International Journal for Uncertainty Quantification, 3 (3): 225–240 (2013)

PREDICTABILITY-BASED ADAPTIVE MOUSE INTERACTION AND ZOOMING FOR VISUAL FLOW EXPLORATION

Marcel Hlawatsch, [*] ***Filip Sadlo, & Daniel Weiskopf***

VISUS, University of Stuttgart, Stuttgart, Germany

Original Manuscript Submitted: 12/31/2011; Final Draft Received: 07/25/2012

Flow fields are often investigated by adopting a Lagrangian view, for example, by particle tracing of integral curves such as streamlines and path lines or by computing delocalized quantities. For visual exploration, mouse interaction is predominantly used to define starting points for time-dependent Lagrangian methods. This paper focuses on the uncertainty of mouse input and its impact on the visualization process. In typical cases, the interaction is achieved by mouse motion, exhibiting uncertainty in the range of a screen pixel. From the perspective of dynamical systems theory, an integral curve represents an initial value problem, the uncertainty a perturbation of its initial condition, and the uncertainty of the visualization procedure a predictability problem. Predictability analysis is concerned with the growth of perturbations under the action of flow. In our case, it is not unusual that the perturbations grow from single pixels to substantial deviations. We therefore present an interaction scheme based on the largest finite-time Lyapunov exponent and the flow map gradient, providing accurate, smooth, and easy-to-use flow exploration. This scheme employs data-driven adaptation of mouse speed and direction as well as optional augmentation by an adaptive zoom lens with consistent magnification. We compare our approach to nonadaptive mouse interaction and demonstrate it for several examples of data sets. Furthermore, we present results from a user study with nine domain experts.

KEY WORDS: *interactive flow visualization, predictability, finite-time Lyapunov exponent, adaptive mouse speed, adaptive zoom*

1. INTRODUCTION

In visualization, uncertainty is present in manifold forms. Most important, it is present in the data to visualize (1) and in the visualization techniques we apply. Focusing on the techniques, there are again different manifestations of uncertainty. A major, but unfortunately yet widely ignored, source of uncertainty is on the algorithmic side of visualization techniques (2). Assuming that uncertainty sources (1) and (2) are negligible or not subject to investigation, there is another major source of uncertainty: user input (3). Uncertainty in user input can emerge from insufficient control or insufficient (e.g., only pixel-accurate) precision of the input device with respect to the visualization task. Finally, uncertainty is also present on the output side, i.e., in the display (4). There, the fixed resolution of the display can lead to uncertainty due to discretization errors.

In this paper, we address the latter two aspects (3) and (4). Focusing on interactive visualization with a feedback loop (Fig. 1), we provide means that adapt the user's mouse input to its uncertainty. The effectiveness of the interaction loop is further improved by reducing the uncertainty on the output side with a data-driven zoom lens. Both methods help the users reach their aimed result faster, i.e., to gain relevant information for their task in a more efficient way. In the case of flow exploration, a prominent example of an aimed result is the determination from where a quantity, such as heat, is transported.

In our work, we consider the visualization of vector fields. Thereby, we base our technique on interactive visualization using integral curves such as streamlines and path lines. The uncertainty with respect to user input poses a

[*]Correspond to Marcel Hlawatsch, E-mail: hlawatsch@visus.uni-stuttgart.de

2152–5080/13/$35.00

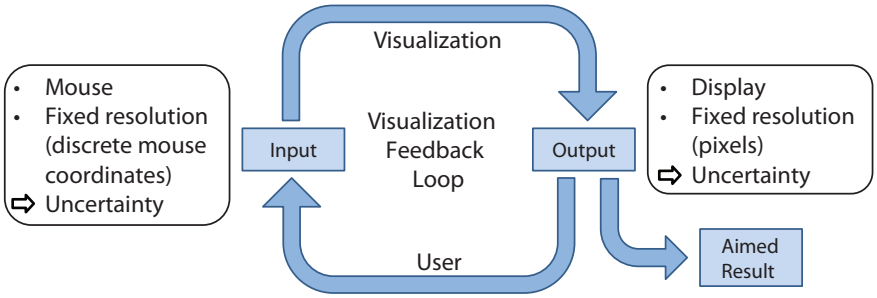

FIG. 1: Convergence to aimed result within the feedback loop of interactive visualization. Convergence speed is affected by uncertainty on the input and output side.

predictability problem in that context. User input includes the seeding location, seeding time, and integration length of the curves. In the Lagrangian framework, often the end points of integral curves are of special interest since they provide information about transport and interrelation. In terms of the predictability problem, the perturbations are associated with the user input, and it is of interest how these perturbations influence the resulting integral curves, in particular their end points. Since both seeding time and advection time are typically entered numerically, we do not address these two sources of error but focus on the seed placement using the mouse.

A prominent and widely used concept for quantifying predictability is the Lyapunov exponent (in problems with infinite time domain) and its variant for finite-time domains called finite-time Lyapunov exponent (FTLE). Since the Lyapunov exponent is identical to the FTLE in its special case of infinite advection time, we build upon the FTLE for generality. As detailed in Section 3, the FTLE represents a conservative measure for the growth of perturbations over finite advection time intervals (Fig. 2). Hence, it provides a quantitative measure of the deviation of the end points of streamlines or path lines with respect to the uncertainty of their seeds—constituting the basis of our predictability-based interactive flow visualization technique.

2. RELATED WORK

Uncertainty in visualization and its visualization remain important problems [1]. Zuk and Carpendale [2] provide a theoretical analysis of uncertainty visualization. Pang et al. [3] give an overview of different methods for the visualization of uncertainty. Recent publications cover the extraction of isosurfaces from uncertain scalar fields [4, 5] and the exploration of continuous parameter spaces considering uncertainty [6]. There are a couple of different approaches

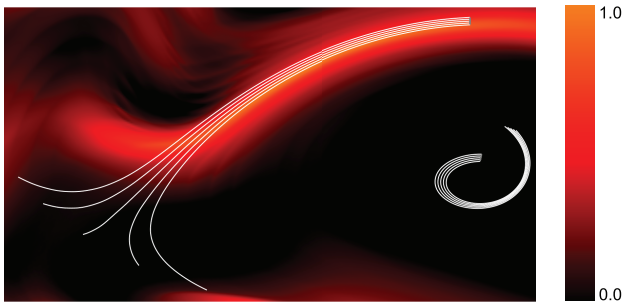

FIG. 2: The FTLE and predictability. (a) Nearby seeded path lines started in regions with low FTLE values (center right) exhibit coherent behavior, whereas identically seeded path lines started at locations with high FTLE (upper right) exhibit divergent behavior. Hence, the predictability of path line behavior is low when interactively seeding in areas of high FTLE. (b) This color map is used for FTLE visualization throughout this paper except for the images from the user study (Fig. 11). Negative FTLE values are clamped to zero prior to normalization.

to the visualization of uncertainty in flow: glyph-based methods [7, 8], texture-based methods [9, 10], a combination of both for bidirectional vector fields [11], uncertainty techniques vector field topology [12, 13], and the collection of geometric and animated techniques [14]. Waser et al. [15] discuss uncertainty in computational steering of flow simulations.

The other aspect relevant to our paper is interactive seeding of integral curves. For a comprehensive overview of streamline seeding, we refer to the survey by McLoughlin et al. [16]. Laramee describes various seeding techniques for flow visualization [17, 18]. Interactive seeding in an virtual environment is part of the work by Bryson and Levit [19]. Bürger et al. [20] present a seeding approach for integral curves based on the FTLE. Interactive seeding in blood flow data is presented by Van Pelt et al. [21]. However, all these techniques do not address seeding in the context of uncertain user input.

In general, there is plenty of previous research on data-driven user input adaptation. Many examples are concerned with one-dimensional (1D) input [22–24], e.g., for browsing through lists. However, we are not aware of any previous work on uncertainty-guided seeding of integral curves. The other adaptation aspect of our paper is related to variable zooming. Previous examples of adaptive zooming include speed-dependent zooming for browsing large documents [25], smooth zooming and panning for two-dimensional (2D) maps [26], and automatic zooming in scrolling interfaces [27]. However, none of these papers considers flow fields.

To the best of our knowledge, there is no specific previous work on the uncertainty of user input in the context of Lagrangian flow visualization. The only work in this direction is by Brecheisen et al. [28], who perform sensitivity analysis for DTI fiber tracking. In contrast to our work, they focus on tensor fields instead of vector fields and on stopping criteria for tracking instead of seeding and flow field analysis.

Finally, the work by Sanyal et al. [29] is related to our paper in the sense that they also perform a user study on uncertainty visualization techniques. However, they investigate the visualization of uncertain data, whereas we consider uncertainty in user input and display output.

We base our predictability analysis on the FTLE, in particular, in combination with the flow map. The flow map-based FTLE was made popular by Haller [30] (see, e.g., [31] and the references in [32, 33] for prior works). An overview of topology-based and FTLE-related techniques can be found in the recent survey by Pobitzer et al. [34]. FTLE computations are time-consuming and can be accelerated by hierarchical integration and by GPU calculations [35], which we use in this paper. The FTLE in the context of uncertain vector fields is discussed by Schneider et al. [36], once again without considering uncertainty in user input and display output.

3. FINITE-TIME LYAPUNOV EXPONENT AND FLOW MAP

This section briefly reviews the background of the FTLE, which is central to our method. The FTLE provides a conservative measure for predictability of trajectory end points with respect to their seeds, and motivates our predictability-based techniques presented in Section 4.

The Lyapunov exponent (LE) was originally introduced for predictability analysis in temporally unconstrained dynamical systems. For an n-dimensional system, or vector field, it is a spectrum of n exponents, of which the largest, σ_1, is defined as

$$\sigma_1(\mathbf{x}) = \lim_{T\to\infty} \lim_{\|\boldsymbol{\delta}(\mathbf{x},t_0)\|\to 0} \frac{1}{|T|} \ln \frac{\|\boldsymbol{\delta}(\mathbf{x}, t_0+T)\|}{\|\boldsymbol{\delta}(\mathbf{x}, t_0)\|}, \tag{1}$$

where $\boldsymbol{\delta}(\mathbf{x}, t)$ is the perturbation at time t, having originated at position $\mathbf{x}$ and time t_0, and oriented at time t_0 such that σ_1 becomes maximal. In other words, the perturbation $\boldsymbol{\delta}(\mathbf{x}, t) = \boldsymbol{\xi}(t, \mathbf{x}+\boldsymbol{\delta}(\mathbf{x}, t_0), t_0) - \boldsymbol{\xi}(t, \mathbf{x}, t_0)$ represents the deviation of the perturbed trajectory $\boldsymbol{\xi}(t, \mathbf{x}+\boldsymbol{\delta}(\mathbf{x}, t_0), t_0)$ from the reference trajectory $\boldsymbol{\xi}(t, \mathbf{x}, t_0)$, both started at time t_0 but at positions $\mathbf{x}+\boldsymbol{\delta}(\mathbf{x}, t_0)$ and $\mathbf{x}$, respectively. The largest exponent σ_1 is of major importance since it represents the upper bound for the growth of a perturbation applied at location $\mathbf{x}$ and time t_0, and it is therefore often denoted as the Lyapunov exponent itself. A counterpart for time-constrained problems is the finite-time Lyapunov exponent, which is commonly computed using the *flow map*

$$\boldsymbol{\xi}_{t_0}^{T} : \mathbb{R}^n \to \mathbb{R}^n, \mathbf{x} \mapsto \boldsymbol{\xi}_{t_0}^{T}(\mathbf{x}), \tag{2}$$

mapping from the seed $\mathbf{x}$ at time t_0 of a trajectory to its end point after advection for finite time T. One reason for the success of the FTLE in visualization is that vector field data are often temporally bounded and hence only amenable to analysis by temporally finite concepts.

The maximum perturbation growth corresponds to the major eigenvector of the right Cauchy-Green deformation tensor

$$\mathbf{C}_{t_0}^{T}(\mathbf{x}) = [\nabla \boldsymbol{\xi}_{t_0}^{T}(\mathbf{x})]^{\top} \nabla \boldsymbol{\xi}_{t_0}^{T}(\mathbf{x}). \tag{3}$$

In other words, a perturbation oriented in this direction at time $t_0 + T$ will have experienced locally maximum growth under the action of the flow for time T. Hence, predictability is lowest with respect to these perturbations at time t_0 and position $\mathbf{x}$. The maximum growth factor can be obtained using the spectral norm of the flow map gradient $\nabla \boldsymbol{\xi}_{t_0}^{T}(\mathbf{x})$, the square root of the major eigenvalue $\lambda_{\max}(\cdot)$ of $\mathbf{C}_{t_0}^{T}(\mathbf{x})$. Hence, the maximum FTLE, in this paper simply denoted as the FTLE or σ, at position $\mathbf{x}$, time t_0, and advection time T, computes

$$\sigma_{t_0}^{T}(\mathbf{x}) = \frac{1}{|T|} \ln \sqrt{\lambda_{\max}(\mathbf{C}_{t_0}^{T}(\mathbf{x}))}. \tag{4}$$

We provide an example of the relation between the FTLE field and the predictability of selected trajectories in Fig. 2.

The FTLE has recently gained importance in the field of time-dependent vector field topology. Haller [30] proposed to use ridges in the FTLE field to identify Lagrangian coherent structures. These separate the domain into regions of qualitatively similar flow behavior and hence provide an overall view of the space-time structure of time-dependent vector fields. Note that our work is *not* concerned with topology, we address predictability. Although the ridges in the FTLE field exhibit locally maximum FTLE and hence locally minimum predictability, this correlation is only implicit, i.e., we do not account for FTLE ridges.

4. INTERACTION METHOD

Our method addresses two topics: predictability-based interaction (Section 4.1) and zooming (Section 4.2). These building blocks can be used separately, but provide synergies when combined.

4.1 Predictability-Based Mouse Interaction

We base adaptive interaction on predictability by introducing a data-driven mapping from the mouse input to the mouse cursor. Since the FTLE represents unpredictability, the basic idea is to decelerate the mouse according to the FTLE field. This improves interaction by providing better control of seed placement in regions with low predictability, i.e., high uncertainty of the result with respect to the user input. Furthermore, it allows for more coherent and convergent interactive visualization sessions. Compared to the traditionally interleaved zooming phases for seed placement and visualization phases for investigating integral curves (Fig. 1), our method avoids loss of context that would be introduced by switching between different views and scales. It provides a coherent yet accurate exploration. Nevertheless, our method does not replace traditional mouse interaction. We allow the user to freely switch between standard interaction and our technique to combine the best of the two. For example, it turned out in our user study (Section 7) that standard interaction is often preferred for navigation and coarse exploration, whereas our approach serves well for detailed analysis.

4.1.1 Coordinate Systems

Our method involves three different coordinate systems: (mouse) input coordinates $\mathbf{m} \in \mathbb{Z}^p$ from the operating system, our subpixel accurate cursor coordinates $\tilde{\mathbf{m}} \in \mathbb{R}^p$, and the physical coordinates $\mathbf{x} \in \mathbb{R}^n$ of the respective n-dimensional data set. Common desktop mouse interaction results in $p = 2$, whereas interaction with a three-dimensional (3D) mouse involves $p = 3$. The data-driven mapping $\mathbf{m} \mapsto \tilde{\mathbf{m}}$ is provided by our technique, whereas the mapping $\tilde{\mathbf{m}} \mapsto \mathbf{x}$ is accomplished by a simple transformation $\mathbf{x} = \boldsymbol{\phi}(\tilde{\mathbf{m}})$, e.g., by back projection. Due to the adaptive nature of our approach, we do not maintain an explicit mapping $\mathbf{m} \mapsto \tilde{\mathbf{m}}$ but rather map changes: Obtaining the input vector $\Delta\mathbf{m}_i = \mathbf{m}_{i+1} - \mathbf{m}_i$, we apply the mapping $\Delta\mathbf{m}_i \mapsto \Delta\tilde{\mathbf{m}}_i$ and compute $\tilde{\mathbf{m}}_{i+1} = \tilde{\mathbf{m}}_i + \Delta\tilde{\mathbf{m}}_i$. The corresponding physical coordinates are evaluated by $\mathbf{x}_{i+1} = \boldsymbol{\phi}(\tilde{\mathbf{m}}_{i+1})$.

4.1.2 Adaptation

Assuming small changes $\Delta\mathbf{m}_i$, $\Delta\tilde{\mathbf{m}}_i$, and $\Delta\mathbf{x}_i$, a first-order approximation by linearization with infinitesimal vectors $\mathrm{d}\mathbf{m}_i$, $\mathrm{d}\tilde{\mathbf{m}}_i$, and $\mathrm{d}\mathbf{x}_i$ can be used:

$$\mathrm{d}\mathbf{x}_i = [\nabla\boldsymbol{\phi}(\tilde{\mathbf{m}}_i)]\mathrm{d}\tilde{\mathbf{m}}_i. \tag{5}$$

From this follows, with the definition $s_\Phi(\tilde{\mathbf{m}}_i) := \|[\nabla\boldsymbol{\phi}(\tilde{\mathbf{m}}_i)]\mathrm{d}\tilde{\mathbf{m}}_i\| \,/\, \|\mathrm{d}\tilde{\mathbf{m}}_i\|$:

$$\|\mathrm{d}\mathbf{x}_i\| = s_\Phi(\tilde{\mathbf{m}}_i)\,\|\mathrm{d}\tilde{\mathbf{m}}_i\|\,. \tag{6}$$

We now assume that $\boldsymbol{\phi}(\tilde{\mathbf{m}})$ is restricted to translation, rotation, and uniform scaling, valid and constant for the whole scene. In the case $p = n-1$, e.g., 2D mouse controlling a position in 3D space, $\mathbf{x}$ is assumed to stay on a user-defined hyperplane. With these assumptions, $s_\Phi(\tilde{\mathbf{m}}_i)$ is a constant scalar factor, i.e.,

$$\|\mathrm{d}\mathbf{x}_i\| = s_\Phi\,\|\mathrm{d}\tilde{\mathbf{m}}_i\| = \|[\nabla\boldsymbol{\phi}(\tilde{\mathbf{m}}_i)]\|\,\|\mathrm{d}\tilde{\mathbf{m}}_i\|\,. \tag{7}$$

For our data-driven adaptation, we require the growth factor $\gamma(\mathbf{x}_i)$ of a perturbation $\mathrm{d}\mathbf{x}_i$, seeded at position $\mathbf{x}_i$ and resulting in the perturbation $\mathrm{d}\boldsymbol{\xi}_i$ after advection. In our case, the perturbation corresponds to the motion of the trajectory's seed from position $\mathbf{x}_i = \boldsymbol{\phi}(\tilde{\mathbf{m}}_i)$ due to cursor displacement $\mathrm{d}\tilde{\mathbf{m}}_i$:

$$\|\mathrm{d}\boldsymbol{\xi}_i\| = \gamma(\mathbf{x}_i)\,\|\mathrm{d}\mathbf{x}_i\| = \gamma(\boldsymbol{\phi}(\tilde{\mathbf{m}}_i))s_\Phi\,\|\mathrm{d}\tilde{\mathbf{m}}_i\|\,. \tag{8}$$

For the adaptation, we now require that the size of the perturbation $\mathrm{d}\boldsymbol{\xi}_i$ is proportional to the size of the input vector $\mathrm{d}\mathbf{m}_i$, with user-defined ratio s_u controlling the speed of the end point of the trajectory:

$$\|\mathrm{d}\boldsymbol{\xi}_i\| = \gamma(\boldsymbol{\phi}(\tilde{\mathbf{m}}_i))s_\Phi\,\|\mathrm{d}\tilde{\mathbf{m}}_i\| \overset{!}{=} s_u\,\|\mathrm{d}\mathbf{m}_i\|\,, \tag{9}$$

leading to

$$\|\mathrm{d}\tilde{\mathbf{m}}_i\| = (s_u/s_\Phi)\,\|\mathrm{d}\mathbf{m}_i\|\,/\gamma(\boldsymbol{\phi}(\tilde{\mathbf{m}}_i)). \tag{10}$$

Hence, the growth factor $\gamma(\boldsymbol{\phi}(\tilde{\mathbf{m}}_i))$ needs to be determined for the adaptation.

A straightforward approach would be to derive $\gamma(\mathbf{x})$ from the FTLE. In this case, the maximum growth factor from the definition of the FTLE (Section 3) would be used:

$$\gamma_F(\mathbf{x}) = \sqrt{\lambda_{\max}(\mathbf{C}_{t_0}^T(\mathbf{x}))} = \exp(\sigma_{t_0}^T(\mathbf{x})\cdot T) \tag{11}$$

However, as noted in Section 3, the FTLE represents only an upper bound of a typically anisotropic growth property: differently oriented perturbations usually undergo different growth rates. Thus, the above adaptation would not depend on the orientation of $\mathrm{d}\mathbf{m}$.

Instead, we aim at adapting the mouse input speed such that, given constant mouse motion speed, the end point of the seeded trajectory also moves at constant speed. The aforementioned straightforward approach would accomplish this only for mouse motion along the maximizing perturbation. Thus, we account for the effective growth factor $\gamma_G(\mathbf{x})$ of a perturbation corresponding to $\mathrm{d}\mathbf{m}$ at $\mathbf{x} = \boldsymbol{\phi}(\tilde{\mathbf{m}})$. This can be achieved using the flow map gradient $\nabla\boldsymbol{\xi}_{t_0}^T(\mathbf{x})$:

$$\gamma_G(\boldsymbol{\phi}(\tilde{\mathbf{m}})) = \left\| \nabla\boldsymbol{\xi}_{t_0}^T(\boldsymbol{\phi}(\tilde{\mathbf{m}}))\frac{\nabla\boldsymbol{\phi}(\tilde{\mathbf{m}})\mathrm{d}\mathbf{m}}{\|\nabla\boldsymbol{\phi}(\tilde{\mathbf{m}})\mathrm{d}\mathbf{m}\|} \right\|. \tag{12}$$

Here, we have assumed that our adaptation does not change the direction of the mouse input, i.e., $\mathrm{d}\tilde{\mathbf{m}} \propto \mathrm{d}\mathbf{m}$.

Similarly, for the discretized computation, the adapted motion vector $\Delta\tilde{\mathbf{m}}_i$ has the same direction as the input vector $\Delta\mathbf{m}_i$ and is only scaled according to (10):

$$\Delta\tilde{\mathbf{m}}_i = (s_u/s_\Phi)\Delta\mathbf{m}_i/\gamma(\boldsymbol{\phi}(\tilde{\mathbf{m}}_i)). \tag{13}$$

If $\Delta\mathbf{m}_i$ is too large for approximation by linearization, we split our adaptation into k parts by subdividing $\Delta\mathbf{m}_i$ and applying (13) for each of its substeps $\Delta\mathbf{m}_i' = \Delta\mathbf{m}_i/k$. Large k increases the accuracy, but also the computational

effort, hence, k has to be limited to maintain interactive rates. We used $k = 5$ in our experiments (see Section 5 for performance details).

Figure 3 illustrates how our method works in an interactive exploration application based on path lines. The results were generated with simulated mouse input to assure comparability. Figure 3(b) shows how cursor motion is decelerated when moving into areas of low predictability. This helps analyze the flow structure; incoherent motion of the end point of the path line is avoided. In regions of high predictability, the cursor can move faster. The higher mouse speed there allows a fast exploration without the risk of missing important details. Note that we clamp the growth factor $\gamma(\mathbf{x})$ to 1.0 if $\gamma(\mathbf{x}) < 1.0$ for better usability. This limits the cursor speed $\Delta\tilde{\mathbf{m}}$ to the input speed $\Delta\mathbf{m}$, i.e., we avoid small mouse motions from resulting in large cursor motions.

4.1.3 2D and 3D Applications

Our description so far holds for adapting mouse motion in data sets of arbitrary dimension n. The only part that does not generalize in a straightforward manner is $\boldsymbol{\phi}(\cdot)$, the mapping from input coordinates to physical coordinates. In 2D, $\boldsymbol{\phi}(\cdot)$ typically represents translation and uniform scaling only. In the case of 3D data and if a 3D input device is used, $\mathbf{m} \in \mathbb{R}^3$ and $\tilde{\mathbf{m}} \in \mathbb{R}^3$. In this case, $\boldsymbol{\phi}(\cdot)$ may reduce to the identity map $\boldsymbol{\phi}(\tilde{\mathbf{m}}) \equiv \tilde{\mathbf{m}}$ in our formulation. However, if 2D mouse input is used for 3D data sets, $\boldsymbol{\phi}(\cdot)$ maps from $\mathbb{R}^2$ to $\mathbb{R}^3$. The involved mapping requires the definition of depth. In this case, we use a common approach, a user-defined hyperplane to which the cursor is aligned, called "exploration plane" in our experiments (Fig. 10).

4.2 Predictability-Based Zoom Lens

There are also situations where uncertainty on the *output* side hinders convergence of the interactive visualization feedback loop (Fig. 1). To account for these cases where uncertainty due to discretization of the display into pixels, i.e., undersampling, affects appropriate perception, we introduce a zoom lens with data-driven magnification. In accordance with our overall approach, we keep the lens centered at the (subpixel accurate) mouse cursor position. At the same time, this improves convergence of interactive visualization by automatically adjusting views and scales. In contrast to the adaptation of the mouse motion, where anisotropic adaptation (12) based on $\nabla\boldsymbol{\xi}_{t_0}^T(\mathbf{x})$ provides adequate behavior, we apply *isotropic* magnification adaptation for the lens, based on the FTLE. In our implementation, the radius of the lens is kept fixed; however, it would be straightforward to implement a lens with a user-controlled radius.

A prominent scenario for the zoom lens, also present in our user study (Section 7), is the inspection of finely folded FTLE ridges [Fig. 4(a)]. Trajectories started in these regions typically undergo massive separation, and it is

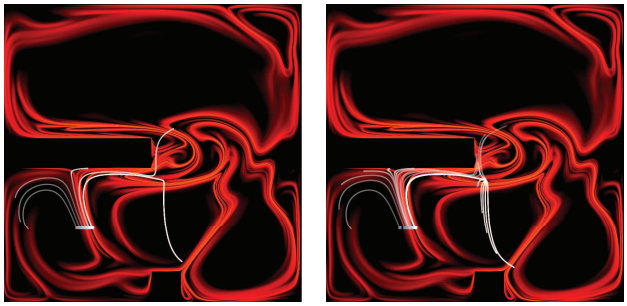

FIG. 3: Comparison of direct (a) and adaptive (b) mouse input for interactive exploration with path lines. For both images, the same simulated mouse input with constant speed and direction was used. The FTLE field is shown in the background. The direct method exhibits uniform motion of the path line seed, whereas the adaptive method exhibits uniform motion of its end point. The temporal sequence of path line motion is shown in a single image with older steps having higher transparency.

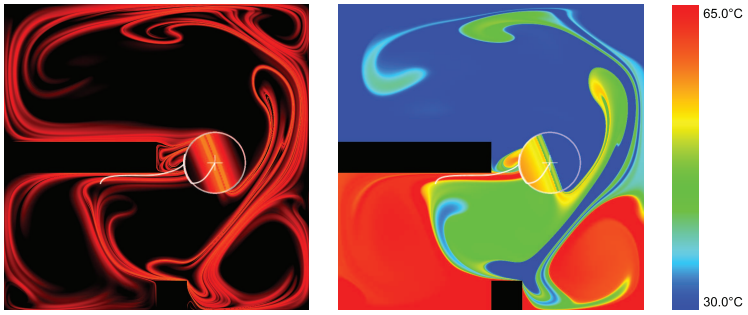

FIG. 4: Applications for the adaptive zoom lens. (a) Analysis and interpretation of ridge lines in the FTLE field. (b) Visualization of delocalized temperature [37], i.e., temperature averaged along path lines. (c) This color map is used for all images with temperature distribution.

often unclear where our subpixel accurate mouse cursor is located, i.e., at which of the finely folded FTLE ridges the trajectory was started. Moving with the subpixel accurate mouse cursor and adapting the magnification to the FTLE, our lens provides appropriate view and scale for interpreting the underlying dynamics. Another application for the lens is the visualization of delocalized criteria [37] or the advection of quantities such as vorticity [38], which also exhibit fine-scale structures related to path line behavior [Fig. 4(b)].

The zoom lens can be freely turned on and off in our technique. When active, the lens is kept centered with $\boldsymbol{\phi}(\tilde{\mathbf{m}})$, the position of our subpixel accurate mouse cursor. We also evaluate the FTLE $\sigma_{t_0}^{T}(\boldsymbol{\phi}(\tilde{\mathbf{m}}))$ there and adjust the magnification factor of the lens to

$$f_z \cdot \sigma_{t_0}^{T}(\boldsymbol{\phi}(\tilde{\mathbf{m}})) \tag{14}$$

during interaction. The parameter f_z defaults to 1 and allows the user to adapt the range of magnification to their needs.

From a symmetry point of view (Fig. 1), the straightforward choice for controlling the magnification level of the lens would use the growth factor γ_F (11), instead of $\sigma_{t_0}^{T}(\boldsymbol{\phi}(\tilde{\mathbf{m}}))$. Although this would result in a motion of the content within the lens conforming to the speed of the end point of the respective trajectory (the speed factor between the seed and the end point would be compensated by the lens magnification), it would exhibit a major drawback: since separation factors in the flow map typically vary by several orders of magnitude, the same highly dynamic variation would appear in the magnification—leading to incoherent visualization. The logarithmic mapping inherent in the FTLE (4) and the independence from the orientation of mouse motion calm down the temporal magnification change and motivate (14). In Section 6.4, we present results for the zoom lens with direct and with adaptive mouse input. In our experiments with 3D data sets, the lens is kept aligned to the exploration plane [see Fig. 10(c)]. Since the magnification operates on this plane, the FTLE $\sigma_{t_0}^{T}(\boldsymbol{\phi}(\tilde{\mathbf{m}}))$ in (14) is computed by projecting the flow map gradient $\nabla\boldsymbol{\xi}_{t_0}^{T}(\mathbf{x})$ to the plane.

5. IMPLEMENTATION

Our prototype is implemented with the Open Graphics Library (OpenGL), the OpenGL Utility Toolkit (GLUT), and the Compute Unified Device Architecture (CUDA). OpenGL is used for rendering, GLUT is used for platform-independent management of the graphical user interface, and CUDA is used for fast computation of flow maps and integral curves.

Since we are to manipulate the mapping from mouse input to the mouse cursor, we need access to this mapping. Normally, the window manager controls this mapping and the application receives the window coordinates $(\mathbf{m})$ of mouse events. A first step is therefore to manage and display our own subpixel accurate cursor $(\tilde{\mathbf{m}})$ in the application to gain control over mouse cursor motion. To avoid confusing the user with two cursors moving at different speeds and directions, we hide the cursor of the window manager when displaying our own.

A remaining problem is that the cursor of the window manager cannot move beyond the border of the desktop. As a result, our cursor would be blocked in the respective direction. Thus, we set the hidden cursor of the window manager back after every mouse move event. Setting it back to the center of the desktop assures sufficient range for possible mouse movement.

Our method to alter the user's mouse input $\Delta\mathbf{m}$ can now be integrated as described in Section 4.1. The flow map gradient $\nabla\boldsymbol{\xi}_{t_0}^T(\boldsymbol{\phi}(\tilde{\mathbf{m}}))$ and the FTLE $\sigma_{t_0}^T(\boldsymbol{\phi}(\tilde{\mathbf{m}}))$ at position $\boldsymbol{\phi}(\tilde{\mathbf{m}})$ are computed on-the-fly. The GPU implementation allows us to calculate the four (2D) or six (3D) path lines required for the central-difference-based gradient computation at interactive frame rates. The motion vector $\Delta\mathbf{m}$ is then used as described to adapt the internal cursor $\tilde{\mathbf{m}}$. Even though the internal cursor is only displayed with pixel accuracy, its floating-point accurate representation is used for all purposes, such as the seeding of the integral curves and positioning of the zoom lens. Including the on-the-fly integration of trajectories and all other operations, our technique exhibits, compared to traditional mouse interaction, a computational overhead of about 10 milliseconds for $k = 5$ substeps (Section 4.1) for the example from Fig. 9.

The zoom lens is implemented in the fragment shader, which is executed when displaying precomputed quantities like the FTLE field. Complete FTLE fields are typically precomputed because their computational expense prohibits on-the-fly computation. Precomputed FTLE fields are stored in a texture; texture coordinates are transformed with respect to the obtained magnification factor. If the available computational power is high enough, the content inside the zoom lens, e.g., the FTLE field, can be computed on-the-fly at the respective zoom level. This can improve the accuracy of the visualization compared to simple magnification of the precomputed field.

For the analysis of 3D flow, the user defines the position and the normal of the exploration plane with a widget. This plane is used for the transformation from 2D mouse coordinates $\mathbf{m}$ to 3D physical coordinates $\mathbf{x}$ (see Section 4.1). This transformation is further defined such that the cursor moves along the respective image space axes. Hence, the path line seed moves in the expected way, but aligned to the exploration plane. In the 3D case, the lens is still 2D but aligned to the exploration plane. Thus, it is implemented in the fragment shader as in the 2D case.

6. EXAMPLES

We demonstrate our method using three 2D time-dependent data sets and one 3D stationary data set. The analytical quad gyre data set [33] is an extension of the double gyre [39]. It was sampled at a resolution of 161×161 nodes and 241 time steps. The buoyant flow data set resulted from a computational fluid dynamic (CFD) simulation of buoyant air flow in a closed container with barriers, where the bottom is heated and the top cooled. Its resolution is 101×101 nodes and it spans 1001 time steps. The next data set is also from CFD simulation and exhibits a von Kármán vortex street in water flow. Its resolution is 101×301 nodes with 801 time steps. The last data set is a 3D CFD simulation of a static mixer. It features two tangential inlets, one for hot air and one for cold, and an outlet for the mixed flow. For our GPU-based implementation, we resampled the unstructured grid at $201 \times 301 \times 201$ nodes.

The seeds of the path lines are marked by small squares in all images. The temporal sequence of a moving path line is merged into a single image and transparency is used to emphasize temporal progression with older path lines exhibiting higher transparency.

6.1 Quad Gyre

Figure 5 shows a result for the quad gyre data set, with a forward ($T > 0$) path line overlaid on the corresponding FTLE field. Path line motion resulting from mouse input moving upward, across the ridge at the center of the domain, is visualized. In contrast to the incoherent visualization from direct mouse input [Fig. 5(a)], our method [Fig. 5(b)] provides a smooth exploration of the diverging flow, revealing the flow behavior responsible for the ridge.

6.2 Buoyant Flow

In the case of the buoyant flow data set, the temperature distribution was examined (Fig. 6). Warm air is advected from the bottom wall, whereas cold air is transported from the top wall. Concurrently, thermal conductivity leads to

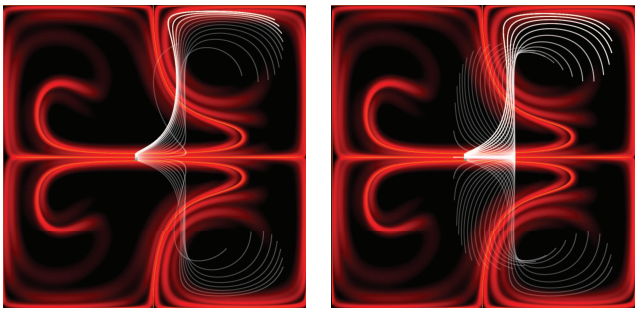

FIG. 5: Results for the quad gyre data set. (a) Resulting path line motion for direct mouse input in upward direction with the highest possible accuracy bounded by the screen resolution. The result for the identical input with our method is shown in (b). Our method provides much higher accuracy for interactive seed placement.

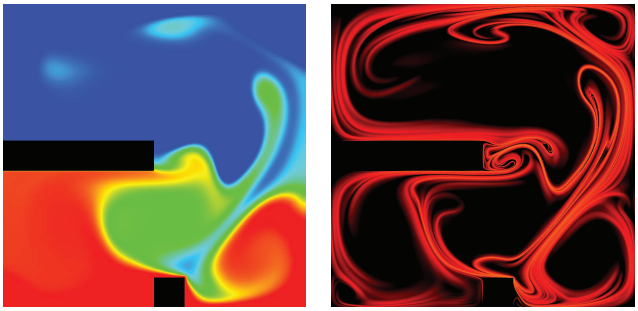

FIG. 6: Overview on the buoyant flow data set. Temperature distribution (a) and the corresponding reverse ($T < 0$) FTLE field (b) are shown.

diffusion, equaling temperature distribution. The hot and cold air is predominantly mixing in the lower right quarter. The correlation between temperature and the reverse ($T < 0$) FTLE is apparent.

The results from interactive exploration with a path line are shown in Fig. 7. Reverse path lines show from where a quantity is advected. Here, we again compare direct mouse input with our method. To obtain comparable results,

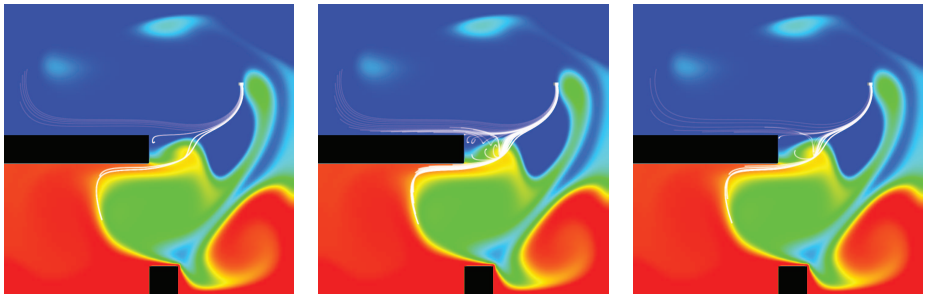

FIG. 7: Results for the buoyant flow data set with reverse path lines. Simulated path line seed motion with direct mouse input at the highest accuracy (single pixel steps) is visualized in (a). Image (b) shows the result with our method applied to the same mouse input. Because of the deceleration of our method, typical user input would consist of faster motion. A possible result may then look like the result in (c). It can be clearly seen that cold air was advected from the upper half, whereas hot air was entrained from the left lower quarter.

the same generated mouse input was used. Using direct mouse input, the accuracy is bounded by the screen resolution [Fig. 7(a)]. In this case, it is difficult to analyze the mixing behavior of the flow. With our method, the same user input results in a much higher accuracy of the seeding [Fig. 7(b)]. The path line behavior can be clearly identified and tracked. As visible in the image, our method provides a very dense sampling of the flow behavior. Therefore, the user is able to use faster mouse input without missing important details [Fig. 7(c)].

6.3 Von Kármán Vortex Street

With the data set of the von Kármán vortex street, we demonstrate how other visualization techniques, like the visualization of vorticity magnitude in this case, can be enhanced with our interactive seeding approach [Fig. 8(a)]. It is apparent that vorticity is advected from the obstacle but at the same time experiences diffusion. Figure 8(b) demonstrates that our interaction and exploration concept is also suitable in the context of FTLE-based topological analysis of CFD flow fields. The adaptive interaction allows the user to precisely control the interactive seeding of the path line, allowing for a detailed inspection of the flow behavior that led to a given FTLE ridge. In the case of Fig. 8(b), flow separation at the obstacle helps the user understand the origin of the FTLE ridge.

6.4 Zoom Lens

Our predictability-driven zoom lens is particularly useful for inspecting transport phenomena in quantities that exhibit fine structure. One such example is the investigation of delocalized temperature [37], shown in Fig. 4 for the buoyant flow data set. Another example is the analysis of FTLE fields. Figure 9 demonstrates the usage of the adaptive zoom lens for the analysis of the FTLE field in the same data set. The adaptive zoom magnifies regions with high FTLE values and low predictability. It is apparent in Fig. 4 that regions with different delocalized temperature values correspond to regions of qualitatively different behavior and hence their interfaces correspond to FTLE ridges. The FTLE-dependent magnification is also highly useful at positions with neighboring ridge lines (Fig. 4). In combination with the visualization of the path line seeded at the analyzed position, the interpretation of ridge lines in the FTLE field is substantially eased. As a comparison of Figs. 9(a)–9(c) and Figs. 9(d)–9(j) shows, it is advantageous to combine the adaptive zoom lens with adaptive mouse input.

6.5 Static Mixer

The result for the 3D flow in the mixer data set is shown in Fig. 10. The same behavior as in the 2D case can be observed. With direct mouse input [Fig. 10(a)], flow features can be missed because the motion of the path line end point is not uniform. With our predictability-based input adaptation [Fig. 10(b)], uniform movement of the end point

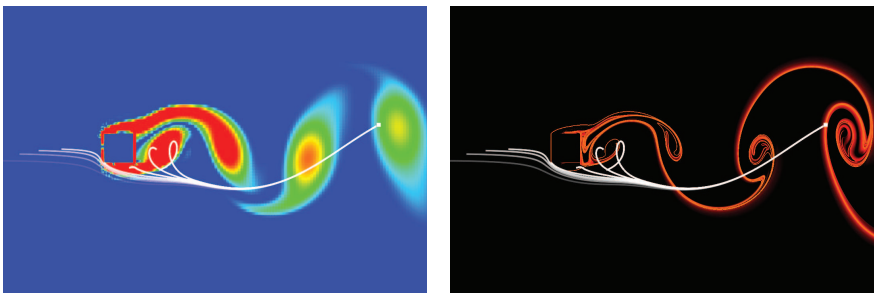

FIG. 8: Results for the von Kármán vortex street data set. Simulated interactive exploration using a reverse path line with adaptive mouse motion to the right. Image (a) shows the vorticity magnitude in the background visualized with the rainbow color map similar to Fig. 4(c) (blue: low values, red: high values). It is apparent that vorticity is advected from the shear flow at the surface of the obstacle by flow separation. (b) Moving path line from (a) now overlaid on the reverse FTLE field of the data set explains the presence of the FTLE ridge (flow separation at obstacle).

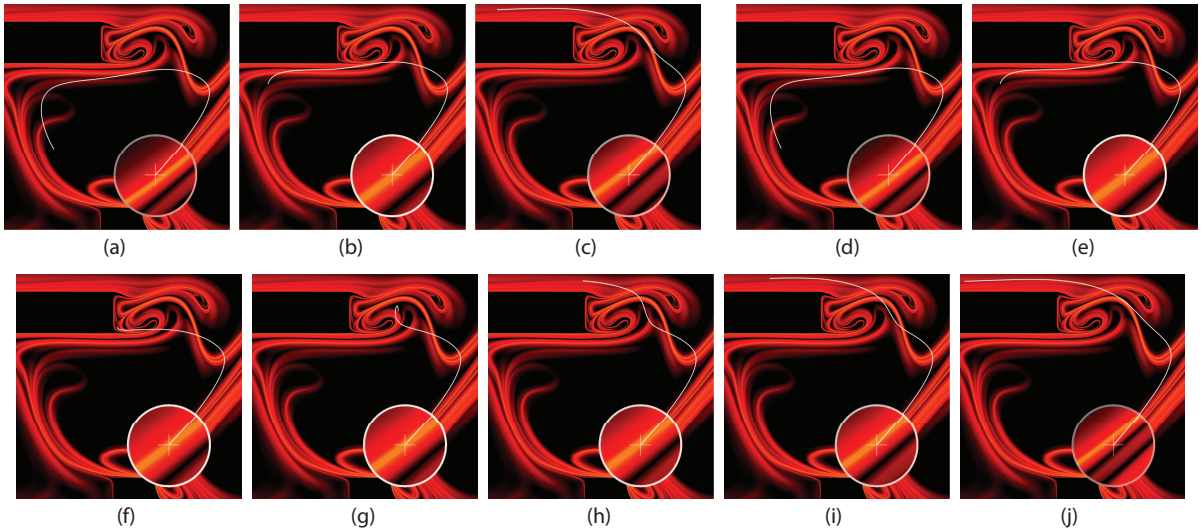

FIG. 9: Application of the adaptive zoom lens for the analysis of FTLE ridges in the buoyant flow data set. Images (a)–(c) show a sequence of motion steps with direct input. Images (d)–(j) show a sequence of motion steps with adaptive input. The same simulated mouse input and starting position was used for both results. The adaptive method allows a more precise movement of the zoom lens and the path line. This allows one to navigate between the finely folded ridges with a smoother change of the adaptive magnification of the lens.

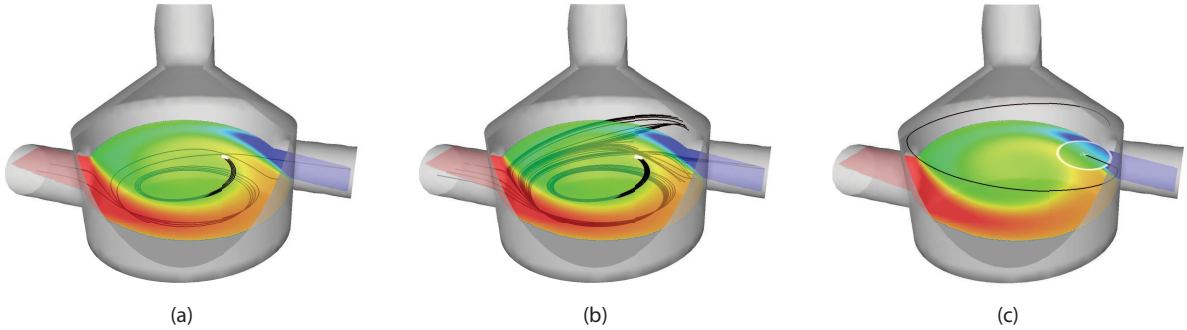

FIG. 10: Example of 3D flow in the mixer data set. Images (a) and (b) show the temporal sequence of a moving path line (black for better contrast, seed marked with white square) controlled with the same simulated mouse input and starting from the same initial position. Temperature is mapped onto the exploration plane (see Section 4.1.3). The path line in (a) was moved with direct mouse input, the path line in (b) with adaptive mouse input. Direct input leads to uniform movement of the path line seed, which has the effect that the path line does not capture all features of the 3D flow. With our adaptive method, the motion of the path line end point is uniform, leading to a better exploration of the flow. Image (c) shows the application of the adaptive zoom lens to 3D flow.

is achieved, which can reveal additional structures in the flow. Due to the 3D data, the adaptive zoom lens operates here on the exploration plane [Fig. 10(c)].

7. USER STUDY

We conducted a user study with nine domain experts from the field of CFD simulation and flow visualization to assess the usefulness of our adaptation method. We compared adaptive mouse input to direct mouse input and tested the utility of the adaptive zoom lens.

7.1 Study Procedure

For the user study, two 2D transient data sets were used: the buoyant flow data set (Section 6.2) and a data set from a CFD simulation of flow around several obstacles [Fig. 11(a)]. The principal flow direction in the latter data set is from the left to the right.

The participants were researchers from different departments of the University of Stuttgart. They had to accomplish five different tasks (Fig. 11). For every task, both interaction methods had to be used in separate passes. In the case of the adaptive method, the participants were allowed to use direct mouse input for exploration, but had to place lines with the adaptive method. The participants were allowed to adjust the speed s_u of both methods during the tasks. Different starting times t_0 were used for every pass to avoid the use of gained knowledge about the data. The order of interaction methods was permuted and balanced between tasks to assure a fair comparison and compensate for learning effects.

We tried to keep the number of tasks and hence the time required for the study low to maintain the motivation and attention of our participants during all tasks. Furthermore, we wanted to focus on the effectiveness and utility of the mouse input adaptation. Therefore, we decided to collect more data about the comparison of direct and adaptive mouse input and tested the adaptive zoom lens only in the last task.

In an initial training phase, the software and all of its features were demonstrated to the participants and they had time to try out the tool and to set up the initial speed of both methods. After each task, the participants were

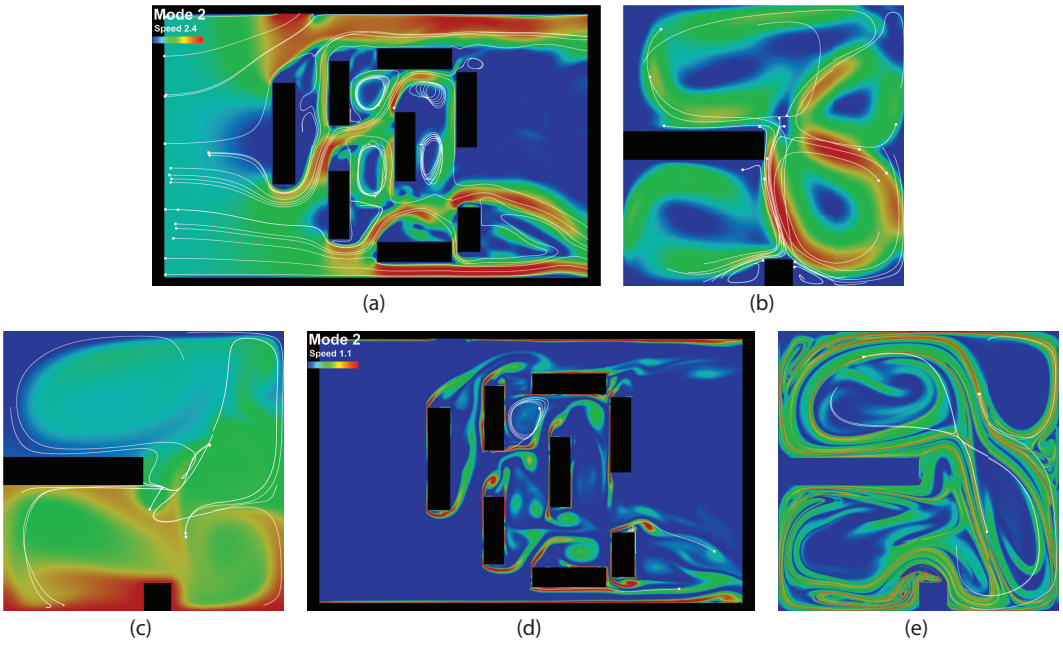

FIG. 11: Screenshots for the five tasks of the user study. Placement of representative path lines for: (Task 1) the obstacle data set (a) and (Task 2) the buoyant flow data set (b). Velocity magnitude at the starting time (t_0) is shown in the background. (c) Analysis and visualization of heat transport in the buoyant flow data set (Task 3), temperature at time t_0 is displayed in the background. (d) Analysis and visualization of vorticity transport in the obstacle data set (Task 4), vorticity magnitude is shown in the background. Reverse path lines are shown in (c) and (d). (e) Analysis of ridges in the FTLE field (background) of the buoyant flow data set (Task 5). All path lines were manually placed by the user during the study, their seeds are marked with white squares. A darkened rainbow color map (blue: low values, red: high values) was used for all background visualizations in the study to emphasize the path lines. In the upper left corner of images (a) and (d), the user-selected interaction mode, mouse speed, and the color map of the background visualization are shown. This area was cut out in the other screenshots to just show the quadratic domain of the data set.

interviewed. They were asked to decide with which method they solved the task better and to rate the benefit of the preferred method compared to the other one. They were also asked to give comments about both methods. To obfuscate that we focused on the interaction method and avoid biased ratings, we not only posed questions about the interaction method, but also about the value of path line visualization and its interactive placement. On average, it took around one hour to complete all tasks and answer all questions. The study supervisor interviewed the participants and observed how they worked on the tasks. The study sessions were additionally recorded on camera for later evaluation and screenshots were taken from the solutions of the participants for the different tasks (Fig. 11). The goal was to primarily perform a qualitative evaluation with additional quantitative subjective measurements.

7.2 Tasks

The experts worked on the following tasks:

Task 1 The participants had to place path lines to generate a representative visualization of the obstacle flow data set [Fig. 11(a)] within a time limit of 3 min for every pass.

Task 2 Same as Task 1 but for the buoyant flow data set [Fig. 11(b)].

Task 3 The participants had to analyze transport of heat in the buoyant flow data set [Fig. 11(c)]. Reverse path lines were displayed to indicate the origin of transport. They had to find and precisely mark three positions where the origin of path lines is abruptly changing between the cool area at the top and the warm area at the bottom.

Task 4 Similar to Task 3, the participants had to analyze transport of vorticity in the obstacle data set [Fig. 11(d)] with reverse path lines. They had to find three positions with increased vorticity and precisely mark the location on the boundary where vorticity originated.

Task 5 Ridge lines in the FTLE field of the buoyant flow data set were analyzed in this task [Fig. 11(e)]. The participants had to visualize the separating flow with path lines for three different positions. In this task, the adaptive zoom lens combined with adaptive mouse input was tested additionally.

7.3 Results

Figure 12 shows the preferred interaction methods together with ratings of how much the preferred method outperformed the other. The ratings are in the range from 0.0 (very low benefit) to 1.0 (very high benefit). There is no clear winner for explorative tasks, like in the first two tasks of the study. Even participants that preferred the adaptive method liked to initially explore the data with the direct method. The experts who preferred the direct method said

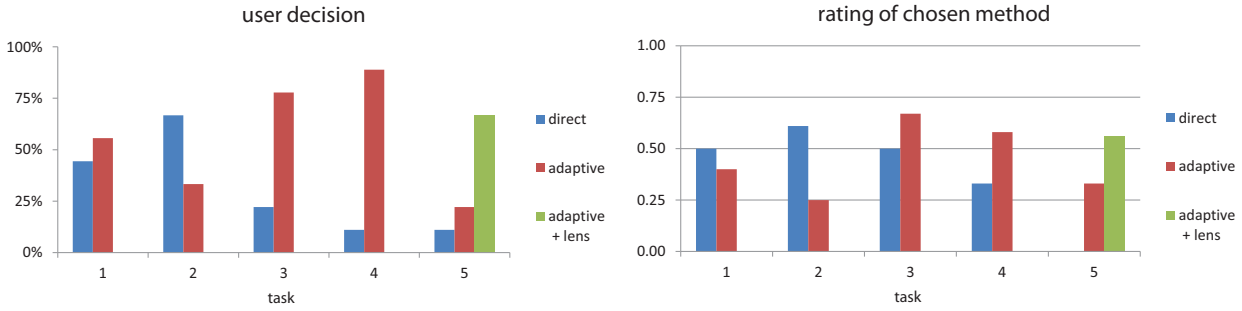

FIG. 12: Results of the user study. The upper chart shows which method was preferred by how many participants. The lower shows the average rating by all participants who preferred the respective method.

that the adaptive mode was too slow and that they did not need its precision for this task. When it comes to detailed analysis of flow behavior at specific positions, e.g., the analysis of transport or FTLE ridges, most participants preferred adaptive mouse input. They liked its higher precision and the sensitivity to the underlying data. For the analysis of FTLE fields, most participants saw a benefit in the adaptive zoom lens. Observations during the study showed that the reasons for slowing down the mouse are not always clear to the user if the FTLE is not visible.

We also asked the participants to rate the interactive placement of path lines. They all rated it very helpful (0.9 average rating over all tasks). Even when an algorithm can generate a meaningful seeding, which is difficult for path lines, they would like to further explore the data with path lines and place additional lines.

8. CONCLUSION AND FUTURE WORK

We have presented two approaches for handling uncertainty in the feedback loop of interactive visualization: predictability-based adaptation of mouse motion for input uncertainty and predictability-based zooming for output uncertainty. Both methods can accelerate the convergence to the aimed visualization result, especially in combination.

By adapting the user input to the predictability in the explored area, the user is naturally guided and does not have to change between interaction styles when exploring areas of low predictability. The risk to miss important features is substantially lowered and the overall exploration is made more efficient. We demonstrated our approach for the interactive seeding of path lines, a common tool in flow visualization. The results show that the analysis of important features in flow fields is actively supported. The application of the method to 3D data is straightforward as shown with our example. With the predictability-based zoom lens, there is no need to manually adapt the zooming level and view, which can lead to context loss or missing important features.

The results from the study show that the adaptive input method improves the detailed analysis of flow and its features. For exploration on larger scales, direct mouse input is preferred. The results of the study also show that our adaptive zoom lens can improve the analysis of FTLE fields.

A drawback of the input adaptation method is that the resulting cursor motion is not always comprehensible for the user. This is especially the case when the FTLE field is not displayed during the interactive exploration. However, since we allow to interactively switch between traditional and predictability-based interaction, the user can choose the optimal technique at any time during interaction.

Future work could include the application of our adaptation concept to other explorative tools. With haptic devices, it is possible to provide the user with force feedback that may further improve exploration of flow fields. In 3D applications, perception of path line motion is influenced by the view. Therefore, it would be interesting to adapt the viewing parameters to improve the perception of path line motion.

ACKNOWLEDGMENTS

The authors would like to thank the German Research Foundation (DFG) for financial support of the project within the *Cluster of Excellence in Simulation Technology* (EXC 310/1) and the Collaborative Research Centre SFB-TRR 75 at the University of Stuttgart. The authors would also like to thank Holger Theisel for helpful discussions.

REFERENCES

1. Johnson, C., Top scientific visualization research problems, *IEEE Comput. Graphics Appl.*, 24(4):13–17, 2004.
2. Zuk, T. and Carpendale, S., Theoretical analysis of uncertainty visualizations, In *Proceedings of SPIE-IS&T Conference on Electronic Imaging*, pp. 66–79, 2006.
3. Pang, A. T., Wittenbrink, C. M., and Lodha, S. K., Approaches to uncertainty visualization, *Visual Comput.*, 13(8):370–390, 1997.
4. Pfaffelmoser, T., Reitinger, M., and Westermann, R., Visualizing the positional and geometrical variability of isosurfaces in uncertain scalar fields, *Comput. Graphics Forum*, 30(3):951–960, 2011.
5. Pöthkow, K., Weber, B., and Hege, H.-C., Probabilistic marching cubes, *Computer Graphics Forum*, 30(3):931–940, 2011.

6. Berger, W., Piringer, H., Filzmoser, P., and Gröller, E., Uncertainty-aware exploration of continuous parameter spaces using multivariate prediction, *Comput. Graphics Forum*, 30(3):911–920, 2011.
7. Hlawatsch, M., Leube, P., Nowak, W., and Weiskopf, D., Flow radar glyphs–static visualization of unsteady flow with uncertainty, *IEEE Trans. Visualization Comput. Graphics*, 17(12):1949–1958, 2011.
8. Wittenbrink, C. M., Pang, A. T., and Lodha, S. K., Glyphs for visualizing uncertainty in vector fields, *IEEE Trans. Visualization Comput. Graphics*, 2(3):266–279, 1996.
9. Allendes Osorio, R. S. and Brodlie, K. W., Uncertain flow visualization using LIC, In *Proceedings of EG UK Theory and Practice of Computer Graphics*, pp. 1–9, 2009.
10. Botchen, R., Weiskopf, D., and Ertl, T., Texture-based visualization of uncertainty in flow fields, In *Proceedings of IEEE Visualization*, pp. 647–654, 2005.
11. Zuk, T., Downton, J., Gray, D., Carpendale, S., and Liang, J., Exploration of uncertainty in bidirectional vector fields, In *Proceedings of SPIE-IS&T Conference on Electronic Imaging*, 2008.
12. Otto, M., Germer, T., Hege, H.-C., and Theisel, H., Uncertain 2D vector field topology, *Comput. Graphics Forum*, 29(2):347–356, 2010.
13. Otto, M., Germer, T., and Theisel, H., Uncertain topology of 3D vector fields, In *Proceedings of IEEE PacificVis*, pp. 67–74, 2011.
14. Lodha, S. K., Pang, A., Sheehan, R. E., and Wittenbrink, C. M., UFLOW: Visualizing uncertainty in fluid flow, In *Proceedings of IEEE Visualization*, pp. 249–254, 1996.
15. Waser, J., Ribicic, H., Fuchs, R., Hirsch, C., Schindler, B., Bloschl, G., and Gröller, M., Nodes on ropes: A comprehensive data and control flow for steering ensemble simulations, *IEEE Trans. Visualization Comput. Graphics*, 17(12):1872–1881, 2011.
16. McLoughlin, T., Laramee, R. S., Peikert, R., Post, F. H., and Chen, M., Over two decades of integration-based, geometric flow visualization, *Comput. Graphics Forum*, 29(6):1807–1829, 2010.
17. Laramee, R. S., Interactive 3D flow visualization using a streamrunner, In *Extended Abstracts of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 804–805, 2002.
18. Laramee, R. S., *Interactive 3D Flow Visualization Based on Textures and Geometric Primitives*, PhD Thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology, 2004.
19. Bryson, S. and Levit, C., The virtual wind tunnel, *IEEE Comput. Graphics Appl.*, 12:25–34, 1992.
20. Bürger, K., Kondratieva, P., Krüger, J., and Westermann, R., Importance-driven particle techniques for flow visualization, In *Proceedings of IEEE PacificVis*, pp. 71–78, 2008.
21. Pelt, R.v., Olivan Bescos, J., Breeuwer, M., Clough, R., Gröller, M., Haar Romeny, B.t., and Vilanova, A., Interactive virtual probing of 4D MRI blood-flow, *IEEE Trans. Visualization Comput. Graphics*, 17(12):2153–2162, 2011.
22. Ahlberg, C. and Shneiderman, B., The alphaslider: A compact and rapid selector, In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 365–371, 1994.
23. Appert, C. and Fekete, J.-D., Orthozoom scroller: 1D multi-scale navigation, In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 21–30, 2006.
24. Masui, T., Kashiwagi, K., and Borden, G. R., Elastic graphical interfaces to precise data manipulation, In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 143–144, 1995.
25. Igarashi, T. and Hinckley, K., Speed-dependent automatic zooming for browsing large documents, In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pp. 139–148, 2000.
26. Wijk, J.v. and Nuij, W., A model for smooth viewing and navigation of large 2D information spaces, *IEEE Trans. Visualization Comput. Graphics*, 10(4):447–458, 2004.
27. Cockburn, A., Savage, J., and Wallace, A., Tuning and testing scrolling interfaces that automatically zoom, In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 71–80, 2005.
28. Brecheisen, R., Vilanova, A., Platel, B., and Haar Romeny, B.t., Parameter sensitivity visualization for DTI fiber tracking, *IEEE Trans. Visualization Comput. Graphics*, 15(6):1441–1448, 2009.
29. Sanyal, J., Zhang, S., Bhattacharya, G., Amburn, P., and Moorhead, R., A user study to compare four uncertainty visualization methods for 1D and 2D data sets, *IEEE Trans. Visualization Comput. Graphics*, 15(6):1209–1218, 2009.

30. Haller, G., Distinguished material surfaces and coherent structures in three-dimensional fluid flows, *Physica D*, 149(4):248–277, 2001.

31. Nese, J., Quantifying local predictability in phase space, *Physica D*, 35(1-2):237–250, 1989.

32. Sadlo, F. and Peikert, R., Efficient visualization of Lagrangian coherent structures by filtered AMR ridge extraction, *IEEE Trans. Visualization Comput. Graphics*, 13(6):1456–1463, 2007.

33. Sadlo, F. and Weiskopf, D., Time-dependent 2-D vector field topology: An approach inspired by Lagrangian coherent structures, *Comput. Graphics Forum*, 29(1):88–100, 2010.

34. Pobitzer, A., Peikert, R., Fuchs, R., Schindler, B., Kuhn, A., Theisel, H., Matkovic, K., and Hauser, H., The state of the art in topology-based visualization of unsteady flow, *Comput. Graphics Forum*, 30(6):1789–1811, 2011.

35. Hlawatsch, M., Sadlo, F., and Weiskopf, D., Hierarchical line integration, *IEEE Trans. Visualization Comput. Graphics*, 17(8):1148–1163, 2011.

36. Schneider, D., Reich, W., Scheuermann, G., and Fuhrmann, J., A variance based FTLE like method for uncertain vector fields, In *Topological Methods in Data Analysis and Visualization II (Proceedings of TopoInVis)*, pp. 255–268, 2012.

37. Fuchs, R., Peikert, R., Sadlo, F., Alsallakh, B., and Gröller, M. E., Delocalized unsteady vortex region detectors, In *Proceedings of VMV*, pp. 81–90, 2008.

38. Sadlo, F., Peikert, R., and Sick, M., Visualization tools for vorticity transport analysis in incompressible flow, *IEEE Trans. Visualization Comput. Graphics*, 12(5):949–956, 2006.

39. Shadden, S., Lekien, F., and Marsden, J., Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows, *Physica D*, 212(3-4):271–304, 2005.