Detecting Topologically Relevant Structures in Flows by Surface Integrals

Wieland Reich
Image and Signal Processing Group
University of Leipzig
PF 100920
04009 Leipzig, Germany
reich@informatik.uni-leipzig.de

Jens Kasten
Image and Signal Processing Group
University of Leipzig
PF 100920
04009 Leipzig, Germany
kasten@informatik.uni-leipzig.de

Gerik Scheuermann
Image and Signal Processing Group
University of Leipzig
PF 100920
04009 Leipzig, Germany
scheuermann@informatik.uni-leipzig.de

ABSTRACT

Gauss' theorem, which relates the flow through a surface to the vector field inside the surface, is an important tool in Flow Visualization. We are exploit the fact that the theorem can be further refined on polygonal cells and construct a process that encodes the particle movement through the boundary facets of these cells using transition matrices. By pure power iteration of transition matrices, various topological features, such as separation and invariant sets, can be extracted without having to rely on the classical techniques, e.g., interpolation, differentiation and numerical streamline integration. We will apply our method to steady vector fields with a focus on three dimensions.

Keywords

Surface integrals, vector field topology, flow visualization, transition matrices, stochastic processes

1 INTRODUCTION

Vector field topology reveals the basic features of a flow field, i.e., critical points, separatrices, separation surfaces and other invariant manifolds. For instance, it became a widely used tool for the feature-based analysis of stationary flow fields. For time-dependent flow fields, the finite-time Lyapunov exponent (FTLE) is often used to extract time-dependent counterparts of the structures known from vector field topology. Nevertheless, methods to extract vector field topology are based on interpolation, differentiation and numerical streamline integration.

Instead of relying on streamlines, one can use transition matrices. The entries of a transition matrix represent the probability that particles contained in one cell will enter another cell after some advection time. This leads to the theory of time-discrete Markov processes, which are a widely studied object of probabilistic theory. Here, the question is, if powers of the transition matrix will converge to a limit state. This corresponds to particles reaching a limit set, e.g., a critical point.

The generation of the transition matrices is a hard task. In [22], Reich and Scheuermann have used a combinatorial flow map, that was introduced by Chen et al. [3] to compute the outer approximation of an integration image of a cell after some advection time. Then, they were able to compute probabilities of particles leaving one cell and entering another cell. To compute the outer approximation, all edges of a cell have to be integrated adaptively.

In this paper, we present a novel algorithm to compute the transition matrices. The idea is to look at the outflow region of every cell. For two adjacent cells in 2D, the probability of the transition can be computed by relating the outflow at the edge to the outflow of the whole cell. This approach can be naturally extended to 3 dimensions by integrating the outflow along the cell surface instead of the edges.

While the approach using the method of Chen might result in transitions between cells that are not neighbored, our novel approach guarantees transitions only between neighbored cells. We therefore sample the transitions at the finest scale possible in this discrete setting. The speed of convergence of the transition matrices depends linearly on this scaling, because the computationally most costly stage is multiplying sparse matrices with vectors, which rises linearly in the number of cells, resulting in fast computations of the attracting and repelling limit sets. However, computing separation is still a computationally costly technique, because much more vectors will have to be iterated.

Our method is evaluated using data sets of different levels of complexity. Due to the extension to 3 dimensions of the existing work, we are now able to analyze more real world data with our novel approach.

The remainder of this paper is structured as follows: In section 2, we present approaches that are related to our method. The basics of surface integrals and transition matrices are explained in section 3. Afterwards, we present our method in section 4. The results are pre-

sented and discussed in section 5. The paper concludes with section 6.

2 RELATED WORK

Vector field topology has been introduced to by Helman and Hesselink [11] more than 20 years ago to the visualization community. Since then a lot of research has been devoted to the extraction of topological invariants. Initially, Helman and Hesselink extracted critical points and classified the points by linearizing the flow in the vicinity. Afterwards, the flow is segmented into basins of similar flow behavior using the separatrices that emanate from the saddle points. This method was then extended by several researchers. Scheuermann et al. [25] analyzed the boundary of the domain to extract a finer topology. Weinkauf et al. [32] extended this approach to three dimensions. Theisel et al.[29] introduced the concept of saddle connectors, which are the intersection of the separation surfaces emanating from the saddles points. Wischgoll et al. [34] extracted attracting and repelling periodic orbits in planar flows by searching for cell cycles.

Tricoche [31] studied the connection between the Poincaré index and vector field topology. Polthier et al. [20] use a discrete hodge decomposition to extract vector field singularities. Bhathia et al. [1] use edge maps as an alternative to streamlines.

The aforementioned methods analyze the vector field as a continuously given data set. This approach does not incorporate the grid and makes interpolation necessary. In contrast, discrete methods do not rely on interpolations but analyze the raw data. In particular, the work of Reininghaus [23] covers the extraction of topological structures in combinatorial vector fields. Here, the grid is represented as a graph and the vector field as an matching on this graph. Their method is based on the work of Forman [6]. Another combinatorial approach was presented by Chen et al. [3]. They use the images of triangular cells under advection to encode the flow field into a graph. Conley index theory is used to classify the strongly connected components as features. The work of Boczko et al. [5] can be seen as a special case of a Morse decomposition. Szymczak presented Morse decompositions of piecewise constant vector fields [28].

An approach to compute vector field topology in a discrete setting was presented by Reich et al. [22]. They use the theory of Markov processes [27] to extract vector field invariants.

In the last two decades, the extraction of topological structures in uncertain vector fields had come to attention in the flow visualization community, e.g., see Pang et al. [18]. Otto et al. [17] formulated convergence criteria for gaussian distributed density functions by Euler integration. Their method also uses the uncertain Poincaré index to distinguish between critical distributions. Petz et al. [19] presented an approach to analyze the probability of a critical point to be contained in a cell for uncertain vector fields. Schneider et al.[26] uses principal component analysis to detect separation in uncertain flows.

The list of the aforementioned publications related to vector field topology is naturally not complete. For a good overview that also cover topics of flow visualization, we refer to Weiskopf and Erlebacher [33], and Post et al. [21].

For time-dependent flow fields, vector field topology is not sensible to extract anymore. Here, a lot of analysis approaches search for Lagrangian coherent structures. An important approach to find these features was introduced by Haller [10] by introducing the Finite-time Lyapunov exponent (FTLE). Within the visualization community a lot of computational improvements or alternative computation methods have been proposed, see [2, 7, 13]. There is also some research done to compare vector field topology to structures extracted from the FTLE, e.g., see [24].

The stochastic processes in our work are also an important tool for image segmentation and pattern analysis [8].

3 MATHEMATICAL PRELIMINARIES

3.1 Surface Integrals

Surface integrals can be described as an observable quantity that measures the amount of leaving (entering) flow through a bounded surface in one time step. A famous theorem related to surface integrals is from C.F. Gauss. It states, that the flow f through a piecewise differentiable boundary of an area Ω is equal to divergence integral over the enclosed Ω:

$$\int_{\partial\Omega} < f, n > dA = \int_{\Omega} div\ f\ dV, \qquad (1)$$

where n denotes the outer unit normal to $\partial\Omega$ and $< ., . >$ the inner product. The left side is a surface integral, while the right side is a integration over a volume.

The theorem has many applications in other sciences, e.g., in electrodynamics it implicates that there can be no electric field inside a hollow object. In one dimension, it is equivalent to the fundamental theorem of calculus.

In this chapter, we are going to introduce surface integrals in the Euclidean spaces $\mathbb{R}^2$ and $\mathbb{R}^3$. We will show that for triangular and tetrahedral cells in a piecewise linear (or piecewise constant) flow, the surface integrals reduce to (relatively) simple formulas. For interpolation schemes of higher order, there is no guarantee that there exists a closed formula, but the flow integral can still be calculated by using numerical integration techniques, like the Gaussian quadrature.

3.1.1 The 2D-Case

Since the boundary of a triangular cell in two dimensions is a closed path, the surface integral reduces to a line integral.

Lemma:
Let $f(x) : \mathbb{R}^2 \to \mathbb{R}^2$ be an affine linear field, i.e., it is of the form $A(x)+b$, with a matrix A and a constant vector b. Let p_1 and p_2 be two positions that bound an edge e of a triangle in $\mathbb{R}^2$. Then the surface integral of the flow f through e is

$$\int_e < f,n > dx = ||p_1 - p_2||_2 \cdot < \frac{f(p_1)+f(p_2)}{2}, n >, \tag{2}$$

where $||p_1 - p_2||_2$ is the length of edge e and n the outer unit normal of the edge.

3.1.2 The 3D-Case

Lemma:
Let $f(x) : \mathbb{R}^3 \to \mathbb{R}^3$ be an affine linear field again. Let T be a triangle in space, e. g. the face of a tetrahedron, with vertices positions p_1, p_2 and p_3. Then the surface integral of the flow f through T is

$$\int_T < f,n > d\sigma = \mathscr{A}(T) \cdot < \frac{f(p_1)+f(p_2)+f(p_3)}{3}, n >, \tag{3}$$

where $\mathscr{A}(T)$ is the area $\frac{1}{2} \cdot ||(p_1 - p_3) \times (p_2 - p_3)||_2$ spanned by the triangle T.

For the following sections, in particular 4.1, we rely on our gained, ready to implement formulas. Readers interested in the theory of multidimensional integration might also have a look in any vector calculus book, e.g., [16].

3.2 Transition Matrices

3.2.1 What are transition matrices?

In this section, we are going to give an insight in the basics of transition matrices, also called time-discrete Markov chains.
These matrices are linear operators that map a distribution vector v_1 to another distribution vector v_2 of the same dimension, preserving that every entry in the vector is greater or equal to zero and the sum of all entries is 1. In particular, a row-stochastic transition matrix M has the property, that all entries are greater or equal to zero and the sum of all entries in each row is 1, i.e.,

$$\sum_j m_{ij} = 1.$$

The entry m_{ij} describes the probability of the system from going from state i to state j.
Transition matrices are a stochastic processes with only a finite number of states which coincide with the size of matrix. The image of a distribution vector after k discrete time-steps is generated by multiplying the transposed distribution vector from the left n times. We have

$$v_{k+1}^T = v_k^T \cdot M. \tag{4}$$

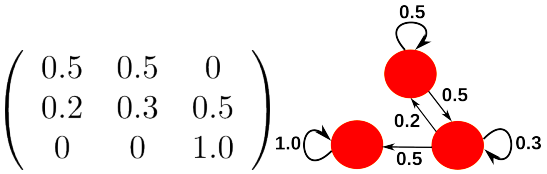
As a sidenote, if we want to have a multiplication of the vector from the right, then our matrix M has to be column-stochastic instead. We have chosen the row-stochastic form, because it seems more intuitive when we move from state i to j than from j to i.
As one can see easily, the operator M is memoryless, i.e., the next state of the system only depends the current state, not on those before. The spectrum of M and the long term behavior of multiplication operations are of particular interest in probability theory [27].
Transition matrices may have many stationary states, these are vectors that do not change by multiplication formula (4). As a consequence, they have to be (left-) eigenvectors to the eigenvalue of 1:

$$v_k^T \cdot 1 = v_k^T \cdot M$$

The existence of at least one eigenvalue 1 is guaranteed for every M.
Though M and all distribution vectors v_k will always be bound in their norm by 1, not all multiplications involving transition matrices converge to a stationary state by power iteration. We will present a solution for this issue in section 4.
One of the most important theorems related to transition matrices is by Perron and Frobenius [27]. It states that if every entry in the matrix M is greater than zero, then there exists a unique eigenvector to the eigenvalue $\lambda = 1$ and every power iteration algorithm will converge to that eigenvector, which is the only stationary state.
Google [14] makes use of the Perron-Frobenius-Theorem to construct transition matrices that are guaranteed to converge. The states in the Google matrix are websites and the transition probabilities are determined by hyperlinks, that guide the user from one site to another. Further, there exists a very small chance, that the user chooses a completely random website, so the Google-Matrix will be densely populated. As a consequence, the unique stationary state of the Google matrix can be calculated by power iteration and delivers the page rank of each site, a measure for its importance, that can be used to order the results of search requests by the user.
For the case of interpretation difficulties of transition matrices, it often helps to sketch a probabilistic graph that describes the movement, e.g., like in Figure 1.

3.2.2 Relation to Vector Field Topology

Transition matrices have been used before to extract topological features of a flow induced by a vector field.



(a) (b)

Figure 1: One example that expresses the duality between a (row-stochastic) transition matrix (a) and a probabilistic graph (b). The probability m_{ij} describes the likelihood to move from node i to node j.

The main idea of the preceding approaches was to encode particle movement from numerical streamline integration to a transition probability from cell to cell. Two important publications in that context are [4] and [22].
There is a duality between both dynamical systems. For a flow $\phi(t,x)$ associated with a vector field we have

$$\phi(0,x) = x$$

and

$$\phi(t_1,\phi(t_2,x)) = \phi(t_1+t_2,x).$$

For a (row-stochastic) transition matrix

$$v^T \cdot M^0 = v^T$$

and

$$(v^T \cdot M^{n_1}) \cdot M^{n_2} = v^T \cdot M^{n_1+n_2}$$

holds. Further, the product of any(!) transition matrices M_1 and M_2 of the same dimension is a transition matrix again.
The conversion of a continuous system to a a discrete system poses several challenges:
While the flow $\phi(x,t)$ is invertible in the range of its existence, i.e., we are able to go back to our original position by an integration using the same time with a negative sign, transition matrices do not need to be invertible. A possible solution is to construct two transition matrices, one describing the forward movement (M_+) and one for the backward movement (M_-). However, it is still not guaranteed that the equation

$$M_+ \cdot M_- = Id$$

is fulfilled here; so it is natural to ask why we convert the flow to that discrete system. The reason is not only the gained robustness and uniform treatment of invariant sets. Transition matrices allow us to analyze the sensitivity of the initial value problem at infinite times, which is not possible with purely particle-distance-based algorithms like FTLE. In publication [22], Reich et al. make use of that fact and extract separating features of planar flows.

4 THE ALGORITHM

Now we are going to combine the surface integrals with transition matrices, i.e., we move from local feature extraction to a global topology by describing the interaction between the cells and the flow through their common facettes.
Unlike the preceding work, our algorithm will be completely independent from numerical streamline integration and works in any dimension. However, our presented results will primarily contain 3-dimensional flows, while, for the sake of simplicity sub-steps of the method are illustrated in 2D.

4.1 Encoding Particle Movement

Recall the Gauss Theorem (1). If we look at the right side, we have a volume integral over the divergence of a region Ω, say, a cell of a piecewise linear vector field. While the integral can be zero, e.g., the cell contains a purely rotational stationary point, there are still particle movements between the cell and its neighbor cells. So the right side is of no use when we want to create transition matrices. Let us have a look at the surface integral instead:

$$\int_{\partial\Omega} < f,n > dA.$$

This can be further refined to

$$\sum_i \int_{(\partial\Omega)_i} < f,n > dA,$$

where $(\partial\Omega)_i$ is a boundary segment of our cell, i.e. an edge i of a triangle, or a face i of a tetrahedron. Further, we can refine the formula by distinguishing between in- and outflow:

$$\sum_i \int_{(\partial\Omega_+)_i} < f,n > dA \; + \sum_i \int_{(\partial\Omega_-)_i} < f,n > dA,$$

where $(\partial\Omega_+)_i$ is the region where $< f,n > \geq 0$ holds (outflow), and $(\partial\Omega_-)_i$ the region where $< f,n >$ is smaller than 0 (inflow).
It follows immediately from our formulas, that if we have a piecewise linear flow that is divergence-free, then

$$\sum_i \int_{(\partial\Omega_+)_i} < f,n > dA = -\sum_i \int_{(\partial\Omega_-)_i} < f,n > dA$$

must hold for every cell.
By assuming a linear field, the flow relative to each boundary edge/facette can change its behavior just once, at a tangential point in 2D, or a tangential line in 3D. The tangential point, respectively the endpoints of the tangential line can computed by seeking a solution λ in $[0..1]$ satisfying

$$\lambda = \frac{< n, f(p_2) >}{< n, f(p_2) - f(p_1) >}$$

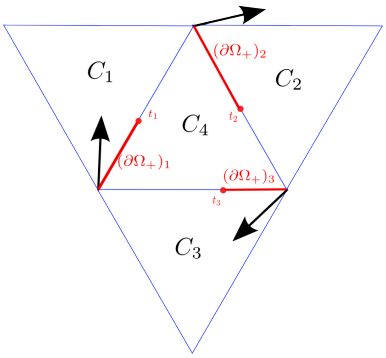


Figure 2: An example of encoding particle movement. All three edges of the cell C_4 have a non-trivial exit set. We compute the integrals of $< f,n >$ by our formulas from section 3 using the vertices and the tangential points as integration range (red lines). The transition probability from cell C_4 to C_1 is determined by the value of the surface integral on $(\partial\Omega_+)_1$ divided by the value on $\sum_i (\partial\Omega_+)_i$.

on an edge spanned by p_1 and p_2. Afterwards, the tangential point t can be computed by $\lambda \cdot p_1 + (1-\lambda)\cdot p_2$. The vector at a tangential point or a point on a tangential line is never needed to be evaluated, because it has no component in direction of the normal, so it can be assumed as zero in our integration formulas. We do not need interpolation at any sub-step of the algorithm.
As an example, the surface integral for a edge with tangential point t in 2D splits up into

$$\begin{aligned}\int_e < f,n > dx &= \int_{e_+} < f,n > dx + \int_{e_-} < f,n > dx \\ &= ||p_1 - t||_2 \cdot < \frac{f(p_1)+0}{2}, n > \\ &+ ||t - p_2||_2 \cdot < \frac{0+f(p_2)}{2}, n >,\end{aligned}$$

where it still has to be checked which term is the positive part.
In case of a face of a tetrahedron, a tangential line decomposes a boundary face into a triangle and a quad. The latter one can again be decomposed into two triangles, so it is necessary to evaluate the 3D-surface integral three times per face. If the flow is transverse at a boundary edge/face, which is the common case, in particular for flows with weak rotation, we can compute the surface integral directly in one step. For a 2D-illustration of a evaluation also see Figure 2.

The following is the key aspect of the whole paper. We are going to put the outflow through a boundary edge/face in relation to the outflow of the whole cell. Which yields the quotient "outflow through a face i that connects cell a with cell b" divided by "outflow through the whole cell a", or

$$m_{ab} = \frac{\int_{(\partial\Omega_+)_i} < f,n > dA}{\sum_i \int_{(\partial\Omega_+)_i} < f,n > dA}, \qquad (5)$$

where $\partial\Omega$ is the boundary of cell a and face i is connecting a with b. The values of m_{ab} fill our transition matrix M_+. We can state a analogous formula for M_- by just substituting $\partial\Omega_+$ with $\partial\Omega_-$.
If we sum up all m_{ab} from a cell a and all of its neighbors b, the result will always be 1.0, so M_+ and M_- will be transition matrices, that describe the weighted outflow/inflow of a linear flow through a cell.
To avoid division by zero, we must intercept the cases, where there is no outflow/inflow at all. These cases are cells containing nodal stationary points, so we just set m_{aa} to 1.0 and all m_{ab} are 0.0 for $a \neq b$.

4.2 Transition Matrix Processing

From now on, the rest of the algorithm will be pure matrix-vector-iterations with our constructed transition matrices M_+ and M_-.
Since the amount of cell neighbors is always limited to 3 (in 2D) or 4 (in 3D), our transition matrix will be sparse of the compressed size $(N\times 3)$ or $(N\times 4)$, where N is the number of cells in our dataset. There are programs with proper data structures [9] who are designed for the procession of sparse matrices. Alternatively, one could also use an own implementation, e.g., using the construct $vector < map < unsigned\ int, float >>$ in C++, which also has been used to process the transition matrices generated from the datasets in this paper. We did not experience any significant computational time changes when switching from our classes to [9]. The complexity of a matrix-vector-multiplication is reduced from $O(N^2)$ to $O(N)$, when the matrix is not densely populated.
The power iteration is a task always to be executed the same way:
Given an equivalence precision $\varepsilon > 0$ and an initial vector v_0 we compute

$$v_{k+1}^T = v_k^T \cdot M,$$

until $||v_{k+1} - v_k|| < \varepsilon$.
A small ε will lead to accurate results but can increase the computational time significantly. A summary on multiple power iteration methods for matrices can be found in [30].
We are going to use three types of useful initial distributions:

- The uniform distribution $v_0 = u$, where all entries of u are $\frac{1}{N}$ with N being the number of cells.
- The impulse distribution $v_0 = e_i$, where the i-th position of the vector contains a 1.0 and all others are 0.0. This distribution is localized in cell i only.
- The neighborhood distribution $v_0 = n_i$, where for all neighbors j of a cell i, who share common edges/faces, have the value $\frac{1}{3}$ (in 2D) or $\frac{1}{4}$ (in 3D).

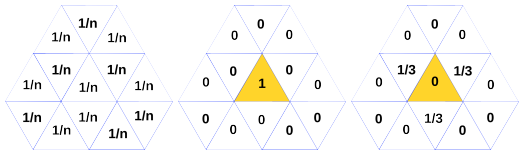


Figure 3: All three distributions that were used by us for power iterations: (a) a uniform distribution, n is the number of cells, (b) an impulse distribution, which is only located in the yellow cell, (c) a neighborhood distribution, located around the yellow cell, for a tetrahedron, the values have to be $\frac{1}{4}$.

This distribution is localized in the neighborhood of cell i in triangular or tetrahedral mesh.

An illustration of all distributions can be found in Figure 3. We do not consider cells as 'proper' neighbors if they share just a common vertex, because the resulting flow integrals will be trivial on a set with measure 0.

4.2.1 *Invariant Sets*

Since we are no longer moving in the conventional dynamical system $\phi(x,t)$, which is induced by a vector field, we must redefine the invariant sets:
An invariant set is an eigenvector of either M_+ or M_- to the eigenvalue of 1. An invariant set is attracting, if the matrix is M_+, else it is repelling.
As a remark, these are indeed invariants, because if we see the eigenvector as a collection of cells, then the set of these cells, that are represented by non-zero entries, does not change by any new multiplication with the transition matrix. The set of all cells in an invariant set is always connected, because the movement between those cells always takes place between their common edges/faces.
To extract all attracting stationary states, we just need to iterate with M_+ and the initial vector u, which represents a distribution over the whole domain. For the repelling ones we take M_-. We experienced that purely rotational stationary points can be extracted, together with their neighborhood, by both methods.
Using u as an initial vector also has the side effect, that sinks and sources are also weighted with the size of their α/ω-basin. Some invariants might attract or repel "more" particles than others.

4.2.2 *Separation*

Separation manifolds cannot be automatically derived from the spectrum of our transition matrices. However, we still are able to do a power iteration with the initial vectors e_i and n_i and compare the resulting stationary states by the l_1-metric and eventually measure the dependence of the iteration process from the initial vector. We obtain forward separation by using M_+ and backward separation by using M_-. That method has by far the highest complexity, which can be estimated by the computational time of extracting all invariants times the number of cells in the dataset. Separation features extracted by transition matrices have to be seen in a global context by describing particles that tend to have different limit sets. Every finite-time expansion of flow, as well as particle distances, will not influence our result.

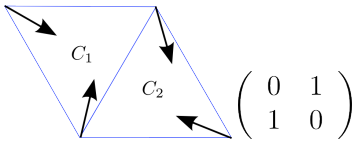


Figure 4: A possible case that could, without using equation (6), lead to a divergent transition matrix : (a) there is clearly a singularity in one of those cells, but if the flow leaves cell C_1, it will be towards C_2 with probability 1.0 and vice versa. The resulting matrix (b) will be divergent, except one chooses the initial distribution $(0.5, 0.5)$.

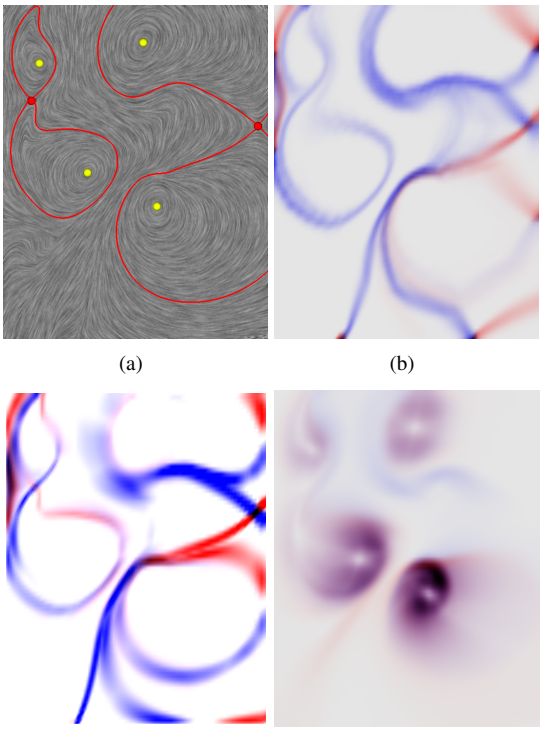
4.2.3 *Boundary Topology*

We process the boundary similar to the method by Mahrous et al. [15]. First, we extract all connected boundary segments, where the flow leaves the domain. These segments will be treated like additional cells and have mapping of probability 1.0 to themselves. Finally all cells adjacent to them, will be mapped to these artificial invariants by the probability that is, as in the ordinary case, determined by the outflow integral. The size of the matrix will grow by the number of so-called exit sets of the domain, which are in general of much smaller cardinality than the number of cells in our dataset.

4.3 How to prevent divergent transition matrices

Transition matrices that are generated from surface integrals may be divergent, i.e., not every stationary state may be reached by power-iteration only. Most of these cases are clusters of cells that are ordered in a cycle, where the transition probability from one cell to its successor is 1.0. If one puts an impulse distribution in one of these cells, the power-iteration will just move that distribution around the cycle without reaching a stationary state. A special case is given in Figure 4, where a critical point near the common edge of two cells leads to a divergent 2-cycle.
However, we are able to perform one simple operation, so that a new matrix M_{new} will have the same stationary states, but will be convergent by potentiation. We set

$$M_{new} = \frac{M_{old}^2 + M_{old}}{2} \tag{6}$$



(a) (b)

(c) (d)

Figure 5: A CFD-dataset simulating a fluid entering at the left bottom: (a) 4 center points (yellow) and 2 saddles (red) with their separatrices, the left ones form two homoclinic orbits, the background is a LIC, (b) separation computed with a streamline-based method from the preceding work ([22]), (c) separation computed with surface integrals by iterating all distributions e_i and n_i, (d) iteration with uniform distribution u reveals the center points.

and obtain a process with the same stationary states. It can be easily shown that the only remaining eigenvalue in M_{new}, that has magnitude 1, is 1 itself. There are no other eigenvalues on the unit circle of the complex plane present, which could cause divergent behaviour. Our new transition matrix will always converge by power iteration. In the special case of Figure 4, the resulting new matrix block has only the entries 0.5 and both cells belong to the same stationary state which is associated with the critical point of the vector field.

It is not necessary to compute the explicit second power in practice, the propagated iteration scheme just needs to be extended by another step with an additional averaging of two distribution vectors. As a side effect, vector fields containing highly rotational flows and many closed streamlines are also faster processed, because distributions get blurred immediately along the probabilistic streamline.

The discrepancy to the former streamline-based method in [22] is, that transition matrices that have been generated by flow maps are practically never divergent, because a flow map never maps one cell with a probability of exactly 1 to another cell.

Google [14] uses a similar method when computing the page rank of websites. Their so-called matrix damping formula is

$$\widetilde{M}_{google} = (1-\alpha)\cdot M + \alpha \cdot G, \tag{7}$$

where $G = \{\frac{1}{N}\}$, N being the size of G.

The convergence is ensured by the Perron-Frobenius-Theorem [27] there. If we would have wanted to apply such a convex linear combination to our problems, the consequences would be devastating. Not only that the resulting dense matrices make efficient computations with large datasets extremely costly. The resulting matrix from (7) has an unique stationary state and all power iterations will converge to that eigenvector. Measuring separation will be impossible. Finally, our formula is superior in the feature extraction of flows, because it preserves the low population of entries in the sparse matrices and the multidimensionality of the Eigenspace of eigenvalue 1.

Another interesting aspect is a geometric interpretation of M^{∞}_{new}, which is now well-defined. We already know that the norm of transition matrices is bounded by 1, implicating the same for all (possibly complex) eigenvalues . If we consider

$$M^{n}_{new} \cdot v = \lambda^{n} \cdot v$$

for $n \to \infty$, all λ obeying $|\lambda| < 1$ will be set to zero. From the existence of M^{∞}_{new} it can be excluded that, with the exception of 1, there are any other eigenvalues of magnitude 1. Eventually 1 and 0 are the only "surviving" eigenvalues in M^{∞}_{new}, which can now be considered as a projection operator. We project an initial distribution into the Eigenspace of eigenvalue 1, which is spanned by all stationary distributions. In fact, all of our used power iterations are projections.

5 RESULTS AND DISCUSSION

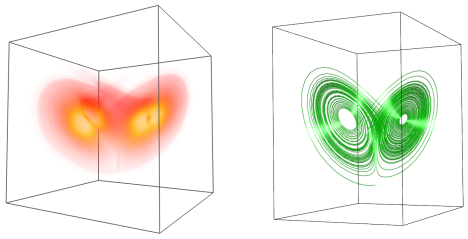
All iteration methods result in scalar data which is either visualized by a color map (2D) or volume rendering and isosurfaces (3D).

5.1 Artificial Data

5.1.1 The Lorenz Attractor

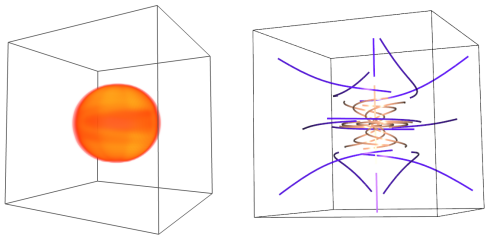
The Lorenz-Attractor was discovered when E.N. Lorenz attempted to set up a system of differential equations that would explain some of the unpredictable behavior of the weather. It is one of the most popular chaotic systems featuring a dense collection of unstable streamlines. The attractor in our example obeys the ODE-system

$$\begin{aligned} x' &= 10(y-x) \\ y' &= x(28-z)-y \\ z' &= xy - \frac{8}{3}z, \end{aligned}$$



(a) (b)

Figure 6: The Lorenz-Attractor: (a) The uniform distribution u is iterated by a transition matrix that was generated by surface integrals, (b) illuminated streamlines were planted in the detected region.



(a) (b)

Figure 7: The Invariant Sphere: (a) e_i and n_i were cellwise iterated by a transition matrix generated by surface integrals, (b) streamlines were planted in various positions, the violett colored ones were outside of the domain $|r| = 1$ and diverge while the red colored ones were inside and remain in the sphere for all integration times.

and can be seen in Figure 6. We had chosen the uniform distribution and used power iteration with M_+.

5.1.2 *The Invariant Sphere*

The invariant sphere can be produced by using the ODE-system

$$\begin{aligned} x' &= -y + x \cdot (1-r) \\ y' &= x + y \cdot (1-r) \\ z' &= z \cdot (1-r), \end{aligned}$$

where $r = \sqrt{x^2+y^2+z^2}$. It consists of an infinite number of dense closed streamlines for $r = 1$, every streamline seeded in the neighborhood of the sphere will converge to one of these closed orbits. Low-precision integrators tend to produce unstable solutions here. We computed the backward separation with M_- and successfully extracted the whole sphere, which is visible in Figure 7.

5.2 CFD-Simulations

5.2.1 *Swirling Jet*

Figure 5 will be our only example of a 2D-dataset. In contrast to the preceding work, our focus will be on 3 dimensions. We computed forward and backward separation with M_+ and M_- and extracted the topological skeleton of the fluid simulation quite accurate. Moreover, the surface integral method seems to extract boundary related features better than the streamline-based approach in [22].

5.2.2 *Gas Furnace Chamber (Velocity)*

In Figure 8 we analyzed the backward separation field of the gas furnace chamber. The gas furnace chamber is a divergence-free vector field on a grid that contains approximately $2 \cdot 10^5$ cells. The forward separation field is of minor interest, because all particles will end in the same exit set. High separation values around the injectors were detected and their influence to the topology of the field is also revealed.
The global separation is hard to interpret by seeding stream surfaces in the stable and unstable manifold of saddle points, because there are simply too many of them inside the vector field.

6 CONCLUSION AND FUTURE WORK

We presented a novel approach to the topology of steady 3D vector fields by exploiting that surface integrals can be expressed as simple formulas on piecewise linear vector fields. We constructed transition matrices by the information of these integrals, which allow an infinite-time evaluation of separation and are able to extract many topological features of 3D flows without having to rely on numerical integration schemes, e.g., a forth-order Runge-Kutta. The latter advantage develops into great robustness towards classical problems, like critical points located near the boundary of cell, boundary slip conditions, and stiffness problems of ordinary differential equations. Transition matrices are much easier constructed with surface integrals in any dimension than with the streamline based approach, which is still a not completely solved task for a tetrahedral mesh ([3], [22]). While the surface integral based method produces smoother separating structures in locations near the boundary of the domain, both methods do not differ much in their high computational times, which can be several hours or even days for large data.
Further, we neither need a differential operator, nor evaluation (interpolation) of values outside of our vertices in our grid. The algorithm also includes the boundary of our domain into its calculations.
Regarding computational costs, the topology of the vector field is much more influential than the number of cells. Distributions in gradient fields converge very quickly to a stationary state. Highly rotational fields take much longer.
The prospects on the following work can be subdivided into three branches:

1. **Reduction of the Computational Costs** To make significant progress in reducing the computational time, a GPU-implementation will be necessary. To the best of our knowledge, prevalent GPU-based linear algebra software parallelizes row- and column-operations of matrix-vector-products. However, what we need is a parallelization in a much more extensive context, i. e., allowing multiple vectors being operated on by the same matrix.
   Further, one might think about a better initial distribution seeding, e. g., similar to a divide and conquer approach, so that we do not have to iterate each single cell by an impulse distribution.

2. **Uncertainty** Because transition matrices are a special type of stochastic processes, it would make them a very useful tool to explore uncertainty in dynamics, which has been stated as one of the most important branches in the future of visualization [12]. We do not need to change our method at any stage for that. We can manipulate initial distributions, or even our matrix, in any way we want and study the changes that they create.

3. **Time-Dependent Flows** The fact that we are able to process steady 3D vector fields automatically opens the gates to a method, which can create transition matrices of a time-dependent 2D vector field by joining all time slices in their order. Unfortunately, the concept of stationary states of distributions works only, if the time-dependent field is periodic.

7 REFERENCES

[1] H. Bhatia, S. Jadhav, P.-T. Bremer, G. Chen, J. A. Levine, L. G. Nonato, and V. Pascucci. Edge maps: Representing flow with bounded error. In *PacificVis*, pages 75–82, 2011.

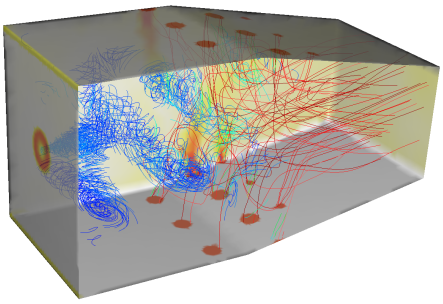
[2] S. L. Brunton and C. W. Rowley. Fast computation of finite-time lyapunov exponent fields for unsteady flows. In *Chaos*, volume 20, 2010.

[3] G. Chen, K. Mischakow, R.S.Laramee, and E. Zhang. Efficient morse decompositions of vector fields. In *IEEE Transactions on Visualization and Computer Graphics*, volume 14, pages 848–862, 2008.

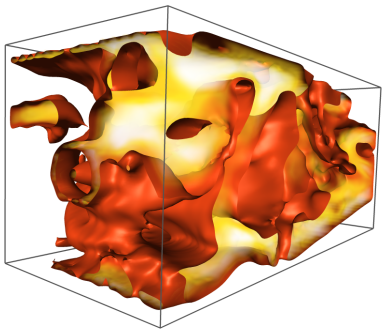
[4] M. Dellnitz and O. Junge. On the approximation of complicated dynamical behavior. In *SIAM Journal on Numerical Analysis*, volume 36, pages 491–515, 1999.

[5] K. M. E. Boczko, W. Kalies. Polygonal approximation of flows. 2004.

[6] R. Forman. Morse theory for cell complexes. In *Advances in Mathematics*, volume 134, pages 90–145, 1998.



(a)



(b)

(c)

Figure 8: The gas furnace chamber: (a) One main injector located on the left, where the blue streamlines were seeded. Many smaller injectors on the bottom and top of the chamber, where the red streamlines are entering. (b) Backward separation field of the gas furnace chamber: isosurfaces on the highest separation values, (c) volume rendering on the same dataset with a viewpoint from above.

[7] C. Garth, A. Wiebel, X. Tricoche, K. Joy, and G. Scheuermann. Lagrangian visualization of flow-embedded surface structures. In *Computer Graphics Forum*, volume 27, pages 767–774, 2008.

[8] L. Grady. Random Walks for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, 2006.

[9] G. Guennebaud, B. Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[10] G. Haller. Distinguished material surfaces and coherent structures in three-dimensional flows. In *Physica D*, volume 149, pages 248–277, 2001.

[11] J. Helman and L. Hesselink. Visualizing vector field topology in fluid flows. In *IEEE Computer Graphics and Applications*, volume 11, pages 36–46, 1991.

[12] C. Johnson. Top scientific visualization research problems. In *IEEE Computer Graphics and Applications*, volume 24, pages 13–17, 2004.

[13] J. Kasten, C. Petz, I. Hotz, B. Noack, and H.-C. Hege. Localized finite-time lyapunov exponent for unsteady flow analysis. In *Proceedings Vision, Modeling and Visualization 2008*, pages 265–274, 2009.

[14] A. N. Langville and C. D. Meyer. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, 2006.

[15] K. M. Mahrous, B. H. J. C. Bennett, and K. I. Joy. Improving topological segmentation of three-dimensional vector fields. In *IEEE TCVG Symposium on Visualization*, 2003.

[16] J. Marsden and A. Tromba. *Vector Calculus*. W. H. Freeman, 2003.

[17] M. Otto, T. Germer, H.-C. Hege, and H. Theisel. Uncertain 2d vector field topology. In *Computer Graphics Forum*, volume 29, pages 347–356, 2010.

[18] A.-T. Pang, C.-M. Wittenbrink, and S.-K. Lodh. Approaches to uncertainty visualization. *The Visual Computer*, 13:370–390, 1996.

[19] C. Petz, K. Pöthkow, and H.-C. Hege. Probabilistic local features in uncertain vector fields with spatial correlation. *Computer Graphics Forum*, 31(3):1045 – 1054, 2012.

[20] K. Polthier and E. Preuss. Identifying vector field singularities using a discrete hodge decomposition. pages 112–134. Springer Verlag, 2002.

[21] F. Post, B. Vrolijk, H. Hauser, R. Laramee, and H. Doleisch. The state of art in flow visualization: Feature extraction and tracking. In *Computer Graphics Forum*, volume 22, pages 775–792, 2003.

[22] W. Reich and G. Scheuermann. Analysis of streamline separation at infinity using time-discrete markov chains. *IEEE Transactions on Visualization and Computer Graphics*, 18, 2012.

[23] J. Reininghaus and I. Hotz. Combinatorial 2d vector field topology extraction and simplification. In *Topology in Visualization*, 2010.

[24] F. Sadlo and D. Weiskopf. Time-Dependent 2-D Vector Field Topology: An Approach Inspired by Lagrangian Coherent Structures. *Computer Graphics Forum*, 29(1):88–100, 2010.

[25] G. Scheuermann, B. Hamann, K. Joy, and W. Kollmann. Visualizing Local Vector Field Topology. *Journal of Electronic Imaging*, 9(4):356–367, 2000.

[26] D. Schneider, J. Fuhrmann, W. Reich, and G. Scheuermann. A variance based ftle-like method for unsteady uncertain vector fields. In *Topological Methods in Data Analysis and Visualization II*, pages 255–268, 2012.

[27] W. J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[28] A. Szymczak and E. Zhang. Robust Morse Decompositions of Piecewise Constant Vector Fields. *IEEE Transactions on Visualization and Computer Graphics*, 18(6):938–951, 2012.

[29] H. Theisel, T. Weinkauf, H.-C. Hege, and H.-P. Seidel. Saddle connectors - an approach to visualizing the topological skeleton of complex 3d vector fields. In G. Turk, J. J. van Wijk, and R. Moorhead, editors, *Proc. IEEE Visualization 2003*, pages 225–232, Seattle, U.S.A., October 2003.

[30] L.-N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, 1997.

[31] X. Tricoche. *Vector and Tensor Field Topology Simplification, Tracking, and Visualization*. PhD thesis, 2002.

[32] T. Weinkauf, H. Theisel, H. c. Hege, and H. p. Seidel. Boundary switch connectors for topological visualization of complex 3d vector fields. In *In Proc. VisSym 04*, pages 183–192, 2004.

[33] D. Weiskopf and B. Erlebacher. Overview of flow visualization. In *The Visualization Handbook*, pages 261–278, 2005.

[34] T. Wischgoll and G. Scheuermann. Detection and visualization of planar closed streamlines. In *IEEE Trans. Visualization and CG*, volume 7, pages 165–172, 2001.