Implicit Visualization of Vector Field Topology

Grzegorz K. Karch, Alexander Straub, Filip Sadlo, Thomas Ertl

Abstract We present implicit visualization of vector field topology by means of adaptive sampling by streamlines and determination if the streamlines sufficiently approach a critical point. We exemplify our approach using different 2D flow fields.

1 Introduction

Vector field topology is concerned with the identification and extraction of regions of qualitatively different streamline behavior within the domain of a steady vector field. The behavior is determined with respect to spatial convergence as time reaches $\pm\infty$. The structures that the streamlines converge to include critical points (isolated zeros of the vector field) and periodic orbits (isolated closed streamlines). Both critical points and periodic orbits can be classified in terms of their attraction/repulsion behavior. Those exhibiting both attracting and repelling behavior are classified as *hyperbolic* or *saddle-type*.

Traditionally, vector field topology is visualized by means of graphical representation [4] of the critical points, periodic orbits, and their manifolds (so-called *separatrices*, consisting of streamlines that converge to saddle-type critical points or saddle-type periodic orbits in forward-time or reverse direction). Although this approach is very successful, it involves several difficulties: First, it depicts the boundaries of the different regions by means of separatrices, but does not depict the regions themselves, hence complicating interpretation. Second, it may require the extraction of further topological structures such as attachment/detachment points [4], boundary switch points [8], and bifurcation lines [5], together with the respective separatrices that converge to those structures in forward or reverse time direction. Beyond that, it may require explicit treatment of more intricate structures such as strange attractors.

All this motivates us to propose implicit visualization of vector topology, i.e., instead of extracting separatrices, i.e., the boundaries of the regions, we focus here on the visualization of the regions themselves. In the present state of this work, we accomplish this by extracting all critical points of a vector field, densely seeding streamlines, and assigning each seed of a streamline the ID of the critical point that

Grzegorz K. Karch, Alexander Straub, Thomas Ertl
University of Stuttgart, Germany, e-mail: karchgz|straubar|ertl@visus.uni-stuttgart.de
Filip Sadlo
Heidelberg University, Germany, e-mail: sadlo@uni-heidelberg.de

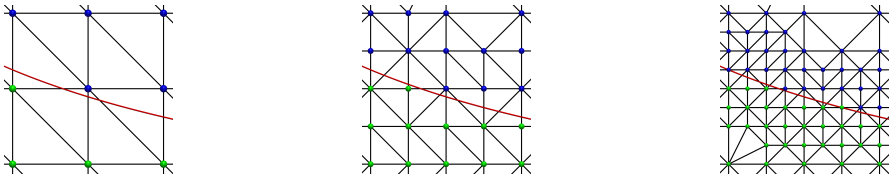


Fig. 1 Grid refinement. Red curve represents separatrix to be captured, with streamlines seeded on both of its sides reaching different critical points. (a) After initial streamline integration from the grid nodes, the ID of the respective critical point is stored at the node (blue/green). (b) If two nodes with different labels share an edge, all edges sharing those nodes are subdivided by inserting a new node at the edge midpoint. These new nodes are incorporated into the grid using Delaunay triangulation. (c) The process is repeated until minimum edge length threshold is reached.

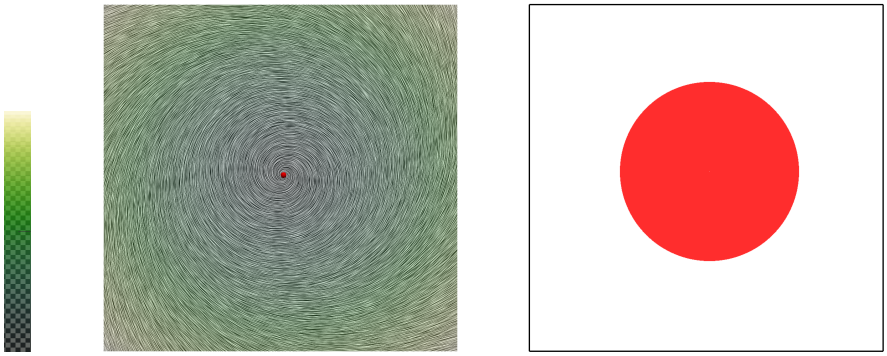
it converges to. Since it would take infinite time to test if a streamline reaches a critical point, we define a sphere with radius ε around each critical point and test if the streamline enters such a sphere. To reduce the costly streamline integration, we present a refinement scheme that increases the sampling at the resulting boundaries, i.e., at the implicitly visualized separatrices. We exemplify our approach with 2D vector fields, but it can be trivially extended to 3D, which we plan to do.

2 Method

The input to our technique is the vector field, the positions of its critical points, the diameter ε of the spheres around the critical points, a 2D uniform sampling grid (representing the region of interest for analysis) with cell size δ, and a refinement threshold σ. We choose $\delta < 2\varepsilon$ to make sure that there is at least one node of the sampling grid within each sphere. The algorithm consists of two phases.

In the initial phase, each critical point is given a unique non-negative ID, and a streamline is seeded at each node of the sampling grid and integrated over the predetermined time with the fourth-order Runge-Kutta scheme. After integration, each streamline p consists of N_p vertices such that $\mathbf{x}_{p,0}$ is the seed and $\mathbf{x}_{p,N_p-1}$ is the last vertex of the streamline. Subsequently, we search for the first vertex along the streamline that is contained in one of the spheres centered at the critical points. If we find such a vertex, we store the respective ID of the critical point at the seed node of the sampling grid. Otherwise, we set the respective node to ID -1. This results in a scalar field defined on our sampling grid.

In the second phase, the sampling grid is iteratively refined to capture the details of the topology of the vector field. First, we employ Delaunay triangulation of the sampling nodes to obtain a triangular mesh. Then, for each edge of this mesh with edge length $\leq \sigma$, the IDs of its two vertices are compared. If the IDs differ, all edges sharing these two vertices are subdivided, i.e., a sample node is added at their mid-



(a) (b)

Fig. 2 *Periodic Orbit* dataset. (a) LIC [2] representation of the vector field with attracting-focus critical point at its center (red dot), and color legend on the left. (b) Implicit visualization reveals topological region (red) where streamlines are attracted toward the critical point. The boundary of this region implicitly depicts the periodic orbit.

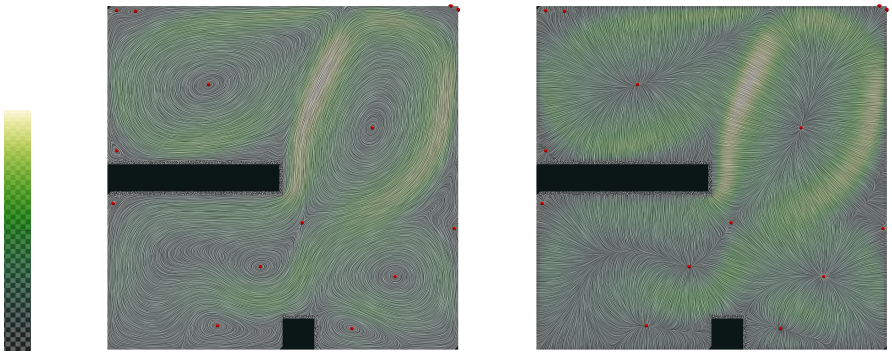
point, see Figure 1. Once all additional nodes are determined, we seed streamlines at their position and determine their ID as described above. This process is iterated, i.e., we continue with the Delaunay triangulation step above. Since the edges with different IDs are halved in each step, this procedure converges to a sampling of the vector field topology at resolution σ.

The whole algorithm is performed forward and backward in time in order to capture the regions of different behavior, e.g., with respect to convergence to sinks and sources. We implemented our approach as a ParaView plugin [1], where the vector field and critical points are given as input, and we plan to make the source code publicly available.

3 Results

The first data set we apply our technique to, is a synthetic vector field with a repelling periodic orbit around a critical point of type attracting focus (see Figure 2(a)). Our technique extracts in forward time the topological region connected to the critical point (Figure 2(b), i.e., the region within the periodic orbit. The disk boundary implicitly reveals the periodic orbit, since the streamlines started outside the periodic orbit reach the domain boundary.

The *Buoyant Flow* dataset is a single time step from a computational fluid dynamics simulation of buoyant flow in a closed container with resolution of 102×102 nodes (see Figure 3(a)). This dataset exhibits critical points of type focus (very close to the center type, i.e., very small outflow/inflow) and saddle. The *Rotated Flow* dataset was obtained by rotating the vectors of the *Buoyant Flow* dataset by



(a) (b)

Fig. 3 (a) LIC representation of the 2D *Buoyant Flow* dataset, with color indicating velocity magnitude (color legend on the left). Critical points by red dots. (b) *Rotated Flow* dataset.

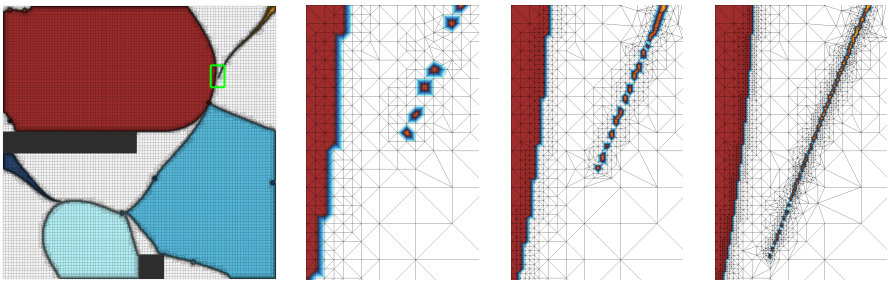
(a) (b) (c)

Fig. 4 *Rotated Flow* dataset. Implicit visualization with forward (a) and backward (b) integration. (c) Overlay of (a) and (b) reveals the topological structure of the vector field.

90 degrees counterclockwise (Figure 3(b)). This way, sources and sinks were obtained from centers with clockwise and counterclockwise rotation, respectively, while saddles have retained their type.

For both datasets, the threshold σ for the minimum edge length was set to 1.2×10^{-4}, which corresponds to $0.12d$, where d is the diagonal of a cell from the dataset. For the *Rotated Flow* dataset, the refinement resulting from different values of σ is shown in Figure 5, where already with the chosen σ (Figure 5(c)) the details are well captured. The critical point radius ε was set to $\varepsilon = 7.071 \times 10^{-4}$ for the *Rotated Flow* dataset. As will be shown below, this value was insufficient to capture the whole topological structure in the *Buoyant Flow* dataset, in which case we used $\varepsilon = 14.142 \times 10^{-4}$.

In the *Rotated Flow* dataset, Figure 4(a) and (b) shows topological regions for forward and backward streamline integration. The critical points, representing sources and sinks, can be seen as colored dots which have been captured by the refine-



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

Fig. 5 *Buoyant Flow* dataset (forward integration, as in Figure 4(a)), with region marked with green box (a) enlarged for $\sigma = 2.4 \times 10^{-4}$ (b), $\sigma = 1.2 \times 10^{-4}$ (c) and Fig. 4, and $\sigma = 0.6 \times 10^{-4}$ (d). Smaller values of σ provide improved accuracy but cannot reveal additional regions.

ment. The color of the regions indicates the respective critical point ID. The overlay of our result from both integration directions (see Figure 4(c)) reveals the topological structure of the vector field, with region boundaries representing separatrices. Interestingly, a partially developed region corresponding to the saddle point in the green box in Figure 4(b) can bee seen. The backward integration captures the diagonal region (top left to bottom right) corresponding to a sepratrix, since streamlines do not diverge strongly from the separatrix, and hence more remote streamlines reach this critical point. In case of forward integration, however, the streamlines diverge faster from the separatrix (bottom left to top right) and hence the separatrix is not captured at the given maximum resolution σ. Interestingly, in the *Buoyant Flow* dataset, the regions corresponding to the separatrices emerging from that critical point are captured as fully developed closed curves (see green box in Figure 6(c)). It is worth noting that, as can be seen in Figure 6(a) and (b), the smaller critical point radius $\varepsilon = 7.071 \times 10^{-4}$ leads to disconnected separatrices corresponding to the saddle point (green box), since, due to relatively high streamline divergence around the sepatrices, few of them reach the critical point. Only after doubling the radius to $\varepsilon = 14.142 \times 10^{-4}$, do the streamlines reach the saddle point, therefore closing the separatrix.

Regarding the computation characteristics, a substantial part of computation is spent in streamline tracing, where in the first iterations, it consumes about hundred times more time than Delaunay triangulation (i.e., 40 s versus 0.4 s). With decreasing number of inserted streamlines and increasing number of grid points, this ratio changes, and in the final iterations (after 50 iterations) Delaunay triangulation takes three times longer than streamline computation (i.e., 11 s versus 4 s).

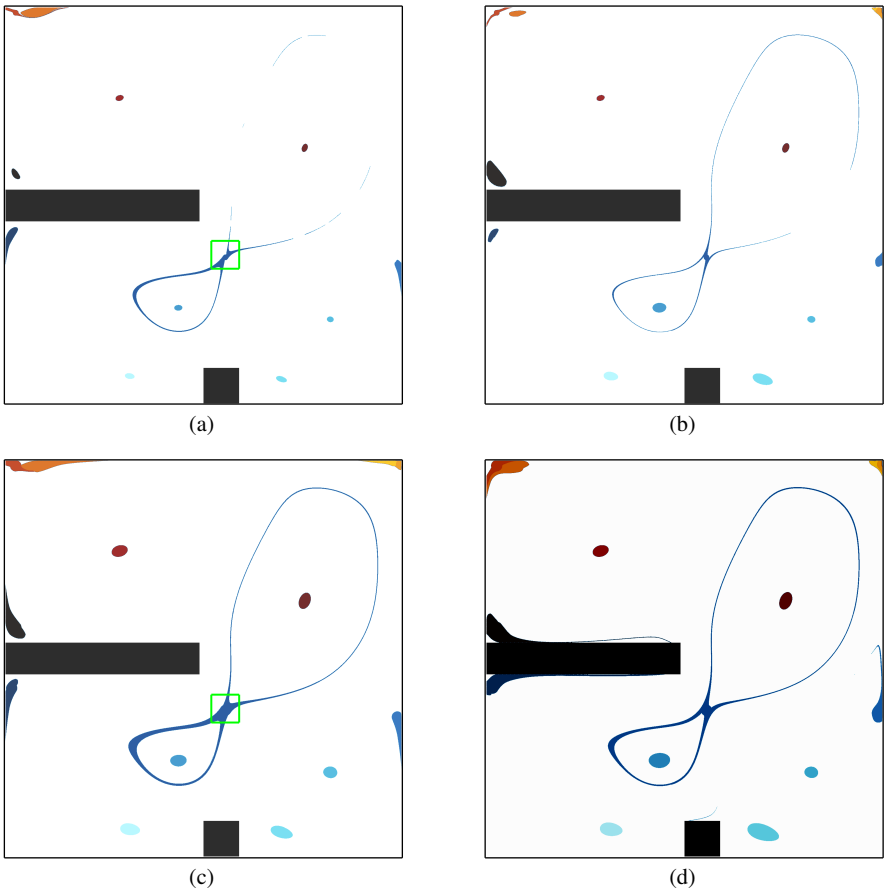


Fig. 6 *Buoyant Flow* dataset. With tolerance $\varepsilon = 7.071 \times 10^{-4}$, the blue thin lines representing separatrices are disconnected for forward (a) and backward (b) integration due to limited sampling. With tolerance $\varepsilon = 14.142 \times 10^{-4}$, the lines are closed for both integration directions ((c) and (d)), and therefore clearly show the topological region constrained by the separatices.

4 Conclusion

In this paper, we presented a technique for implicit visualization of vector field topology. It extracts regions of different flow behavior by assigning the ID of the critical point reached by a streamline started at a given point. This way, separatrices and periodic orbits enclosing repelling or attracting critical points are revealed as the boundaries of these regions. We presented a grid refinement method to improve the quality of the extracted regions, and exemplified its utility with different datasets. As future work, we would like to come up with a more sophisticated test to decide if a streamline converges to a critical point, possibly inspired by the approach by

Friederici et al. [3]. We also want to visualize the separatrices by isosurfaces, in that regard our approach represents an adaptive version of implicit integral surface extraction [7, 6]. The presented method also lends itself for extension to 3D vector fields, where graphics processing units could be used to accelerate the costly streamline computation.

Acknowledgements This work was partially funded by Deutsche Forschungsgemeinschaft (DFG) as part of the Cluster of Excellence EXC 310 "SimTech" (50131014), Transregional Collaborative Research Center SFB/Transregio 75 (84292822), and the International Research Training Group GRK 2160 "DROPIT" (270852890).

References

1. U. Ayachit. *The ParaView Guide: A Parallel Visualization Application*. Kitware, 2015.
2. B. Cabral and L. C. Leedom. Imaging vector fields using line integral convolution. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pages 263–270. ACM, 1993.
3. A. Friederici, C. Rössl, and H. Theisel. Finite time steady 2D vector field topology. In *Proc. Topo-In-Vis 2015*, 2015.
4. J. Helman and L. Hesselink. Representation and display of vector field topology in fluid flow data sets. *Computer*, 22(8):27–36, 1989.
5. G. M. Machado, F. Sadlo, and T. Ertl. Local extraction of bifurcation lines. In *Proceedings of International Workshop on Vision, Modeling and Visualization (VMV)*, pages 17–24, 2013.
6. T. Stöter, T. Weinkauf, H.-P. Seidel, and H. Theisel. Implicit integral surfaces. In *Proc. Vision, Modeling and Visualization (VMV)*, pages 127–134, 2012.
7. J. J. van Wijk. Implicit stream surfaces. In *Proc. IEEE Conference on Visualization '93*, pages 245–252, 1993.
8. T. Weinkauf, H. Theisel, H.-C. Hege, and H.-P. Seidel. Boundary switch connectors for topological visualization of complex 3D vector fields. In *Proc. Joint Eurographics - IEEE TCVG Symposium on Visualization (VisSym '04)*, pages 183–192, 2004.