Evaluation of Stream Surfaces Using Error Quantification Metrics

Ayan Biswas[a] and Han-Wei Shen[a]

[a]The Ohio State University, 2015 Neil Avenue, Columbus, Ohio, USA

ABSTRACT

Visualizing stream surfaces in three-dimensional flow fields is a popular flow visualization method for its ability to depict flow structures with better depth cues compared to simply rendering a large number of streamlines. Computing stream surfaces accurately, however, is non-trivial since the result can be sensitive to multiple factors such as the accuracy of numerical integration, placement of sampling seeds, and tessellation of sample points to generate high quality polygonal meshes. To date, there exist multiple stream surface generation algorithms but verification and evaluation of the quality of the stream surfaces remain an open area of research. In this paper we address this issue, propose different stream surface evaluation metrics and study different aspects of stream surface generation process like choice of algorithms, seeding curve placement, initial seeding curve density, choice of algorithm parameters with four verification metrics to reach meaningful conclusions.

Keywords: Stream Surfaces, Error Quantification, Error Metrics

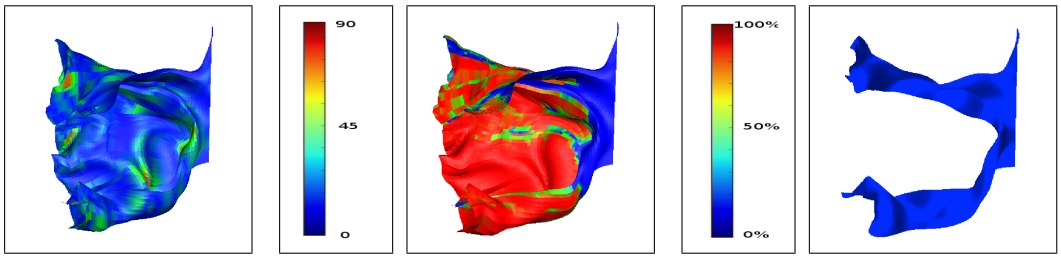
1. INTRODUCTION

Streamlines and stream surfaces are extensively used for visualization of flow fields. A streamline is traced out from a given seed point and integrated over the flow field so that it is everywhere tangent to the flow field. Streamlines are intuitive, simple to calculate, and they reveal the flow characteristics quite well. Stream surfaces are extensions of streamlines, where the normal direction of any point on the stream surface is perpendicular to the underlying flow field. As they provide better 3D depth cues, these surfaces can be more powerful in revealing flow features, although they are more prone to issues like occlusion. Compared to the computation of streamlines, generation of stream surfaces is less straightforward and is more susceptible to discretization errors. To date, there exist several algorithms designed to address various issues to improve the quality of stream surfaces. While the relative difficulty in construction of stream surfaces has led researchers to look for algorithms[1,2] which are less complex, generating a high quality surface has remained the basic requirement of any stream surface generation algorithm.

Showing error and uncertainty in the visualizations have become a very common practice as it can guide the user towards a better understanding of the dataset. Although stream surface is a popular visualization tool among scientists, a thorough discussion of the stream surface errors is mostly lacking and the need of this sort of works is ubiquitous. Figure 1(a)-(e) can be used as one of the motivations for coming up with the error metrics for stream surfaces. Ignoring the color mapping on the surfaces, if the user only looks at the stream surface as shown in Figure 1(a) and Figure 1(c), the user may not know that this stream surface has large erroneous regions and the topology of the surface is not correct. In this example, Figure 1(c) shows the erroneous regions and Figure 1(e) shows the correct topology for the surface. Using the metrics proposed in this work, we try to quantify the geometric errors present in the stream surfaces.

Experiments with stream surfaces involve exploration of a parameter space which has many dimensions. There are many algorithms available at this moment and choosing one of them is just one of the dimensions. Then, choosing where to place the initial seeding curve is another very important aspect as the experimental results show that, depending on where the stream surface is seeded, different algorithms behave differently. The next decision problem is the seeding curve sampling density which affects the final output from different algorithms in a different way. After these three steps, the user needs to decide the parameters used by the stream surface algorithms for finally generating the surface. In this exploration process, the need for stream surface quality metrics is self-evident although this kind of work is mostly lacking up to this point. In this paper, we put forward four different methods to evaluate the stream surfaces quantitatively. With the

Further author information: (Send correspondence to A.B.)
A.B.: E-mail: biswas.36@osu.edu, Telephone: 1 614 270 8216
H.S.: E-mail: hwshen@cse.ohio-state.edu, Telephone: 1 614 292 0060



(a) (b) (c) (d) (e)

Figure 1. (a) Sampled local error visualization for a stream surface. (b) Color map shows the average angle deviation (in degrees) for the surface patches. (c) Sampled global error visualization for the same stream surface. This metric reveals the erroneous regions not shown by the local error metric. (d) Color map shows the percentage of the sample points which miss the seeding curve by more than a threshold value when traced back. (e) Streamline simulation of the stream surface with the same seeding curve as (a) and (c).

metrics, we explore the different dimensions attached to the stream surface generation process with three popular stream surface generation algorithms.With the proposed error metrics, if one is attempting to come up with a new stream surface generation algorithm, it also becomes much easier to verify the behavior of the new algorithm and refine the algorithm accordingly. To the best of our knowledge, this is one of the very few papers that discuss stream surface verification with error quantification and visualization.

The rest of the paper is divided as follows: in Section 2 we review the related works, in Section 3 we discuss about the basics of stream surfaces, in Section 4 we propose four stream surface verification techniques, in Section 5 we discuss the results in light of our verification methods, and in Section 6, we give the conclusion and future work.

2. RELATED WORKS

In this section, we provide a brief review about the areas of research which are directly related to the topic of this paper: the stream surface construction methods, seeding curve placement strategies for stream surfaces, and the analysis and visualization of uncertainty and error in flow field data.

A very well known algorithm on stream surface computation was given by Hultquist.[3] The algorithm performs a triangulation of the neighboring streamlines which are originating from the seeding curve. Depending on the divergence or convergence of these neighboring streamlines, a seed is inserted or removed. Garth et al.[4] proposed an improvement on this algorithm where the surface enters complex areas of flow by using a higher order integration method, parameterization of the arc length, and inserting seeds by detecting high curvature in the neighboring ribbons. Van Wijk[5] proposed an implicit stream surface computation algorithm using the isosurface of stream function. An improved implicit stream surface algorithm was provided by Stöter et al.[6] where the authors used flow maps to compute implicit stream surfaces for arbitrarily defined seeding curves. Scheuermann et al.[7] presented a method which constructs stream surfaces for tetrahedral grids by propagating the surfaces through the grids. Schafhitzel et al.[1] and McLoughlin et al.[2] proposed ways to construct stream surfaces using less complex data structures. Schneider et al.[8] have proposed the use of bicubic patches to produce smoother curves of fourth order precision to increase the precision of stream surfaces. To convey the properties of the vector field in stream surface visualization, Laramee et al.[9] have provided a hybrid visualization method that uses texture advection on the stream surfaces. In a recent work, Schulze et al.[10] have proposed a stream surface generation scheme by forcing the front line of the surface to be perpendicular to the flow by minimization of an error function and thus producing well-shaped mesh elements in turbulent flow regions. In a recent survey by Edmunds et al.,[11] the surface-based flow visualization methods are discussed with their applicability and relative strengths and drawbacks.

Despite being an important part of the stream surface generation algorithms, the error and uncertainty quantification metrics for stream surfaces are mostly missing. Garth et al.[12] have taken initiative to verify the accuracy of the generated stream surfaces where they have deployed a collection of streamlines as the ground truth. Schneider et al.[13] have proposed a way to calculate a stream surface more accurately when it approaches a critical point. For uncertainty and error analysis in flow fields, Pang et al.[14] have given a detailed description of the sources of errors and presented several uncertainty based flow visualization techniques. In a recent paper by Pöthkow et al.,[15] a comprehensive list of references about uncertainty and error visualization is provided, along with discussions of positional uncertainty for isosurfaces.

3. STREAM SURFACE AND ERROR ESTIMATION

Stream surfaces are surfaces where all points are always tangential to the flow field. This is formally described[3] as a two dimensional parametric surface which is embedded in three dimensional flow. This parameterization can be done along parameters s and t. Using the parameter $s \in [0,1]$, we can parameterize the streamlines according to their seed locations. The second parameterization is done for time $t \in [t_0, t_{max}]$ along every streamline. A constant s curve is a streamline and a constant t curve is a timeline. A stream surface basically consists of an infinite number of streamlines which are seeded from the seeding curves. When calculating stream surfaces, we assume that the flow field is time-invariant. If the flow field is time-varying, the surface is called a path surface, which is a natural extension of a stream surface.

There exist several stream surface generation algorithms, and we select the following three algorithms as the representative ones to study: Hultquist's method,[3] Garth et al.'s method[4] and McLoughlin et al.'s method.[2] Hultquist's algorithm is one of the earliest stream surface algorithms and most other stream surface algorithms are based on it. Garth's algorithm is generally regarded as the best stream surface algorithm for handling high curvature flows. McLoughlin's method generates quads instead of triangles, and is relatively easy to implement.

When streamlines are constructed, the main source of uncertainty comes from the error incurred in the numerical integration step. Depending on the integration scheme in use, we can have a theoretical bound on the error. Since a stream surface is ideally thought of as a collection of streamlines, this integration error is present also in stream surfaces. The other important factor is the sampling decision. Generally, stream surface algorithms start computing streamlines from points on the seeding curve. This seeding curve sampling density is normally not enough, so the algorithms insert new seeds as the surface is being constructed. As described in the three algorithms above, the insertion of new seeds is performed based on heuristics. Since there is no guarantee that these new seeds are actually originating from the seeding curve, there are chances of more error being introduced in the surface. We call this phenomenon the uncertainty due to insufficient sampling of the seeding curve. Here we assume that the use of high order Runge-Kutta methods makes the integration error quite small. We are primarily concerned with capturing the error due to insufficient sampling of the seeding curve and its effect over the generated stream surface. The metrics to quantify error are lacking, although the need for this is self-evident.

4. VERIFICATION METHODS FOR STREAM SURFACES

4.1 Sampled Local Error Method

Most of the existing stream surface generation techniques produce a triangle/quad mesh as the output, which is then used for rendering. From the stream surface definition, all the points in a given surface patch should be tangential to the underlying flow, or in other words, the surface normal of that given patch should be perpendicular to the flow direction. This gives a way of calculating the error in a stream surface patch. The idea is to sample the given stream surface patch, find N sample points randomly distributed over the region, and then for each point, evaluate the dot product of the unit surface normal and the unit flow vector. Since for a perfectly error free region, the values should be all zeros, we can look at the distribution of the dot product values to determine the quality of the patch. We call this error estimation technique "local" as this error metric only measures the quality of the given surface patch in that region of flow. If the stream surface has deviated from the ideal location starting from the seeding curve, this method may not detect that error.

For algorithms which generate triangular meshes, finding the surface normal is relatively straightforward. On the other hand, the quads generated by stream surface algorithms, are not guaranteed to be coplanar, e.g. in case of a complex flow region, McLoughlin et al.'s method produces quads whose four vertices are not necessarily in the same plane. If four points of a quad are not coplanar, then an equation of the plane, which describes the quad, cannot be found and hence generating samples on the quad would be less straightforward. For this work, each sample point within the quad was found by using two random numbers which were used for bilinear interpolation between four vertices. To get the normal at the sample point, we first calculate the vertex normal per quad vertex, and then apply bilinear interpolation to calculate the normal at this sample point from the vertex normals. The same two random numbers were also used here for bilinear interpolation. Then the dot product can be computed. A local error visualization is shown in the Figure 1(a)-(b). Here the error is measured as the average angle deviation of the surface patches and is color mapped. For any algorithm that produces triangular or quad mesh, this method can be used to visualize the local error.

4.2 Sampled Global Error Method

The previous method suffers from the drawback that if the stream surface has drifted off from the ideal stream surface, the local metric will not detect it as it only checks whether the surface patch is correct with respect to the local flow. A global error check enforces that the surface patch is locally correct and it originates from the given seeding curve. Our global error calculation method finds sample points on a patch of the surface, and backtraces them to find out how many of these samples go back to the seeding curve. If the sample point of the generated surface results in a streamline connecting it to the seeding curve, then it can be assumed that the sample point is of good quality. If we assume that the integration error in the streamline computation is small, then this method yields a good verification result.

The process of finding the sample points on the mesh remains the same as the method described above. While sampling the quads/triangles, the number of forward integrations done up to that point, T, is known. For integration step size S, initially the sample points are backtraced $T-1$ steps and then it is backtraced M times with step size S/M and the minimum distance obtained at any step, is stored. M is set to 20 when S takes the value 1.0 voxel size. These minimum distance values from sample points are gathered to get a distribution and the mean and standard deviation of this distribution of values should be close to 0 for a good quality surface. For quads, this backtracing can be sped up by using the (s,t) parameterization associated with each sample point where the parameterization can be used to predict the integration step size within the quad. Figure 1(c)-(d) shows a global error study for the stream surface shown in Figure 1(a)-(b). It is observed that, in the particular case shown in Figure 1(a)-(d), the local metric was not able to capture the erroneous regions of the surface but the global metric successfully highlighted the error.

4.3 Densely Seeded Streamlines Method

Conceptually, a stream surface can be thought of as a collection of stream lines which are seeded infinitely densely from the original seeding curve. But practically it is impossible to sample the seeding curve infinitely densely. We simulate this by putting the seeds very densely on the seeding curve. Without considering numerical integration error, this collection of streamlines should be close to the ideal stream surface. Figure 1(e) shows a streamline simulation of the stream surface whose initial seeding curve is placed at the same region as the ones shown in Figure 1(a) and 1(c). It is evident that it shows the region that the stream surface should be passing through compared to the previously generated stream surface which had erroneous regions. This method is similar to the work of Garth et al.[12] with an important difference. In their work, the authors assumed the collection of the streamlines to be the ground truth but our experiments involving the datasets having complex flows, reveal that it is not always safe to assume that the collection of streamlines will be representing the true surface. If the number of integration steps is large and the underlying flow is very divergent, then finding out the suitable seeding curve density is non-trivial as the neighboring streamlines will start to diverge. Given this fact, we take this collection of streamlines as a subset of the true stream surface and compare with the surfaces generated by the algorithms.

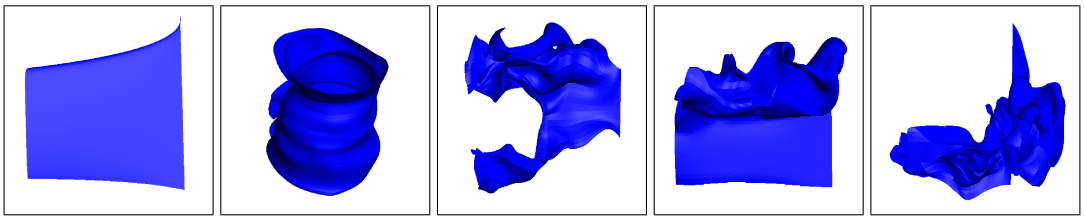
Comparison of the two stream surfaces, one generated from the densely seeded streamlines and one from an existing stream surface algorithm, is also non-trivial. In this work, the Hausdorff distance[16] is used to measure the difference between the two surfaces. Given two non-empty sets P and Q, the Hausdorff distance between them $d_H(P,Q)$ is defined as

$$d_H(P,Q) = max(\hat{d}(P,Q),\hat{d}(Q,P)),\ where\ \ \hat{d}(P,Q) = \max_{x\in P}\min_{y\in Q}||x-y||. \tag{1}$$

Specific to this case, $||x-y||$ is taken as the Euclidean distance and one sided Hausdorff distance from the streamline set to the algorithm generated stream surface is used. The Hausdorff distance calculation was sped up using GPUs and the comparison of two sets containing 1M points can be done within 10 seconds. A small value of this metric will indicate a high quality stream surface. But if the Hausdorff distance value is large, then it may be difficult to compare the results of two algorithms because the large distance may be caused by a small number of outliers in the sample. To avoid this, we look at the distribution of the minimum distance values and make a conclusion about the performance of the algorithms.

4.4 All Vertex Backward Tracing Method

No matter how densely the seeding curve is sampled, in the case of a complex flow, there will be cases where the neighboring streamlines will start to diverge. This scenario is also encountered in forward stream surface generation algorithms which is handled by putting more seeds in between two neighboring streamlines. Since these seeds are not guaranteed to be coming from the seeding curve and, even with many streamlines, there are chances that some features might still



(a) (b) (c) (d) (e)

Figure 2. Five stream surfaces showing different cases: a) StreamSurface1 showing straight areas of the flow, b) StreamSurface2 showing the vortex structure and high curvature areas of the flow, c) StreamSurface3 showing a general complex stream surface, d) StreamSurface4 has complex flow in the top part and straight flow in the lower region, e) StreamSurface5 is seeded in more turbulent region.

be missed. We can avoid this by doing backward integration from all points in space. All point backward tracing was previously employed by Van Wijk[5] in his implicit stream surface computation. The natural discretization of the flow field is given in the form of grid points. A way of checking the quality of a stream surface would be to trace out streamlines in backward direction from all grid points. The points which trace back to the seeding curve are the points which should be included in the stream surface. As we cannot sample the flow field infinitely, every grid point can be checked to find out the closest distance of that backtraced point to the seeding curve. As, the number of forward integration steps is known for a given stream surface, each grid can be backtraced those many steps and any grid point that comes closer than a predetermined threshold value while backtracing, will be included in a set that stores the candidate grid points to be included in the stream surface. The threshold can be set to a small value like 0.5 or 1.0. This way, a new collection of points will be formed. The one sided Hausdorff distance can be calculated between this point set and the generated stream surface point set. If the resulting value is low, then the generated stream surface will be considered to be accurate. As the grid resolution is increased, this metric will perform better in estimating the quality.

5. RESULTS AND DISCUSSIONS

The experiments were conducted on a Linux machine with the Intel core i7-2600 CPU and 16 GB of RAM. The two datasets used were the Plume and the Isabel. The Plume dataset is a simulation of the thermal downflow plumes on the surface of the sun and it has $126 \times 126 \times 512$ grid points. The dataset Isabel is a hurricane simulation with a resolution of $500 \times 500 \times 100$. To illustrate different seeding curve locations, five representative stream surfaces have been chosen, as presented in Figure 2(a)-(e). StreamSurface1 represents a very straight flow region of the field from the Plume dataset generated by integrating 100 steps, StreamSurface2 is the hurricane eye from the Isabel dataset, generated by integrating 150 steps, and is representative of the vortex like structures which is quite common in flow fields, StreamSurface3 represents a general stream surface from the Plume dataset, generated by 100 integration steps, which has a complex underlying flow, StreamSurface4 has both complex and straight flow regions from the Plume dataset and it was generated by 100 integration steps, and StreamSurface5 is seeded in a complex region and has a very complex structure from the Plume dataset and this was generated by 50 integration steps. We implemented the three stream surface algorithms for our experiments and adaptive Runge-Kutta Four-Five integration scheme was used for all three algorithms. The datasets were also normalized so that these algorithms could be compared on a common base.

5.1 Effect of the Choice of the Stream Surface Generation Algorithms and Placement of Initial Seeding Curve

In this section, the effect of choosing the stream surface algorithm for different initial seeding curve placements is presented. The three algorithms are tested against the five stream surfaces based on the four error metrics proposed in this paper. The evaluation based on the four different metrics is presented separately in the following sections.

5.1.1 Results for the Sampled Local Error Method

In Table 1, we show the result of the application of the sampled local error metric as described in Section 4.1. The average angle deviation for each triangle/quad was measured by the average dot product value of each triangle/quad and then the final weighted average was calculated by taking the areas of the triangle/quad as the weights. Finally, the result is expressed in degrees . The area was used as the weights while averaging since different algorithms produce different numbers of triangles/quads. For each quad/triangle, 60 random samples have been used.

Table 1. Results for the Sampled Local Error Method and the Sampled Global Error Method.

	StreamSurface1		StreamSurface2		StreamSurface3		StreamSurface4		StreamSurface5	
	Local Error	Global Error	Local Error	Global Error	Local Error	Global Error	Local Error	Global Error	Local Error	Global Error
Hultquist	0.15	0.03	3.02	0.072	8.51	2.17	6.18	3.54	14.51	4.81
Garth	0.15	0.03	2.74	0.055	6.79	1.67	5.42	2.95	12.94	3.51
Quad	0.15	0.02	2.91	0.062	8.17	1.82	7.54	4.1	13.22	3.71

Table 2. Results for the Densely Seeded Streamlines Method. H-dis, Avg. and Stddev. stand for Hausdorff Distance, Average and Standard Deviation respectively.

	StreamSurface1			StreamSurface2			StreamSurface3			StreamSurface4			StreamSurface5		
	H-dis	Avg.	Stddev.	H-dis	Avg.	Stddev.	H-dis	Avg.	Stddev.	H-dis	Avg.	Stddev.	H-dis	Avg.	Stddev.
Hultquist	2.01	0.27	0.09	5.51	1.32	0.97	19.33	0.93	1.24	17.72	2.57	2.01	14.39	1.85	1.91
Garth	2.01	0.27	0.09	4.02	0.77	0.35	13.75	0.61	0.97	16.77	2.09	1.72	10.59	1.12	1.38
Quad	2.01	0.28	0.10	5.26	1.26	0.89	18.87	0.81	1.14	17.89	2.95	2.45	12.41	1.44	1.78

5.1.2 Results for the Sampled Global Error Method

In the Table 1, we also represent the results of the application of the second metric, the sampled global error metric, as described in Section 4.2, for the five chosen surfaces. We collect the closest distances of all the sample points from the seeding curve after the sample points are backtraced. The final average value is also weighted by the area of each triangle/quad. For each quad/triangle, 60 random samples have been used. These results are quite similar to the results given by the previous method and it is seen that Hultquist's algorithm gives worse values when compared to two other metrics in complex flow areas in general. But, as shown by StreamSurface4, it can be better than quad algorithm in certain situations.

5.1.3 Results for the Densely Seeded Streamlines Method

In this section, we discuss the results of the three stream surface algorithms when compared against a point cloud which is created using the set of streamlines by sampling the seeding curve very densely. Now, with this point set, the outputs of the three algorithms are compared, and the minimum distances are collected for each point of the point set. The one sided Hausdorff distance is calculated as the maximum value by which the points on the streamlines differ from the stream surface points. Also, the mean and standard deviation of these minimum value distribution are collected and the complete result is shown in Table 2.

5.1.4 Results for the All Vertex Backward Tracing Method

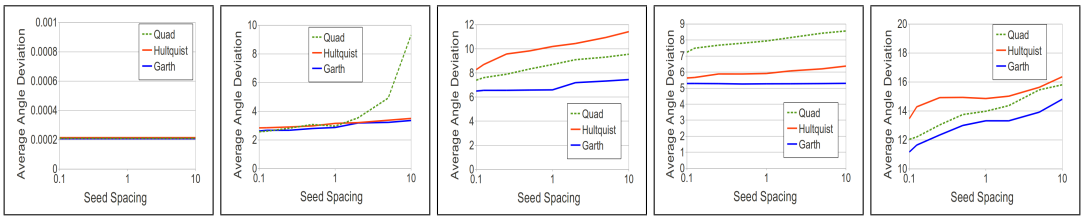
Here we discuss the results of the application of all vertex backtracing method. From all grid points of the flow field data, a streamline is traced back and at each step, it is checked to see how close it gets to the seeding curve. The cut off threshold distance is kept as 0.75 and whenever a backward streamline gets any closer to the seeding curve than the cutoff distance, the grid point is marked. All these marked points form the point set that gives the rough outline of the ideal stream surface. From these points, we find the minimum distances to the point set generated by the existing stream surface algorithms. The one sided Hausdorff distance, mean, and standard deviation are listed in Table 3.

5.2 Effects of Varying Seeding Density of the Seeding Curve

In this section, initiative is taken to observe how the initial seeding curve sampling density influences different stream surface algorithms. Experiments have been conducted by varying the seeding density from one seed every 10 voxels to one seed every 0.1 voxel size for the previously mentioned five stream surfaces. In this case, as all the error metrics are

Table 3. Results for the All Vertex Backtracking Method. H-dis, Avg. and Stddev. stand for Hausdorff Distance, Average and Standard Deviation respectively.

	StreamSurface1			StreamSurface2			StreamSurface3			StreamSurface4			StreamSurface5		
	H-dis	Avg.	Stddev.	H-dis	Avg.	Stddev.	H-dis	Avg.	Stddev.	H-dis	Avg.	Stddev.	H-dis	Avg.	Stddev.
Hultquist	1.69	0.38	0.22	8.31	0.85	0.79	17.52	1.81	2.52	19.94	4.92	4.03	13.67	1.29	1.3
Garth	1.69	0.38	0.22	6.68	0.44	0.61	13.75	1.18	1.15	19.44	4.78	3.97	9.81	0.95	0.75
Quad	1.69	0.34	0.24	7.72	0.71	0.78	16.38	1.26	1.33	20.13	5.31	4.12	10.21	1.12	1.23



(a) (b) (c) (d) (e)

Figure 3. Effect of initial seeding curve sampling density on the five stream surfaces for the three stream surface generation algorithms.

showing the same trend, the sampled local error metric is chosen to show the results in Figure 3(a)-(e). It is observed that, if the stream surface is very flat, similar to StreamSurface1, seeding density does not affect the quality that much for all the three algorithms. If the stream surface gets complicated gradually, as StreamSurface3 and StreamSurface4, then the algorithms get the chance to dynamically adjust the seeding density by adding new seeds in the first few steps. In this case, although the amount of error increases as the seeding density decreases, this increase is quite moderate as shown in Figure 3(c)- (d). StreamSurface5 was seeded in a very complex flow region, and although the algorithms tried to modify the seeding density in the first few steps, still the error incurred in those steps were quite high and the effect of sparse seeding is evident in Figure 3(e). Figure 3(b) is the case where quad algorithm performs really bad as the seeding density is decreased as compared to the other two algorithms. In this case, quad algorithm is not able to generate new seeds which other two algorithms could because quad algorithm's inner angle criterion is not satisfied while the surface generation and large quads have been produced.

5.3 Effect of Parameter Choice of Algorithms

In this section, we investigate the effect of parameters for the three algorithms using the sampled local error metric. In Hultquist's algorithm, the new seed insertion is done based on the ratio of the width and height of a quadrilateral while generating the mesh. Figure 4(a) shows the variation in quality for different choice of this width-to-height ratio threshold. For StreamSurface1 and StreamSurface2, there is little variation in quality and number of triangles generated remain almost the same except at a very low ratio, unnecessary triangles are generated which do not aid in improving the quality that much. For StreamSurface3 and StreamSurface4, as the ratio threshold is decreased from 3.5 to close to 1, quality increases but also the triangle count goes up around 5 times from around 30K triangles to 150K triangles in both the cases. For StreamSurface5, the increase in triangle count is around 50 times from 33K to 1700K.

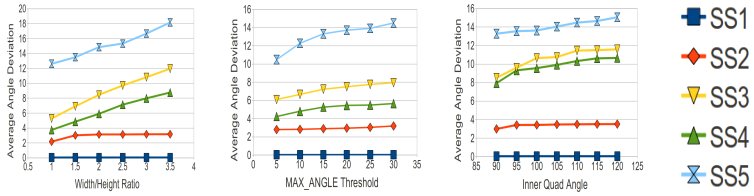


Figure 4. Effect of the user defined parameters: (a) Different values of widht-to-height ratio, used as the criterion for new seed insertion in Hultquist's algorithm. (b) Different Maximum Angle Threshold for Garth's algorithm. (c) Different inner angle threshold, used for detecting divergence in quad algorithm. (d) Legend

In Garth's algorithm, there are two user defined parameters, namely the maximum angle threshold and minimum angle threshold which govern seed insertion and deletion. Here, we chose to vary the maximum angle threshold from 5 degrees to 30 degrees keeping the minimum angle threshold fixed at 2 degrees. The effect of this parameter is shown in Figure 4(b). It is observed that the quality of StreamSurface1 and StreamSurface2 remain almost unaffected by the change of this parameter although the triangle count varies from 25K to 1800K for StreamSurface2 when the angle threshold varies from 30 degrees to 5 degrees. StreamSurface3 and StreamSurface4 show that as the parameter is decreased, the quality increases with an increase of triangle count from 66K to 1150K and 61K to 1200K for the two surfaces respectively. StreamSurface5 shows the most variability with this parameter. In this case, the triangle count variation is 360K to 16300K.

For quad algorithm, the split criterion is studied by changing the threshold angle from 120 degrees to close to 90 degrees and the result is shown in Figure 4(c). For StreamSurface1 and StreamSurface2, the variation in quality is relatively small and also the quad count remains almost the same for the two cases. For StreamSurface3 and StreamSurface4, the quad count is almost doubled from around 6K to 13K for both the cases as the threshold is changed. For StreamSurface5, the

change in quad count relatively less compared to the previous two surfaces. The quad count changes from 6.5K to 9.5K and also, the quality does not show a very big variation for different threshold values.

6. CONCLUSION AND FUTURE WORK

We have presented four different stream surface verification methods to study various aspects of error involved in stream surface computation. While it is difficult to single out one specific algorithm that always gives the best stream surface in terms of quality, computation time and space utilization, the quad algorithm is the fastest and most space efficient, easy to implement and generally its stream surfaces are of good quality with some exceptions. We have also confirmed that the algorithm by Garth et al. can consistently produce good stream surfaces, albeit at higher computation cost. As an application of these metrics, users would be able to modify the parameter values of an algorithm by looking at the error amount and checking whether the error is decreased after the modification. As a next step, we plan to extend this work for time-varying flow fields and streak surfaces, and include more stream surface generation algorithms for study. More knowledge about different stream surface generation algorithms can lead us to a new stream surface algorithm where we start the stream surface generation with a less complex algorithm and switch to a more accurate and time consuming one when error is increased.

REFERENCES

[1] Schafhitzel, T., Tejada, E., Weiskopf, D., and Ertl, T., "Point-based stream surfaces and path surfaces," in [*Proceedings of Graphics Interface 2007*], *GI '07*, 289–296 (2007).

[2] McLoughlin, T., Laramee, R. S., and Zhang, E., "Easy integral surfaces: a fast, quad-based stream and path surface algorithm," in [*Proceedings of the 2009 Computer Graphics International Conference*], *CGI '09*, 73–82 (2009).

[3] Hultquist, J. P. M., "Constructing stream surfaces in steady 3d vector fields," in [*Proceedings of the 3rd conference on Visualization '92*], *VIS '92*, 171–178 (1992).

[4] Garth, C., Tricoche, X., Salzbrunn, T., Bobach, T., and Scheuermann, G., "Surface techniques for vortex visualization," in [*VisSym 2004, Symposium on Visualization, Konstanz, Germany, May 19-21, 2004*], 155 (2004).

[5] van Wijk, J. J., "Implicit stream surfaces," in [*Proceedings of the 4th conference on Visualization '93*], *VIS '93*, 245–252, IEEE Computer Society, Washington, DC, USA (1993).

[6] Stöter, T., Weinkauf, T., Seidel, H.-P., and Theisel, H., "Implicit integral surfaces," in [*Proc. Vision, Modeling and Visualization*], to appear (November 2012).

[7] Scheuermann, G., Bobach, T., Hagen, H., Mahrous, K., Hamann, B., Joy, K. I., and Kollmann, W., "A tetrahedra-based stream surface algorithm," in [*Proceedings of IEEE Visualization 2001*], 83–91 (2001).

[8] Schneider, D., Wiebel, A., and Scheuermann, G., "Smooth stream surfaces of fourth order precision," *Comput. Graph. Forum* **28**(3), 871–878 (2009).

[9] Laramee, R. S., Garth, C., Schneider, J., and Hauser, H., "Texture advection on stream surfaces: a novel hybrid visualization applied to cfd simulation results," in [*Proceedings of the Eighth Joint Eurographics / IEEE VGTC conference on Visualization*], *EUROVIS'06*, 155–162 (2006).

[10] Schulze, M., Germer, T., Rössl, C., and Theisel, H., "Stream surface parametrization by flow-orthogonal front lines," *Computer Graphics Forum (Proc. SGP)* **31**(5), 1725–1734 (2012).

[11] Edmunds, M., Laramee, R. S., Chen, G., Max, N., Zhang, E., and Ware, C., "Surface-based flow visualization," *Computers & Graphics* **36**, 974–990 (2012).

[12] Garth, C., Krishnan, H., Tricoche, X., Bobach, T., and Joy, K., "Generation of accurate integral surfaces in time-dependent vector fields," *Visualization and Computer Graphics, IEEE Transactions on* **14**(6), 1404 –1411 (2008).

[13] Schneider, D., Reich, W., Wiebel, A., and Scheuermann, G., "Topology aware stream surfaces," *Comput. Graph. Forum* **29**(3), 1153–1161 (2010).

[14] Pang, A. T., Wittenbrink, C. M., and Lodha, S. K., "Approaches to uncertainty visualization," *The Visual Computer* **13**, 370–390 (1997).

[15] Pothkow, K. and Hege, H.-C., "Positional uncertainty of isocontours: Condition analysis and probabilistic measures," *IEEE Transactions on Visualization and Computer Graphics* **17**(10), 1393–1406 (2011).

[16] Alt, H., Braß, P., Godau, M., Knauer, C., and Wenk, C., "Computing the Hausdorff distance of geometric patterns and shapes," in [*Discrete and Computational Geometry. The Goodman–Pollack Festschrift*], **25**, 65–76 (2003).