization, pp. 115–122.

Biemann, C., 2006. Chinese whispers – an efficient graph clustering algorithm and its application to natural language processing problems. In: Proceedings of the HLT-NAACL-06 Workshop on Textgraphs, New York, USA.

Boyell, R.L., Ruston, H., 1963. Hybrid techniques for real-time radar simulation. In: Proceedings of the 1963 Fall Joint Computer Conference. IEEE, pp. 445–458.

Bremer, P.T., Weber, G., Pascucci, V., Day, M., Bell, J., 2010. Analyzing and tracking burning structures in lean premixed hydrogen flames. IEEE Transactions on Visualization and Computer Graphics 16, 248–260.

Bürger, R., Hauser, H., 2007. Visualization of multi-variate scientific data. In: EuroGraphics 2007 State of the Art Reports, pp. 117–134.

Carr, H., Snoeyink, J., 2003. Path seeds and flexible isosurfaces using topology for exploratory visualization. In: VISSYM'03: Proceedings of the Symposium on Data Visualisation 2003. Eurographics Association, Aire-la-Ville, Switzerland, pp. 49–58.

Carr, H., Snoeyink, J., Axen, U., 2003. Computing contour trees in all dimensions. Computational Geometry 24, 75–94.

Carr, H., Snoeyink, J., van de Panne, M., 2004. Simplifying flexible isosurfaces using local geometric measures. In: Proc. IEEE Visualization'04. IEEE Computer Society, Washington, DC, USA, pp. 497–504.

Fruchterman, T.M.J., Reingold, E.M., 1991. Graph drawing by force-directed placement. Software Practice and Experience 21, 1129–1164.

Fuchs, R., Waser, J., Gröller, M.E., 2009. Visual human + machine learning. IEEE Transactions on Visualization and Computer Graphics 15, 1327–1334.

Gosink, L.J., Anderson, J.C., Bethel, E.W., Joy, K.I., 2007. Variable interactions in query-driven visualization. IEEE Transactions on Visualization and Computer Graphics 13, 1400–1408.
Jänicke, H., Böttinger, M., Scheuermann, G., 2008. Brushing of attribute clouds for the visualization of multivariate data. IEEE Transactions on Visualization and Computer Graphics 14, 1459–1466.
Jänicke, H., Wiebel, A., Scheuermann, G., Kollmann, W., 2007. Multifield visualization using local statistical complexity. IEEE Transactions on Visualization and Computer Graphics 13, 1384–1399.
Jeong, J., Hussain, F., 1995. On the identification of a vortex. Journal of Fluid Mechanics, 69–94.
Kniss, J., Kindlmann, G., Hansen, C.D., 2001. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In: Proc. of Visualization 2001, pp. 255–262, 562.
Lampe, O.D., Correa, C., Ma, K.L., Hauser, H., 2009. Curve-centric volume reformation for comparative visualization. IEEE Transactions on Visualization and Computer Graphics 15, 1235–1242.
Linsen, L., Long, T.V., Rosenthal, P., 2009. Linking multidimensional feature space cluster visualization to multifield surface extraction. IEEE Computer Graphics and Applications 29, 85–89.
Luan, H., Qi, F., Xue, Z., Chen, L., Shen, D., 2008. Multimodality image registration by maximization of quantitative-qualitative measure of mutual information. Pattern Recognition 41, 285–298.
Manders, E.M.M., Hoebe, R., Strackee, J., Vossepoel, A.M., Aten, J.A., 1996. Largest contour segmentation: A tool for the localization of spots in confocal images. Cytometry 23, 15–21.
Sauber, N., Theisel, H., Seidel, H.P., 2006. Multifield-graphs: An approach to visualizing correlations in multifield scalar data. IEEE Transactions on Visualization and Computer Graphics 12, 917–924.
Schneider, D., Wiebel, A., Carr, H., Hlawitschka, M., Scheuermann, G., 2008. Interactive comparison of scalar fields based on largest contours with applications to flow visualization. IEEE Transactions on Visualization and Computer Graphics 14, 1475–1482.
Weber, G.H., Dillard, S.E., Carr, H., Pascucci, V., Hamann, B., 2007. Topology-controlled volume rendering. IEEE Transactions on Visualization and Computer Graphics 13, 330–341.
Weigle, C., Taylor II, R.M., 2005. Visualizing intersecting surfaces with nested-surface techniques. In: IEEE Visualization. IEEE Computer Society, p. 64.
Wilhelm, A., 2008. Handbook of Data Visualization. Springer Handbooks of Computational Statistics. Springer.
Woodring, J., Shen, H.W., 2006. Multi-variate, time varying, and comparative visualization with contextual cues. IEEE Transactions on Visualization and Computer Graphics 12, 909–916.
Yao, Y.Y., 2003. Information-Theoretic Measures for Knowledge Discovery and Data Mining. Springer.