A Unified Approach to Streamline Selection and Viewpoint Selection for 3D Flow Visualization

Jun Tao, *Student Member*, *IEEE*, Jun Ma, *Student Member*, *IEEE*,
Chaoli Wang, *Member*, *IEEE*, and Ching-Kuang Shene, *Member*, *IEEE Computer Society*

Abstract—We treat streamline selection and viewpoint selection as symmetric problems which are formulated into a unified information-theoretic framework. This is achieved by building two interrelated information channels between a pool of candidate streamlines and a set of sample viewpoints. We define the streamline information to select best streamlines and in a similar manner, define the viewpoint information to select best viewpoints. Furthermore, we propose solutions to streamline clustering and viewpoint partitioning based on the representativeness of streamlines and viewpoints, respectively. Finally, we define a camera path that passes through all selected viewpoints for automatic flow field exploration. We demonstrate the robustness of our approach by showing experimental results with different flow data sets, and conducting rigorous comparisons between our algorithm and other seed placement or streamline selection algorithms based on information theory.

Index Terms—Flow visualization, information channel, streamline selection, viewpoint selection, camera path

1 Introduction

Flow simulation plays an important role in many scientific and engineering disciplines such as climate modeling, turbulent combustion, and automobile design. To visualize data generated from these simulations, one common method is to display flow lines such as streamlines computed from particle tracing. This method gains its popularity because streamlines are easy to compute via standard numerical integration and can be rendered at interactive frame rates. We refer interested readers to a survey by McLoughlin et al. [18] for integration-based flow visualization techniques.

Effective streamline visualization can be formulated as the problem of *seed placement* or *streamline selection*. Seed placement aims at carefully placing seeds in the domain to generate streamlines that capture flow features. Streamline selection aims at carefully selecting streamlines from a large streamline pool for effective display. Streamline seeding for 2D and 3D vector fields has been well studied and continues to receive much attention [11], [15], [19], [26], [28], [33], [34]. Compared to selecting seeds, selecting streamlines is directly related to the final visualization results. Seed placement algorithms were first proposed about fifteen years ago. With the rapid advances of general-purpose computing on GPUs, it is quite affordable nowadays to generate a large pool of streamlines. As such, streamline selection has become a promising alternative to seed placement and has received increasing attention [4], [13], [17]. Besides streamline selection, selecting good viewpoints is also critical for understanding large and complex 3D flow fields. This is because automatically guiding the viewers to good viewpoints improves both the speed and the efficiency of data understanding. While viewpoint selection for volume data has been extensively studied [2], [10], [25], [29], the same issue for flow visualization remains to be thoroughly investigated.

In this paper, we present a unified information-theoretic framework, which solves the problems of streamline selection and viewpoint selection by constructing two interrelated *information channels* between a set of streamlines and a set of viewpoints. Based on the information channel from streamline to viewpoint, we define *streamline information* (SI) as a measure of streamline quality to guide streamline selection. Similarly, in the inverted channel from viewpoint to streamline, we define *viewpoint information* to guide viewpoint selection for the selected streamlines. Leveraging the two channels, we also present a unified algorithm for streamline clustering and viewpoint partitioning. In addition, a camera path is designed for automatic exploration of the flow field. Our approach results in a rigorous and robust framework for selecting good streamlines and viewpoints, clustering streamlines, and partitioning viewpoints, which we demonstrate with flow fields of different characteristics. We also compare out results against results produced from other information theory-based methods, through both objective comparison and subjective evaluation, to show the effectiveness of our approach.

• *The authors are with the Department of Computer Science, Michigan Technological University, Houghton, MI 49931.*
E-mail: {junt, junm, chaoliw, shene}@mtu.edu.

Manuscript received 20 July 2011; revised 11 Jan. 2012; accepted 4 June 2012; published online 12 June 2012.
Recommended for acceptance by D. Weiskopf.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2011-07-0162.
Digital Object Identifier no. 10.1109/TVCG.2012.143.

2 Related Work

Seed placement. Seed placement is a widely used strategy in flow visualization. Early work includes image-guided [26] and evenly spaced [11] streamline placements. In a similar spirit, Mebarki et al. [19] suggested to place a seed at the position that is farthest from all existing streamlines, i.e., the center of largest void area. Verma et al. [28] introduced

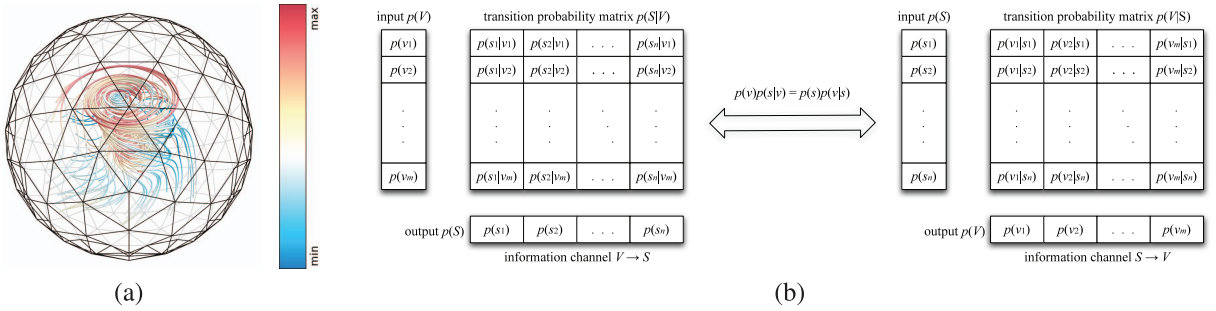


Fig. 1. We model the problems of streamline selection and viewpoint selection in a single, unified framework. (a) Sample viewpoints are constructed along a sphere from the recursive discretization of an icosahedron. Velocity magnitudes are mapped to streamline colors. (b) The information channel $V \rightarrow S$ (left) and the inverted channel $S \rightarrow V$ (right) are connected via the Bayes theorem.

a feature-based approach which detects critical points and uses seeding templates to capture the 2D flow field features. This approach was extended to 3D vector fields by Ye et al. [34]. Li and Shen [15] placed seeds on a 2D projection plane and unprojected the seeds back to the 3D vector field to avoid clutter. Xu et al. [33] used seeding templates for regions with high entropy and then placed additional seeds at locations where the conditional entropy is high, i.e., much information is still unrevealed. Other seeding techniques include priority seeding [23], dual seeding [21], and surface seeding [24].

Streamline selection. An alternative to seed placement is streamline selection. Previously, Chen et al. [4] defined a metric for local similarity between streamlines and used it to explicitly control the streamline density displayed. Marchesin et al. [17] measured the contribution of each streamline to the understanding of the vector field, and selected those streamlines that have higher contribution and lower probability leading to visual clutter. Lee et al. [13] proposed to generate a maximum entropy projection buffer and then select the streamlines that cause the minimum occlusion to important regions. Unlike these view-dependent solutions [13], [17], we select streamlines in a *view-independent* manner by considering their contributions to all sample viewpoints.

Viewpoint selection. Solutions to viewpoint selection have been proposed for the problem of modeling a 3D object from range data [31] and from images [7], for object recognition [1], and for cinematography [8]. Vázquez et al. [27] defined the viewpoint entropy from the projected areas of the faces of the geometric models in the scene to derive good viewpoints. Bordoloi and Shen [2] presented a view selection method for volume rendering, in which a voxel-based entropy function was used to evaluate each viewpoint. Takahashi et al. [25] proposed to decompose an entire volume into a set of feature components, and then compromise between the locally optimal viewpoints for the components to achieve a globally best viewpoint. Ji and Shen [10] suggested a method that combines static view selection with dynamic programming to select time-varying viewpoints and produce a smooth animation.

Information channel. An overview of information theory in visualization can be found in [3], [30]. Our work is inspired by previous work on volumetric and polygonal data that were built upon similar information channels [6], [22], [29]. Viola et al. [29] used an information channel built between viewpoints and volumetric objects and evaluated viewpoints using mutual information computed from that channel. Feixas et al. [6] proposed an information-theoretic framework for polygonal data in which a channel from viewpoints to polygons and its inverted channel were built, and mutual information of viewpoints and polygons were defined, respectively. Ruiz et al. [22] applied a similar method to define voxel information in volume visualization. Several challenges exist when applying this information-theoretic framework to flow fields. Unlike voxels which are fine-grained elements and polygons which are fairly localized data items, a streamline could stretch across the entire field and have a very complex shape. This makes it difficult to analyze the conditional probability $p(s|v)$ for a streamline. In addition, there exists no inherent concept of neighbors for streamlines because no connectivity information is given. We contribute to the state-of-the-art flow visualization by introducing a new way to evaluate streamline information and viewpoint information, and presenting a unified framework for streamline selection and viewpoint selection.

3 Our Approach

3.1 Information Channel

We solve the problems of streamline selection and viewpoint selection in a single, unified framework. We consider a set of streamlines $S = \{s_1, s_2, \ldots, s_n\}$ and a set of viewpoints $V = \{v_1, v_2, \ldots, v_m\}$ as discrete random variables and build two interrelated information channels between them: $V \rightarrow S$ and $S \rightarrow V$. Our assumptions for viewpoints are 1) the flow field is centered in a sphere of sample viewpoints constructed from the recursive discretization of an icosahedron; and 2) the camera at a sample viewpoint is looking at the center of the sphere. Fig. 1a shows sample viewpoints along the sphere. Throughout this paper, we use modified spectral colors [16] for streamline coloring based on the velocity magnitude.

The main components in the information channel $V \rightarrow S$ are the following:

- The *transition probability matrix* $p(S|V)$ where conditional probability $p(s|v)$ represents the probability of "seeing" streamline s from viewpoint v (i.e., the importance of s with respect to v).
- The *input probability distribution* $p(V)$ where $p(v)$ represents the probability of selecting viewpoint v. If

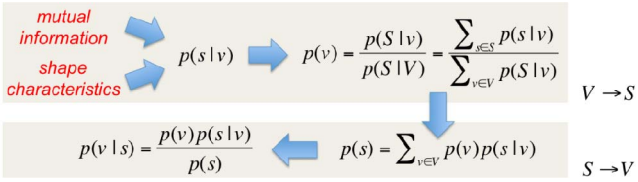


Fig. 2. The order of computing the probabilities for the two channels.

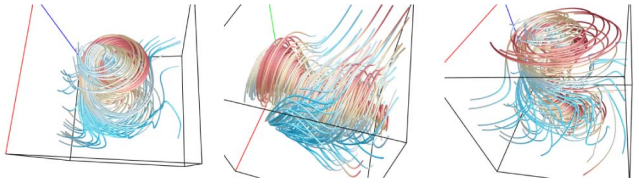


Fig. 3. The best viewpoint selection result based on $p(v)$ that considers mutual information only (left), shape characteristics only (middle), and both mutual information and shape characteristics (right).

we assume $p(v)$ to be evenly distributed, then $p(v) = 1/m$ where m is the number of sample viewpoints.

- The *output probability distribution* $p(S)$ where $p(s)$ represents the average probability that streamline s is seen from all viewpoints V. That is, $p(s) = \sum_{v\in V} p(v)p(s|v)$.

Similarly, we can construct the inverted information channel $S \rightarrow V$, where the input and output probability distributions are swapped: $p(S)$ becomes the input and $p(V)$ becomes the output. In this inverted channel, the new transition probability matrix is $p(V|S)$, where $p(v|s)$ represents the probability of selecting viewpoint v given streamline s. As shown in Fig. 1b, these two channels are connected via the Bayes theorem, i.e., $p(v)p(s|v) = p(v,s) = p(s,v) = p(s)p(v|s)$, which provides us a means to compute $p(v|s)$ given $p(v)$, $p(s)$, and $p(s|v)$.

3.2 Conditional Probability Definition

The key for deriving the information channel $V \rightarrow S$ lies in how to define the conditional probability $p(s|v)$. In our scenario, we consider the following two view-dependent factors for computing $p(s|v)$:

Mutual information. This measure, denoted as $I(s; s_v)$, indicates how much information about streamline s is revealed in its 2D projection s_v under viewpoint v. We know that information loss is inevitable due to streamline projection. A large value of $I(s; s_v)$ shows that 3D streamline s itself contains a high amount of information and its 2D projection s_v preserves well the information of s. Therefore, the probability of "seeing" s from v is high. Conversely, if s itself contains a low amount of information or its 2D projection s_v loses much of the information of s, then the probability of "seeing" s from v is low. $I(s; s_v)$ is defined as [5]

$$I(s; s_v) = \sum_{i\in s} \sum_{j\in s_v} p(i,j) \log \frac{p(i,j)}{p(i)p(j)}, \tag{1}$$

where $p(i)$ and $p(j)$ are the marginal probabilities of s and s_v, respectively, and $p(i,j)$ is their joint probability. Here, we treat a streamline as a finite set of points. That is, i and j loop through the lists of points obtained either from streamline tracing or parameterization by the arc length along s and s_v, respectively. To compute $p(i)$, we interpolate vectors from the original flow data based on the positions of all the points along s. These vectors are used to construct a 2D histogram based on vector magnitude and direction. To compute $p(j)$, we use the projections of these vectors along s_v to construct the corresponding 2D histogram. To quantize vector directions, we use the recursive discretization of an icosahedron for 3D quantization, and the even circle partition by angle for 2D quantization. All vectors falling into the same range are quantized into the same bin of vector direction. The joint probability $p(i,j)$ can be computed by constructing a joint histogram for s and s_v where each of the two axes consists of all vector direction and magnitude combinations. In the joint histogram, the normalized bin count corresponds to $p(i,j)$.

Shape characteristics. This property indicates how stereoscopic the shape of streamline s is reflected under viewpoint v. Since the number of points along each streamline could be fairly large (e.g., in the order of hundreds or thousands of points), we opt to approximate a streamline using its *skeleton* for fast shape characteristics analysis. The "skeleton" of a streamline is obtained using a uniform subsampling scheme along the integration points of the streamline to reduce the number of points to a smaller scale (e.g., in the order of tens of points). Let us denote the skeleton of streamline s as $\tilde{s} = \{\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_k\}$, the viewing vector as $\overrightarrow{v}$, and the angle between $\overrightarrow{v}$ and $\overrightarrow{\tilde{p}_i\tilde{p}_{i+1}}$ as θ. We define the shape characteristics of $\overrightarrow{\tilde{p}_i\tilde{p}_{i+1}}$ as

$$\alpha_{\tilde{p}_i\tilde{p}_{i+1};v} = \alpha_{\min} + (1.0 - \alpha_{\min})\left(1.0 - \frac{|\pi/4 - \theta'|}{\pi/4}\right), \tag{2}$$

where $\alpha_{\min}$ is the minimum value for the shape characteristics (we set $\alpha_{\min} = 0.1$ in this paper) and

$$\theta' = \begin{cases} \pi - \theta, & \theta > \pi/2 \\ \theta, & \theta \leq \pi/2. \end{cases} \tag{3}$$

The intuition is that $\alpha_{\tilde{p}_i\tilde{p}_{i+1};v}$ gets its maximum (minimum) value of 1.0 ($\alpha_{\min}$) when $\overrightarrow{v}$ and $\overrightarrow{\tilde{p}_i\tilde{p}_{i+1}}$ form a 45 or 135 degrees (0, 90, or 180 degrees) angle. The shape characteristics of streamline skeleton $\tilde{s}$ is defined as

$$\alpha_{\tilde{s};v} = \frac{\sum_{i=1}^{k-1} \alpha_{\tilde{p}_i\tilde{p}_{i+1};v} \|\tilde{p}_i\tilde{p}_{i+1}\|}{\sum_{i=1}^{k-1} \|\tilde{p}_i\tilde{p}_{i+1}\|}. \tag{4}$$

Conditional probability. With mutual information and shape characteristics defined for streamline s under viewpoint v, we define conditional probability $p(s|v)$ as

$$p(s|v) = \frac{\alpha_{\tilde{s};v} I(s; s_v)}{\sum_{s\in S} \alpha_{\tilde{s};v} I(s; s_v)}. \tag{5}$$

With $p(s|v)$ defined, besides simply assuming $p(v) = 1/m$, we can also obtain $p(v)$ from the normalization of the summation of all streamlines' conditional probabilities under v over all viewpoints V. That is, $p(v) = p(S|v)/p(S|V)$, where $p(S|v) = \sum_{s\in S} p(s|v)$ and $p(S|V) = \sum_{v\in V} p(S|v)$. We use this nonuniform specification of $p(v)$ in our work. Fig. 2 summarizes the order of computing the probabilities for the two channels.

Fig. 3 shows a comparison of selecting the best viewpoint based on $p(v)$ with considering mutual information only, shape characteristics only, and both. When only mutual information is considered, the main axis of the

tornado is almost parallel to the viewing vector, making $\overrightarrow{v}$ and $\overrightarrow{\tilde{p}_i\tilde{p}_{i+1}}$ form an almost 0 or 180 degrees angle. $p(s|v)$ achieves its maximum for almost every streamline, letting $p(v)$ get its highest value. When only shape characteristics is considered, $\overrightarrow{v}$ and $\overrightarrow{\tilde{p}_i\tilde{p}_{i+1}}$ now form an almost 45 or 135 degrees angle. The best viewpoint selected is still not desirable. When considering both mutual information and shape characteristics into $p(v)$ evaluation, we can select the more desirable best viewpoint as the overall structure of the tornado is best perceived.

3.3 Best Streamlines Selection

For streamline selection, we start from a pool of randomly or uniformly traced streamlines and select the *best streamlines* for display. For streamline tracing, we use the Runge-Kutta method to integrate streamlines as long as possible until they leave the domain or reach critical points. The "best" streamlines are those that best capture flow features by passing through the vicinity of critical points or interesting regions. In this section, we propose two methods to evaluate each individual streamline and then introduce our selection process.

Our first method uses the probability distribution $p(S)$. Since $p(s|v)$ indicates how interesting streamline s is from viewpoint v, $p(s)$ gives us the summation of importance of s from all viewpoints V. If the distribution $p(V)$ is not uniform, $p(s)$ can be considered as a weighted summation, in which a more interesting viewpoint has a higher weight.

Our second method uses the *streamline information*. In the information channel $S \rightarrow V$, we define SI as

$$I(s;V) = \sum_{v \in V} p(v|s) \log \frac{p(v|s)}{p(v)}, \tag{6}$$

which represents the degree of dependence between streamline s and the set of viewpoints V. Intuitively, SI indicates the quality of s with respect to V. Note that $I(s;V)$ is the contribution of streamline s to $I(S;V)$, which expresses the degree of correlation between the set of streamlines S and the set of viewpoints V. Low values of SI correspond to streamlines seen by a large number of viewpoints in a *balanced* way. The term "balance" indicates that the conditional probability distribution $p(V|s)$ is similar to $p(V)$. This similarity can be expressed by the Kullback-Leibler divergence [12] between $p(V|s)$ and $p(V)$, which equals zero when $p(V|s) = p(V)$. Conversely, a high value of $I(s;V)$ means a high degree of dependence between streamline s and the set of viewpoints V. Therefore, streamline s that shows more information over the set of viewpoints V have a lower value of SI. The advantage of this streamline information over the streamline entropy, i.e., $H(V|s) = -\sum_{v \in V} p(v|s) \log p(v|s)$, lies in its robustness to deal with any type of discretization or resolution of the viewpoints V. This property has been shown by Viola et al. [29] in the context of volume visualization.

After streamline evaluation, we sort all the streamlines S into a priority queue. If $p(s)$ is used, the streamlines are sorted in the *decreasing* order of $p(s)$, where a streamline with a higher value of $p(s)$ is preferred. If SI is used, the streamlines are sorted in the *increasing* order of SI, since a streamline with a lower value of SI is better.

We can now select the best streamlines according to the sorted order, but the similarities between streamlines are still not taken into account. It is very likely that two or more streamlines which are close to each other in the space and have a similar shape would be selected together for display. However, these streamlines are not only redundant but also likely to cause occlusion and clutter. Therefore, we check the pairwise dissimilarity between two streamlines to avoid selecting streamlines that are very similar to each other. To measure streamline dissimilarity, we use the *mean of closest point distances* as suggested by Moberts et al. [20] in DTI fiber clustering. Our selection process starts from selecting the first streamline in the priority queue. Then, we check the next streamline and select it if its distance to the first one is larger than a given distance threshold d_s. At each step, we consider one new streamline, and compute the distance between it and every streamline previously selected. This streamline is selected if and only if the distances are all larger than d_s. The selection process stops when a given number of streamlines is selected or all streamlines in the pool are considered.

3.4 Best Viewpoints Selection

Similar to SI, in the information channel $V \rightarrow S$, we define the *viewpoint information* as

$$I(v;S) = \sum_{s \in S} p(s|v) \log \frac{p(s|v)}{p(s)}, \tag{7}$$

which represents the degree of dependence between viewpoint v and the set of streamlines S. Note that in our scenario, the set of streamlines now is actually the set of *selected* streamlines, not the original pool of streamlines. This corresponds to 1) removing all rows in the transition probability matrix $p(V|S)$ in the channel $S \rightarrow V$ and the input probability distribution $p(S)$ as shows in Fig. 1b for all streamlines that are not selected; and 2) renormalizing all remaining $p(s)$ in $p(S)$ and recomputing all $p(v)$ in the output probability distribution $p(V)$. For simplicity, we still use the notation S in this section when referring to the selected streamlines.

Similar to streamline selection, the best viewpoints can be defined either by $p(v)$ or VI. If we use $p(v)$ to select the best viewpoints, we mainly consider the amount of information about the set of streamlines S revealed by viewpoint v. As a result, the best viewpoints are those that show more information of S than others. If we use VI to select best viewpoints, VI indicates the quality of viewpoint v with respect to the set of streamlines S. Low (high) values of VI correspond to more independent (coupled) viewpoints. Thus, viewpoints with low values of VI are considered as better ones.

Till now, our algorithm could select very similar neighboring viewpoints as the best viewpoints which is clearly not desirable. To avoid this, we make use of the distribution $p(S|v)$ computed in the last step. Considering $p(S|v)$ as a vector associated with each viewpoint, i.e., $p(S|v) = <p(s_1|v), p(s_2|v), \ldots, p(s_n|v)>$, the difference between two viewpoints can be expressed as the Euclidean distance between their corresponding vectors. Thus, a viewpoint is not selected if its distance to any of the selected viewpoints falls below a given threshold d_v.

3.5 Streamline Clustering

We propose a streamline clustering algorithm using the information channels built between S and V. The first stage of our algorithm is to find the *representative streamlines*. Unlike the "best" streamlines (Section 3.3) which are evaluated individually, the "representative" streamlines are defined as a small set of streamlines in which the streamlines as an entirety best characterize the flow field. This is formed by selecting the streamlines such that their merging minimizes the distance to the target distribution $p(V)$. That is, our selection algorithm should select n' streamlines $(n' << n)$ so that their merging $\hat{s}$ minimizes $I(\hat{s}; V)$. Since finding an optimal solution to this problem is NP-complete [22], we adopt a greedy strategy by selecting successive streamlines to minimize $I(\hat{s}; V)$. At each merging step, we aim to maximize the Jensen-Shannon divergence between the set of previously merged streamlines and the new streamline to be selected.

Our solution proceeds as follows: First, we select the best streamline s_1 with distribution $p(V|s_1)$ corresponding to the minimum $I(s; V)$. Next, we select s_2 such that the mixed distribution $\frac{p(s_1)}{p(\hat{s})}p(V|s_1) + \frac{p(s_2)}{p(\hat{s})}p(V|s_2)$ minimizes $I(\hat{s}; V)$, where $\hat{s}$ represents the merging of s_1 and s_2 and $p(\hat{s}) = p(s_1) + p(s_2)$. At each step, a new mixed distribution

$$\frac{p(s_1)}{p(\hat{s})}p(V|s_1) + \frac{p(s_2)}{p(\hat{s})}p(V|s_2) + \cdots + \frac{p(s_i)}{p(\hat{s})}p(V|s_i), \qquad (8)$$

where $p(\hat{s}) = p(s_1) + p(s_2) + \cdots + p(s_i)$, is produced until the *streamline information ratio* (SIR), denoted as $I(\hat{s}; V)/I(S; V)$, is lower than a given threshold or we have selected n' streamlines. The SIR can be interpreted as a measure of the representativeness of the selected streamlines.

The second stage of our algorithm is to cluster other streamlines to the representatives we have identified in the first stage. Following the data processing inequality [5], we know that any clustering of streamlines reduces the mutual information $I(S; V)$ between the set of streamlines S and the set of viewpoints V. Therefore, a good clustering is the one that minimizes this mutual information loss. Assuming that two streamlines s_1 and s_2 are merged into one cluster $\hat{s}$, the reduction of mutual information can be described by

$$\begin{aligned}\delta I(s_1; s_2) &= I(S; V) - I(\hat{S}; V) \\ &= p(s_1)I(s_1; V) + p(s_2)I(s_2; V) \\ &\quad - p(\hat{s})I(\hat{s}; V),\end{aligned} \qquad (9)$$

where $\hat{S}$ is the resulting streamline set and $p(\hat{s}) = p(s_1) + p(s_2)$. Note that $\delta I(s_1; s_2)$ is small if the two streamlines have very similar distributions, i.e., $p(V|s_1) \approx p(V|s_2)$, and it reaches zero if the two streamlines share the same distribution, i.e., $p(V|s_1) = p(V|s_2)$. At each step, we pick a streamline s and calculate $\delta I(s; s')$ for each of the streamlines s' in the representative set. Then, s is merged into the cluster in which $\delta I(s; s')$ between s and its representative s' is minimal.

We use the elbow criterion to determine the proper number of clusters. That is, we should choose a number of clusters so that adding another cluster does not greatly increase the percentage of variance explained (i.e., the ratio of the between-group variance to the total variance). Specifically, if we plot the percentage of variance explained by the clusters against the number of clusters, the first few clusters will add much information (explain a lot of variance), but at some point the marginal gain will drop, giving an angle in the graph (the elbow). In practice, we run from two to 10 clusters from which we choose the appropriate number of clusters.

3.6 Viewpoint Partitioning

Similar to streamline clustering, we can perform viewpoint partitioning in two stages. The first stage is the selection of representative viewpoints and the second stage is clustering other viewpoints to the representatives. The most representative viewpoints are a small number of viewpoints $(m' << m)$ that provide the best representation of the selected streamlines. Leveraging the VI measure (7), our solution for viewpoint selection is the same as the greedy solution we propose for identifying representative streamlines (Section 3.5) with the only difference being the swap of notations for streamline and viewpoint. The viewpoint selection process stops when the *viewpoint information ratio* (VIR), denoted as $I(\hat{v}; S)/I(V; S)$, is lower than a given threshold or we have selected m' viewpoints. Similar to the SIR, the VIR can be interpreted as a measure of the representativeness of the selected viewpoints.

For viewpoint partitioning, we measure the difference between two viewpoints by the reduction of mutual information, where the reduction $\delta I(v_1; v_2)$ is defined in the same way as $\delta I(s_1; s_2)$ (9). Then, we apply the same procedure of streamline clustering to partitioning viewpoints in a similar manner: at each step, a viewpoint v is merged into the partition whose representative v' minimizes the information loss measured by $\delta I(v; v')$. Similarly, we use the elbow criterion to identify the proper number of partitions for all viewpoints.

3.7 Camera Path

Given a set of best (Section 3.4) or representative (Section 3.6) viewpoints, we construct a smooth camera path that goes through all selected viewpoints for automatic flow field exploration. Our algorithm creates a graph by treating all sample viewpoints as nodes and their neighboring relationships as edges. The weight of an edge is defined as the Jensen-Shannon divergence between the two viewpoints. With this graph, we can define the camera path by finding the shortest path among the set of selected viewpoints using Dijkstra's algorithm. Specifically, we use the best (or the most representative) viewpoint as the starting point, and find the nearest viewpoint (with the minimum Jensen-Shannon divergence) from selected viewpoints as the next target viewpoint. The path between these two viewpoints is derived from the shortest path between their corresponding nodes in the graph. When the first target viewpoint is achieved, we select a new target viewpoint among the rest of selected viewpoints and proceed in the same way until all viewpoints have been considered.

4 Results

4.1 Timing Performance and Parameter Choices

We experimented our approach with 10 flow data sets of different sizes and characteristics. For all these data sets, the

TABLE 1
The 10 Flow Data Sets and Their Parameter Values

data set	dimension	initial # lines	selected # lines	rep. # lines	avg. # pts.	rep. #views	line distance threshold d_s	view distance threshold d_v
five critical points	$51 \times 51 \times 51$	800	140	5	112.9	3	4.1	0.2
tornado	$64 \times 64 \times 64$	500	60	7	295.2	3	5.8	0.1
two swirls	$64 \times 64 \times 64$	500	100	6	157.3	3	3.8	0.18
supernova	$100 \times 100 \times 100$	500	140	5	184.5	4	4.5	0.15
car flow	$368 \times 234 \times 60$	600	140	5	185.5	3	5.9	1.0
crayfish	$322 \times 162 \times 119$	800	100	7	208.7	3	18.7	0.15
solar plume	$126 \times 126 \times 512$	600	140	4	100.2	4	15.0	0.1
computer room	$417 \times 345 \times 60$	800	200	6	172.9	4	17.3	0.15
hurricane	$500 \times 500 \times 100$	600	140	5	341.1	3	31.6	0.15
ABC flow	$1024 \times 1024 \times 1024$	800	140	7	179.8	4	68.3	0.15

All use 320 initial viewpoints. For d_s, we use the average of all pairwise mean of closest point distances between streamlines divided by a constant factor. Refer to Section 4.1 for detail.

initial pool of streamlines was generated by dense placement of seeds randomly. In Table 1, we list the parameter values used for each data set. We used 320 viewpoints for all data sets, and determined the number of initial streamlines based on the SIR. Normally, a larger number of initial streamlines should be generated for a data set with a larger spatial dimension or a more complicated structure. For the streamline distance threshold d_s, we used the average of all pairwise mean of closest point distances between streamlines divided by a constant factor. This constant factor should not be too small, in case that the distance becomes the dominant factor in streamline selection. It should not be too large, so that it remains effective in reducing the occlusion. Our experiment shows that data sets containing more features or patterns need a smaller constant factor, so that the distance threshold is larger. Unlike the streamline distance threshold, the viewpoint distance threshold d_v was chosen as a constant, since the average distance between viewpoints does not vary that much for different data sets. Please refer to Table 1 for the actual values we used for d_s and d_v for each data set.

Table 2 shows the timing results on three benchmark data sets for both CPU and GPU versions of our implementation, tested on a PC with an Intel Core 2 Q6600 quad-core CPU running at 2.4 GHz and an nVidia GeForce GTX 465 graphics card. Although the ideal speedup is the same for these three tasks, the experiment shows that streamline selection based on both $P(s)$ and $I(S;V)$ gains a more significant speedup than streamline clustering does. This might be the result of the following two facts. First, streamline clustering, which is sequential in nature, requires more synchronizations than streamline selection does. Second, most tasks of streamline selection are free from control flow instructions, i.e., containing just one branch, which benefits more from the CUDA hardware structure, since every streaming multiprocessor runs threads in lockstep.

Fig. 4 shows the SIR and VIR with respect to the number of representative streamlines and viewpoints selected, respectively. From the SIR plot, we see that when the number of representative streamlines selected is small (less than 100), a larger pool of initial streamlines can reduce the SIR more effectively. This is because as we have more streamlines to choose from, we are more likely to select a better set of representative streamlines to represent the flow field. As the number of representative streamlines selected continues to increase, the reduction of SIR is not significant anymore. We conclude that using an initial pool of 500 to 1,000 streamlines is sufficient for the computer room data set, and selecting about 120 to 140 streamlines suffices as the SIR is below 0.0002. Such an experiment provides us with a quantitative means to determine the appropriate numbers of initial and selected streamlines. From the VIR plot, we see that the VIR is already less than 0.01 for the five critical points data set when 10 representative viewpoints are selected. It is clear that using 80 sample viewpoints is actually sufficient in terms of VIR reduction since the differences among the three curves are not substantial. In practice, we chose 320 viewpoints instead for smoother view sphere rendering and camera path planning.

TABLE 2
The Timing Result for Different Data Sets

	task	tornado	crayfish	hurricane
CPU	best selection $P(S)$	137min	185min	324min
	best selection $I(S;V)$	165min	248min	368min
	clustering	4.0sec	6.4sec	2.5sec
GPU	best selection $P(S)$	6.2sec	9.3sec	7.0sec
	best selection $I(S;V)$	6.2sec	9.3sec	7.0sec
	clustering	0.01sec	0.01sec	0.008sec

The time for both best streamlines selection based on $P(S)$ and that based on $I(S;V)$ include the cost for streamline evaluation. The time for clustering includes representative streamline selection, which dominates the clustering time.

4.2 Streamlines Selection Results

Fig. 5 shows the comparison of streamline selection results for four different methods: best selections based on $p(s)$ and $I(s;V)$, representative (REP) selection, and random selection. For the hurricane data set, both selections based on $p(s)$ and $I(s;V)$ yield similar results. The two circling

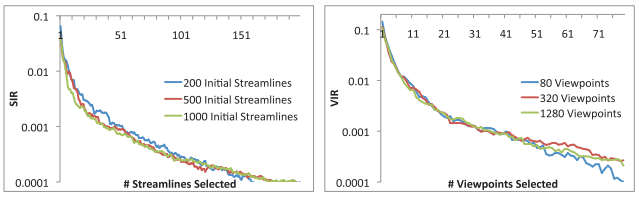


Fig. 4. Left: the SIR plot for the computer room data set. Right: the VIR plot for the five critical points data set.

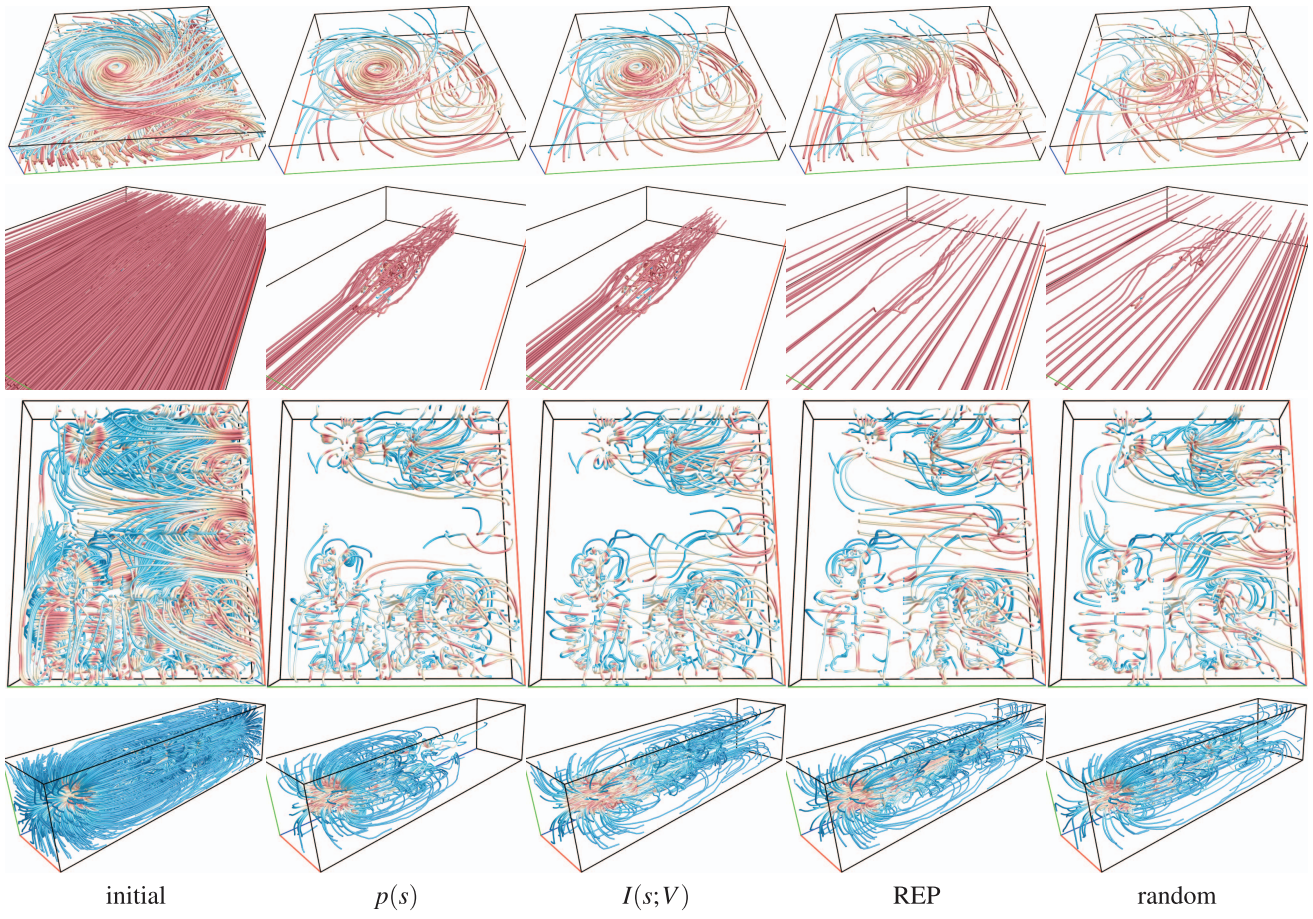


Fig. 5. Streamline selection. First row: the hurricane data set, 600 streamlines, 60 selected. Second row: the car flow data set, 600 streamlines, 40 selected. Third row: the computer room data set, 800 streamlines, 140 selected. Fourth row: the solar plume data set, 600 streamlines, 100 selected. Streamline selection based on $I(s;V)$ achieves the most consistent results among all four cases compared.

patterns of the velocity field are clearly visible. REP selection produces less accentuated circling patterns due to the need to cover the domain more evenly in order to best represent the entire field. Random selection leads to a similar result as REP selection, but the circling pattern in the right side of the image is much less obvious. For the car flow data set, both selections based on $p(s)$ and $I(s;V)$ are similar and are clear winners. REP and random selections produce undesirable results because interesting flow features are fairly localized in the domain. For the computer room data set, all four selections produce reasonable results. The two selections based on $p(s)$ and $I(s;V)$ reveal more swirling flow patterns. For the solar plume data set, all but the selection based on $p(s)$ produce acceptable results. The selection based on $p(s)$ misses the portion of the field that is less turbulent. Overall, we conclude that streamline selection based on $I(s;V)$ achieves the most consistent results.

4.3 Viewpoint Selection Results

In Fig. 6, we show the ranking of viewpoints based on $p(v)$ and $I(v;S)$ for the tornado data set, together with the corresponding best and worst viewpoints. As expected, the view sphere images indicate that neighboring viewpoints have similar rankings and the viewpoint ranking varies gradually over the view sphere. Although the two methods based on $p(v)$ and $I(v;S)$ give less similar results, the best and worst viewpoints convey the same meaning. That is, the best viewpoint corresponds to a view which clearly reveals the swirling pattern, while the worst viewpoint

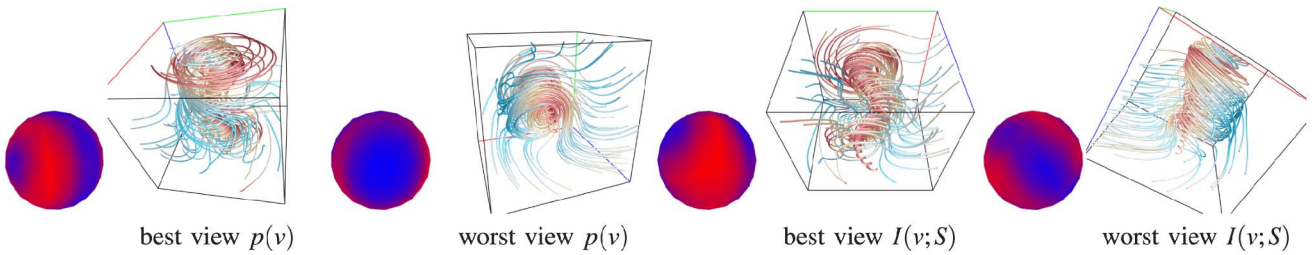


Fig. 6. Viewpoint ranking of the tornado data set. In each of the view sphere images, red to blue is for the best viewpoint to the worst viewpoint. Streamline rendering from the best viewpoint and the worst viewpoint is also shown. All cases use the best streamlines selected.

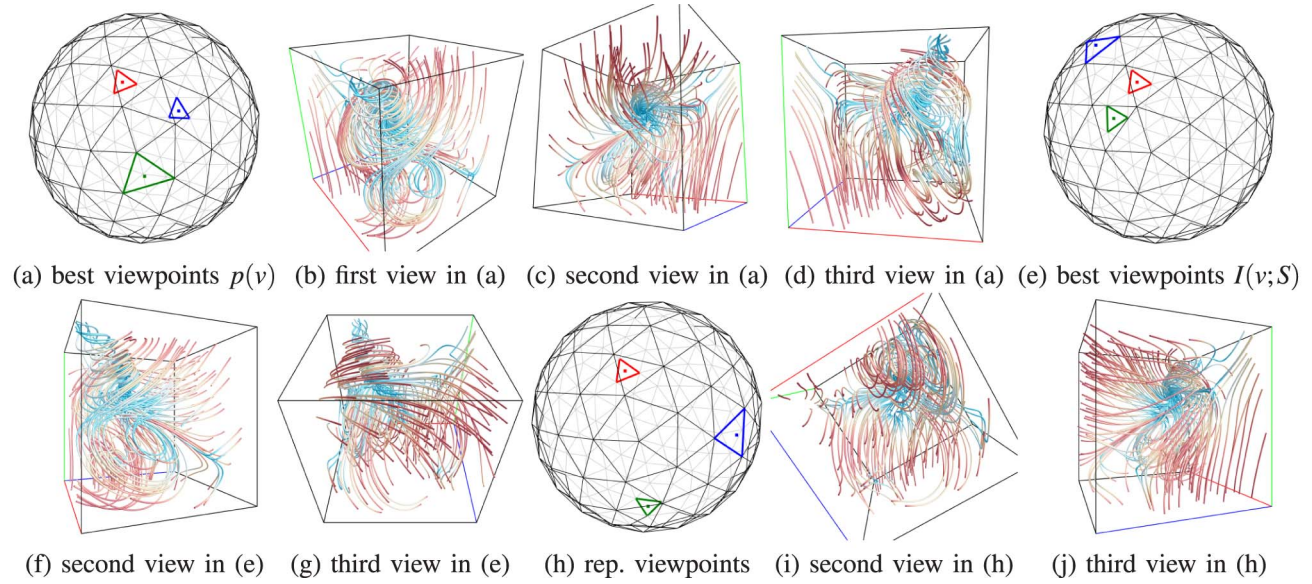


Fig. 7. Viewpoint selection of the five critical points data set. (a)-(d): best viewpoint selection based on $p(v)$. (e)-(g): best viewpoint selection based on $I(v;S)$, the first view in (e) is the same as (b). (h)-(j): representative viewpoint selection, the first view in (h) is the same as (b). In (a), (e), and (h), the first, second, and third best viewpoints are highlighted in red, green, and blue, respectively.

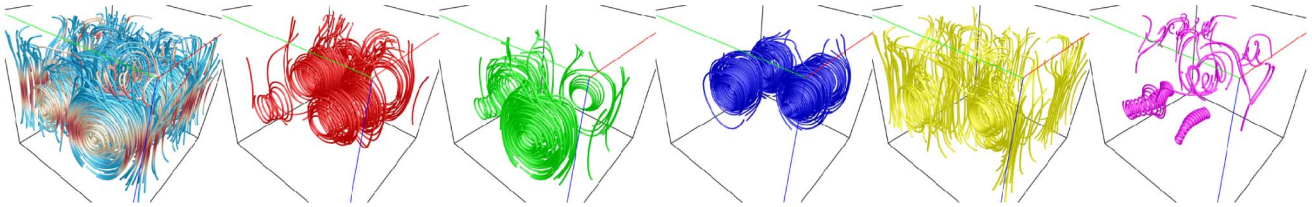


Fig. 8. Streamline clustering of the two swirls data set. Five clusters are produced from 500 streamlines. The appropriate number of clusters is suggested by the elbow criterion.

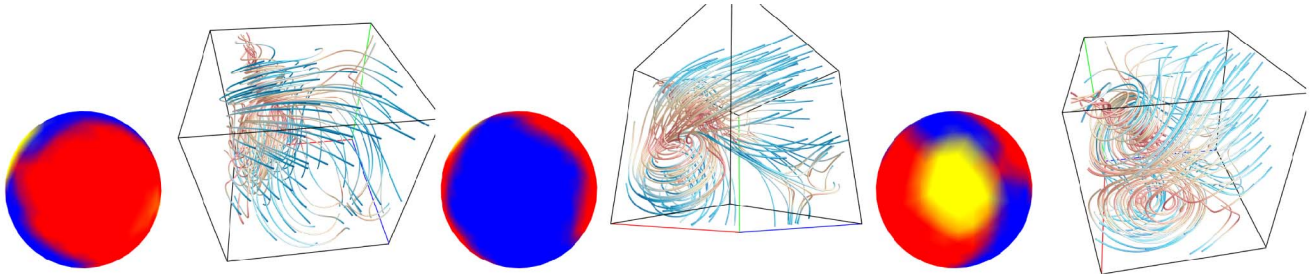


Fig. 9. Viewpoint partitioning of the five critical points data set. We denote the three partitions in red, blue, and yellow, respectively. Streamline rendering corresponding to the viewpoint centering at each of the view sphere partition images is also shown. In this example, 1,280 sample viewpoints are used for smooth view sphere rendering.

corresponds to a view where the swirling pattern is least clear. Our result is consistent with the viewpoint evaluation work reported in [13].

Fig. 7 shows the comparison among best viewpoint selections based on $p(v)$ and $I(v;S)$, and REP viewpoint selection. For each method, we marked where the first three choices are on the view sphere. Note that from the second choice onward, the selection based on $p(v)$ and $I(v;S)$ depends on the threshold d_v used. The three selection methods find the same viewpoint as their first choice. In general, the first choices based on $I(v;S)$ selection and REP selection are always the same since both select the one with the minimum $I(v;S)$ value first. But the first choice based on $p(v)$ selection may not always be the same as the other two methods. The three methods pick different viewpoints for their second and third choices. Nevertheless, we observe that all viewpoints selected are good as they reveal some new information about different critical regions that are not immediately visible from the previous selected viewpoints.

4.4 Streamline Clustering and Viewpoint Partitioning

Our solution for streamline clustering is solely based on the reduction of mutual information (9). The results with the two swirls data sets are shown in Fig. 8. The blue, yellow, and pink clusters are quite distinct which capture the internal swirls, external swirls, and outliers, respectively. The red and green clusters are in between the blue and yellow ones. We point out that even though our clustering results may not be as good as other clustering methods [20] based on spatial neighborhood and geometrical similarity, our method is very fast and produces quite meaningful streamline clusters. Fig. 9 shows the result of viewpoint partitioning. Three partitions are denoted in different colors

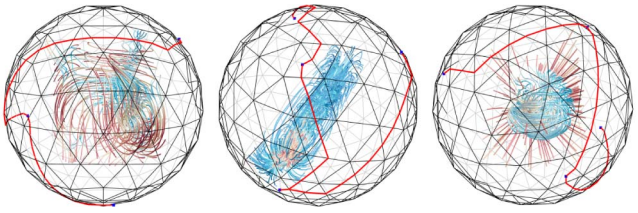


Fig. 10. Camera paths for the five critical points data set (left), the solar plume data set (middle), and the supernova data set (right).

in the view sphere partition images. We show a selected viewpoint from each partition to highlight the distinction among the three partitions.

4.5 Camera Path for Visual Exploration

Fig. 10 shows the camera paths we derived using the shortest path strategy. The shortest path is not based on geodesic distances, but according to the Jensen-Shannon divergences. Representative viewpoints were used to plan the camera path. Each path visits the representative viewpoints one by one. The resulting camera path is smooth because the shortest path between any two target viewpoints ensures that the change along the path is minimized. In other words, the viewpoints selected along the path are the most stable.

5 Comparison and Evaluation

5.1 Comparison with Other Methods

We compared our algorithm with other information theory based streamline placement and streamline selection algorithms. For streamline placement, we chose to implement a prototype of the entropy-guided streamline placement algorithm proposed by Xu et al. [33]. We implemented their template-based seeding technique based on the derived entropy field in conjunction with redundant streamline pruning. We used a moving window of 9^3 to compute the entropy centered at each voxel. Vector directions are quantized into 50 bins for histogram computation. If a voxel has a high entropy value, we placed the seeds at the voxel and also its eight corners of the 9^3 window (a total of nine seeds are placed). For streamline selection, we selected the view-dependent streamline visualization algorithm presented by Marchesin et al. [17]. We implemented their streamline evaluation based on angular and linear entropies and an approach similar to their occupancy buffer to account for streamline occlusion. We weighted angular and linear components equally by setting $\alpha = \beta = 0.5$. For the initial streamlines, we used the same pool of streamlines used in our method. For streamline pruning in [33] and occlusion consideration in [17], we used the mean of the closest point distances between two streamlines. The threshold was set as 5.0. This parameter determines the minimum distance between any two streamlines in the streamline pool that can be selected for visualization.

Fig. 11 shows the comparison of the results on six data sets. Our judgment is that our approach yields results that are as good as the ones produced by the other two methods for the first five data sets. In Sections 5.2 and 5.3, we show objective comparison and user study to justify this. For the computer room data set, our results perform better. In [33], in order to avoid large voids, the seeding method needs to consider the conditional entropy between the original field and the field reconstructed from currently displayed streamlines to decide the additional seeding locations. In [17], using the linear and angular entropies actually favors streamlines that have a constant angle change and a constant segment length along the points of the streamlines. This criterion, however, actually prefers well-behaved streamlines and misses those interesting streamlines that vary greatly in length and angle along their points. Our information channel approach works well, and is conceptually simple and easy to understand. It does not involve several steps as required in other methods for additional touch-up treatment (e.g., importance-based seed sampling in [33] and view-dependent streamline addition in [17]). Moreover, our approach is powerful as the same solution for streamline selection applies to viewpoint selection in the inverted information channel. This feature is not available in other methods.

5.2 Objective Comparison

To objectively evaluate the effectiveness of our methods with other streamline placement and selection algorithms, we reconstructed a vector field from the streamlines produced from each method and compared it with the original vector field. The rationale is that if the visualization successfully reveals most of the information in the field, then the field should be reconstructed from the visualization with minimal errors [4], [9], [14], [33]. The 3D reconstructed vector field can be obtained by linear interpolation via the Delaunay triangulation [4] or through the distance field [14], or using Gaussian smoothing and streamline diffusion [33]. We used the gradient vector flow (GVF) proposed by Xu and Prince [32] to reconstruct the vector field from the streamlines. As pointed out by Xu et al. [33], the GVF consisting of two terms—the smoothing term and the boundary term—allows us to properly cover the entire field in the sense that 1) the difference between the adjacent vectors in the reconstructed vector field is minimized (the smoothing term); and 2) the difference between the original vectors on the streamlines and the approximate vectors is minimized (the boundary term). We set $\mu = 0.1$ for the smoothing term in the equation. The minimization can be achieved iteratively using generalized diffusion equations. More details about the solution and the convergence of the diffusion can be found in [32], [33].

We reconstructed the initial vector at a voxel by considering a sphere with a radius of 0.5 (where 1.0 is the distance between two neighboring voxel along either x, y, or z-axes) and interpolated known vectors at points along the streamlines within the sphere. The interpolation was done using weighted average with the weight derived using a Gaussian function: the larger the distance between the voxel and the point along a streamline, the smaller the weight. From initial vectors reconstructed, we used the generalized diffusion equations to update the vectors in an iterative manner to reconstruct the vector field (we run 100 iterations in our test). After that, we calculated the angle difference between the original and reconstructed vectors for all voxels. Then, we computed the error between the original and reconstructed vector fields as the average angle

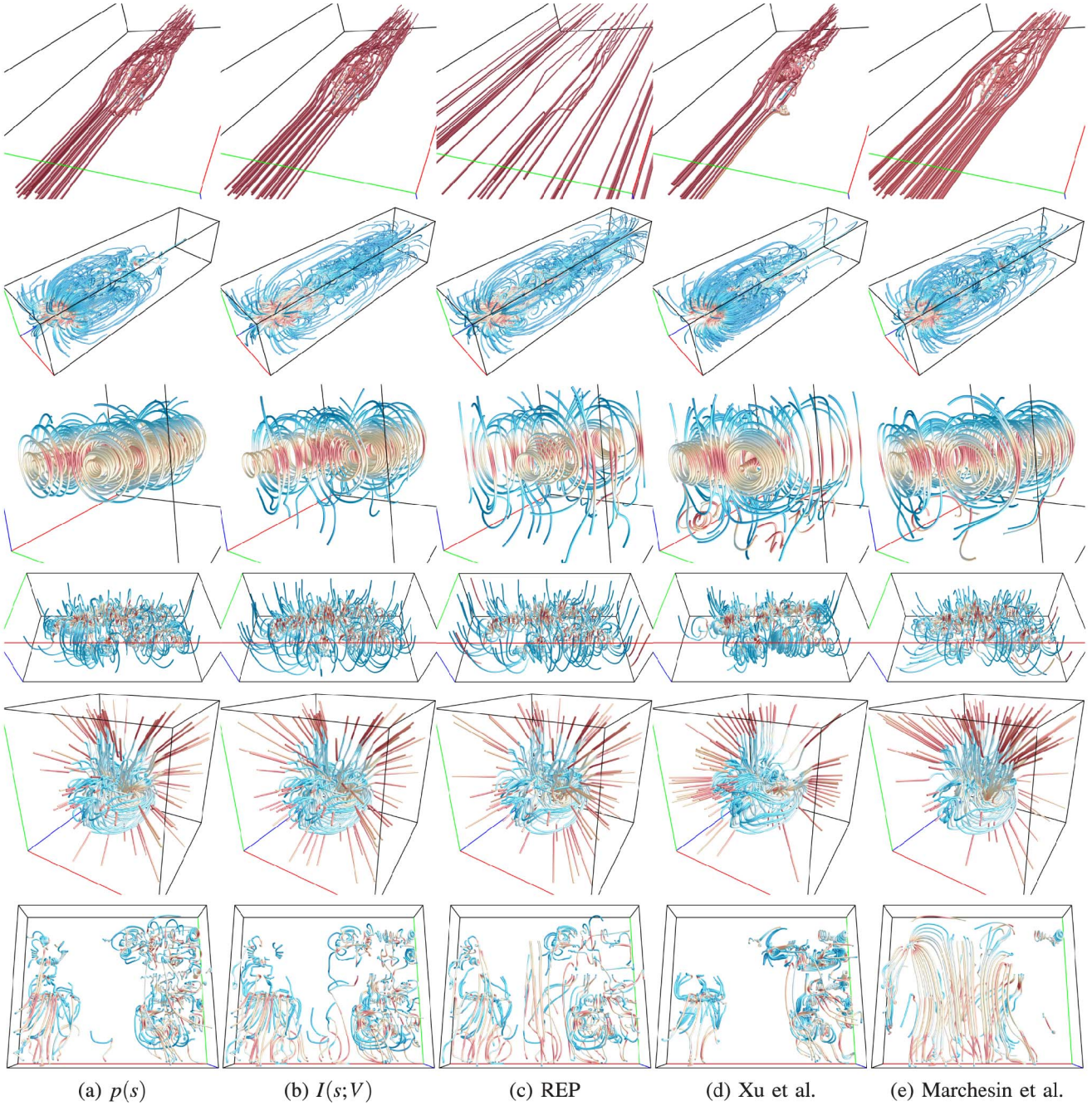


Fig. 11. Comparison of our approach based on $p(s)$, $I(s;V)$, and REP with Xu et al. and Marchesin et al. Top to bottom are the car flow, solar plume, two swirls, crayfish, supernova, and computer room data sets, respectively. All five methods show the same number of streamlines: 40, 100, 60, 70, 100, and 100 for the six data sets, respectively.

difference for all voxel pairs. We normalized the error to [0, 1] by dividing the average angle difference by π. In Table 3, we report the initial average boundary vector errors and final average reconstructed vector errors on the six data sets experimented. We see that although the errors are close, for all six data sets, the smallest error is always generated by one of our approaches ($p(s)$, $I(s;V)$, or REP).

5.3 User Study

We conducted a user study to evaluate the effectiveness of our approach based on $p(s)$, $I(s;V)$, and representative (REP). We also implemented Xu et al. [33] and Marchesin et al. [17] for comparison. We did not use the conditional entropy to introduce new streamlines as in [33], because this technique could also be applied to other streamline selection methods to fill in void regions. All methods are view independent, except for [17] where we selected the streamlines with respect to a good viewpoint and kept the set of streamlines selected for view-independent observation. The major goal of this study is to find out how effective our methods are compared to the existing ones and whether our methods work in the way they are designed to be.

The five methods were evaluated anonymously by a set of questions without timing followed by a feature identification task with timing. The users were 20 unpaid graduate students, including 12 students majoring in computer

TABLE 3
Error Comparison of Our Approach Based on $p(s)$, $I(s;V)$, and REP with Xu et al. and Marchesin et al. for the Six Data Sets As Shown in Fig. 11

	$p(s)$	$I(s;V)$	REP	Xu	Marchesin
initial average boundary vector error					
car flow	0.01098	0.01075	0.00245	0.00467	0.01955
solar plume	0.01859	0.01397	0.01368	0.01906	0.01698
two swirls	0.00790	0.00815	0.00905	0.00933	0.00764
crayfish	0.00715	0.01068	0.00829	0.00559	0.00767
supernova	0.00399	0.00454	0.00331	0.01203	0.00949
comp rm	0.01784	0.01688	0.01471	0.02551	0.01117
final average reconstructed vector error					
car flow	**0.01335**	0.01344	0.01696	0.05301	0.02107
solar plume	0.13851	**0.11917**	0.11967	0.17513	0.13027
two swirls	0.17485	0.17583	**0.16208**	0.17782	0.18441
crayfish	0.14950	**0.14786**	0.15204	0.17477	0.16078
supernova	0.10318	0.09300	**0.07752**	0.18122	0.16829
comp rm	**0.28903**	0.29236	0.30306	0.30334	0.32645

The smallest final average reconstructed vector errors are highlight in bold.

science (CS) and eight majoring in mechanical engineering, physics, and mathematics. All students majoring in computer science have knowledge in flow visualization and the students from other disciplines (non-CS) have flow field backgrounds.

Rating task design and procedure. We conducted a *within-subjects experiment* for this task using five data sets: the car flow, crayfish, solar plume, supernova, and two swirls. Two more data sets were used for initial practice: the computer room and tornado. The users were asked to rate the five methods for each data set in the following three aspects:

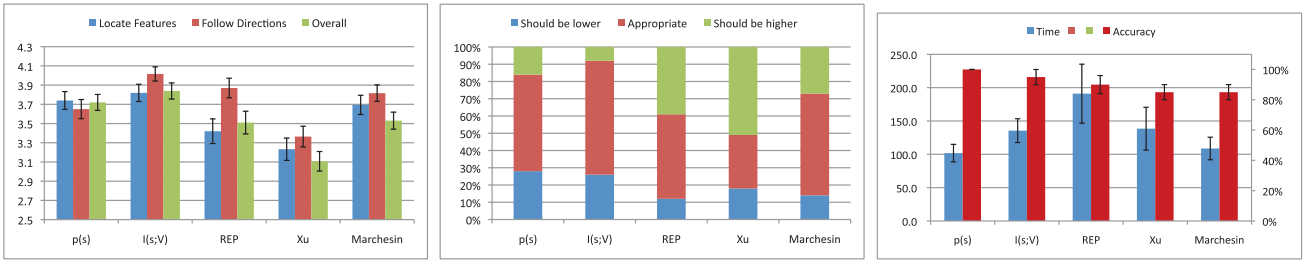
- *ease to locate flow features and identify their patterns*;
- *ease to follow flow directions*; and
- *overall effectiveness to help understand the flow field*.

For each method, each of these three aspects was rated by an integer between 1 and 5 with 1 being the worst and 5 the best. We collected the evaluation scores and the background information of the users (rank and major). This part of the evaluation was not timed and the users had enough time to complete the work.

This user study was conducted in a lab using four PCs with the same configuration. Each PC has a monitor with the resolution of $1{,}920 \times 1{,}080$ and the visualization result occupied an 800×800 viewport. The users could sit in any fashion they found comfortable. They started with a practice session to become familiar with our visualization system and the rating criteria. They could ask questions about the interface, interaction, and rating criteria, but not which visualization result is better. Evaluation activities began when the users felt ready and were performed one data set at a time. The users were not allowed to go back to a previous data set once they move forward. For each data set, the five methods were displayed anonymously in a random order, and the user could switch among the visualization results of all five methods for cross comparison. Specifically, we used a 5×5 Latin square for counterbalancing to rule out the learning effect. The order of methods for each of the five data sets was decided by a row of the Latin square. For the five methods to be evaluated, the users could rotate and zoom, but could not change the number of streamlines displayed. As a reference, the user could also display streamlines randomly selected from the streamline pool and rotate, zoom, and change the number of selected streamlines. This helps them answer questions such as if the predetermined streamline density for each of the five methods is appropriate or not. Random selection also avoids any bias in the users' subsequent rating of the five methods. Two sets of open questions were asked for the crayfish and solar plume data sets, which required the users to elaborate why the most and least helpful methods were selected and to comment on the limitation of each method. For each user, it took about 20 minutes for introduction, 40 minutes for the rating tasks, and 10 minutes for the timing and accuracy tasks.

Effectiveness evaluation. Using Kolmogorov-Smirnov test, we found out that most of our data do not pass the normality test. Therefore, instead of using ANOVA and Student's t-test, we mainly used Kruskal-Wallis nonparametric test (KW-test) and Mann-Whitney U-test for effectiveness evaluation. We used significant level $\alpha = 0.05$ in all tests and investigated the following four important issues.

First, we study the effectiveness of locating flow features. Since Fig. 12a shows that the average scores for our REP and Xu et al. are lower than the others, there is a significant difference for the five methods ($H(4) = 11.35$, $p = 0.023$). Further analysis shows that excluding Xu et al. yield an insignificant result, and pairwise U-tests suggest significant difference between REP and Xu et al. and other methods. Consequently, $p(s)$, $I(s;V)$, and Marchesin et al. are comparable to each other ($H(2) = 1.40, p = 0.50$) and better than our REP and Xu et al. in terms of locating features. Additionally, both the CS ($H(4) = 4.39, p = 0.36$) and non-CS ($H(4) = 8.44, p = 0.077$) groups show no



(a) (b) (c)

Fig. 12. (a) Mean values and standard errors of user rating for "ease to locate flow features," "ease to follow directions," and "overall effectiveness." (b) User rating for streamline density. (c) Mean values and standard errors of the completion time (in seconds) and accuracy for identifying five critical points.

difference among the five methods. For the non-CS group, the p-value is much higher $(H(3) = 2.22, p = 0.53)$ for the four methods excluding Xu et al. As mentioned earlier, we only implemented the entropy-based seeding part of Xu et al. Since a seed placed around the critical regions does not guarantee that the streamline will capture the features, this method might not show a clear flow pattern. Our REP is designed to focus on the general flow patterns, which makes it less effective to locate the features. However, our $p(s)$, $I(s;V)$, and Marchesin et al. are all based on streamline importance evaluation (albeit different criteria), which might explain why they were viewed similarly.

Second, we investigate the effectiveness of following flow directions. A significant effect is found for the five methods $(H(4) = 19.71, p = 0.0006)$, and there is no significant difference for the remaining four methods if Xu et al. is excluded $(H(3) = 7.12, p = 0.068)$. Moreover, the CS group $(H(3) = 8.67, p = 0.034)$ exhibits a significant difference while the non-CS group $(H(3) = 2.59, p = 0.459)$ does not. Our $I(s;V)$ has the highest average score of 4.02, and our REP and Marchesin et al. are very close, while that of Xu et al. is lower. Furthermore, since the p-values of U-test between Xu et al. against other methods are all small and the other four methods have no significant difference $(H(3) = 7.12, p = 0.068)$, our $p(s)$, $I(s;V)$, REP, and Marchesin et al. do not have a significant performance difference. In addition, a U-test was performed to compare locating flow features and following flow directions and the test result $(z = -2.62$, $p = 0.009)$ suggests that REP is better in following flow directions (the average score is 3.87) than in locating features (the average score is 3.42). Since our REP only considers the overall information revealed by the selected streamlines without evaluating each individual streamline, its selection result provides a good indication in terms of general flow directions but does not guarantee that the detailed features will be captured.

Third, as for overall effectiveness, our analysis indicates that there is a significant difference for the five methods $(H(4) = 19.65, p = 0.0006)$. If Xu et al. is excluded, no significant difference is found $(H(3) = 5.25, p = 0.155)$. Thus, our $p(s)$, $I(s;V)$ and REP, and Marchesin et al. do not have a significant performance difference. The CS group and non-CS group do not exhibit in-group difference.

Fourth, for density analysis, the five methods do exhibit a significant difference $(H(4) = 25.32, p = 0.00004)$. We divided the five methods into two groups, and found that there is no significant difference between our $p(s)$ and $I(s;V)(H(1) = 0.04, p = 0.831)$ and among the other three methods $(H(2) = 3.30, p = 0.192)$. Therefore, our $p(s)$ and $I(s;V)$ do not have a significant performance difference and are better than our REP, Xu et al. and Marchesin et al. as indicated by the averages shown in Fig. 12b.

User comments. For the crayfish data set, we asked the users which method was the most/least helpful to "locate and identify the features," and also requested them to comment on each method. Our $p(s)$ and $I(s;V)$ were each selected four times as the most helpful methods with similar reasons and typical comments were "*it provides general idea of surrounding streamlines, while putting more streamlines in the focus regions*" and "*it captures the characteristics of the feature regions with less occlusion*." One user selected $p(s)$ as the least helpful method because the feature regions were too dense, and three users selected $I(s;V)$ as the least helpful one because the feature regions could be a little denser. REP was selected by two users as the most helpful one to locate the features, yet by seven users as the least helpful one, although this method does not focus on the feature regions. Xu et al. was selected five times as the most helpful method, but it was also selected seven times as the least helpful one. Some users stated that it mainly placed streamlines in the interesting regions, which made the features stand out, while other users considered the feature regions to be too cluttered. Marchesin et al. was rated as the most and least helpful methods by five and two users, respectively, with similar reasons as our $I(s;V)$.

For the solar plume data set, we asked the users to select the most/least helpful method to "show the flow directions," and also requested them to comment on each method. Our REP was selected by 11 users as the most helpful one, mostly due to "*it fills the entire volume evenly without much occlusion.*" Our $I(s;V)$ was also considered as the most helpful one by six users for a similar reason. Note that these two methods are the only two that take the spacing and overall density into consideration. Our $p(s)$ was rated as the least helpful one by 15 users, since it left a large portion empty. Xu et al. was selected as the most helpful one by two users, since they believed a few streamlines were enough for the nonfeature regions. On the other hand, three users considered it as the least helpful one because some regions were too sparse. Marchesin et al. was neither selected as the most helpful one nor as the least helpful one.

Timing and accuracy task. We conducted a *between-subjects experiment* for this task using the five critical points data set. The ABC flow data set was used for initial practice. The users were asked to locate the five critical points in the task. Since it would be difficult to locate 3D points using mouse, the users selected only the 2D projection of each critical point by mouse clicking. For each critical point selected, an image was saved with a red circle marking the selected position. We then graded these images manually to derive the accuracy of user selection. Each user was required to complete the task with one method, and each method was performed by four users. We informed the users that accuracy is more important than timing, so that they would try their best to identify the correct locations of critical points. Moreover, the users could also switch to previous selection results and make modification if needed. The timer started when the data set was displayed, and stopped when the users clicked a button to finish.

Fig. 12c shows our $p(s)$ has the shortest average completion time and is closely followed by that of Marchesin et al. Our REP has the longest average completion time with the largest standard error, since the representatives do not necessarily capture the features. In terms of accuracy, our $p(s)$ is the highest (100 percent correct), while the other methods are close. In terms of the type of critical points, seven users missed one saddle, and one user missed one saddle and one spiral. This is probably because streamlines passing a saddle do not have high importance values compared to those passing spirals. We also observed that most users took a long time to find saddles. Among the five methods, our $p(s)$ appears to be the best one in terms of capturing saddles, since all users

located the two saddles successfully. Both timing and accuracy results indicate that our $p(s)$ is a good performer in terms of locating features. However, this is not verified by statistical testing, since our sample size is too small.

Summary. Our methods are designed to focus on different aspects: $p(s)$ selects more streamlines that show interesting patterns, REP mainly produces evenly spaced results, and $I(s;V)$ is somewhat in between. The major goal of this evaluation is to determine whether our methods are effective in the way they are designed to be. The averages of rating scores seem to support this to some degree. Our REP has high scores for following flow directions and low scores for locating flow features, and $p(s)$ has higher scores for locating features than following flow directions. In addition, our $I(s;V)$ has the highest average scores for all the three aspects. The timing and accuracy study shows a consistent result that $p(s)$ has highest accuracy with the least completion time, while REP takes the longest time to complete.

Hypothesis tests based on KW-test suggest that our methods do not have a significant performance difference as other existing methods. KW-test also indicates that our REP is more helpful to follow flow directions than to locate flow features, which confirms that it focuses on a different aspect compared to other methods. This is also an advantage, since we may benefit not only from the fact that our framework can provide different meaningful results, but also from the potential that we can develop a hybrid method based on this framework.

User comments indicate that streamline density is a very important factor. The methods that generate higher density around the feature regions and lead to a balanced overall density are highly appreciated. We also found that there was a connection between the three ratings and the density rating (Fig. 12b). Our $I(s;V)$, which has the highest average score, also has the highest percentage of being rated "appropriate." The users tended to rate the density of methods that are not satisfactory to be either "should be higher" or "should be lower," although some users also mentioned that the problems for those methods might be the locations of streamlines instead of the number of streamlines. The methods that miss certain kind of streamlines are more likely to be rated "should be higher," e.g., our REP might miss the features and Xu et al. might miss the surrounding streamlines. Finally, the methods that place many streamlines in the feature regions are often rated "should be lower."

6 Concluding Remarks

As the size and complexity of 3D flow field data continue to grow, automatic identifying good streamlines and viewpoints for effective flow visualization is a heated quest. Our information-theoretic framework provides an elegant solution to achieve both goals. Compared to other existing information theory guided approaches, the uniqueness of our approach lies in the formulation of streamline selection and viewpoint selection into a unified and rigorous framework using an information channel. Therefore, these two problems become symmetric and solving one problem immediately leads to the solution for the other. We demonstrate the effectiveness and robustness of our complete framework by showing both qualitative and quantitative results on a variety of flow data sets, and comparing, both objectively and subjectively, our approach to other information theory-based seed placement and streamline selection algorithms. To the best of our knowledge, this is the first work that applies dual information channels to solve flow visualization problems. We anticipate more applications and wider usage of information-theoretic approaches in flow field data analysis and visualization.

Acknowledgments

This research was supported in part by the US National Science Foundation (NSF) through grant IIS-1017935, and Michigan Technological University through a REF Research Seed grant. The authors would like to thank the anonymous reviewers for their insightful comments.

References

[1] A. Arbel and F.P. Ferrie, "Viewpoint Selection by Navigation through Entropy Maps," *Proc. IEEE Seventh Int'l Conf. Computer Vision,* pp. 248-254, 1999.

[2] U.D. Bordoloi and H.-W. Shen, "View Selection for Volume Rendering," *Proc. IEEE Visualization Conf.,* pp. 487-494, 2005.

[3] M. Chen and H. Jänicke, "An Information-Theoretic Framework for Visualization," *IEEE Trans. Visualization and Computer Graphics,* vol. 16, no. 6, pp. 1206-1215, Nov./Dec. 2010.

[4] Y. Chen, J.D. Cohen, and J.H. Krolik, "Similarity-Guided Streamline Placement with Error Evaluation," *IEEE Trans. Visualization and Computer Graphics,* vol. 13, no. 6, pp. 1448-1455, Nov./Dec. 2007.

[5] T.M. Cover and J.A. Thomas, *Elements of Information Theory,* second ed. Wiley-Interscience, 2006.

[6] M. Feixas, M. Sbert, and F. González, "A Unified Information-Theoretic Framework for Viewpoint Selection and Mesh Saliency," *ACM Trans. Applied Perception,* vol. 6, article 1, 2009.

[7] S. Fleishman, D. Cohen-Or, and D. Lischinski, "Automatic Camera Placement for Image-Based Modeling," *Computer Graphics Forum,* vol. 19, no. 2, pp. 101-110, 2000.

[8] L.-W. He, M.F. Cohen, and D.H. Salesin, "The Virtual Cinematographer: A Paradigm for Automatic Real-Time Camera Control and Directing," *Proc. ACM SIGGRAPH Conf.,* pp. 217-224, 1996.

[9] H. Jänicke, T. Weidner, D. Chung, R.S. Laramee, P. Townsend, and M. Chen, "Visual Reconstructability As a Quality Metric for Flow Visualization," *Computer Graphics Forum,* vol. 30, no. 3, pp. 781-790, 2011.

[10] G. Ji and H.-W. Shen, "Dynamic View Selection for Time-Varying Volumes," *IEEE Trans. Visualization and Computer Graphics,* vol. 12, no. 5, pp. 1109-1116, Sept./Oct. 2006.

[11] B. Jobard and W. Lefer, "Creating Evenly-Spaced Streamlines of Arbitrary Density," *Proc. Eurographics Workshop Visualization in Scientific Computing,* pp. 43-56, 1997.

[12] S. Kullback and R.A. Leibler, "On Information and Sufficiency," *Ann. of Math. Statistics,* vol. 22, no. 1, pp. 79-86, 1951.

[13] T.-Y. Lee, O. Mishchenko, H.-W. Shen, and R. Crawfis, "View Point Evaluation and Streamline Filtering for Flow Visualization," *Proc. IEEE Pacific Visualization Symp.,* pp. 83-90, 2011.

[14] L. Li, H.-H. Hsieh, and H.-W. Shen, "Illustrative Streamline Placement and Visualization," *Proc. IEEE Pacific Visualization Symp.,* pp. 79-86, 2008.

[15] L. Li and H.-W. Shen, "Image-Based Streamline Generation and Rendering," *IEEE Trans. Visualization and Computer Graphics,* vol. 13, no. 3, pp. 630-640, May/June 2007.

[16] A. Light and P.J. Bartlein, "The End of the Rainbow? Color Schemes for Improved Data Graphics," *EOS Trans. Am. Geophysical Union,* vol. 85, no. 40, p. 385, 2004.

[17] S. Marchesin, C.-K. Chen, C. Ho, and K.-L. Ma, "View-Dependent Streamlines for 3D Vector Fields," *IEEE Trans. Visualization and Computer Graphics,* vol. 16, no. 6, pp. 1578-1586, Nov./Dec. 2010.

[18] T. McLoughlin, R.S. Laramee, R. Peikert, F.H. Post, and M. Chen, "Over Two Decades of Integration-Based, Geometric Flow Visualization," *Computer Graphics Forum,* vol. 29, no. 6, pp. 1807-1829, 2010.
[19] A. Mebarki, P. Alliez, and O. Devillers, "Farthest Point Seeding for Efficient Placement of Streamlines," *Proc. IEEE Visualization Conf.,* pp. 479-486, 2005.
[20] B. Moberts, A. Vilanova, and J.J. van Wijk, "Evaluation of Fiber Clustering Methods for Diffusion Tensor Imaging," *Proc. IEEE Visualization Conf.,* pp. 65-72, 2005.
[21] O. Rosanwo, C. Petz, S. Prohaska, H.-C. Hege, and I. Hotz, "Dual Streamline Seeding," *Proc. IEEE Pacific Visualization Symp.,* pp. 9-16, 2009.
[22] M. Ruiz, I. Boada, M. Feixas, and M. Sbert, "Viewpoint Information Channel for Illustrative Volume Rendering," *Computers & Graphics,* vol. 34, no. 4, pp. 351-360, 2010.
[23] M. Schlemmer, I. Hotz, B. Hamann, F. Morr, and H. Hagen, "Priority Streamlines: A Context-Based Visualization of Flow Fields," *Proc. Eurographics/IEEE VGTC Symp. Visualization,* pp. 227-234, 2007.
[24] B. Spencer, R.S. Laramee, G. Chen, and E. Zhang, "Evenly Spaced Streamlines for Surfaces: An Image-Based Approach," *Computer Graphics Forum,* vol. 28, no. 6, pp. 1618-1631, 2009.
[25] S. Takahashi, I. Fujishiro, Y. Takeshima, and T. Nishita, "A Feature-Driven Approach to Locating Optimal Viewpoints for Volume Visualization," *Proc. IEEE Visualization Conf.,* pp. 495-502, 2005.
[26] G. Turk and D. Banks, "Image-Guided Streamline Placement," *Proc. ACM SIGGRAPH Conf.,* pp. 453-460, 1996.
[27] P.-P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich, "Viewpoint Selection Using Viewpoint Entropy," *Proc. Vision, Modeling, and Visualization Conf.,* pp. 273-280, 2001.
[28] V. Verma, D. Kao, and A. Pang, "A Flow-Guided Streamline Seeding Strategy," *Proc. IEEE Visualization Conf.,* pp. 163-190, 2000.
[29] I. Viola, M. Feixas, M. Sbert, and M.E. Gröller, "Importance-Driven Focus of Attention," *IEEE Trans. Visualization and Computer Graphics,* vol. 12, no. 5, pp. 933-940, Sept./Oct. 2006.
[30] C. Wang and H.-W. Shen, "Information Theory in Scientific Visualization," *Entropy,* vol. 13, no. 1, pp. 254-273, 2010.
[31] L. Wong, C. Dumont, and M. Abidi, "Next Best View System in a 3-D Modeling Task," *Proc. Int'l Symp. Computational Intelligence in Robotics and Automation,* pp. 306-311, 1999.
[32] C. Xu and J.L. Prince, "Gradient Vector Flow: A New External Force for Snakes," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* pp. 66-71, 1997.
[33] L. Xu, T.-Y. Lee, and H.-W. Shen, "An Information-Theoretic Framework for Flow Visualization," *IEEE Trans. Visualization and Computer Graphics,* vol. 16, no. 6, pp. 1216-1224, Nov./Dec. 2010.
[34] X. Ye, D. Kao, and A. Pang, "Strategy for Seeding 3D Streamlines," *Proc. IEEE Visualization Conf.,* pp. 471-478, 2005.

Jun Tao received the BS degree in software engineering from Sun Yat-sen University, China, in 2008, and the MS degree in computer science from Michigan Technological University in 2010. He is currently working toward the PhD degree in computer science at Michigan Technological University. His research interests include flow visualization, image resizing, and mesh editing. He is a student member of the IEEE.

Jun Ma received the BS degree in computer science from Xidian University, China, in 2006 and the MS degree in computer science from Michigan Technological University in 2009. He is currently working toward the PhD degree in computer science at Michigan Technological University. His research interests include flow visualization, large-scale data analysis and visualization, and mesh processing. He is a student member of the IEEE.

Chaoli Wang received the BE and ME degrees in computer science from Fuzhou University, China, in 1998 and 2001, respectively, and the PhD degree in computer and information science from The Ohio State University in 2006. He is an assistant professor of computer science at Michigan Technological University. His research focuses on large-scale data analysis and visualization, high-performance computing, and user interfaces and interaction. From 2007 to 2009, he was a postdoctoral researcher at the University of California, Davis. He has served on the program committees of the IEEE Visualization Conference and the IEEE Pacific Visualization Symposium. He is a member of the IEEE.

Ching-Kuang Shene received the PhD degree in computer science from The Johns Hopkins University in 1992. He is a professor of computer science at Michigan Technological University. His research interests include geometric modeling, mesh processing, software visualization, and computer science education. He is a member of the ACM, AMS, Eurographics, IEEE Computer Society, MAA, and SIAM.