EUROGRAPHICS Symposium on Sketch-Based Interfaces and Modeling (2010)
M. Alexa and E. Do (Editors)

Drawing with the Flow: A Sketch-Based Interface for Illustrative Visualization of 2D Vector Fields

D. Schroeder,[1] D, Coffey,[1] and D. Keefe[1]

[1]IV Lab University of Minnesota

Abstract

This paper presents Drawing with the Flow, a sketch-based interface for illustrating 2D vector fields. Drawing with the Flow explores the problem of making scientific visualization tools accessible to artists and illustrators, who have a finely tuned visual design sense, but do not typically have the programming or mathematical background required to work with modern visualization algorithms and tools. Using a sketch-based interface that is accessible to illustrations, Drawing with the Flow makes it possible for illustrators to explore new visual designs for streamline placement in order to create illustrations of 2D vector fields, such as simulated fluid flows. The interface includes a method for interpreting hand-drawn marks relative to an underlying vector field, utilizing an "ink-data settling" procedure to subtly maintain an image consistent with the underlying data. A variable-density interactive streamline seeding algorithm tied to sketch-based input is also introduced, and design lessons learned and limitations are discussed. Several example illustration results have been created during sessions ranging from five to twenty-five minutes. The results presented demonstrate different styles of illustration for describing simulated data for 2D flows past a cylinder and 2D flows that include several different types of critical points.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction Techniques I.3.m [Computer Graphics]: Miscellaneous—Visual Arts

1. Introduction

Scientific visualization has a rich, lengthy history of drawing upon artistic and illustrative techniques to represent complex data. The work of Leonardo DaVinci is perhaps the best classic example of combining art/illustration and science. His pen and ink studies of water [LS05] are timeless examples that demonstrate the power that an illustrator's eye (and pen) can bring to the problem of representing complex scientific phenomena in visual form.

Today, the modern field of scientific visualization continues to draw upon art and illustration for inspiration. Non-photorealistic rendering is employed to create volume rendering techniques that mimic pen and ink illustration [RE01], multi-variate data are visualized using techniques inspired by oil painting [KKL05, HE02], and many other compelling visualization strategies draw inspiration from art and illustration (e.g. [LM02, HMCM]). One limitation of the current generation of illustrative visualization techniques is that they do not, aside from a few noted exceptions [KAM*08, IEGC08], utilize interfaces that make it possible for illustrators (the artists themselves) to contribute to the process of visualization. Instead, most approaches are fully automated with just a few parameter settings available to the user, or they require programming knowledge in order to adjust the design of the visualization.

This paper introduces a new sketch-based interface we call "Drawing with the Flow" as a first step toward addressing the problem of making computer-based scientific illustration tools accessible to illustrators. Our approach targets a specific visual design problem in the area of 2D vector field visualization: effective placement of streamlines to produce compelling illustrative visualizations of 2D flow.

Although there has been much work within the visualization community on the topic of automatic streamline place-

DOI: 10.2312/SBM/SBM10/049-056

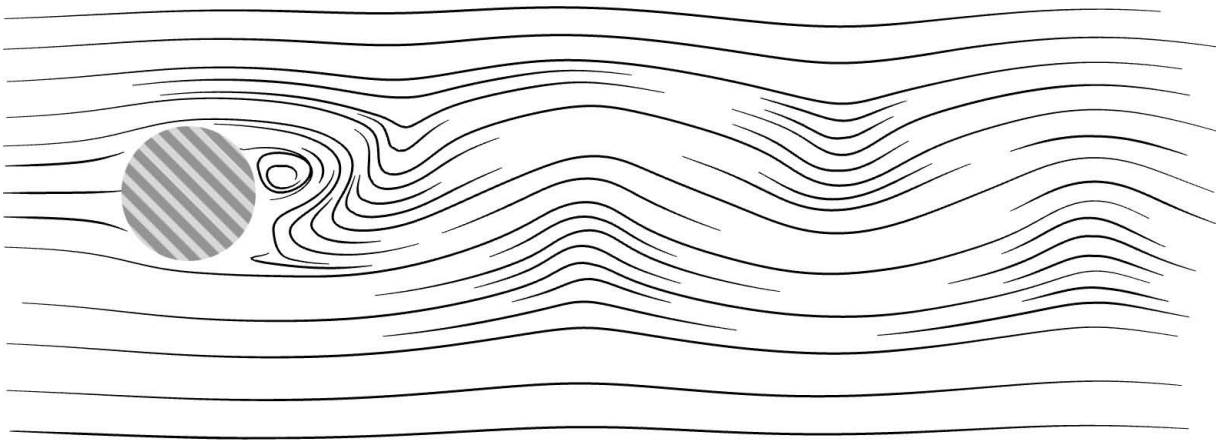

Figure 1: *An illustration of simulated 2D fluid flow past a cylinder created using Drawing with the Flow.*

ment (see [MLP*09] for a review), it is also widely recognized (even in the same literature) that often the most effective illustrations are hand-drawn by a person [TB96]. At the most basic level, the main difference between auto-generated images and human-drawn images lies in the placement and density of the streamlines. Most automatic algorithms attempt to detect an optimal global density and then fill in the image with streamlines positioned so as to achieve this density evenly across the image. In contrast, many of the most successful hand-drawn illustrations include streamlines at variable densities and/or placed in very specific locations, so as to most effectively capture the overall sense of the flow as well as specific features of interest. Despite their often preferred visual qualities, hand-drawn illustrations also suffer from several limitations. First, they are difficult to create. The illustrator must first understand the flow to depict, then recreate the flow by drawing either on paper or using a digital tool, such as a spline drawing package. Second, the illustrations that result are not tied to any underlying data. Some of the best examples of flow illustrations that we have are drawings of canonical flows that have been well studied. A pencil and paper illustration can be used to tell the story of this type of data, but the picture is not actually linked to any underlying data describing the flow. Thus, hand-drawn illustrations are compelling, but they are at best only representative of the underlying flow structures, not directly tied to those structures, making it difficult for hand-drawn visualizations to scale beyond anything other than very simple flow cases.

Drawing with the Flow attempts to address these limitations. A sketch-based interface is utilized to maintain the feel and workflow of traditional illustration, but rather than illustrating away from the data, in Drawing with the Flow, illustrators draw directly "on top of" the data. An "ink-data settling" feature subtly enforces the constraint that hand-drawn streamlines always agree with the structure of the underlying flow field, thus, the illustrations generated are always true to the data. (See Figure 1.)

The contributions of this work include: 1. The demonstration of a sketch-based system for making illustrative visualization of 2D vector fields accessible to illustrators. 2. A method for interpreting hand-drawn marks relative to an underlying vector field, including an "ink-data settling" process that subtly maintains an image consistent with the underlying data. 3. A variable-density interactive streamline seeding algorithm tied to sketch-based input.

The remainder of the paper begins with a discussion of relevant related work in illustrative visualization, streamline placement, and sketch-based interfaces. Then, the key advances in the Drawing with the Flow system are described. This is followed by an analysis of several illustration results created using the system.

2. Related Work

Drawing with the Flow connects to and builds upon three main threads of related research described in the following sections: illustrative visualization techniques, streamline seeding algorithms, and applications of sketch-based interfaces to visual design and visualization.

2.1. Illustrative Visualization

A consistent recent theme within the visualization literature has been utilizing art-based or illustrative techniques to describe complex data. Some techniques mimic the appearance of brushstrokes and encode data in the color, width, and other characteristics of the stroke [KML99, HE02]. These techniques can encode a great deal of information in a single

image and they often result in engaging, aesthetically pleasing visualizations [THE07]. Other examples mimic specific pen-and-ink styles of illustration, including methods for increasing emphasis in certain portions of an image by adding detail via additional pen strokes [RE01]. Many other examples build on related illustration+visualization themes, e.g. [SJEG05, HMCM].

Somewhat surprisingly, given the wealth of recent research in this area, there are only a few examples of research in art-inspired visualization that include real artists and illustrators in the process. Notable examples include the work of Donna Cox [Cox88] and collaborative work at Brown University and the Rhode Island School of Design, which has demonstrated the potential of using illustrators for expert critiques of scientific visualizations [JAL*03] and several virtual reality interfaces for applying artistic 3D modeling to visualization design [KFM*01, KZL07, KAM*08]. Our work builds on these recent results and attempts to connect research in illustrative visualization to sketch-based interfaces for illustration and design. Ultimately, the aim of our work is to provide the right interface to enable artists and illustrators to contribute to solving visual design problems in scientific domains, such as flow visualization.

2.2. Optimized Streamline Placement for Flow Visualization

Streamlines and their derivatives (streaklines, pathlines) are one of the most fundamental features that can be used to describe a fluid flow or other vector field. Since streamlines are constructed by integrating through a vector field starting from a seed point, an infinite number of streamlines can be created for a given flow field by starting from different seed points. However, including too many streamlines in a single image can clutter and confuse it, thus, the topic of picking the best set of streamlines to show in order to most accurately depict the important features of a flow is one that has received much attention in the literature.

Turk and Banks introduced an image-based technique to create images with an even density of streamlines by low-pass filtering the image generated and optimizing streamline placement to reduce variation in the image [TB96]. The visual results from this work were reproduced by Jobard and Lefer using a more deterministic algorithm with better runtime characteristics. This algorithm starts with an initial streamline and iteratively adds new streamlines at a specified distance from existing streamlines until the image is filled, ensuring a consistent density [JL97]. We introduce an extension to this algorithm to support variable density automatic streamline seeding. Variations in density can be useful for illustrating flows. Automatic algorithms have attempted to leverage this by seeding streamlines in specific patterns around critical points [VKP00] or by only displaying streamlines that are quite different from each other [LHS08]. Our interactive tool can be used to create similar results, where streamlines are placed precisely and deliberately, but through an interactive process controlled by the illustrator.

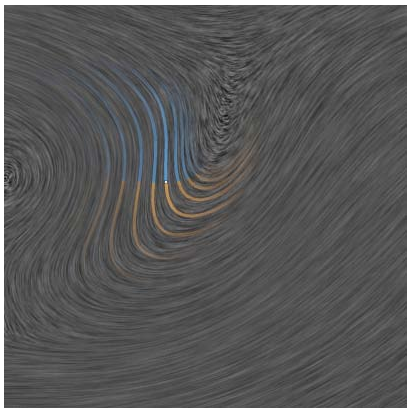

Figure 2: *A sense of the underlying flow data is provided to the illustrator using an underpainting of the flow field and a local flow preview widget (blue and orange curves) that updates interactively as the stylus moves over the display.*

2.3. Sketch-Based Interfaces in Visualization and Design

Our work builds upon a number of recent advances in sketch-based interfaces, most notably, the 3D modeling and design system, ILoveSketch [BBS08], which includes a notion of ink drying that has a similar visual aesthetic to our ink-data settling procedure. In general, our approach strives for a similar, fluid sketching, user experience. As in the Lineogrammer system [ZBLF08], many of the marks drawn by the user in our system are interpreted based upon an underlying constraint, however, in our case, these constraints come from underlying 2D vector fields.

Sketch-based interfaces have been applied to visualization applications before [Ake06, SLJ*09]. Most closely related to our work is the exploratory flow visualization system created by Isenberg et al. [IEGC08], which also supports drawing on top of fluid flows, but with the aim of creating animated, exploratory visualizations rather than our goal of static visualizations carefully designed by an illustrator.

3. Drawing with the Flow

Drawing with the Flow aims to make it as easy as possible for an illustrator to explore a variety of illustration styles using sketching, a natural mode for design work [Bux07], while also creating visualizations that are truthful in their representation of the underlying vector field data.

There are three main components to Drawing with the Flow, which are described in the following sections: 1. a

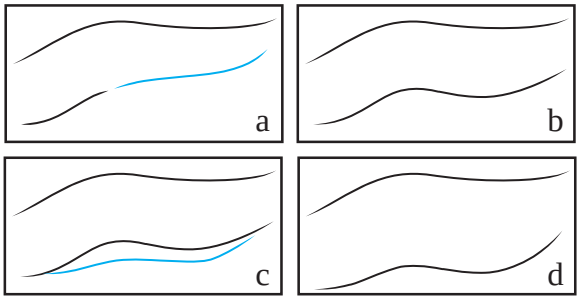

Figure 3: *The illustrator can refine existing strokes in two ways. In a, the illustrator draws an extension (in blue) to the existing stroke, which the ink-data settling algorithm refines to the image shown in b. In c, the illustrator indicates that the bottom stroke should take a different path. The program first combines the original stroke and the refinement stroke, and then settles the stroke to match the underlying data.*

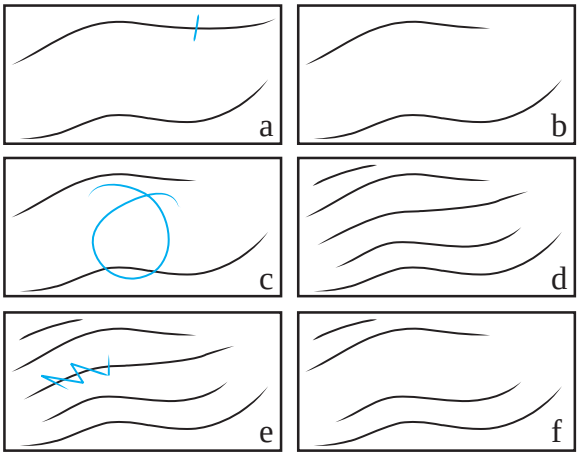

Figure 4: *Three gestures are supported: crop, more, and delete. In a, the illustrator draws a crop line (in blue) through the top stroke, which gets cropped to the result in b. In c, the illustrator performs the more gesture, which instructs the program to add more streamlines to the illustration, resulting in d. In e, the user scratches out a stroke, deleting it from the visualization, shown in f.*

sketch-based interface, 2. ink-data settling, and 3. illustration with automatic streamlines.

3.1. A Sketch-Based Interface for Illustrators

Drawing with the Flow utilizes a pen-based interface. The current implementation uses a 21-inch Wacom Cintiq tablet display. When the application first starts, it displays a subtle line-integral convolution (LIC) [CL93] visualization of the vector field as an underpainting on the display in order to provide the illustrator with a visual cue for the underlying data, see background image in Figure 2. The LIC underpainting provides information for every point in the flow without including sharp, emphasized flow lines, which makes it well suited to the task of providing the illustrator with a visual cue for the underlying data without dominating the illustration that is being designed. In general, LIC has been shown to be an effective visual style for evoking the sense of a flow, although other visualization methods are preferred for many data analysis tasks [LKJ*05].

To augment the underpainting display, a local flow preview widget was developed to provide the illustrator with a finer sense of the flow immediately under the stylus, in the style of a traditional magic lens [BSP97]. This lens draws a small number of streamlines seeded in the vicinity of the stylus point and updates interactively as the stylus is moved on the display. The streamlines in this widget are colored to indicate the direction of the flow (orange=forward flow, blue=backward flow), which is something that is not readily conveyed in LIC-style visualizations. A number of other important design decisions are included in this widget. The seeding positions of the streamlines are arranged such that the direction from one seeding point to the next is perpendicular to the flow direction, ensuring that the streamlines adequately capture twists and turns in the flow. The size of the cursor also changes in reaction to the local speed of the flow; slower speeds are represented by shorter streamlines.

Given some understanding of the structure of the underlying vector field provided by the underpainting and the local flow preview widget, the illustrator is ready to begin designing a custom streamline visualization. This is done through sketching. When the illustrator draws a stroke, it is interpreted as either a gesture, a new stroke to add to the visualization, or a refinement of an existing stroke. A refinement stroke can be either an extension of an existing stroke or a re-routing of an existing stroke, as demonstrated in Figure 3. An extension of an existing stroke is indicated by drawing a line out of the end of the existing stroke, while a re-routing is indicated by overdrawing.

Several important gestures are recognized by the system, as shown in Figure 4. A stroke drawn through an existing mark in a direction roughly perpendicular to the existing mark is recognized as a crop gesture and divides the stroke at the point of intersection, deleting the smaller half. A scribble-out gesture, as used in a number of previous sketch-based interfaces, e.g. [BBS08], is useful for quickly deleting marks. Finally, “more” and “less” operations that automatically add or delete streamlines in the illustration (described in section 3.3) are activated using clockwise and counter-clockwise loop gestures.

Strokes that are not recognized as gestures are interpreted either as new streamlines to add to the illustration or as modifications to existing streamlines. In both cases, the strokes drawn by the illustrator may not exactly match the underly-

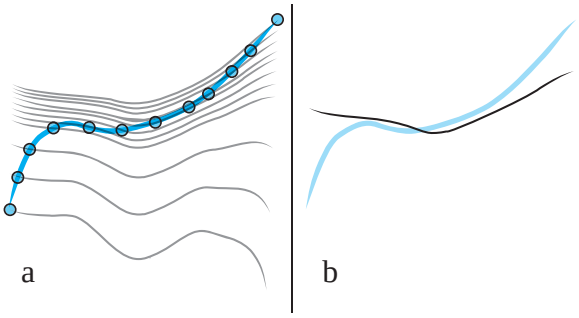

Figure 5: *Fitting a streamline to a stroke. In a, several sample points (blue circles) are chosen on the illustrator's stroke (blue stroke). Through each of these points, a streamline (gray line) is generated based on the vector field. A similarity metric is evaluated to determine which streamline is most similar to the original stroke. Part b of the figure shows the best fit in this case. The input stroke will "settle" into this streamline.*

ing vector field data. Drawing with the Flow reconciles these differences through a procedure we call ink-data settling, described in the following section.

3.2. Ink-Data Settling

Since the visualizations produced by the technique are intended to be used for flow analysis, it is important to ensure that the visualization accurately represents the underlying data. Thus, when the illustrator draws a stroke that represents a streamline, the stroke is interpreted relative to the underlying data and then refined to match these data.

The first step in this process is to identify a correspondence between the illustrator's stroke and candidate streamlines. This is done by sampling a series of candidate streamlines in the vicinity of the drawn stroke and then selecting the best streamline based upon a similarity metric. First, the illustrator's stroke is resampled to achieve an even distribution of twenty samples along its length. Then, from each of these sample points, a streamline of the same length as the illustrator's stoke is generated according to the underling data. (Figure 5 shows an example.) Each of the new candidate streamlines is compared to the original stroke using a similarity metric computed as follows. For each point on the original stroke (p_1), the nearest point on the candidate streamline is found (p_2). At each of these points, the stroke direction is found ($\vec{d}_1$ and $\vec{d}_2$, respectively). The similarity metric S for this point is then calculated as follows.

$$S = w_{dist} * \text{distance}(p_1, p_2)^2 + w_{dir} * |\vec{d}_1 \cdot \vec{d}_2|$$

The variables w_{dist} and w_{dir} determine the relative importance of the distance and direction in determining similarity. In our implementation, the values $w_{dist} = 10$ and $w_{dir} = 1$ are used with a coordinate system that ranges from zero to one across the image.

This metric has the effect of selecting streamlines that have significant overlap with the drawn stroke. We found this approach to be more intuitive than an initial version of the metric based only upon minimizing the distance between closest points, which can lead to selecting a streamline that is, on average, close to the drawn stroke, but does not often pass through the drawn stoke.

Once the match between drawn stroke and streamline has been established, an animation is used to create the visual effect of the ink settling onto the data. The animation is driven by a linear interpolation between points in the illustrator's stroke and the fitted stroke. The timing of the interpolation is set to produce an ease in and out effect.

The length and density of the streamlines are critical to the overall style of the illustration, thus these parameters are not interpreted relative to the underlying data and are instead left for the illustrator to specify directly through drawing.

3.3. Illustration with Automatic Streamlines

While generating an illustration in this style, one tedious task that can arise is filling in a region of the drawing with "similar" streamlines. This section describes an algorithm to assist the illustrator in this task. The technique is controlled via the "more" and "less" gestures shown in Figure 4. The intent is to utilize the context provided by the illustrator's recent actions to determine a region and style (length, density) for new streamlines to automatically add to the illustration. From the illustrator's viewpoint, the interface should feel intuitive, e.g. "give me more of what I just did."

3.3.1. Review of Constant Density Streamline Seeding

Our algorithm extends Jobard and Lefer's constant density automatic streamline seeding algorithm [JL97]. The original algorithm is reviewed briefly here, our extensions are described in detail in the next section. The algorithm iteratively attempts to add new streamlines to gaps in the image given the constraint that new streamlines must be a specified distance (d_{sep}) away from all existing lines. The algorithm begins with a single original streamline added to a queue, then iterates as follows:

- Pop the next streamline S from the queue.
- For each sample point along S find the candidate seed points p_1 and p_2 that are a perpendicular distance d_{sep} from the streamline on both sides of the streamline.
- Check each point p_1 and p_2. If the distance from the point to any other existing streamline is $< d_{sep}$ then discard the point.
- Otherwise, create a new streamline at the location of the candidate point, integrating both forward and backward through the vector field until the streamline either exits

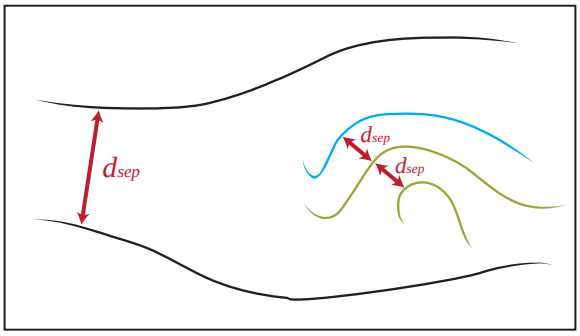

Figure 6: *Automatically generated streamlines are seeded by examining the local separation of streamlines in the illustration. Here, the user has already drawn the black streamlines and the green streamlines. The green streamlines were drawn most recently, so new streamlines will be added close to these lines. In this region of the image, the separation of streamlines d_{sep} is relatively small. Thus, when a new streamline is added in this region (the blue line), this small, local separation value is used to seed the line.*

the flow or comes too close to another existing streamline. (Too close is typically defined as within $0.5 * d_{sep}$ of another streamline.)
- Add the newly created streamline to the queue.

3.3.2. Extension to Support Variable Density Seeding

We introduce an extension to this algorithm to support variable spacing between streamlines, which is an important aspect of many hand-drawn flow illustrations. Rather than a global d_{sep}, our approach varies the ideal value for the separation of streamlines at a local level throughout the image. Since streamlines can be long and travel through a large portion of the image, setting the separation distance at the streamline level is not sufficient, thus, we store a local value for d_{sep} for each sample in each streamline. This local value is used to find the candidate points p_1 and p_2 in the original algorithm.

To drive this variable density seeding strategy from illustrator input, the local d_{sep} for each sample on each streamline needs to be computed. This value should be representative of the local separation of neighboring streamlines, thus, the algorithm sets the local d_{sep} value for each streamline sample to be equal to the distance from that sample to the closest streamline.

3.3.3. Automatic Streamline Generation

Figure 6 demonstrates the use of the variable density streamline seeding algorithm within a flow illustration. Several streamlines have already been drawn by hand and are shown in black. The illustrator then performs a "more" gesture, as described earlier, which initiates the automatic seeding and displays a small circular widget to indicate the gesture was recognized. Now, as the stylus continues to move in a circular pattern around the widget, a single new streamline is automatically added to the image for each quarter revolution of the pen around the circle. Since just a single streamline is added at a time, there is no need store a queue of streamlines as described in the original seeding algorithm. Instead, drawn streamlines and sample points are selected at random until a new successful candidate streamline is found.

The automatically generated streamlines can be removed by simply reversing the direction of rotation around the circle widget. Returning to the forward direction of rotation will begin adding new streamlines again, this time picking new candidate seed points. This allows illustrators to scrub back and forth using the circle widget to explore different seeding possibilities.

As shown in Figure 6, recently drawn lines are used to infer attributes for the automatically generated streamlines. The length of new streamlines is constrained to be equal to the average length of the last two lines that were drawn by the illustrator. This constraint is removed if the last two drawn lines were longer than half of the total width of the image. The system interprets this as meaning that the illustrator intends to draw long lines that likely exit the flow field on both ends.

Automatically generated lines are also constrained to appear within a region of recent activity, if such a region can be identified from the illustrator's recent drawing. If the last two drawn lines are within a circular region with a radius of twenty percent of the image, only candidate points inside this region will be considered.

4. Results

Figures 7–9 show result illustrations created using Drawing with the Flow running on an Intel Core 2 Duo 2.66 GHz Mac Mini with a GeForce 9400 and a Cintiq 21UX Wacom Tablet.

Figure 7 shows two illustrations that together demonstrate how 2D simulated flow past a cylinder changes depending upon the Reynolds number used in the simulation (Re=100 vs. Re=500). Each illustration was completed in approximately twenty minutes. The illustrative style utilized adds detail to selected areas in each illustration to highlight the differences between the two cases.

Figure 8 demonstrates the variety of illustrative styles that can be achieved, even when working with the same dataset. The minimalist style on the left is similar to what a fluid mechanics professor might draw on paper to illustrate this canonical flow case to a student. The short lines style utilized the "more" gesture to quickly populate the background of the image. Each of these illustrations took from five to ten minutes to create. The final two illustrations demonstrate

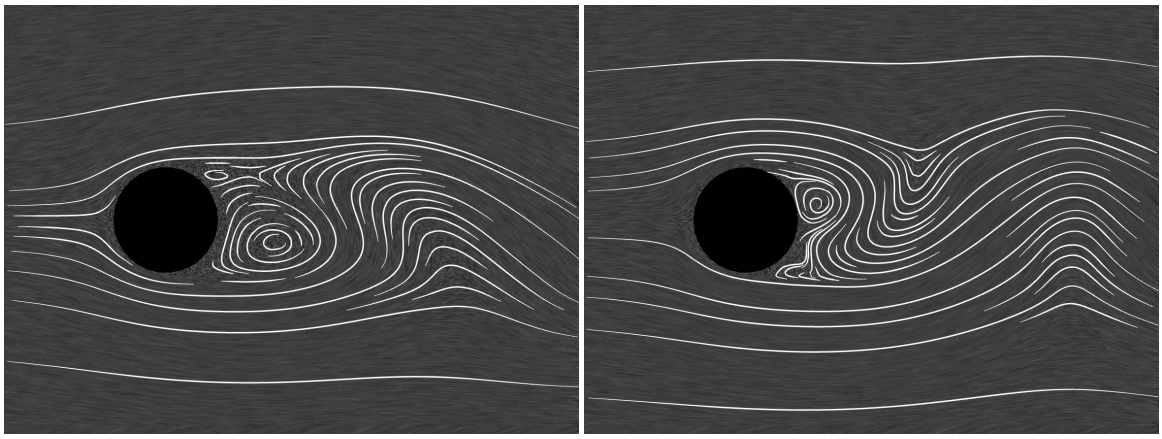

Figure 7: *Two illustrative visualizations comparing flow past a cylinder with Reynolds number 100 (left) and 500 (right). The illustrator has attempted to emphasize the differences between the two flows.*

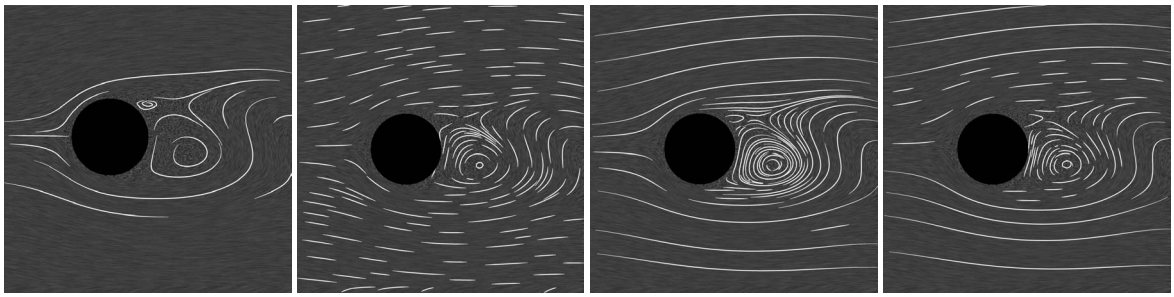

Figure 8: *Illustrative visualizations of the same flow in four distinct styles, varying streamline density, length, and positioning.*

additional styles made possible by selectively including or excluding detail from the images. These took approximately twenty minutes to create.

Figure 9 shows illustrations of the type of 2D vector fields that are often used to evaluate automatic streamline seeding algorithms. Each of these fields contains at least one critical point. A smart selection of streamlines should make these critical points easy to identify, while a poor selection of streamlines can make them difficult to find. Each of these visualizations took approximately ten minutes to complete; the illustrator decided which streamlines to draw in each of these cases based on the goal of making critical points in the flow easy to identify.

5. Conclusion

In this paper, we presented a sketch-based system for creating illustrative visualizations of 2D vector fields. The system is accessible to illustrators and provides several tools to facilitate creating visualizations in a variety of illustrative styles, including styles that adjust visual emphasis to clearly depict specific features in the data. Although the approach supports a great deal of artistic freedom, it also maintains the constraint that the marks displayed are accurate with respect to the underlying vector field data. In addition to addressing the very specific visual design problem of placing streamlines to illustrate 2D vector fields, we hope this sketch-based interface can also serve as an example to more broadly motivate future work in the area of making data-driven visualization tools accessible to illustrators, artists, and other visual thinkers.

References

[Ake06] AKERS D.: CINCH: A cooperatively designed marking interface for 3D pathway selection. In *Symposium on User Interface Software and Technology* (2006), pp. 33–42.

[BBS08] BAE S., BALAKRISHNAN R., SINGH K.: ILoveSketch: as-natural-as-possible sketching system for creating 3D curve models. In *Proceedings of the 21st annual ACM symposium on User interface software and technology* (2008), ACM, pp. 151–160.

[BSP97] BIER E., STONE M., PIER K.: Enhanced illustration using magic lens filters. *IEEE Comput. Graph. Appl. 17*, 6 (1997), 62–70.

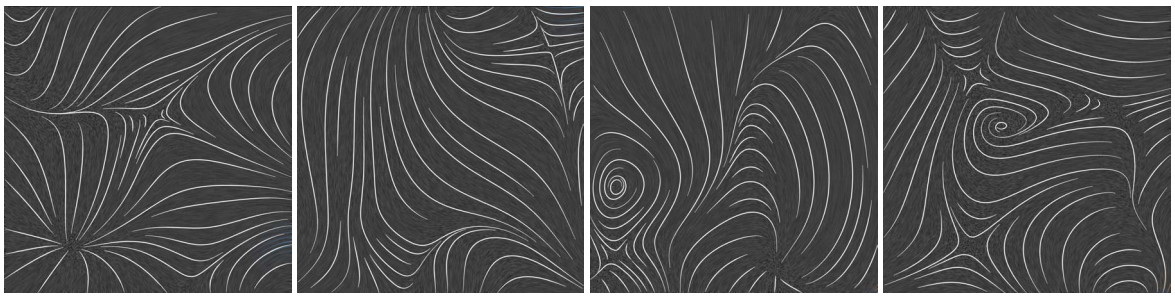

Figure 9: *Illustrations of 4 fields containing different sets of critical points. Detail was added to highlight critical points, whereas areas of little change are left relatively open.*

[Bux07] BUXTON B.: *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann, 2007.

[CL93] CABRAL B., LEEDOM L. C.: Imaging vector fields using line integral convolution. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1993), ACM, pp. 263–270.

[Cox88] COX D.: Using the supercomputer to visualize higher dimensions: An artist's contribution to scientific visualization. *Leonardo 21*, 3 (1988), 233–242.

[HE02] HEALEY C. G., ENNS J. T.: Perception and painting: A search for effective, engaging visualizations. *IEEE Computer Graphics and Applications 22*, 2 (2002), 10–15.

[HMCM] HSU W., MEI J., CORREA C., MA K.: Depicting Time Evolving Flow with Illustrative Visualization Techniques. *Arts and Technology*, 136–147.

[IEGC08] ISENBERG T., EVERTS M., GRUBERT J., CARPENDALE S.: Interactive Exploratory Visualization of 2D Vector Fields. In *Computer Graphics Forum* (2008), vol. 27, Blackwell Publishing, pp. 983–990.

[JAL*03] JACKSON C., ACEVEDO D., LAIDLAW D. H., DRURY F., VOTE E., KEEFE D.: Designer-critiqued comparison of 2D vector visualization methods: A pilot study. ACM SIGGRAPH 2003 Sketches and Applications, July 2003.

[JL97] JOBARD B., LEFER W.: Creating evenly-spaced streamlines of arbitrary density. *Visualization in Scientific Computing 97* (1997), 43–56.

[KAM*08] KEEFE D. F., ACEVEDO D., MILES J., DRURY F., SWARTZ S. M., LAIDLAW D. H.: Scientific sketching for collaborative VR visualization design. *IEEE Transactions on Visualization and Computer Graphics 14*, 4 (2008), 835–847.

[KFM*01] KEEFE D. F., FELIZ D. A., MOSCOVICH T., LAIDLAW D. H., LAVIOLA JR. J. J.: CavePainting: A fully immersive 3D artistic medium and interactive experience. In *Proceedings of I3D 2001* (2001), pp. 85–93.

[KKL05] KIRBY M. M., KEEFE D. F., LAIDLAW D. H.: *The Visualization Handbook. Painting and Visualization*. Elsevier Inc., 2005, pp. 873–891.

[KML99] KIRBY M., MARMANIS H., LAIDLAW D. H.: Visualizing multivalued data from 2D incompressible flows using concepts from painting. In *Proceedings of IEEE Visualization 1999* (1999), pp. 333–340.

[KZL07] KEEFE D. F., ZELEZNIK R. C., LAIDLAW D. H.: Drawing on air: Input techniques for controlled 3D line illustration. *IEEE Transactions on Visualization and Computer Graphics 13*, 5 (2007), 1067–1081.

[LHS08] LI L., HSIEH H., SHEN H.: Illustrative streamline placement and visualization. In *IEEE Pacific Visualization Symposium* (2008), pp. 79–86.

[LKJ*05] LAIDLAW D. H., KIRBY M., JACKSON C., DAVIDSON J. S., MILLER T., DASILVA M., WARREN W., TARR M.: Comparing 2d vector field visualization methods: A user study. *IEEE Transactions on Visualization and Computer Graphics 11*, 1 (2005), 59–70.

[LM02] LUM E. B., MA K.-L.: Hardware-accelerated parallel non-photorealistic volume rendering. In *Proceedings of the International Symposium on Nonphotorealistic Animation and Rendering* (2002).

[LS05] LEONARDO, SUH H. A.: *Leonardo's notebooks / Leonardo da Vinci ; edited by H. Anna Suh*. Black Dog & Leventhal, New York :, 2005.

[MLP*09] MCLOUGHLIN T., LARAMEE R., PEIKERT R., POST F., CHEN M.: Over two decades of integration-based, geometric flow visualization. *Eurographics 2009, State of the Art Reports* (2009), 73–92.

[RE01] RHEINGANS P., EBERT D.: Volume illustration: Nonphotorealistic rendering of volume models. *IEEE Transactions on Visualization and Computer Graphics 7*, 3 (2001), 253–264.

[SJEG05] SVAKHINE N., JANG Y., EBERT D., GAITHER K.: Illustration and photography inspired visualization of flows and volumes. *IEEE Visualization, 2005. VIS 05* (2005), 687–694.

[SLJ*09] SOWELL R., LIU L., JU T., GRIMM C., ABRAHAM C., GOKHROO G., LOW D.: Volume viewer: an interactive tool for fitting surfaces to volume data. In *SBIM '09: Proceedings of the 6th Eurographics Symposium on Sketch-Based Interfaces and Modeling* (New York, NY, USA, 2009), ACM, pp. 141–148.

[TB96] TURK G., BANKS D.: Image-guided streamline placement. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (1996), ACM, p. 460.

[THE07] TATEOSIAN L., HEALEY C., ENNS J.: Engaging viewers through nonphotorealistic visualizations. In *Proceedings of the 5th international symposium on Non-photorealistic animation and rendering* (2007), ACM, p. 102.

[VKP00] VERMA V., KAO D., PANG A.: A flow-guided streamline seeding strategy. In *Proceedings of the conference on Visualization'00* (2000), IEEE Computer Society Press, p. 170.

[ZBLF08] ZELEZNIK R. C., BRAGDON A., LIU C.-C., FORSBERG A.: Lineogrammer: creating diagrams by drawing. In *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology* (New York, NY, USA, 2008), ACM, pp. 161–170.