Ridge Surface Methods for the Visualization of Lagrangian Coherent Structures

Ronald Peikert, Benjamin Schindler, Robert Carnecky
ETH Zurich, Dept. of Computer Science
peikert@in.ethz.ch

ABSTRACT

In the visualization of unsteady flow, Lagrangian coherent structures (LCSs) have received much attention in recent years. The standard approach is to compute the finite-time Lyapunov exponent (FTLE) on a regular grid and use a transfer function to generate a color image. In the 2D case, this is sufficient because the ridges in the image, i.e., the LCSs are easy to recognize. However, a direct volume rendering of 3D FTLE data shows high FTLE values rather than ridges. Therefore, an explicit ridge extraction method is often preferred. We describe a method for efficiently computing FTLEs and ridge surfaces, while avoiding aliasing artifacts.

1. Introduction

Lagrangian coherent structures (LCSs) can be defined [1] as the ridges of the finite-time Lyapunov exponent (FTLE). There exist methods for computing FTLEs at high accuracy. However, if FTLEs are computed for the pixels of an image, point-wise accuracy is not the goal, as it would create aliasing artifacts, but a low-pass filtered FTLE signal is needed. The appropriate method is therefore to compute trajectories on the nodes of a grid, resulting in a *flow map*, and then to estimate its gradient tensor **F**. From **F**, the (largest) FTLE is obtained as

$$f = \ln\sqrt{\lambda_1}\,/\,|\tau| \qquad (1)$$

where λ_1 is the largest eigenvalue of the Cauchy-Green tensor $\mathbf{C}=\mathbf{F}^T\mathbf{F}$ and τ is the integration time.

Estimation of the gradient, and of first and second derivatives in the subsequent ridge extraction, is best done by convolution with derivatives of a Gaussian, to avoid that high frequencies are (re-)introduced. The spread of the Gaussian to be chosen depends on the flow data and the integration time, but we found 1-1.5 times the grid resolution to be typically sufficient.

2. Ridge Surfaces

The *height ridge* [2,3] is defined for scalar functions $f:\mathbb{R}^n\to\mathbb{R}$ and generalizes the notion of a local maximum to *k*-dimensional structures $(0\le k<n)$. In the case of our interest, where *f* is an FTLE field, the most interesting ridges are *n*-1-dimensional. The two conditions for a point $\mathbf{x}\in\mathbb{R}^n$ to lie on a height ridge of co-dimension 1 are

$$\nabla f(\mathbf{x})\cdot\mathbf{e}(\mathbf{x})=0 \qquad (2)$$

$$\mathbf{e}(\mathbf{x})^T\,\nabla^2 f(\mathbf{x})\;\mathbf{e}(\mathbf{x})<0 \qquad (3)$$

where $\mathbf{e}(\mathbf{x})$ is the eigenvector of the Hessian $\nabla^2 f(\mathbf{x})$ corresponding to the smallest eigenvalue (by signed, not absolute value).

By substituting a different *transverse direction* for $\mathbf{e}(\mathbf{x})$, other ridge definitions can be obtained. A simple example is the use of a constant direction, which leads to poor results at places where $\mathbf{e}(\mathbf{x})$ is far from orthogonal to the ridge. If *f* is an FTLE field, however, a good choice for $\mathbf{e}(\mathbf{x})$ is the eigenvector of the Cauchy-Green tensor **C** corresponding to the largest eigenvalue λ_1. By this choice, (2) becomes one of the requirements for a *weak LCS* (see [4], Thm. 7). The resulting *C-ridges* [5] have the computational advantage over height ridges that only first derivatives have to be estimated for (2), which reflects in a smoother surface. Second derivatives are still needed for (3), which algorithmically means to trim the triangle mesh at the curve where the left hand side of (3) equals zero.

In principle, (2) amounts to an isosurface extraction by a method such as *Marching Cubes* or *Marching Tetrahedra*. However, in the cases where $\mathbf{e}(\mathbf{x})$ is not a vector but an eigenvector, the additional step of making these eigenvectors cell-wise consistently oriented is necessary. In the *Marching Ridges* method [6], principal component analysis is used for this step.

The standard *Marching Ridges* method generates a triangle mesh with vertices lying on the edges of the volume grid, obtained by linear interpolation. If the fields $f(\mathbf{x})$ and $\mathbf{e}(\mathbf{x})$ are given analytically, a *corrector step* can be added to improve the interpolation along the edge. This is the case, e.g., if convolution with kernels is used for estimating derivatives. We observed that the corrector step improves the quality of the triangle mesh and obtained corrections of up to a third of the edge length, with an average in the order of one percent of the edge length [7].

The final step in computing the ridge surface is the *trimming* of the triangle mesh. The minimum is to remove parts of the mesh where (3) does not hold. Typically, one also wants to remove "weak" and noisy ridges, which can be achieved by trimming the ridge at a lower threshold for $f(\mathbf{x})$. Finally, part of the points solving (2) and (3) do not have the characteristic of a ridge, namely if $\nabla f(\mathbf{x})$ is far from being tangent to the ridge. Therefore, it is advisable to also trim away parts of the triangle mesh where this angle exceeds a certain threshold such as 45° [8].

3. Nonorientability

The fact that ridge surfaces are in general not orientable causes some problems in their further processing. Gouraud shading, the standard method in computer graphics, requires the vertex normals to be consistently oriented. The use of two-sided lighting does not solve the problem.

A possible solution is to re-orient the normals on the fly in the vertex shader. However, it is wrong to simply

orient them toward the camera (see Fig. 1). Rather, vertex normals need to be oriented consistently with the facet normal oriented to the camera. For orienting the facet normal, a geometry shader is required.

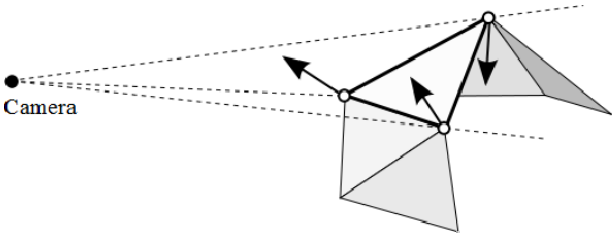

Fig. 1. Example of a triangle with inconsistent vertex normals (thick outlined). Nevertheless, all normals (i.e., their projections onto corresponding view rays) are oriented toward the camera.

An alternative solution is to traverse the triangle mesh, orient normals consistently where possible and cut the mesh where needed. Along cuts, vertices are duplicated, and the copy of the vertex is given the opposite normal. Cutting the mesh does not cause visible artifacts, provided that normals have been computed based on the full and correctly oriented neighborhood. Cuts can affect other processing, however, such as texturing, hatching, or "smart transparency" calculation. This limitation does not apply if a multilayer screen-space data structure as in [9] is used.

4. Results

We computed LCSs from simulation results of a revolving door where an air curtain has been added to prevent cold air entering the building along the floor (Fig.2). Since the flow barrier generated by the air curtain has a dynamically changing geometry, LCSs are well suited to track it over time. Using interactive viewing of the LCSs (Fig.3), we were able to find a "leak" in the flow barrier represented by the system of LCSs that opens for a fraction of the rotational period of the door.

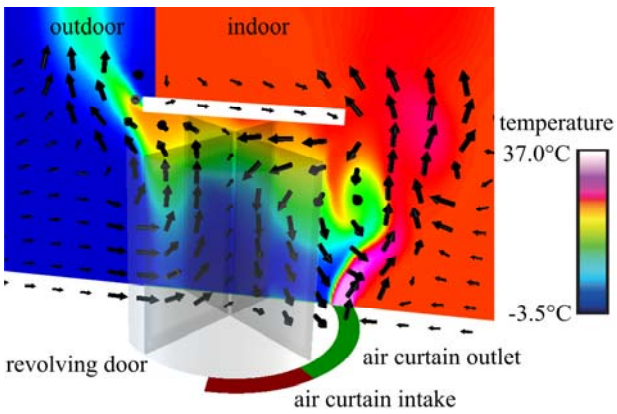

Fig. 2. Overview of the simulated setup with revolving door and air curtain.

With a modified design (using an asymmetric air curtain), a more tight flow barrier was achieved.

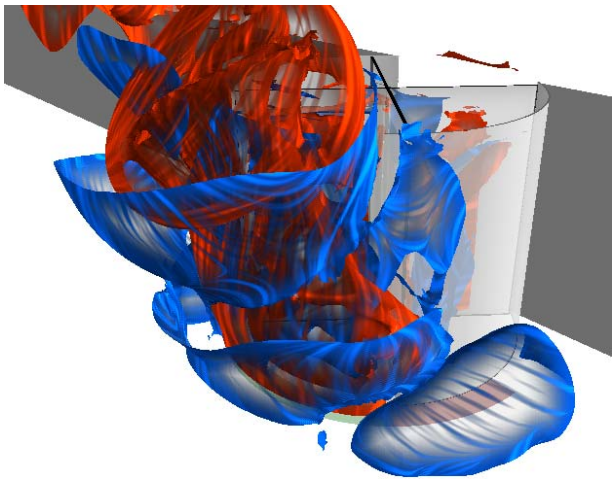

Fig. 3. Repelling (blue) and attracting (red) LCSs, visualized with context-sensitive transparency and texture along projected velocity.

Acknowledgments

This work was supported by the 7th Framework Programme for Research of the European Commission, under FET-Open grant 226042. We thank Stefan Barp and Matthias Röthlin for the CFD simulation.

References

[1] G. Haller, "Lagrangian coherent structures from approximate velocity data", *Phys. Fluids A*, **14** (2002) 1851–1861.

[2] D. Eberly, *Ridges in Image and Data Analysis*, Kluwer (1996).

[3] J. Damon, "Properties of Ridges and Cores in Two-Dimensional Images", *J Math Imaging Vis*, **10**, 2 (1999) 163-174.

[4] G. Haller, "A variational theory of hyperbolic Lagrangian Coherent Structures", *Physica D*, **240** (2011) 574–598.

[5] B. Schindler, R. Peikert, R. Fuchs, H. Theisel, "Ridge Concepts for the Visualization of Lagrangian Coherent Structures", in: *Topological Methods in Data Analysis and Visualization II*, Springer (2012) 221-236.

[6] J.D. Furst and S.M. Pizer, "Marching ridges", in: *Proc. IASTED Int'l Conf. on Signal and Image Processing* (2001) 22-26.

[7] B. Schindler, R. Fuchs, J. Waser, R. Peikert, "Marching Correctors — Fast and Precise Polygonal Isosurfaces of SPH Data", in *Proc. 6th SPHERIC workshop* (2011) 125-132.

[8] R. Peikert, F. Sadlo, "Height Ridge Computation and Filtering for Visualization", in *Proc. IEEE PacificVis* (2008) 119-126.

[9] R. Carnecky, B. Schindler, R. Fuchs, R. Peikert, "Multi-layer illustrative dense flow visualization", *Computer Graphics Forum*, **31**,3 (2012) 895-904.

[10] B. Schindler, R. Fuchs, S. Barp, J. Waser, A. Pobitzer, K. Matković, R. Peikert, "Lagrangian coherent structures for design analysis of revolving doors" (submitted).