Hands-On Analysis and Illustration: Interactive Exploratory Visualization of Vector Fields

Tobias Isenberg[1,2] Jens Grubert[2] Maarten H. Everts[1] Sheelagh Carpendale[2]
[1]University of Groningen, Netherlands [2]University of Calgary, Canada
{isenberg | maarten}@cs.rug.nl {jgrubert | sheelagh}@cpsc.ucalgary.ca

Abstract: We present techniques to interactively explore representations of 2D vector fields. Through a set of simple hand postures used on large, touch-sensitive displays, our approach allows individuals to custom-design glyphs (arrows, lines, etc.) that best reveal patterns of the underlying dataset. Interactive exploration of vector fields is facilitated through freedom of glyph placement, glyph density control, and animation. The custom glyphs can be applied individually to probe specific areas of the data but can also be applied in groups to explore larger regions of a vector field. Repositionable sources from which glyphs—animated according to the local vector field—continue to emerge are used to examine the vector field dynamically. The combination of these techniques results in an engaging visualization with which the user can rapidly explore and analyze varying types of 2D vector fields, using a virtually infinite number of custom-designed glyphs.

Keywords: Interactive exploratory visualization, vector field visualization & illustration, interaction on large touch-sensitive wall displays, intuitive interfaces.

1 Introduction

Vector field data arises in many scientific and technical applications. Thus, vector and flow visualization has been an important research domain for visualization. Many successful techniques have been developed to help people understand the properties of such datasets. However, traditional vector field visualization typically relies on producing static images. Line integral convolution (LIC, [1]) or the extraction of topologic properties [10], e. g., both produce one image per vector field (or possibly an animation or 3D shape) that can then be examined by the viewer. Using these techniques, exploration of a dataset is limited to setting parameters for the automatic image generation and then browsing through the final results. By providing techniques to interactively explore vector data in chosen regions using a set of custom-designed glyphs, we offer additional exploration possibilities that go beyond simple parameter changes of automated glyph placement algorithms.

Interactive exploratory visualization of vector fields allows people who need to analyze such information to probe vector data locally, to place multiple glyphs to show larger-scale properties, and to place glyph sources to explore the directional properties of the data. Our interface allows people to use hand postures to sketch custom glyphs that are best able to reveal data properties and supports interactive distribution of these glyphs. This combination of custom-designed glyphs with direct-touch interaction through a minimalistic interface enables users to both explore the data in-depth as well as to annotate traditional vector field renditions. We emphasize with our approach the necessity of physically and intuitively interacting with visualizations, rather than just tweaking parameters and observing their effects in still images. While our system works best on touch-sensitive wall displays, it can also be used on desktop and mouse setups.

The paper is organized as follows. In Sec. 2 we review previous work related to our approach in the area of vector visualization. Next, Sec. 3 introduces our concept of interactive exploratory vector field visualization before Sec. 4 shows possible application scenarios. Sec. 5 discusses technical aspects of the realization and points out some limitations. We conclude the paper in Sec. 6 and suggest directions for future work.

2 Related Work

Previous work in vector field or flow visualization can be classified in one of four categories [11]: direct visualization, dense texture-based visualization, geometric visualization, and feature-based visualization. We discuss related work according to each of these categories.

Direct visualization techniques use a direct mapping from data to visualization to produce an overall picture of the flow. A common approach is to position glyphs at grid points to convey properties such as direction or velocity. A simple example is an arrow plot, but more complex glyphs have been used to convey, e. g., uncertainty [21]. Alternatively, simulated painted brush strokes can be used as glyphs to obtain non-photorealistic visualizations of multi-dimensional data elements [4, 16]. Kirby et al. [7], inspired by layering in painting, use layers of glyphs and backgrounds to visualize different aspects of a vector dataset.

Texture-based techniques provide a dense view of the vector field by computing a texture that conveys both local and global properties. Similar to the direct techniques, these texture-based techniques also provide an overall picture of the flow. Line Integral Convo-

lution (LIC)[1] is an early but commonly used texture-based method. While many variations of the LIC technique exist, both in 2D and 3D, the basic idea is that a noise texture is smeared out in the direction of the vector field. More recent texture-based approaches are image-based flow visualization [18] and the level-set based dye advection approach presented in [19]. Two-dimensional texture-based visualizations that have the same spatial domain as the original data can be used in our system as background visualizations.

The last two groups are geometric and feature-based methods. Geometric techniques integrate particle paths to form geometric objects such as streamlines, streaklines, or pathlines. These can show the long-term behavior of a field. These are related to our approach as the glyphs we are using sometimes serve a similar purpose. Feature-based methods extract features from the vector field and visualize them, resulting in a potentially less cluttered visualizations. Examples of such features include convergence, divergence, flow rotation, shear, and flow vorticity. A detailed account on feature-based flow visualization can be found in [13] or, more recently, for topology-based techniques in [10].

As studies have shown that visualizations that are custom-designed (e. g., by artists) with adequate tools can yield better results than standard methods [6], and that different visualization techniques for vector fields are suited for certain tasks to varying degrees [9], we believe that interactive exploration of vector data is a promising path to investigate. Most existing literature on this subject focuses on the visualization of 3D vector fields and on generating those visualizations at interactive frame-rates, only few approaches explore the interaction with the data. In [2] a method is presented that allows users to interactively probe and annotate 3D flow fields using complex glyphs that show field properties such as velocity, shear, and rotation. In addition, the approaches presented in [15, 17, 8] allow people to customize glyphs to use for data visualization by associating data values with glyph parameters. However, these customizations are restricted to simple geometric 3D glyph shapes. An interactive approach using dye advection is presented in [20]. This work along with earlier work [12, 19] relate to our system in that they discuss the placement of sources and interactively compute which part of the flow is affected by them.

3 Interacting with Vector Fields

Vector field visualizations tend to focus on creating global representations of the data. If the data sets are either large or complex, the resulting visualizations can have overwhelming visual complexity. As a result it can be hard to pick out pertinent details and difficult to use these visualizations to communicate convincingly. We begin to address this visual complexity by providing personalization tools that enable:

- **the creation and personalization of the actual glyphs to be used:** Glyphs can be designed to be more understandable to the targeted group of people, and custom glyphs may better reveal particular patterns in the data.
- **the placement, and/or animation of glyphs in specifically chosen locations:** This enables us control the visual complexity by annotating visualizations in specific location only, extending these local annotations at will to increase the visual complexity at a rate chosen to maintain comprehensibility, and thus allowing our interaction methods to be used to create even large visualizations.
- **emphasis and annotation:** The capability for local vector visualization combined with the use of both personalized glyphs and color provides for a great variety of possibilities for creating emphasis and adding annotation.

In keeping with our goals to provide simple, interactive exploration tools for vector data, we look to develop a simple interface by avoiding, when possible, a complex parameterization of the program control by, instead, utilizing natural hand postures. In this section we describe how these goals are realized. Building from the smallest component, the glyph, we start by describing how hand postures can be used to create a huge variety of personalized glyphs. Next, we outline the underlying framework that provides the animation and simplifies the localization of visual effects. Then Sec. 3.3 explains how, with the same postures, one can create and adjust the vector visualizations.

3.1 Hand-Designed Glyphs

Glyphs are frequently used to illustrate properties of vector fields as they can show local data parameters by adapting their orientation, size, and other properties. Also glyphs, by their shape, can communicate properties of the data. For instance, arrows are often used to indicate the directionality of a field. Thus, glyphs are a natural choice as elements to work with to enable users to interactively explore vector data.

Traditionally, the design of glyphs used in vector field visualizations was finalized before run-time or even before compile-time. This can lead to sub-optimal visualizations as glyphs often depend on the type of data that is being used. Thus, we allow individuals to personally sketch and design the glyphs that are used in visualizing the data (arrows, lines, etc.; see Fig. 1). This provides the freedom to create glyphs that, for the intended usage, best reveal patterns in the current vector data. In addition, sketching custom glyphs can allow individuals to more rapidly test out and realize ideas with respect to given visualization tasks.

For the personal creation of glyphs we have to support a number of tasks: people need to be able to indicate when they are creating glyphs and not interacting with the vector data, to draw a variety of lines to de-

Fig. 1: Examples for hand-sketched directional glyphs.

sign glyphs, and to erase these lines or parts of them. At the same time, our goal was to reduce visual clutter and to provide a minimalistic but easily remembered interface. This lead us to explore the use of hand postures for controlling the interface instead of common on-screen menus. People are easily able to remember a small number of hand postures and we can, thus, trade in visible interface elements for a closer and more immediate 'invisible' interaction with the visualization.

The design of hand postures for interaction was guided by our goal to provide robust and intuitive interaction that is easily learned and by a number of technical constraints. Our technical setup comprises a SMART Technology DViT that provides touch sensitivity for large projected, plasma, or LC displays. This technology uses four cameras in the display's corners which see the shadow of an interacting object (e. g., hand or pen) in front of a strip of infrared LEDs along the side of the display. This yields the center position as well as the approximated width and height of the object which we use in turn for recognizing hand positions and postures.

Inaccuracies during the recognition process of interaction positions and their parameters limit the number and type of possible hand postures. For instance, the physical area covered by one hand posture has to be disjoint from areas covered by other postures. In addition, we accounted for individual differences between people's hand and finger sizes for our design. Finally, we also considered what postures people are able to form easily with their hands and which ones are used for natural interactions with other people. Examples for such natural postures include the use of one finger for pointing or forming a fist for showing emphasis.

Based on these constraints, we developed four natural hand postures that can be discriminated by our hardware setup for a variety of hand sizes and that can be formed easily with a single hand: one-finger pointing, two-finger pointing, a fist, and a flat hand (see Fig. 2). While this small number of distinguishable hand postures is advantageous in that they are easy to remember, they do limit the number of functionalities that we can map while still avoiding menus or keyboard interactions. The postures also do not lend themselves intuitively to encode the distinction between sketching new glyphs and using these glyphs to explore the vector field as we envisioned the posture-to-functionality mapping to make sense to people in the context of directly-manipulative sketching and whiteboard interaction, avoiding a lengthy learning process.

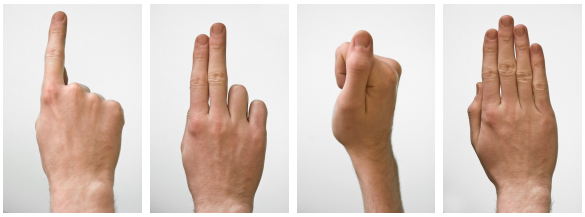
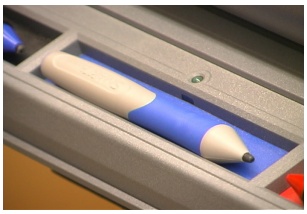
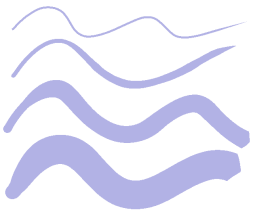


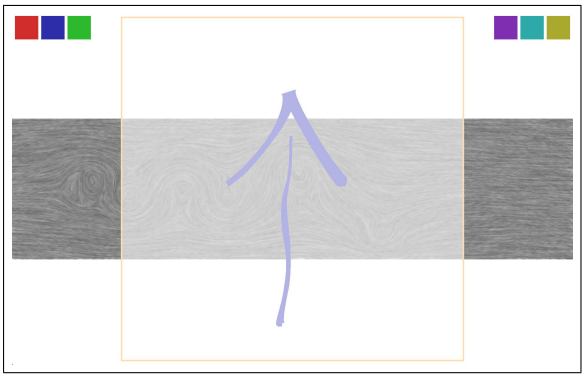
Fig. 2: The four hand postures that can be recognized.



(a) Pen on the pen-tray. Lifting the pen indicates the switch into the glyph sketching phase.



(b) Stroke widths that can be created with the pen and the four hand postures.



(c) The drawing area displayed during the glyph sketching phase.

Fig. 3: Sketching glyphs with pen or hand postures.

Therefore, we made use of a pen tray that is attached to SMART Technology's DViT boards (see Fig. 3(a)) which can detect when a pen is lifted from its associated pocket and when it is placed back. While technically it would be possible to distinguish four different pens and, thus, map four more sets of functionalities to the four hand postures, we felt that this might make it hard to remember the mappings. Thus, we used only one additional set of mappings, indicated by any pen being lifted from its pocket.

This action was consequently mapped to naturally indicate the shift from interacting with the flow visualization (pen down) to sketching a new glyph (pen lifted) because the use of a pen can easily be associated with a sketching action. In addition, the fact that one pen is lifted during the sketching actions also enabled us to add an additional "hand posture," i. e. the pen, and use it for even finer control (tip diameter ca. 3 mm) than a single pointing finger. Thus, to draw a new glyph, people can use the picked-up pen as well as hand postures to control the stroke width: using the pen provides the thinnest lines while using one finger, two fingers, and a fist creates increasingly thicker lines (see

Fig. 3(b)). The flat hand is used for erasing which is inspired by the common erasing action on blackboards.

The described interactions allow people to sketch a wide variety of glyphs for use in vector field exploration and illustration (see example of sketching in Fig. 3). As the vector field's direction is used to orient the glyphs, this consequently means that each glyph has an inherent directionality. We address this by employing the notion that the direction of a sketched glyph is straight up. In addition, the sampling point of the glyph is in the middle of the sketch area.

3.2 Interactive Glyph-Based Vector Fields

Using and interacting with a potentially high number of glyphs on a large display also poses a number of challenges. Interaction should offer a way to affect certain glyph properties. A system needs to be able to accommodate and render a high number of glyphs without losing interactive speed. For this purpose we make use of interface buffers (i-buffers, [5]). These i-buffers store properties (e. g., size, orientation, or color) in a spatial manner so that primitives can access them locally to update their appearance accordingly. By interactively changing the buffer contents it is now possible to interact with the displayed primitives.

To visualize vector field data, we represent the vector directions in one i-buffer and the length of the vectors (i. e., the local strength of the field) in another one. These properties can now be used to control the appearance of glyphs whose location is projected to i-buffer positions. The vector field's direction is mapped to the orientation of the glyph while the field's strength is represented by the glyph size. As glyphs move across the buffers, they constantly check for new values with respect to their position and update their rendering accordingly. Alternatively, tools can modify the values of the i-buffers, to update the representation of glyphs reading from them. We make use of this latter technique, for instance, to control glyph transparency.

3.3 Interactive Vector Field Exploration

Once custom glyphs have been designed, we use these to support the interactive exploration of vector data. For this purpose we offer a second set of hand posture mappings for the phase when the pen is placed in its tray (one finger, two fingers, fist, and flat hand). These mappings need to support exploration tasks but also have to follow naturally from the postures to provide an intuitive interface. We selected four exploration actions for our visualizations: probing the vector field locally with a single glyph, placing a number of glyphs simultaneously at a larger scale, de-emphasizing and removing glyphs, and placing and re-locating sources that continuously emit glyphs which then start moving across the data, following the vector field's direction.

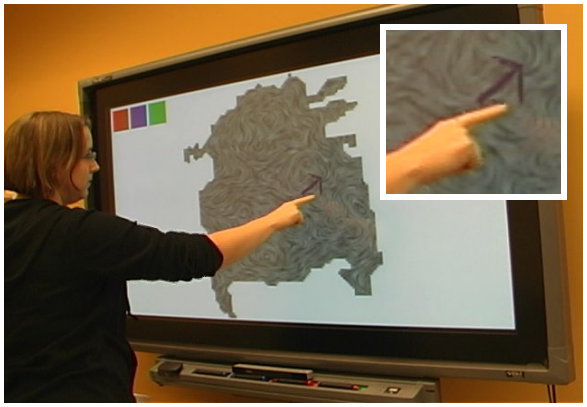
Probing the vector data is enabled with the one-finger posture. When just one finger touches the display, the system automatically creates a new instance of the most recently sketched glyph. While the finger is still touching, moving the finger also moves the glyph, whose rendition adapts to vector field's data values below its current position (Fig. 4(a)). As fingers are relatively blunt tools (compared to, e. g., a mouse pointer's tip), the glyph is off-set from the actual finger position to the top left (top right for left-handed people) to guarantee visibility [14]. As soon as the finger is released from the screen, the glyph remains at its last location, visualizing the local conditions at this point.

While the above interaction allows users to explore the dataset locally, the fist posture is employed to explore the vector field at a larger scale. By placing the fist on and moving it across the screen, the system continuously emits new instances of the most recently sketched glyph at random locations [3] within a radius around the interaction point (Fig. 4(b)). Using this technique, regions or the entire dataset can be covered with glyphs within a short time to explore the behavior of the field on a larger scale. We found relatively small and simple glyphs to be particularly useful for this type of interaction as they do not cause too much visual clutter. The resulting images resemble those created by techniques which place short streamlines on vector data as well as traditional glyph-based visualizations.

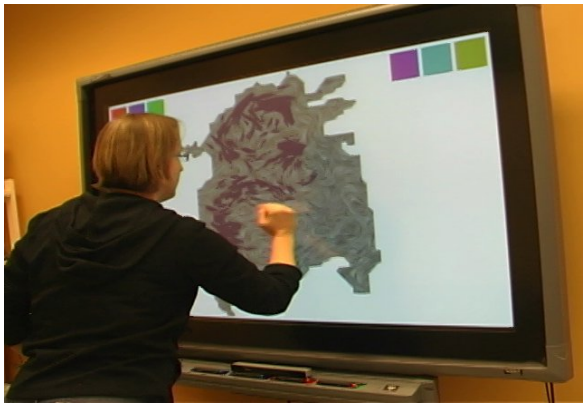
The flat-hand posture provides de-emphasis or can completely erase glyphs (Fig. 4(c)), similar to its mapping in the sketching phase. De-emphasis of glyphs is realized by first increasing their transparency when the hand is first placed on the display and only deleting glyphs once their transparency has reached a given threshold. This de-emphasis can be used to provide an indication of the field's general direction and size properties. The probing interaction can be used on top of such a visualization to examine specific locations.

While static glyphs can be useful to explore a flow dataset, the display of animated glyphs following the streamlines can further aid the detection of patterns in the data. We offer exploration with animated glyphs through the use of the two-finger posture to create glyph sources that can be placed on and moved over the dataset. Each source continuously emits instances of the most recently sketched glyph. The instances are again generated at random positions within the radius of the glyph sources to allow users to explore local variations of the data. These glyph instances, in contrast to the ones previously created with other postures, not only derive their size and orientation from the vector data but also move according to the streamline direction (Fig. 5). This movement is realized by simple stepping along the local vector direction. Although it is not a physically correct integration of the vector field, it is sufficient to produce animations that help to illustrate the flow character and properties of a vector field.

Visual clutter is a problem in flow visualizations



(a) Probing the dataset with a single glyph and the one-finger posture. The glyph is off-set to ensure its visibility.



(b) Exploring larger areas of the dataset using a fist.

(c) Glyph de-emphasis & erasing using the flat hand.

Fig. 4: Interaction with one finger, fist, and flat hand.

when too many glyphs, placed in close vicinity, overlap each other, as this makes it difficult to extract glyph properties from their representations. Visual clutter can become a problem with our interactive techniques when too many sources or individual glyphs have been placed. In order to disambiguate glyphs we provide the possibility to assign a color to individual glyphs and sources which then consequently color the glyphs that are generated by them. For this purpose the interface has "color pots" at the top of the screen into which sources or the probing finger can be "dipped," assigning the respective color to them (Fig. 6).

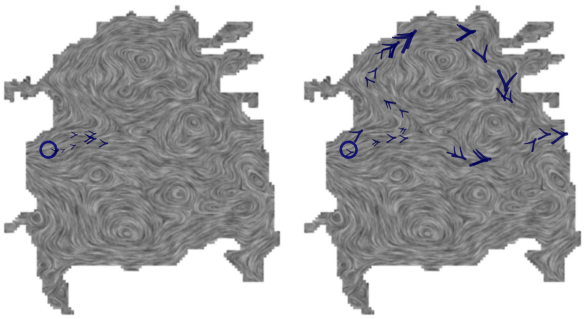


Fig. 5: Using glyph sources to illustrate flow properties. Sources are placed with the two-finger posture.

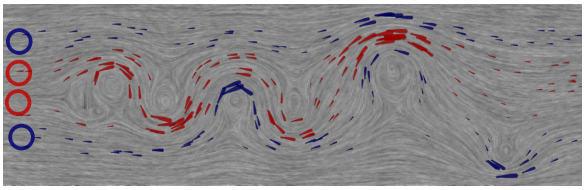


Fig. 6: Using colored glyph sources to identify trends.

3.4 Interactive Visualization and Illustration

In the previous section we discussed how our techniques benefit the interactive exploration of vector field data. The same interaction techniques can also be used to interactively create static illustrations of a dataset. Illustrations can be used to emphasize interesting aspects in the vector field or to prove or disprove assumptions about a dataset. For these kinds of tasks we rely on representing some aspects of the vector data using glyphs and combining this with traditional vector field visualization techniques. Examples for such traditional visualizations include LIC images, height images, vector field topology, vorticity images, or even a visual encoding of the vector data. Those visualizations are created in a pre-processing step and are rendered as a background. By using the same spatial domain as the original vector field and aligning the extra visualization layers with the dataset, people can match features in the traditional visualization with the properties that they see from the glyphs they place.

The same posture mappings are employed for the illustration of datasets as were used to enable exploration. Glyphs can be interactively placed using one finger or a fist as well as de-emphasized and removed using the flat hand. The probing interaction can now be used to place specially designed glyphs purposefully at locations that are worth pointing out. Glyph sources are typically not as useful in this case as they provide a different image at each frame, but may be used if desired. Generally, we found that more complex and elaborate glyphs are useful for illustrations since, when compared with exploration, fewer glyphs are typically placed to emphasize or annotate. This way users can create static images, using the interaction to show aspects of vector data that are otherwise not visible in

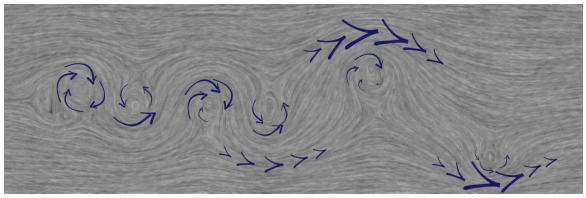


Fig. 7: Annotation of a LIC image of a vortex simulation. Three hand-sketched glyphs point out aspects that are otherwise not visible such as the field's strength.

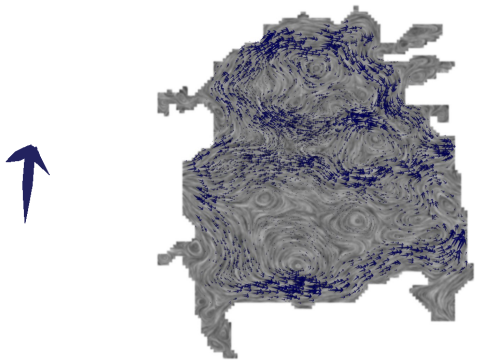


Fig. 8: Exploring trends using the fist posture.

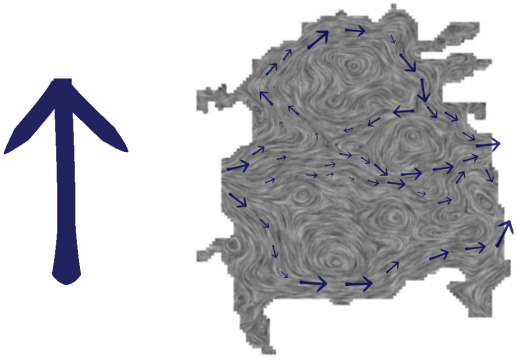


Fig. 9: Illustrating the identified trends using the one-finger posture and a larger glyph.

a visualization. One example for such an illustration is highlighting the strength of the flow in a particular region in a LIC visualization (Fig. 7).

4 Case Studies

To further illustrate the use of our methods we discuss them in the context of case studies of two simulated datasets: a water flow in a part of the Baltic sea and a fluid flow around a half-cylinder in 2D. Both datasets are time-dependent and we focus on visualizing individual slices. Users can cycle through the time slices, causing placed glyphs to adapt accordingly while additional glyphs can be added at any time. For both datasets we use additional visualizations (e. g., LIC and vorticity images) and annotate them.

Water Flow Simulation. To explore the larger scale water flow simulation dataset, one may start by sketching a small arrow glyph and then "flooding" the entire area with it using the fist posture (Fig. 8). Once a large region of the field has been covered with glyphs, the major streams in the flow and their strengths are revealed. After streams have been identified, one can delete all placed glyphs, draw a larger arrow as the next glyph, and use it to probe the field more precisely. By continuously leaving glyphs at interesting spots one can end up with an illustration of the major flows on top of the LIC background image (Fig. 9). Finally, to confirm the flow characteristics with animated glyphs, one may draw a small line glyph and place sources onto the dataset. By placing the sources at the inflow of the water body one can reveal how the glyphs proceed along the major streams (Fig. 10, left). This flow pattern may change significantly when a different time step of the simulation is chosen, revealing temporal relationships between the time slices (Fig. 10, right).

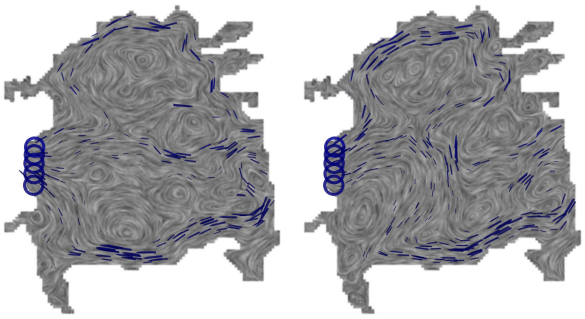


Fig. 10: Sources showing the flow's time-dependency.

Turbulence Simulation. To explore the small-scale water flow simulation around a half-cylinder, one can start by drawing an arrow without the center bar and probing the dataset with it, using the LIC image in the background as a guide and leaving instances of the glyph as annotations (Fig. 11, top). Next, using the same glyph, the fist posture is used to discover larger trends, in particular the the flow's strength (Fig. 11, bottom). Now, to illustrate the different rotation directions of the vortices in the dataset, one can choose to draw an arrow that is bent clockwise, to show the vortices that rotate that way, and to repeat the same action with a mirrored arrow for vortices that rotate counter-clockwise (Fig. 12, top). To further illustrate the differences, the second set of arrows can be colored in red to distinguish them from the first set (blue). Finally, to confirm the findings of the previous step, the LIC image in the background is replaced by a visualization of the vorticity of the flow (Fig. 12, bottom).

5 Realization Aspects and Limitations

The interactive vector field exploration system is implemented using OpenGL and relies on representing the vector data in i-buffers [5]. Each glyph is represented by an OpenGL quad that is texture-mapped with the previously sketched glyph. During the glyph sketching

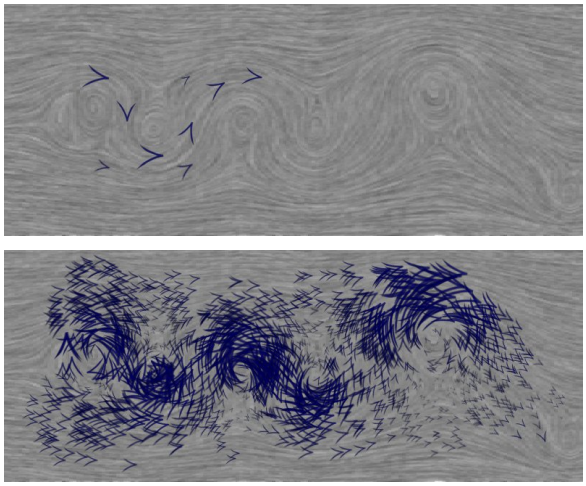


Fig. 11: Probing the dataset with few glyphs vs. revealing the flow's strength and direction using many.

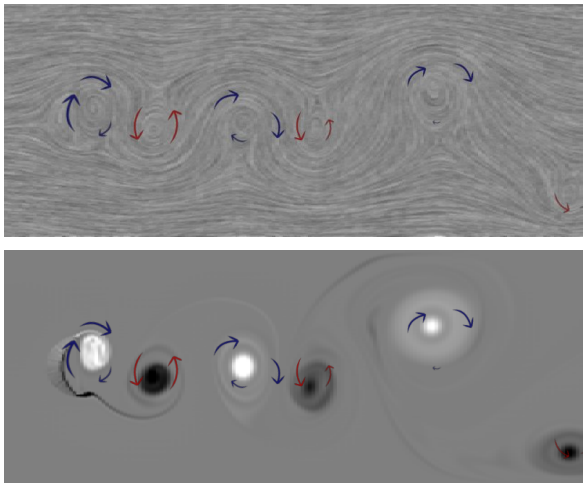


Fig. 12: Illustrating the rotation direction using bent arrows, using a vorticity visualization for confirmation.

phase, the strokes that form the glyphs are represented analytically in order to accommodate the easy erasing of parts of them. Once the glyph is completed, the strokes are rendered in an off-screen buffer and copied into an OpenGL texture.

Each glyph instance, at each animation step, looks up the i-buffer values for its position, i. e., its center. If this position does not happen to fall exactly on one of the buffer values, the value is linearly interpolated. Transparency is maintained by each glyph individually and is also modified through an i-buffer. For this purpose, the tool that is invoked by the flat hand posture writes transparency offsets into this separate i-buffer, while glyphs use this value to update their specific transparency values. Glyphs are deleted once this value drops below a given threshold. Glyphs are also deleted when their size drops below a small threshold. This causes glyphs that move out of the dataset to be deleted automatically as the values in the background part of the size i-buffer are initialized to zero. This ensures that no glyphs are maintained or rendered longer than necessary, keeping the interaction responsive and the animation running smoothly.

Our entire interactive system is realized on the CPU with i-buffers and data being maintained in main memory. While this provides much flexibility for the implementation of the animation, it also has its limits with respect to how much data can be accommodated. For example, not the entire datasets shown in Sec. 4 (which contain a stack of time steps) can be converted to screen-sized i-buffer data directly, as this would require more memory than is available on typical PCs. Instead, whenever we switch between time steps, the current i-buffer set as well as the texture for the current background visualization are created from the data. However, the smaller glyphs are all maintained as textures on the graphics card and we did not notice any problems even when using a large number of different glyphs simultaneously.

While the presented approach works nicely for many datasets, it may not be as powerful with very noisy vector data that does not have strong "streams." We experimented, for example, with vector fields extracted from diffusion tensor imaging (DTI) datasets which arise, for instance, in biology. In such cases where there are no clear trends it is also difficult to place meaningful glyphs with our technique to illustrate aspects of the dataset.

6 Conclusion and Future Work

In summary, we have presented a system for *interactive exploratory visualization of vector fields* on large, touch-sensitive wall displays. By allowing users to sketch their own glyphs we can let them create ones that work best for the specific data. These custom glyphs can then be used to interactively explore vector fields as well annotate traditional visualizations of a dataset. The interface is controlled through hand postures which allow us to largely abstain from using keyboard or menus. Certainly, different mappings as well as interactions are possible to control such a tool. Also, some parameters such as specific stroke widths, animation speeds, or thresholds to detect hand postures are hidden and/or empirically determined. We think, however, that limiting the number of exposed parameters in a minimalistic approach provides an intuitive and easy-to-understand interface for flexible, human-guided visualization. Our technique complements existing visualizations by facilitating intuitive explorations of data and later the illustration of specific aspects.

For future work we are considering a number of paths. We would like to investigate how to show a third dimension of a vector field using either glyph shape or color. We also want to include a more precise integration method for moving glyphs. Special datasets may also require a way to deal with sinks inside the vector

field. In this case we would need to track the position of glyphs to eventually delete them. In addition, other types of interaction could be investigated such as the zooming and/or panning of large datasets or the interactive placement of chains of glyphs. Finally, we are pursuing the use of different types of vector datasets from various sources as well as a formal study of the interaction techniques with domain scientists.

Acknowledgments

We thank Kurt Frischmuth (Univ. of Rostock, Germany) for the Greifswalder Bodden dataset and Michel Westenberg (Univ. of Groningen, Netherlands) for the turbulence dataset. In addition, we thank Petra Isenberg for her help with creating the video as well as Henk Bekker and Samuli Ollila for discussions on the technique. We also would like to acknowledge our funding providers Alberta Igenuity, NWO, NSERC, CFI, iCORE, and SMART Technologies.

References

[1] B. Cabral and L. C. Leedom. Imaging Vector Fields using Line Integral Convolution. In *Proc. SIGGRAPH*, pages 263–270, New York, 1993. ACM Press.

[2] W. de Leeuw and J. van Wijk. A Probe for Local Flow Field Visualization. In *Proc. IEEE VIS*, pages 39–45, Los Alamitos, 1993. IEEE Computer Society.

[3] D. Dovey. Vector Plots for Irregular Grids. In *Proc. IEEE VIS*, pages 248–253, Los Alamitos, 1995. IEEE Computer Society.

[4] C. G. Healey, L. Tateosian, J. T. Enns, and M. Remple. Perceptually-Based Brush Strokes for Nonphotorealistic Visualization. *ACM Transactions on Graphics*, 23(1):64–96, Jan. 2004.

[5] T. Isenberg, A. Miede, and S. Carpendale. A Buffer Framework for Supporting Responsive Interaction in Information Visualization Interfaces. In *Proc. C^5*, pages 262–269, Los Alamitos, 2006. IEEE Computer Society.

[6] D. F. Keefe, D. B. Karelitz, E. L. Vote, and D. H. Laidlaw. Artistic Collaboration in Designing VR Visualizations. *Computer Graphics and Applications*, 25(2):18–23, Mar./Apr. 2005.

[7] R. M. Kirby, H. Marmanis, and D. H. Laidlaw. Visualizing Multivalued Data from 2D Incompressible Flows Using Concepts from Painting. In *Proc. IEEE VIS*, pages 333–340, Los Alamitos, 1999. IEEE Computer Society.

[8] M. Kraus and T. Ertl. Interactive Data Exploration with Customized Glyphs. In *Proc. WSCG*, pages 20–23, 2001.

[9] D. H. Laidlaw, R. M. Kirby, C. D. Jackson, J. S. Davidson, T. S. Miller, M. da Silva, W. H. Warren, and M. J. Tarr. Comparing 2D Vector Field Visualization Methods: A User Study. *IEEE Transactions on Visualization and Computer Graphics*, 11(1):59–70, Jan./Feb. 2005.

[10] R. Laramee, H. Hauser, L. Zhao, and F. Post. Topology-Based Flow Visualization, The State of the Art. In H. Hauser, H. Hagen, and H. Theisel, editors, *Topology-based Methods in Visualization*, pages 1–19. Springer-Verlag, Berlin, 2007.

[11] R. S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F. H. Post, and D. Weiskopf. The State of the Art in Flow Visualization: Dense and Texture-Based Techniques. *Computer Graphics Forum*, 23(2):203–221, June 2004.

[12] K.-L. Ma and P. J. Smith. Virtual Smoke: An Interactive 3D Flow Visualization Technique. In *Proc. VIS*, pages 46–53, Los Alamitos, 1992. IEEE Computer Society.

[13] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch. Feature Extraction and Visualization of Flow Fields. In *Eurographics 2002 State of the Art Reports*, pages 69–100. Eurographics Assoc., Aire-la-Ville, Switzerland, 2002.

[14] R. L. Potter, L. J. Weldon, and B. Shneiderman. Improving the Accuracy of Touch Screens: An Experimental Evaluation of Three Strategies. In *Proc. CHI*, pages 27–32, New York, 1988. ACM Press.

[15] W. Ribarsky, E. Ayers, J. Eble, and S. Mukherjea. Glyphmaker: Creating Customized Visualizations of Complex Data. *Computer*, 27(7):57–64, July 1994.

[16] L. G. Tateosian, C. G. Healey, and J. T. Enns. Engaging Viewers Through Nonphotorealistic Visualizations. In *Proc. NPAR*, pages 93–102, New York, 2007. ACM Press.

[17] R. van Teylingen, W. Ribarsky, and C. van der Mast. Virtual Data Visualizer. *IEEE Transactions on Visualization and Computer Graphics*, 3(1):65–74, Jan.–Mar. 1997.

[18] J. van Wijk. Image Based Flow Visualization. *ACM Transactions on Graphics*, 21(3):745–754, July 2002.

[19] D. Weiskopf. Dye Advection without the Blur: A Level-Set Approach for Texture-Based Visualization of Unsteady Flow. *Computer Graphics Forum*, 23(3):479–488, Sept. 2004.

[20] D. Weiskopf, R. P. Botchen, and T. Ertl. Interactive Visualization of Divergence in Unsteady Flow by Level-Set Dye Advection. In *Proc. SimVis*, pages 221–232, Erlangen, 2005. SCS European Publishing House.

[21] C. M. Wittenbrink, A. T. Pang, and S. K. Lodha. Glyphs for Visualizing Uncertainty in Vector Fields. *IEEE Transactions on Visualization and Computer Graphics*, 2(3):266–279, Sept. 1996.