ADVANCED VISUALIZATION OF ENGINE SIMULATION DATA USING TEXTURE SYNTHESIS AND TOPOLOGICAL ANALYSIS

Guoning Chen(1), Robert S. Laramee(2) and Eugene Zhang(1)

1. Oregon State University[1], 2. University of Wales Swansea[2]

SUMMARY

We present a selection of advanced flow visualization techniques in order to investigate, analyze, and present the simulation of flow inside an automotive engine. More specifically, our analysis and visualization is targeted at the simulation of in-cylinder flow, namely, the visualization of swirl and tumble motion found inside diesel and gas engines, respectively.

Texture-based flow visualization techniques use texture-mapping and synthesis in order to reflect the characteristics of the underlying vector field and have the advantage of providing complete coverage of the geometric domain. We present an advanced texture-based flow technique that can be used to visualize the flow from engine simulation data and advects texture properties in order to depict the underlying fluid motion. Next we apply topology-based flow visualization methods that allow the engineer to focus on a subset of the fluid flow, e.g., the fixed points in the vector field. Our topology-based flow analysis first extracts the fixed points in a vector field, automatically, and then shows the relationships between those fixed points. Furthermore, we show how vector field simplification can be used to aid the engineer in gaining insight into their simulation results.

1: INTRODUCTION

Computer simulation has become an integral part of many engineering design and testing process. In engine design, data obtained through simulation, once understood, can not only enable engine designers to detect flaws in the current design but also provide hints on how they might be corrected or improved. However, properly interpreting engine simulation datasets is a challenging problem, partly due to the

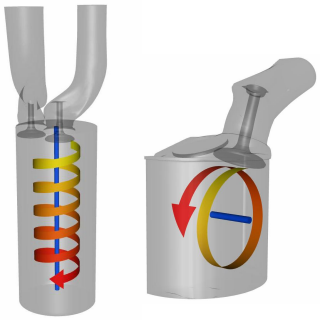

Figure 1: Idealized in-cylinder flow through a diesel engine (left) and a gas engine (right).

[1] School of Electrical Engineering and Computer Science, Oregon State University, 1148 Kelley Engineering Center, Corvallis, OR 97331,USA, {chengu, zhange}@eecs.oregonstate.edu

[2] Department of Computer Science, Swansea University, SA2 8PP, Wales, United Kingdom, Laramee@swansea.ac.uk

limitations of data analysis and visualization tools that are being employed. In this paper, we present a selection of advanced flow visualization techniques in order to investigate, analyze, and present the simulation of flow inside an automotive engine. More specifically, our analysis and visualization is targeted at the simulation of in-cylinder flow, namely, the visualization of swirl and tumble motion found inside diesel and gas engines.

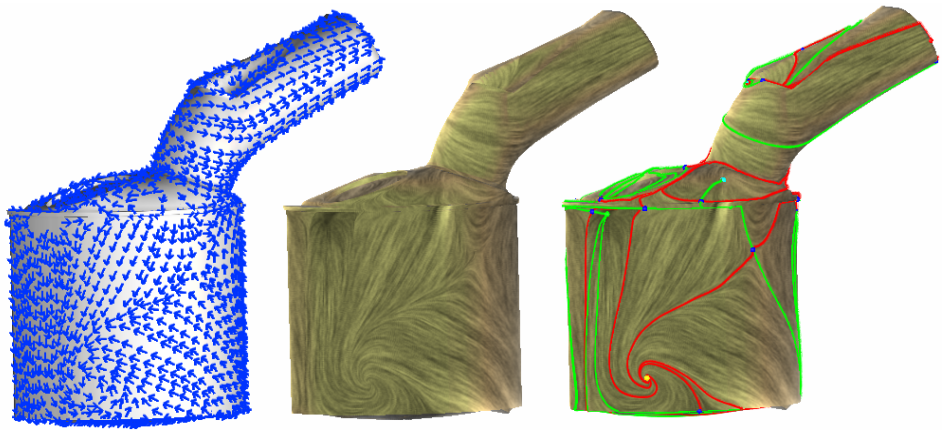

Figure 2: Comparison of different vector field visualization techniques for a simulated tumble motion for a gas engine: (left) an arrow plot of the tumble motion dataset, (middle) a texture-based vector field visualization (IBFVS), (right) the texture-based visualization of the same data set with vector field topology being highlighted. (green dots=sources, red dots =sinks, orange dots=attracting focus, cyan dots=repelling focus and blue dots=saddles. Green curves indicate the separatrices connecting saddle to sources and red curves indicate the separatrices connecting saddles to sinks).

One type of flow, referred to as the *swirl motion*, is shown in Figure 1 (left). In this case, engineers responsible for the design of a diesel engine try to create an ideal pattern of motion, which can be described by a swirling flow around an imaginary axis aligned with the valve cylinder. Another type of motion, termed *tumble flow*, is shown in Figure 1 (right). The axis of rotation in the tumble case is orthogonal to that of the swirl case. Achieving these ideal patterns of flow optimizes the mixture of oxygen and fuel during the ignition phase of the valve cycle. Optimal ignition leads to very desirable consequences associated with the combustion process including: more burnt fuel (less wasted fuel), lower emissions, and more output power [2][8].

The datasets this paper experiments on are the boundary velocity fields and are obtained through simulation on the designed engines. What engineers expect to see is whether the flows on the surface follow the ideal patterns or not. To answer such query, vector field analysis and visualization techniques can provide the critical insights that would have been difficult to obtain otherwise. To illustrate this point, let us consider the simulated tumble dataset shown in Figure 2. To visualize the flow datasets, direct visualization techniques, such as arrow plots, are firstly used due to its simplicity (Figure 2, left). But it is difficult for engineers to imagine the continuous flow patterns from the

discrete samples. People prefer a visualization technique that can show the flow continuously over the whole engine surfaces. Texture-based visualization methods are then proposed. Due to the continuity of texture images (at the pixel level), texture-based visualization techniques have the ability to show the underlying flow completely over the whole geometric domain (Figure 2, middle). In this paper, we present one of the latest texture-based visualization techniques that are based on the advection of texture properties in order to depict the underlying fluid motion. It also has been extended to visualize the boundary surface vector fields which we employ to visualize the engine simulation datasets.

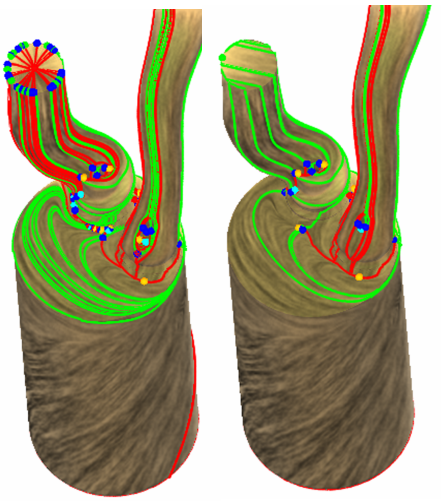

Figure 3: The visualization of a simulated swirl motion inside a diesel engine before (left) and after (right) simplification

Although texture-based visualization techniques have the advantage of the complete coverage of the geometric domain of the vector field, it is often difficult to discern the core features in the field from the output images (Figure 2, middle), especially for a dataset as complex as the one showed in Figure 3 (left). Vector field topology can be an efficient means for this problem. It highlights the essential structure of the vector field (Figure 2, right). Therefore, topology-based vector field analysis and visualization techniques are indispensable in complementing the texture-based vector field visualization techniques. In this paper, we will introduce the basic concepts of vector field topology and the approach to compute the topology from the given simulated datasets.

A large simulation dataset, such as the simulated swirl motion shown in Figure 3 (left), on one hand, usually contains noise that may lead to spurious features in the flow. Furthermore, engineers are often concerned about the flow behavior under a larger scale. For example, engineers may want to know the overall flow behavior at the top of the left intake port instead of the individual fixed points on it (Figure 3, left). Therefore, any vector field processing tools should also provide user with means to reduce noise and provide multiple-scale visualization of the flow. Vector field simplification can be used to reduce the noise and allow engineers to focus on the major features of the vector field. In this paper, we will provide users with a tool to remove undesired features from the vector field and retain the other features (Figure 3, right).

The rest of the paper is organized as follows: Section 2 reviews some basic concepts of vector fields. Section 3 introduces the state-of-art of texture-based vector field visualization techniques and some algorithmic details behind them.

Section 4 presents the approaches to calculate vector field topology. Section 5 addresses the vector field simplification problem, which is followed by a report of the results applying these techniques to the engine simulation data in Section 6. Section 7 concludes and summaries the work.

2: VECTOR FIELD BACKGROUND

2.1 basic concepts of vector fields

a) vector fields

A vector field V on some domain $X \subset R^n$ ($n = 2$ *or* 3) is a function which associates a vector V to each point x of X at time t. $\dot{x} = V(t, x)$. If the flow is time-independent, then the velocity in the field does not depend on time ($\dot{x} = V(x)$). Otherwise, the vector field is time-dependent. In the remainder of the paper, we will only discuss time-independent flows.

b) fixed points (Helman and Hesselink [4])

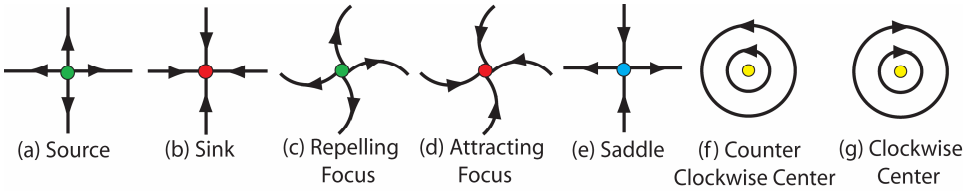

Figure 4: The classification of fixed points.

A fixed point (also referred to as singularity) in a vector field is a point in the domain where the vector value vanishes. Fixed points can be classified according to the eigen-values of the Jacobian of the vector field V with respect to the position at the fixed point.

Generally, there are four kinds of fixed points in a vector field. They are sources, sinks, saddles and centers (Figure 4).

c) trajectories and separatrices

A trajectory describes the path of a particle moving inside a vector field. A trajectory that starts from $r(t_0) = p_0$ at time t_0 is the solution of the initial value problem of an ordinary differential equation $dr(t)/dt = V(r(t))$, which can be solved by any proper integration methods. In our implementation, an adaptive fourth order Runge-Kutta method [12] is used.

Separatrices are special trajectories which start from the neighbourhoods of each saddle and follow the four eigenvector directions of the Jacobian of the saddle, two incoming and two outgoing, respectively (Figure 2, Figure 6, left). Through separatrices, we can out all the connections between saddles and other fixed points, which form the skeleton of the vector field topology [4].

2.2 vector field representation

We now describe the computational model of our system. In this model, the underlying domain is represented by a triangular mesh. Vector values are defined at the vertices only, and interpolation is used to obtain values on the edges and inside triangles. This applies to analysis such as fixed point and separatrix extraction and vector field simplification. For the planar case, we use the piecewise linear interpolation method [15]. On curved surfaces, we borrow the interpolation scheme of Zhang et al. [19], which guarantees vector field continuity across the vertices and edges of the mesh.

3: TEXTURE-BASED VECTOR FIELD VISUALIZATION

3.1 previous work

Many vector field visualization techniques have been proposed during the past decade. According to the state of art report by Laramee et al. (2004), there are four groups of vector field visualization techniques. The first group is direct flow visualization, such as arrow plots (Figure 2, left). The second one is streamline based method. This group includes the algorithms by Turk and Banks[16], Jobard and Lefer[5]. Both of these two groups of algorithms are based on discrete sampling of the vector fields. Therefore, they lack of the ability to produce continuous images of the flow. The third group is the feature-based visualization. This group performs vector field analysis first and visualizes only important features (topology) of the flow. The last group is texture-based visualization. This group includes the famous line integral convolution (LIC) by Cabral and Leedom[1], the recently introduced image-based flow visualization technique (IBFV) by van Wijk[17], and its extension to curved surfaces—IBFVS[18] and ISA[6]. Texture based flow visualization techniques use texture as the primitives to depict a vector field. They first compute the flow like textures based on the underlying vector fields. Then, they map the obtained textures onto the geometry surfaces. Therefore, the visualization results can show us complete flow patterns of the vector fields on the geometry models. In this section, we will present a texture-based visualization and its two extensions to surface applications which we use to visualize the engine simulation datasets.

3.2 texture-based vector field visualization - IBFV

Image based flow visualization (IBFV), introduced by van Wijk [17], greatly improves the performance of the texture based vector field visualization techniques. IBFV is based on advecting a noise texture according to the flow field and blending the result with the periodic generated noise. This process can be formulated by

$$F(p_k;k) = (1-\alpha)F(p_{k-1};k-1) + \alpha G(p_k;k)$$

where F represents an image, G represents the background image, p_k indicates the set of particles in frame k, α refers to the blending factor.

The background image G is noise image consisting of a series of low pass filtered white noise. The purpose of using low frequency noise is to eliminate the time consuming step, post filtering process, which is used to remove high frequency components. G is blended with the flow like image to maintain the contrast of output image. Note that larger blending factor α will make the output image more noisy, smaller α will decrease the contrast of the image. In our implementation, α is chosen between 0.05 and 0.2.

Noise images are generated as a function of time. One can consider each point G_i of the background texture as a spot or particle that periodically grows up and decays according to some function $w(t)$. The equation for noise generation is as follows.

$$G_{i,k} = w((k / M + \phi) \bmod 1)$$

Thus, the pipeline of IBFV is: 1) initialize image, 2) distort the polygon according to the vector field, 3) map the texture which is the previously saved image on the distorted mesh 4) blend fresh low pass filtered white noise into the image 5) save the current image to memory for next round procedure.

3.3 texture-based visualization on surfaces – ISA and IBFVS

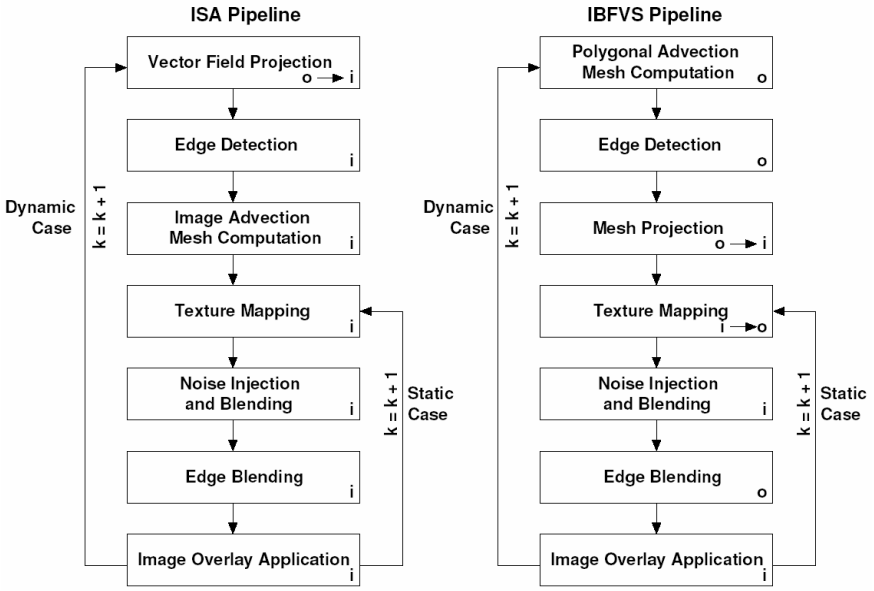

Figure 5: The processing pipeline of the texture-based flow visualization subsystem. *i* indicates the operation takes place in image space and *o* for object space. o ⟶ i refers to a transition takes between object space and image space, and vice versa.

Two extensions of IBFV have been proposed to visualize vector fields on curved surfaces. One is ISA[6], the other is IBFVS[18]. The ISA and IBFVS algorithms simplify the problem of advecting textures on surfaces by confining the advection of texture properties to image space. After a projection to image space phase, a series of textures are mapped, blended, and advected. The ISA

method for visualization of flow on surfaces is comprised of the following procedure (Figure 5, left)[9]: (1) project the vector field to the image plane, (2) detect geometric edge discontinuities, (3) compute advected texture coordinates, (4) advect the image, (5) inject and blend in noise, (6) blend additional noise along geometric edge discontinuities, and (7) apply shading and other additional graphics. The IBFVS method is very similar (Figure 5, right). It is the direct extension of IBFV for surface vector field visualization [18]. The essential difference of IBFVS from ISA is that the advected texture coordinates are computed in object space rather than image space. Steps 1-7 of the pipeline are necessary for the dynamic cases of time-dependent geometry, rotation, translation, and scaling, and only a subset is needed for the static cases (steps 4-7) involving no changes to the view-point and steady flow. Each stage is described in more detail in previous research [7].

4: VECTOR FIELD TOPOLOGY EXTRACTION

4.1 previous work

Topology-based vector field visualization has been introduced by Helman and Hesselink in 1989 [4]. They also propose efficient algorithms to extract vector field topology. Following their footsteps, much research has been conducted in topological analysis of vector fields. For example, Scheuermann et al. [13] use Clifford Algebra to study the non-linear singularities of a vector field and propose an efficient algorithm to merge nearby first-order singularities. Tricoche et al. [15] and Polthier and Preuβ [10] present efficient methods to locate fixed points in a vector field. More related work can be found in [11].

4.2 fixed point extraction

Under the setting stated in Section 2.2, the vector field inside each triangle satisfies following linear equations.

$$\begin{cases} F(x,y) = ax+by+c \\ G(x,y) = dx+ey+f \end{cases}$$ (*a, b, c, d, e, f* are unknown coefficients and determined based the coordinates of the three vertices and the corresponding vector values.

If the triangle contains a fixed point, there will be some point inside the triangle then the same point is a solution to the equation $F(x,y)=G(x,y)=0$. This can be used to judge whether a triangle contains a fixed point or not. Considering efficiency, Poincaré index theory is applied to help locating the triangles containing fixed points [15].

For surface vector fields, it becomes more difficult to locate fixed points. Interested readers please refer to Zhang et al.'s work [19] for details on how to locate fixed points in surface vector fields.

4.3 separatrix calculation

Since eparatrices are special trajectories starting from saddles, after calculating the eigen-vectors of all saddles, we can computer four points close to each saddle to trace the separatrices from them [19].

5: VECTOR FIELD SIMPLIFICATION

5.1 pevious work

Vector field simplification is used to reduce the complexity of a vector field. The common process of vector field simplification is: calculate a region containing the features of interests; then, replace the original field with a simpler vector field. There are two classes of simplification techniques: topology-based (TB), and non-topology-based (NTB) [19]. The TB methods calculate the region according to some topology connection in the topological skeleton of the vector field. The NTB methods calculate the region based on the user specified boundaries of the region. Existing NTB techniques are usually based on performing Laplacian smoothing on the potential of a vector field inside the specified region [14]. TB techniques simplify the topology of a vector field explicitly. Tricoche et al.[15] simplify a planar vector field by performing a sequence of cancellation on fixed point pairs that are connected by a separatrix. Zhang et al. [19] provide a singularity pair cancellation method based on Conley theory.

5.2 user-guided simplification

User-guided simplification is one of the NTB vector field simplification. The inputs of it are the underlying mesh, the vector field defined on the mesh, and the user specified region boundaries. The basic idea of the algorithm is to first find all the inner vertices and boundary vertices of the region enclosed by the user specified boundaries, and then perform the Laplacian smoothing on the vector values of the inner vertices based on the constraints of the boundary vertices. Figure 6 shows an example of user-guided simplification.

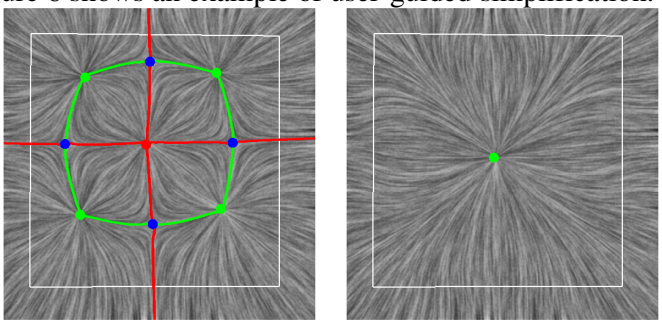

Figure 6: An example of user-guided simplification. The white rectangle infers to the user specified region.

The vector-valued discrete Laplacian equation over a region N in the domain has the following form, where the vector values at the boundary vertices of N are the constraints:

$$\overline{V}(v_i) = \sum_{j \in J} \omega_{ij} \overline{V}(v_j)$$

where v_i is an interior vertex, v_j's are the adjacent vertices that are either in the interior or on the boundary of N, and V represents the vector field. The weights

ω_{ij}'s are determined using Floater's mean-value coordinates [3]. The equation above is a sparse linear system, which we solve by using a conjugate gradient method [12]. For convenience, we refer to a vertex *v* as being *fixed* if the vector value at *v* is part of the constraints. Otherwise, *v* is *free*.

6: RESULTS

6.1 visualization and Topology Analysis

We have applied the aforementioned techniques to the two engine simulation datasets. In Figure 7 we visualize the flow and its topology inside the combustion chamber from the diesel engine simulation. We have sliced through the geometry in the same manner that engineers do when analyzing the simulation results. The first slice, at 10% the length of the volume, indicates a swirl pattern that deviates rather strongly from the ideal which would result in a simple recirculation orbit around the center. The second slice, at 25% down the chamber geometry, shows a closed orbit close to the center that starts to approximate the ideal swirl motion. However, other less ideal singularities are found near the perimeter of the geometry. Figure 2, 3, 8 and 9 show the visualization and analysis results of the two datasets on the boundary surfaces of the engines, respectively. The topology analysis results for the two datasets are shown in following table. The experiments are carried out on a PC with 3.6GHZ CPU and 3GB RAM.

dataset name	# of polygons	# of fixed points	times extracting fixed points (s)	time computing separatrices(s)	time total(s)
Gas engine	105,192	56	0.16	9.09	9.25
Diesel engine	886,296	226	3.16	4.48	7.64

6.2 user-guided simplification

We have applied the user-guided simplification to large scale CFD simulation datasets. Unlike Zhang et al. [19] who accept a topological disk, we now allow a region to have any number of boundaries. Figure 8 shows the results of user-guided flow smoothing on the engine simulation data of in-cylinder flow in a gas engine. The field on the right was obtained by a sequence of five user-guided smoothing operations (the actual region boundaries are not shown). Notice the field is considerably simpler than the original field (Figure 8, middle). The simplified vector field retains the important larger scale tumble motion characteristics while smoothing non-ideal behavior. Figure 9 compares the diesel engine dataset (middle) with the one obtained from a series of six user-guided simplification operations (right). The results show that flow smoothing is an efficient method of reducing the complexity of a vector field.

7: CONCLUSION

In this paper, we have explored and visualized a number of engine simulation datasets to help engineers evaluate the quality of the engine design. Vector

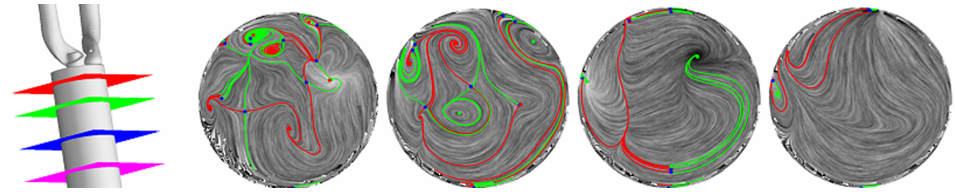

Figure 7: Visualizing the simulation of flow in a diesel engine: the combustion chamber (leftmost) and four planar slices of the flow inside the chamber for which the plane normals are along the main axis of the chamber. From left to right are slices cut at around 10%, 25%, 50%, and 75% of the length of the cylinder from the top where the intake ports meet the chamber. The vector fields are defined as zeros on the boundary of the geometry (*no-slip condition*). The automatic extraction and visualization of flow topology allows the engineer to gain insight into where the ideal pattern of swirl motion is realized inside the combustion chamber. In fact, the behavior of the flow and its associated topology is much more complicated than the ideal. Figure 3 provides complementary visualization of the flow on the boundary of the diesel engine.

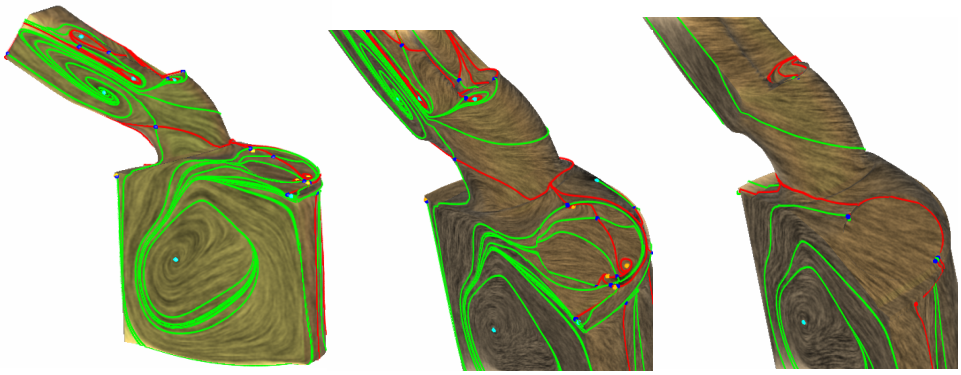

Figure 8: The flow-like images show the tumble motion boundary vector field (left). The two images to the right compare the results of user-guided simplification. The right most field is obtained after 5 round user-guided simplification.

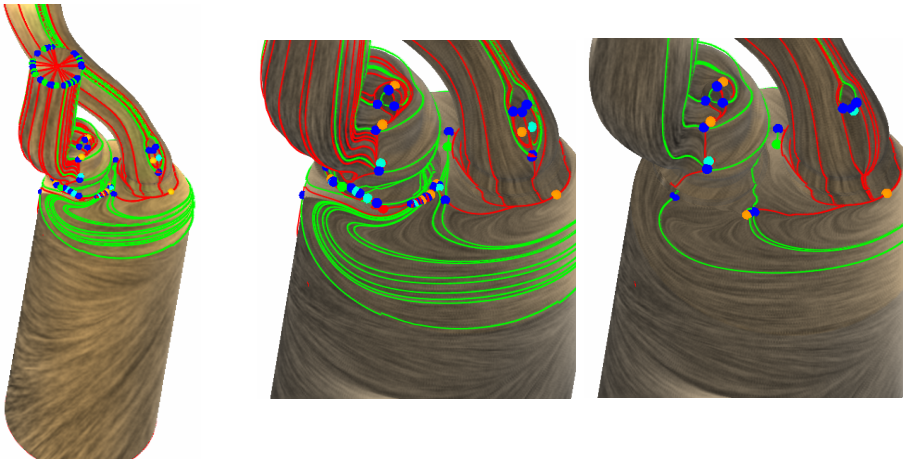

Figure 10: The flow-like images show the swirl motion boundary vector field (left). The two images to the right compare the results of user-guided simplification. The right most field is obtained after 6 round user-guided simplification.

field visualization, analysis and simplification are combined to get the optimal visualization of the simulation datasets. To provide engineers a complete view of the flow behavior of the datasets, we have presented a texture-based visualization technique. Complementing this texture-based visualization technique, vector field topology has been extracted to help engineers identify the essential structures of the flow. To compute the vector field topology, we first extract all the fixed points in the field, then we calculate the separatrices starting from each detected saddle. These separatrices will build the connections between pair of fixed points, which form the skeleton of the vector field topology.

Due to the numerical inaccuracy during simulation, noise may be included in the datasets. Our tool provides engineers an approach to reduce the noise by simply specifying a region on the domain and performing smoothing operation. The results show that this user-guided vector field simplification does help to reduce noise and achieve better visualization results.

ACKNOWLEDGEMENT

We would like to thank Christoph Garth, Konstantin Mischaikow, Juergen Schneider and Greg Turk for their valuable contributions.

REFERENCES

1. CABRAL, B. and LEEDOM, L. C. - Imaging vector fields using line integral convolution. Proceedings of ACM SIGGRAPH 93, pp. 263-272, 1993.
2. CHEN, G., MISCHAIKOW, K., LARAMEE, R.S., PILARCZYK, P. and ZHANG, E. - Vector field editing and periodic orbit extraction using morse decomposition. IEEE Transactions on Visualization and Computer Graphics (to appear), Technical report available at: *http://eecs.oregonstate.edu/library/files/2006-27/vecfldedit_tr.pdf*, 2006.
3. FLOATER, M. S. - Mean value coordinates. Computer Aided Geometric Design, **20**(1), pp.19-27, 2003.
4. HELMAN, J. L. and HESSELINK, L. - Representation and display of vector field topology in fluid flow data sets. IEEE Computer, **22**(8), pp.36-46,1989.
5. JOBARD, B. and LEFER, W. - Creating evenly-spaced streamlines of arbitrary density. EG Workshop on visualization in scientific computing, pp.43-56, 1997.
6. LARAMEE, R. S. and JOBARD, B. and HAUSER, H. - Image space based visualization of unsteady flow on surfaces. Proceedings IEEE Visualization 03, pp. 131-138, 2003.

7. LARAMEE, R. S. and HAUSER, H. and DOLEISCH, H. and POST, F. H. and VROLIJK, B. and WEISKOPF, D. - The state of the art in flow visualization: dense and texture-based techniques. Computer Graphics Forum, **23**(2), pp.203-221, 2004.
8. LARAMEE, R. S.and WEISKOPF, D. and SCHNEIDER, J. and HAUSER, H. - Investigating swirl and tumble flow with a comparison of visualization techniques. Proceedings IEEE Visualization 04, pp. 51-58, 2004.
9. LARAMEE, R. S. and HADWIGER, M. and HAUSER, H. - Design and implementation of geometric and texture-based flow visualization techniques. Proceedings of the 21st Spring Conference on Computer Graphics 2005(SCCG 2005), pp. 67-74, 2005.
10. POLTHIER, K. and PREUSS, E. - Identifying vector fields singularities using a discrete hodge decomposition. In H.C. Hege, K. Polthier (eds). Mathematical Visualization III, pp. 112-134, 2003.
11. POST, F. H. and VROLIJK, B. and HAUSER, H. and LARAMEE, R. S. and DOLEISCH, H. - The state of the art in flow visualization: feature extraction and tracking. Computer Graphics Forum, **22**(4), pp.775-792, 2003.
12. PRESS, W. H. and FLANNERY, B. P. and TEUKOLSKY, S. A. and VETTERLING, W. T. - Numerical Recipes in C: The Art of Scientific Computing. New York, NY, USA: Cambridge University Press, 1992.
13. SCHEUERMANN, G. and KRÜGER, H. and MENZEL, M. and ROCKWOOK, A. P. - Visualizing nonlinear vector field topology. IEEE Transactions on Visualization and Computer Graphics, **4**(2), pp.109-116, 1998.
14. TONG, Y. and LOMBEYDA, and HIRANI, S. A. and DESBRUN, M. - Discrete multiscale vector field decomposition. ACM Transactions on Graphics (SIGGRAPH 03), **22**(3), pp. 445-452, 2003.
15. TRICOCHE, X. and SCHEUERMANN, G. and HAGEN, H. - Continuous topology simplification of planar vector fields. Proceedings of IEEE Visualization 01, pp. 159-166, 2001.
16. TURK, G. and BANK, D. - Image-guided streamline placement, Proceedings of ACM SIGGRAPH 96, pp. 453-460, 1996.
17. VAN WIJK, J. J. - Image based flow visualization. ACM Transactions on Graphics (SIGGRAPH 02), **21**(3), pp. 745-754, 2002.
18. VAN WIJK, J.J. - Image based flow visualization for curved surfaces. Proceedings IEEE Visualization 03, pp.123-130, 2003.
19. ZHANG, E. and MISCHAIKOW, K. and TURK, G. - Vector field design on surfaces. ACM Transactions on Graphics, **25**(4), pp. 1294-1326, 2006.