Texture-Based Visualization of Uncertainty in Flow Fields

Ralf P. Botchen[1] Daniel Weiskopf[1,2] Thomas Ertl[1]

[1]University of Stuttgart* [2]Simon Fraser University

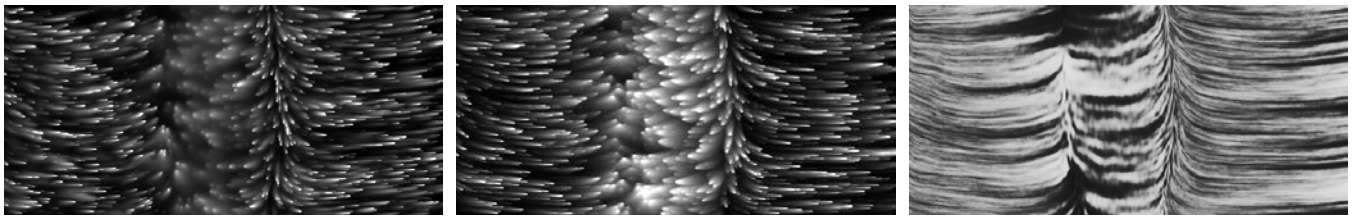

Figure 1: Three different advection schemes applied to a PIV measurement of a fluid flow data set. Left image: Gaussian error diffusion approach. Center image: Cross advection method. Right image: Semi-Lagrangian advection using multi-frequency noise.

ABSTRACT

In this paper, we present two novel texture-based techniques to visualize uncertainty in time-dependent 2D flow fields. Both methods use semi-Lagrangian texture advection to show flow direction by streaklines and convey uncertainty by blurring these streaklines. The first approach applies a cross advection perpendicular to the flow direction. The second method employs isotropic diffusion that can be implemented by Gaussian filtering. Both methods are derived from a generic filtering process that is incorporated into the traditional texture advection pipeline. Our visualization methods allow for a continuous change of the density of flow representation by adapting the density of particle injection. All methods can be mapped to efficient GPU implementations. Therefore, the user can interactively control all important characteristics of the system like particle density, error influence, or dye injection to create meaningful illustrations of the underlying uncertainty. Even though there are many sources of uncertainties, we focus on uncertainty that occurs during data acquisition. We demonstrate the usefulness of our methods for the example of real-world fluid flow data measured with the particle image velocimetry (PIV) technique. Furthermore, we compare these techniques with an adapted multi-frequency noise approach.

CR Categories: I.3.3 [Computer Graphics]: Picture / Image Generation I.3.6 [Computer Graphics]: Methodology and Techniques I.3.8 [Computer Graphics]: Applications

Keywords: Uncertainty visualization, unsteady flow visualization, texture advection, GPU programming.

1 INTRODUCTION

Exposing uncertainties of real-world flow data becomes a more and more important topic in the field of vector field visualization. Engineers and scientists are increasingly interested in which region and extent uncertainties occur in measured data. The absence of this information can lead to wrong conclusions and therefore affect the analysis process in a negative way. Several types of uncertainties or errors can appear on the way from data acquisition to the final step of visualization [11, 14, 24]. In this work, we focus on errors that arise during the measurement step by inaccuracies of the measuring device or technique. Although scientists and engineers are aware of the fact that almost every data set has intrinsic uncertainties, most visualization techniques neglect or completely ignore them. One reason for this is the difficulty to find a good uncertainty representation that can be combined with the visualization of the original data. By including the error measure into the process of visualization we need to add one more variate. For one-dimensional data, uncertainty can be displayed by a traditional plot that shows the average, minimum, and maximum value of a data point mapped to the second dimension. For multivariate data, this mapping becomes a challenging problem, since several dimensions have to be mapped to 2D screen space.

The goal of this paper is to provide new texture-based visualization schemes that do not only represent particle positions along streaklines but also bring out measuring errors and their influence on particles in the form of smeared-out visual structures perpendicular to the flow direction. We adopt traditional semi-Lagrangian texture advection [8, 19, 22], as briefly described in Section 4, and extend it by an error-dependent additional filter process that reveals uncertainty by essentially changing the spatial distribution of visual patterns perpendicular to the flow. This generic error visualization strategy is introduced in Section 5. We have developed two specific visualization methods that are derived from this generic strategy: First, the cross advection method, which uses a line-oriented filter perpendicular to the flow direction (Section 6); second, an artificial error-guided diffusion filter (Section 7). Our visualization methods lend themselves to efficient GPU (graphics processing units) implementations, which is demonstrated by providing details of the implementation along with performance measurements. Moreover, we compare our new methods with an adaptation of the multi-frequency noise method [9] and show results of uncertainty visualization applied to real-world fluid flow data from PIV measurements (see Section 9).

2 PREVIOUS WORK

Previous work on uncertainty visualization has focused on representing uncertainty in simulation or analytical data. For example, Lodha et al. [12] evaluate and compare the quality of surface interpolants and introduced geometric uncertainty as a measure of interpolation error. Furthermore, they propose UFLOW [11], a system to visualize uncertainties in streamlines of fluid flow with glyphs,

*{botchen|weiskopf|ertl}@vis.uni-stuttgart.de

IEEE Visualization 2005
October 23-28, Minneapolis, MN, USA
0-7803-9462-3/05/$20.00 ©2005 IEEE.

envelopes, or animation. Wittenbrink et al. [24] present several different glyphs like uncertainty glyphs and arrow glyphs to visualize wind and ocean currents. They map uncertainty direction and magnitude to various types of glyphs and glyph attributes. Pang et al. [14] give a classification of possibilities for visualizing uncertainties for a wide field of applications. Two techniques for uncertainty visualization in isosurface rendering are proposed by Rhodes et al. [16]. The first one changes one of the vertex color parameters hue, saturation, or brightness. The second technique maps an additional texture on top of the surface and varies the opacity of the texture according to the error measure. Finally, Brown [1] introduces a method of visual vibrations to indicate the amount of error.

Measurement of real-world data is a typical source of error or uncertainty. A widely used technique to acquire the velocity field of a fluid flow is based on particle image velocimetry (PIV) [6, 15]. Since the quality and field of application of PIV measurements has improved in recent years, the analysis of PIV data is becoming increasingly interesting. For example, the recent work by Ebling et al. [3] applies image-processing methods to analyze PIV measurements.

Research in the field of texture-based flow visualization has been strongly advanced lately, not only because of the rapidly increasing performance and functionality of GPUs. The availability of dense noise-based and sparse dye-based representations, as well as the possibility for the user to interact and manipulate all important parameters on-the-fly also play an important role. The state-of-the-art in texture-based flow visualization is surveyed by Laramee et al. [10]. Early work on texture-based methods comprises spot noise [18], line integral convolution (LIC) [2], and texture advection [13]. More recent 2D techniques rely on semi-Lagrangian texture advection: Image Based Flow Visualization (IBFV) by Van Wijk [19] and Lagrangian-Eulerian Advection (LEA) by Jobard et al. [8]. Since dense texture representations need a large number of computations, graphics hardware can improve the performance of 2D texture-based flow visualization [5, 7, 23]. An example for a sparse representation is the metaphor of dye advection [21, 22].

3 Particle Image Velocimetry

Real-world fluid flow can be measured by particle image velocimetry (PIV) [6, 15]. The basic principle of PIV is to inject particles into the flow and to measure the movement of these particles between two light pulses. Usually, a planar laser light sheet technique is used. In very short intervals the target area is illuminated twice by a double-pulsed laser and recorded onto the CCD array of a digital camera (see Figure 2). Since the CCD chip must be able to capture each light pulse in separate image frames, the resolution in time is bound to the image frequency of the camera. Afterwards, appropriate algorithms evaluate consecutive images and determine the displacement of particles in the flow. The most common way of measuring displacement is to divide the image plane into small interrogation areas (IA) and cross correlate the images from the two time exposures. With this method, even unsteady or non-periodic flow fields can be measured.

One possible cross correlation algorithm works as follows: Each image is divided into l small IAs with edge length K (in pixels); the IAs are then shifted and compared with other parts of the image. For each translation $\Delta\mathbf{x}$ of one IA A with K^2 pixels to another domain B of the same size, the cross correlation

$$C(\Delta\mathbf{x}) = \sum_{i=1}^{K}\sum_{j=1}^{K} B_{ij}\cdot A_{ij}(\Delta\mathbf{x}) \qquad (1)$$

can be computed. If the coefficient C is a maximum and/or minimum for a translation $\Delta\mathbf{x}$ (depending on the matrix function), then

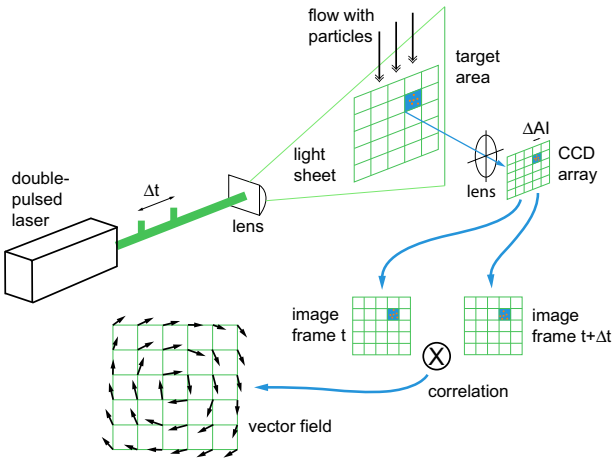

Figure 2: Configuration of a particle image velocimetry system.

the largest match between A and B and thus the shift of the particles has been found for IA A. Solving Eq. (1) for all IAs $\Delta\mathbf{x}l$ of the image results in a collection of displacement vectors that can be converted to velocity vectors by using the time difference between both evaluated images:

$$\mathbf{v}_l(\mathbf{x}_l) = \frac{\mathrm{d}\mathbf{x}_l}{\mathrm{d}t} \approx \frac{\Delta\mathbf{x}_l}{t_2 - t_1} \quad .$$

On the way from data acquisition to the stage of visualization, several different uncertainties or errors can be introduced to the data [11, 14]. Figure 3 shows the three stage pipeline that data has to

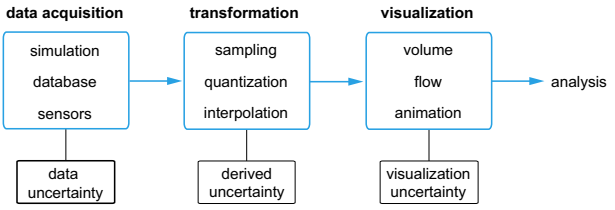

Figure 3: Possible sources for uncertainties (adopted from [14]).

go through until it can be analyzed. In the visualization step, algorithmic uncertainties can occur, e.g., by approximating factors for global illumination or by interpolating data values on slices in volume rendering. Transformation uncertainties are introduced by converting from one unit to another or by scaling, resampling, or quantization. In our work, we focus on uncertainties that appear in the first stage—during data acquisition. We think that this stage is most interesting for uncertainty visualization because the errors from data acquisition typically cannot be influenced or reduced by subsequent processes. In contrast, algorithmic or numerical errors from the other two stages of the pipeline can be neglected if algorithms of appropriate quality are applied. Nevertheless, error from any source could be used as input to, and thus shown by, our uncertainty visualization methods.

During data acquisition, errors can be introduced into PIV data sets from several sources [15]:

- Random error due to noise in the recorded images.
- Bias error arising from the process of computing the signal peak location to sub-pixel accuracy.
- Calibration inaccuracy of the CCD camera.

- Acceleration error caused by approximating the local Eulerian velocity from the Lagrangian motion of tracer particles.
- Gradient error resulting from rotation and deformation of the flow within an interrogation area leading to a loss of correlation.
- Dimension of the cross-correlated interrogation areas.
- Tracking error resulting from the inability of a particle to follow the flow without slip.

Some of these errors can be minimized by carefully choosing the experimental conditions, but others cannot be eliminated and thus the final data inherits uncertainties by the nature of the measuring system. Since engineers are fully aware of the existence of those uncertainties it can be of great advantage to have an interactive visual feedback during analysis.

In our system, the PIV method yields N measurements for the vector field at each spatial location and time step. The raw data measurements are denoted $\mathbf{v}_i$ with $i = 1 \ldots N$. The average vector $\mathbf{v}$ serves as basis for traditional vector field visualization. Then, the root mean square

$$r_{\text{rms}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} ||\mathbf{v}_i - \mathbf{v}||^2} \,,$$

can be used as the measure of uncertainty.

Our test data set was acquired in a laminar water channel with a 3D S-PIV method and a resolution of 81×45 in each of nine separated layers. Each layer was measured $N = 25$ times. Since our computation takes place in the 2D domain, we slice the 3D vector field into nine separated 2D layers.

4 Semi-Lagrangian Texture Advection

Texture advection is a well-established and versatile method for visualizing unsteady flow [8, 13, 19]. Semi-Lagrangian transport [8, 17], which is the basis for our implementation of texture advection, is briefly described in this section.

Particles or injected dye are represented on a regularly sampled grid or texture. This property field is denoted by $\rho(\mathbf{x})$. The points $\mathbf{x}$ are from the domain of the nD vector field, $\mathbb{R}^n$. For the Eulerian approach, particles lose their individuality and their position is implicitly given by the location of the corresponding texel in the property field. Particles are transported along streamlines for steady, or along pathlines for unsteady vector fields $\mathbf{v}(\mathbf{x},t)$, where t denotes time. The Lagrangian formulation of the underlying equation of motion,

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = \mathbf{v}(\mathbf{x}(t),t) \,,$$

can be integrated to compute the pathline of an advected massless particle,

$$\mathbf{x}(t_1) = \mathbf{x}(t_0) + \int_{t_0}^{t_1} \mathbf{v}(\mathbf{x}(t),t)\,\mathrm{d}t \,. \tag{2}$$

The evolution of the property field $\rho(\mathbf{x},t)$ is governed by

$$\frac{\partial \rho(\mathbf{x},t)}{\partial t} + \mathbf{v}(\mathbf{x},t) \cdot \nabla \rho(\mathbf{x},t) = 0 \,.$$

This partial differential equation can be solved by semi-Lagrangian transport [8, 17], which leads to a stable evolution even for large step sizes. A backward texture lookup is often employed:

$$\rho(\mathbf{x}(t_0),t_0) = \rho(\mathbf{x}(t_0-\Delta t),t_0-\Delta t) \,. \tag{3}$$

Starting from the current time step t_0, an integration backwards in time according to Eq. (2) provides the position at the previous time step, $\mathbf{x}(t_0-\Delta t)$. Texture-based methods often produce only short streamlines or streaklines and, therefore, first-order Euler integration typically provides sufficient accuracy:

$$\mathbf{x}(t_0-\Delta t) = \mathbf{x}(t_0) - \Delta t\,\mathbf{v}(\mathbf{x}(t_0),t_0) \,.$$

The property field is evaluated at this previous position to access the particle that is transported to the current position. Tensor-product linear interpolation (bilinear in 2D or trilinear in 3D) is applied to reconstruct the property field at locations different from grid points.

Texture advection can be used to visualize larger structures in the flow, like streamlines in steady flow or streaklines in time-varying flow. These structures can be generated by injecting smooth dye patterns or noise-based particles into the property field ρ after each advection step, following the IBFV approach [19]. Newly injected material is combined with the advected property field by means of alpha blending, which represents the discretized version of an exponential filter kernel [4] in the context of LIC [2]. We use texture advection as basis for uncertainty visualization because it offers important benefits. First, the injection scheme is flexible in allowing the user to gradually change the density of new particles from a few randomly injected particles up to a densely filled property field represented by white noise. Second, texture advection lends itself to a direct and efficient mapping to GPUs, which leads to an interactive visualization of large data sets.

5 Strategy for Uncertainty Visualization

The goal of this paper is to enrich texture advection by a visualization of flow uncertainty, without losing the benefits of texture advection, i.e., its flexibility and efficiency. Uncertainty visualization essentially requires to represent one additional single-valued attribute: a measure for uncertainty or error.

From an abstract point of view, uncertainty visualization can be structured into a three-stage process (see Figure 4). Raw data is

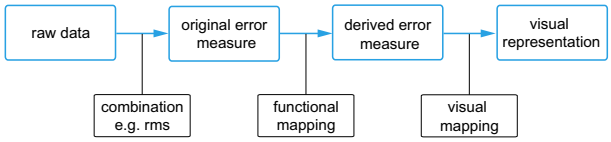

Figure 4: Uncertainty pipeline.

the basis to compute an error value. As detailed in Section 3, the error value for PIV data is typically based on the root mean square. However, any other error computation could be used for this part of the visualization process. In the second stage, this original error value is mapped to a derived error measure, e.g., by a linear or a non-linear function, in order to obtain values in a useful range. We denote this, possibly space-variant and time-dependent, uncertainty measure by $u(\mathbf{x},t)$. The third step comprises the actual mapping to visual primitives. This step is the main objective of uncertainty visualization.

Various means of a visual mapping of multi-variate data could be employed to represent the error attribute. A simple and well-known method is to map the derived error value to color and to overlay this color on top of the underlying flow visualization. However, this kind of visualization method only shows uncertainty at a respective point, i.e., it results in a localized representation. In contrast, uncertainty in a flow leads to an uncertainty of particle transport, which should also be represented by means of "uncertain" particle traces. It is reasonable to mark a particle that has been advected throughout an error-affected area and to emphasize this in the further process of the flow. Furthermore, an uncertainty-affected region can influence

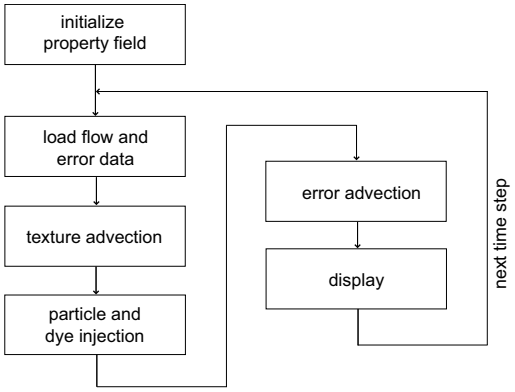

Figure 5: Flowchart of the algorithm.

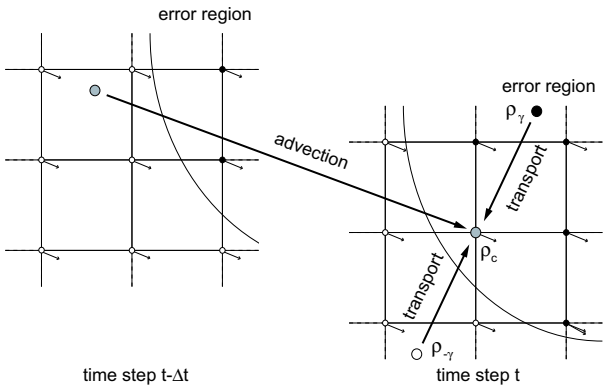

Figure 6: Semi-Lagrangian transport of a particle along flow direction into an error region from time step $t-\Delta t$ to time step t. The cross advection is applied in time step t.

its neighborhood and this should be taken into account in the visualization step. Therefore, our goal is to incorporate the uncertainty representation within the concept of texture-based particle transport. Another advantage of this approach is that the error attribute is encoded in the same perceptual channel as the original flow—in the form of a texture—and, thus, other perceptual channels (e.g., color) could be used to encode additional attributes.

We propose the following basic visualization process that combines uncertainty visualization with traditional texture advection. As laid out in Figure 5, the first steps of the visualization cycle ("load flow data", "texture advection", and "particle and dye injection") are identical to traditional texture advection. Uncertainty visualization is exclusively based on an additional error filtering stage, that is completely decoupled from the texture advection computation. Error filtering aims at manipulating the spatial frequency perpendicular to particle traces to show uncertainty $u(\mathbf{x},t)$, i.e., we do not only change the spatial frequency along the flow as in LIC [2] or basic texture advection [10].

Error filtering modifies the property field and, in its generic form, can be formulated as

$$\rho_{\text{filtered}}(\mathbf{x}) = \int_{V(\mathbf{x},\mathbf{v})} f(u,\tilde{\mathbf{x}},\mathbf{v})\rho(\mathbf{x}+\tilde{\mathbf{x}})\,\mathrm{d}^n\tilde{x}\,. \tag{4}$$

The compact nD filter domain $V(\mathbf{x},\mathbf{v})$ may be space-variant and flow-dependent, and so is the filter f. The integral is evaluated at a fixed time t. In this paper, we restrict ourselves to 2D visualization with $n=2$ and we use two special cases for Eq. (4), which are detailed in Sections 6 and 7. Different "flavors" of uncertainty visualization are achieved by choosing specific parameters for the filter kernel and integration domain.

6 Cross Advection

Cross advection is the first specific example for error filtering. Here, the filter domain is reduced to a line perpendicular to the current flow direction. The fundamental idea is to transport a particle laterally to the flow direction and thus smear out the streakline of the particle. This essentially leads to a 1D convolution similar to that of traditional texture advection. The only difference is that a symmetric filter (in both directions) is applied.

Discretization of the filter leads to

$$\rho_{\text{filtered}} = \sum_{i\in\{-1,0,1\}} f_i\rho_i\,, \tag{5}$$

where f_i is the discrete filter kernel and ρ_i are samples of the property field along the perpendicular line,

$$\rho_i = \rho(\mathbf{x} + i\mu\Delta t M_{\text{rot}}\mathbf{v}(\mathbf{x},t))\,. \tag{6}$$

Here, μ denotes the error-depending relative step size, Δt is the step size of traditional texture advection, and M_{rot} is a 2D matrix for a rotation by 90 degrees. The integration width and, thus, the length scale for smearing is controlled by the error value via μ. Equation (5) implements the analog of line integral convolution based on texture advection, with the following differences: First, a convolution perpendicular to the flow direction is performed; second, a symmetric convolution filter in both directions is applied. This filtering maintains a constant overall brightness if a normalized kernel is used (i.e., $\sum f_i = 1$). A typical choice is $f_i = 0.25, 0.5, 0.25$.

Figure 6 illustrates the two steps performed for the complete cross advection approach. In the first step, a particle (grey dot) is transported by traditional semi-Lagrangian advection along the flow direction into an uncertainty region. Black grid points lie inside the error region, white grid points outside. In the next step shown on the right side of Figure 6, we compute the intensity values of the current point (grey dot) and the two cross directional points (white and black dots) by combining the values according to Eq. (5).

The filtering process can be slightly modified by replacing the weighted sum in Eq. (5) by a maximum function according to

$$\rho_{\text{filtered}} = \max\{\rho_i | i = 1,2,3\}\,. \tag{7}$$

The max() function guarantees that (sparsely seeded) streaklines are not reduced to very small intensities in regions of large error, and therefore smeared-out streaklines are maintained. In general, the max() function does not provide a constant overall brightness but tends to increase the image brightness. Therefore, this variant is primarily designed for sparse representations with a few, clearly separated streaklines.

Cross advection can be considered an image-processing operation that can be directly mapped to GPU fragment operations. The update equations (5) or (7) work independently on 2D grid cells of the property field. By identifying this 2D grid with a texture, the update of cells (i.e., texels) is reduced to updating the underlying texture through a fragment program. In fact, ping-pong rendering is employed to update textures: Two copies of the property field are held in texture memory; one serves as render target, the other one serves as input texture. After each update, the role of the two textures is exchanged. This cross advection process is one example for an "error advection module" in Figure 5.

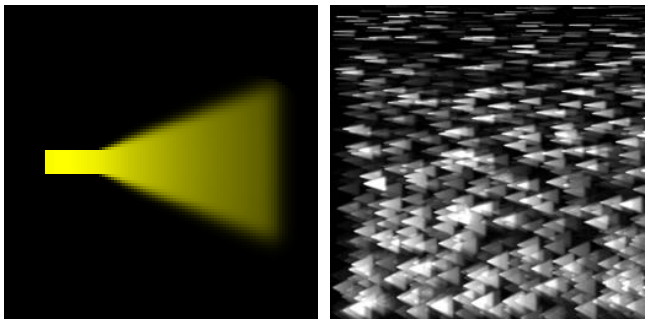

Figure 7: Left image: Cross advection with a single dye pattern in a laminar flow field and with piecewise constant uncertainty. Right image: Sparsely injected particles into the same flow with increasing uncertainty from top to bottom.

A HLSL fragment program of the basic transport mechanism for Eq. (7) is given in Figure 8. The mapping from the input uncertainty measure to the relative step size μ is implemented by a dependent-texture lookup. The space-variant input uncertainty is held in a 2D domain-filling texture. We need to perform three texture lookups in the property texture to evaluate Eq. (6). In the 2D case, the rotation matrix M_{rot} can be realized by a component swizzle followed by a simple multiplication. Finally, the texel with maximum intensity is written to the output. In a variant of this fragment program, the max() function is replaced by a weighted sum to implement Eq. (5).

The left image of Figure 7 shows an example of a single dye pattern injected into a uniform flow field. The underlying uncertainty begins a few steps away from the injection point and then stays constant. The exact shape of the streakline is visible in the error-free part of the flow that changes to a symmetric expansion of both sides. The right image illustrates a sparse injection of random particles, while the uncertainty increases from top to bottom. Since uncertainty magnitude effects the step size of cross transport, streaklines in regions with large errors (bottom) spread more than those in unaffected regions (top).

```
// lookup in all textures at current TexCoord position
float4 direction = tex2D( VectorField, TexCoord );
// result from previous traditional texture advection
float4 thistxl   = tex2D( AdvectedTex, TexCoord );
// uncertainty measure u(x,t)
float  error     = tex2D( ErrorField, TexCoord );
// cross advection step size: mu * Delta t
float  stepSize  = tex2D( ErrorStepSize, error );
float4 maxintens;

// perform cross advection in both directions
direction.yx     = direction.xy * stepSize.xx;
direction.x      *= -1.0f;                          // rotate -90
newpos.xy        = TexCoords - direction.xy;
float4 lefttxl   = tex2D( AdvectedTex, newpos.xy );
newpos.xy        = TexCoord + direction.xy;         // rotate 180
float4 righttxl  = tex2D( AdvectedTex, newpos.xy );

// find texel with maximum intensity
maxintens.x      = max( lefttxl.x, righttxl.x );
maxintens.x      = max( thistxl.x, maxintens.x );

// final output
Output.RGBA      = maxintens.xxxx;
return Output;
```

Figure 8: Main part of the HLSL fragment program for the cross advection approach.

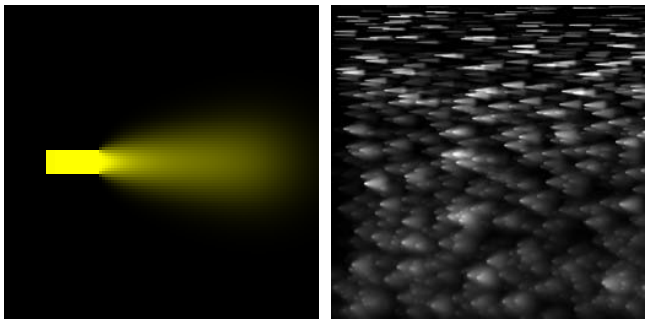

Figure 9: Left image: Gaussian error diffusion with a single dye pattern in a laminar flow with piecewise constant uncertainty. Right image: Sparsely injected particles into the same flow, with uncertainty magnitude increasing from top to bottom.

7 Error Diffusion

Error diffusion is our second technique based on the generic filtering process from Eq. (4). In contrast to cross advection, error diffusion applies an isotropic 2D filter kernel, independent of the direction of the flow. Filtering is space-variant: The uncertainty value determines the strength of smearing out. While cross advection blurs the streakline sideways to the flow direction, the diffusion filter affects texels not only in direction of the flow but in all directions. Since an error-affected data point exerts influence to all its adjacent points in real data measurements, this filtering process imitates natural diffusion. A typical filter kernel is a 2D Gaussian function, normalized in order to maintain a constant brightness. We do not recommend to employ a max() function here because this would lead to an overly fast increase in brightness—due to the larger support of the filter, there is a higher probability of collecting bright contributions than in the cross advection approach.

The diffusion computation is completely detached from the advection step and can be computed separately as shown in Figure 5. In this second step, we apply a discretized 2D filter kernel to the previously advected particles. For a GPU implementation, it is inefficient to use a large filter kernel because this would increase the computation costs dramatically. Therefore, we implemented only a discrete, separated 3×3 Gaussian filter. A larger filter kernel is achieved by successive application of this 3×3 filter, where the number of filtering steps is determined by the extent of uncertainty. A fine-grained control of filter strength through the uncertainty value is achieved by modifying the entries in the filter mask. The actual filter is constructed from a linear interpolation between an identity mapping and a full Gaussian kernel, where the interpolation weight is determined by the uncertainty value. Linear interpolation guarantees that the integral over the interpolated filter remains constant and normalized. In this way, we are able to obtain are range of filtering results, all the way from an identity mapping in regions with no uncertainty (which results in exact streaklines) up to a standard Gauss filter in regions with maximum uncertainty (which strongly blurs the streakline in all directions). For the GPU implementation, both the identity mapping and the Gaussian function are held in a floating-point texture. During runtime we perform a bilinear lookup in this texture and compute the linear interpolation to obtain a modified filter kernel.

The left image of Figure 9 illustrates dye injected into a laminar flow field with constant uncertainty, beginning a few steps away from the injection point. Well recognizable is the one-to-one mapping of the streakline in areas without error influence, which changes to a constant blurring in all directions in the error-affected region. The right image shows the Gaussian error diffusion approach applied to sparsely injected particles into the same flow with

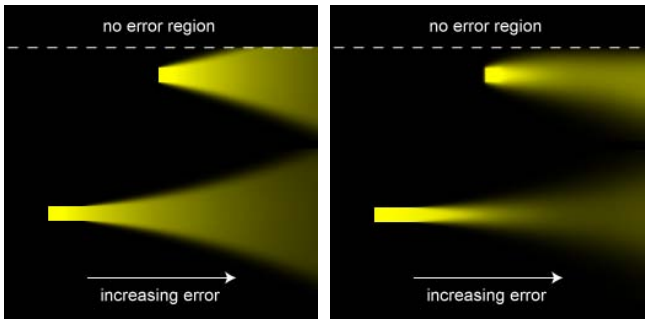

Figure 10: Comparison of cross advection and Gaussian error diffusion with two dye patterns injected into a flow with increasing uncertainty from left to right.

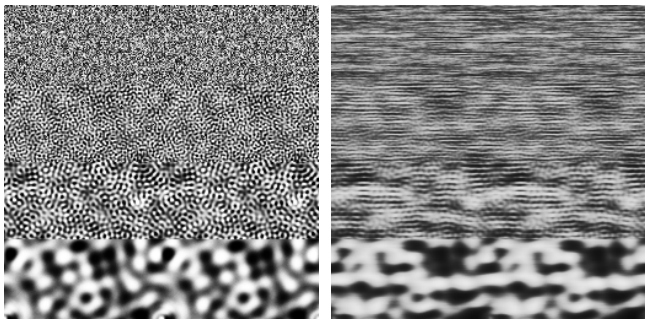

Figure 11: Left image: Four layers of multi-frequency noise. The frequency depends on the uncertainty value. Right image: Advection of multi-frequency noise in a laminar flow.

increasing uncertainty from top to bottom. This picture illustrates that the size and weight of the filter kernel depend on the uncertainty magnitude; hence streaklines in the lower part of the flow are heavily blurred while streaklines in the upper part are mapped to identity. Figure 10 shows both approaches with two injected dye patterns into a laminar flow field with increasing uncertainty from left to right. Considering the lower dye in the left image, one can see how uncertainty magnitude affects the step size of the cross advection and how the streakline widens due to an increasing step size. Depending on the 2D convolution kernel, the wavefront of a streakline computed with the error diffusion approach, runs faster than traditional texture advection or the cross advection approach.

8 Multi-Frequency Noise

We adopt the generic multi-frequency noise approach [9] as third technique for a dense vector field representation. Here, uncertainty is used to control the spatial frequency of noise injection. This technique can be directly incorporated into semi-Lagrangian advection by slightly modifying the injected noise. The original 2D noise is replaced by a 3D noise texture whose layers are filled with noise patterns of varying maximum spatial frequency. The first slice contains original white noise; all successive slices contain filtered versions of this white noise with decreasing maximum frequency. Filtering is based on the fast Fourier transform: First, the original white noise is transformed to frequency space, then a low pass filter is applied in frequency space and finally we perform the inverse Fourier transformation back to image space. Since low-pass filtered images lose contrast, we apply histogram equalization to match the contrast of the original image and sharpen the low-frequency structures. To access the different layers, the noise lookup is extended by a third dimension that is controlled by the error value. Figure 11 gives an example of four different noise patterns and their application to the visualization of a uniform flow field.

9 Discussion and Results

In Sections 6–8, we have discussed three different techniques to visualize uncertainties in vector fields. All methods are suitable for a dense representation and manipulate the spatial frequency according to the uncertainty value. Multi-frequency noise is a prefiltering method—all necessary spatial frequencies have to be precomputed and stored in a 3D texture. Our novel approaches apply post-processed filtering to the streaklines, and therefore, compared to the multi-frequency approach, the results are similar for dense noise injection but differ drastically for a sparse injection. An important advantage of both new methods is that they permit a continuous transition from a dense to a sparse representation, whereas the multi-frequency approach only works for a dense representation. A sparse noise injection or the injection of a smooth dye pattern allows the analyst to focus on single streaklines and their behavior according to the underlying flow field.

We illustrate all techniques on our test data set, which was measured in a laminar water channel with the PIV method. This data set contains one time step of water streaming through the test channel, producing vortices in the flow. The upper image of Figure 12 shows the fifth layer of the data set. Clear streaklines are generated with traditional semi-Lagrangian texture advection using alpha blending as exponential filter along flow direction. Streaklines only widen in divergent parts of the vector field and due to numerical diffusion. The second image shows the same flow visualized with the cross advection approach. In regions with marginal uncertainty, the streaklines remain clear but in uncertainty-affected regions they become blurred perpendicular to the flow by an extent that depends on the uncertainty value. The same applies to the third image, generated by Gaussian error diffusion. With both techniques, the user can still see structures of the flow in error regions, though the spatial frequency has been reduced. Furthermore, even the orientation of the flow is distinguishable due to the OLIC-like [20] structure of the streaklines.

For a sparse particle injection, the multi-frequency approach is not suitable. We have not included a picture in Figure 12 for comparison because low-pass filtering of sparse injected particles would lead to artifacts as known from heavy JPEG compressed images. Further, one would recognize single, isolated and large streaks, but their width would not change depending on uncertainty. Therefore, this approach is not intuitive enough for uncertainty visualization. Figure 13 directly compares our three uncertainty approaches by using dense particle injection represented by a white noise pattern. As anticipated, the spatial frequencies strongly decrease in regions of large error, irrespectively of the method being pre-filtered or post-filtered. All techniques produce similar results and eliminate structures of the streaklines in error regions. Our new approaches can be applied for dye patterns in the same way as for dense particle textures, which enables the engineer to interactively release dye in interesting regions to explore features of the flow. Accompanying video material can be found on the project web page[1].

In Figures 14 and 15, we use two synthetically generated data sets to demonstrate the behavior of the uncertainty advection approaches in regions of typical vector field features. Figure 14 shows three different topological structures such as source, sink, and two vortices, visualized with the cross advection method. Each feature point has an artificial error applied. The value of the error region is independent of the radius, but depends on the rotation angle. The

[1] http://www.vis.uni-stuttgart.de/texflowvis/uncertainties

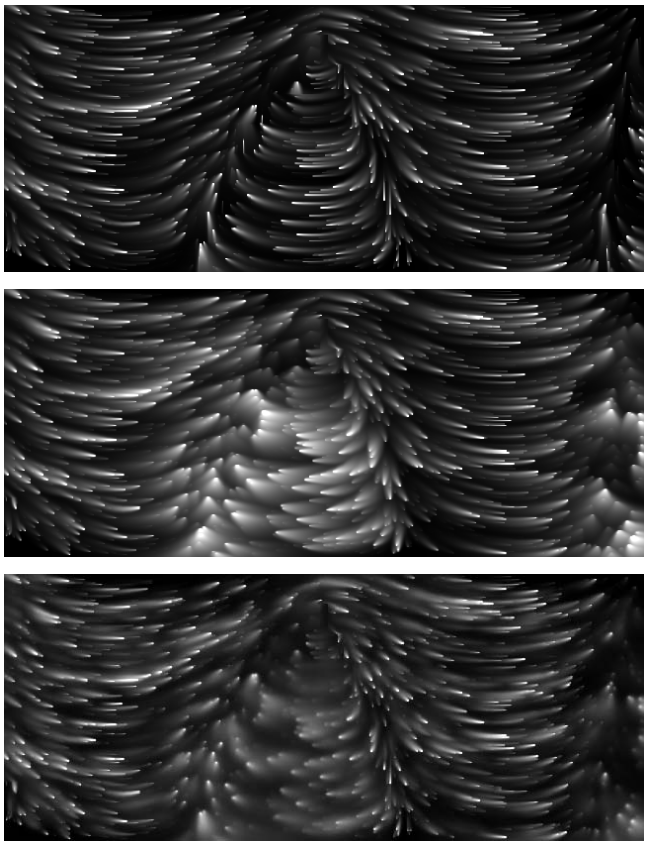

Figure 12: Visualizing the 5th layer of the PIV data set. The upper three images show semi-Lagrangian texture advection, cross advection, and Gaussian error diffusion with sparsely injected particles.

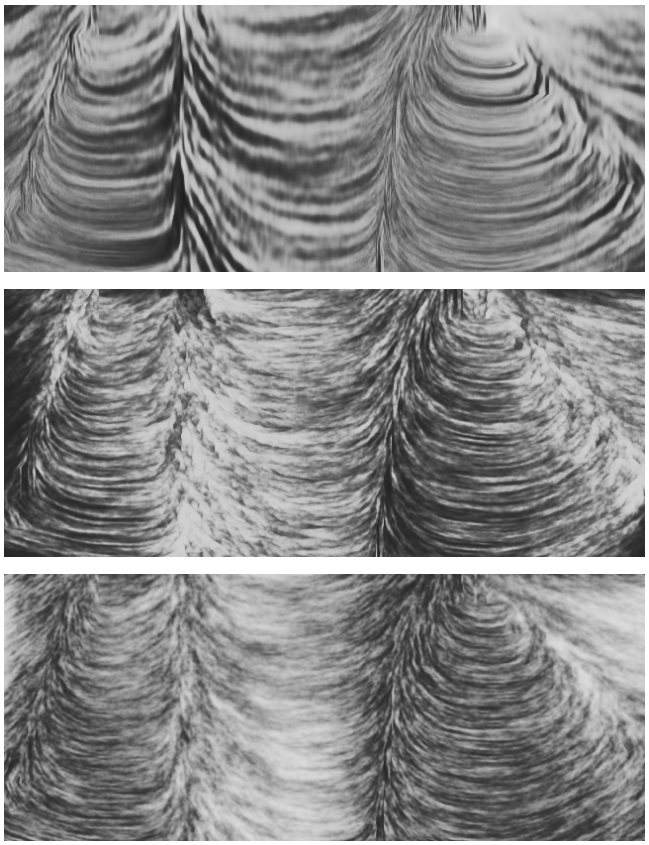

Figure 13: Illustration of all three approaches using a dense representation. Upper image: Multi-frequency noise. Middle image: Cross advection. Lower image: Gaussian error diffusion.

error decreases from 0 to 180 degrees and increases again up to 360 degrees. Even a source or a sink, where the structure of streaklines is manipulated due to divergence or convergence of the underlying vector field, the error-dependent widening of the streaklines is clearly visible. Figure 15 combines an elliptical flow with a sink. The radial error region is centered at the middle of the feature point. Visible is the increasing error on the right side of the flow. Though all streaklines meet in one point, there are no recognizable artifacts.

Our implementation of texture-based uncertainty visualization is based on C++, Direct3D 9.0, HLSL, and FX files. Respective performance numbers are documented in Table 1. Rendering speed depends linearly on the number of texels, as shown in the comparison of different viewport sizes. Cross advection and error diffusion have similar performance since both techniques need an additional render pass and more texture lookups. Due to single pass rendering and half as much lookups, the multi-frequency approach is twice as fast.

Table 1: Performance of all three GPU-based 2D advection techniques in frames per second measured on an NVIDIA GeForce 6800 GT graphics board.

Viewport size	256^2	512^2	1024^2
Multi-frequency noise	2073.0	558.0	125.0
Cross advection	1238.0	312.0	66.0
Gaussian error diffusion	1164.0	307.0	61.0

10 Conclusion and Future Work

We have presented a generic texture-based strategy to visualize uncertainty in time-dependent flow. As two specific examples for this strategy, we have proposed cross advection and error diffusion. According to underlying uncertainty of the data, both techniques change spatial frequency perpendicular to the flow direction. The main advantages of our techniques are their flexibility and generality. They can be directly combined with semi-Lagrangian advection by including one additional filtering step. Therefore, they can be applied to any density of texture representation ranging from dense noise-based up to sparse dye-based methods. Moreover, our approaches can be directly mapped to GPUs in order to achieve real-time visualization. In this way, the user can interactively explore the flow field.

For future work, an extension of uncertainty visualization to 3D flow will be a hallenging task. Moreover, the relationship between texture-based uncertainty visualization of flow and the visualization of symmetric second-order tensor fields could be investigated because the two eigenvector fields of a tensor data set could be related to the flow and uncertainty directions.

Acknowledgements

We would like to thank Simon "six to go" Stegmaier for his help and Matthias Lang and Ulrich Rist of the Institute for Aerodynamics and Gasdynamics at the University of Stuttgart for providing the real-world fluid flow data set.

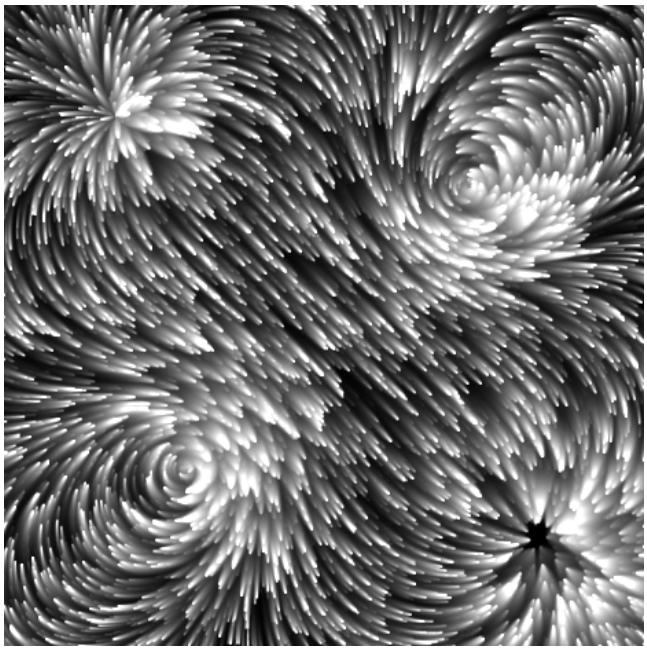

Figure 14: This synthetic data set combines different flow features like source, sink, and vortices to clarify the behavior of the cross advection approach in critical regions.

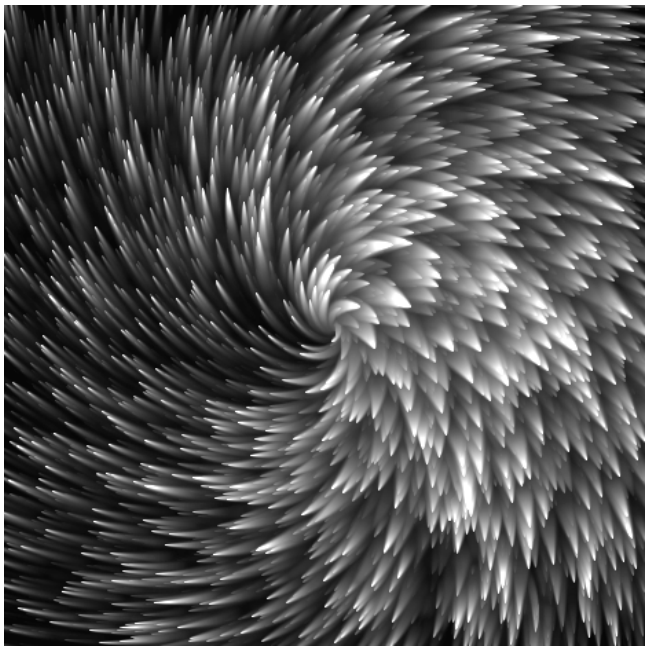

Figure 15: Cross advection approach for the example of a synthetic data set that combines a circular flow with a sink in the middle of the domain.

REFERENCES

[1] R. Brown. Animated visual vibrations as an uncertainty visualisation technique. In *Proc. GRAPHITE 2004*, pages 84–89, 2004.

[2] B. Cabral and L. C. Leedom. Imaging vector fields using line integral convolution. In *Proc. ACM SIGGRAPH*, pages 263–270, 1993.

[3] J. Ebling, G. Scheuermann, and B. G. van der Wall. Analysis and visualization of 3-C PIV images from HART II using image processing methods. In *Eurovis 2005 (Eurographics / IEEE VGTC Symposium on Visualization)*, pages 161–168, 2005.

[4] G. Erlebacher, B. Jobard, and D. Weiskopf. Flow textures: High-resolution flow visualization. In C. D. Hansen and C. R. Johnson, editors, *The Visualization Handbook*, pages 279–293. Elsevier, Amsterdam, 2005.

[5] W. Heidrich, R. Westermann, H.-P. Seidel, and T. Ertl. Applications of pixel textures in visualization and realistic image synthesis. In *ACM Symposium on Interactive 3D Graphics*, pages 127–134, 1999.

[6] K. D. Hinsch. Particle image velocimetry. In R. S. Sirohi, editor, *Speckle Metrology*, pages 235–324. Marcel Dekker, New York, 1993.

[7] B. Jobard, G. Erlebacher, and M. Y. Hussaini. Hardware-accelerated texture advection for unsteady flow visualization. In *IEEE Visualization 2000*, pages 155–162, 2000.

[8] B. Jobard, G. Erlebacher, and M. Y. Hussaini. Lagrangian-Eulerian advection of noise and dye textures for unsteady flow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):211–222, 2002.

[9] M. Kiu and D. C. Banks. Multi-frequency noise for LIC. In *IEEE Visualization '96*, pages 121–126, 1996.

[10] R. S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F. H. Post, and D. Weiskopf. The state of the art in flow visualization: Dense and texture-based techniques. *Computer Graphics Forum*, 23(2):143–161, 2004.

[11] S. K. Lodha, A. Pang, R. E. Sheehan, and C. M. Wittenbrink. UFLOW: Visualizing uncertainty in fluid flow. In *IEEE Visualization '96*, pages 249–254, 1996.

[12] S. K. Lodha, B. Sheehan, A. T. Pang, and C. M. Wittenbrink. Visualizing geometric uncertainty of surface interpolants. In *Graphics Interface '96*, pages 238–245, 1996.

[13] N. Max and B. Becker. Flow visualization using moving textures. In *Proc. ICASW/LaRC Symposium on Visualizing Time-Varying Data*, pages 77–87, 1995.

[14] A. T. Pang, C. M. Wittenbrink, and S. K. Lodha. Approaches to uncertainty visualization. *The Visual Computer*, 13(8):370–390, 1997.

[15] A. Prasad. Particle image velocimetry. *Current Science*, 79(1):101–110, 2000.

[16] P. J. Rhodes, R. S. Laramee, R. D. Bergeron, and T. M. Sparr. Uncertainty visualization methods in isosurface rendering. In *Eurographics 2003 Short Papers*, pages 83–88, 2003.

[17] J. Stam. Stable fluids. In *Proc. ACM SIGGRAPH*, pages 121–128, 1999.

[18] J. J. van Wijk. Spot noise – texture synthesis for data visualization. *Computer Graphics (Proc. ACM SIGGRAPH 91)*, 25:309–318, 1991.

[19] J. J. van Wijk. Image based flow visualization. *ACM Transactions on Graphics*, 21(3):745–754, 2002.

[20] R. Wegenkittl, E. Gröller, and W. Purgathofer. Animating flow fields: Rendering of oriented line integral convolution. In *Computer Animation '97*, pages 15–21, 1997.

[21] D. Weiskopf. Dye advection without the blur: A level-set approach for texture-based visualization of unsteady flow. *Computer Graphics Forum (Proc. Eurographics 2004)*, 23(3):479–488, 2004.

[22] D. Weiskopf, R. P. Botchen, and T. Ertl. Interactive visualization of divergence in unsteady flow by level-set dye advection. In *Proc. SimVis 2005*, pages 221–232, 2005.

[23] D. Weiskopf, M. Hopf, and T. Ertl. Hardware-accelerated visualization of time-varying 2D and 3D vector fields by texture advection via programmable per-pixel operations. In *Proc. VMV 2001*, pages 439–446, 2001.

[24] C. M. Wittenbrink, A. T. Pang, and S. K. Lodha. Glyphs for visualizing uncertainty in vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 2(3):266–279, 1996.