Flow Web: A Graph Based User Interface for 3D Flow Field Exploration

Lijie Xu and Han-Wei Shen

Ohio State University, 395 Dreese Laboratories 2015 Neil Avenue, Columbus Ohio, USA

ABSTRACT

While there have been intensive efforts in developing better 3D flow visualization techniques, little attention has been paid to the design of better user interfaces and more effective data exploration work flow. In this paper, we propose a novel graph-based user interface called *Flow Web* to enable more systematic explorations of 3D flow data.

The Flow Web is a node-link graph that is constructed to highlight the essential flow structures where a node represents a region in the field and a link connects two nodes if there exist particles traveling between the regions. The direction of an edge implies the flow path, and the weight of an edge indicates the number of particles traveling through the connected nodes. Hierarchical flow webs are created by splitting or merging nodes and edges to allow for easy understanding of the underlying flow structures. To draw the Flow Web, we adopt force based graph drawing algorithms to minimize edge crossings, and use a hierarchical layout to facilitate the study of flow patterns step by step. The Flow Web also supports user queries to the properties of nodes and links. Examples of the queries for node properties include the degrees, complexity, and some associated physical attributes such as velocity magnitude. Queries for edges include weights, flow path lengths, existence of circles and so on. It is also possible to combine multiple queries using operators such as *and* , *or*, *not*. The Flow Web supports several types of user interactions. For instance, the user can select nodes from the subgraph returned by a query and inspect the nodes with more details at different levels of detail.

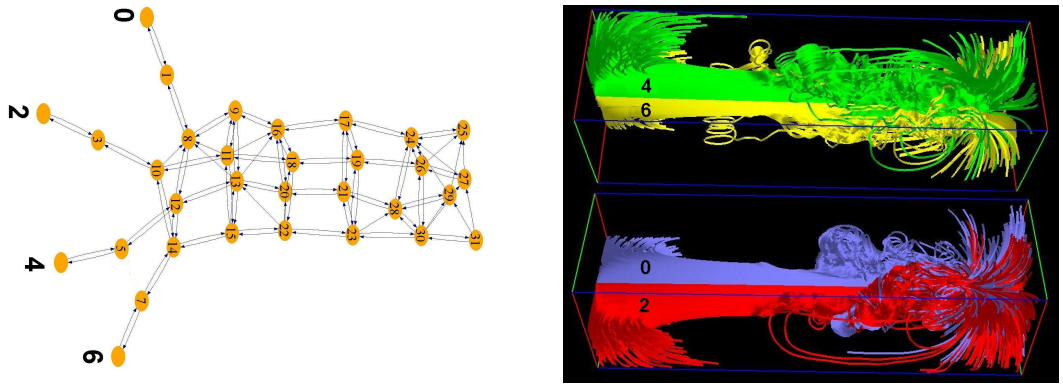
There are multiple advantages of using the graph-based user interface. One is that the user can identify regions of interest much more easily since, unlike inspecting 3D regions, there is very little occlusion. It is also much more convenient for the user to query statistical information about the nodes and links at different levels of detail. With the Flow Web, it becomes easier for the user to log and track the progress of data exploration which is crucial for exploring large data sets. We demonstrate how to construct and draw the Flow Web effectively, and how to query the Flow Web to retrieve useful information from the data. Case studies are provided to demonstrate the exploration process.

Keywords: Field Exploration, User Interface, Scientific Visualization, Graph Drawing, Graph Query

1. INTRODUCTION

Effective analysis of 3D flow data plays an important role in understanding complex physical phenomena modeled by large scale scientific simulations. Toward this goal, there have been many techniques developed to facilitate effective analysis and visualization of 3D flow fields. Topics of interest include, but are not limited to, generation of flow textures, placement of 2D and 3D streamline seeds, and analysis of critical points and salient flow features. While there have been great efforts in developing effective and efficient visualization algorithms, less attention has been paid to the issue of creating intuitive user interfaces to help the user explore the flow data more easily. Use streamline visualization as an example. It is usually difficult to decide where to place the streamline seeds although the streamlines can be computed and rendered interactively. And once many streamlines are displayed,

Further author information: (Send correspondence to Lijie Xu.)
Lijie Xu: E-mail: xu.136@osu.edu, Telephone: 1 614 688 3862



(a) The Flow Web created by our algorithm.

(b) The actual flow corresponding to the paths starting from node 0, 2, 4, 6 in the Flow Web.

Figure 1. Flow Web and Actual Flow.

it becomes difficult to know precisely the structures of the flow, and whether there are any important regions in the domain left unexplored due to projection and 3D occlusion.

In this paper, we propose a novel graph-based user interface, called *Flow Web*, to assist the exploration of 3D flow fields. Just as maps can help people tour around an unfamiliar city, the goal of the Flow Web is to help the user navigate through various regions of interest and query the flow properties in the field. Traditionally graphs have been commonly used for information visualization applications. In fact, some of the problems faced there are similar to problems in visualizing flow fields. For example, the goal of visualizing social networks is to analyze relationships within and among clusters of people; similarly in flow visualization, we want to understand the structure and connections of the flows through different spatial regions. The Flow Web summarizes the flow relations between spatial regions captured from sampling the field with a large amount of random particles. The nodes in the Flow Web represent 3D regions, and edges between nodes correspond to the paths of flow. Figure 1(a) gives an example of the Flow Web. In Figure 1(b), we cut open the flow field to show the four flow paths starting from node 0, 2, 4, 6 in the Flow Web. It can be seen that the actual flow in the spatial domain is summarized succinctly by the Flow Web, where the flows initially go straight then start to mix with each other into a turbulent region as they propagate. The 3D occlusion problem is alleviated by displaying the Flow Web with a force based layout model. Figure 6(b) gives a finer resolution of the Flow Web that reveals more detail.

There are multiple advantages of using the Flow Web graph-based user interface. First, with the global context presented, we can overcome 3D occlusion and allow the user to identify regions of interest much more easily. Second, the user is able to select nodes directly on the Flow Web to perform detailed exploration, or query statistical information about a node and the relations between nodes. Third, the user is able to visualize the flow field at different levels of detail by interactively refining or simplifying the Flow Web to systematically explore the flow field. Last but not the least, it becomes much easier for the user to log and track the progress of data exploration by marking the nodes and links that have already been explored. The user can also review or resume the exploration process at a later time.

The remainder of the paper is organized as follows. First we give a brief review of previous work in user interface for visualizing 3D flow fields. Then we propose the concept of the Flow Web and discuss how to construct the Flow Web from the input 3D data. Next, we describe the layout algorithms that are used, and the Flow Web's query features. To demonstrate the utility of our interface, we provide examples of exploring 3D flow fields with the Flow Web interface.

2. RELATED WORK

Understanding flow fields has been a focus of visualization research for decades. Many algorithms proposed in this area. Examples of classic texture based methods include Spot Noise,[1] LIC,[2] IBFV,[3] and intelligent seed placement such as.[4–7] Even though many 2D flow visualization algorithms can be easily extended to 3D, occlusion prevents them from sharing the same success as their 2D counterparts. For example, it is difficult to use LIC[2] to study 3D flow features without using auxiliary tools such as specially designed transfer functions or cutting planes. Although it allows the user to visualize part of the data at a time, the global context is often lost. The same problem exists for visualizing streamlines. For example, when using the evenly spaced streamline seed placement algorithms in 3D, the projections of 3D streamlines may arbitrarily intersect on the 2D screen so the final result may still not be satisfactory. Among the few 3D seed placement algorithms, Ye et al.[8] suggest to use different templates to place seeds around critical points to avoid cluttering. In this method, occlusion is still possible and the user needs to fill up the empty space with other methods which is difficult for non-experts.

Putting users in the loop and letting them explore the data through effective user interface provide promising solutions. There exists certain domain specific data exploration tools that provide valuable inspirations for good user interfaces. One such work is presented in visualization of astrophysical environment.[9] Volume rendering research has provided many good examples too, especially for medical data visualization where focus and context is of major concerns. One example is the magnification lens[10] that renders high levels of detail at the focus region while lower resolution contents are used as context. To visualize data at different levels of detail, Wang et al.[11] present a user interface using treemaps. Relatively speaking, very little effort has been put on user interface for flow field visualization. One example is to provide a selection tool to let the user examine vortices individually.[12] A recent work by Correa et al.[13] allows the manipulation of 3D LIC volumes to create the effect of peeling and deforming. SimVis[14] provides some examples of possible user interfaces for studying flow fields.

Besides the above research, our work is related to graph layout as well. Quigley et al.[15,16] utilize hierarchical data structures such as quadtree and octree to create hierarchical graphs and speed up force based layout of large graphs.

3. SYSTEM OVERVIEW

In this paper, we present a graph-based user interface, called Flow Web, to explore 3D flow fields. The purpose of the Flow Web is to provide the user with a global context of the flow field and a user interface to assist real time data exploration. The Flow Web allows the user to quickly identify important local regions by examining the nodes and links in the graph. Selection of the local regions and flows can be done by simple mouse clicks. The Flow Web is drawn on a 2D plane by a force based layout algorithm to minimize occlusion. With the Flow Web, the history of data exploration can be saved, and undo or redo of the exploration steps is thus feasible if so desired.

There are three major components in the Flow Web system:

Flow Web construction: Given an input flow field, we partition the domain into sub-regions and construct a directed graph, the Flow Web, to capture the flows through the sub-regions. The sub-regions are represented as nodes, and the flows that connect the sub-regions are represented as edges. Levels of detail selection of view to the domain is achieved by building a hierarchical Flow Web.

Flow Web drawing: After the Flow Web is constructed, it is important to draw it in a way that the user can obtain a clear view of the flow structure. In our system, the graph is drawn using a force based method to avoid cluttering. Other different layout methods are studied and used for different purposes of exploration.

Flow Web query and 3D Exploration: Different query operations are provided to allow the user to manipulate the Flow Web in order to hide or reveal information related to the flow field in different levels of detail. Users can explore the 3D flow field by interacting with the Flow Web and observing the visualization results in the spatial domain. Examples of Flow Web queries are shown in Section 6.

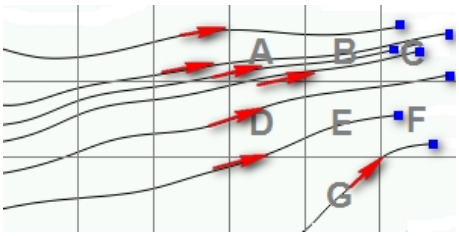


Figure 2. Flow Web construction: sampling with backward particles tracing.

4. FLOW WEB CONSTRUCTION

In this section, we describe in detail how to construct the Flow Web. Taking a 3D flow field as input, the construction algorithm outputs a weighted directed graph: the Flow Web. Nodes in the Flow Web represent spatial sub-regions in 3D space, while edges between the nodes indicate how the flow (i.e. particle traces) passes through the sub-regions. Weights of edges reflect the number of streamlines going through the nodes. It is important to note that the goal of the Flow Web is not to replace the conventional visualization techniques such as streamlines, but to provide an interface with an overview of the field so that the user can quickly decide regions of interest to perform more detailed explorations.

4.1. Spatial Partitioning and Streamline Sampling

The construction of the Flow Web contains two main steps. The first is to create the nodes of the Flow Web by subdividing the domain and have each node in the graph represent a 3D sub-region. The second step is to connect the nodes with edges generated from streamline sampling.

To construct the nodes, we first partition the 3D domain into a collection of sub-regions. To allow the user to navigate through the nodes in the graph and find the corresponding spatial sub-regions in the domain more easily, we chose to use regular axis aligned sub-regions to partition the field and create the Flow Web nodes. This allows the Flow Web to represent a simple but intuitive skeleton of the underlying domain.

Once the nodes of the Flow Web are determined, edges are constructed to connect the nodes by sampling the field using streamlines. With a good coverage of the domain by a large number of random streamlines, we can obtain an overview of how the spatial sub-regions are linked together by the flows. To generate the streamlines, particles are dropped at random positions within each subregion. Streamlines are traced backwards from the seeds so that we know the exact source of the flow that passes through the current sub-region. Backward particle tracing is used to ensure no regions in the domain will be missed without enough samples. Discussions about ensuring an even coverage of area by backward streamline tracing for texture advection can be also found in work by Becker et al.[17] and Jobard et al.[18]

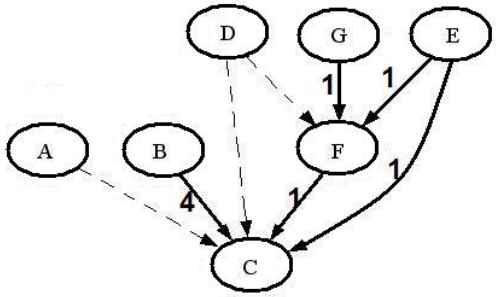
The number of particles dropped in each sub-region is determined by its volume. Each particle carries an id of its originating node and is traced backward until it goes out of bound or gets trapped by a critical point. Figure 2 illustrates how the sampling is done. The blue squares demonstrate where the particles are placed and the arrows show the flow directions. The 2D particle paths are shown here only for illustration purpose.

4.2. Flow Connection Matrix

To record the number of streamlines traveling from one region to another, we create a data structure called *Flow Connection Matrix*. The Flow Connection Matrix is an n x n matrix where n equals the total number of sub-regions. Each entry [A,B] in the matrix records the number of particles traveling from node A to node B following the forward flow direction. Since we trace backward streamlines, a unit is added to [A,B] when there is a backward streamline starting from B and passing A. With the Flow Connection Matrix, information about how streamlines travel through different regions can be recovered. It is worth noting that the information stored

	A	B	C	D	E	F	G
A	*	*	4	*	*	*	*
B	*	*	4	*	*	*	*
C	*	*	*	*	*	*	*
D	*	*	2	*	*	1	*
E	*	*	1	*	*	1	*
F	*	*	1	*	*	*	*
G	*	*	*	*	*	1	*

(a) A sample Flow Connection Matrix.



(b) A Flow Web constructed by displaying links between nodes represent adjacent regions in space (solid lines).

Figure 3. Flow Web construction: Flow Connection Matrix to Flow Web

in the matrix is usually not redundant and the value in one entry may not be derivable from other entries. For example, non-zero entries in [A,B] and [B,C] do not guarantee a non-zero entry in [A,C] in the matrix. This is because even if there exists particles traveling from A to B, and from B to C, there may not be any particle traveling from A to C.

The Flow Connection Matrix is constructed as follows. The entries of the Flow Connection Matrix are initialized to zero. When a particle entering a sub-region corresponding to a graph node, its id is deposited locally indicating the source of the flow. After the sampling of backward streamlines is done for all nodes, we collect the deposited ids inside each sub-region. If in the region of node B we found x ids from node A, we then set the value of the entry [B, A] to x in the Flow Connection Matrix. Figure 3(a) shows the Flow Connection Matrix generated from the example in Figure 2. In the example, five particles were placed in node C and traced backwards. Four of the particles pass through A and hence [A, C]=4; four of the particles pass through B and hence [B, C]=4; two of the particles pass through D so [D, C]=2; finally one particle passes through E and F, so [E, C]=1 and [F, C]=1.

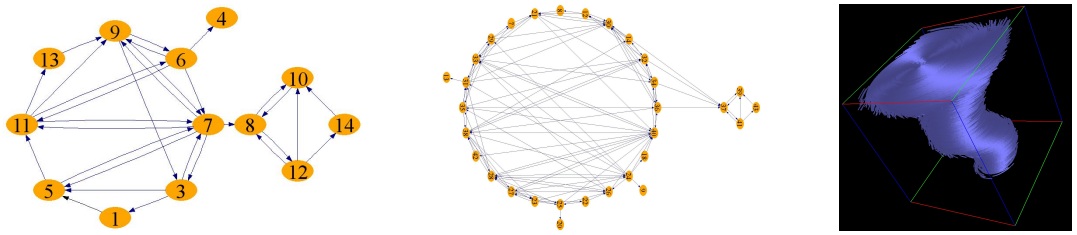
4.3. Flow Web Creation

After the Flow Connection Matrix is constructed, the Flow Web can be created. Initially, we display only the links between the nodes that are spatially adjacent. To prevent a large number of links from cluttering the graph, other links are not shown although the user can make queries to the Flow Connection Matrix to augment and modify the initial graph. In the example shown in Figure 2, node B and C are adjacent to each other, while node A and node C are remote nodes because they do not share any common boundaries. The edges that connect adjacent nodes are displayed in the Flow Web (solid lines in Figure 3(b)). Edges connecting remote nodes are ignored (dotted lines in Figure 3(b)).

The computation time for flow web construction includes two major parts: the time to integrate streamlines and the time to generate the graph layout. We create approximately two hundred thousands streamlines needed for 96x96x96 tornado dataset, for example, in 20 seconds on a Pentium IV computer. For generating the graph layout we use the GraphViz[19] engine which takes 1 second on a force based model for a graph with 256 nodes and 490 edges.

4.4. Flow Web Hierarchy

Having a hierarchical structure for the Flow Web, and presenting the graph to the user at an appropriate level of detail is crucial to making a large graph manageable for exploration. For this purpose, a hierarchical representation of the Flow Web is built on top of a multi-resolution Flow Connection Matrix as follows. First the data set is diced up by an octree data structure. The size of the leaf nodes are user selected which determines



(a) A Flow Web created by our system.

(b) An example of higher resolution for the same Flow Web.

(c) The actual flow.

Figure 4. Hierarchy of the Flow Web

how deep the hierarchy will go. The finest resolution of the Flow Connection Matrix is constructed with one node for each leaf region in the octree. Coarser resolutions of the Flow Connection Matrix are computed by combining sibling nodes into one. The adjacent nodes are merged, and the weights are summed up accordingly. Then a Flow Web is built based on the Flow Connection Matrix at the current resolution. Because streamline sampling only needs to be done once at the finest resolution, different levels of the hierarchy can be created on demand with real time performance.

With the hierarchical representation of the flow field, we are able to display the Flow Web at different scales. For example, Figure 4(b) shows a higher resolution version of the Flow Web from the one in Figure 4(a). Nodes on the larger circle in Figure 4(a) have been expanded to a finer resolution and thus more detail are shown.

The user can also decide to show different level of detail for nodes in the flow web either open up the Flow Web by clicking on the nodes of interest or keep it at the current level. If the user selects to open up the Flow Web, the next level is retrieved and displayed. For example, node 20 (drawn in red) in Figure 5(a) has been opened up and more details are displayed (Figure 5(b)). Figure 5(c) is a zoom-in view of the opened up area.

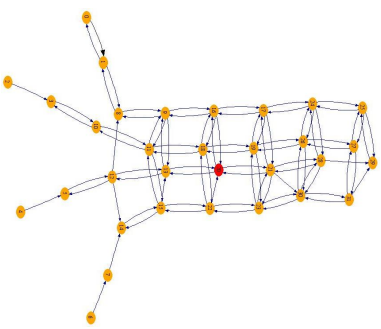
The Hierarchical Flow Web is important for exploring large-scale complex data sets because displaying the whole Flow Web all at once can overwhelm the user. By selecting a small set of nodes to explore at the beginning, the user can focus on the features of interest without losing the global context. With the hierarchy, some existing focus and context algorithms for visualizing graphs can also be used such as the one described in.[20]

5. FLOW WEB DRAWING

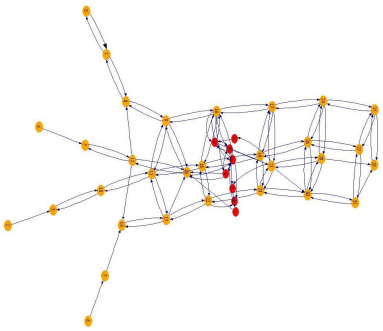
A proper Flow Web layout is essential to having a clear display of the structure in the flow field. In our system, different layout algorithms can be chosen according to different exploration goals. The user will start the exploration process with a primary layout which is the first Flow Web shown to the user. The goal of the primary layout is to give the user an overview of the flow field; in other words, the primary graph should have minimum cluttering and in the mean time preserve the major flow structures among the nodes.

To generate the primary Flow Web layout, we use a force based graph layout algorithm,[21] which is a classic algorithm to reduce node and edge cluttering by modeling nodes as steel rings and edges as elastic strings. Although the force based layout algorithm is usually used for undirected graph drawing, it suits our needs quite well.

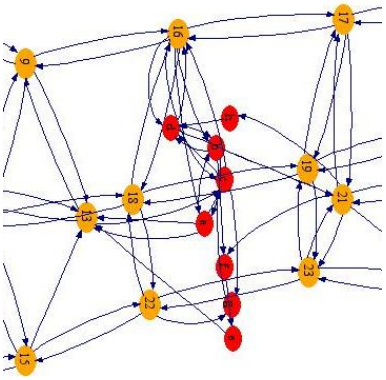
When creating the Flow Web edges by integrating streamlines, we put an upper bound for adaptive step sizes to ensure that streamlines will not skip adjacent regions. This way, when two nodes are linked together, they must be two neighboring regions in space. This gives our Flow Web a nice property in that a node connects to other nodes only through its immediate neighbors (8-neighbor for 2D, 26-neighbor for 3D). Even if the force driven layout method is not necessarily designed to display hierarchical relations of data, this property of edges prevents the force model from pulling two nodes together when they are far away in space.



(a) Flow web at a coarse resolution.



(b) Flow web at a finer resolution, with node 20 (red) opened up.



(c) A closer look at the opened up node.

Figure 5. Multilevel views by *open* operation.

In addition to the needs of separating nodes and minimizing edge crossings, we also want the Flow Web to depict the spatial locations of the nodes in the domain as intuitively as possible. Because we allow the user to open and close nodes in the Flow Web, the Flow Web may change frequently. In order to let the user easily relate a Flow Web node to its corresponding spatial location, an important property for the Flow Web is that its layout should be stable. That is, the layout of the Flow Web should change locally if the Flow Web changes locally too.

Since the output of the force based layout algorithm depends on the initial positions of the nodes, in our algorithm, we start by carefully selecting the initial 2D position for each node representing a 3D sub-region. We first take the largest value of X, Y, Z coordinates of the region centroid to be the y position of the node on the 2D Flow Web, and then map the remaining two coordinates to the node's x position. To avoid regular patterns and also edge cluttering for nodes that have the same y coordinate, we jitter the resulting (x,y) coordinates for the nodes. In addition, we take into account the minimum edge length allowed to scale and separate the nodes. Figure 1(a) shows an example of the Flow Web computed from the plume data set, where the flow starts from the left with four bundles and enters turbulent regions as it propagates to the right.

In addition to the force based method for generating the primary layout, there are several other classic graph drawing algorithms that can be used for different data exploration purposes. Currently we use the Graphviz[19] toolkit as the main graph drawing program, and the user is allowed to choose different layouts for different purposes. The circular layout, for example, shows local flow paths clearly if the directions of edges are monotone. This method can be used to detect and display circular flows by following the monotone edge directions until the

starting node is reached again. Not only are cycles are easy to observe in this layout but also the nesting relations between different sizes of cycles are clearly illustrated. In addition to the two layout algorithms discussed here, we can also use the hierarchical layout to follow the global flow directions. It is also easy to spot the source and sink nodes and follow the streamlines.

6. FLOW WEB QUERY

Just like touring an unfamiliar city, exploring an unknown data set can be confusing. By using the Flow Web as a user interface, a roadmap is in the user's hand so data exploration can be performed in a more systematic way. Besides providing an overview of the flow structure, an important function of the Flow Web is to allow the user to perform queries. The goal of queries is to retrieve the nodes and edges that satisfy certain criteria and to reduce the size of the Flow Web graph. This way, the user only needs to focus on a small subset of the Flow Web to explore the data.

The data set we use in the examples is a solar plumes data set (126 x 126 x 512), which was generated by a simulation of thermal downflow plumes on the surface layer of the sun. The data contain both laminar and turbulent flow structures. The Flow Web we created for the plume data set contains 32 nodes and 155 edges at the top most level.

6.1. Queries

6.1.1. Query Quantities of Each Node

Queries can be related to the nodes or to the relations between nodes. For queries that are related to the nodes, the focus is on the statistics of the underlying flow related quantities. After the queries, only the nodes that satisfy the criteria and the edges connected to those nodes are displayed. Colors can be used to highlight the nodes returned from the query.

Query Scalar Quantities of a Node: The user can filter the Flow Web by thresholding the nodes according to some auxiliary scalar quantities such as velocity magnitude, entropy, and pressure.

Node complexity: A node complexity is defined as the standard deviation of vectors inside the node. Higher node complexity means a more complex flow pattern in the sub-regions so they may require further explorations.

Degree of node: A node with a smaller degree often means that the flow going through it is relatively simple and hence can be combined with its neighboring nodes or ignored. On the other hand, nodes with larger degrees often exist in places where the directions of flow changes rapidly such as the areas near critical points. Sorting the nodes by their degrees can be a hint to the user for prioritizing the exploration order.

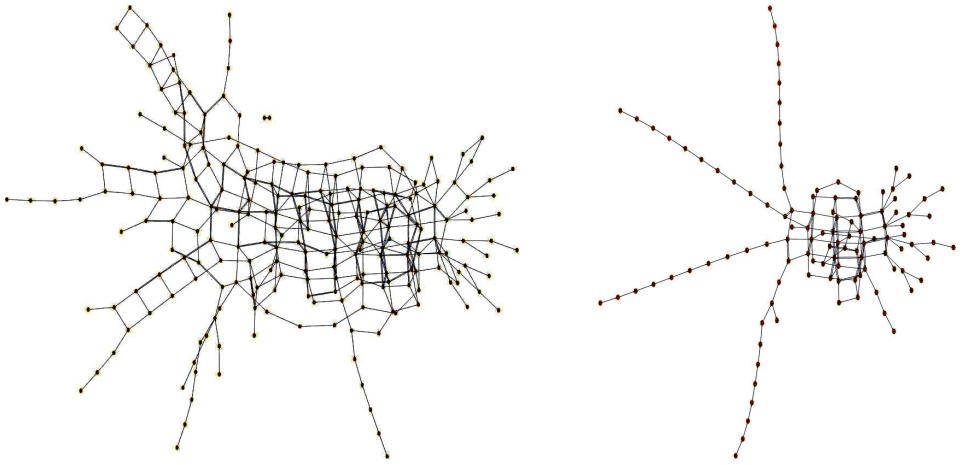
6.1.2. Query Relations Between Nodes

Querying the relations between nodes is an essential way to explore the flow field. Properties of interest include the weight of an edge, the length of streamline paths between nodes, locations of connected components, nodes with circular connections in between, streamline bundles, and subdivision of the field, just to name a few. In the following, we discuss these queries.

Weight of edge: As described earlier, the weight of an edge is the number of sampled streamlines connecting two nodes. The user can set a threshold to remove the weaker connections(edges) in the Flow Web to simplify the graph without loosing important flow features.

Length of streamline path between nodes: For this type of query, the user can choose any two nodes in the Flow Web and select the streamline paths based on their lengths. The flow along a longer path between two nodes is usually more complex than the shorter ones since a shorter path implies relatively more straight flow.

Connected components: For this type of query, the user can choose one node and extract the nodes that are connected by one or multiple edges to that node. This query allows the user to roughly segment the field into



(a) Flow Web created from the Plume data set.

(b) The Flow Web nodes are filtered by their velocity magnitudes.

Figure 6. Query helps reduce the cluttered graph

different separate components. Note that this query is carried out using the information stored in the Flow Connection Matrix since the Flow Web being displayed may not contain all edges.

Circular connections between two nodes: A flow is simple if it has a pattern of $A \rightarrow B \rightarrow C$. If node A and node B have flows go into each other such as $A \leftrightarrow B$, then there exists a complex flow pattern in the region between the two nodes. In this case, the user can select a finer resolution and perform further exploration.

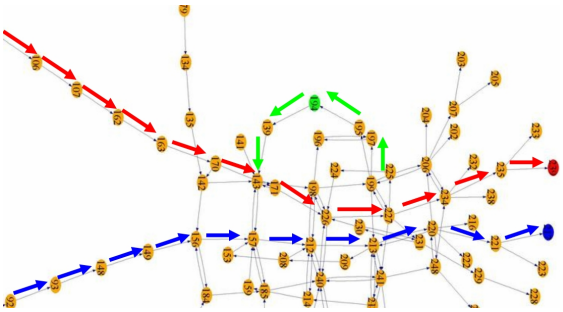
Streamline bundles: Streamline bundles are defined as a group of streamlines that travel sufficiently close to each other along most of their paths. Streamlines belonging to the same bundle exhibit similar characteristics and hence studying them together can reduce the user's exploration time. Applications in diffusion tensor imaging (DTI),[22] [23] also have the need to identify such bundles in the data. With the Flow Web, the user can issue a query to detect and verify streamline bundles in a progressive manner. The user can pick multiple nodes that are connected on the Flow Web. During the selection of each node, the display is updated such that only streamlines that pass through all the selected nodes will be displayed. By selecting and deselecting nodes, the user can interactively detect or verify hypothesized bundles.

6.2. Combinations of Queries

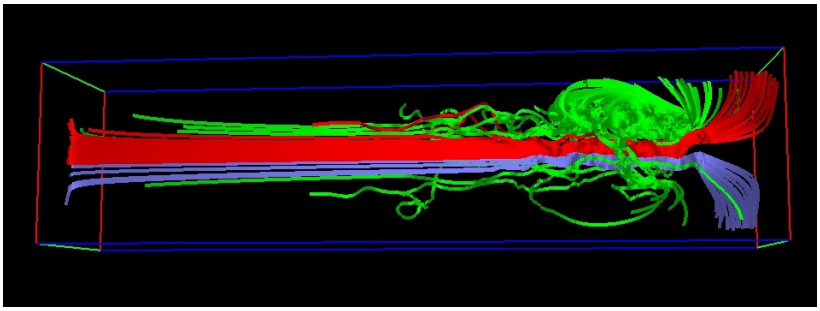
Although each query described above is used in a standalone manner, more complicated queries can be built by combining multiple queries using logic operators like *and, or, not*.

Combining queries together has several advantages. One is that the user can build arbitrarily complicated queries from simple building blocks. For example, the user may be interested in seeing the longest path that goes through nodes of the maximum degree. The query can be translated into our building blocks as $\{degreeofnode + loopremoval + topologicalsort + pathlength\}$

Secondly, combination of queries can be used to track the history of the exploration. Being able to record the query history so that it can be easily modified is important to help the user recover from previous undesired exploration steps, and build more effective queries from experience. Besides, keeping the query history allows the user to easily understand what has been done and how it was done. The user can modify a sequence of queries by adding or deleting one or multiple existing queries, as long as there are no order sensitive queries in the sequence. One example of order sensitive combination is *loop removal + topological sort*. These two queries must be added/removed all at once. Adding/removing only one of them may result in invalid results.



(a) Query by clicking the selected nodes.



(b) The actual 3D streamline bundles queried.

Figure 7. Query for streamline bundles by mouse.

Combining queries can be implemented by giving each node/edge a true/false flag for each query. The final true/false state of each node/edge is determined by logically combining all the query results. Removing a query from the sequence is the same as removing all the flags from the Flow Web related to the removed query and recalculating the state for each node/edge from the remaining flags.

6.3. Explore the Flow Web

With a manageable sized Flow Web obtained by filtering, the user can start the exploration of the flow field with the queries. Figure 8 gives an example of exploration that identify streamline bundles that consist of streamlines with similar characteristics to study them separately. In this example we attempt to identify three classes of streamlines: long turbulent streamlines, long straight streamlines and short streamlines. To achieve this goal, we set up the queries as a combination of the degrees of nodes and length of paths. For long and turbulent streamlines, we place seeds at high degree nodes, denoted as seed 1 in Figure 8. For long and straight streamlines, we choose root nodes of straight flow web paths as the seeds, indicated as seed 2 and seed 3 in the figure. For short streamlines, seeds are placed at the roots of short paths in the flow web, indicated as Seed 4, 5 and 6 in the figure. Note that the hierarchical layout can facilitate choosing the flow paths more easily since it is easy to hide unnecessary detail from the data. As we can see, with the help of the flow web, it is relatively easier to locate the interested seeds than probing without guidance. At the same time, the exploration for an individual category of flow path is more systemic controllable with the context presented.

7. CONCLUSIONS

In this paper, we present a novel graph-based user interface called Flow Web to explore 3D flow fields. The Flow Web is used to interactively query and search the properties of nodes and edges, and to guide user exploration and identification of salient flow features. The hierarchical structure of the Flow Web makes it possible to analyze large scale data sets in different levels of detail. The 3D occlusion problem is alleviated by effective display of

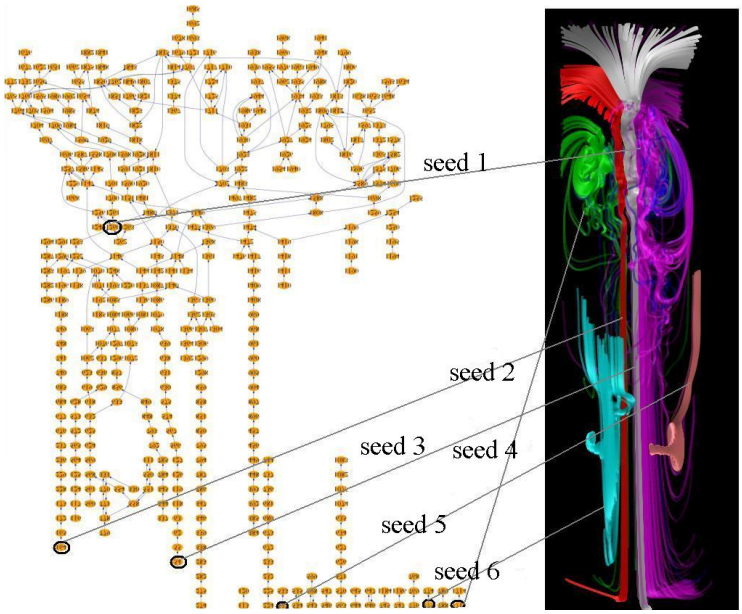


Figure 8. Exploration with the flow web

the Flow Web in a 2D graph. Besides displaying the flow structure at different levels of detail, the Flow Web can provide the user with quantitative properties associated with nodes through queries. With the Flow Web user interface, the user is able to quickly focus on regions of interest without losing the global context. The exploration process can also be logged to facilitate explorations for large datasets.

8. FUTURE WORK

Although our current Flow Web construction method provides satisfactory results, we believe it can be improved so that the number of nodes and edges in the final Flow Web is further reduced. One alternative is to provide the user with the option of creating irregular shaped regions for the nodes. The tool box of query in our system can be extended. In addition to the graph algorithms we have already adopted, there are several other graph theories to be studied to extend the utility of the Flow Web. Max-flow-min-cut theory, for example, is one candidate to study in the future research. With the help of this algorithm, we can partition the Flow Web more effectively. Projecting the 2D partitioning boundary to 3D may give us a meaningful segmentation boundary in the domain. Other research directions include designing new graph drawing methods and extending our interface to handle time-varying data.

REFERENCES

1. J. J. van Wijk, "Spot noise texture synthesis for data visualization," in *Proceedings of ACM SIGGRAPH '91*, pp. 309–318, 1991.
2. B. Cabral and C. Leedom, "Imaging vector fields using line integral convolution," in *Proceedings of ACM SIGGRAPH '95*, pp. 263–270, 1995.
3. J. J. van Wijk., "Image based flow visualization," in *Proceedings of ACM SIGGRAPH '02*, pp. 745 – 754, 2002.
4. G. Turk and D. Banks, "Image-guided streamline placement," in *Proceedings of ACM SIGGRAPH '96*, pp. 453–460, 1996.

5. B. Jobard and W. Lefer, "Creating evenly-spaced streamlines of arbitrary density," in *Visualization in Scientific Computing*, pp. 43–56, 1997.
6. V. Verma, D. T. Kao, and A. Pang, "A flow-guided streamline seeding strategy," in *IEEE Visualization*, pp. 163–170, 2000.
7. A. Mebarki, P. Alliez, and O. Devillers, "Farthest point seeding for efficient placement of streamlines," in *IEEE Visualization*, pp. 479–486, 2005.
8. X. Ye, D. T. Kao, and A. Pang, "Strategy for seeding 3D streamlines," in *IEEE Visualization*, pp. 471–478, 2006.
9. A. J. H. Yinggang Li, Chi-Wing Fu, "Scalable wim: Effective exploration in large-scale astrophysical environments," *IEEE Transactions on Visualization and Computer Graphics* **12**(5), pp. 1005–1011, 2006.
10. L. Wang, Y. Zhao, K. Mueller, and A. E. Kaufman, "The magic volume lens: An interactive focus+context technique for volume rendering," in *Proceedings of IEEE Visualization*, pp. 367–374, 2005.
11. C. Wang and H.-W. Shen, "Lod map - a visual interface for navigating multiresolution volume visualization," *IEEE Transactions on Visualization and Computer Graphics* **12**(5), pp. 1029–1036, 2006.
12. S. Stegmaier, U. Rist, and T. Ertl, "Opening the can of worms: An exploration tool for vortical flows," in *Proceedings of IEEE Visualization*, pp. 463–470, 2005.
13. C. Correa, D. Silver, and M. Chen, "Illustrative deformation for data exploration," *IEEE Transactions on Visualization and Computer Graphics* **13**(6), pp. 1320–1327, 2007.
14. "Simvis." http://www.vrvis.at/simvis/index.html.
15. A. J. Quigley, "Large scale relational information visualization, clustering, and abstraction," *PhD thesis, University of Newcastle* , 2001.
16. A. J. Quigley and P. Eades, "Fade: Graph drawing, clustering, and visual abstraction," in *GD '00: Proceedings of the 8th International Symposium on Graph Drawing*, pp. 197–210, Springer-Verlag, (London, UK), 2001.
17. D. L. B.G. Becker and N. Max, "Unsteady flow volumes," in *IEEE Visualization*, pp. 329–335, 1995.
18. G. Jobard, B. Erlebacher and M. Hussaini, "Lagrangian-eulerian advection for unsteady flow visualization," *IEEE Transactions on Visualization and Computer Graphics* **8**(3), pp. 211–222, 2002.
19. "graphviz- graph visualization software." http://www.graphviz.org.
20. E. Gansner, Y. Koren, and S. North, "Topological fisheye views for visualizing large graphs," *IEEE Symposium on Information Visualization 2004* , 2004.
21. T. M. J. Fruchterman and E. M. Reingold, "Graph drawing by force-directed placement," *Software-Practice and Experience* **21**(11), pp. 1129–1164, 1991.
22. A. Brun, H. Knutsson, H. Park, M. E. Shenton, and C. Westin, "Clustering fiber traces using normalized cuts," in *In: MICCAI*, pp. 368–375, 2004.
23. S. Zhang and D. H. Laidlaw, "Dti fiber clustering in the whole brain," in *Proceedings of IEEE Visualization*, pp. 28–28, 2004.