Visualizing flow fields using fractal dimensions

Ross Vasko, Han-Wei Shen and Rephael Wenger

Abstract— Streamlines are a popular way of visualizing flow in vector fields. A major challenge in streamline visualization is selecting the streamlines for visualization. Rendering too many streamlines clutters the visualization and makes features of the field difficult to identify. Rending too few streamlines causes viewers to completely miss features of the flow field not rendered.

The fractal dimension of a streamline represents its space-filling properties. To identify complex or interesting streamlines, we build a regular grid of scalar values which represent the fractal dimension of streamlines around each grid vertex. High fractal dimension indicates vortices or turbulent regions. We use this scalar grid both to filter streamlines by fractal dimension and to identify and visualize regions containing vortices and turbulence. We describe an interactive tool which allows for quick streamline selection and visualization of regions containing vortices and turbulence.

Index Terms—Streamlines, fractal dimension.

1 Introduction

2 Related Work

3 Box Counting Ratio

4 Streamline Complexity Grid

For each regular point p in a vector field $f : \mathbb{R}^3 \Rightarrow \mathbb{R}^3$, let ζ_p be the unique streamline passing through p. Define $\phi_w(p)$ as the local box counting ratio of ζ_p in a $w \times w \times w$ region around p, where boxes have edge lengths 1 and 2. Function ϕ_w defines a scalar field on the regular points of f. The scalar $\phi_w(p)$ represents the complexity of the streamline through p in a $w \times w \times w$ neighborhood of p. We call the scalar field ϕ_w, the streamline complexity field of f.

To construct a scalar grid representing the streamline complexity field of f, we compute a set of streamlines so that every voxel is intersected by at least one streamline. We then compute the box counting ratio around each voxel v, by selecting a streamline ζ_v intersecting the voxel v and computing the local box counting ratio of ζ_v in a $w \times w \times w$ region centered at v. We call the resulting scalar grid, the streamline complexity grid of f.

To compute a set of streamlines intersecting every voxel, we...

To compute the local box counting ratio of ζ_v in a $w \times w \times w$ region centered at v, we...

5 Visualization Techniques

The streamline complexity computations can be used in a variety of ways to filter streamlines of varying complexities or to provide a method of visualization for the flow field.

Streamline filtering by value: The user is able to choose two constants, a and b where $a < b$, and only display streamlines with a complexity inbetween the chosen values. Specifically, the streamline ζ_p will be displayed only if $a \leq \phi_w(p)$ and $\phi_w(p) \leq b$. The constants allow the user to choose the level of streamline complexity that they will view. By choosing constants of values near 1, streamlines with a low box counting ratio and smooth flow will be displayed. By choosing constants of values above 2, streamlines with a relatively high box counting ratio and turbulent or complex flow will be displayed.

Local maximums: A significant amount of clutter in the streamline display will remain if additional filtering methods are not considered. Several streamlines in the visualization will be visuallly similar or provide redundant information. We are able to only show the local maximum streamlines to filter streamlines that all represent a single region or feature of the flow. Local maximum filtering will show a streamline ζ_p only if for all of the 8 points q that directly neighbor p on the complexity grid, $\phi_w(p) > \phi_w(q)$. This method allows for single streamlines representatives to be shown for each feature or region rather than several, cluttered streamlines.

Colored plane: A color plane can be used to allow the user to visualize the scalar complexity grid, ϕ_p, directly. A color gradient from blue to green to red is able to be defined and mapped to values in the range 0 to 3, for each of the possible box counting dimensions ratios. Low scalar values will be displayed as blue colors, while high scalar values will be displayed as red colors. A plane is then defined on the scalar complexity grid and each point on the plane is colored from this defined color gradient. The user is able to control the plane through the scalar complexity grid to identfiy regions varying complexity in the grid. Once regions of interest are identified through the color plane, the user can display streamlines near that region to understand its behavior.

Isosurfaces: An isosurface can be used to highlight regions of the flow field with a high complexity. An isosurface, which is defined by

$$\{x \mid \phi_w(x) = \sigma\} \tag{1}$$

where σ is the isovalue and ϕ_w is the streamline complexity field, will seperate all scalar values above σ from the values below σ on the streamlines complexity. This isosurface will enclose the regions of streamlines with a fractal dimension higher than the σ value and provide a simple way to identify regions of a defined complexity. At particularly high σ values, the isosurface will only enclose vortices and turbulent features of the flow field that the user may have otherwise missed.

Gradient magnitudes: The gradient magnitudes of the streamline complexity grid can be calculated to create a new gradient magnitude scalar grid ϕ_g. The scalar $\phi_g(x)$ is given by $\|\nabla\phi_p(x)\|$. Another isosurface can be used to visualization the function ϕ_g to identify regions of high change of turbulence or complexity. Vortices in the flow field tend to have high complexity values recorded near their centers, with values quickly decreasing towards their boundaries. When a high isovalue is chosen for the gradient magnitude isosurface's isovalue, the isosurface will highlight these isolated regions of turbulence or turbulent regions that quickly become smooth.

6 Examples

7 Evaluation

8 Summary and Future Work

The Ohio State University. E-mail: vasko.38@osu.edu, shen.94@osu.edu and wenger.4@osu.edu

Manuscript received 31 March 2007; accepted 1 August 2007; posted online 2 November 2007.

For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.