An Information-Theoretic Framework for Flow Visualization

Lijie Xu, Teng-Yok Lee, *Student Member, IEEE*, and Han-Wei Shen

Abstract—The process of visualization can be seen as a visual communication channel where the input to the channel is the raw data, and the output is the result of a visualization algorithm. From this point of view, we can evaluate the effectiveness of visualization by measuring how much information in the original data is being communicated through the visual communication channel. In this paper, we present an information-theoretic framework for flow visualization with a special focus on streamline generation. In our framework, a vector field is modeled as a distribution of directions from which Shannon's entropy is used to measure the information content in the field. The effectiveness of the streamlines displayed in visualization can be measured by first constructing a new distribution of vectors derived from the existing streamlines, and then comparing this distribution with that of the original data set using the conditional entropy. The conditional entropy between these two distributions indicates how much information in the original data remains hidden after the selected streamlines are displayed. The quality of the visualization can be improved by progressively introducing new streamlines until the conditional entropy converges to a small value. We describe the key components of our framework with detailed analysis, and show that the framework can effectively visualize 2D and 3D flow data.

Index Terms—Flow field visualization, information theory, streamline generation.

1 Introduction

Fluid flow plays an important role in many scientific and engineering disciplines. To visualize the vector data generated from flow simulations, one common method is to display flow lines such as streamlines or pathlines computed from numerical integration. The main challenge of displaying flow lines, however, is to have a proper placement of seeds. An ideal flow line seeding algorithm should generate visually pleasing and technically illustrative images. It should also allow the user to focus on important flow features.

While there has been extensive research effort in developing streamline placement algorithms [12, 16, 25], there is a lack of general approaches to quantitatively measure the quality of the visualization results. In addition, there are not enough guidelines regarding how to choose the algorithm parameters, for example, the streamline distance threshold, used to minimize the chance of missing important features. Furthermore, the flow lines generated from these algorithms cannot guarantee that salient regions in the input field will always be highlighted with proper visual focus. Meanwhile, occlusion among the streamlines in 3D flow fields can become a major obstacle to conduct a detailed analysis for more complex data sets.

In this paper, we tackle the problems of flow visualization from an information-theoretic point of view. Specifically, our goal is to quantitatively measure how the information from the input data revealed as flow lines are progressively introduced to the field. The concepts of entropy and conditional entropy in information theory are exploited to achieve this goal. By modeling the input vector field as a random variable represented as X, we can measure the amount of information or uncertainty contained in a local region using Shannon's entropy $H(X)$. We can also model the visualization output consisting of a set of flow lines as another random variable Y. The conditional entropy between the two variables $H(X|Y)$ tells us how much uncertainty in the input data X still remains after the flow lines in Y are shown. With the conditional entropy measure, our streamline placement algorithm can adaptively generate more streamlines in salient regions, and regions whose information has not been fully revealed. The conditional entropy measure can also indicate whether enough streamlines have been placed to represent the entire input field, preventing us from placing too many streamlines and hence avoiding visual clutters. With the entropy measures computed from the data, we are able to use different visual cues to highlight more important streamlines and reduce occlusion around the salient structures.

Our contributions in this paper are three-fold. First, we propose a theoretical model to quantitatively measure the visualization quality of flow lines based on the amount of information contained in the data set. Second, we present an information-aware streamline placement algorithm allowing the viewer to quickly focus on more salient regions with fewer streamlines. Third, our rendering algorithm can emphasize features according to the local entropy values and thus minimizes visual cluttering and creates final results with better visual focus.

The remainder of the paper is organized as follows. After reviewing related work in Section 2, we introduce the theoretical framework behind the ideas presented in this paper in Section 3 and Section 4. Our streamline generation algorithm is described in Section 5, and the evaluation and comparison with some existing methods are shown in Section 6. Section 7 discusses the limitations and future work of our algorithm, and Section 8 concludes this paper.

2 Related Work

In computer graphics, information theory [6] has been utilized to solve a variety of problems. Flixas *et al.* [7] presented an information-theoretic approach for the analysis of scene visibility and radiosity complexity. Vázquez *et al.* [24] introduced the viewpoint entropy as a measure to automatically compute good viewing positions for polygonal scenes. Gumhold [9] presented an entropy-based solution for placing light sources given camera parameters in a scene. For volume rendering, Bordoloi and Shen proposed a view point selection algorithm based on Shannon's entropy driven by voxel importance [3], Viola *et al.* proposed another information-theoretic view selection framework that selects the viewpoint to maximize the mutual information between the rendering objects [26] and the view point. The focus in Wang *et al.*'s work [27] is to select salient time steps that contain major features in volumetric time-varying data. Rigau *et al*. [19] used Shannon's entropy as the aesthetics measure for paintings by hypothesizing that the quality of arts is related to its complexity level; namely, the more information it contains, the higher the entropy will be.

Information theory is widely used in image processing and computer vision. One major application is image registration, which aligns two images such that the corresponding structures in both images overlap. When the images are well aligned, the uncertainty about the pixel intensity in one image should be minimized given the corresponding pixel in the other image. Mutual information in information theory has been widely used to compute uncertainty [18]. Another application is image thresholding, where the goal is to search for an optimal intensity threshold to segment the foreground objects. From the perspective of information theory, the foreground and background come

• *Lijie Xu, Teng-Yok Lee and Han-Wei Shen are with The Ohio State University, E-mail: {xul,leeten,hwshen}@cse.ohio-state.edu.*

Manuscript received 31 March 2010; accepted 1 August 2010; posted online 24 October 2010; mailed on 16 October 2010.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org.

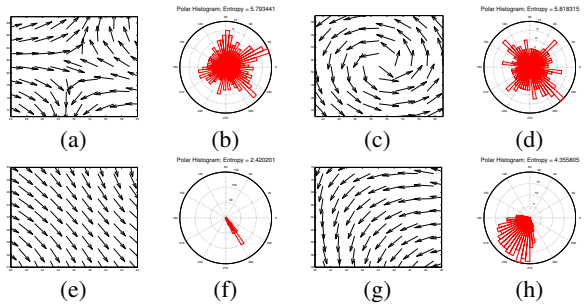


Fig. 1. Distributions of vector directions in different vector fields. (a) and (c) are vector fields that have a saddle and a spiral source. (e) and (g) are two vector fields without critical points. The polar histograms with 60 bins created from the vector fields are shown in figure (b),(d),(f),(h). The entropies of the vector fields in (a), (c), (e) and (g) are 5.79, 5.82, 2.42 and 4.36, respectively. The range of the entropy with 60 bins is $[0, log_2(60) = 5.9]$.

from different signal sources, and thus the optimal threshold can be obtained when the summation of the entropy for the foreground and background are maximized [22].

Several streamline placement algorithms have been proposed previously for 2D flow fields, including the evenly-spaced streamline placement method [12] which was later extended in [15, 23, 29], the farthest point seeding method [16], and the flow topology based method [25, 29]. The goal in [12] is to place streamlines evenly in space to avoid visual cluttering. The farthest point seeding method in [16] places seeds in regions that have the largest void, where the goal is to generate streamlines that are as long as possible. Liu *et al.* [15] speeded up the algorithm in [12] by reducing the number of samples using a Hermite polynomial interpolation. The number of discontinuous streamlines are reduced by employing a double queue data structure. Wu *et al.* [29] presented a topology-aware evenly-spaced streamline algorithm which segments the field based on the locations of singularities and separatrices, and then carefully selects seed paths to avoid short streamlines. Verma *et al.* presented a topology-based method to capture flow features with pre-defined seed templates around critical points to ensure that salient flow features are properly highlighted [25].

For 3D flow fields, Ye *et al.* [31] extended the topology-based seeding method by Verma *et al.* [25] and defined 3D seed templates to highlight flows around critical points. Li and Shen proposed an image-based 3D streamline seeding algorithm, where seeds are dropped on 2.5 D surfaces to avoid occlusion and visual clutter [14]. Chen *et al.* proposed a 3D streamline pruning algorithm, which considers the Euclidean distance, direction, and the shape between a pair of streamlines. The algorithm stops the advection of streamlines if the distance between the current and the existing streamlines is smaller than a threshold [5].

In addition to streamline placement, feature analysis of flow fields is also a significant research topic. Recently, Janicke used statistical complexity analysis to analyze time-varying multifields, which was then applied to time-dependent flow fields [10]. Schlemmer *et al.* utilized the moment invariants of vector fields to detect 2D flow features under rotation [21]. In [8], Furuya and Itoh applied information theory to measure the complexity of existing streamlines based on the distribution of the tangent orientations. While both [8] and our framework use information theory to analyze flow fields, our framework not only measures the complexity inside the field but also generates streamlines accordingly. Besides, our framework can measure the remaining uncertainty about the flow field given the known streamlines.

3 Information Measure of Vector Fields

In this section we explain how to model data in a vector field as a random variable, and how to measure the information content using Shannon's entropy.

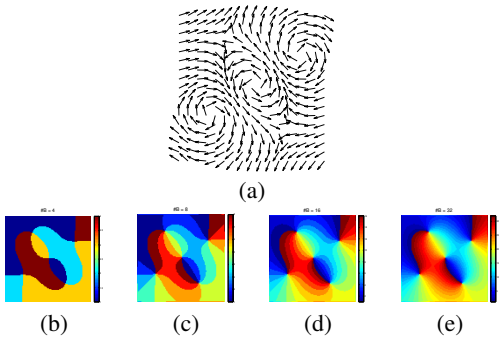


Fig. 2. A vector field (a) and the distributions of vectors in the histogram bins. (b) - (e): The number of histogram bins (B) used for the vector field in (a) are 4, 8, 16 and 32, respectively, where each bin is shown in a different color.

3.1 Entropy of Vector Fields

Information theory provides a complete theoretical framework to quantitatively measure the information content from a distribution of data values. Given a random variable X with a sequence of possible outcomes x, $x \in \{x_1, x_2, \ldots x_n\}$, if we know that the probability for the random variable X to have the outcome x_i is $p(x_i)$, then the information content for the random variable can be computed using Shannon's entropy as:

$$H(X) = -\sum_{x_i \in X} p(x_i) log_2 p(x_i) \quad (1)$$

Shannon's entropy is to measure the uncertainty of a random variable. An important property of the entropy is that $H(X)$ is convex and reaches its maximum when $p(x_i)$ is equal for all x_i.

To apply Shannon's entropy to measure the information content of a vector field, a histogram needs to be created in order to approximate the probability mass function $p(x)$. This can be done by first partitioning the range of the vectors, represented as a polar angle $\theta, 0 \leq \theta \leq 2\pi$ for two dimensional vectors, into a finite number of bins $x_i, i = 1 \ldots n$. For 3D vectors, we decompose a unit sphere into 360 patches of equal area with small diameter [13], and use the cones that connect the patches and the sphere center to quantize the 3D vectors. A histogram can be created from the vectors in the data set by assigning the vectors to appropriate bins. With the histogram, the probability of the vectors in bin x_i can be computed using Equation 2:

$$p(x_i) = \frac{C(x_i)}{\sum_{i=1}^{n} C(x_i)} \quad (2)$$

where $C(x_i)$ is the number of vectors in bin x_i. Using the probabilities calculated here, we can compute the entropy from equation 1. Vector fields with a higher degree of variation in the vector directions will receive higher entropy values, and thus are considered to contain more information than vector fields where most of the vectors are parallel to each other.

One important flow feature in a vector field is the presence of critical points, which can be captured well with the entropy measure. This is because the flow directions in the local neighborhood around a critical point exhibit a higher degree of variation, and thus a higher entropy. Figure 1 presents vector fields with and without critical points and their corresponding histograms presented in a polar form. It can be seen that the vectors in regions that contain critical points have higher variations, i.e., a more uniform distribution in the histograms.

It is noteworthy that the value of probability, and hence the entropy, is affected by the number of histogram bins used. Given the vector field in Figure 2 (a), Figures 2 (b) - (e) show the distributions of vectors from histograms that have different number of bins, where each bin is shown in a different color. It can be seen that vectors around the critical points always spread across a relatively large number of bins compared to other regions regardless of the number of bins being used

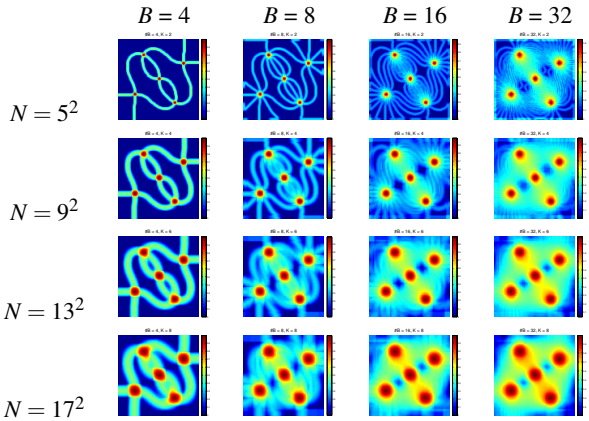


Fig. 3. The entropy fields computed from the vector field in Figure 2 (a) using different neighborhood size(N) and numbers of histogram bins (B).

by the histogram, and thus will receive a relatively high entropy. In the next section, the effect of the number of bins to the entropy is discussed in more detail.

3.2 Entropy Field

To measure the information content in the local regions across a vector field, we compute the entropy in the local neighborhood around each grid point. We call this resulting scalar field an *entropy field*. The value in the entropy field for a point indicates the degree of vector variation in its local neighborhood, providing a cue for us to decide where the field is more salient. This entropy field can be used to assist us in streamline placement and rendering, described in section 5.

The computation of the entropy field requires two parameters, the number of histogram bins and the neighborhood size. Figure 3 shows the entropy fields computed from the vector field in Figure 2 (a) using different number of bins B and neighborhood sizes N. Higher entropy values are mapped to warmer colors. The neighborhood size is related to the number of samples used to calculate the histogram for the entropy computation. An insufficient sample size can create artifacts in the computed entropy field. The image at $N = 5^2, B = 32$ shows an example of insufficient neighborhood size where the number of samples at each point is actually smaller than the number of bins. The artifacts can be alleviated by increasing the neighborhood size, as shown in the images where $N = 9^2$, 13^2 and 17^2. In our implementation, we empirically set $N = 13^2$ for 2D and $N = 13^3$ for 3D. Samples are taken from the grid points of the input field. We use 60 bins for 2D and 360 bins for 3D vector fields, where 360 is the number of patches in the unit sphere. For points near the domain boundary where the neighborhood exceeds the boundary, we mirror the data to compute the entropy.

The benefit of using the entropy field is that it can highlight not only regions near the critical points but also regions that contain other flow features. To illustrate this, Figure 4 shows a comparison between using the entropy and the norm of the Jacobian matrix computed from a vector field. The Jacobian represents the local vector gradient and is often used to characterize the flow topology such as the type of critical points. The vector field is shown in Figure 4 (a), which is the result of simulating an intense hurricane system *Isabel*. In Figures 4 (b) and (c), each pixel displays the entropy and the average Frobenius norm of the Jacobian matrix of its surrounding neighborhood, respectively. It can be seen that the entropy field not only highlights the critical points, but also the regions near the separation lines, which is less apparent in the image of Jacobian norm.

4 Information Measurement of Streamlines

Based on the information measures, in this section we show how to evaluate the effectiveness of a given set of known streamlines to represent the input vector field. The key idea of our approach is that, when

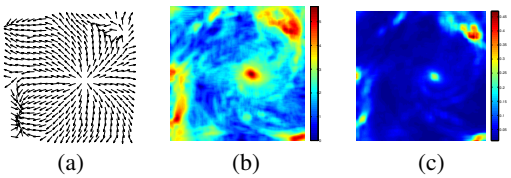


Fig. 4. Entropy versus the norm of Jacobian. (a) The vector field *Isabel* (b) The entropy field. (c) The field of the average of the Jacobian norm.

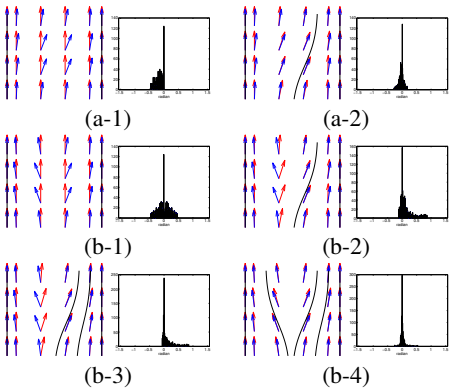


Fig. 5. An example to show that the number of required streamlines is Influenced by the distribution of vectors: in (a-1) and (b-1), the angle differences between the original (blue) and the intermediate vector fields (red) have the same L^2 norm but different distributions. While an additional streamline in (a-2) can effectively reduce the angle difference between the original and the reconstructed intermediate fields, more streamlines are needed for the case in (b-1), as the sequence shown in (b-2), (b-3), and (b-4). The conditional entropies of (a-1) and (a-2) are 2.40 and 1.65. The conditional entropies from (b-1) to (b-4) are 3.02, 2.16, 1.98, and 1.44, respectively.

streamlines are placed in the field, not only the vector directions along the streamlines are known, the vectors in the local neighborhood, to some extent, can also be inferred. If we generate an intermediate vector field from the known streamlines, the difference in the information content between the intermediate vector field and the original vector field should be very small if the known streamlines can sufficiently represent the original vector field. In the following, we describe our approach in detail.

4.1 Conditional Entropy Measure

Two important steps are involved in measuring the effectiveness of streamlines computed from an input vector field. One is to create an intermediate field based on the assumption of local coherence, and the other is to compute the discrepancy in the information content between the input vector field and the intermediate field. The purpose of computing the intermediate field is to estimate how much information can be inferred from the known streamlines. While it is obvious that the vector directions along the streamlines contribute to the knowledge of the data, more can be gained if we assume there exists spatial coherence, i.e., the directions of vectors in the field will change gradually from the known streamlines to their nearby regions. Obviously, this assumption will not be true if no sufficient streamlines are present. Therefore, we can evaluate the representativeness of known streamlines for the input field by comparing the original field and the intermediate field. In this subsection, we focus on comparing the information content between the two fields. We defer the discussion of generating the intermediate field to Section 4.2.

To compare two vector fields, conventional similarity metrics such as the L^2 distance norm consider only the average of the vector differences between the two fields but not how the differences of the vectors distribute in the value space. For this reason, the L^2 norm cannot always correctly indicate whether more streamlines are needed to gen-

erate an intermediate vector field that matches the original field. In the following, we use an example to illustrate the problem.

Figures 5 (a-1) and (b-1) show two cases where the angle differences between the reconstructed vector field and the original vector field have the same L^2 norm but the distribution of the vector differences is different. The intermediate vectors in Figure 5 (a-1) are all pointing to the left of the original vectors. In this case, with a single additional streamline shown in Figure 5 (a-2), the reconstructed vector field becomes very close to the original vector field. On the other hand, the intermediate vectors in Figure 5 (b-1) are pointing to both left and right of the original vectors. In this case, if not enough streamlines are added, as shown in Figures 5 (b-2) and (b-3), the reconstructed intermediate field will not be close enough to the original one. Figure 5 (b-4) shows that more streamlines are needed to make the intermediate vector field converge to the input vector field. From this example, we can see that the number of streamlines needed to make the intermediate field converge to the input vector field depends on the distribution of the vector differences between these two fields.

To consider the distribution of two vector fields, Shannon's conditional entropy can be used. Assuming the original vector field is represented as a random variable X, and the intermediate field is denoted as a random variable Y, the conditional entropy of X given Y is defined as:

$$H(X|Y) = \sum_{y \in Y} p(y) H(X|Y = y) \tag{3}$$

where

$$H(X|Y = y) = -\sum_{x \in X} p(x|y) log_2 p(x|y) \tag{4}$$

To calculate the conditional entropy between the two vector fields X and Y, we first estimate the joint probability distribution between the two fields by constructing a two dimensional joint histogram, where each vector field is represented by an axis. In our implementation, we partition the range of the vector into 60 bins for 2D vectors, and 360 bins for 3D vectors. The histogram cell (i, j) is incremented by one if a vector x in the input vector field and a vector y in the intermediate vector field from the same location of the domain fall into the histogram bin (i, j). The joint histogram is an estimate of the joint probability distribution function, which can be used to derive the marginal probability of $p(y)$ and the conditional probability $p(x|y)$ for Equations 3 and 4.

Two mathematical properties of the conditional entropy are important to our problem. First, the upper bound of $H(X|Y)$ is $H(X)$ [6]. It means that given two vector fields, the conditional entropy computed from the original vector field that contains more information will be higher, thus requiring more streamlines. For instance, the conditional entropy in Figure 5 (a-1) is 2.40, while that of the more complex vector field in Figure 5 (b-1) is 3.02. Second, $H(X|Y)$ measures the remaining entropy, or uncertainty, of X given that the values of the random variable Y are known. In the context of our problem, it indicates how much information in the original vector field remains unknown given the intermediate vector field derived from the known streamlines. The conditional entropy will converge to a small value if enough streamlines have been placed. This property can be utilized to avoid the placement of unnecessary streamlines, therefore reducing visual clutter.

In addition to conditional entropy, we note that there exist other similarity metrics such as the normalized cross-correlation-based metrics (NCC) between two distributions. NCC is the cross correlation of two random variables divided by the product of their standard deviations. The major benefit of NCC is that it is invariant under the change of amplitude, which makes it especially useful in image processing for detecting similar objects under different illumination. NCC, however, is not suitable for us to measure the difference between two vector fields. This is because NCC is divided by the product of two standard deviations, hence the flow field with small standard deviation can still obtain a high score. Figure 6 shows two examples. From Figures 6 (a) and (c), it can be seen that the angle difference between the original and the intermediate vectors is small. Nevertheless, because the joint distributions in these two examples are quite different, as shown in Figures 6 (b) and (d), their NCC values are very different. The difference of the two NCC covers 75% of the whole range $[-1, +1]$, which is too large. On the other hand, the difference between the conditional entropy in these two examples is much smaller compared to the range of the conditional entropy. We also note that if non-normalized cross correlation is used to compare flow fields, the scale of the cross correlation near the critical points can be too large, as shown in Figures 7 (a) - (c). This is because the vector field near the critical points is more complex and thus has much larger variations. Consequently, the cross correlation in other regions becomes indistinguishable and cannot be used to guide where to place more streamlines.

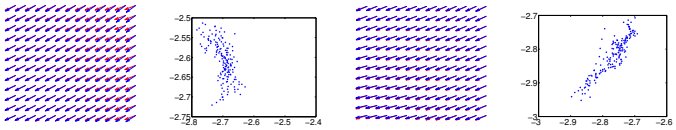


Fig. 6. Comparison of the normalized cross correlation and the conditional entropy. (a) and (c): Two vector fields (blue) and their intermediate vectors (red), where the difference between the original and the intermediate vectors is small. (b) and (d): the scatter plots of the input vectors (x axis) versus the intermediate vectors (y axis) of (a) and (c), represented as orientations, respectively. The values of the normalized cross correlation in (a) and (c) are -0.56 and 0.90 in the range $[-1, +1]$, respectively, and the corresponding conditional entropy values are 0.56 and 1.08 in the range $[0, log_2(60)]$.

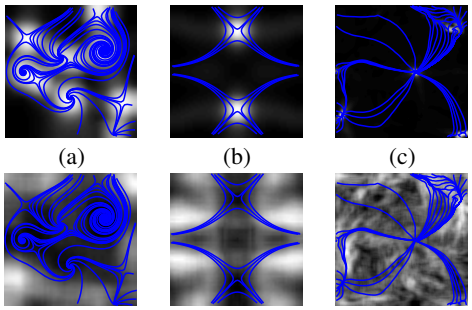


Fig. 7. Comparisons between the non-normalized cross correlation fields (top) and the conditional entropy fields (bottom) computed from the intermediate vector fields and the original fields.

4.2 Streamline Diffusion

Clearly, an important task in our framework is to create the intermediate vector field from a given set of streamlines. To do this, a straightforward method is to use a low pass filter such as the Gaussian kernel to smooth out the vectors on the streamlines to their surrounding regions. This method, however, has several drawbacks. First, choosing an appropriate kernel size for the filter is non-trivial. Kernels that are too small will not be able to cover the entire field when only a sparse set of streamlines are present. On the other hand, when the size of the kernel is too large, salient features in the field will be destroyed.

Figure 8 illustrates the problem of using the Gaussian kernel. Figure 8 (a) shows a test vector field with six streamlines being selected, shown in blue, and their surrounding vectors. A Gaussian kernel is applied to the vectors on the streamlines to generate the intermediate vectors near the streamlines. Figures 8 (b), (c), and (d) show the resulting vectors in the small red box in (a) with different kernel sizes. It can be seen that an apparent disparity between the original vectors (in black) and the resulting vectors from Gaussian smoothing (in red) exists, even on the input streamlines.

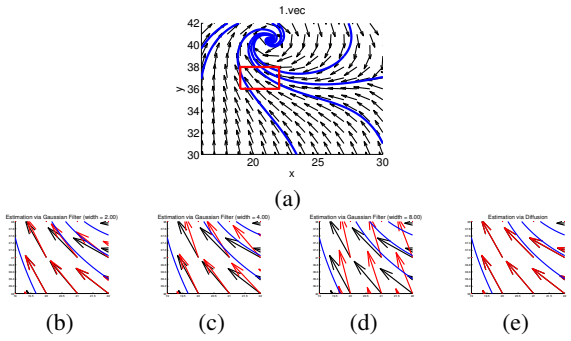


Fig. 8. Intermediate vectors computed using Gaussian smoothing and streamline diffusion: (a) The test vector field, where the black arrows represent the original vectors and the blue streamlines are computed from the original data. The estimated vector field inside the red rectangle in (a) is shown as the red arrows in (b), (c), (d), and (e). Vectors in (b), (c), and (d) are estimated via 2D Gaussian filters with widths 2, 4, and 8, respectively, which show apparent disparity between the original vectors (black) and the resulting vectors from Gaussian smoothing (red). Vectors in (e) are computed via vector diffusion, where the resulting vectors converge to the original vectors.

For our problem, what is needed is a method that has the following desired properties when generating the intermediate vector field. One is that the entire field needs to be properly covered. The difference between the adjacent vectors in the output field, and the difference between the true vectors on the known streamlines and the approximate vectors after smoothing should be minimized. To achieve this goal, we formulate the problem of generating such a vector field $\mathbf{Y}(\mathbf{x}=(x,y,z))=(u(\mathbf{x}),v(\mathbf{x}),w(\mathbf{x}))$ with respect to the field $\hat{\mathbf{X}}(\mathbf{x})$ that minimizes the following energy function:

$$\varepsilon(\mathbf{Y})=\int \varepsilon_1(\mathbf{Y}(\mathbf{x}),\hat{\mathbf{X}}(\mathbf{x}))+\mu\varepsilon_2(\mathbf{Y}(\mathbf{x}))d\mathbf{x} \tag{5}$$

where

$$\varepsilon_1(\mathbf{Y}(\mathbf{x}),\hat{\mathbf{X}}(\mathbf{x}))=|\hat{\mathbf{X}}(\mathbf{x})|^2|\mathbf{Y}(\mathbf{x})-\hat{\mathbf{X}}(\mathbf{x})|^2$$
$$\varepsilon_2(\mathbf{Y}=(u(\mathbf{x}),v(\mathbf{x}),w(\mathbf{x})))=|\nabla u(\mathbf{x})|^2+|\nabla v(\mathbf{x})|^2+|\nabla w(\mathbf{x})|^2$$

In the term $\varepsilon_1(\mathbf{Y}(\mathbf{x}),\hat{\mathbf{X}}(\mathbf{x}))$, $\hat{\mathbf{X}}(\mathbf{x})$ equals the original field $\mathbf{X}(\mathbf{x})$ along the streamlines and zero elsewhere. Therefore, ε_1 is always zero regardless of the intermediate vector $\mathbf{Y(x)}$ for $\mathbf{x}$ not occupied by the streamlines since $\hat{\mathbf{X}}(\mathbf{x})$ is zero. This term will be minimized when the reconstructed vector $\mathbf{Y}(\mathbf{x})$ along the streamlines is equal to the original vector $\mathbf{X}(\mathbf{x})$. The term $\varepsilon_2(\mathbf{Y})$ is the sum of the gradient magnitude of the three component u, v, and w at location $\mathbf{x}$, which can be minimized when the neighboring vectors are identical.

In other words, the diffusion process is modeled as a constraint optimization problem with soft boundary condition, where ε_1 penalizes the violation of the boundary condition along the streamlines, and ε_2 measures the smoothness of the reconstructed vector field, where the tradeoff between the boundary condition and the smoothness is controlled by the parameter μ. By setting $\mu=0.1$ in our experiments, empirically, the reconstructed vector converges to the input vector along the streamlines, thus preserving the boundary condition.

It is noteworthy that the energy equation presented here is similar to the force used in the *gradient vector field* snake [30], which can be solved by the *generalized diffusion equations* described in the fluid flow literature [4]. More details about the solution and the convergence of the diffusion can be found in [30]. In the image processing field, there exists similar research such as [11, 17] that distributes colors on a curve to its surroundings. The major difference between the two is that the gradient constraint in the diffusion curve method demands high gradients on the input curves to preserve object boundaries. In streamline diffusion, the smoothness constraint is applied everywhere even for the streamline boundaries. The intermediate field computed by streamline diffusion is used to measure the streamline information and the placement of seeds as explained in the following sections.

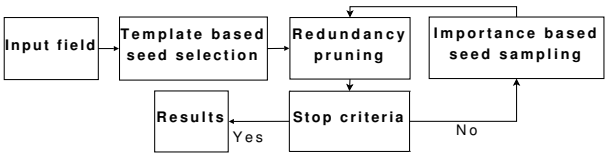


Fig. 9. Flow chart for our information-aware streamline placement algorithm.

5 Information-Aware Streamline Placement

In this section, we demonstrate how to use the information-theoretic framework to select streamline seeds. The goal of our algorithm is to choose streamlines that can succinctly present the directional information and highlight salient features such as critical points in the vector field. Based on the conditional entropy measure described previously, we can obtain a quantitative measurement of the quality for the chosen streamlines, and avoid displaying streamlines that do not contribute much to the overall understanding of the data in the visualization. Figure 9 shows the flow chart of our seeding algorithm. In our algorithm, we first place streamlines near regions that have larger values in the entropy field, and then iteratively place more streamlines in the domain according to the conditional entropy between the input field and the intermediate vector field derived from the known streamlines. Our algorithm terminates when the reduction of the conditional entropy converges. In the following, we describe our algorithm in detail.

5.1 Template-Based Initial Seed Selection

As discussed earlier, regions where the directions of vectors change more rapidly will result in higher entropy values compared to regions with vectors that are relatively more homogeneous. From the analysis in Figure 1, we also know that regions near critical points have larger directional variations and therefore tend to have higher entropies. Based on this observation, our algorithm first detects local maxima in the entropy field to capture the more salient regions. After discarding the local maximum points whose entropy values are too small to be considered, we place a set of initial seeds around the remaining extreme points. Empirically, we discard the local maximum points whose entropy values are smaller than 0.7 times the maximum entropy value among the candidates.

The seeds are distributed using a diamond shape template, which can be seen from the location of the red points in Figure 10 (b) and (h). The template used here is inspired by the flow topology based method proposed in [25]. Since we only choose regions that have larger entropy values without explicitly detecting the existence of critical points and their types, our template is a combination of the templates used in [25], whose aim is to capture the local flow topology near the salient regions. For 3D data, we use an octahedral shaped template. Each template places 27 seeds: one seed is placed at the centroid of the octahedron; 6 seeds are placed at the 6 vertices; 8 seeds are placed at the centers of the 8 faces, and 12 seeds are put at the midpoints of the 12 edges.

5.2 Importance-Based Seed Sampling

Because our template-based seeding method greedily selects seeds only around the high entropy regions, after the initial seeding stage, there may still exist void regions whose information remained uncovered and thus more seeds will be needed. To avoid visual clutter, we will place fewer seeds in regions where the information has been revealed by the known streamlines. This can be done by checking the conditional entropy computed from the original vector field and the intermediate vector field derived from the known streamlines.

Our algorithm models the placement of additional streamline seeds as an importance-based sampling problem where the probability to

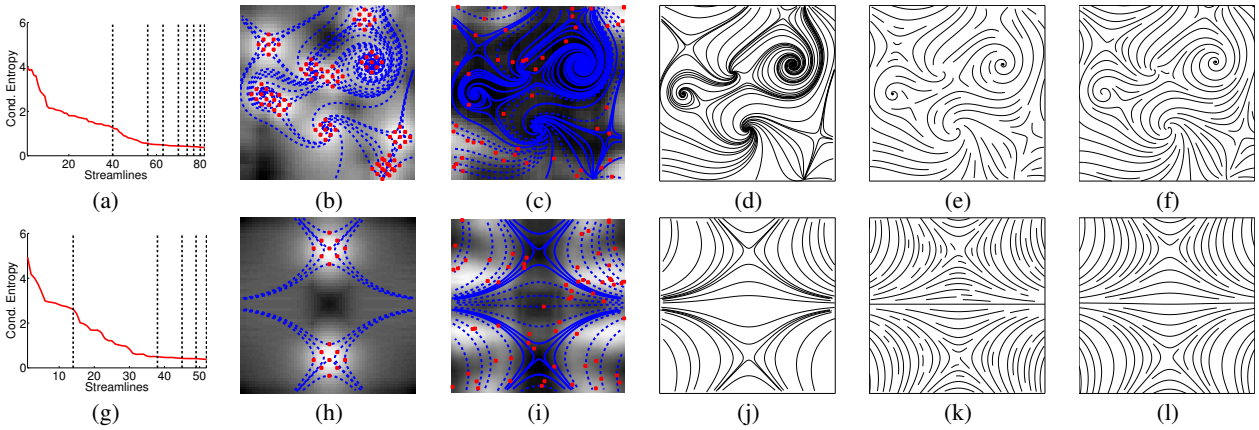


Fig. 10. The process of streamline generation for two different 2D vector fields (top and bottom). (a) and (g) plot the values of the conditional entropy as more streamlines are placed in the fields. The vertical black dashed lines indicate the end of each iteration. (b) and (h) show the initial seeds (in red) and the resulting streamlines (in blue) using our seeding template. (c) and (i) show the resulting streamlines after the first iteration of importance-based seeding. (d) and (j) show the streamlines when the conditional entropy has converged. (e) and (k) are the streamlines generated by the evenly-spaced seeding method [12]. (f) and (l) show streamlines generated by the farthest-point seeding method [16].

drop a seed at a point is proportional to the conditional entropy computed from its local neighborhood. To compute the local conditional entropy, we used the same neighborhood size and number of histogram bins as in the computation of the entropy field described earlier. From the conditional entropy at each point, the expected probability $p(x,y)$ of dropping a seed at a spatial point (x,y) is computed by Equation 6 below,

$$p(x,y) = \frac{h(x,y)}{\sum_{\forall x,y} h(x,y)}, \tag{6}$$

where $h(x,y)$ is the conditional entropy at point (x,y). The probability at each point forms the probability distribution function (PDF) of seeding in the domain. With the seeding PDF, we can distribute the seeds according to the probability at each point by importance sampling. As a well studied problem in statistics, importance sampling can be achieved by either the inverse transformation method or the acceptance-rejection method. In this framework, we use the inverse transform method with the chain rule for multidimensional data. More details about these methods can be found in [20].

5.3 Redundant Streamline Pruning

With the importance-based sampling method described above, fewer streamlines are placed in less important regions where the conditional entropy is low. To further eliminate streamlines that contribute little to the understanding of the data, our algorithm contains a pruning stage to eliminate redundant streamlines. In theory, we can easily decide if a streamline is necessary by comparing the conditional entropy computed for the field before and after the streamline is introduced. If no sufficient reduction is observed, the streamline does not contribute much to the knowledge of the data and therefore can be pruned.

While simple, the above algorithm is computationally very expensive because the intermediate vector fields with and without the candidate streamline need to be computed every time when such a decision is to be made. To accelerate the computation, we use the following information-guided distance based method to perform the pruning.

Intuitively, if a streamline is close to any of the existing streamlines, there is a possibility that it contributes little to the understanding of the underlying vector field. In a lower entropy region, where fewer streamlines are needed, the possibility for the streamline to be redundant is higher compared to the case when the streamline is in a higher entropy region. To account for the entropy values when pruning redundant streamlines, we allow higher entropy regions to have a smaller distance threshold than lower entropy regions. We choose a distance threshold from a scalar range $[R1,R2], R1 < R2$, where R1 is used for regions that have the maximum entropy and R2 is used for regions with the minimum entropy. Regions that have entropy in between will use a threshold linearly interpolated between R1 and R2. With the thresholds, for a given streamline, if all of the points on the streamline have a neighboring streamline that is closer than the threshold calculated from its entropy, the streamline is pruned, otherwise it is retained. In our implementation, we set R2 to 2% of the smallest dimension in the domain, and R1 to half of R2.

It is noteworthy that the result of pruning is order dependent. That is, if we place less important streamlines first, more salient streamlines will be pruned if they are too close to the unimportant ones. To solve this problem, the streamlines are first sorted based on the entropy values at their seeds in a decreasing order before the pruning process starts. Therefore, more salient streamlines will have a higher chance to survive.

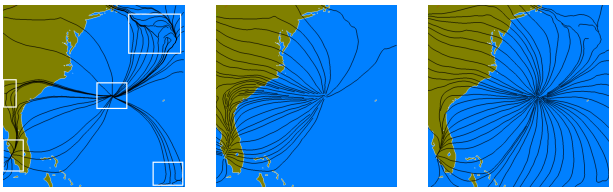


Fig. 11. 54 streamlines generated the *Isabel* data set using different methods. The subfigures from left to right were generated by our information-aware method, evenly-spaced seeding method, and the farthest point seeding method, respectively.

5.4 Stopping Criteria

The above importance-based sampling and seed pruning steps are performed iteratively. Within each iteration, a pre-determined number of seeds will be introduced to the field. We set the number of seeds in each iteration as the square root of the number of grid points. The process is repeated until the conditional entropy between the original vector field and the intermediate field converges.

6 Results and Analysis

This section presents analysis of our information-theoretic framework, including comparisons with other seed placement algorithms, entropy-augmented rendering for 3D data, and the performance of our algorithm.

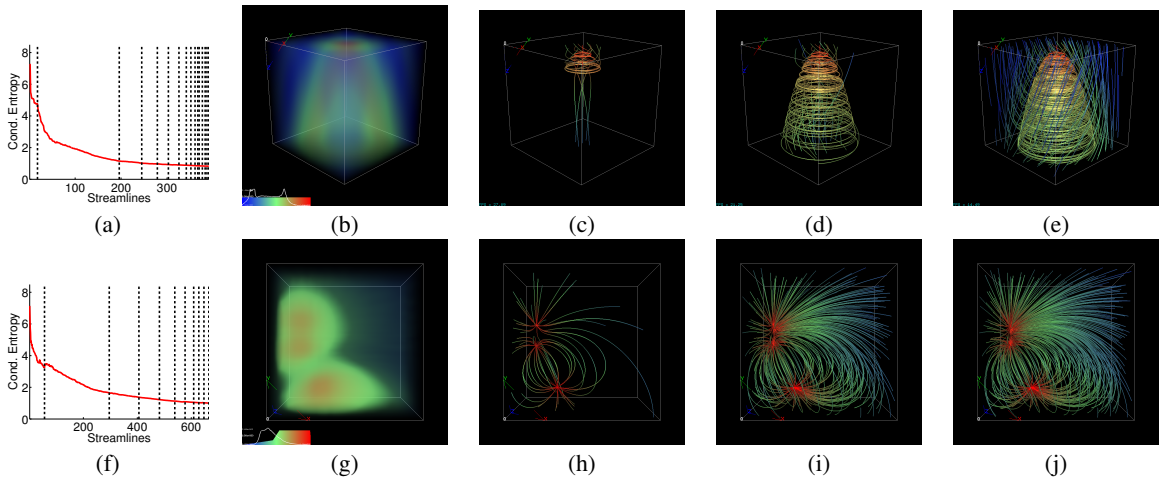


Fig. 12. Streamlines generated from the 3D vector fields *Circle* (top) and *Electro* (bottom). The first column lists the conditional entropy values as more streamlines are introduced. The second column presents the entropy field computed from the original vector field. The third column show the initial streamlines generated from our template. (d) and (e) show the first 50 and 200 streamlines, respectively, for *Circle*. (i) and (j) show the streamlines after the second and third iterations for *Electro*.

6.1 Seeding

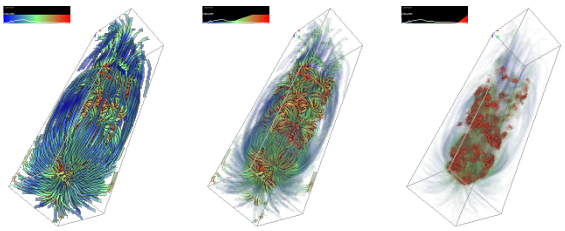
Figure 10 shows the streamlines generated by our algorithm for two 2D vector fields. The datasets used were published by Mebarki *et al.* in [16]. The leftmost column shown in Figure 10 plots the values of the conditional entropy for each of the data sets as more streamlines are placed. The vertical dashed lines signifies the end of each iteration in our seeding algorithm. From the figure, we can see that after the first iteration, the numbers of seeds introduced in the subsequent iterations decreased and the conditional entropy quickly converged.

The second column in Figure 10 shows the initial seeds generated by our template, with the entropy values plotted as the background. It can be seen that the initial seeds are distributed in high entropy regions, which match pretty well with where the critical points reside. The third column shows the streamlines generated after the first iteration of importance-based seeding, with the conditional entropy values plotted as the background. We can see that in this iteration, our algorithm placed streamlines in the void regions. The fourth column in Figure 10 shows the streamlines when the conditional entropy has converged.

We compare the results with two commonly used distance-based streamline placement methods [12] and [16] in the fifth and sixth columns, respectively. The goal of the 2D evenly-spaced streamline placement algorithm presented in [12] is to create a visually pleasing rendering while the farthest point seeding algorithm [16] aims at avoiding generating discontinued streamlines. It is note worthy that Liu *et al.* [15] speeded up the method in [12] by creating fewer number of samples using a Hermite polynomial interpolation. According to the authors in [15], their algorithm has a comparable visual quality with the algorithm in [16]. Therefore, here we only compare our results with Mebarkis's algorithm because the implementation of this algorithm is publicly available. Figure 10 (e) and (k) show streamlines generated by the 2D evenly-spaced streamline placement algorithm in [12], and Figure 10 (f) and (l) show results from the farthest point seeding algorithm in [16]. In both of the methods, it can be seen that all regions received an equal weight when placing the streamlines, and hence there is a lack of proper visual focus. In contrast, as shown in Figure 10 (d) and (j), our algorithm emphasizes regions with more information according to the entropy measure and places denser and longer streamlines there. It is also worth to point out that using a distance-based threshold without considering the underlying data saliency can produce short and broken streamlines near critical points, as shown in Figure 10 (e), (f), (k) and (l). This will disrupt the process of contour forming, an important perceptual component for effective flow visualization advocated in [28]. It is worth noting that seeding algorithms based on density control can be extended to highlight flow features by adjusting the distance threshold according to certain importance metrics such as vorticity, curvature, etc. However, such importance metrics do not take into account the redundancy among the streamlines which may result in cluttered rendering in regions of interest. Chen *et al.* use streamline shape as a metric to cut down redundancy [5]. However, the metric proposed there is sensitive to the lengths of streamlines and the distances among them. In contrast, our method evaluates directly the contribution of each individual streamline to the final visualization.

Figure 11 displays the streamlines generated by the three algorithms from the benchmark data *Isabel* for IEEE Vis Design Contest 2004, which simulates the hurricane *Isabel* from September 2003 over the west Atlantic region. A 2D *XY* slice near the ground at time step 20 was chosen to perform this test. The purpose of this figure is to demonstrate that the result of our information analysis allowed our algorithm to highlight salient regions earlier with fewer streamlines compared to the other two algorithms. Figure 11 (a) shows 54 streamlines generated by our method, which is a snapshot taken from the end of the first iteration in our importance-based seeding process. For the same number of streamlines, the results from the evenly-spaced algorithm and the farthest point seeding algorithm are shown in Figure 11 (b) and (c). As can be seen, at this point many salient points that exhibit interesting flow features (highlighted in the red boxes in (a)) have not been captured in those algorithms. We believe our information-theoretic framework can be easily adopted by the existing algorithms such as the two methods we compare with here.

Figure 12 shows the results of streamline seeding and how the conditional entropy change its value as more streamlines are introduced. The data set demonstrated in the top row, denoted as *Circle*, exhibits circular flows, while the data at the bottom, denoted as *Electro*, contains three electrodes. The second column in Figure 12 shows volume rendered images generated from the entropy fields. The color and opacity maps used are shown in the lower left corner of each image. The warm colored region near the top of Figure 12 (b) contains critical points which were picked up early by the entropy field. Figure 12 (c) shows the initial streamlines generated from the template which illustrates the more complex structure in the field. Figure 12 (d) shows the first 50 streamlines produced by our algorithm. As can be seen from Figure 12 (a), at the time when the first 50 streamlines were drawn, the conditional entropy has dropped sharply which implies that most of the essential information from the data has been captured. This allows the user to focus on the more important part of data with minimum occlusion. Figure 12 (e) presents the streamlines after the first iteration of importance-based seeding, which already covers most of the important regions in the field.



(a) (b) (c)

Fig. 13. Streamline rendering results for the dataset *Plume*. In (a), salient streamlines (in warm color) are occluded. In (b) and (c), The occlusion is reduced by modulating the opacity according to the entropy values. The transfer functions used to map the entropy to color/opacity are shown at the upper left corners.

Table 1. Algorithm performance (in seconds) for 2D Datasets

Dataset	Fig. 10 (d)	Fig. 10 (j)	*Isabel*
Dimension	50^2	64^2	250^2
Template	2.0	0.5	14
(#Local Max)	(9)	(2)	(34)
Importance	0.6	0.3	12

Table 2. Algorithm performance (in seconds) for 3D Datasets

Dataset	*Circle*	*Electro*	*Plume*
Dimension	64^3	64^3	$126 \times 126 \times 512$
Template	17	24	480
(#Local Max)	(2)	(3)	(48)
Importance	24	30	584

For the data set *Electro*, Figure 12 (h) shows the initial streamlines generated by the template, which corresponds well with the high entropy regions that contain critical points. Streamlines in Figure 12 (h) can highlight the flow patterns around the three critical points. From the conditional entropy plot in Figure 12 (f), it can be seen that after the first few iterations of importance-based seeding, the conditional entropy went flat and only dropped slightly before it converged. To confirm this, Figure 12 (i) and (j) depict the streamlines after the second and third iterations, which shows very little difference.

Figure 13 shows streamlines generated from another 3D dataset *Plume*, which simulates the thermal downflow plumes on the surface layer of the sun by scientists in National Center for Atmospheric Research. The resolution of the data is $126 \times 126 \times 512$. In Figure 13 (a), the first 600 streamlines generated by our method are shown. The color near the internal region is warmer, indicating higher entropy values and hence more complex flow in that region. In this image, the detailed structure of these warmer colored streamlines are occluded by the blue streamlines with smaller entropy, causing difficulties for more detailed analysis. To reduce the occlusion, the entropy field can be utilized to adjust the color and opacity of the streamline segments and make those streamlines in regions of lower entropy less opaque. Figures 13 (b) and (c) show two rendered images using different transfer functions shown at the upper left corner of each sub-figure. Figure 13 (c) shows a result that maps the low entropy region to a smaller opacity, revealing a clearer structure in the high entropy region, while Figure 13 (b) provides more cues about the streamlines in the outer regions as a context. Our framework allows the user to manipulate the entropy-based transfer function to explore data that have different degrees of saliency.

6.2 Performance

Tables 1 and 2 list the performance of our algorithm for 2D and 3D data sets. All timings were measured on a machine with Intel Core 2 Duo 6700 processor, 3GB system memory, and an nVidia GeForce GTX 280 graphics adapter. In the two tables, the second rows show the timing of template-based seeding and streamline calculation. The number of detected local entropy maxima are shown in parentheses. The third rows show the performance of importance-based seeding. Here we show the average time per iteration in our experiments since different data sets took different iterations to converge. It can be seen that for the data sets generated from the more complex simulations such as *Isabel* and *Plume*, the template seeding stage took longer computation time. This is because in these two data sets, the entropy field contains more local maxima, and many of them are very close to each other in the domain. To accelerate the process, we can use either a higher threshold or certain image denoising methods to remove the less important local maxima. To speed up the process of importance-based seeding, since many streamlines generated in the later iterations are non-essential as shown previously, we can relax the convergence threshold to avoid unnecessary computation.

7 Limitations and Future Work

While in this paper we have demonstrated that information theory can be successfully used to assist the choice of streamline seeds and measuring the information content for both the data input and the visualization output, there exist several limitations in our framework. First, the Shannon entropy measures only consider the statistical properties of the data , i.e., the histograms, but not their spatial distribution. Consequently, two vector fields that have the same histogram will have the same entropy value even if their vectors distribute differently in the spatial domain. To differentiate complex vector fields from simpler ones, considering the local differentials of the vectors such as the Jacobian in addition to the entropy can be beneficial. It is also noteworthy that Shannon's entropy does not consider a shift of histogram bins. Two distributions can have the same entropy value even though their histograms are different (one is created by shifting the bins from the other, for example). Combining Shannon's entropy together with the shape of histogram [22] could be a better metric.

Another limitation of Shannon entropy is that it is designed for discrete random variables. Converting continuous vectors into discrete representation using quantization can make the resulting entropy sensitive to the number of bins used in the histogram. To alleviate this problem, continuous entropy can be used. To compute the continuous entropy, the probability distribution of the density should be estimated in analytic form, which can be done using the non-parametric density estimation methods commonly used for machine learning and pattern classification.

When creating histograms, how the distribution of the data is sampled needs to be done carefully. For efficiency purposes, currently in our framework the histogram is calculated using the data defined on the grid points, even though the input vector field is defined continuously over the domain. A more accurate histogram can be obtained by considering the volume of voxels that match the vector direction of each histogram bin. This is similar to creating the continuous scatterplot for a continuous scalar field [1, 2]. To speed up the computation of continuous scatterplots, the GPU-based approximation described in [2] can be used to accelerate the computation of 2D joint histograms, which can be applied to obtain the orientation histogram of a 2D vector field by first computing the joint histogram of x- and y- components of the vectors and then mapping each bin in the joint histogram to the corresponding orientation. For the orientation histogram of a 3D vector field and the joint histogram of two vector fields, where each location contains two 2D or 3D vectors, further investigation is needed to extend their GPU-based implementation from 2D to higher dimensions. Meanwhile, in spite that a 3D vector orientation can be converted to two angles in spherical coordinates, the interpolation of spherical coordinates is not linear inside the grids, while the experiments in [2] only demonstrate the cases where the scalars are linearly varying. As a result, we plan to investigate the extension for high dimensional data and to test the accuracy of their approximation for vector orientations in our future work.

Further study about the conditional entropy used in our framework

is needed. Since the conditional entropy considers only the joint distribution between the original and the reconstructed vector fields, it ignores the magnitude of errors between these two fields. As a result, a region with high error magnitudes between the two fields can still have a low conditional entropy, which may result in an insufficient number of seeds being placed. Empirically, we have found this case happens primarily near the critical points or regions with larger variations of the vectors. Since those regions will have higher entropy values, more seeds will be placed in the initial stage so we have not seen this problem in our tests.

Additionally, several places in our current implementation can be optimized. For example, we use a fixed kernel size to compute the entropy and conditional entropy, which can be improved by using an adaptive kernel size for different regions and datasets. Besides, currently only flow directions are considered since we are only concerned with streamlines. The magnitude of vectors needs to be taken into account in the case of time-varying data sets, a direction of our future work.

Finally, besides streamline placement, we believe that the proposed information-theoretic framework can potentially benefit other flow visualization techniques. Information-aware glyph placement can be an effective extension to the conventional glyph-based algorithms by using the entropy measures to control the density of glyphs in local regions. The information measure can also be used to enhance the rendering of 3D Line Integral Convolution (LIC) results, where better transfer functions can be designed to enhance regions with more information. We believe it is possible to extend our method to handle time-varying data and guide the placement of pathlines or streaklines.

8 Conclusions

In this paper, we present an information-theoretic framework for flow visualization. Our framework allows quantitative measurements of information content in the input flow field, which in turn can be used to assist effective seeding of 2D and 3D streamlines. With our framework, the distribution of streamlines is controlled by the information content of the data so that more streamlines will be seeded in regions with higher information content. We compared our framework with two commonly used streamline seeding algorithms, and showed that it is possible to place fewer streamlines while still capturing salient features in the field. Based on the information measurement, methods to enhance rendering of 3D streamlines are proposed. By increasing the visibility of streamlines in regions with higher information content, 3D visual clutter and occlusion can be reduced.

Acknowledgments

The authors would like to thank Torsten Möller, Carrie Stein, and the anonymous reviewers for their comments. This work was supported in part by NSF ITR Grant ACI-0325934, NSF RI Grant CNS-0403342, NSF Career Award CCF-0346883, and DOE SciDAC grant DE-FC02-06ER25779.

References

[1] S. Bachthaler and D. Weiskopf. Continuous scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1428–1435, 2008.

[2] S. Bachthaler and D. Weiskopf. Efficient and adaptive rendering of 2D continuous scatterplots. *Computer Graphics Forum*, 28(3):743–750, 2009.

[3] U. Bordoloi and H.-W. Shen. View selection for volume rendering. In *Vis '05: Proceedings of the IEEE Visualization 2005*, pages 487–494, 2005.

[4] A. H. Charles and T. A. Porsching. *Numerical Analysis of Partial Differential Equations*. Prentice Hall, Engelwood, 1990.

[5] Y. Chen, J. Cohen, and J. Krolik. Similarity-guided streamline placement with error evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1448–1455, 2007.

[6] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 99th edition, August 1991.

[7] M. Feixas, E. del Acebo, P. Bekaert, and M. Sbert. An information theory framework for the analysis of scene complexity. *Computer Graphics Forum*, 18(3):95–106, 1999.

[8] S. Furuya and T. Itoh. A streamline selection technique for integrated scalar and vector visualization. In *Vis '08: IEEE Visualization 2008 (Poster)*, 2008.

[9] S. Gumhold. Maximum entropy light source placement. In *VIS '02: Proceedings of the IEEE Visualization 2002*, pages 275–282, 2002.

[10] H. Jänicke, A. Wiebel, G. Scheuermann, and W. Kollmann. Multifield visualization using local statistical complexity. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1384–1391, 2007.

[11] S. Jeschke, D. Cline, and P. Wonka. A GPU Laplacian solver for diffusion curves and poisson image editing. *ACM Transactions on Graphics*, 28(5):1–8, 2009.

[12] B. Jobard and W. Lefer. Creating evenly-spaced streamlines of arbitrary density. In *Proceedings of Eighth Eurographics Workshop on Visualization in Scientific Computing*, pages 45–55, 1997.

[13] P. Leopardi. A partition of the unit sphere into regions of equal area and small diameter. *Electronic Transactions on Numerical Analysis*, 25:309–327, 2006.

[14] L. Li and H.-W. Shen. Image-based streamline generation and rendering. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):630–640, 2007.

[15] Z. Liu, R. Moorhead, and J. Groner. An advanced evenly-spaced streamline placement algorithm. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):965–972, 2006.

[16] A. Mebarki, P. Alliez, and O. Devillers. Farthest point seeding for efficient placement of streamlines. In *Vis '05: Proceedings of the IEEE Visualization 2005*, pages 479–486, 2005.

[17] A. Orzan, A. Bousseau, H. Winnemöller, P. Barla, J. Thollot, and D. Salesin. Diffusion curves: a vector representation for smooth-shaded images. *ACM Transactions on Graphics*, 27(3):1–8, 2008.

[18] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever. Mutual information based registration of medical images: A survey. *IEEE Transcations on Medical Imaging*, 22(8):986–1004, 2003.

[19] J. Rigau, M. Feixas, and M. Sbert. Informational aesthetics measures. *IEEE Computer Graphics and Applications*, 28(2):24–34, 2008.

[20] R. Y. Rubinstein. *Simulation and the Monte Carlo Method*. John Wiley & Sons, 2008.

[21] M. Schlemmer, M. Heringer, F. Morr, I. Hotz, M. Hering-Bertram, C. Garth, W. Kollmann, B. Hamann, and H. Hagen. Moment invariants for the analysis of 2D flow fields. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1743–1750, 2007.

[22] M. Sezgin and B. Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, 13(1):146–168, 2004.

[23] B. Spencer, R. S. Laramee, G. Chen, and E. Zhang. Evenly spaced streamlines for surfaces: An image-based approach. *Computer Graphics Forum*, 28(6):1618–1631, 2009.

[24] P.-P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Automatic view selection using viewpoint entropy and its applications to image-based modelling. *Computer Graphics Forum*, 22(4):689–700, 2003.

[25] V. Verma, D. T. Kao, and A. Pang. A flow-guided streamline seeding strategy. In *Vis '00: Proceedings of the IEEE Visualization 2000*, pages 163–170, 2000.

[26] I. Viola, M. Feixas, M. Sbert, and M. E. Gröller. Importance-driven focus of attention. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):933–940, 2006.

[27] C. Wang, H. Yu, and K.-L. Ma. Importance-driven time-varying data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1547–1554, 2008.

[28] C. Ware. Toward a perceptual theory of flow visualization. *IEEE Computer Graphics and Applications*, 28(2):6–11, 2008.

[29] K. Wu, Z. Liu, S. Zhang, and R. J. Moorhead II. Topology-aware evenly-spaced streamline placement. *IEEE Transactions on Visualization and Computer Graphics*, 16(5):791–801, 2010.

[30] C. Xu and J. L. Prince. Gradient vector flow: A new external force for snakes. In *CVPR '97: Proceedings of the IEEE Computer Vision and Pattern Recognition 1997*, page 66, 1997.

[31] X. Ye, D. T. Kao, and A. Pang. Strategy for seeding 3D streamlines. In *Vis '05: Proceedings of the IEEE Visualization 2005*, pages 471–478, 2005.