

Coherent view-dependent streamline selection for importance-driven flow visualization

Jun Ma^a, Chaoli Wang^a, and Ching-Kuang Shene^a

^aDepartment of Computer Science, Michigan Technological University, Houghton, MI 49931

ABSTRACT

Streamline visualization can be formulated as the problem of streamline placement or streamline selection. In this paper, we present an importance-driven approach to view-dependent streamline selection that guarantees coherent streamline update when the view changes gradually. Given a large number of randomly or uniformly seeded and traced streamlines and sample viewpoints, our approach evaluates, for each streamline, the view-dependent importance by considering the amount of information shared by the 3D streamline and its 2D projection as well as how stereoscopic the streamline's shape is reflected under each viewpoint. We achieve coherent view-dependent streamline selection following a two-pass solution that considers i) the relationships between local viewpoints and the global streamline set selected in a view-independent manner and ii) the continuity between adjacent viewpoints. We demonstrate the effectiveness of our approach with several synthesized and simulated flow fields and compare our view-dependent streamline selection algorithm with a naïve algorithm that selects streamlines solely based on the information at the current viewpoint.

Keywords: flow visualization, importance-driven, view-dependent, streamline placement, streamline selection, coherent update

1. INTRODUCTION

In many fields of science and engineering, visualizing vector fields plays an essential role in visual interpretation and understanding of the underlying flow features and patterns. Well-known vector field visualization techniques include geometry-based methods such as particle tracing, texture-based methods such as line integral convolution (LIC),¹ spot noise,² and image-based flow visualization (IBFV).³ Visualization of streamlines and pathlines is still the most commonly used method because they are easy to compute and can be rendered at various resolutions with interactive rates.

A central issue for streamline visualization is *seed placement*. There exist several effective seeding strategies for 2D and 3D vector fields including image-guided⁴⁻⁶ and flow-guided^{7,8} algorithms. For 3D flow fields, seeding too many or too few streamlines is not able to reveal flow features and patterns well either because it easily leads to visual clutter in rendering (too many) or it conveys little information about the flow field (too few). Not only does the number of streamlines placed matter, their spatial relationships also influence our understanding of the flow field. Ideally, a streamline seed placement algorithm should retain important features in the vector field so that desired insights can be gained.

An alternative to seed placement is *streamline selection*. That is, we first place a large number of seeds either randomly or uniformly in the domain to produce a pool of streamlines. We then either automatically select representative or interesting streamlines from the pool^{9,10} or manually sketch a pattern to match similar streamlines for selective display.¹¹ Although the task is shifted from selecting *good seeds* to selecting *good streamlines*, the goal remains the same: we aim to produce a set of streamlines that capture flow features and patterns. In this paper, we present an importance-driven approach to interactive 3D streamline selection and visualization. Our goal is to perform selective streamline display which could not only reduce visual clutter

Further author information:(Send correspondence to Jun Ma)

Jun Ma: E-mail: junm@mtu.edu, Telephone: 1 906 370 7748

Chaoli Wang: E-mail: chaoliw@mtu.edu

Ching-Kuang Shene: E-mail: shene@mtu.edu

but also well characterize view-dependent vector field features. We also aim to maintain coherent streamlines updates between adjacent viewpoints.

Our approach is based on the observation that each streamline has a range of views that show its characteristics in the least ambiguous manner. We refer to these views as streamline *intrinsic views*. We leverage the concepts of information theory to derive the view-dependent importance of a streamline by computing the amount of information shared by the 3D streamline and its 2D projection under different viewpoints. Taking into account the shape characteristic of the streamline under different viewpoints as well, we obtain an importance measure that allows us to identify the intrinsic views of the streamline. Based on this importance measure, we present solutions for both view-independent and view-dependent streamline selection and visualization. For the view-independent case, our solution selects a set of overall important streamlines among all viewpoints and treats it as the globally optimal streamline set. For the view-dependent case, our algorithm dynamically selects important streamlines on the fly and is able to maintain coherent update of streamlines displayed by considering the relationships between local viewpoints and the global streamline set as well as the continuity between two adjacent viewpoints. We experiment our algorithm with several synthesized and simulated flow fields of different characteristics. The effectiveness of our algorithm is demonstrated through qualitative and quantitative results and comparison with a naïve view-dependent streamline selection algorithm that selects streamlines solely based on the information at the current viewpoint.

The main contributions of our paper are as follows: First, we propose a novel view-dependent streamline selection algorithm which guarantees coherent update of streamlines displayed when the viewpoints change gradually. Second, we also provide a view-independent streamline selection algorithm which selects a set of overall important streamlines among all viewpoints. Third, we identify the intrinsic views for streamlines based on their importance value which combines two view-dependent criteria: mutual information and shape characteristic.

2. RELATED WORK

One of the main focuses on flow visualization is seed placement. Jobard and Lefer⁵ presented an evenly-spaced seeding algorithm. They took a greedy strategy to place seeds in the neighborhood of previously placed streamlines. A distance threshold is used to explicitly control the density of streamlines. Liu et al.¹² proposed another evenly-spaced streamline placement algorithm for fast, high-quality and robust layout of flow lines. Their solution features double queues to prioritize topological seeding and adaptive distance control to minimize discontinuities. Verma et al.⁷ argued that the goal of streamline placement is to clearly reveal flow features such as critical points. Therefore, they proposed a flow-guided streamline seeding algorithm that explicitly detects critical points first and then applies different seeding templates to different types of critical points for feature highlighting. This approach was later extended to 3D streamline seeding by Ye et al.⁸ Mebarki et al.¹³ took a farthest seeding strategy and placed the seed successively at the place that is farthest away from all previously placed streamlines (i.e., the center of the biggest void region in the field). Schlemmer et al.¹⁴ presented another seeding solution that leverages a user-specified scalar function to control the streamline density. Streamlines are prioritized accordingly and those in the most important regions are drawn first to depict flow features. Li and Shen⁶ presented an image-based 3D streamline placement strategy that resolves visual clutter due to streamline projection by placing seeds in the 2D image space. Liu and Moorhead¹⁵ proposed an interactive view-driven evenly spaced streamline placement algorithm for 3D surface flows. Their algorithm integrates streamlines in 3D space while controlling the streamlines density in the 2D view space. Spencer et al.¹⁶ proposed an efficient algorithm to generate evenly spaced streamlines over surfaces by performing streamline integration in the image space. Li et al.¹⁷ proposed illustrative streamline placement to depict the flow patterns succinctly. This algorithm places a new streamline only when it represents flow characteristics that have not been shown by previously placed streamlines. Xu et al.¹⁸ presented an information-theoretic approach for streamline seeding. Their approach first uses seed templates to place streamlines near regions of high entropy values, then successively places more streamlines according to the conditional entropy between the original flow field and the field reconstructed from previously placed streamlines. Wu et al.¹⁹ presented a streamline placement algorithm that produces evenly spaced long streamlines while preserving topological features of a flow field. The flow field is decomposed into several topological regions and in each region seeds are placed along a seeding path.

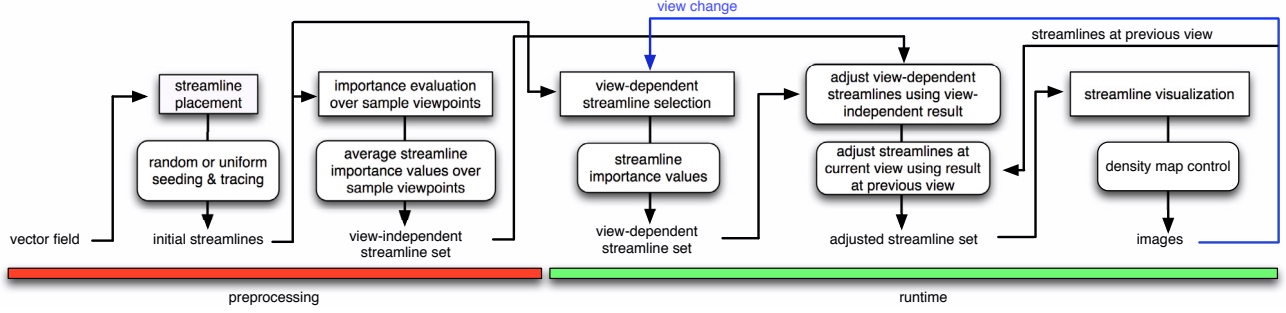


Figure 1. The overview of our coherent importance-driven streamline selection and visualization. Our approach consists of a view-independent solution that selects best streamlines considering all sample viewpoints and a view-dependent solution that dynamically selects important streamlines on the fly.

An alternative to seed placement is to either uniformly or randomly place seeds in the field and then adjust the resulting streamlines or select a subset of streamlines for informative visualization. Turk and Banks⁴ proposed to use an energy function to guide streamline placement. Their algorithm starts with uniformly or randomly seeded streamlines and then follows an iterative process to improve the visualization by taking several primitive streamline operations (move, insert, delete, lengthen, shorten and combine) and gradually reducing the energy. The energy is defined as the difference between a low-pass filtered version of the streamline image and the desired visual density. Chen et al.⁹ selected streamlines from randomly-seeded candidates based on their distance, shape and orientation to accentuate regions of interest. Their similarity-guided approach produces streamlines that accentuate regions of interest without explicit feature detection and extraction. Marchesin et al.¹⁰ presented a view-dependent solution for streamline selection. Starting from a pool of randomly seeded streamlines, they first removed streamlines that have low 3D entropies or have a large overlap with other streamlines given the view, then added new streamlines to cover empty areas to provide more context information. Lee et al.²⁰ presented a view-dependent algorithm that minimizes the occlusion and reveals important flow features for 3D flow fields. They utilized Shannon’s entropy as a measure of vector complexity and derived an entropy field from the input vector field. Using the maximal entropy projection (MEP) framebuffer that stores maximal entropy values as well as the corresponding depth values for a given viewpoint, they developed a view-dependent algorithm to evaluate and choose streamlines guided by the MEP framebuffer.

Our work falls into the category of streamline selection. Unlike the work by Marchesin et al.¹⁰ which only evaluates 3D linear and angular streamline entropies, we evaluate the information loss when 3D streamlines are projected to the 2D image plane. Our strategy is similar to the work of Furuya and Itoh.²¹ Instead of only considering a streamline’s projected length using entropy,²¹ we take into account both direction and magnitude of the vectors along the 3D streamline and its 2D projection using mutual information. Moreover, we also incorporate the streamline’s shape characteristic to obtain our streamline importance measure. Since our solution takes into account view changes in streamline importance evaluation and we carefully select streamlines under each viewpoint by considering the overlap of streamlines between the current and previous viewpoints, we are able to produce *coherent* transition between viewpoints as the user rotates the flow field, which is not achieved in the work of Marchesin et al.¹⁰ and Lee et al.²⁰ Both Xu et al.¹⁸ and our method uses information theory for importance evaluation. While the evaluation of information content in Xu et al.¹⁸ is on the flow field, we evaluate the information content on the integrated streamlines.

3. ALGORITHM OVERVIEW

We sketch the main steps of our algorithm in Figure 1. Given an input 3D vector field, we first produce a large number of randomly or uniformly seeded and traced streamlines over the field. To favor long streamlines that better reveal the continuity of the flow field, we integrate each streamline as long as possible until it leaves the domain or the velocity becomes zero. This step of streamline placement can be stopped until a target number of streamlines has been generated or the streamline pool produced is dense enough (e.g., every voxel has been passed through by at least one streamline). Then we evaluate the importance for each streamline and order them into a

priority queue for every single sample viewpoint. More important streamlines are those whose 3D information is high and in the meanwhile, whose 2D projections correspond to their respective intrinsic views that reveal most of their 3D information. In other words, more important streamlines are those whose 2D projections are able to present more 3D shape information of the underlying flow field at the current viewpoint.

With the streamlines prioritized, we are able to perform view-independent or view-dependent streamline selection and visualization. For the view-independent scenario, our algorithm selects best streamlines considering all sample viewpoints. Overall, the selected streamlines are important from different viewpoints. For the view-dependent scenario, our algorithm dynamically selects important streamlines. We leverage a 2D *density map* and its effective area to control the density of streamlines displayed in the image plane. Since the view-dependent selection is based on both the global streamline information and the continuity between local adjacent viewpoints, our algorithm is able to maintain coherent update of streamline displayed when the view changes gradually.

4. STREAMLINE IMPORTANCE EVALUATION

Given a viewpoint, we evaluate the importance of each streamline by considering three criteria. In the following we describe these three criteria and present their combination in the form of a joint importance measure.

- *How much information the streamline contains in 3D.* A 3D streamline has more information if its entropy is high, i.e., it shows a more even distribution of vectors associated with the streamline’s points in terms of direction and magnitude. (The streamline’s points are the equidistant positions along the streamline with respect to an arc-length parameterization.) Therefore, a streamline traced over a turbulent flow region is likely to contain more information than a streamline traced over a laminar flow region.
- *How much information about the 3D streamline is revealed in the 2D projection.* Due to the 3D to 2D projection, information loss is inevitable. Some viewpoints can preserve the characteristics of the 3D streamline better than other viewpoints. A good projection is the one that shows the streamline in the least ambiguous way. That is, the corresponding viewpoint is an intrinsic view.
- *How stereoscopic the shape of the streamline is reflected under the given viewpoint.* We call this criterion the *shape characteristic* of the streamline. Since the shape of the streamline’s 2D projection varies dramatically under different viewpoints, a good shape characteristic should reveal the 3D pattern of the streamline as much as possible. For example, the 2D projection of a spiral that allows us to infer both its curvature and torsion information is preferred than those revealing only one of them.

4.1 Streamline Entropy

We evaluate the streamline importance based on its entropy value. For each streamline, we employ a sliding window technique along each point and evaluate its entropy within the local window region. To better evaluate the entropy, we assume that each streamline has been reparameterized by the arc length and we use newly created sample points along the reparameterized streamline. The entropy of a discrete random variable X with alphabet $\mathcal{X}$ takes the following equation

$$H(X) = - \sum_x p(x) \log p(x), \quad (1)$$

where $p(x)$ is the probability mass function, $x \in \mathcal{X}$. To compute $p(x)$ for every point on the streamline, we interpolate its vector from the original vector field and evaluate the vector variation within the window centered at the point. We consider both *direction* and *magnitude* of the vectors. For vector direction, we decompose a unit sphere into a certain number of patches of equal area with small diameter following the algorithm proposed by Leopardi.²² (We also use this method to partition a view sphere into sample viewpoints, refer to Figure 2 (a).) All vectors falling into the same patch will be quantized into the same bin of vector direction. For vector magnitude, we quantize it into a certain number of levels. A 2D histogram consisting of vector direction and magnitude is created for each sliding window. $p(x)$ is computed as the normalized bin count of the 2D histogram.

4.2 Mutual Information

We quantify the view-dependent streamline importance by utilizing two view-sensitive terms. The *mutual information* between the 3D streamline X and its 2D projection Y is one of them.²³ Given two discrete random variables X and Y , the mutual information is defined as

$$I(X; Y) = \sum_x \sum_y p(x, y) \log \frac{p(x, y)}{p(x)p(y)}, \quad (2)$$

where $p(x)$ and $p(y)$ are the marginal probabilities of X and Y respectively, and $p(x, y)$ is their joint probability. If $I(X; Y)$ is high, then the 3D streamline has a high entropy and its 2D projection preserves much of the 3D information. Conversely, if the 3D streamline has a low entropy, or its 2D projection loses much of the 3D information (even though the 3D streamline has a high entropy), then $I(X; Y)$ is low. Therefore, we favor streamlines that have high information content while their 2D projections reveal their characteristics well.

To compute the marginal probability $p(x)$, we use a similar solution presented in entropy evaluation and consider both vector direction and magnitude for the points along each streamline. The only difference is that we do not use the sliding window here and $p(x)$ is taken over the entire streamline. To compute the marginal probability $p(y)$, we use the projections of all vectors along all points of the streamline. To quantize projected 2D vector directions, we evenly partition a unit circle into a certain number of angle ranges. All vectors falling into the same range will be quantized into the same bin of vector direction. For projected vector magnitude, we quantize it into a certain number of levels as well. To compute the joint probability $p(x, y)$, we create a 2D joint histogram where the two axes are for all vector direction and magnitude combinations for variables X (3D streamline) and Y (streamline 2D projection), respectively. In the joint histogram, the normalized bin count corresponds to $p(x, y)$.

4.3 Shape Characteristic

The *shape characteristic* is the other term for computing view-dependent streamline importance. Since the shape of a streamline X varies along with the changing of viewpoints, we use this term to indicate how stereoscopic the shape of X is under a specific viewpoint v . We also define the *simplification* of a streamline by uniformly downsampling the original full-scale streamline. Since the number of points along a streamline is usually fairly large (e.g., in the order of hundreds or thousands), we utilize the streamline simplification to compute its shape characteristic to reduce computation complexity. Let us denote the simplification of streamline X as $\tilde{X} = \{\tilde{p}_1, \tilde{p}_2, \dots, \tilde{p}_k\}$, the viewing vector as $\vec{v}$, and the angle between $\vec{v}$ and $\overrightarrow{\tilde{p}_i \tilde{p}_{i+1}}$ as θ . Then we can define the shape characteristic of $\tilde{p}_i \tilde{p}_{i+1}$ as

$$\alpha(\tilde{p}_i \tilde{p}_{i+1}, v) = \alpha_{\min} + (1.0 - \alpha_{\min}) \left(1.0 - \frac{|\pi/4 - \theta'|}{\pi/4} \right), \quad (3)$$

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

$$\theta' = \begin{cases} \pi - \theta, & \theta > \pi/2 \\ \theta, & \theta \leq \pi/2 \end{cases} \quad (4)$$

It is obvious that $\alpha(\tilde{p}_i \tilde{p}_{i+1}, v)$ gets its maximum (minimum) value of 1.0 ($\alpha_{\min}$) when $\vec{v}$ and $\overrightarrow{\tilde{p}_i \tilde{p}_{i+1}}$ form a 45° or 135° (0° , 90° or 180°) angle. Now we define the shape characteristic of streamline simplification $\tilde{X}$ as

$$\alpha(\tilde{X}, v) = \frac{\sum_{i=1}^{k-1} \alpha(\tilde{p}_i \tilde{p}_{i+1}, v) \|\tilde{p}_i \tilde{p}_{i+1}\|}{\sum_{i=1}^{k-1} \|\tilde{p}_i \tilde{p}_{i+1}\|}. \quad (5)$$

Intuitively, large shape characteristic value indicates a more “stereoscopic” shape of the streamline under a specific viewpoint which can reveal more streamline shape patterns. To gain a more comprehensive understanding of how the shape characteristic works, please refer to the work of Tao et al.²³ where they made a side-by-side comparison between two viewpoint selection results with and without considering shape characteristic.

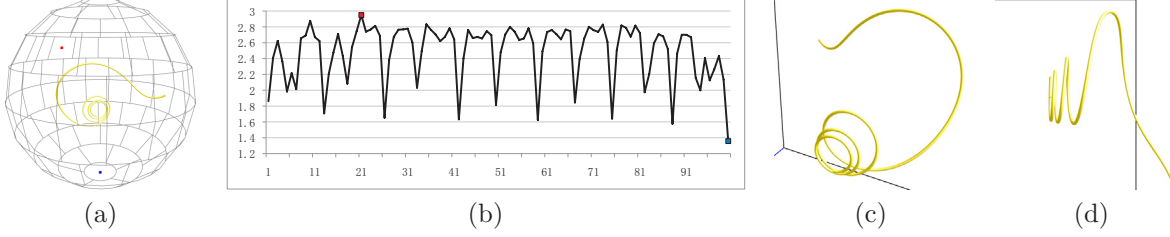


Figure 2. (a) a streamline and all 100 sample views based on the spherical partition. (b) the variation of the streamline’s importance value as the view changes gradually from the north pole to the south pole. (c) the best view of the streamline. (d) the worst view of the streamline. In (a) and (b), the best and worst views are marked in red and blue, respectively.

4.4 View-dependent Streamline Importance

With mutual information and shape characteristic defined above, we obtain the view-dependent importance $M(X, v)$ of streamline X under viewpoint v as

$$M(X, v) = \frac{\alpha(\tilde{X}, v)I(X; X_v)}{\sum_{X \in \mathbf{X}} \alpha(\tilde{X}, v)I(X; X_v)}, \quad (6)$$

where X_v denotes the 2D projection of X under v and $\mathbf{X}$ denotes the streamline pool. In Figure 2, we show an example streamline and the variation of its importance value with all sample viewpoints. Two views corresponding to the best and worst cases are also given. As we can see, the best case corresponds to an intrinsic view having an almost 45° angle with the streamline where much of the 3D streamline characteristics (curvature and torsion) is revealed in the 2D projection. The worst case hides most of the 3D information and displays the streamline in the least certain way.

5. STREAMLINE SELECTION

Our view-independent streamline selection serves as the first step for view-dependent streamline selection. Streamlines selected in the view-independent manner will be used to adjust the view-dependent selection results so that the selected streamlines under each viewpoint always inherit the global “flavor”. In the following, we introduce our view-independent streamline selection algorithm first, followed by the view-dependent algorithm.

5.1 View-independent Selection Algorithm

After importance evaluation, all streamlines are sorted in a priority queue based on their importance values (Equation 6). As the view changes, the priority queue gets updated as well. To select streamlines for visualization in a view-independent manner, we go over all sample viewpoints and compute the accumulated importance value for each streamline. The final priority queue is derived based on the *average* importance value of each streamline. We add a distance check to avoid selecting redundant streamlines even though their accumulated importance values are high. To achieve this, we compute the *minimum* distance of the current streamline under consideration to all streamlines that have been selected. The distance between two streamlines is defined as the Euclidean distance between their corresponding importance values under all sample viewpoints. If this minimum distance is larger than a given distance threshold δ_s , then the current streamline is selected. Otherwise, it is discarded. The creation of streamline priority queue based on all sample viewpoints can be done during the preprocessing (refer to Figure 1). At runtime, we simply select a certain number of top-ranked streamlines that pass the distance check for the viewing.

5.2 View-dependent Selection Algorithm

Our view-dependent selection algorithm consists of five steps. First, we obtain a global streamline set S from the view-independent selection algorithm. Second, all the streamlines are sorted based on their importance values under a given viewpoint. Third, we combine the top-ranked streamlines of the first and second steps and put them into a streamline set S_i . Forth, in order to consider the coherence between current and previous viewpoints, we create a new streamline set which is the combination of streamline sets S_i under the current

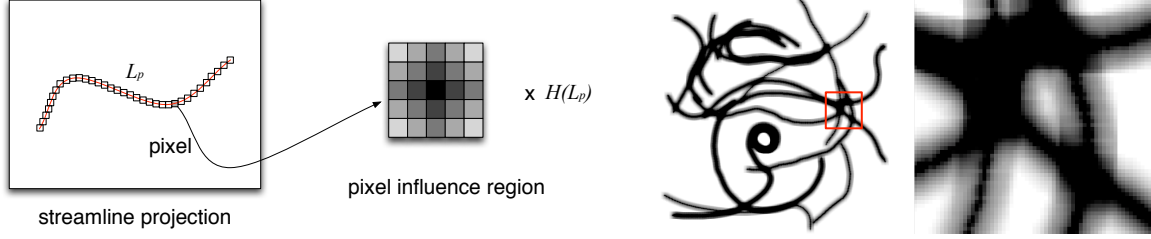


Figure 3. Left: an example of the 5×5 influence region for each pixel along the streamline projection. The weight assigned to each pixel in the influence region is inversely proportional to its Manhattan distance to the central pixel. The weighted average mask is used to compute the importance of the streamline projection to the density map. Right: the density map of a hurricane data set and a zoom-in to its middle-right region after several streamlines are selected.

viewpoint and S_{i-1} under the previous viewpoint. Finally, we dequeue each streamline in the new set from step four and leverage a density map to determine whether the streamline should be displayed in the final image or not. By adjusting parameters of the density map, users can easily control streamline density in the display. In the following, we describe our view-dependent streamline selection algorithm in detail.

- Step 1: Sort the initial N streamlines in the pool based on the view-independent selection algorithm and obtain a global set S by choosing a certain number of top-ranked streamlines. This global streamline set is the initial reference for view-dependent streamline selection. The number of streamlines selected in S is chosen large enough for the rest of steps. In this paper, since the number of finally selected streamlines is usually $1/4$ of the total streamline number N , we double this value and set the size of S to $N/2$.
- Step 2: Given a viewpoint v_i , update the priority queue for all streamlines according to their view-dependent importance values in the descending order. Choose the first $N/2$ streamlines as the initial streamline set S_i under v_i . The reason for us to choose $N/2$ streamlines is to ensure that S and S_i share the same size.
- Step 3: Compute the overlap between S and S_i and keep the common streamlines in S_i . Then for the rest of streamlines, remove a certain number of streamlines from S_i based on the mean of the closest point (MCP) distances.²⁴ Specifically, we compute the MCP distance for each streamline s_i in S_i to all streamlines in S and define the distance from s_i to S as the *maximum* MCP distance. In order to maintain the global streamline information under the current viewpoint, we always prefer the streamlines in S_i with small distance values since they are close to S . By contrast, the streamlines in S_i with large distances to S will be discarded. Next, the same number of streamlines from S will be added to S_i based on their view-independent importance. That is, we traverse each streamline in S according to the decreasing importance value and check whether the streamline is shared by S and S_i . If not, we add it into S_i . We keep doing this until we reach the required adding number. Now the newly selected streamlines in S_i contain both view-dependent and view-independent characteristics. The adding or removing number is user-defined. A larger value indicates that the new set S_i is more similar to S while a smaller value means that S_i preserves more of the local information. We test several candidate values and find that $1/5$ of the size of S is appropriate which well balances global and local streamline characteristics in S .
- Step 4: In order to maintain a coherent streamline update between two adjacent viewpoints, we compare S_i under viewpoint v_i with S_{i-1} under its previous viewpoint v_{i-1} . This procedure is almost the same as Step 3. First of all, we compute the streamline overlap between S_i and S_{i-1} and keep the common streamlines in S_i . Then for the rest of streamlines, we remove a certain number of streamlines from S_i based on their MCP distances to S_{i-1} . Next, we add the same number of streamlines from S_{i-1} to S_i according to their importance values. Now we obtain a new S_i which considers the coherence of the current and previous viewpoints.
- Step 5: During this step, we compute the final streamline set for view-dependent display. We propose to use a *density map* to keep track of which regions in the rendered image have been covered by streamline projections and which regions have not. We define the effective area of a density map under a specific

viewpoint as the projection area of the data set’s bounding box. This would allow us not only to control the streamline number based on the effective area but also to balance streamline selection by reducing visual clutter while revealing interesting flow features and patterns. Note that we do not require the density map to have the same resolution as the final image. A low resolution density map can speed up its update and the subsequent streamline selection process. We assume that each streamline projection L_p has its own *influence region* on the density map. For simplicity, we use a $m \times m$ local mask for each pixel along the projection where the actual mask size is proportional to the final image size. Figure 3 shows an example with a 5×5 mask. This step includes the following sub-steps:

- Sub-step 1: Initialize the density map with an equal density value for all pixels. In the following sub-steps, a streamline L will gain some density value from the pixels it passes through and we define the total density value gained by L as its importance value. Compute the overall effective density value as the summation of all pixels’ density values inside the effective area.
- Sub-step 2: Dequeue the streamline L with the highest priority value from S_i and compute its 2D projection’s entropy value $H(L_p)$.
- Sub-step 3: Maintain a pixel list that records each pixel along L_p in the image plane. We also define the influence region of the pixel in the list as a $m \times m$ local square centered at that pixel. Then for each pixel in L_p , use a weighted average mask (the influence region) multiplied by $H(L_p)$ to accumulate the importance value gained by L from the density map (see Figure 3). Normally, the weight for the central pixel in the mask is set to 1.0. The importance value gained by L from one pixel is bounded above by a maximum importance threshold δ_i .
- Sub-step 4: Subtract the importance values in the pixel list from the density map. Each pixel’s density value is bounded below by zero. The summation of total density value loss is defined as the final importance value gained by streamline L from the density map. If this value is above a given density threshold δ_d (i.e., L gains enough importance from the density map), L is selected. Otherwise, L is discarded.
- Sub-step 5: Go to Sub-step 2 until the total importance gained by all selected streamlines is above a given threshold. In this paper, we set this threshold to be $2/3$ of the overall effective density value. The user can adjust this value to control the density of streamlines displayed.

With this view-dependent streamline selection algorithm outlined above, the final streamlines set S_i is determined not only by the local importance of streamlines but also by their relationships with the global streamline set as well as the streamline set under the previous viewpoint. The motivation for using the initial density map with an equal value is to favor evenly-placed streamlines across the image instead of being cluttered in any one location. This is similar to the image-guided streamline placement algorithm introduced by Turk and Banks.⁴ We assign a larger importance value to a streamline with a higher 2D projection entropy. Such a streamline, if selected, would be less likely to be occluded by other streamlines. Setting a maximum importance threshold for each influenced pixel is to ensure that heavily self-occluded streamlines would not get an excessively high importance value. Furthermore, the use of effective area helps us balance the number of streamlines selected under different viewpoints based on the projection of the volume’s bounding box.

6. RESULTS AND DISCUSSION

We experimented our approach with eight flow data sets which are listed in Table 1. The five critical points data set⁸ is a synthesized flow field consisting of two spirals, two saddles and one source. The two swirls data set is from a simulation of swirls resulting from wake vortices. The tornado data set is from a simulation of a tornado event. The supernova data set is from a simulation of the explosion of stars. The crayfish data set is from a simulation of the heat flow around a cooking crayfish. The solar plume data set is from a simulation of down-flowing solar plumes for studying the heat, momentum and magnetic field of the sun. The computer room data set is from a simulation of air flows inside a computer room. Finally, the hurricane data set is from a simulation of Hurricane Isabel, a strong hurricane in the west Atlantic region in September 2003. In the following, we present the machine configuration and timing results, followed by streamline selection results using

data set	dimension	threshold δ_s	initial # lines	average # points per line	initial # views	selected # lines	importance evaluation time	line selection time
five critical pts	$51 \times 51 \times 51$	40.0	500	110	360	250	11.27s	0.006s
two swirls	$64 \times 64 \times 64$	40.0	500	157	360	250	11.52s	0.008s
tornado	$64 \times 64 \times 64$	40.0	500	295	360	250	12.05s	0.008s
supernova	$100 \times 100 \times 100$	50.0	500	184	360	250	12.20s	0.008s
crayfish	$322 \times 162 \times 119$	50.0	800	209	360	400	19.12s	0.012s
solar plume	$126 \times 126 \times 512$	50.0	600	100	360	300	13.90s	0.007s
computer room	$417 \times 345 \times 60$	60.0	800	173	360	400	18.60s	0.010s
hurricane	$500 \times 500 \times 100$	60.0	600	341	360	300	14.18s	0.010s

Table 1. The threshold and timing results of eight flow data sets for view-independent streamline selection. The view-independent timing is the total time to handle 360 sample viewpoints.

data set	density map				timing	
	dimension	mask	threshold δ_i	threshold δ_d	importance evaluation	line selection
five critical pts	400×400	3×3	1.0	80.0	0.031s	0.340s
two swirls	400×400	3×3	1.0	80.0	0.032s	0.427s
tornado	400×400	3×3	1.0	100.0	0.033s	0.514s
supernova	600×600	7×7	1.0	120.0	0.034s	0.485s
crayfish	600×600	7×7	1.0	250.0	0.053s	0.897s
solar plume	800×800	15×15	1.0	120.0	0.039s	0.982s
computer room	800×800	15×15	1.0	200.0	0.052s	0.857s
hurricane	800×800	15×15	1.0	200.0	0.039s	1.215s

Table 2. The thresholds and timing results of eight data sets for view-dependent streamline selection. The view-dependent timing is the average time to handle one of the 360 sample viewpoints.

our view-independent and view-dependent algorithms. We refer readers to the accompanying video for the best evaluation of our approach.

6.1 Configuration and Timing

We used a hybrid CPU-GPU solution in our computation with the following hardware configuration: Intel Core i7 quad-core CPU running at 3.20GHz, 24GB main memory and an nVidia GeForce GTX 580 graphics card. We implemented streamline importance evaluation in the GPU using CUDA. For the view-independent case, the global streamline set was computed using the GPU. For the view-dependent case, all computations were done by using the CPU including streamline selection due to the sequential nature of streamline selection with the use of the density map. The timing results are reported in Tables 1 and 2. As we can see, it took up to 20 seconds for view-independent streamline selection, evaluating all 360 sample viewpoints. However, since this step can be done during the preprocess stage and the timing for view-dependent streamline selection took only less than one second, our implementation can deliver a performance for supporting realtime interaction.

6.2 View-independent Selection Results

Figure 4 shows the results of view-independent streamline selection with the supernova data set. A total of 100 streamlines are selected from the initial pool of 500 randomly seeded streamlines. The first image shows the overall streamline pool while the rest three snapshots show the selected streamlines under three different viewpoints. We map velocity magnitude to streamline color: blue to white to red is for low to medium to high magnitude. Our streamline selection favors “interesting” streamlines that reveal critical flow feature and patterns in a less cluttered view. Redundant streamlines, even with high importance values, are pruned to avoid repetition. However, since this view-independent selection algorithm only considers the global information, it is possible that the results may miss some flow patterns due to the lack of considering the view-dependent information, such as local clutter and occlusion under some particular viewpoints.

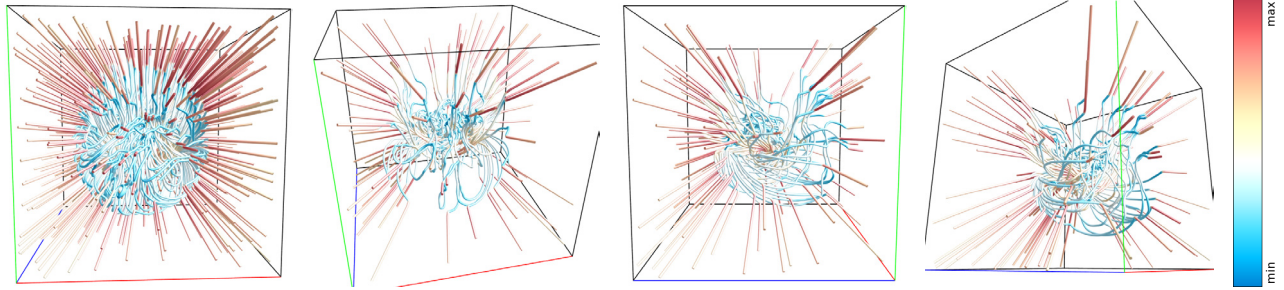


Figure 4. View-independent streamline selection with the supernova data set. The leftmost image shows the total 500 streamlines from the initial pool, followed by three images showing the same top 100 streamlines selected in the view-independent manner. Velocity magnitude is mapped to streamline color.

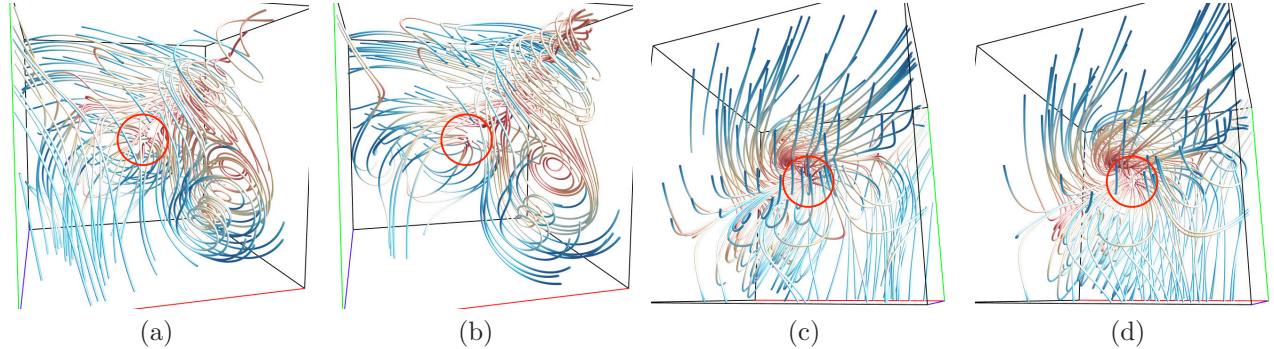


Figure 5. View-independent vs. view-dependent streamline selection with the five critical points data set under two different viewpoints. In both cases, we can observe that one critical point (source) near the center of the vector field is occluded in the view-independent selection results shown in (a) and (c). By contrast, this source is clearly visible in the view-dependent selection results shown in (b) and (d). The source is highlighted in a red circle.

6.3 View-dependent Selection Results

In Figure 5, we compare streamline selection under view-independent and view-dependent cases with the five critical points data set. Two different viewpoints are shown in the figure. Clearly, in both cases, the view-dependent selection tends to better cover regions with less dense streamlines due to the use of the density map. Since the streamlines with high priority mainly go through local critical regions and they gain the most importance value from the density map, the streamlines with low priority will not obtain enough importance value to be selected. This is the reason why the local interesting regions are less occluded by dense streamlines. The view-independent selection, however, tends to select more interesting streamlines in regions even though they are already pretty dense in the projection. This is because the view-independent selection only cares the overall importance of the streamlines but never considers local streamline occlusion under a given viewpoint. Specifically, in both cases, the view-independent selection hides a critical point (source) near the center of the vector field while the view-dependent selection shows this critical point much more clearly.

We also verified the usefulness of using the effective area by showing how it influences the view-dependent selection results with the hurricane data set. As shown in Figure 6, we used five consecutive viewpoints along the view sphere to show how the streamline number changes under different viewpoints. From the quantitative results obtained, it is clear that the number of streamlines selected varies proportionally with the change of effective area. Our solution is able to balance the number of streamlines as well as their density in the projection under different viewpoints.

6.4 Coherent Streamline Update between Adjacent Viewpoints

Figure 7 shows the streamline update along four consecutive viewpoints with the two swirls, solar plume and tornado data sets. In order to show our coherent streamline update effect in a more intuitive way, for the tornado data set, we differentiated the streamlines selected from the previous viewpoint in gold and the newly selected

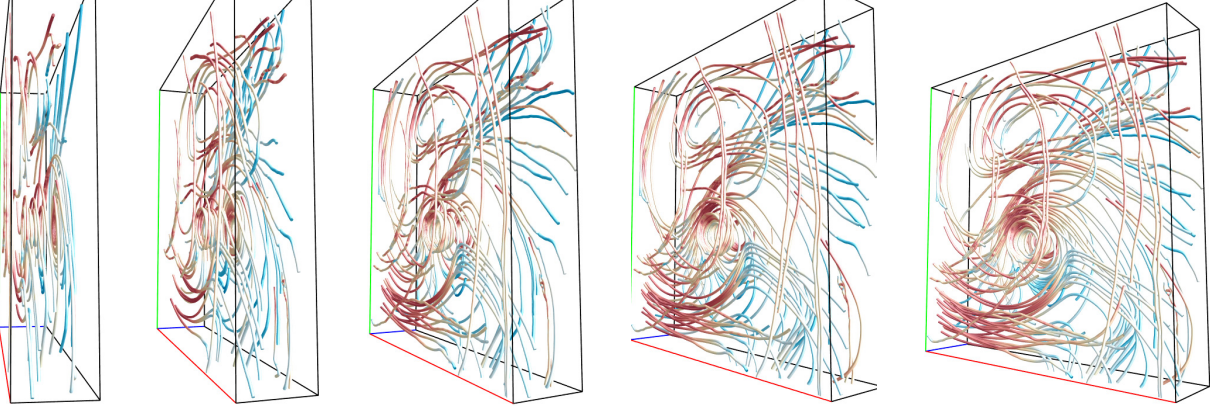


Figure 6. The number of streamlines selected increases with the increase of effective area for the hurricane data set. From left to right, the numbers of streamlines displayed are 57, 86, 103, 116 and 128, respectively. The corresponding overall effective density values are 87970, 96659, 163477, 189974 and 209153, respectively. The five viewpoints shown are consecutive on the view sphere.

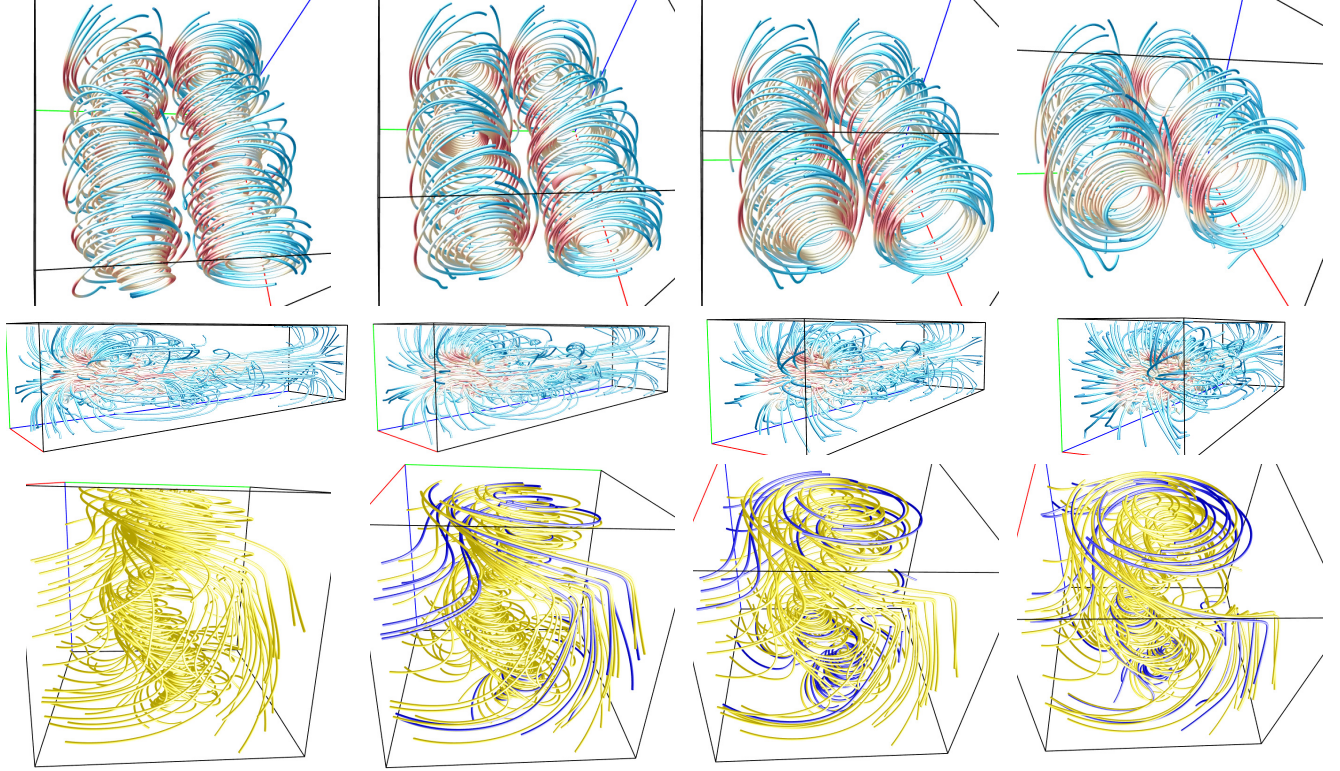


Figure 7. Coherent streamline update of three data sets: two swirls (top), solar plume (middle) and tornado (bottom). Left to right, each image is from one snapshot of four consecutive viewpoints. For the tornado data set, we differentiate the newly selected streamlines in blue while the streamlines remaining from the previous viewpoint are in gold.

streamlines in blue. Clearly, the less number of blue streamlines is, the better the current viewpoint preserves the previous viewpoint's information and the more coherent the view-dependent selection results are.

Figure 8 shows the statistics of the numbers of streamlines selected and shared with the supernova data set. We can see that the number of streamlines shared closely follows the trend of the number of streamlines selected. This is also confirmed by their ratio which remains flat over all sample viewpoints. These results show that our algorithm can guarantee coherent streamline update between consecutive viewpoints.

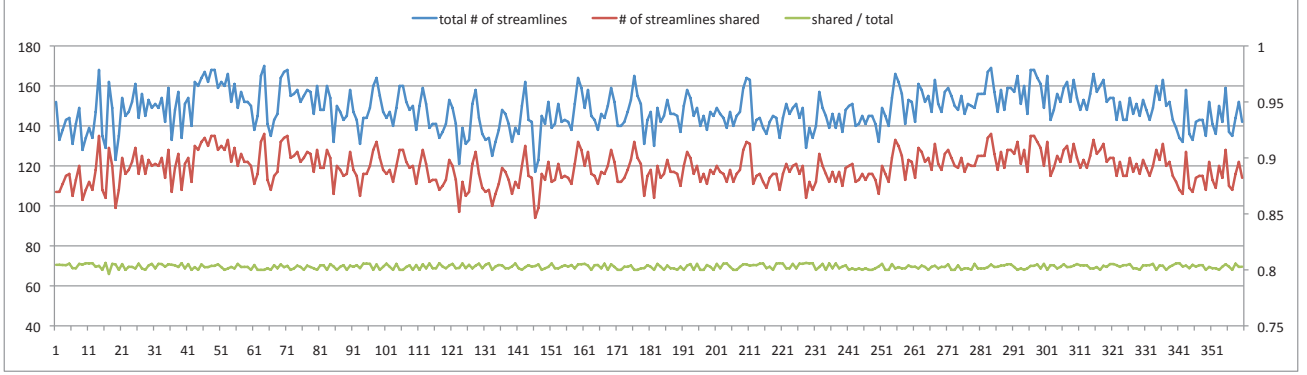


Figure 8. The statistics of the numbers of streamlines selected and shared with the supernova data set over 360 sample viewpoints. The blue line represents the total number of streamlines selected N_{all} under the current viewpoint. The red line indicates the total number of streamlines shared N_{shared} by the current and previous viewpoints. The green line displays the ratio: $r = N_{shared}/N_{all}$.

We also compared our algorithm with a naïve view-dependent streamline selection algorithm that directly selects streamlines based on their decreasing importance values under the current viewpoint in conjunction with the density map control. We conducted the comparison with the computer room and crayfish data sets. For the computer room data set shown in Figure 9, we differentiated the streamlines in the same way as we did in Figure 7. For each pair of images under the same viewpoint, we selected the same number of streamlines for fairness. Our algorithm always yields less blue streamlines than the naïve algorithm under the same viewpoint. For the newly selected streamlines (shown in blue), they are also more similar to the streamlines (shown in yellow) that have been selected under the previous viewpoint. Therefore, we confirm that our algorithm achieves more coherent streamline update than the naïve algorithm.

For the crayfish data set shown in Figure 10, the results under four consecutive viewpoints clearly show that our algorithm maintains a more coherent streamline update. The statistics results also confirm that our algorithm yields more shared streamlines (and therefore, less different streamlines) than the naïve algorithm. As a matter of fact, the streamline selection result produced from our view-dependent algorithm is in between the result from the naïve view-dependent algorithm (where the coherence is not considered) and the result from the view-independent algorithm (fully coherent, but the variation between views is not considered). The user has the freedom to adjust the values of related parameters to yield desirable results with different degrees of coherence.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented an importance-driven approach to coherent view-dependent streamline selection and visualization. By defining a streamline importance measure that combines mutual information and shape characteristic, we prioritize a large pool of candidate streamlines accordingly for selective visualization. We develop an algorithm for coherent view-dependent streamline selection by taking into account both the relationships between local and global viewpoint information and the coherence of the current and previous viewpoints. We demonstrate our view-dependent streamline selection algorithm with several flow data sets and compare it with a naïve algorithm. By presenting interesting streamlines in the first place to reveal essential view-dependent flow features, we are able to quickly convey the desired insight into the flow fields. This helps the users gain more understanding about the data in a shorter amount of time, which is critical for tackling data at scales. Coherent streamline update provides the users with a smooth way to explore the vector field over the view sphere. Our approach thus nicely complements existing work on streamline placement by providing a viable alternative to effective 3D streamline visualization.

However, there are several limitations in our current work which we would like to improve in the future. First, our method is not a flow-guided method such as the work by Ye et al.⁸ Therefore, like all solutions based on random or uniform seeding, we may need to generate a large pool of initial streamlines through dense seeding in the first place in order to capture all important features in the flow field. We may also need to select

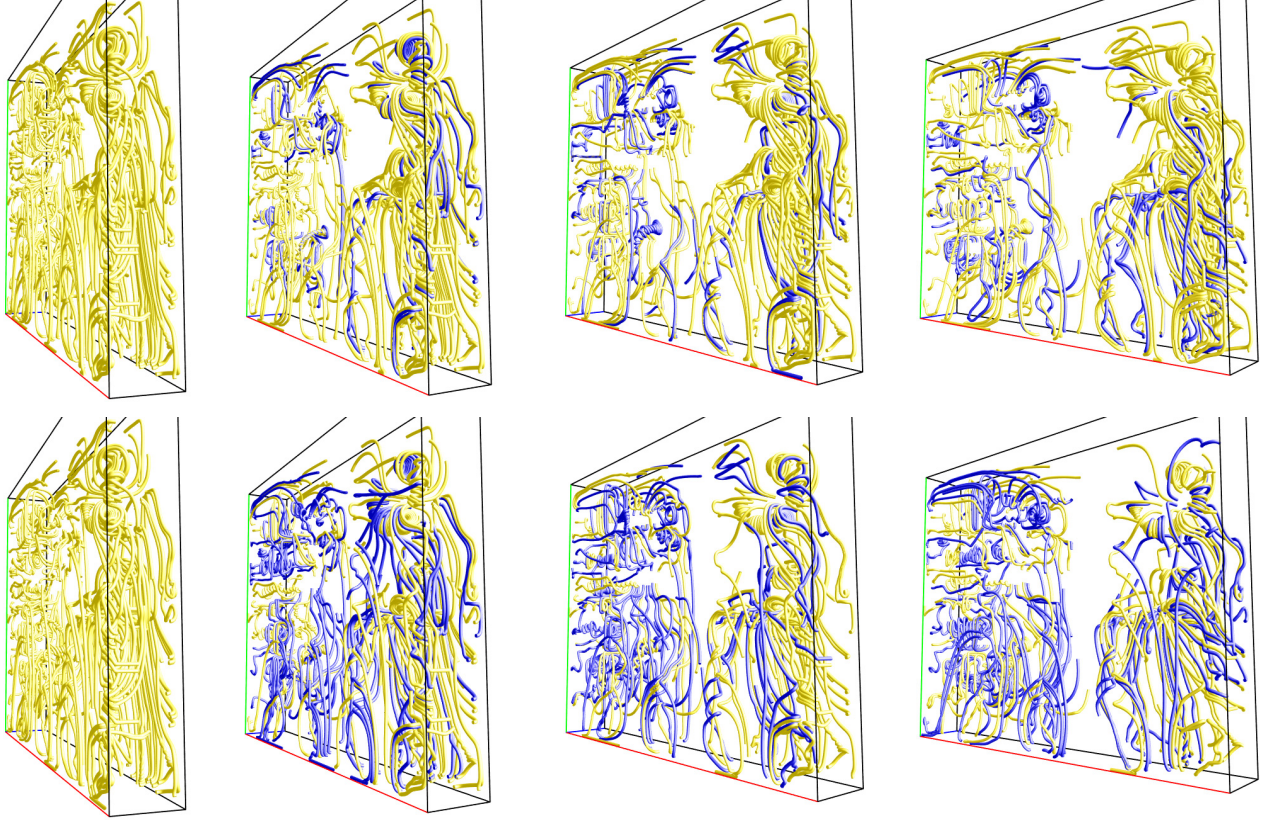


Figure 9. Our coherent vs. naïve view-dependent streamline selection with the computer room data set. Top: the view-dependent selection results from four consecutive viewpoints using our coherent algorithm. Bottom: the corresponding results using the naïve algorithm. We differentiate the streamlines selected in the previous view in gold and the streamlines newly selected in blue. From left to right: the total numbers of streamline selected are 138, 161, 193 and 198, respectively for both algorithms. Except for the leftmost image, 32 (66), 38 (79) and 40 (82) blue streamlines are shown for the coherent (naïve) algorithm, respectively.

more streamlines than the explicit feature-driven techniques in the visualization. Furthermore, our solution does not work well when the flow field consists of features at different scales as the entropy-based method is not sensitive to small-scale features. In this scenario, some shorter streamlines revealing small-scale features might be of low importance values due to their low 3D streamline entropies. We would like to improve our importance measure by taking into account the scale so that features of various scales can be captured without substantially increasing the number of streamlines selected. Second, the view sphere partition algorithm²² we adopted does not produce “smooth” view change as the change would become more dramatic towards the two poles (please refer to the accompanying video). We will consider adding intermediate viewpoints to improve the smoothness of view change. Third, for our view-dependent selection algorithm, we only consider the correlation between the current viewpoint and its previous one along a predefined traversal scheme to maintain coherent streamline update. However, we can instead utilize information from several viewpoints rather than only one. For instance, we may take into account several neighboring viewpoints centered at the current one. This allows the current viewpoint to incorporate more information from its neighborhood. As a result, we can achieve coherent streamline update with different viewpoint traversal schemes.

ACKNOWLEDGMENTS

This work was supported in part by the U.S. National Science Foundation through grants IIS-1017935, DUE-1105047 and CNS-1229297. We would like to thank Dr. Alex Pang for his valuable input to this work. We also thank the anonymous reviewers for their insightful comments.

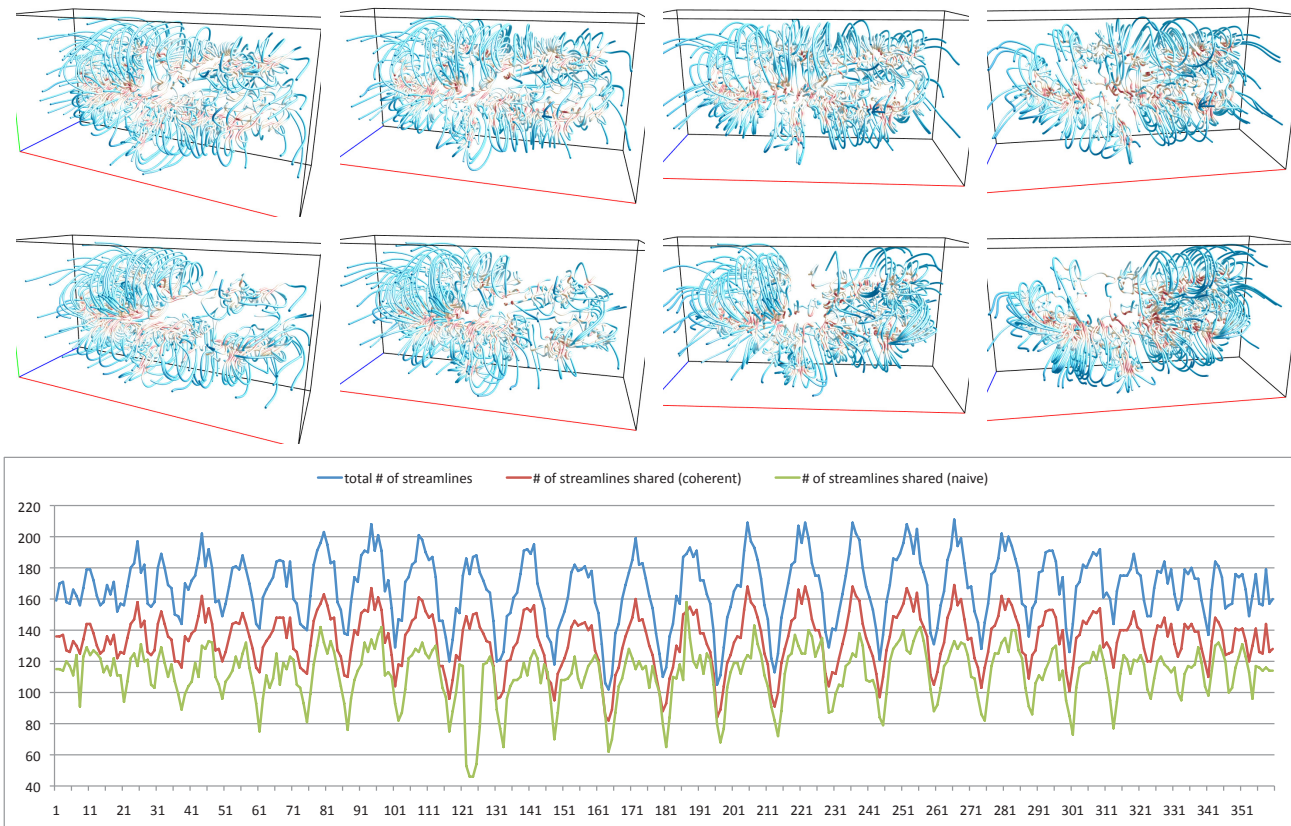


Figure 10. Our coherent vs. naïve view-dependent streamline selection with the crayfish data set. First row: the view-dependent selection results from four consecutive viewpoints using our coherent algorithm. Second row: the corresponding results using the naïve algorithm. From left to right: the total numbers of streamline selected are 78, 80, 92 and 97, respectively for both algorithms. Except for the leftmost image, 16 (41), 18 (49) and 19 (48) streamlines are newly selected for the coherent (naïve) algorithm, respectively. Third row: the statistics of the numbers of streamlines selected and shared over 360 sample viewpoints. The blue line indicates the total number of streamlines selected under each viewpoint. The red line shows the total number of streamlines shared by the current and previous viewpoints using our coherent algorithm while the green line shows the corresponding numbers using the naïve algorithm.

REFERENCES

- [1] Cabral, B. and Leedom, C., “Imaging vector fields using line integral convolution,” in *[Proceedings of ACM SIGGRAPH Conference]*, 263–270 (1993).
- [2] van Wijk, J. J., “Spot noise texture synthesis for data visualization,” in *[Proceedings of ACM SIGGRAPH Conference]*, 309–318 (1991).
- [3] van Wijk, J. J., “Image based flow visualization,” in *[Proceedings of ACM SIGGRAPH Conference]*, 745–754 (2002).
- [4] Turk, G. and Banks, D., “Image-guided streamline placement,” in *[Proceedings of ACM SIGGRAPH Conference]*, 453–460 (1996).
- [5] Jobard, B. and Lefer, W., “Creating evenly-spaced streamlines of arbitrary density,” in *[Visualization in Scientific Computing]*, 43–56 (1997).
- [6] Li, L. and Shen, H.-W., “Image-based streamline generation and rendering,” *IEEE Transactions on Visualization and Computer Graphics* **13**(3), 630–640 (2007).
- [7] Verma, V., Kao, D., and Pang, A., “A flow-guided streamline seeding strategy,” in *[Proceedings of IEEE Visualization Conference]*, 163–170 (2000).

- [8] Ye, X., Kao, D., and Pang, A., “Strategy for seeding 3D streamlines,” in [*Proceedings of IEEE Visualization Conference*], 471–478 (2005).
- [9] Chen, Y., Cohen, J. D., and Krolik, J. H., “Similarity-guided streamline placement with error evaluation,” *IEEE Transactions on Visualization and Computer Graphics* **13**(6), 1448–1455 (2007).
- [10] Marchesin, S., Chen, C.-K., Ho, C., and Ma, K.-L., “View-dependent streamlines for 3D vector fields,” *IEEE Transactions on Visualization and Computer Graphics* **16**(6), 1578–1586 (2010).
- [11] Wei, J., Wang, C., Yu, H., and Ma, K.-L., “A sketch-based interface for classifying and visualizing vector fields,” in [*Proceedings of IEEE VGTC Pacific Visualization Symposium*], 129–136 (2010).
- [12] Liu, Z., Moorhead, R. J., and Groner, J., “An advanced evenly-spaced streamline placement algorithm,” *IEEE Transactions on Visualization and Computer Graphics* **12**(5), 965–972 (2006).
- [13] Mebarki, A., Alliez, P., and Devillers, O., “Farthest point seeding for efficient placement of streamlines,” in [*Proceedings of IEEE Visualization Conference*], 479–486 (2005).
- [14] Schlemmer, M., Hotz, I., Hamann, B., Morr, F., and Hagen, H., “Priority streamlines: A context-based visualization of flow fields,” in [*Proceedings of Eurographics/IEEE VGTC Symposium on Visualization*], 227–234 (2007).
- [15] Liu, Z. and Moorhead, R. J., “Interactive view-driven evenly spaced streamline placement,” in [*Proceedings of IS&T/SPIE Conference on Visualization and Data Analysis*], (2006).
- [16] Spencer, B., Laramée, R. S., Chen, G., and Zhang, E., “Evenly spaced streamlines for surfaces: An image-based approach,” *Computer Graphics Forum* **28**(6), 1618–1631 (2009).
- [17] Li, L., Hsieh, H.-H., and Shen, H.-W., “Illustrative streamline placement and visualization,” in [*Proceedings of IEEE VGTC Pacific Visualization Symposium*], 79–86 (2008).
- [18] Xu, L., Lee, T.-Y., and Shen, H.-W., “An information-theoretic framework for flow visualization,” *IEEE Transactions on Visualization and Computer Graphics* **16**(6), 1216–1224 (2010).
- [19] Wu, K., Liu, Z., Zhang, S., and Moorhead, R. J., “Topology-aware evenly spaced streamline placement,” *IEEE Transactions on Visualization and Computer Graphics* **16**(5), 791–801 (2010).
- [20] Lee, T.-Y., Mishchenko, O., Shen, H.-W., and Crawfis, R., “View point evaluation and streamline filtering for flow visualization,” in [*Proceedings of IEEE Pacific Visualization Symposium*], 83–90 (2011).
- [21] Furuya, S. and Itoh, T., “A streamline selection technique for integrated scalar and vector visualization,” in [*IEEE Visualization Conference Poster Compendium*], (2008).
- [22] Leopardi, P., “A partition of the unit sphere into regions of equal area and small diameter,” *Electronic Transactions on Numerical Analysis* **25**, 309–327 (2006).
- [23] Tao, J., Ma, J., Wang, C., and Shene, C.-K., “A unified approach to streamline selection and viewpoint selection for 3D flow visualization,” *IEEE Transactions on Visualization and Computer Graphics*. Accepted.
- [24] Moberts, B., Vilanova, A., and van Wijk, J. J., “Evaluation of fiber clustering methods for diffusion tensor imaging,” in [*Proceedings of IEEE Visualization Conference*], 65–72 (2005).