EUROGRAPHICS 2013 / I. Navazo, P. Poulin
(Guest Editors)

Volume 32 (2013), Number 2

Global Selection of Stream Surfaces

Janick Martinez Esturo, Maik Schulze, Christian Rössl, and Holger Theisel

Visual Computing Group, University of Magdeburg

Abstract

Stream surfaces are well-known and widely-used structures for 3D flow visualization. A single surface can be sufficient to represent important global flow characteristics. Unfortunately, due to the huge space of possible stream surfaces, finding the globally most representative stream surface turns out to be a hard task that is usually performed by time-consuming manual trial and error exploration using slight modifications of seed geometries. To assist users we propose a new stream surface selection method that acts as an automatic preprocessing step before data analysis. We measure stream surface relevance by a novel surface-based quality measure that prefers surfaces where the flow is aligned with principal curvature directions. The problem of seed structure selection can then be reduced to the computation of simple minimal paths in a weighted graph spanning the domain. We apply a simulated annealing-based optimization method to find smooth seed curves of globally near-optimal stream surfaces. We illustrate the effectiveness of our method on a series of synthetic and real-world data sets.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems

1 Introduction

The visualization of 3D flow and vector fields is one of the core topics in Scientific Visualization. Although a variety of techniques has been proposed, 3D flow visualization is still challenging and bears a number of open problems. Stream surfaces are well-accepted structures for the visualization of 3D vector fields. In fact, deep and comprehensive research has been done on the problem how to integrate and how to graphically represent stream surfaces. However, the question *which* surface to integrate and present has been rarely studied in the literature. A common manual technique to solve this problem is an interactive expert-driven trial and error approach: a certain simple seed structure (e.g., a straight line) is moved around until relevant flow features are sufficiently visualized. This is a time-consuming task that requires a lot of experience, and the restriction to a particular seed curve geometry is a dramatic reduction of the search space of all possible stream surfaces. In fact, most stream surfaces cannot be obtained by straight seed curves.

This paper presents an automatic tool that assists the user with stream surface selection, i.e., we solve the following problem: given a 3D steady vector field $\mathbf{v}(\mathbf{x})$, find a stream surface that describes the properties of $\mathbf{v}$ best. Note that we are focusing on *one* representative stream surface, i.e., it is not the goal to densely cover the domain with surfaces. Furthermore, we search for a view-independent solution.

Until now there has not been a formal description of what a good stream surface is. Hence, we introduce a new quality measure for stream surfaces. Our quality measure prefers surfaces where lines of curvature are aligned with stream lines on the surface as well as possible. This quality measure is motivated by the fact that for such surfaces common line renderings of the geometry and the flow coincide and are therefore not interfering with each other. To make such a quality measure applicable, we show that it can be computed without explicitly estimating curvature fields on stream surfaces. Based on our quality measures, we provide a completely automatic algorithm for finding *globally* optimal stream surfaces. The utility of the quality measures and the extraction method is then demonstrated for a number of data sets. The contributions of the paper are:

- an analysis of the problem of stream surface selection, leading to the fact that it can only be solved by a global technique, and that the search space (the number of possible stream surfaces) is huge,
- a set of quality measures for stream surfaces, rewarding surfaces whose lines of curvature are as well as possible aligned with stream lines on it,
- a proof that such a quality measure can be computed without estimating the curvature tensor on the stream surface,
- an algorithm to select a globally optimal stream surface that is based on *simulated annealing*.

Definitions and notations. We make use of the following formal concepts: Given is a 3D steady differentiable vector field $\mathbf{v}(\mathbf{x})$ over the domain D. Let $\mathbf{J}(\mathbf{x})$ be the Jacobian matrix field of $\mathbf{v}$. To ease the formal presentation, we assume $D = \mathbb{R}^3$ to prevent boundary effects for the integration. This is not a limitation: in practice, integration is stopped when reaching the domain boundary. We use the concept of a *flow map* ϕ: $\phi^t(\mathbf{x})$ describes the location of a particle seeded at $\mathbf{x}$ after integrating in $\mathbf{v}$ over a time period t. Let $\mathbf{x}(s,t)$ be a differentiable parametric surface in D, and let $\mathbf{n}(s,t)$ be its unit normal. Then the defining property of $\mathbf{x}$ being a stream surface is $\mathbf{v}(\mathbf{x}(s,t))^{\mathsf{T}}\mathbf{n}(s,t) = 0$ for every (s,t) of the surface

domain. One way to obtain a stream surface is to start with a parametric seed curve $\mathbf{s}(s)$. A stream surface can then be written in parametric form as $\mathbf{x}(s,t) = \phi^t(\mathbf{s}(s))$.

2 Related Work

There is a large body of research on stream surface integration as well as stream surface rendering. A number of approaches is available to select suitable stream lines. Only few approaches exist for automatic stream surface selection.

Stream surface integration. The problem of stream surface integration is well-understood. Approaches range from Hultquist's classical advancing front line method [Hul92] to more advanced global time-scaling algorithms by Schulze et al. [SGRT12]. We refer the reader to the survey by Edmunds et al. [ELC*12] for an overview on stream surface integration and perception-improving rendering methods.

Stream surfaces for interactive flow exploration. Seed structures for stream geometry are usually placed manually. Seeds for stream ribbons and particles can be interactively moved around for real-time exploration [KKKW05]. The appearance of stream ribbons and surfaces can be enhanced using illustrative techniques as shown by Born et al. [BWF*10]. Hummel et al. [HGH*10] apply screen-space curvature approximations to enhance integral surface visualizations. These approaches focus on the visual representation and not on finding representative stream surfaces.

Stream line selection. The problem of finding good distributions of stream lines on 2D manifolds is well-researched. Turk and Banks [TB96] are first to propose an image-based algorithm, and Jobard and Lefer [JL97] propose a direct method to solve the stream line placement problem. Stream line placement in 3D flow is a more difficult problem. Several approaches have been proposed to find well-distributed seeds in data space as well as occlusion-aware methods that work in view space [MCHM10] or apply clustering [CYY*11]. We again refer to the survey by McLoughlin et al. [MLP*10] for an overview on advanced stream line seeding in both 2D and 3D flows. Note that the problem of stream line selection is less complex than our problem since the search space is significantly smaller: there is a unique stream line that passes through a point.

Automatic stream surface selection. Only few methods have been proposed for automatic selections of stream surfaces. Cai and Heng [CH97] present the first method to compute starting points for stream surfaces automatically. Their approach is based on implicit stream surfaces presented by van Wijk [vW93]. The resulting surfaces can either be extracted as isosurfaces or rendered directly via volume rendering. However, the necessary stream function integral of the vector field can only be computed for curl-free flow. Theisel et al. [TWHS03], Weinkauf et al. [WTHS04], and Peikert et al. [PS09] automatically find seed curves from topological structures. Unfortunately, their methods may either extract too many or not enough stream surfaces. Recently Edmunds et al. [EML*12] use isolines on domain boundaries as seed curves and propagate stream surfaces into surrounding space. Their method is limited to the existence non-vanishing flux on domain boundaries. In a follow-up work, a clustering approach of *local* flow properties is applied to find seed curve locations [ELM*12]. However, no surface-based measure is applied to evaluate the optimality of resulting stream surfaces. Since stream surfaces are *global* structures due to integration, our new approach is based on a global flow field analysis and optimization by measuring optimality of complete stream surfaces. In particular, our method is independent of vector field topology, or the existence of curl or outflow boundaries.

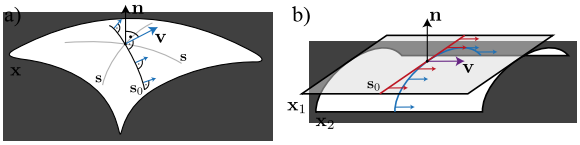


Figure 1: *Stream surface parameterizations. a) Different seed curves* $\mathbf{s}$ *describing the same stream surface. Among all stream surfaces there is a unique orthogonal-optimal one* $\mathbf{s}_0$. *b) Different orthogonal-optimal seed curves produce different stream surfaces, here for the example of a constant* $\mathbf{v}$.

3 On the Complexity of the Search Space

In this section we give a deeper analysis of the problem.

Why a global surface-based approach? All other stream surface selection methods mentioned above have in common that they are *local* methods: stream surface are exclusively selected by the quality of seed curves, i.e., by considering $\mathbf{v}$ and its derivatives along the curves. Even if their seed curves perfectly fulfill such local criteria [ELM*12], nothing prevents the stream surfaces from being integrated into areas where they are either less interesting or even lead to poor visualizations by hiding interesting structures. Instead, it is necessary to evaluate the quality of stream surfaces that are integrated from these seed curves. This requires a surface-based quality measure, which is *global* by construction due to the domain-wide stream surface integration. We introduce such a surface-based quality measure in Section 4.

Why only one stream surface? Common approaches for stream line selection focus on finding a set of stream lines that cover the domain in a certain way. The situation for stream surfaces is different: even a single surface can cover the complete screen space. Moreover, it is known that even in low numbers stream surfaces tend to hide each other, leading to cluttered visualization. Also, having a look into classic literature where 3D flow illustrations are used reveals that many examples of 3D flows are only illustrated with a single carefully chosen stream surface [AS92, Dal83]. This assumption fails for highly turbulent flow for which surface-based approaches are generally considered unsuitable.

How complex is the search space? Before proposing a solution, we analyze how complex the problem of stream surface selection is. In other words: we answer the question how

many stream surfaces exist for a 3D vector field. Consider a point $\mathbf{q} \in D$ and a normal $\mathbf{n}$ with $\mathbf{n}^{\mathrm{T}}\mathbf{v} = 0$. We analyze how many different stream surfaces exist that pass through $\mathbf{q}$ and have the surface normal $\mathbf{n}$ there. Every stream surface can be described by a seed curve $\mathbf{s}(s)$ with

$$\mathbf{s}(0) = \mathbf{q}\,, \quad \dot{\mathbf{s}}(0)^{\mathrm{T}}\mathbf{n} = 0\,, \quad \dot{\mathbf{s}}(0) \times \mathbf{v}(\mathbf{q}) \neq \mathbf{0}\,, \qquad (1)$$

where $\dot{\mathbf{s}} := \frac{\mathrm{d}}{\mathrm{d}s}\mathbf{s}(s)$ denotes the tangent vector of $\mathbf{s}$. Note that different seed curves fulfilling (1) describe the same stream surface. In fact, a seed curve $\mathbf{s}'(s) = \phi^{\lambda(s)}(\mathbf{s}(s))$ for any differentiable function $\lambda(s)$ with $\lambda(0) = 0$ gives the same stream surface in a different parametrization. Among all seed curves describing the same surface there is one unique representative: the seed curve fulfilling (1) and additionally the ordinary differential equation (ODE) $\dot{\mathbf{s}}(s)^{\mathrm{T}}\mathbf{v}(\mathbf{s}(s)) = 0$ for any s. We call such a seed curve an *orthogonal-optimal seed curve* $\mathbf{s}_0$. Figure 1 (a) illustrates the concept.

In order to find the number of different stream surfaces, we have to find the number of different orthogonal-optimal seed curves, i.e., curves that are the solution of the ODE with the initial value $\dot{\mathbf{s}}(0) = \mathbf{v}(\mathbf{q}) \times \mathbf{n}$. Note that this ODE is underdefined: solving it as an initial value problem for a certain s does not give a unique direction of continuation but a one-parametric family of directions. Thus, the set of different orthogonal-optimal seed curves is an infinite-dimensional infinite set. Figure 1 (b) exemplifies this with the aid of a constant vector field where two different stream surfaces through the same point with the same normal are shown. From this it follows that the cardinality of the number of different stream surfaces through $\mathbf{q}$ with normal $\mathbf{n}$ is above the cardinality of $\mathbb{R}^n$ for any natural number n.

4 Quality Measures for Stream Surfaces

Using *perceptual* arguments we investigate the question which intrinsic properties characterize a good stream surface. Their graphical representation is challenging because two pieces of information have to be simultaneously presented: the surface shape and the flow on the surface. The shape is usually represented by (non)-photorealistic rendering techniques [BWF*10]. To show the flow on the surface, most 2D steady flow visualization techniques can be adapted to stream surfaces, such as image based techniques [LvWJH04], stream lines [SLCZ09], or elementary techniques [PGL*12]. However, the perception of the shape and the flow are not independent – see for example the work by Laramee et al. [LvWJH04], where image-based flow visualizations on the surface limit the perception of the shape. Hence, it is desirable to find stream surfaces where the representation of their shapes and the embedded flows yield similar structures and therefore do not interfere with each other.

In non-photorealistic rendering of shapes, standard techniques for surface representation are line rendering and hatching [DC90, SS02]. It is known that lines of curvature are good candidates for representing the surface shape [RKS00]. A variety of user-studies show that lines of curvature improve the perceptibility of surfaces [KHSI04, SW04]. If these lines are stream lines at the same time, they can represent *both* shape and flow. Hence, our approach is to search for stream surfaces where the stream lines are aligned with the principal directions as good as possible. Figure 2 (a,b) illustrates the setting. Our experiments indicate that this alignment measure characterizes representative stream surfaces of a flow domain. Note that recently curvature approximations have already shown to be beneficial for integral surface rendering [HGH*10].

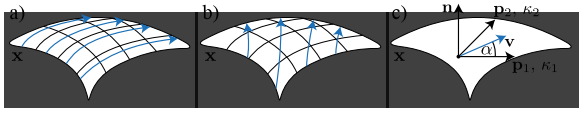


Figure 2: *Stream surface quality. a) A stream surface of high quality: the stream lines (•) are aligned with the lines of curvature (•), b) a stream surface of lower quality. c) Setting of the local alignment error:* α *is the angle between* $\mathbf{v}$ *and one of the principal directions* $\mathbf{p}_1, \mathbf{p}_2$ *of the surface* $\mathbf{x}$.

To formulate a local alignment error, consider a stream surface $\mathbf{x}(s,t)$ at a certain point, and let κ_1, κ_2 be the principal curvatures and $\mathbf{p}_1, \mathbf{p}_2$ the corresponding principal directions of $\mathbf{x}$ in its tangent space [BKP*10]. Furthermore, let $\mathbf{v}$ be the velocity at $\mathbf{x}$, and let α be one of the angles between $\mathbf{v}$ and one of the principal directions. Figure 2 (c) gives an illustration. Then, we define the *local alignment error*

$$e_a = \cos\alpha \, \sin\alpha \, (\kappa_2 - \kappa_1)\,. \qquad (2)$$

Note that e_a^2 neither depends on the choice of the principle direction nor on its orientation, and it vanishes if $\mathbf{v}$ is aligned with one of the principal directions and at umbilical points.

The error e_a has a related differential geometric interpretation providing an additional motivation for the measure: following Euler's theorem the *normal curvature* κ_n of $\mathbf{x}$ in the direction $\mathbf{v}$ is $\kappa_n = \kappa_1 \cos^2\alpha + \kappa_2 \sin^2\alpha$, which gives $e_a = \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}\alpha}\kappa_n$. Hence, e_a is the directional derivative of κ_n in flow direction. Surfaces that minimize e_a^2 therefore have low normal curvature variation in the direction of $\mathbf{v}$. The process of curvature variation minimization is generally termed *surface fairing*, which follows the *principle of simplest shape*: the surface should be free of any unnecessary details or oscillations (see, e.g., [BKP*10]). Hence, minimizing e_a^2 globally yields stream surfaces that do not only capture local flow details but rather represent global flow features.

The error e_a is not yet suitable for minimization, because it requires a local estimation of the curvature tensor of $\mathbf{x}$. We consider the first and second partial derivatives of $\mathbf{x}$ from which the surface curvature can be computed:

$$\mathbf{x}_s = \dot{\mathbf{s}} \quad \mathbf{x}_t = \mathbf{v} \quad \mathbf{x}_{ss} = \ddot{\mathbf{s}} \quad \mathbf{x}_{st} = \mathbf{J}\dot{\mathbf{s}} \quad \mathbf{x}_{tt} = \mathbf{J}\mathbf{v}\,. \qquad (3)$$

From (3) it is straightforward to compute $\kappa_1, \kappa_2, \mathbf{p}_1, \mathbf{p}_2$. Inserting these values into (2) gives

$$e_a = \frac{\mathbf{n}^{\mathrm{T}}\mathbf{J}\,(\mathbf{v}\times\mathbf{n})}{\|\mathbf{v}\|^2}. \qquad (4)$$

(The step from (2) to (4) is a straightforward computation for which we provide an accompanying Maple sheet.) Equation (4) shows a *remarkable* property: the local alignment error at $\mathbf{x}$ with normal $\mathbf{n}$ does not depend on the seed curve but only on $\mathbf{n}$, $\mathbf{v}$, and $\mathbf{J}$. In other words: all stream surfaces through a

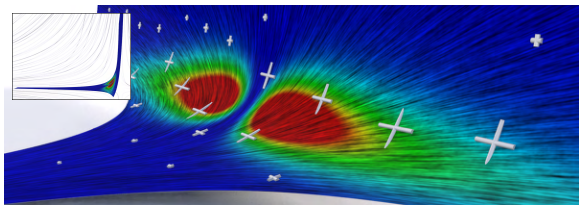


Figure 3: *Stream surface quality. A simple stream surface in a saddle vector field. Superimposed are principal curvature directions (●, scaled by $|\kappa_1 - \kappa_2|$) and the local vector field (LIC). On the surface the squared local alignment error e_a^2 is color coded (low error ●, high error ●), which depends on the alignment of the flow and* any *principal direction.*

point $\mathbf{x}$ with normal $\mathbf{n}$ have the same local alignment error! See Figure 3 for an example.

Based on e_a, we compute the *average squared alignment error* by integrating e_a^2 over the stream surface:

$$E_a = \frac{1}{A}\int_{t_0}^{t_1}\int_{s_0}^{s_1} e_a^2 \,\|\mathbf{x}_s \times \mathbf{x}_t\| \;\mathrm{d}s\;\mathrm{d}t \tag{5}$$

for the stream surface $\mathbf{x}(s,t)$ with $(s,t) \in [s_0,s_1]\times[t_0,t_1]$ and surface area A. The measure E_a is non-negative and comparable for stream surfaces of different area.

The error E_a will be the target function for minimization. However, E_a has trivial minimizers, e.g., stream surfaces in laminar flows with almost vanishing Jacobian: in these areas, nearly planar stream surfaces minimize (5) because e_a vanishes. To exclude these trivial solutions, we expect a good stream surface to have non-vanishing average normal curvature. Setting $E = \mathbf{x}_s{}^{\mathrm{T}}\mathbf{x}_s$, $F = \mathbf{x}_s{}^{\mathrm{T}}\mathbf{x}_t$, $G = \mathbf{x}_t{}^{\mathrm{T}}\mathbf{x}_t$, $L = \mathbf{n}^{\mathrm{T}}\mathbf{x}_{ss}$, $M = \mathbf{n}^{\mathrm{T}}\mathbf{x}_{st}$, and $N = \mathbf{n}^{\mathrm{T}}\mathbf{x}_{tt}$, the normal curvature is the ratio of first and second fundamental forms [BKP*10], i.e.,

$$\kappa_n = \frac{L\,\mathrm{d}s^2 \;+\; 2M\,\mathrm{d}s\,\mathrm{d}t \;+\; N\,\mathrm{d}t^2}{E\,\mathrm{d}s^2 \;+\; 2F\,\mathrm{d}s\,\mathrm{d}t \;+\; G\,\mathrm{d}t^2}\;. \tag{6}$$

Since the tangential direction $\mathbf{v}$ corresponds to $(\mathrm{d}s,\mathrm{d}t) = (0,1)$, inserting (3) in (6) gives

$$\kappa_n = \frac{\mathbf{n}^{\mathrm{T}}\mathbf{J}\,\mathbf{v}}{\|\mathbf{v}\|^2}\;.$$

To obtain a comparable measure we compute the *average squared normal curvature*

$$K_n = \frac{1}{A}\int_{t_0}^{t_1}\int_{s_0}^{s_1} \kappa_n^2 \,\|\mathbf{x}_s \times \mathbf{x}_t\| \;\mathrm{d}s\;\mathrm{d}t\;. \tag{7}$$

We will later minimize deviation of K_n from a prescribed K_0.

A third quality measure solves the ambiguity of the seed curve of a stream surface: we expect a good seed curve to be as perpendicular as possible to the flow. We define

$$E_p = \frac{1}{\ell}\int_{s_0}^{s_1}\left(\frac{\mathbf{v}^{\mathrm{T}}\dot{\mathbf{s}}}{\|\mathbf{v}\|\|\dot{\mathbf{s}}\|}\right)^2 \|\dot{\mathbf{s}}\|\;\mathrm{d}s \tag{8}$$

where ℓ is the arc length of $\mathbf{s}$. Without loss of generality we assume $\mathbf{s}$ to be arc length parameterized such that $\ell = s_1 - s_0$. Finally, we have to consider the area A of the stream surface as a fourth quality measure.

The aim is to find a stream surface that minimizes these four quality measures. Direct numerical optimization of this problem is hardly feasible due to the search space complexity. Instead, we develop a combinatorial approach yielding near optimal stream surfaces w.r.t. our quality measure.

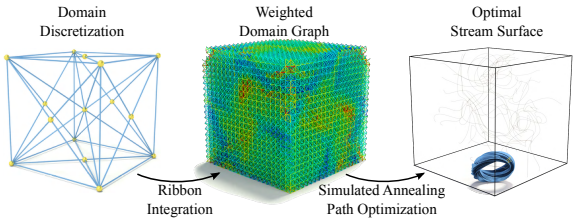


Figure 4: *Algorithm overview. We discretize the domain using a domain graph of refined cubic cells (left) and perform a stream ribbon integration for each edge to assign edge costs w_i by* (9) *(middle, low costs ●, high costs ●). A simple path of minimal costs computed using simulated annealing then yields an optimal seed structure (right).*

5 Automatic Seed Curve Selection

Using the stream surface quality measures we are now able to present our stream surface selection strategy. The algorithm consists of the following major steps that are executed automatically for a user-provided flow domain and a set of parameters. See Figure 4 for an overview of the method. First, we create a spatial graph that densely covers the domain. Then, a stream ribbon integration is performed for each edge of this graph. The quality of each stream ribbon defines a cost value for each grid edge. We then perform a global optimization for paths of approximate minimal costs using a simulated annealing algorithm. In a final smoothing step on the resulting path, we obtain the seed curve for the integration of the resulting stream surface.

Domain graph. In order to perform a global optimization on the space of stream surfaces we need a suitable discretization of this space or equivalently of the space of seed curves. For this discretization we use the following *inclusion property for seed curves* of stream surface:

Given a seed curve $\mathbf{s}(s)$, $s_0 \le s \le s_1$, of a stream surface $\mathbf{x}$, this curve can be subdivided at a point $\mathbf{s}(s_s)$ into two curves $\mathbf{s}_a(s_a)$, $s_0 \le s_a \le s_s$ and $\mathbf{s}_b(s_b)$, $s_s \le s_b \le s_1$, such that $\mathbf{s} \equiv \mathbf{s}_a \cup \mathbf{s}_b$ holds. Likewise, when $\mathbf{s}_a$ and $\mathbf{s}_b$ are used as seed curves for new stream surfaces $\mathbf{x}_a$ and $\mathbf{x}_b$, then also $\mathbf{x} \equiv \mathbf{x}_a \cup \mathbf{x}_b$ holds because both $\mathbf{x}_a$ and $\mathbf{x}_b$ share the unique streamline starting at $\mathbf{s}(s_s)$. One can therefore always extend a stream surface with another stream surface if their corresponding seed curves connect in a common point.

This allows a domain discretization with short curve segments joining at common points. We use linear curve segments and the following subdivision scheme: we first create a grid of uniform cubic cells that covers the whole domain. The edges of these cells are all axis-aligned. In order to provide more directional degrees of freedom, each cell is subdivided by inserting vertices at the six face centers and the cell center, which are connected by face diagonals and cell diagonals, respectively. Figure 4 (left) shows this setting.

The sets of all vertices V and all edges E define an undirected *domain graph* $G = (V,E)$ with vertices embedded in

D. Let P be the set of all simple paths in G, i.e. paths with no vertex repetitions. We use simple paths to approximate the seed curves: any simple sequence of edges yields a piecewise linear seed structure candidate. Possible candidates are evaluated w.r.t. the quality of the stream surfaces that they define. The quality measure can be evaluated for each edge independently to define edge costs. Due to the inclusion property, the quality or equally the costs of a path results in a summation of associated edge costs.

Edge costs. The cost $w_i > 0$ of each edge $e_i \in E$ aggregates intrinsic properties of narrow stream ribbons $\mathbf{x}_i \subset D$ (i.e., of stream surface with short seed curves). We calculate each $\mathbf{x}_i$ by performing a stream surface integration using each edge as the seed geometry. Integration is performed for a predefined maximum time range and is stopped at the domain boundary. There are no special requirements at this stage: we use Hultquist's algorithm [Hul92]; any other adaptive method for stream surface integration is applicable as well. A stream ribbon is integrated for each edge of the domain graph. This is the most time-consuming step of our method (see Section 6.4). However, this way edge costs are based on truly *nonlocal* features of the vector field.

There are two remarks on this step that are worth noting: As ribbons are integrated independently, we use a parallelized computation. Also, we observe that reliable edge costs generally do not require ribbons of high resolution. Therefore, a relatively coarse tessellation of the stream surface meshes is sufficient.

Edge costs are modeled as weighted combinations of the quality measures defined in Section 4, which are evaluated on each stream ribbon $\mathbf{x}_i$. Edge costs shall be minimal if a linear combination of quality measures is minimized.

We compute a discrete approximation of the surface integrals (5) and (7) by quadrature, which samples local values of e_a and κ_n at each triangle center and weights samples by triangle area. This yields values E_a^i and K_n^i for the ribbon $\mathbf{x}_i$ with surface area A^i. Similarly, E_p^i is an approximation to the line integral (8), which is also evaluated by quadrature.

The absolute values of these measures differ and cannot be compared between data sets. In order to have parameters that are independent of the data, a normalization of measures is required that yields *relative* values. We normalize E_a^i, K_n^i, E_p^i, and A^i to the range $[0,1]$ to obtain $\bar{E}_a^i$, $\bar{K}_n^i$, $\bar{E}_p^i$, and $\bar{A}^i$. For the curvature-based measures E_a^i and K_n^i we apply an additional log-transformation. This is due to the fact that curvatures are not bounded and can become very large. Moreover, they are not equally distributed, meaning that only in small regions large values appear, while large regions have rather small values for e_a and κ_n.

We model the final cost for an edge of length ℓ_i by weighted quadratic contributions as

$$w_i = \ell_i \sum_{x \in \{a,n,p,A\}} \beta_x \, (F_x^i)^2 \qquad (9)$$

with $F_a^i = \bar{E}_a^i$, $F_n^i = \bar{K}_n^i - \tau$, $F_p^i = \bar{E}_p^i$, and $F_A^i = 1 - \bar{A}^i$. Cost minima of w_i minimize the alignment error (5). Moreover, higher costs are assigned to edges that are aligned with the flow and whose stream ribbons have a smaller surface area. The weights β_x, $x \in \{a,n,p,A\}$ define the relative weighting of cost components. Generally, a good choice is $\beta_x = 1$: we use this setting unless stated otherwise for some experiments in Section 7. Then $\tau \in [0,1]$ remains the only parameter provided by the user: it steers the desired amount of average normal curvature with $\tau = 1$ for higher and $\tau = 0$ for lower values. Essentially, τ captures the prescribed average squared normal curvature K_0 (see Section 4) after normalization. If an edge is aligned with the flow, i.e. $\bar{E}_p^i > \delta$ (we use $\delta = 0.8$), we ignore its weight and discard the edge from further processing by setting $w_i = \infty$. Figure 4 (middle) shows a graph with color-coded costs for each edge.

Path costs. We define the total costs of a path p_k as

$$c_\gamma(p_k) = (1-\gamma) \sum_{i \in E_k} w_i + \gamma \, \kappa(p_k) \ .$$

$E_k \subset E$ is the set of edges of p_k. This is a linear blend between edge costs and a normalized measure of discrete polyline curvature κ. We include the additional path curvature term to penalize undesirable "space-filling" paths in areas of constant low edge costs (e.g., in laminar flow areas). The weight $0 \leq \gamma < 1$ is a user parameter that trades minimal edge costs versus straightest seed curves (see Section 6.1 for results of different choices for γ). A simple estimation of the (normalized) curvature of the path is sufficient, we use $\kappa(p_k) = \frac{1}{\pi} \sum_{e_i, e_{i+1} \in E_k} \pi - \sphericalangle(e_i, e_{i+1})$. Here, $\sphericalangle(e_i, e_{i+1})$ denotes the angle between two consecutive edges on the path.

Global optimization problem. Our goal is to find a seed curve of the best stream surface w.r.t. the defined quality measure, i.e., most importantly, we prefer surfaces with stream lines aligned with lines of curvature. This requires searching for the globally best seed curve in the entire domain. We formulate this global optimization problem as a combinatorial problem: the space of seed curves is discretized as the space of simple paths in the domain graph, which covers the domain densely. The quality of a seed curve is evaluated as the sum of edge costs in a path. The prescribed arc length ℓ of seed curves is measured as the number of edges in a path, giving $n = \lfloor \ell / \ell^e_{\text{avg}} \rceil$ for an average graph edge length ℓ^e_{avg}. Formally, we want to optimize

$$p^\star = \text{argmin}_{p_k \in P} \left\{ c_\gamma(p_k) \;\middle|\; |E_k| = n \right\} \qquad (10)$$

for an optimal simple path $p^\star$ of length n. This is a combinatorial assignment problem on a large finite search space P. In the literature the dual problem is called the *Heavy Path Problem* [KBL12]. It is known to be NP-hard, i.e., it is practically infeasible to compute the *exact* solution. Instead, we present an algorithm for computing an approximate solution in the following sections.

Local minimal paths. A key idea for a practical solution of (10) is to prune the search space. Instead of all simple paths of length n in P, we consider only a subset $Q \subset P$ that includes the simple minimal path starting in v of length n for each vertex $v \in V$. This implies $|Q| = |V|$, where the paths in

Q are determined by a *local* minimization. This pruning step is justified by the fact that any other paths $p \notin Q$ do not even constitute local minima and are therefore not considered as candidates for a global minimum.

The locally minimal paths can be computed for each vertex using, e.g., Dynamic Programming, an algorithmic standard technique that was recently used for finding Heavy Paths [KBL12]. However, in practice this is only feasible for $n \leq 6$ due to the exponential combination complexity, which results from the high branching factor of the graph (most vertices have 16 neighbors). We therefore approximate Q with a greedy iterative deepening depth-first search that restarts a depth-first search phase after each d steps from the current optimal solution. This is justified by the fact that edges of minimal costs are distributed along line structures – hence the depth-first search phase – in the graph due to the penalization of flow-aligned edges. We use $d = 5$ in all our experiments. This allows computing approximate local minimal paths of length $n > 15$ efficiently. We implemented a lazy evaluation of Q, i.e., locally minimal paths are computed on demand and then stored for subsequent evaluations.

5.1 Global Path Optimization by Simulated Annealing

An often successfully used meta-heuristic to approximately solve problems similar to Equation (10) is the *Simulated Annealing* (SA) algorithm. It is a physically inspired algorithm that models the controlled slow cooling of heated materials. Slow cooling reduces defects in the crystal structure of the material, which can be interpreted as a form of internal energy that is optimized. The SA algorithm was introduced by Kirkpatrick et al. [KGV83] as a general heuristic global optimization tool for hard problems. For instance, SA was only recently used in the visualization community by Sigg et al. to solve the NP-hard problem of generating optimal cutaway illustrations [SFCP12]. Further applications of SA can be found in the survey of Suman and Kumar [SK06]. We briefly describe the SA algorithm in the context of our problem and refer to the survey for more details on the method.

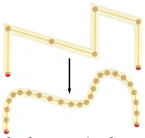
The SA algorithm introduces a system temperature $T > 0$ that serves as a control parameter. A candidate solution is improved (the amount of change depends on T), and T is slowly decreased following a temperature schedule.

Single optimization step. We use the SA algorithm for an iterative update of a candidate path that starts at the current vertex v_c, which is initialized randomly. A new candidate path is randomly sampled in the domain by selecting a new starting vertex v_n in the Euclidean neighborhood of v_c according to a normal distribution with mean $\mathbf{x}_c$ and variance $\sqrt{T}/2$, weighted by the length of the diagonal of the domain bounding box. Higher temperatures increase the probability for selecting distant candidate vertices while at lower temperatures variations are more local. The probability that v_n is accepted for the next iteration depends on the cost difference $d = c_\gamma(v_c) - c_\gamma(v_n)$. Here, $c_\gamma(v)$ denotes the cost of the minimal simple path $p_v \in Q$ that starts in vertex v. The probability is given by the Metropolis transition probability $a_{v_c \to v_n} = \min\left\{1, e^{d/T}\right\}$. $a_{v_c \to v_n}$ guarantees that better solutions ($d > 0$) are always accepted. Likewise, worse solutions ($d < 0$) can also be accepted, as $a_{v_c \to v_n} > 0$ still holds, and the probability for these events is enlarged for higher temperatures T. This property enables SA to not get stuck in local minima but leave them once entered.

Temperature schedule. The classic SA algorithm consists of a heating (annealing) and a cooling phase that schedule the temperature [KGV83]. We found the heating phase results in almost equal initial temperatures in all our experiments – we therefore usually skip heating and start the cooling phase with a fixed value T_0. In the cooling phase multiple optimization steps are performed at a fixed temperature, and we count the number of accepted steps m_a. If $m_a > m_s$ for a fixed value m_s, the temperature is lowered by multiplying it with a constant factor $0 < \lambda < 1$. The scaling of the temperature by λ is also performed if the number of total steps exceeds a maximum value $m_{max} > m_s$. This scheme results in an exponential temperature decay. We stop the iteration if m_{max} is not reached for three consecutive times and consider the solution to be converged to $p^\star$. We used the constant factors $T_0 = 1/2$, $\lambda = 0.9$ that performed well in all our experiments. We analyze choices for m_s and m_{max} in Section 6.3.

5.2 Seed Curve Generation

The path $p^\star$ represents a piecewise linear seed curve. We interpret this as an approximation to a smooth, C^2-continuous, seed curve $\mathbf{s}^\star$ that is obtained by smoothing $p^\star$ by a univariate subdivision scheme. Due to the close proximity of the spatial locations and tangential directions of $p^\star$ and $\mathbf{s}^\star$, we expect the resulting stream surfaces to have similar behavior.



We use a simple corner cutting scheme that yields C^2-continuous cubic splines in the limit (see, e.g., Sabin [Sab10]) with an additional endpoint interpolation rule. In each iteration, the scheme generates new points by the rules $\mathbf{y}_{2i}^{k+1} = \frac{1}{2}\mathbf{y}_i^k + \frac{1}{2}\mathbf{y}_{i+1}^k$, $\mathbf{y}_{2i+1}^{k+1} = \frac{1}{8}\mathbf{y}_i^k + \frac{6}{8}\mathbf{y}_{i+1}^k + \frac{1}{8}\mathbf{y}_{i+2}^k$. Typically, three subdivisions provide a sufficiently close approximation to the smooth curve. We obtain the resulting optimal stream surface $\mathbf{x}^\star$ by starting an integration from $\mathbf{s}^\star$.

6 Results

We evaluate our method by a parameter description and present results of applying our approach to both analytical and real world vector fields. Additionally, we provide a quantitative analysis of the convergence behavior of the optimization and timings of our method.

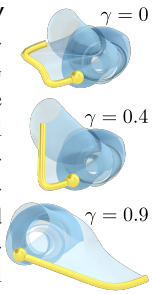
6.1 Parameters

Our method requires specification of the domain graph resolution. It strongly influences the runtime of the method, as it prescribes the number of grid edges and therefore the number of stream ribbon integrations and the size of the search space of the optimization. However, we found that even relatively low resolutions yield sound results, e.g., all the analytical examples have a graph resolution of 20^3 cells. The

highest resolution used in this work was a $20 \times 40 \times 15$ cells graph for the ACOUTLET data set shown in Figure 5. In general, we observe that a good default value for the resolution is given by the length ratio of vector field grid cell diagonal to domain graph cell diagonal around five (see Table 1). We evaluate the influence of different resolutions in Section 7.

Also, depending on the concrete vector field characteristics, the integration direction (i.e., forward only or forward and backward) and the maximum integration time should be specified. For vector fields with an inflow / outflow area (e.g., in the CYLINDER data set) we integrate in forward direction only until the integration reaches the boundary. In all other cases we integrate into both directions.

The number of edges n of the optimized paths dictates the total length of the optimized seed curve. It is related to the graph resolution, which specifies the range of edge lengths, and it influences the performance of the optimization, as local minimal path computations are cheaper for smaller n. We require the user to supply a value for n. Note that once the edge costs are computed, the less time-consuming path optimization step can be redone with a different n and γ, as required (see Section 6.4).



For different path curvature parameters γ we illustrate resulting minimal paths (starting at the *same* vertex in the DELTAWING flow) in the inset. All paths still respect the local edge cost distribution (e.g., they still have the orthogonal-optimal property by lying in a plane orthogonal to the flow). Increasing values of γ comes with increased edge costs and therefore less optimal resulting stream surfaces. However, the minimal path in terms of path costs only might not always be the most desirable: for example, in areas of constant low edge costs the minimal path would be a space-filling path with high curvature, which we found to be a spurious result. In all our experiments we found that using a value of $\gamma = 0.2$ avoids this issue and yields all results presented in this work.

6.2 Stream Surface Selection Results

We continue to present results of our approach and refer to the accompanying video for more examples.

Analytic data sets. Figure 5 (top) shows a series of automatically selected stream surfaces in well-known simple analytic vector fields. These analytic vector fields are well-suited to exemplify our quality concept of alignment (see Figures 3 and 5 (bottom)), combined with a variable prescribed mean normal curvature. To do so, we show results that minimize ($\tau = 0$) and maximize ($\tau = 1$) mean normal curvature while still optimizing the other quality measures (cf. Equation (9)). We call the resulting optimal surfaces $\mathbf{x}^\star_{\tau=0}$ and $\mathbf{x}^\star_{\tau=1}$. In the SADDLE field the optimal stream surface $\mathbf{x}^\star_{\tau=0}$ (with $\beta_n = 1$) is planar. In fact, it is *exactly* the same solution that is obtained for $\beta_n = 0$, i.e., if the mean normal curvature is ignored in the optimization. This confirms our theoretical proposition that the alignment measure is minimized by surfaces of vanishing curvature. Naturally, the absolute alignment measure E_a is higher for $\mathbf{x}^\star_{\tau=1}$. Still, the surface is the best-aligned solution given the additional mean normal curvature constraint.

Note that the optimized seed structures are not necessarily line segments only. We also note that these simple linear vector fields exhibit a high degree of symmetry and that the same optimal solutions can be generated by different seed structures. Moreover, a given solution might not be the unique globally optimal solution. Our algorithm selects one of the possible minimal solutions. As practical problems do in general not have such perfect symmetries, we expect to find unique solutions in these kinds of data sets.

Real world data sets. We apply our selection method to a number of complex vector fields of different domains and varying characteristics. The resulting stream surfaces are shown in Figure 5 (middle). Again the optimal seed structures are general curves. The CYLINDER vector field represents the flow around a square cylinder [CSBI05]. It is a well-known and extensively studied phenomenon [BFTW09, ELM*12, SGRT12]. As we choose to integrate in forward direction only, both stream surfaces $\mathbf{x}^\star_{\tau=0}$ and $\mathbf{x}^\star_{\tau=1}$ are seeded at the boundary of the inflow area to maximize the area constraint. Additionally, both surfaces conform to the chosen τ parameter (low and high mean normal curvature) while still minimizing the alignment error. This property does also hold for the DELTAWING data set. It is a flow simulation around a triangle-shaped airplane (see [BWF*10, GKT*08, GTS*04, HGH*10, SRWS10, SGRT12] for other approaches using a similar data set). Here, the mean normal curvature of $\mathbf{x}^\star_{\tau=0}$ vanishes. In contrast, the $\mathbf{x}^\star_{\tau=1}$ stream surface is well aligned with the two dominant vortex features. It is a single stream surface where both vortical parts are connected by laminar flow areas above and beneath the airplane. The ACOUTLET data set represents the flow in the outlet of an air conditioning unit. It is used to predict the degeneration of filters in the (hidden) dissipation grid layer. With a resolution of 1.6×10^7 grid cells it is the largest data set we tested. Our method is able to select a curved seed structure of a stream surface with a high outflow rate. The outlet area of a hydroelectric turbine is simulated in the TURBINE data set where the flow is split at a bifurcation. Here, $\mathbf{x}^\star_{\tau=1}$ is a stream surface leaving the domain on both sides of the bifurcation. The ANEURYSM is a blood flow simulation at a human cerebral aneurysm, which is a weakness of the vessel wall and potentially leads to rupture and life-threatening bleeding. The selected stream surface $\mathbf{x}^\star_{\tau=1}$ is clinically relevant as it covers a large fraction of the volume of the aneurysm. The flow of the measured BUBBLECHAMBER data set of a bioreactor (see [SWH05]) has different flow characteristics compared to the previous in/out-flow dominant examples. Nevertheless, the selected $\mathbf{x}^\star_{\tau=0}$ and $\mathbf{x}^\star_{\tau=1}$ are still similar to the previous examples concerning the planarity of $\mathbf{x}^\star_{\tau=0}$ and the more feature capturing property of $\mathbf{x}^\star_{\tau=1}$. The alignment measure visualizations in

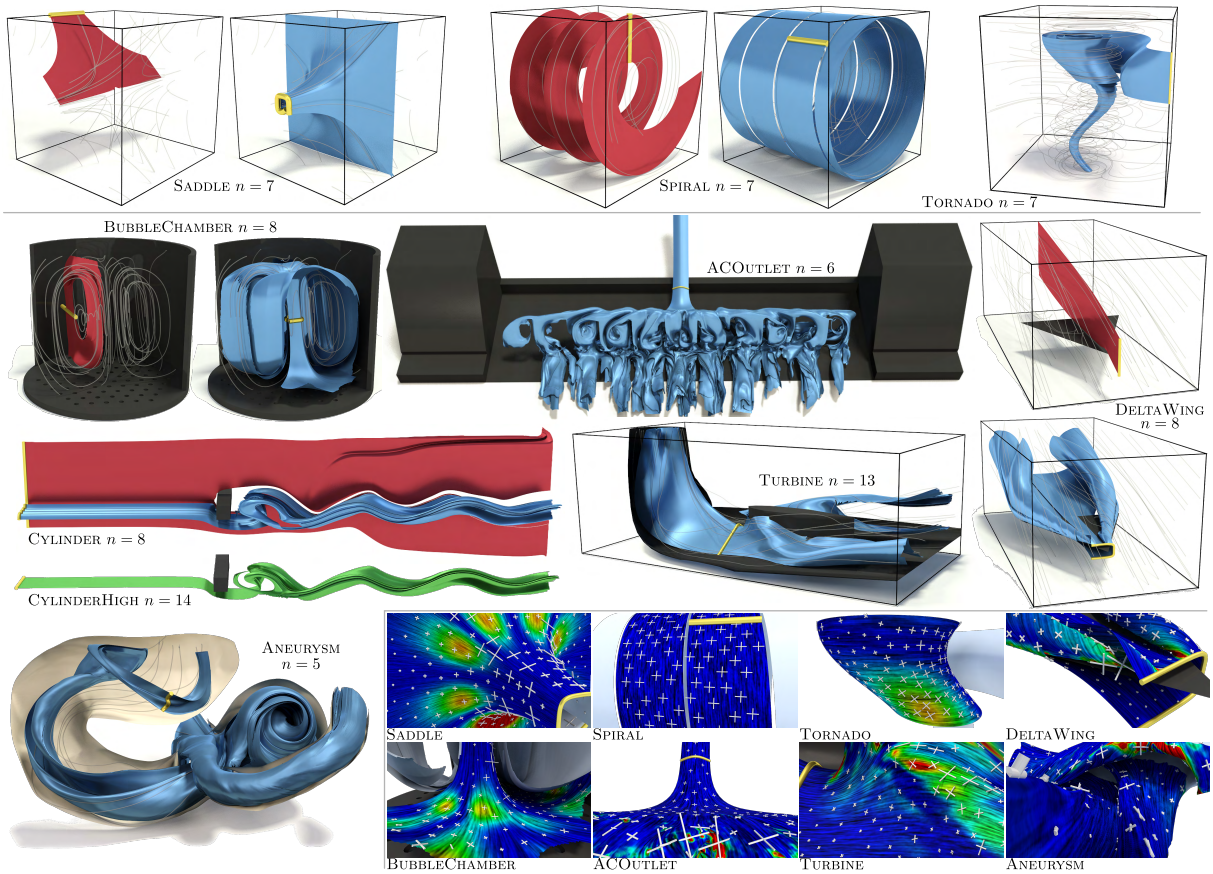


Figure 5: *Automatic selection results for analytical (top) and simulated vector fields (middle). The shown alignment-optimal stream surfaces were integrated from the optimized seed curves (●). Surfaces $\mathbf{x}^\star_{\tau=0}$ (●) minimize and surfaces $\mathbf{x}^\star_{\tau=1}$ (●) maximize mean normal curvature. The surface in* CylinderHigh *(●) is $\mathbf{x}^\star_{\tau=1}$ extracted from a domain graph of higher resolution compared to* Cylinder*. The closeups (bottom) show e_a^2 and scaled principal directions of the $\mathbf{x}^\star_{\tau=1}$ examples (cf. Figure 3).*

Figure 5 (bottom) illustrate the local quality of the selected globally optimal solutions. In all examples the largest portion of the principal directions are well-aligned with the flow as prescribed by the alignment measure.

Our method optimizes for flow alignment of the shape of stream surfaces in combination with additional intrinsic stream surface properties. We expect our automatically selected globally optimal results computed this way to be relevant for the shown application areas. This is indicated by the fact that the stream surfaces we find are very close to the stream surfaces presented in other approaches. See, e.g., the related work on the DeltaWing data set, for which our automatically selected stream surface is similar to the manually picked stream surfaces.

6.3 Simulated Annealing Convergence

For optimization we apply a heuristic search in form of the SA algorithm, which is a randomized method. It is therefore mandatory to analyze the quality and optimality of the results of this algorithm. To do so we performed an evaluation run of our method on a synthetic and a simulated data set. Given a domain graph we find the ground truth optimal minimal path $p^\star$ using a naïve search and exact minimal path computations using full depth-first search. Due to the high branching factor of the graph this can only be done for a low number of edges – we use $n = 4$ in this experiment. As proposed by Sigg et al. [SFCP12], we use a fixed iteration number ratio of $m_{max}/m_s = 10$ and perform multiple SA optimizations per m_{max}. The results in Figure 6 show that the SA optimization does converge to a single cost minimum for high enough m_{max}. Moreover, the found solutions are indeed global minima in the graph and are found in a fraction of the time required by naïve search. We use $m_{max} = 200$ in all given examples.

6.4 Timings

To evaluate the performance of our algorithm we measure the amount of time required for each step of our method on an Intel Core i7-2600 3.4GHz Linux PC with eight logical CPU cores and 16GB of main memory. The resulting timings for different data sets and different graph resolutions are given in Table 1. Our method is not able to produce results instantaneously. However, as no user interaction is required, we consider our method to be an offline process for

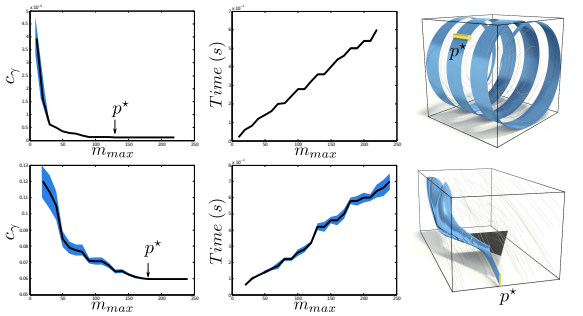


Figure 6: *SA Convergence. For the* SPIRAL *(top) and* DELTAWING *(bottom) data sets we perform 100 SA optimizations per m_{max} and fixed $n = 4$. Graphs show the mean path costs and optimization times (•) and the* 95% *confidence interval (•). SA optimization converges to the globally optimal path $p^\star$ (shown right) for both data sets. Ground truth solution of $p^\star$ are found in* 55*s (*SPIRAL*) and* 61*s (*DELTAWING*), respectively, by a naïve search.*

global data analysis. Still, all results in this work are computed within a few minutes by our method that has no previous knowledge of the given data set. Although we parallelize its computation, it is obvious that the full stream ribbon integration is the most time-consuming part of our method. It depends on both the graph resolution and data set characteristics. For example, integration in the simple TORNADO field ($2.5 \cdot 10^7$ triangles on all ribbons) can be done in a fraction of the time required for the CYLINDER ($1.1 \cdot 10^8$ triangles on all ribbons) data set, although the total number of ribbons is three times higher. Also, edge costs computations are proportional to both the number of stream ribbons and the number of total triangles, but in general they are cheaper than integration. The optimization using the SA algorithm turns out to be the fastest part of our algorithm. This is because the SA optimization reuses the results of previous computations in a highly condensed way in form of edge costs. Moreover, the whole graph is not necessarily visited in the optimization due to the stochastic nature of the algorithm. In the same way our algorithm can be further optimized: surface integration can be deferred by combining the SA optimization with edge cost computations and only needs to be performed when edge costs are evaluated for the first time. This lazy on demand surface integration then only has to be performed for a fraction of the whole domain.

7 Discussion and Limitations

We discretize the search space of seed structure candidates by edges of the domain graph. We will therefore only find solutions that are contained in the graph. We observe that increasing the domain graph resolution does only lead to locally finer solutions in the neighborhood of the coarser solutions. The global location of the optimum does not change as long as the domain is not undersampled. Consider the CYLINDERHIGH surface in Figure 5 as an example for an optimum with increased graph resolution of the CYLINDER data set. Both surfaces have very similar position and shape, although the seed geometry changed slightly. However, the improvements in quality come at a linear expense of performance (see Table 1). We observed that the used graph resolutions are sufficient to find curves that are close to continuous seed structures of globally optimal stream surfaces. An explanation is that for each edge we obtain a *global* optimality estimation due to stream ribbon integration and stream ribbon evaluation. The change in global estimation for a higher graph resolution will only be large at the boundary of separating flow structures, such that in the majority of the domain the estimations stay similar. Therefore, the location of the optimal solution is also unlikely to change due to the higher domain graph resolution.

Data set	DR	Edges	SR	EC	SA $n=4$	SA $n=8$	SA $n=16$
TORNADO	2.0	$\approx 2 \cdot 10^5$	43	11	0.7	1.1	2.2
ACOUTLET	7.2	$\approx 2.5 \cdot 10^5$	308	80	1.6	2.3	3.1
ANEURYSM	na	$\approx 2.2 \cdot 10^5$	270	71	1.7	2.5	4.7
BUBBLECHAMBER	0.5	$\approx 1.1 \cdot 10^5$	60	17	0.9	2.8	6.3
CYLINDER	5.9	$\approx 7 \cdot 10^4$	179	45	0.7	1.6	3.1
CYLINDERHIGH	4.4	$\approx 3 \cdot 10^5$	406	103	1.9	2.8	3.3
DELTAWING	4.0	$\approx 1.4 \cdot 10^5$	170	44	0.6	1.0	2.1
TURBINE	5.7	$\approx 1.8 \cdot 10^5$	253	66	1.4	2.1	2.8

Table 1: *Timings. For each data set the table shows grid cell to graph cell diagonal length ratio (DR), the total number of graph edges, and the* total *computation time (in seconds) of each phase of our algorithm: stream ribbon integration (SR), edge costs computation (EC), and simulated annealing optimization applied to three target edge numbers (SA).*

We optimize for a single stream surface that strives to describe a data set best. This can also be seen as a limitation of our current approach, which is not optimizing for multiple distinct and "distant" stream surfaces *simultaneously*. A simple greedy adaption of our method is to apply the optimization iteratively and modify edge costs according to a distance measure of ribbons to already found optimal solutions. Still, our experience is that multiple stream surfaces quickly tend to occlude each other making visualizations much harder to interpret compared to a single, globally optimal surface. We regard this problem as future research. Additionally, we note that our method will not always converge to a single distinct optimal stream surface in highly turbulent flows. However, it is well-known that surface-based approaches are not suitable to visualize these types of data sets.

8 Conclusions

We presented a novel automatic approach for the selection of stream surfaces in 3D vector fields. Instead of using local flow properties for the selection, our method evaluates global surface-based quality measures on integrated stream surfaces. We showed that our new stream surface quality measure, which is based on flow alignment of principal directions, yields competitive results in a variety of data sets.

Until now we have only used intrinsic surface-based properties to define quality. Investigation of view- and application-dependent surface qualities is an interesting direction for future work, as our selection algorithm can handle

different cost functions without modification. An interesting direction for further research is the extension of our method to time-dependent vector fields to also evaluate and automatically select path, streak, and time surfaces.

Acknowledgments

We thank Tino Weinkauf for resampling the CYLINDER data set. The DELTAWING and ACOUTLET data is courtesy of Markus Rütten, DLR. Axel Seeger and Klaus Affold provided the BUBBLECHAMBER flow. Gábor Janiga simulated the ANEURYSM flow. The TURBINE flow is part of the visualization system AMIRA. The primary author has been partially funded by the Studienstiftung des deutschen Volkes.

References

[AS92] ABRAHAM, SHAW: *Dynamics, the geometry of behavior*, 2nd ed. Addison-Wesley, 1992.

[BFTW09] BÜRGER K., FERSTL F., THEISEL H., WESTERMANN R.: Interactive streak surface visualization on the gpu. *TVCG (Proc. Vis) 15*, 6 (2009), 1259–1266.

[BKP*10] BOTSCH M., KOBBELT L., PAULY M., ALLIEZ P., LEVY B.: *Polygon Mesh Processing*. AK Peters, 2010.

[BWF*10] BORN S., WIEBEL A., FRIEDRICH J., SCHEUERMANN G., BARTZ D.: Illustrative stream surfaces. *TVCG (Proc. Vis) 16*, 6 (2010), 1329–1338.

[CH97] CAI W., HENG P.-A.: Principal stream surfaces. In *Proc. Vis* (1997), pp. 75–81.

[CSBI05] CAMARRI S., SALVETTI M.-V., BUFFONI M., IOLLO A.: Simulation of the three-dimensional flow around a square cylinder between parallel walls at moderate Reynolds numbers. In *AIMETA XVII* (2005).

[CYY*11] CHEN C.-K., YAN S., YU H., MAX N., , MA K.-L.: An illustrative visualization framework for 3d vector fields. *CGF (Proc. PG) 30*, 7 (2011), 1941–1951.

[Dal83] DALLMANN U.: *Topological structures of three-dimensional flow separations*. Tech. rep., DFVLR, 1983.

[DC90] DOOLEY D., COHEN M. F.: Automatic illustration of 3d geometric models: Lines. In *Proc. i3D* (1990), pp. 77–82.

[ELC*12] EDMUNDS M., LARAMEE R. S., CHEN G., MAX N., ZHANG E., WARE C.: Surface-based flow visualization. *C&G 36*, 8 (2012), 974–990.

[ELM*12] EDMUNDS M., LARAMEE R. S., MALKI R., MASTERS I., CROFT T. N., CHEN G., ZHANG E.: Automatic stream surface seeding: A feature-centered approach. *CGF (Proc. EuroVis) 31*, 3 (2012), 1095–1104.

[EML*12] EDMUNDS M., MCLOUGHLIN T., LARAMEE R. S., CHEN G., ZHANG E., MAX N.: Advanced, automatic stream surface seeding and filtering. In *Proc. TPCG* (2012), pp. 53–60.

[GKT*08] GARTH C., KRISHNAN H., TRICOCHE X., BOBACH T., JOY K.: Generation of accurate integral surfaces in time-dependent vector fields. *TVCG 14*, 6 (2008), 1404 –1411.

[GTS*04] GARTH C., TRICOCHE X., SALZBRUNN T., BOBACH T., SCHEUERMANN G.: Surface techniques for vortex visualization. In *Proc. VisSym* (2004), pp. 155–164.

[HGH*10] HUMMEL M., GARTH C., HAMANN B., HAGEN H., JOY K.: Iris: Illustrative rendering for integral surfaces. *TVCG (Proc. Vis) 16*, 6 (2010), 1319–1328.

[Hul92] HULTQUIST J. P. M.: Constructing stream surfaces in steady 3d vector fields. In *Proc. Vis* (1992), pp. 171–178.

[JL97] JOBARD B., LEFER W.: Creating evenly-spaced streamlines of arbitrary density. In *Proc. EG VisSci* (1997), pp. 43–56.

[KBL12] KHABBAZ M., BHAGAT S., LAKSHMANAN L. V. S.: Finding heavy paths in graphs: A rank join approach. *CoRR 1112*, 1117 (2012), 1–16.

[KGV83] KIRKPATRICK S., GELATT C. D., VECCHI M. P.: Optimization by simulated annealing. *Science 220*, 4598 (1983), 671–680.

[KHSI04] KIM S., HAGH-SHENAS H., INTERRANTE V.: Conveying shape with texture: experimental investigations of texture's effects on shape categorization judgments. *TVCG 10*, 4 (2004), 471 –483.

[KKKW05] KRÜGER J., KIPFER P., KONDRATIEVA P., WESTERMANN R.: A particle system for interactive visualization of 3D flows. *TVCG 11*, 6 (2005), 744–756.

[LvWJH04] LARAMEE R. S., VAN WIJK J. J., JOBARD B., HAUSER H.: Isa and ibfvs: Image space based visualization of flow on surfaces. *TVCG 10*, 6 (2004), 637–648.

[MCHM10] MARCHESIN S., CHEN C.-K., HO C., MA K.-L.: View-dependent streamlines for 3d vector fields. *TVCG (Proc. Vis) 16*, 6 (2010), 1578 –1586.

[MLP*10] MCLOUGHLIN T., LARAMEE R. S., PEIKERT R., POST F. H., CHEN M.: Over two decades of integration-based, geometric flow visualization. *CGF 29*, 6 (2010), 1807–1829.

[PGL*12] PENG Z., GRUNDY E., LARAMEE R. S., CHEN G., CROFT N.: Mesh-driven vector field clustering and visualization: An image-based approach. *TVCG 18*, 2 (2012), 283–298.

[PS09] PEIKERT R., SADLO F.: Topologically Relevant Stream Surfaces for Flow Visualization. In *Proc. SCCG* (2009), pp. 43–50.

[RKS00] RÖSSL C., KOBBELT L., SEIDEL H.-P.: Line art rendering of triangulated surfaces using discrete lines of curvature. In *Proc. WSCG* (2000), pp. 168–175.

[Sab10] SABIN M.: *Analysis and Design of Univariate Subdivision Schemes*. Springer, 2010.

[SFCP12] SIGG S., FUCHS R., CARNECKY R., PEIKERT R.: Intelligent cutaway illustrations. In *Proc. PacificVis* (2012), pp. 185–192.

[SGRT12] SCHULZE M., GERMER T., RÖSSL C., THEISEL H.: Stream surface parametrization by flow-orthogonal front lines. *CGF (Proc. SGP) 31*, 5 (2012), 1725–1734.

[SK06] SUMAN B., KUMAR P.: A survey of simulated annealing as a tool for single and multiobjective optimization. *JORS 57*, 1 (2006), 1143–1160.

[SLCZ09] SPENCER B., LARAMEE R. S., CHEN G., ZHANG E.: Evenly spaced streamlines for surfaces: An image-based approach. *CGF 28*, 6 (2009), 1618–1631.

[SRWS10] SCHNEIDER D., REICH W., WIEBEL A., SCHEUERMANN G.: Topology aware stream surfaces. *CGF (Proc. EuroVis) 29*, 3 (2010), 1153–1161.

[SS02] STROTHOTTE T., SCHLECHTWEG S.: *Non-photorealistic computer graphics: modeling, rendering, and animation*. Morgan Kaufmann Publishers Inc., 2002.

[SW04] SWEET G., WARE C.: View direction, surface orientation and texture orientation for perception of surface shape. In *Proc. GI* (2004), pp. 97–106.

[SWH05] SAHNER J., WEINKAUF T., HEGE H.-C.: Galilean invariant extraction and iconic representation of vortex core lines. In *Proc. EuroVis* (2005), pp. 151–160.

[TB96] TURK G., BANKS D.: Image-guided streamline placement. In *Proc. GRAPHITE* (1996), pp. 453–460.

[TWHS03] THEISEL H., WEINKAUF T., HEGE H.-C., SEIDEL H.-P.: Saddle connectors - an approach to visualizing the topological skeleton of complex 3d vector fields. In *Proc. Vis* (2003), pp. 225–232.

[vW93] VAN WIJK J. J.: Implicit stream surfaces. In *Proc. Vis* (1993), pp. 245–252.

[WTHS04] WEINKAUF T., THEISEL H., HEGE H.-C., SEIDEL H.-P.: Boundary switch connectors for topological visualization of complex 3d vector fields. In *Proc. VisSym* (2004), pp. 183–192.