Similarity Measures for Enhancing Interactive Streamline Seeding

Tony McLoughlin, Mark W. Jones, Robert S. Laramee, Rami Malki, Ian Masters, and Charles D. Hansen

Abstract—Streamline seeding rakes are widely used in vector field visualization. We present new approaches for calculating similarity between integral curves (streamlines and pathlines). While others have used similarity distance measures, the computational expense involved with existing techniques is relatively high due to the vast number of Euclidean distance tests, restricting interactivity and their use for streamline seeding rakes. We introduce the novel idea of computing streamline signatures based on a set of curve-based attributes. A signature produces a compact representation for describing a streamline. Similarity comparisons are performed by using a popular statistical measure on the derived signatures. We demonstrate that this novel scheme, including a hierarchical variant, produces good clustering results and is computed over two orders of magnitude faster than previous methods. Similarity-based clustering enables filtering of the streamlines to provide a non-uniform seeding distribution along the seeding object. We show that this method preserves the overall flow behavior while using only a small subset of the original streamline set. We apply focus + context rendering using the clusters which allows for faster and easier analysis in cases of high visual complexity and occlusion. The method provides a high level of interactivity and allows the user to easily fine-tune the clustering results at run-time while avoiding any time-consuming re-computation. Our method maintains interactive rates even when hundreds of streamlines are used.

Index Terms—Flow Visualization, Clustering, Similarity Measures, Focus+Context, streamlines

✦

1 INTRODUCTION

STREAMLINES are curves that are everywhere tangent to a steady-state (time-invariant) vector field. They depict the path a massless fluid element traverses at any given time. The placement of these curves strongly affects the impact of the resultant visualization. Many automatic streamline seeding strategies are presented in visualization literature [19]. However, in practice these are not commonly used by Computation Fluid Dynamics (CFD) experts. Reasons for this stem from requiring knowledge of the seeding algorithm to correctly interpret the results. Also some seeding strategies place emphasis on uniform coverage using evenly-spaced streamlines [13], however changes in the physical proximity of streamlines may convey important properties of the flow that are lost while using a technique based on producing a fixed resolution output. Also domain experts may not be interested in the entire spatial domain. Their efforts may be focused on investigating a specific sub-region. In this case, a global seeding strategy may add visual clutter to the resulting visualization and impede the

- *T. McLoughlin, M. W. Jones and R. S. Laramee are with the Visual Computing Group, Department of Computer Science, Swansea University. email:{cstony, M.W.Jones, R.S.Laramee}@Swansea.ac.uk*

- *R. Malki and I. Masters are with the Marine Energy Research Group, College of Engineering, Swansea University. email:{R.Malki, I.Masters}@Swansea.ac.uk*

- *C. D. Hansen is a Professor at the Scientific Computing and Imaging Institute, University of Utah, Salt Lake City. email: hansen@cs.utah.edu*

investigation by the user. Consequently, CFD engineers rely heavily on manual seeding. In fact, the popular visualization package, TECPLOT [1], includes no automatic seeding of streamlines and relies entirely on the user to do so.

There is less focus on research enhancing the user experience while employing manual seeding. Typically, streamlines are seeded at equidistant positions along a curve or plane with little further opportunity for interaction or refinement. In many cases, this does not result in a visually optimal set of streamlines for the given seeding object. Whilst working with CFD experts we found that they predominantly use interactive seeding when using streamlines to investigate their data. CFD experts rely heavily on the derived visualizations for disseminating the results of their simulations. The work presented here aims to enhance the domain expert user's experience while employing this frequently used tool. We provide novel interaction with, and control of, the set of streamlines produced from interactive seeding objects. This allows the user to easily customize the resultant visualization enabling them to portray their results with more flexibility. Our method relies on only a small number of parameters which are simple to navigate. We place a high-level of importance on this observation in order to provide an improved user experience. The user is not required to navigate an unintuitive, high-dimensional parameter-space.

The core of our method is a set of similarity measures to compare streamlines. Clustering based on similarity is then performed, which then enables several enhancements such as a focus+context visualization and filtering of streamlines to leave an expressive subset of streamlines. The main

contributions of this paper are:

- A novel approach for computing a signature for an integral curve, and its use for similarity testing using the χ^2 (Chi squared) test.
- The production of hierarchical signatures and their use for computing the similarity of spatially shifted integral curves.
- The use of previous and novel measures for integral curves, and their comparison to existing state-of-the-art techniques. The combination of these measures and signature offers computation two orders of magnitude faster.
- An interactive algorithm for streamline filtering along the seeding primitive.
- A focus+context visualization based on the streamline clusters.
- An algorithm that maintains a high level of interaction with a large number of streamlines per seeding rake.

Many observations motivate this work. Our discussion with domain scientists demonstrates that they primarily use rakes to visualize and explore vector fields. Rakes tend to be the first tool of choice because they offer real-time interactivity with no pre-computation, provide an intuitive visual representation of the data and do not rely on complicated user parameters. The drawbacks of rakes are that resulting visualizations can be cluttered, there is no existing way to highlight streamlines or to customize the visualization to produce high quality rendering for communication and presentation purposes. Controlling the streamlines using streamline seeding or placement algorithms could improve this situation, but this introduces lengthy pre-computation. We investigate this area and report an approach that offers a solution to these problems. This work is related to the well researched topic of seeding to control streamline placement and bundling of DTI fibers. Our approach is compared to existing algorithms in those areas. The comparison demonstrates that our new approach has applications to general integral curve similarity calculations.

The rest of the paper is organized as follows. Section 2 provides a survey of related literature. Section 3 provides the overview and detailed description of our method. Section 4 describes the datasets used in the examples in this paper. Domain expert evaluation and enhancements to our algorithm are presented in Section 5. Section 6 contains a discussion of our algorithm in comparison with other state-of-the-art techniques and provides performance results. Finally, Section 7 concludes the paper with directions of future work.

2 RELATED WORK

Here, we discuss related work in the areas of similarity metrics for streamlines and other integral curves, automatic seeding strategies for global placement of streamlines and clustering from a similarity matrix.

2.1 Streamline Similarity Metrics

Streamline similarity metrics have been widely used to control the number and proximity of streamlines for streamline

placement applications. The goal is to produce uncluttered visualizations of flow fields whilst maintaining the depiction of the major features. The area was introduced by Turk and Banks [27] through streamline seeding whilst minimizing an image-space energy function. This was extended using a farthest point seeding by Mebarki *et al.* [20]. Evenly-spaced streamlines [13][26] are another solution to the seeding problem. For example, Liu *et al.* [15], incorporate the goals of maximizing streamline length, seeding based upon distance controls and loop detection to place streamlines. Chen *et al.* [2] observe that (a) streamline placement algorithms tend to use a uniform resolution that either potentially misses salient features or contains redundant streamlines; or (b) rely on feature detection in order to sample streamlines adequately, leading to problems due to incorrect feature identification. They propose a similarity metric that allows them to adapt streamline resolution in the vicinity of dissimilar streamlines. Their similarity metric is based on computing distances between points along a streamline that leads to slower non-interactive computational times compared to our approach. Li *et al.* [14] present a 'less is more' approach to streamline seeding. The goal is to capture the most important flow features using the fewest streamlines. This produces results comparable to hand-drawn diagrams. This similarity metric is also distance based. It is demonstrated in 2D with low numbers of streamlines (relying on a distance transform). Extending to a large 3D volume with the number of streamlines we enable and maintaining interactivity is unresolved as we demonstrate with our comparison in Section 6. Other relevant work includes streamline predicates by Salzbrunn and Scheuermann [24] which are boolean maps that are used to differentiate streamlines based on input queries from the user. Similar to flow topology, the idea is to partition the domain into regions of coherent flow behavior. Shi *et al.* [25] create an interactive environment wherein they calculate various properties for pathlines, and then allow the user to analyze and create selections upon the basis of those attributes. Janicke *et al.* [11] and Daniels *et al.* [5] operate directly on the vector field by mapping multi-dimensional points into an attribute space that is subsequently interactively visualized. Both methods use vortex detection for working with flow fields. The aim is that it will be possible to select similar regions throughout the data. Cucitore *et al.* [4] also propose local criteria for vortex detection that could be built into such a system or used within our streamline attributes (Section 3.1).

2.2 Similarity for DTI Fiber Tracts

Distance metrics have also been applied in the domain of DTI fiber clustering. For an introduction to the area see Moberts *et al.* [21] where they review various clustering approaches and distance metrics for DTI fiber clustering. Two widely implemented and state-of-the-art techniques are by Corouge *et al.* [3] and Zhang *et al.* [29]. Corouge *et al.* [3] introduce a symmetric distance measure based on the mean of all the distances of the closest point on
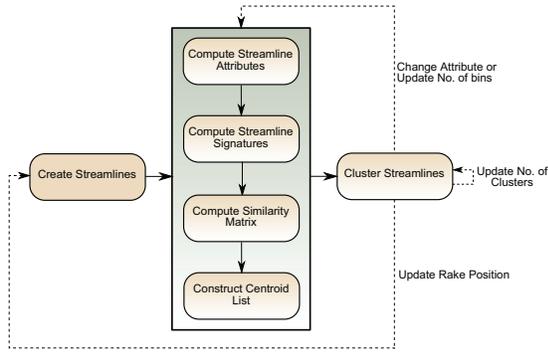
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS

MCLOUGHLIN *et al.*: SIMILARITY MEASURES FOR ENHANCING INTERACTIVE STREAMLINE SEEDING

3

Fig. 1. Overview of our algorithm pipeline. First, stream-lines are traced. Streamline signatures are then computed based upon streamline attributes. The streamline signatures are used to order the streamlines based on similarity. This ordering is then used to extract the cluster centroids and the streamlines are assigned to the relevant cluster. Dotted lines show user interaction and which stages of the pipeline need to be re-computed resulting from interaction.

curve B from each point on curve A. Zhang *et al.* [29] also introduce a threshold into the distance so that curves that are close for a good portion of their length but then diverge widely at the end are quantified as distant. They also compare their method to Corouge *et al*. Demiralp and Laidlaw [6] introduce a weighting term in order to weight the ends of the curve more in the distance calculation and also introduce a perceptual coloring. Jianu *et al.* [12] extend that work further ([6]) with a coordinated views representation of the DTI model and the clustering. They use average linkage hierarchical agglomerative clustering. See Jain *et al.* [10] for a classification of clustering. In Section 6.1 we compare our method to the measures by Corouge *et al.* [3], Zhang *et al.* [29] and Chen [2].

2.3 Streamline Perception in 3D

The goal of good streamline placement is a representation that is free of visual clutter and contains the salient features. There are many algorithms for 2D streamline placement, but 3D placement remains a more challenging problem. Mattausch *et al.* [18] provide several strategies for interacting with evenly-spaced flow data in 3D, also providing a focus+context like visualization by treating the separation distance as a measure of interesting features. More recently Marchesin *et al.* [17] present a view-dependent strategy for seeding streamlines in 3D vector fields. Based on the observation that no distribution of streamlines is ideal for all viewpoints, this method produces a set of streamlines tailored to the current viewpoint. The algorithm begins by seeding a random set of initial streamlines. These are then filtered according to an occupancy buffer, which tracks the number of streamlines for a given pixel and various filtering techniques such as angular entropy.

Visual clutter can be reduced by using differing techniques. For example Mallo *et al.* [16], demonstrate an improvement on illuminated lines [18] that exploits the use of diffuse and specular reflection to streamlines to create better perception of spatial structure. The introduction of

such a shading technique also helps reduce the visual clutter of large numbers of similarly colored lines. Additional techniques include additive blending and edge bundling [9] techniques for streamlines. Our solution is to adapt the ideas from distance-based similarity metrics, improve upon them for computational speed and apply them to interactive seeding rakes.

Yu et al. [28], present an algorithm that computes a hierarchy of streamline clusters. Their streamline similarity metric is based on curvature and torsion. Rather than computing curvature and torsion for each streamline, they derive these two attributes for every data sample in the domain. Their algorithm is prohibitively expensive for a single CPU implementation. The algorithm we present is fully interactive on a single CPU.

3 STREAMLINE SIMILARITY

Our algorithm begins with the user seeding a set of stream-lines using an interactive seeding object. Once the seed positions have been set the streamline trajectories through the vector field are computed. During the integration we ensure that the streamlines are created with their samples placed equidistantly. If an adaptive integrator is used then the streamlines are re-sampled. Next, streamline signatures are computed based on the set of attributes (Sections 3.1 and 3.2). A similarity matrix is constructed using the χ^2 test (Section 3.3). Agglomerative clustering is carried out using the similarity matrix. The user selects the desired number of clusters which are obtained from the clustering dendrogram. Streamlines are associated with an appropriate cluster. The user can vary the number of clusters interactively to customize the level of detail and their desired visualization (Section 3.4). We also provide a hierarchical variant of our algorithm where multiple signatures of differing resolutions are created for each streamline. When comparing a pair of streamlines the χ^2 test is run once for each signature in the hierarchy. This extension requires more computation but addresses issues where shifted signatures may appear very dissimilar.

In order to facilitate user interaction our system only re-computes the necessary stages of the pipeline as the user interacts with the algorithm parameters. For example, once the similarity matrix and dendrogram have been computed the user can vary the number of clusters without recomputing those stages. If the user changes the streamline attribute for the streamline signature, then the signatures, similarity matrix and streamline clustering are re-computed. If the user moves the seeding rake, the algorithm is performed starting with the streamline integration. Figure 1 depicts the algorithm overview. The dashed lines show the stages in the pipeline that are affected by the corresponding user-interaction.

3.1 Streamline Attributes

In order to compute the similarity between streamlines we use a number of new and existing attribute measures – curvature, torsion and tortuosity. Curvature measures

how much a curve deviates from a straight line. Torsion measures how much a curve bends out of its osculating plane. Tortuosity quantifies how twisted a curve is.

We compute a curvature field for the entire spatial domain. The curvature field is computed with the same sampling as the underlying vector field. Curvature at an arbitrary position along a streamline is interpolated from the curvature field. Curvature, **c**, is computed by [23]:

$$\mathbf{c} = \frac{\mathbf{v} \times \mathbf{a}}{|\mathbf{v}|^3} \qquad (1)$$

where **v** is the local velocity and **a** is the local acceleration computed by multiplying the local velocity gradient (Jacobian) with the local velocity $(\nabla \mathbf{v})\mathbf{v}$. We only require the magnitude of the curvature, $|\mathbf{c}|$.

Similarly we pre-compute a torsion field and assign the values to the streamlines. Torsion, τ, is derived by [23]:

$$\tau = \frac{(\mathbf{v} \times \mathbf{a}) \cdot ((\nabla \mathbf{a})\mathbf{a})}{|\mathbf{v} \times \mathbf{a}|^2} \qquad (2)$$

The final attribute we use is *tortuosity*. We have found this to produce good results on streamlines whilst having a low cost to compute. It is the ratio of the length of curve compared the shortest distance between its start and end points. We apply this to streamlines as a measure of deviation from the shortest path. The tortuosity of a streamline is computed by first summing the distances between all streamline segments. This value is then divided by the distance between the start and end points of the streamline:

$$S_T = \frac{1}{||\mathbf{f}(N) - \mathbf{f}(1)||} \sum_{i=1}^{N-1} ||\mathbf{f}(i+1) - \mathbf{f}(i)|| \qquad (3)$$

where $\mathbf{f}(x)$ is the spatial location of each sample in the vector field and N is the number of points in the streamline. Following this definition, the tortuosity of a straight line is one, and streamlines with higher tortuosity will demonstrate greater deviation from the direct path.

For all streamline points we compute the curvature, torsion and tortuosity values. Each attribute value is then normalized to the range [0,1] over all streamlines. All attributes for a given point are then summed. Normalizing each attribute places equal importance on each and prevents a large value in one attribute from reducing the importance of the others. This eliminates the requirement of user-defined weightings for each parameter, thus, making the computation fully automatic.

3.2 Streamline Signatures of Density-Based Streamline Attributes

In exceptional cases an overall quantity using the above metrics may produce the same or similar value for a range of streamlines. Thus, dissimilar streamlines may appear similar according to a given measure. For example, using the curvature criterion, a streamline that spirals three times would produce the same result as a more random curve that exhibits the same amount of curvature over its length (Figure 2). To alleviate this problem, and further differentiate

Fig. 2. Both of these curves exhibit very similar curvature magnitude values. Therefore, a single global measure fails to distinguish streamline sufficiently. In this case, the tortuosity attribute would fair better but there are cases where this would also fail. A better method creates a distinctive signature for the streamlines using density-based attributes.

streamlines, we introduce the novel concept of a *streamline signature*. Our motivation for this approach is that this stores a compact description of a streamline and facilitates a matching algorithm (hence the term signature). It is more descriptive than just the attributes from Section 3.1. The matching algorithm (Section 3.3) produces a single dissimilarity rating based on the signature and is shown to be very effective at distinguishing streamlines. The streamline signature is computed by splitting the streamline into several sub-curves or bins consisting of equal numbers of points. The metric is then computed for each bin. This set of values then describes how the attribute changes over the length of the streamline.

We set a number of points per bin. We discuss the effects of increasing and decreasing this number later (Section 3.3.3). We then iterate over each streamline point and calculate which bin it lies in. The point attributes are then computed (as outlined in the previous sections) and the value is added to the bin. When the entire streamline has been traversed the signature is complete. This computation creates a density-based pattern for each streamline. We refer to these patterns as density-based as they are computed as the sum of the above attributes per unit length of the curve. The signature shows the distribution of the attribute values along a curve. Figure 3 demonstrates some example density-based signatures.

3.3 Similarity Measure

We now introduce a novel approach to computing a similarity measure using the streamline signatures. This measure compares streamline signature patterns using the χ^2 test:

$$\chi^2(P_A, P_B) = \sum_{bin \in B} ((P_{bin,A} - P_{bin,B})^2 / (P_{bin,A} + P_{bin,B})) \quad (4)$$

where patterns P_A and P_B correspond to the streamline signatures of two streamlines, A and B. The χ^2 test utilizes the streamline signatures to provide a single value that more accurately measures the dissimilarity between streamlines. Identical streamlines result in $\chi^2 = 0$, and $\chi^2 > 0$ for non-identical streamlines. A larger result describes a greater magnitude of dissimilarity. The advantage of using the χ^2 test is that it produces a single value measure of dissimilarity between two streamlines just using their signatures. It operates on the binned data, and is therefore fast to compute (compared to operating on the raw streamline data or for example using the distance metrics [2] [14] where
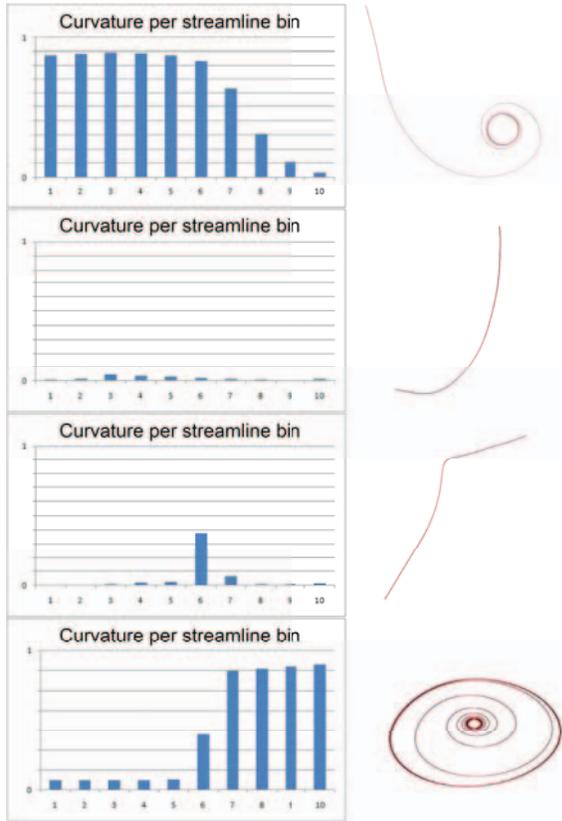
Fig. 3. Curvature over the streamline intervals. Top: The streamline starts in a vortex, but spirals out. Second row: The streamline follows a large arc. Third row: A similar streamline, but this one approaches a saddle point in the middle. Bottom row: A vortex spirals inwards. Note, that to depict the signatures more clearly, this example only demonstrates curvature. The actual computation uses the combination of attributes as outlined in Section 3.1.
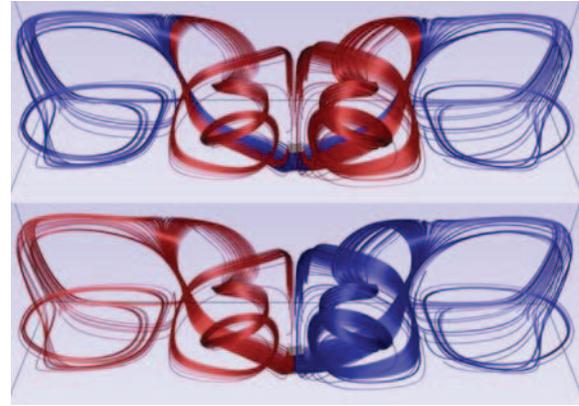

Fig. 4. Streamlines seeded from a seeding plane. The top image shows streamline clustering based solely on using the streamline signatures ($\alpha = 0$). The bottom image shows clustering using only the euclidean distance measure ($\alpha = 1$). Which set of clusters is correct is subjective. Our method provides the flexibility to allow the user to quickly navigate to their preferred results.

distances between numerous points along both streamlines are evaluated).

In the case where the number of bins in P_A and P_B are not equal, we iterate only over the number of bins contained in the shorter streamline. This produces partial matching, where only the corresponding portion of the longer streamline is compared to the shorter one. This produces a lower value when the χ^2 test is performed, i.e., the curves are more similar. Another alternative is to give the smaller streamlines the same number of bins as the largest streamline and assign the bins with a value of 0. Thus, when the χ^2 test is performed a greater value is produced, resulting in the streamlines being more dissimilar. However, we favor the first approach because, if we used the latter approach smaller streamlines (which cover only a small part of the volume) would have large dissimilarity, and so would end up having high significance in the visualization.

3.3.1 Similarity Matrix

The χ^2 test is performed for all streamline pairs, from which, a 2D matrix, M^{sim}, of similarity values is constructed. The similarity matrix provides a fast lookup table

for the clustering phase of our algorithm. Each column in the matrix corresponds to the set of similarity values for a streamline against all others and the row determines which streamline it is measured against. Entry $M^{sim}_{i,j}$ corresponds to the dissimilarity between streamlines i and j. The similarity matrix is therefore a symmetric matrix, whose main diagonal is composed of zeros, i.e., $M^{sim}_{i,j} = M^{sim}_{j,i}$ and $M^{sim}_{i,i} = 0$.

3.3.2 Euclidean Distance Measure

Previous distance metrics attach a high weight to proximity. In those approaches two similarly shaped streamlines far apart are more dissimilar than two dissimilarly shaped streamlines collocated. Since our approach compares streamlines based on signatures related to streamline shape, it is independent of proximity and may match streamlines not collocated. Therefore we introduce a weighting based on distance to give the user more control over this aspect. The default is for no weighting (zero) attached to proximity (so only the signature is matched). If the user desires close streamlines to have a higher similarity, the weighting can be increased using a slider. This occurs in real-time, so the user can explore this parameter space interactively. We provide this option by adding a lightweight distance measure into our pipeline.

Many distance tests result in the degradation in performance of similarity algorithms, this is demonstrated in Section 6. We keep the number of distance tests to a minimum as they are only meant to supplement our χ^2 similarity measure. We record the position of the last point in every bin. The mean of the distances between these end of bin points for each pair of streamlines is used to construct a second similarity matrix.

A new distance similarity table, $M'^{sim}_{i,j}$, is then combined with the $M^{sim}_{i,j}$ similarity table and a weighting coefficient to produce the final result. The similarity value for a given

similarity matrix element, $M_{i,j}^{\prime sim}$, is equivalent to this single measure:

$$M_{i,j}^{\prime sim} = (1 - \alpha)M_{i,j}^{sim} + \alpha mean\_dist(i,j) \qquad (5)$$

where α is the weighting coefficient and $mean\_dist(i,j)$ is the mean distance between streamlines i and j, computed using only a subset of their points as outlined above. This extra measure provides the expert user with more control over the clustering results. Figure 4 shows the effect of this parameter.

3.3.3 Choice of Bin Size

If we have a high number of bins a finer sampled signature is produced. In some cases this may produce a very localized change in the signature. This can lead to problems with the streamline bins incorrectly aligning between a pair of streamlines. For example, a pair of neighboring streamlines that both have a point of inflexion in their signatures, will have a spike due to a large change in curvature. However, if the spike occurs at a slightly different position (arc-length) along each streamline, a finer sampling of the signatures may result in the inflexion point occurring in different bins on the streamlines. This would result in the χ^2 test producing a high dissimilarity for these streamlines. A slightly more coarse sampling for the signatures provides a greater probability that the feature is captured by the same bin(s) and thus gives the desired result. This problem is greatly reduced when using rakes and seeding planes, where the seeding object is set to be orthogonal to the local flow. To reduce this problem further we use a hierarchical approach described next. Hadjidemetriou *et al.* [7] apply a similar technique to create multiresolution histograms for image recognition.

3.3.4 Hierarchical Signatures

During the streamline integration phase we record the maximum streamline length and use this as the basis for computing the number of levels in the signature hierarchy. We take the number of sample positions in the longest

Fig. 5. Our similarity measures and clustering algorithm segment the streamlines on this rake into distinct, intuitive clusters. Two rakes were used to generate two sets of streamlines on a simulation of Bénard convection in this image. Color indicates cluster membership. Note colors are re-used for each rake (i.e. the two red clusters are distinct clusters).

streamline and then round that number up to the nearest power of 2. The number of levels in the signature hierarchy is then computed as:

$$levels = \log_2 x \qquad (6)$$

where x is the lowest power of 2 which is larger than the number of samples in the longest streamline. For example, if the longest streamline contains 1000 sample positions, $x = 1024$ and $levels = 10$.

Next we compute the resolution of the signature at each level. At the lowest level we create x bins. Therefore, for the longest streamline there is a one-to-one mapping between samples and bins. For shorter streamlines with fewer samples we set the values of the extraneous bins to 0. As we proceed up the signature hierarchy the number of samples per bin doubles at each level, i.e., the second lowest level has 2 samples per bin and the next level has 4 samples per bin. For a given level the number of samples per bin can be computed by:

$$samples\_per\_bin = 2^{level} \qquad (7)$$

The top level in the hierarchy has a single bin which contains all of the samples for a given streamline.

The computation of multiple signatures per integral curve is not expensive nor wasteful in terms of memory because we just need to store the signature. The χ^2 test is computed once per level, during each iteration we sum the corresponding bins that comprise the higher levels. A further optimization is to store the summed values at each iteration and over-write some of the values in the bins. For example, if we had 4 bins with values 3,5,1,8 on the second iteration we would change the bin values to 8,9,-,-. The number of bins used halves for each level upwards and so for this iteration we only need to iterate over the first two bins for the χ^2 test. The following iteration would combine the bins again, producing 17,-,-,-. Bins represented by a '-' are unused in the computation.

We present a comparison between using a single signature and the hierarchical scheme. We demonstrate that the hierarchical case is more aware of shifted signatures and further differentiates vastly differing signatures when compared to the non-hierarchical approach.

Figure 6 shows a test case of three signatures. The middle (B) signature is a shifted version of the top (A) signature. The bottom (C) signature varies greatly from the two above.

Table 1 shows the steps of our hierarchical method applied to the signatures shown in Figure 6. The top table shows the similarity computation between the top and middle signatures and the bottom table shows the similarity computation between the top and bottom signatures. The middle signature is the same as the top one but has been shifted along the x-axis. The columns in table 1 show the χ^2 computations for each level of the hierarchy between the corresponding bins of the signatures. The average of all the similarity values is then computed.

The table shows that for the top two levels in the signature hierarchy the signatures between A and B are

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS

MCLOUGHLIN *et al.*: SIMILARITY MEASURES FOR ENHANCING INTERACTIVE STREAMLINE SEEDING

7

Bin	Level 1	2	3	4	5	Result
0	0.000	0.000	0.000	0.000	0.000	
1	0.000	0.000	0.000	0.000		
2	0.000	0.000	14.519			
3	0.000	0.000	21.778			
4	0.000	6.231				
5	0.000	8.805				
6	0.000	24.500				
7	0.000	0.000				
8	0.000					
9	7.364					
10	17.190					
11	0.000					
12	17.190					
13	7.364					
14	0.000					
15	0.000					
Total	49.108	39.536	36.296	0.000	0.000	124.940

Bin	Level 1	2	3	4	5	Result
0	12.250	19.593	29.455	58.909	53.751	
1	7.364	9.941	29.455	9.800		
2	2.667	19.593	13.755			
3	7.364	9.941	0.889			
4	12.250	19.593				
5	7.364	0.615				
6	2.667	0.455				
7	7.364	9.941				
8	12.250					
9	7.364					
10	2.667					
11	0.000					
12	0.714					
13	0.000					
14	2.667					
15	7.364					
Total	92.313	89.671	73.553	68.709	53.751	377.997

TABLE 1

These tables show the χ^2 computations using our hierarchical method for the signatures shown in Figure 6. The top table computes the similarity between the top two signatures and the bottom table computes the signature between the top and bottom signature. The intermediate χ^2 result for each bin is shown in the columns. The overall χ^2 for each column is shown in the bottom row. The final value in the bottom row is the final similarity result.

identical. For the first 3 levels the similarity values are 49.108, 39.536 and 36.296 respectively. When all levels in the hierarchy are averaged a value of 24.988 is obtained.

The bottom table shows that the overall similarity computation between signatures A and C as 75.599. Other than the levels where the signature is identical, using the hierarchical approach, the ratio of dissimilarity between the two pairs of curves is greater than using a single level. Using our hierarchical approach, the level of dissimilarity is more accurately conveyed than using any single level in the non-hierarchical version. The exception to this rule is when both signatures are identical, in which case both methods indicate produce the same results and successfully show them as identical.

Using a hierarchical scheme handles shifted streamline signatures in a more robust way that regards shifted signatures as similar while maintaining the ability to differentiate dissimilar signatures.

3.4 Clustering

Our M'^{sim} matrix represents the similarity between each streamline. The lowest entry of M'^{sim} represents the most

Fig. 6. Test streamlines for the heirarchical algorithm. Left: A streamline with a high peak towards the end. Middle: A shifted version of the top streamline. Right: A dissimilar streamline.

similar two streamlines. We employ agglomerative hierarchical clustering using pairwise average-linkage. See Jain *et al.* [10] for an overview on data clustering, and [2], [3], [29] for examples of use in the integral curve matching literature. The result is a tree (dendogram) recording each merge during the clustering process. On user interaction, a cut can be made through the tree for a given number of desired clusters. Figure 6 shows the results of our clustering algorithm on a simulation of Bénard convection using two seeding objects.

4 DATA SETS

This section discusses the datasets used as examples in this paper.

Hurricane Isabel: The simulation of Hurricane Isabel is sampled at a resolution of $512 \times 512 \times 100$ over 48 timesteps. It is a simulation of a Category 5 hurricane making landfall in North Carolina. This simulation exhibits several examples of interesting behavior such as vortices and saddle points. In the case where we demonstrate our method for unsteady flow using pathlines, we use the entire temporal domain. Hurricane Isabel data produced by the Weather Research and Forecast (WRF) model, courtesy of NCAR and the U.S. National Science Foundation (NSF).

Bénard Convection: A simulation of Rayleigh-Bénard convection. This simulation is sampled at a resolution of $256 \times 128 \times 64$. A plane is heated at the bottom of the spatial domain creating a pattern of Bénard convection cells. This simulation was created and provided by Daniel Weiskopf (University of Stuttgart).

Smoke Plume: A simulation of the evolution of a smoke plume. The simluation was supplied by Han Wei Shen (Ohio State University, Columbus) and is sampled at a resolution of $126 \times 126 \times 512$.

Arnold-Beltrami-Childress Flow: We also use a synthetic dataset of Arnold-Beltrami-Childress (ABC) flow. This describes a closed-form solution of Euler's equation [8]. This type of flow has theoretical importance in fluid dynamics and has been used many times in both fluid dynamics and visualization literature. The vector field is given:

$$v(x,y,z) = \begin{pmatrix} A\ sin(z) + B\ cos(y) \\ B\ sin(x) + C\ cos(z) \\ C\ sin(y) + A\ cos(x) \end{pmatrix}, \ x \in [0, 2\pi]^3 \quad (8)$$

where $A = \sqrt{3}$, $B = \sqrt{2}$ and $C = 1$.

5 DOMAIN EXPERT BASED EVALUATION

5.1 Domain Expert Involvement

The research in this paper resulted from a project team consisting of four computer scientists and two flow engineers that have worked together for over two years on various projects. Five of the team are located in the same building, and a remote collaborator has visited multiple times. The close collaboration between the interdisciplinary team has resulted in the visualization team gaining a good understanding of the research goals and specific problems of the CFD engineers. Research meetings, feedback sessions, demonstrations, brainstorming and research seminars between the group have led to a successful working partnership for creating the visualization approach in this paper and other approaches [22]. Our flow engineer co-authors provided the following design principles:

The techniques should be interactive with low precomputation. Various techniques proposed in the literature (and see Section 5.2.3) frustrate due to the large start up time, or the inability to interact with their data in any systematic way. Whilst they may produce good images, the information derived from them can be quite often low due to the lack of interaction with a static view or slow update time.

Effective visual searching. If a visualization not only captures the flow domain, but also highlights unexpected behaviour this can massively aid the searching of large quantities of data, or even leads to locating features that would otherwise be overlooked. A corollary of this are that views should not suffer from occlusion and the user should be able to focus on specific flow features.

Mapping quantities, visual appearance. Visual appearance is difficult to quantify here. Many visualizations from the literature have been presented to the CFD engineers and, simply, the most attractive or beautiful visualizations always generate a great deal of interest. This could be technique, for example utilizing illuminated lines, opacity variation, through to clarity of color mappings and effective legends. Any parameter space exposed by the visualization should be simple to navigate.

Large aspect ratios. The engineering co-authors tend to work with domains that have large aspect ratio (for example a large area of shallow water for tidal generators, or large area wind farms).

Before this work, the CFD engineers chose the most interactive elements of the visualization software that is available to them. This primarily leads to examining large numbers of 2D color mapped contour plots and cognitively integrating features into 3D. This is because full domain seeding (regular sampling) leads to clutter, or the large domains are unsuitable for full domain algorithms due to speed issues. After this sub-optimal search, the engineer then uses a rake to generate streamlines in the region of interest. One of the big challenges is that either the seeding is dense and results cluttered, or the seeding is not dense enough and critical features are omitted. Finding the right balance is challenging. This is a particular problem of domains with large aspect ratio since streamline generation treats all regions with equal importance, whereas in such domains there are large areas of uninteresting flows with small areas (e.g. the flows around the turbine) where the flow is highly changeable and thus leads to visual clutter. For these domains, the context through the largely uninteresting areas with focus in the interesting regions would be most desireable. For aesthetic reasons, the engineers like evenly spaced streamlines, but for visually searching the domain they find the lack of interactive seeding control to be problematic. They resort to seeding rakes due to the excellent fine control they have over placement.

After our new technique was made available to the engineers, it was identified that it had an impact in all those areas. An example illustrating this is presented in the next section.

Additionally, our flow engineers have specific interest in tidal stream turbines which require enviromental permits from regulators, who require a clear understanding of the changes to flow caused by such devices. This streamline generation technique gives engineers the ability to fully control which streamlines are shown, allowing them to show general contextual flow in reduced opacity, and capture the salient flow around the turbines. The attractive visualizations, control over opacity and color mappings results in a visually desirable form that provides a strong clear context for discusion with regulators. This satisfied our demand for visual appearance, ability to deal with large aspect ratios, and also control and interactivity in order to produce such images.

5.2 Narrative

This section presents a narrative of the features of our algorithm with examples from standard data sets.

5.2.1 *Focus+context Visualization*

Our clustering strategy segregates the streamlines into groups with distinctive behavior. We provide a tool that allows the user to analyze the clusters using a focus+context visualization. This reduces visual clutter and aids in the analysis of the flow. The varying behavior of streamlines along the rake can be quickly and easily explored. The resulting visualization aids the presentation and communication of results by highlighting a particular flow behavior.

The user selects the cluster they wish to analyze, the streamlines belonging to this cluster are mapped to a high opacity for emphasis. This allows the user to focus on the chosen cluster within the context of the entire set of streamlines.

Figure 7 shows the focus+context visualization applied to sets of streamlines on the smoke plume simulation, with each row corresponding to a different seeding configuration. The left column of images show streamlines colored according to velocity magnitude. The streamlines exhibit a high-level of visual complexity. Even using transparency it is difficult to distinguish the different flow characteristics and how they interact with each other. The middle and right

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS

MCLOUGHLIN *et al.*: SIMILARITY MEASURES FOR ENHANCING INTERACTIVE STREAMLINE SEEDING

9

column images show the results of our clustering strategy with focus+context applied. Streamlines are colored according to cluster membership. In each image a different cluster is highlighted.

5.2.2 Streamline Filtering

A benefit of our approach is that it allows us to introduce streamline filtering to produce variable inter-seed distance along seeding objects. Typically streamlines are seeded at equidistant positions along the seeding curve or at regular points along a seeding plane. This results in our users having to specify a dense set of streamlines if they encounter complex flow behavior. The side-effect of this is that there are dense bundles of streamlines in regions where the flow is more uniform, which may lead to visual clutter. An ideal solution is to produce an expressive set of streamlines which captures all of the details of flow behavior while minimizing the redundancy in the visualization.

Streamline filtering is performed by creating cuts through the dendogram representing the clustering. The user is provided with a slider. As the user increases the value of the slider streamlines are removed from the visualization (they are simply not rendered, they do stay in memory).

Figure 8 shows our filtering strategy applied to a set of streamlines generated on a simulation of Arnold-Beltrami-Childress (ABC) flow. The top-left image shows the complete set of 200 streamlines. There is a high level of redundancy with the streamlines. The top-right image shows filtering removing approx 75% of the original set of streamlines. In the bottom-left image the number of streamlines is reduced to 13 using our method. The bottom-right image shows the result of a more uniform filtering strategy leaving the same number of streamlines. Using our method the remaining streamlines still depict helpful information about the flow characteristics. The uniform filtering loses important information.

Figure 9 shows a comparison between a dense set of streamlines and two sparser sets generated on the simulation of Hurricane Isabel. The left image shows the rake at a resolution of 200 streamlines. The middle image shows the results using our filtering method. Using filtering the expert reduced this number of streamlines down to just 12. Using only 12 streamlines they note our method still preserves the interesting flow characteristics – in particular the two regions of vortex behavior. The reduced number of streamlines produces a visualization that suffers from occlusion and visual complexity to a much lesser extent. Note that the controls are easy to use and we are able to produce the result in a matter of seconds. We were particularly interested in variable inter-seed distance along the rake, stating it allows the visualization to express more with less, and that is reduces visual information overload. The right image provides a comparison using equidistantly seeded streamlines. Note that the second vortex region is not visualized and that the separation regions are not as clearly defined.

5.2.3 Interaction

A high level of interactivity with the visualization is desirable. The parameter space should be kept as small as possible. It should also be simple and intuitive to navigate. To this end, our algorithm relies on only a few parameters:

- The number of clusters
- The weighting of the χ^2 term
- The number of streamlines to be filtered from a cluster (this is set on a per-cluster basis).

We utilize GUI slider widgets to control these parameters. As shown in Section 6 our algorithm provides interactive rates and changes to these parameters are displayed in real-time to the user. Fast response from parameter updates also aids the user in navigation, allowing them to quickly find a good set of values for the parameters.

All similarity and clustering algorithms have failure cases in which the end result may not match what a user expects with a fully manual clustering scheme. When using a scheme that has no input parameters such as the one by Corouge *et al.* [3], the user has no control over the final result. If the clustering proves inadequate the user cannot customize the visualization or has to resort to a different algorithm. In contrast, the algorithm of Chen *et al.* [2] does provide user-modifiable parameters. However, their algorithm is computationally expensive (see Section 6) and some parameters, such as window size, result in a complete re-computation – meaning the user has to wait for feedback from the application. Also the effect that a change in the user-parameters produces is unintuitive. This means that the user will have to perform a slow search through this parameter space using trial and error. The method of Zhang *et al.* [29] also requires a re-computation of all similarity distances when there is a change in the minimum distance threshold.

5.3 Unsteady Flow

Our method can be extended to unsteady flow. We compute the torsion and curvature fields for every time-step of the simulation. The algorithm then proceeds as outlined in the method overview (Figure 1). However, instead of streamlines, we trace pathlines. When the pathline attributes are computed, we use the field that matches the corresponding time of the pathline point. Where a pathline point does not lie exactly on a time-step we interpolate between the two closest fields. Figure 11 shows clustering results using our algorithm. In this figure the pathlines have been seeded using a seeding plane and are traced in the simulation of Hurricane Isabel.

6 PEFORMANCE ANALYSIS

Our approach is intended to give fast computation, good selection of similarity, leading to interactive and intuitive rake control. Streamline similarity is a mature research topic, but as mentioned in the related work section, all the measures involve performing a great number of distance tests between streamlines. In this section we compare our
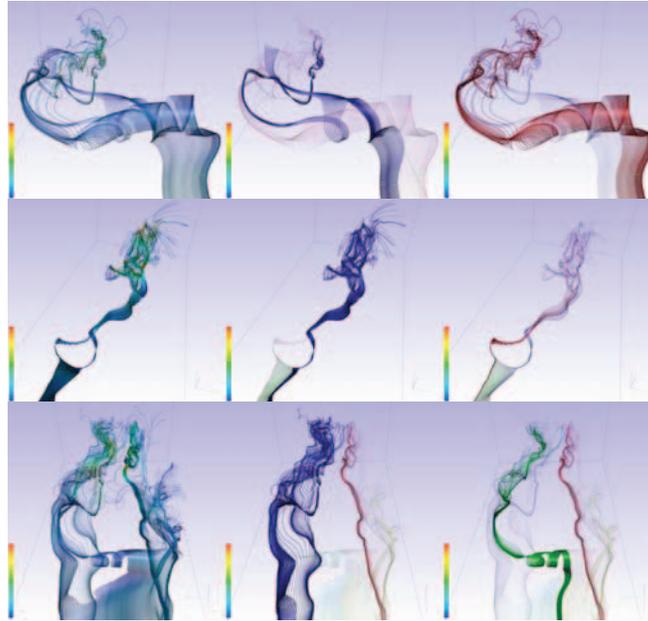
Fig. 7. Focus+context visualization of streamline clusters created using our method on the smoke plume simulation. Each row represents a different seeding configuration. Images in the left column show the set of streamlines colored according to velocity magnitude. The middle and right columns show the focus+context views, both setting different clusters as the focus. The selected cluster (the focus) is shown with a high opacity. The remaining clusters (context) are assigned a lower opacity. This allows the user to analyze each cluster more easily while retaining the context of the cluster in regards to the entire rake. This reduces visual complexity and reduces the effects of occlusion. The focus+context view allows the user to analyze the interactions between the streamlines in more detail.
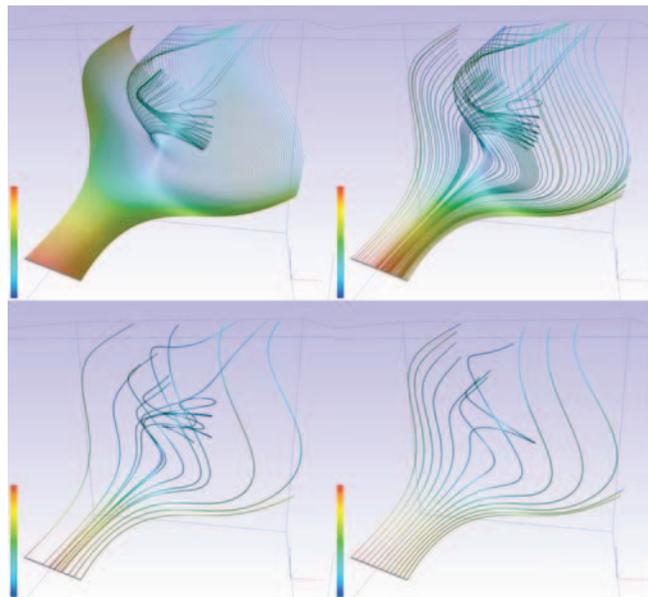

Fig. 8. Our filtering technique allows the user to filter out streamlines based on our similarity measures. The streamline that is most similar to the current set is iteratively removed – leaving the most dissimilar (and, hence, the most illustrative) streamlines. The top-left image shows the original dense set of 200 streamlines. The top-right image shows the our method filtering out approximately 75% of the initial streamlines. The bottom-left image shows filtering using our strategy to leave just 13 streamlines. Using our approach the few streamlines that remain depict the main characteristics of the superset. The bottom-right image shows the 13 remaining streamline using filtering to leave a more uniform distribution of the seeds. Important detail is lost when streamline similarity is not taken into account as shown in the bottom-right image.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS

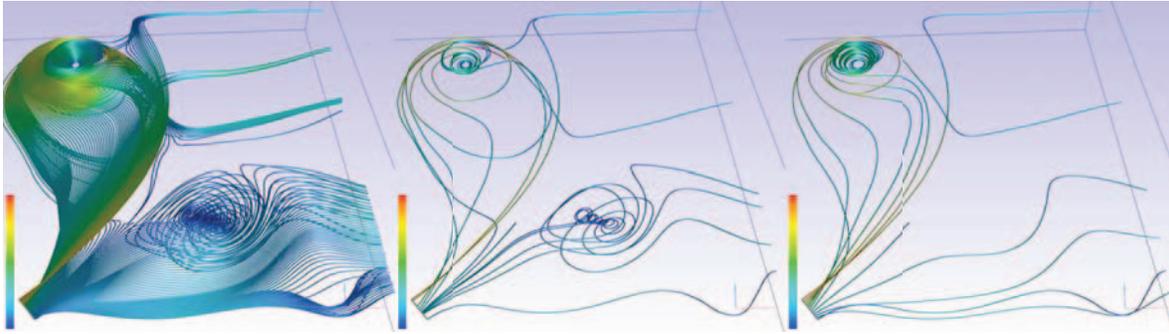MCLOUGHLIN *et al.*: SIMILARITY MEASURES FOR ENHANCING INTERACTIVE STREAMLINE SEEDING 11

Fig. 9. Our filtering strategy applied to the simulation of Hurricane Isabel. 4 clusters are selected and streamlines are filtered on a per-cluster basis. 200 streamlines in the left image are filtered down to 12. The middle image uses filtering based on similarity. The right image shows equidistant seeding of the streamlines. Our method better represents the saddle regions and preserves the second (smaller) vortex structure. The second vortex structure is entirely lost in the right image. Our preserves the overall structure represented through the rake and the most important characteristics.
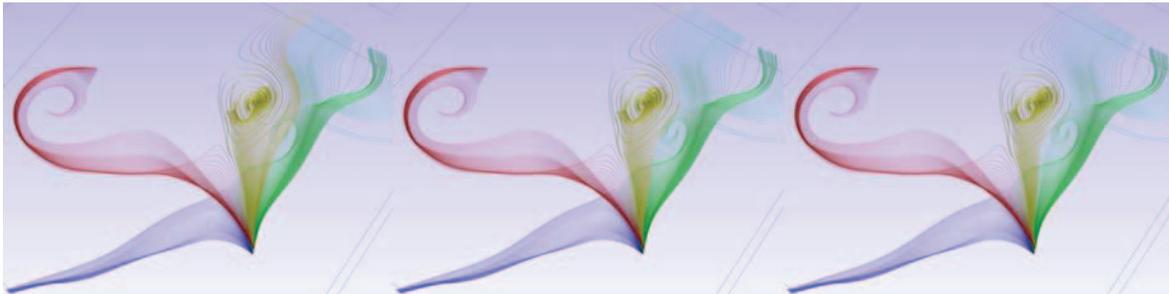

Fig. 10. Three sets of four clusters for the same set of streamlines. (Left) The clustering produced using the similarity metric from [3]. This method produces good quality clustering. However, a user may wish that the gold cluster contains only the streamlines that pass through the vortex region. This method provides no user-parameters for the user to tailor the result to their requirements. (Middle) The set of clusters resulting from our algorithm. Once again, the gold cluster contains streamlines that don't enter the vortex region. However, the user can very quickly fine tune the clustering and ensure that the cluster is constrained only to the streamlines that enter the vortex region (Right). Note that the clustering in the left image took just over 24s to produce. Our method produced the streamlines in just over 0.07s. Our weighting parameter space can be fully explored in just a few seconds. Therefore, the user can easily tailor the visualization to their needs.
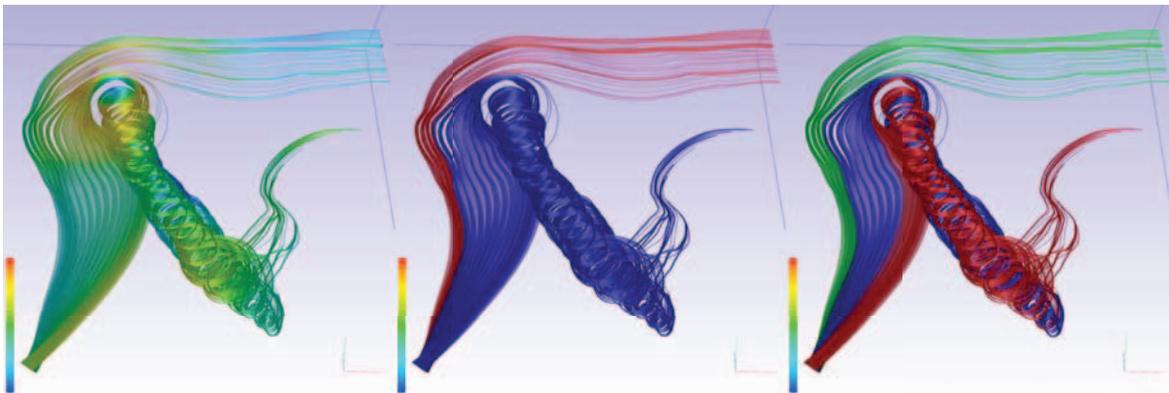

Fig. 11. Our method can be applied to unsteady flow. Torsion and curvature fields are generated for every time-step. Pathlines are then traced and the clustering algorithm is performed. This set of pathlines was generated on the simulation of Hurricane Isabel and seeded from a seeding plane. The left image shows the pathlines with color mapped to velocity magnitude. The middle and right images show the pathlines clustered in 2 and 3 clusters respectively.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS

MCLOUGHLIN *et al.*: SIMILARITY MEASURES FOR ENHANCING INTERACTIVE STREAMLINE SEEDING   12

Algorithm	Integration	Similarity & Clustering	Total	Factor
Our Method	0.017s	0.021s	0.048s	1.0x
Our Method (hierarchical)	0.017s	0.112s	0.129s	2.69x
Chen *et al.* [2]	0.017s	6.103s	6.120s	127.5x
Corouge *et al.* [3]	0.017s	5.260s	5.277s	109.94x
Zhang *et al.* [29]	0.017s	5.450s	5.467s	113.90x
Our Method	0.031s	0.042s	0.073s	1.0x
Our Method (hierarchical)	0.031s	0.267s	0.298s	4.08x
Chen *et al.* [2]	0.031s	22.400s	22.431s	307.27x
Corouge *et al.* [3]	0.031s	20.150s	20.181s	276.45x
Zhang *et al.* [29]	0.031s	20.210s	20.241s	227.27x
Our Method	0.062s	0.165s	0.227s	1.0x
Our Method (hierarchical)	0.062s	1.451s	0.102s	6.39x
Chen *et al.* [2]	0.062s	97.300s	97.362s	428.91x
Corouge *et al.* [3]	0.062s	82.050s	80.112s	352.92
Zhang *et al.* [29]	0.062s	84.010s	84.072s	370.36x

TABLE 2

Performance times of our algorithm in comparison with [2], [3] and [29]. The first column identifies the algorithm used. The second column shows the integration time for the streamlines. The similarity computation and clustering times are combined in the third column and the fourth column shows the total computation time. The final column shows the total computation times as a factor of our algorithm. The top, middle and bottom results were generated using 100, 200 and 400 streamlines respectively.

approach to state-of-the-art approaches for detecting similar integral curves. The distance measures we compare against are Corouge *et al.* [3] (equation 2), Zhang *et al.* [29] (Section 3.2) and Chen *et al.* [2] (Section 3).

Table 2 reports the performance times of our algorithm tested on a 2.4Ghz Intel Core 2 Quad CPU with 4GB RAM using a single thread. We compare our running times against algorithms by Chen *et al.* [2], Corouge *et al.* [3] and Zhang *et al.* [29]. The results in Table 2 are generated using 100, 200 and 400 streamlines, each consisting of up to 1000 points. We report streamline integration times in order to provide a context which to compare the clustering phase. The last column in the table gives the performance times as a factor of our method. In this scenario our algorithm takes 0.073 seconds to complete for 200 streamlines – providing interactive results. In contrast, the techniques of Chen *et al.* [2], Corouge *et al.* [3] and Zhang *et al.* [29] take more than 20 seconds to complete and are thus, prohibitively expensive for use as an interactive technique. As highlighted in the final column of the table, these algorithms can take over 300 times as long as our algorithm to compute. Figure 10 demonstrates that our method produces comparable results against the state-of-the-art. Importantly, our method affords the user the flexibility to modify the clustering results.

The vast majority of the computational workload in these algorithms stems from the large number of distance calculations to compute similarities. Our algorithm alleviates this by greatly reducing the number of distance tests. The small number of distance tests, coupled with our (less computationally expensive) χ^2 test on the binned streamline signatures, produces good clustering results at a fraction of the expense of pure distance-based similarity metrics. The seeding object type has very little effect on the performance times. The main influence is the number of streamlines used.

7 CONCLUSION

We present a tool for enhancing the user experience while interactively seeding streamlines. Streamlines can be clustered together and visualized using focus+context methods giving the user the opportunity to reduce visual complexity and target distinct flow behavior that they wish to investigate. The method also provides a filtering scheme to produce streamlines that are seeded at non-equidistant positions along the seeding object. This technique produces a set of streamlines that preserve the detail of the visualization while greatly reducing the number of streamlines. This is achieved by filtering out the most similar streamlines and preserving the least similar and hence most illustrative set for a given rake.

From domain expert interviews it was found that experts prefer intuitive tools that they can modify to meet their requirements. We reviewed previous similarity metrics and found that they were too computationally expensive to meet these requirements. Thus, we introduce the novel concept of the streamline signature. The streamline signature is produced from binned data that provides a distinct pattern for each streamline. We also employ the χ^2 test on the streamline signatures as a similarity measure. To the authors' knowledge this is the first time the χ^2 test has been used in this context. We also present an extension to our algorithm based on hierarchical signatures. This addresses limitations in the handling of shifted signatures when using a single signature. It also removes the burden of setting bin sizes on the user. We also provide a set of attributes that we found useful for the computation of the streamline signature. This is by no means an exhaustive list and further options are available for further research.

We demonstrate the performance of our algorithm compared to other similarity metrics and show that we can provide similar results one to two orders of magnitude faster. Finally, our tool allows the user to fine tune the end visualization quickly and easily in real-time – reducing the blackbox effect of an automatic algorithm and allowing for the user to correct fail cases should they arise.

The method, as it is presented here, is limited to the situation where we are seeding streamlines using rakes and planar objects. In the future we would like to investigate 2D attribute parameter spaces, allowing the user to investigate how one flow attribute changes with another attribute. We would also like to apply this method to DTI fiber bundling and further investigate the possibility of using our method as a fast 3D, full domain streamline seeding strategy.

REFERENCES

[1] TECPLOT. http://www.tecplot.com/, last accessed 2012-02-21.
[2] Y. Chen, J. D. Cohen, and J. Krolik. Similarity-Guided Streamline Placement with Error Evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1448–1455, 2007.
[3] I. Corouge, S. Gouttard, and G. Gerig. Towards a shape model of white matter fiber bundles using diffusion tensor MRI. In *International Symposium on Biomedical Imaging (ISBI)*, pages 344–347, 2004.
[4] R. Cucitore, M. Quadrio, and A. Baron. On the effectiveness and limitations of local criteria for the identification of a vortex. *European Journal of Mechanics - B/Fluids*, 18(2):261 – 282, 1999.

[5] J. Daniels, E. Anderson, L. Nonato, and C. Silva. Interactive vector field feature identification. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1560 –1568, nov.-dec. 2010.

[6] C. Demiralp and D. H. Laidlaw. Similarity coloring of DTI fiber tracts. In *Proceedings of DMFC Workshop at MICCAI*, 2009.

[7] E. Hadjidemetriou, M. Grossberg, and S. Nayar. Multiresolution histograms and their use for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7):831–847, july 2004.

[8] G. Haller. An objective definition of a vortex. *Journal of Fluid Mechanics*, 525:1–26, 2005.

[9] D. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, 2006.

[10] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, Sept. 1999.

[11] H. Jänicke, M. Böttinger, and G. Scheuermann. Brushing of attribute clouds for the visualization of multivariate data. *IEEE Transactions on Visualization and Computer Graphics*, (6):1459–1466, 2008.

[12] R. Jianu, C. Demiralp, and D. Laidlaw. Exploring 3D DTI fiber tracts with linked 2D representations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1449 –1456, 2009.

[13] B. Jobard and W. Lefer. Creating Evenly–Spaced Streamlines of Arbitrary Density. In *Proceedings of the Eurographics Workshop on Visualization in Scientific Computing '97*, volume 7, pages 45–55, 1997.

[14] L. Li, H. H. Hsien, and H. W. Shen. Illustrative streamline placement and visualization. In *IEEE Pacific Visualization Symposium 2008*, pages 79–86, 2008.

[15] Z. P. Liu and R. J. Moorhead, II. An Advanced Evenly-Spaced Streamline Placement Algorithm. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):965–972, Sep/Oct 2006.

[16] O. Mallo, R. Peikert, C. Sigg, and F. Sadlo. Illuminated Lines Revisited. In *Proceedings IEEE Visualization 2005*, pages 19–26, 2005.

[17] S. Marchesin, C.-K. Chen, C. Ho, and K.-L. Ma. View-dependent streamlines for 3D vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1578–1586, 2010.

[18] O. Mattausch, T. Theußl, H. Hauser, and E. Gröller. Strategies for Interactive Exploration of 3D Flow Using Evenly-Spaced Illuminated Streamlines. In *Proceedings of the 19th Spring Conference on Computer Graphics*, pages 213–222, 2003.

[19] T. McLoughlin, R. S. Laramee, R. Peikert, F. H. Post, and M. Chen. Over Two Decades of Integration-Based, Geometric Flow Visualization. *Computer Graphics Forum*, 29(6):1807–1829, 2010.

[20] A. Mebarki, P. Alliez, and O. Devillers. Farthest Point Seeding for Efficient Placement of Streamlines. In *Proceedings IEEE Visualization 2005*, pages 479–486, 2005.

[21] B. Moberts, A. Vilanova, and J. van Wijk. Evaluation of fiber clustering methods for diffusion tensor imaging. In *Proceedings IEEE Visualization 2005*, pages 65 – 72, 2005.

[22] Z. Peng, E. Grundy, R. S. Laramee, G. Chen, and N. Croft. Mesh-driven vector field clustering and visualization: An image-based approach. *IEEE Transactions on Visualization and Computer Graphics*, 18(2):283–298, 2012.

[23] M. Roth. *Automatic Extraction of Vortex Core Lines and Other Line-Type Features for Scientific Visualization*. PhD Dissertation No. 13673, ETH Zurich, 2000. published by Hartung-Gorre Verlag, Konstanz, ISBN 3-89649-582-8.

[24] T. Salzbrunn and G. Scheuermann. Streamline predicates. *IEEE Transactions on Visualization and Computer Graphics*, 12:1601–1612, 2006.

[25] K. Shi, H. Theisel, H. Hauser, T. Weinkauf, K. Matkovic, H.-C. Hege, and H.-P. Seidel. Path Line Attributes - an Information Visualization Approach to Analyzing the Dynamic Behavior of 3D Time-Dependent Flow Fields. In *Topology-Based Methods in Visualization II*, Springer series of Mathematics and Visualization, pages 75–88, Grimma, Germany, 2009. Springer.

[26] B. Spencer, R. S. Laramee, G. Chen, and E. Zhang. Evenly-spaced Streamlines for Surfaces. *Computer Graphics Forum*, 28(6):1618–1631, 2009.

[27] G. Turk and D. Banks. Image-Guided Streamline Placement. In *ACM SIGGRAPH 96 Conference Proceedings*, pages 453–460, Aug. 1996.

[28] H. Yu, C. Wang, C.-K. Shene, and J. Chen. Hierarchical streamline bundles. *"IEEE Transactions on Visualization and Computer Graphics"*, to appear, 2012.

[29] S. Zhang, S. Correia, and D. H. Laidlaw. Identifying white-matter fiber bundles in DTI data using an automated proximity-based fiber-clustering method. *IEEE Transactions on Visualization and Computer Graphics*, 14(5):1044–53, 2008.
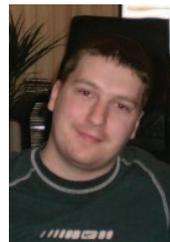
Tony McLoughlin received the BSc degree in Computer Science from Swansea University, UK in 2007. He stayed there to pursue his PhD in scientific visualization which was awarded to him in 2011.

Mark W. Jones received the BSc and PhD degrees from Swansea University, UK. He is now a reader in the Department of Computer Science, Swansea University. His research interests include ray tracing, global illumination, data visualization, volume graphics, voxelization and distance fields.

Robert S. Laramee received a bachelors degree in physics, cum laude, from the University of Massachusetts, Amherst in 1997. In 2000, he received a masters degree in computer science from the University of New Hampshire, Durham. He was awarded a PhD from the Vienna University of Technology, Austria at the Institute of Computer Graphics and Algorithms in 2005. His research interests are in the areas of scientific visualization and human computer interaction. He is now a Senior Lecturer at Swansea University in the Department of Computer Science

Rami Malki Rami Malki received an MEng degree in Civil and Environmental Engineering in 2004 and a PhD in River and Coastal Engineering in 2010, both from Cardiff University. He is currently a numerical modeller within the Marine Energy Research Group at Swansea University where his research interests include tidal stream turbine simulations.

Ian Masters Ian Masters gained a BSc in Engineering Mathematics in 1993 and a PhD in Finite Elements in 1997, both from the University of Wales. He is a Senior Lecturer in Mechanical Engineering at Swansea University. He is principal investigator for the Low Carbon Research Institute Marine Consortium. Research areas include: Design of tidal turbines with combined tide, wave and turbulent flows; CFD modelling of arrays, wakes, scour and deposition; socio-economic-environmental impact analysis of ocean energy.

Charles D. Hansen received a BS in computer science from Memphis State University in 1981 and a PhD in computer science from the University of Utah in 1987. He is a professor of computer science at the University of Utah and associate director of the SCI Institute. From 1989 to 1997, he was a Technical Staff Member in the Advanced Computing Laboratory (ACL) located at Los Alamos National Laboratory, where he formed and directed the visualization efforts in the ACL. He was a Bourse de Chateaubriand PostDoc Fellow at INRIA, Rocquencourt France, in 1987 and 1988. His research interests include large-scale scientific visualization and computer graphics.