Noise-based volume rendering for the visualization of multivariate volumetric data

Rostislav Khlebnikov, Bernhard Kainz, Markus Steinberger, and Dieter Schmalstieg

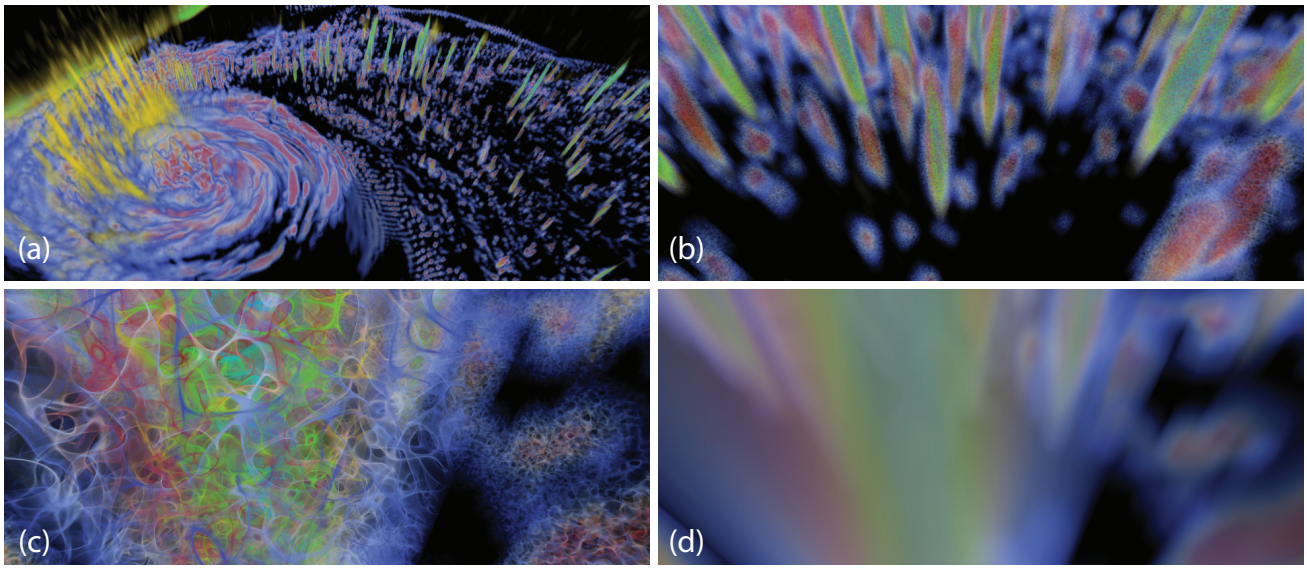

Fig. 1. Visualization of hurricane Isabel simulation data: amount of cloud water (blue-red) and upwind strength (yellow-green). At large distances our method converges to normal direct volume rendering to avoid aliasing (a). However, at smaller distances the user is able to interpret the exact data values for both variables simultaneously (b, c). For example, it is possible to see the high amount of cloud water (red) in the middle of high-velocity upstream (cyan). This is impossible with normal direct volume rendering (d).

Abstract— Analysis of multivariate data is of great importance in many scientific disciplines. However, visualization of 3D spatially-fixed multivariate volumetric data is a very challenging task. In this paper we present a method that allows simultaneous real-time visualization of multivariate data. We redistribute the opacity within a voxel to improve the readability of the color defined by a regular transfer function, and to maintain the see-through capabilities of volume rendering. We use predictable procedural noise – random-phase Gabor noise – to generate a high-frequency redistribution pattern and construct an opacity mapping function, which allows to partition the available space among the displayed data attributes. This mapping function is appropriately filtered to avoid aliasing, while maintaining transparent regions. We show the usefulness of our approach on various data sets and with different example applications. Furthermore, we evaluate our method by comparing it to other visualization techniques in a controlled user study. Overall, the results of our study indicate that users are much more accurate in determining exact data values with our novel 3D volume visualization method. Significantly lower error rates for reading data values and high subjective ranking of our method imply that it has a high chance of being adopted for the purpose of visualization of multivariate 3D data.

Index Terms—Volume rendering, multi-variate data visualization, multi-volume rendering, scientific visualization

1 Introduction

While the representation of volumetric data sets that contain only scalar data has been well researched since the 1980-ies [19], multivariate visualization of multiple dependent or independent data values per volume element is still unsolved and an active topic of research. Applications from medicine and engineering to meteorology and geology require the analysis of such multivariate data and are in need of a comprehensive multivariate data visualization methods. However, currently data is usually mapped to a scalar dimension and evaluated separately with established state-of-the-art volume rendering methods. Furthermore, some data dimensions form secondary information, which should not distract a user from the primary information. Such secondary information can be data uncertainty or secondary sensor information. State-of-the-art volume rendering methods are neither able to visually differentiate between multiple data values nor between different information qualities.

The most straightforward approach to the visualization of spatially fixed scalar data is color mapping. A variety of mapping – or in case of 3D volume rendering – transfer functions have been developed to allow for accurate perception of displayed data, *e.g.*, divergent color scales [21]. However, when multiple co-located continuous scalar values are to be displayed, the direct application of color mapping is not possible. This problem can be solved by blending the colors for different variables. However, Hagh-Shenas and colleagues have shown that

- *Rostislav Khlebnikov, Markus Steinberger, and Dieter Schmalstieg are with Graz University of Technology. E-mail: {khlebnikov |steinberger |schmalstieg} @icg.tugraz.at.*
- *Bernhard Kainz is with Imperial College London. E-mail: b.kainz@imperial.ac.uk.*

Manuscript received 31 March 2013; accepted 1 August 2013; posted online 13 October 2013; mailed on 4 October 2013.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

color blending is inferior to *color weaving*, where each data attribute receives it's own dedicated portion of screen space. They show that the users were significantly more accurate in inferring individual values when the data were inter-weaved using a high-frequency texture pattern than when the colors were blended [10].

However, in 3D volume rendering color blending is inherent to the algorithm itself even for single scalar dataset, as it uses transparency to reveal the internal structures. In this paper, we propose a novel visualization method that allows simultaneous display of multiple 3D spatially fixed data sets (see Fig. 1 for an example visualization achieved with our method). We present the following technical contributions in this work:

- We propose an algorithm for creating an opacity redistribution pattern using procedural noise (Section 3.1). We set the frequency of the noise in a way that no increase in sampling rate is required and the information in the data is not lost. An independent noise function is used for every data variable, which allows simultaneous display of several data variables at the same location. We also propose the way of constructing an opacity mapping function, which maintains the visibility of internal structures dictated by an arbitrary transfer function used for data classification (Section 3.2).

- We propose a filtering approach for the opacity mapping function that allows to avoid aliasing. It does not require increasing the sampling distance along the ray or supersampling in screen space. We also show that our method converges to conventional direct volume rendering when the noise cannot be represented on screen without aliasing (Section 3.3).

We have applied our method for the visualization of three-dimensional simulation data in combination with isosurface uncertainty information, and for multivariate climate data (Section 4). Furthermore, we have evaluated our method by comparing it to other visualization techniques in a controlled user study (Section 6). The results of our study show that users are significantly more accurate in determining exact data values with our visualization method and subjectively prefer it over other methods commonly used for multivariate visualization.

While our method is qualitative in nature, it may be used by experts to derive hypotheses about dependencies between variables directly from 3D renderings. To verify these hypotheses, quantitative data probing methods such as picking or slicing can be used. Overall our method can be seen as a better approach for *overview* and *zoom* tasks identified in the *Visual Information Seeking Mantra*[1]. From this point of view, transfer functions (1D, 2D, and others) are responsible for the *filtering* task and direct picking or slicing for *details-on-demand*.

2 Related Work

Visualization of multivariate 3D data The methods for displaying multivariate 3D data can be roughly subdivided into two distinct classes: fusion-based methods, which fuse the data at various stages of the rendering pipeline, and methods that employ different communication channels or rendering styles for each of the variables within the dataset. In this section we give an overview of these two groups of methods. For more in-depth analysis and categorization of methods for visualization of multivariate data we refer the reader to the recent work of Kehrer and Hauser [12].

For fusion-based methods, the main problem is the exact choice of fusing of multiple values at each location. One common approach is to analyze the data to extract features of the dataset based on multiple variables. For example, Kniss *et al.* approach this problem by specifying a multidimensional transfer functions during the classification stage of the volume rendering pipeline for generic multivariate data [14] and special case of data uncertainty [15]. However, these approaches do not directly display multiple variables at the same location simultaneously.

[1] Overview first, zoom and filter, then details-on-demand.

Another approach is to apply different transfer functions to make the values of different variables visually distinct. Cai and Sakas compare three fusion methods applied at different stages of the rendering pipeline, namely image level intensity fusion, accumulation level opacity fusion, and illumination model level parameters fusion [3]. Rößler *et al.* use straightforward alpha blending in a texture-based volume rendering system to mix the colors acquired from distinct transfer functions to display functional brain data [26]. Similar approaches were also applied in the context of raycasting [9, 11]. Akiba *et al.* additionally employ user-controlled weighted color blending to assign different importance levels to variables when displaying the turbulent combustion simulation data [1].

Other methods use different communication channels for the visualization of different variables within a multivariate dataset. For example, Crawfis and Max apply noise to communicate secondary information in volume data [5]. However, their method is only applicable for splatting-based volume visualization has not been evaluated. Similarly, Djurcilov and colleagues propose to use noisy textures to convey the information about the uncertainty of volumetric data [6]. Yu *et al.* combine volume and particle rendering to display two variables for in-situ visualization of large-scale combustion simulation [29].

Note that our method is not intended to replace the fusion-based methods and may be used in conjunction with them to improve the readability of exact data values and increase the number of variables that can be displayed simultaneously. For instance, at least two variables may be displayed with our method, as opposed to one volume-rendered variable, and an additional one – using glyphs or particles.

Color weaving Our method is similar in spirit to color-weaving approaches that proved to be effective for 2D data. Urness *et al.* propose to weave the color-coded scalar variables using a line integral convolution texture instead of blending the colors [27]. Hagh-Shenas *et al.* have shown that weaving-based approaches are significantly better than blending-based ones in showing distinct values for multiple variables. Additionally, Livingston *et al.* have shown that methods, which rely on subdivision of available screen space and visualization of each attribute with a distinct color scale, are most effective [20]. Technically most related to our work is the research by Khlebnikov *et al.* [13]. They have investigated the possibilities of using Gabor noise based textures with a similar opacity mapping function for visualizing two-dimensional data. The authors state the extension of their work to 3D as unsolved due to severe problems with the user's perception of 3D textures. In contrast to [13], we refrain from using adaptive noise frequencies in order to avoid information loss and use the 3D noise-based texture only as the opacity redistribution pattern, while encoding the values using color. In this way, we avoid problems with the perception of 3D texture properties, such as texture orientation and frequency, which are subject to projective distortion. With our approach, we are able to efficiently visualize two volumetric datasets simultaneously, which is supported by our user study.

3 Method

When developing our method we have set two important goals that we wanted to meet:

- The color-coded scalar values should be easily interpretable by the user.

- The see-through properties of volume-rendering techniques should be maintained.

Unfortunately, these goals contradict each other, because the lower the opacity that allows better see-through capabilities, the lower is the discriminability of the color of a particular voxel. To approach this problem, we notice that at high zoom levels, when the user explores the details of a particular feature within a dataset, every voxel usually occupies a significant portion of the screen. Therefore, we aim at redistributing the opacity of a voxel by making parts of it more opaque, thus improving the readability of the value within this voxel, while making the other parts more transparent and maintaining the see-through capabilities.

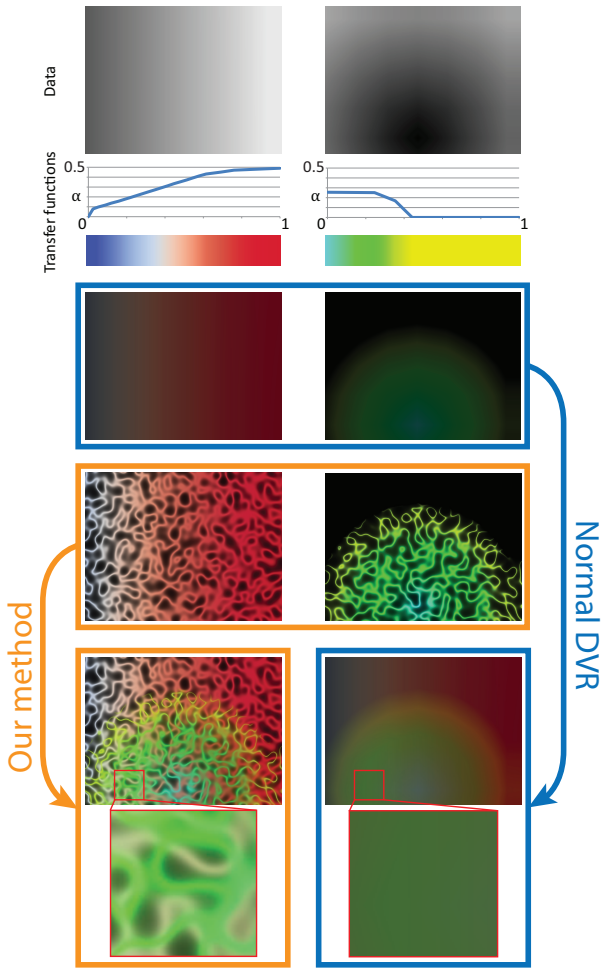

Fig. 2. An artificial example showing the comparison of visualization of two variables with normal direct volume rendering to our method. Our method redistributes the opacity using a noise pattern. Notice that with our method the individual color-coded data values (gray and light green, bottom-left) can be clearly seen, while for DVR distinguishing these colors is impossible due to color mixing (bottom-right). The size of the enlarged areas in the bottom row is one voxel.

To achieve these goals, we superimpose a high-frequency pattern which is used to redistribute the opacity within a voxel (see Fig. 2). This also allows us to avoid color blending when displaying multiple co-located data variables. In this section we will describe how to create such a pattern (Section 3.1), how to ensure that the average opacity is maintained after redistribution (Section 3.2), and how to avoid the potential aliasing that may arise from high frequencies caused by the opacity redistribution (Section 3.3).

3.1 Redistribution pattern

To avoid dependency on the viewing direction caused by regular redistribution patterns, such as shown in Fig, 3, we generate the opacity redistribution pattern using an isotropic noise function. More precisely, we use an isotropic version of random-phase Gabor noise [16], which offers precise control over its parameters and has a predictable intensity distribution as compared to other procedural noise functions [17].

Random-phase Gabor noise [16] is noise of a sparse convolution type. The value of the noise function $\mathcal{N}$ is computed as a convolution of impulses at random positions $\mathbf{x}_i$, which are defined by a Poisson impulse process, using a phase-augmented Gabor kernel:

$$g(\mathbf{x};a,\omega,\phi) = e^{-\pi a^2|\mathbf{x}|^2}\cos(2\pi\mathbf{x}\cdot\omega+\phi), \tag{1}$$

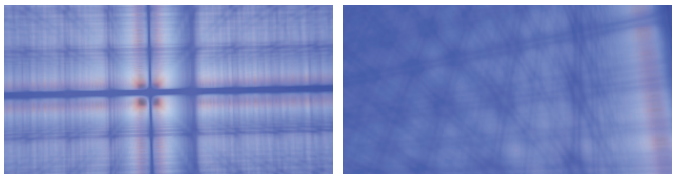

Fig. 3. An illustration of problems with regular redistribution pattern in 3D when viewing along one of the axes and in oblique direction.

$$\mathcal{N}(\mathbf{x};a,\omega) = \sum_i g(\mathbf{x}-\mathbf{x}_i;a,\omega,\phi_i), \tag{2}$$

where a is the bandwidth, ω is the frequency, and ϕ_i are random phases that are distributed according to an uniform distribution in the interval $[0,2\pi)$.

As the support of a Gabor kernel is not limited, the computation of noise values would require computing an infinite sum. However, the exponential part of the Gabor kernel falls off very quickly, so it can be truncated without introducing recognizable artifacts. To compute the noise value, Lagae *et al.* have introduced a virtual grid of size G, which equals the radius of the Gabor kernel truncated at a threshold level t [18]. To compute the noise value for a point within a cell, only impulses within this cell and its neighbors are considered in the summation in Eq. (2).

We set the frequency magnitude of the noise to $|\omega| = 1/v$, where v is the minimum extent of a single voxel in the dataset. On the one hand, this frequency is sufficiently high so that a single voxel will have a wide variety of noise values, which allows the mapped opacity to have both opaque and transparent parts according to the opacity mapping function (see Section 3.2). An illustration of the behavior of the noise function within a single voxel is shown in the bottom-left panel of Fig. 2. On the other hand, this frequency is low enough so that the sampling frequency during raycasting does not need to be increased to maintain visual quality (Section 3.3).

The orientation of the frequency vector ω is randomized to obtain isotropic noise [16]. For convenience, we denote the frequency magnitude as $f = |\omega|$. The size of the virtual grid G is consequently set to: $G = 1/f$ [28, p. 164]. The parameter a is then set to $a = \sqrt{\frac{-\ln(t)}{\pi}} \cdot f$, so that the size of a Gabor kernel truncated at the level t equals the grid size G [18].

3.2 Mapping noise values to opacity

Once the redistribution pattern is set up, it is necessary to define the mapping of its values to opacity, which are used in the final DVR step. To maintain the classification that is already defined by the classical DVR transfer function, we impose the following condition on the opacity mapping of the noise values:

$$\forall\alpha : \int_{-\infty}^{\infty} p(t)\cdot\mathscr{M}_\alpha(t)\mathrm{d}t = \alpha, \tag{3}$$

where $p(t)$ is the probability density function of the intensity distribution of the noise and $\mathscr{M}_\alpha(t)$ is the mapping function that corresponds to a particular opacity value α. In other words, we maintain the average opacity value that is defined by the regular transfer function.

The opacity mapping function $\mathscr{M}_\alpha$ may be chosen arbitrarily, as long as it satisfies condition (3). We use Gaussian mapping functions, which allow us to achieve high quality rendering without requiring much additional computations (see Sec. 3.3 for more details). The family of mapping functions is defined as:

$$\mathscr{M}_\alpha(x) = C_\alpha \cdot e^{-\frac{x^2}{2\sigma_\alpha^2}}, \tag{4}$$

where the parameters C_α and σ_α are chosen to satisfy the condition of Eq. (3). The intensity distribution $p(t)$ of three-dimensional random-

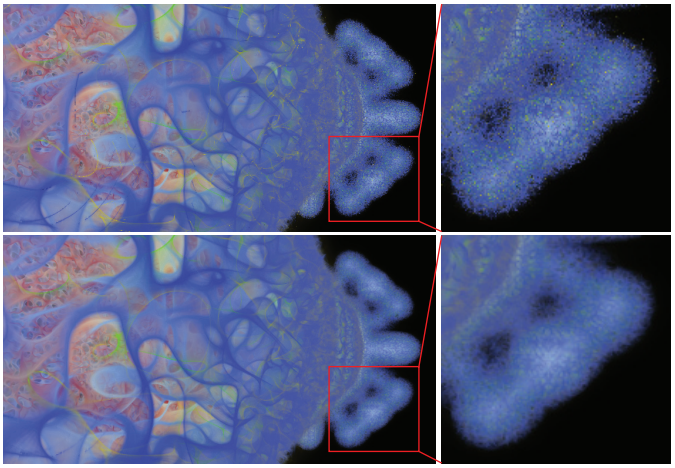

Fig. 4. Aliasing that occurs due to high frequencies in the noise modified by the opacity mapping function (top). Proper filtering eliminates this aliasing (bottom).

phase Gabor noise can be closely approximated by the normal distribution with zero mean [16]:

$$p(t) = \frac{1}{\sqrt{2\pi}\sigma_n} e^{-\frac{t^2}{2\sigma_n^2}}, \text{with} \tag{5}$$

$$\sigma_n^2 = \frac{\lambda}{2\left(\sqrt{2}a\right)^3}, \tag{6}$$

where λ is the Gabor noise impulse density. Substituting equations (4) and (5) to equation (3), we can derive that:

$$\sigma_\alpha = \frac{\sigma_n \alpha}{\sqrt{C_\alpha^2 - \alpha^2}}. \tag{7}$$

To ensure that $\sigma_\alpha \in \mathbb{R}_+$ and that $\mathscr{M}_\alpha(x) : \mathbb{R} \to [0,1]$, the parameter C_α may be chosen arbitrarily from the interval $(\alpha, 1]$. However, an additional constraint on C_α is necessary due to the requirements of the filtering step as described in Section 3.3.

3.3 Filtering

Applying an opacity mapping function directly to the noise pattern introduces additional high frequencies, which may lead to aliasing during rendering. This aliasing may appear both in the dataset space during the accumulation of color for a single pixel, and in screen space (see Fig. 4, top).

To avoid the dataset-space aliasing, we use the properties of the opacity mapping function. As the opacity mapping function is a Gaussian function, we use the possibility for analytic integration of Gaussian transfer functions to reduce the aliasing [14]. Note that this analytic integration applies only to the opacity mapping function and does not impose any constraints on the transfer function used for data classification.

Aliasing in screen space occurs when the screen-space sampling frequency, defined by the image resolution, is not high enough to represent the data projected on the image plane. One way to deal with this undersampling is to use multisample anti-aliasing (MSAA) techniques. The drawback of the MSAA approach is that the number of rays that need to be cast to achieve the final anti-aliased image is significantly increased. In order to avoid this performance penalty, we reduce the data frequency by modifying the opacity mapping function instead of increasing the screen-space sampling frequency.

The maximum frequency of a signal – the signal is the noise function in our case – is modified by our opacity mapping function as [2]:

$$f_{\mathscr{N}_\alpha} = \max_x \left|\mathscr{N}'(x)\right| \cdot f_{\mathscr{M}_\alpha}, \tag{8}$$

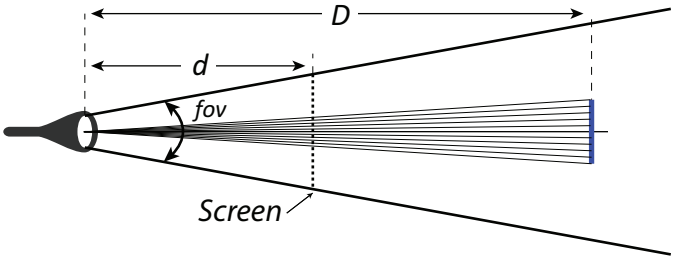

Fig. 5. Sampling of a unit-length segment (blue) according to the pixels on the screen using one ray per pixel. d is the distance to the near plane, fov is the camera's field of view, and D is the distance from the camera to the segment.

where $f_{\mathscr{N}_\alpha}$ is the maximum frequency of the resulting function, $f_{\mathscr{M}_\alpha}$ is the maximum frequency of the opacity mapping function and $\mathscr{N}'(x)$ is the derivative of the noise function.

The function with object-space frequency of $f_{\mathscr{N}_\alpha}$ is sampled on the screen, where every pixel is one sample. To ensure that the opacity-mapped noise function is sufficiently sampled, we have to ensure that the sampling frequency defined by the screen resolution is above the Nyquist rate for the opacity-mapped noise signal projected onto the screen. For convenience, we invert this problem and solve it in object space. Consider a unit-length segment at a distance D from the camera (see Fig. 5). The number of samples N_v along this segment can be computed as:

$$N_v = \frac{H}{2D\tan(fov/2)}, \tag{9}$$

where H is the screen resolution, fov is the camera's field of view, and D is the distance from the camera to the segment. N_v is exactly the frequency at which the projected signal at distance D is sampled by pixels. As we would like to avoid changing this sampling frequency for performance reasons, it is necessary to modify the signal, so that:

$$f_{\mathscr{N}_\alpha} < \frac{N_v}{2} = \frac{H}{4D\tan(fov/2)}. \tag{10}$$

The first multiplier contributing to the frequency of the opacity-mapped noise function (Eq. (8)) is the derivative of the noise. As the noise function is the sum of Gabor kernels (see Eq. (2)), the conservative estimate for its maximum derivative can be computed as the sum of maximum derivatives of the Gabor kernels, given the impulse density λ. As only the cell of the virtual grid the point belongs to and the neighboring cells are considered in the computation of the noise due to negligible influence of impulses located further away, the maximum noise derivative can be conservatively estimated as:

$$\max_x \left|\mathscr{N}'(x)\right| < 27 \cdot \lambda G^3 \cdot \max_x \left|g'(x)\right| \tag{11}$$

By analyzing the structure of the Gabor kernel, we can write:

$$\begin{aligned} &\max_x |g'(x)| = 2\pi e^{-\pi a x_0^2} \cdot (f + a x_0), \\ &\text{where} \quad x_0 = -f/2a + \sqrt{(f/2a)^2 + \frac{1}{2\pi a}} \end{aligned} \tag{12}$$

The second multiplier contributing to the frequency of the opacity-mapped noise function (Eq. (8)) is the maximum frequency of the opacity mapping function. The frequency spectrum of the Gaussian opacity mapping function $\mathscr{M}_\alpha$, defined in Eq. (4), is another Gaussian:

$$\mathscr{F}\left\{\mathscr{M}_\alpha(x)\right\}(f) = C_\alpha \cdot \sqrt{2\pi}\sigma_\alpha e^{-2\sigma_\alpha^2 \pi^2 f^2}, \tag{13}$$

where $\mathscr{F}\{\cdot\}$ is the Fourier transform. Obviously, it contains arbitrarily high frequencies with non-zero amplitude and thus the requirement given in Eq. (10) cannot be strictly satisfied. Therefore, we leave out of account the frequencies whose amplitude is less than a fraction τ of the maximum amplitude M in the frequency spectrum of the opacity

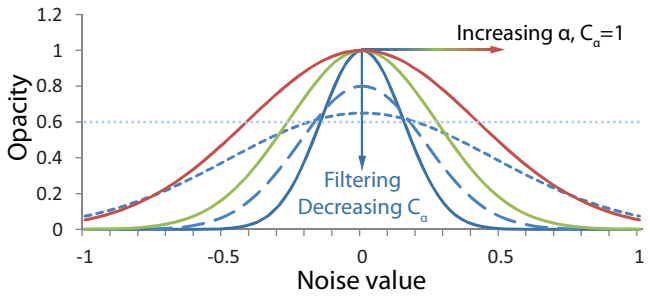

Fig. 6. Several representative opacity mapping functions with varying parameters. The blue graphs show how the opacity mapping function for $\alpha = 0.6$ behaves when additional filtering is required to avoid aliasing. Green and red graphs show the opacity mapping functions for $\alpha = 0.8$ and 0.9 respectively for $C_\alpha = 1$. For all graphs $\sigma_n = 0.2$.

mapping function $\mathscr{M}_\alpha$. This maximum amplitude M is reached at $f = 0$. As the frequency spectrum of the opacity mapping function is a zero-mean Gaussian function, we can define a cutoff frequency f_c, whose amplitude in the spectrum equals $\tau \cdot M$:

$$\begin{aligned} \mathscr{F}\{\mathscr{M}_\alpha(x)\}(f_c) = \tau \cdot \max_f \mathscr{F}\{\mathscr{M}_\alpha(x)\}(f) = \\ = \tau \cdot \mathscr{F}\{\mathscr{M}_\alpha(x)\}(0) = \tau \cdot M \end{aligned} \tag{14}$$

Consequently, all the frequencies with absolute value above f_c will have the amplitude below $\tau \cdot M$:

$$\forall f, |f| > |f_c| : \mathscr{F}\{\mathscr{M}_\alpha(x)\}(f) < \tau \cdot M. \tag{15}$$

Combining equations (13) and (14) and substituting σ_α from Eq. (7) we can derive that:

$$f_c = \frac{\sqrt{-ln(\tau)}}{\sqrt{2}\pi\sigma_\alpha} = \frac{\sqrt{-ln(\tau)\cdot(C_\alpha^2 - \alpha^2)}}{\sqrt{2}\pi\sigma_n\alpha} \tag{16}$$

Finally, we consider that the maximum frequency of the opacity mapping function is $f_{\mathscr{M}_\alpha} = f_c$.

From equations (8), (10), and (16) we get the following expression for C_α:

$$C_\alpha = \min\left(1, \alpha\sqrt{1 - \frac{\pi^2 H^2 \sigma_n^2}{8ln(\tau)\tan^2(fov/2)D^2\max_x|\mathscr{N}'(x)|^2}}\right), \tag{17}$$

which gives exactly one solution for the parameters of the noise opacity mapping function at each sample in conjunction with equation (7). Please see Fig. 6 for an illustration of how the opacity mapping function changes with changing parameters C_α and σ_α. In our implementation, we set $\tau = 0.15$, which is sufficient to avoid recognizable screen-space aliasing. See Fig. 4 for the comparison of filtered and unfiltered results.

Note that when the noise function under certain viewing conditions has the frequency that is too high to be represented on the screen without aliasing, the rendering results with our method will be equivalent to regular volume rendering. For example, when the distance to the sampling point increases, the term under the square root in equation (17) approaches 1, meaning that C_α approaches α, and σ_α approaches infinity. This means that $\forall x : \mathscr{M}_\alpha(x) = \alpha$ and the result of the DVR will be equivalent to conventional volume rendering (see Fig. 7, top row).

4 Applications

In this section, we show two applications of our method. The first application deals with climate analysis. For this example, all the displayed variables are equally important The second application shows how our method can be applied to visualize primary information and secondary information concurrently with low visual distraction. For this example, we use fuel injection simulation data as primary information and the level crossing probability for one of the isovalues as uncertainty information, hence as secondary information. The interaction with these visualizations as well as comparison to conventional DVR is shown in the accompanying video.

4.1 Climate data

Climate data obtained through simulation or measurement often contains multiple scalar attributes, such as air temperature, humidity, pressure and others. Understanding the relationship between these attributes is often crucial. In this example application, we apply our method to visualize the simulation of hurricane Isabel from the National Center for Atmospheric Research in the United States. We have selected two scalar attributes, namely the amount of cloud water and the strength of upwinds, which accompany cloud formation [7]. The resolution of the datasets is $500 \times 500 \times 100$ voxels. The datasets are mapped to color using two distinct color scales. The amount of cloud water in range $[0.0, 0.7]^{g}/_{m^3}$ is mapped to a red-to-blue diverging color scale [21] with constant opacity value of 0.03. The upwind velocity in range $[1.0, 4.7]^{m}/_{s}$ is mapped to a yellow-green-cyan color scale with opacity linearly rising from 0.01 for weak upwinds to 0.12 for strong ones. Figure 7 shows different zoom levels in comparison with conventional DVR. Our visualization allows reading the data values for both variables simultaneously, revealing the high amounts of cloud water deep within the strong upwind plume. With regular DVR, this interdependency of the data variables cannot be seen without additional effort.

4.2 Isosurface uncertainty

In some cases the information provided by an additional data attribute may carry contextual, less important information. To illustrate how our method is applied to such a use case, we visualize the results of a fuel-injection simulation simultaneously with the information about positional uncertainty of one of the isosurfaces of this data. In order to minimize the impact of displaying additional information along with the main data, we set the opacity of the positional uncertainty information to a significantly lower value. In this example, the opacity of the uncertainty information is five to ten times lower than that of the main data.

To keep this example comprehensible, we employ the level-crossing probability (LCP) as presented in [24]. Although it overestimates the probabilities according to Pfaffelmoser and colleagues [23], it is sufficient for our use case. Figure 8 shows the resulting images for this use case. Note that our method can be used with any more sophisticated probability computation method [23, 25].

5 Implementation and Performance

We implemented our method using NVIDIA CUDA [22]. The parallelization of the evaluation procedure is straightforward because there are no dependencies between the pixels. The CUDA kernel, which is executed for every pixel is outlined in Algorithm 1. The datasets are stored in GPU texture memory. We use the GPU texture units to allow hardware accelerated trilinear interpolation and a fast access via the texture cache. The implementation is based on traditional ray-casting to perform DVR. It should be noted, that our method can also be implemented for slice-based volume rendering, similarly to implementing preintegrated transfer functions [8].

The noise values were precomputed and stored in a 256^3 texture with $32\times$ oversampling to ensure high visual quality of the output (*i.e.*, 32 noise voxels per one dataset voxel). The texture wrapping mode was set to *mirror* mode to cover the whole dataset. To obtain high-quality noise, we chose the impulse density λ such that we have 16 impulses per cell of the virtual grid: $\lambda = {}^{16}/_{G^3}$. The noise values were converted to 2-byte fixed-point representation to reduce the memory requirements and to improve texture cache hit rate. The maximum noise derivative (Eq. (11)) is then computed using a Sobel operator.

While it is possible to reduce the size of the precomputed noise patch, it would lead to either decreased visual quality of the noise (in

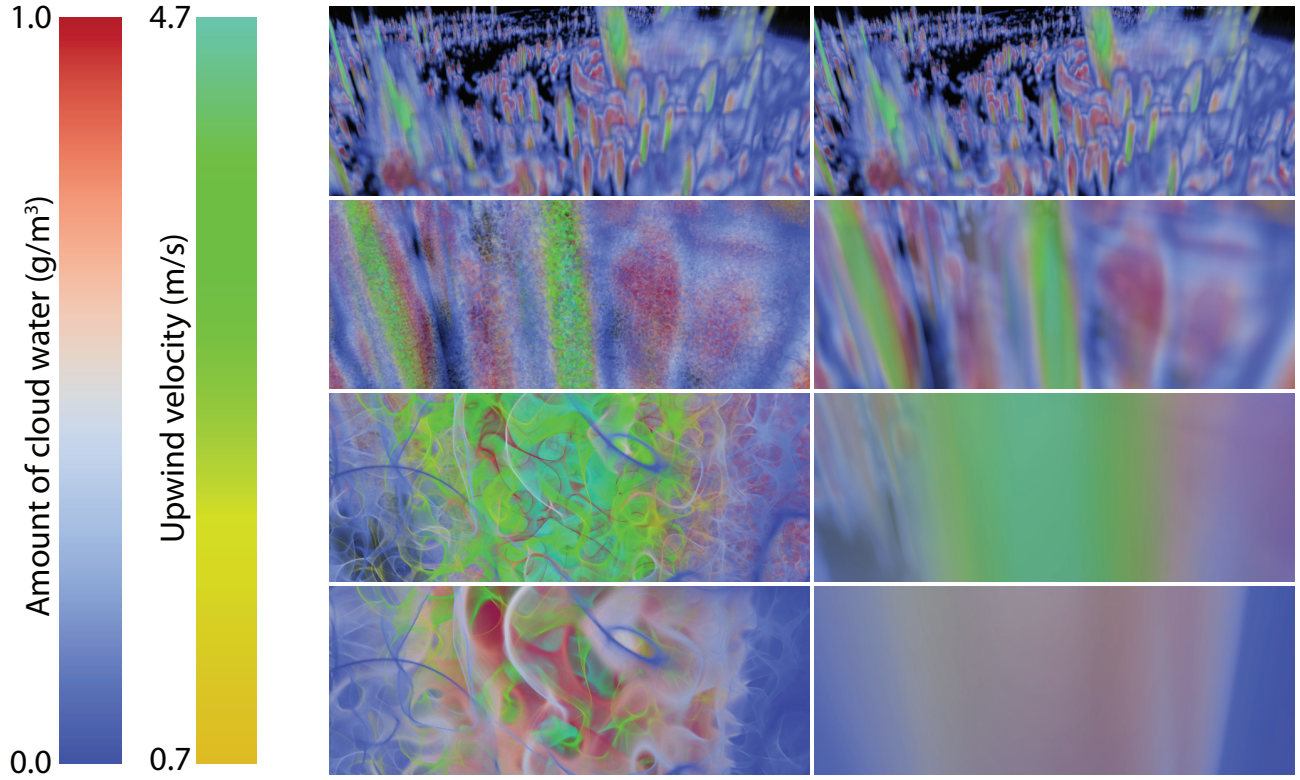

Fig. 7. Climate data visualization with two attributes: amount of cloud water (blue-red) and upwind strength (yellow-green-cyan). Our method is shown on the left, conventional direct volume rendering on the right. The color scales are shown on the far left. Use-case scenario: An analyst at first tries to get an overview of the data (top row). At this zoom level, our method and DVR generate identical images, because the noise is automatically filtered out to avoid aliasing. When analyzing data details, our method shows that also within the plumes with high upwind velocity a high amount of cloud water is present, while this information is completely lost in DVR. Furthermore, if opacity values are chosen non-optimally (bottom row), our method is still able to convey information about all attributes, while DVR hides all information from the analyst.

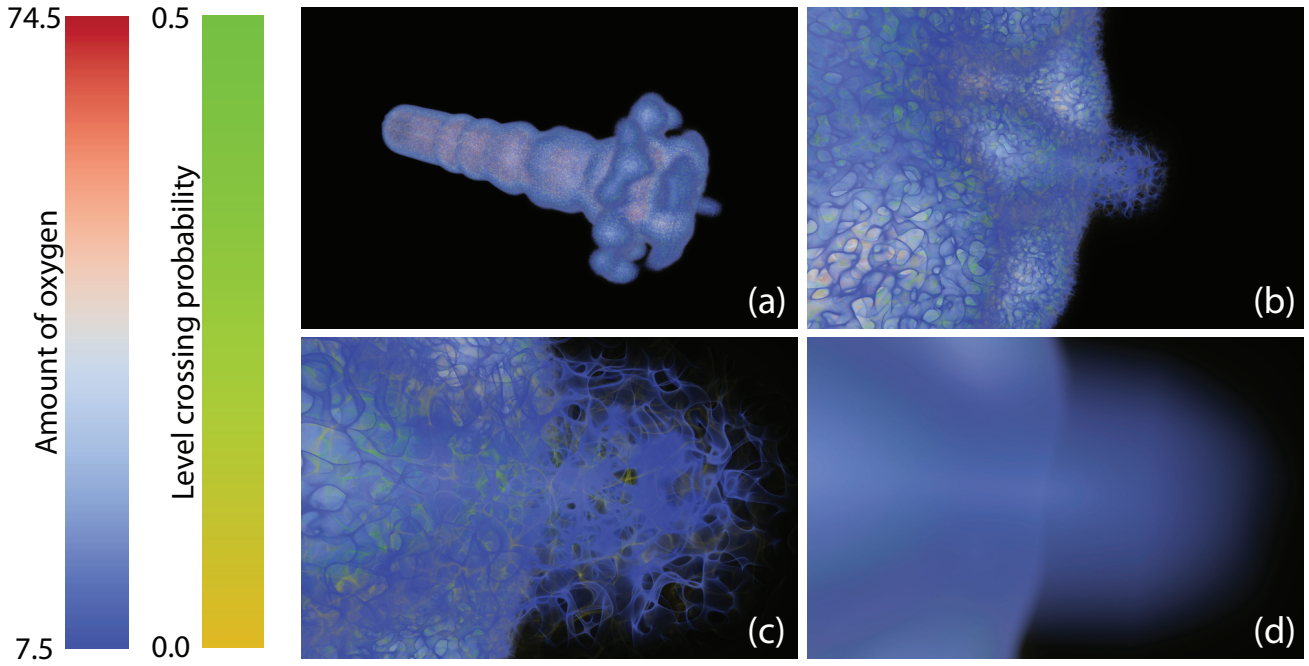

Fig. 8. Volume rendering of uncertainty data with two attributes: amount of oxygen (blue-red diverging, constant opacity of 0.1) and level-crossing probability for the isosurface with value 55 (yellow-green, linear ramp opacity in range $[0.01, 0.02]$). Results of our method are shown in panels (a), (b), and (c). Panel (d) shows the result for conventional DVR. Note how the low-opacity uncertainty information is completely hidden by conventional DVR (bottom) due to color intermixing. The color scales are shown on the left.

case the number of noise voxels per dataset voxel is decreased), or to a noticeable repetitive pattern on the edges between noise blocks. To avoid these problems and still reduce the memory footprint, the noise function can be computed directly as described by Lagae *et al.* [16]. However, for Gabor noise this becomes prohibitively slow, reducing the frame rate up to 10 times even when using only three impulses per cell.

Algorithm 1 Noise-based direct volume rendering CUDA kernel. The noise functions $\mathcal{N}_i$ are separate instances of random-phase Gabor noise with different seeds for random number generator.

```
SetupRay()
pixelColor ← 0
for all sampling points x do
    sampleColor ← 0
    for all datasets do
        value ← SampleDataset(dataset, x)
        color, α ← Classify(value)
        C_α, σ_α ← ComputeParameters(α, x)
        n ← N_i(x)
        α ← IntegrateGaussian(M_α(C_α, σ_α), n, prevN)
        sampleColor ← Blend(sampleColor, color, α)
        prevN ← n
    end for
    pixelColor ← Composite(sampleColor)
end for
```

We tested the overall performance on an Intel Core i7-870, 8GB RAM PC equipped with one nVidia GeForce GTX 680 graphics card. The noise-based volume rendering runs at frame rates between between 5 and 15 frames per second in a 1680×1000 viewport for two datasets with resolution of $500 \times 500 \times 100$, which is 0 to 20% slower than conventional volume rendering.

6 User Study

We conducted a controlled experiment to evaluate the effectiveness of our visualization method in comparison with other multivariate visualization methods. We recruited 20 participants (aged 22 to 35, 19 males, 1 female) from a local university with self-reported normal or corrected-to-normal vision. The participants were from the fields of computer graphics and augmented reality. The goal of the study was to test the influence of redistributing the opacity within a voxel on the ability of the users to estimate the values of two variables within this voxel.

We used the data from the Twentieth Century Reanalysis V2 data, provided by the NOAA/OAR/ESRL PSD, Boulder, Colorado, USA, from their Web site [4]. We use daily mean air temperature and specific humidity fields. Longitude, latitude and pressure level are used as dataset dimensions. The resolution of the datasets (after upsampling of the pressure level dimension) is $180 \times 91 \times 61$ voxels. The datasets are mapped to color using two distinct diverging color scales [21]. The mapped temperature ranges are $[-7,7]^\circ C$ and $[0.0025, 0.0045]$ for air temperature and specific humidity respectively. The opacity for these ranges was set to a constant value of 0.03. This data exhibits various useful patterns which we used to avoid favoring any particular technique (see Section 6.1 for more details).

We displayed the data using four techniques for multivariate data analysis, which we compare in our user study (see Figure 9):

- **Noise-based** - our algorithm.
- **Switching** - the datasets were visualized separately using conventional volume rendering. The users could switch between visualization of temperature and relative humidity by pressing the space bar.
- **Mixture** - the datasets were visualized with conventional volume rendering and colors obtained for two data values were alpha-blended at each sampling point.
- **Isosurfaces** - the datasets were visualized with combination of color-coded isosurfaces for the first variable (air temperature) and volume rendering for the second one (specific humidity).

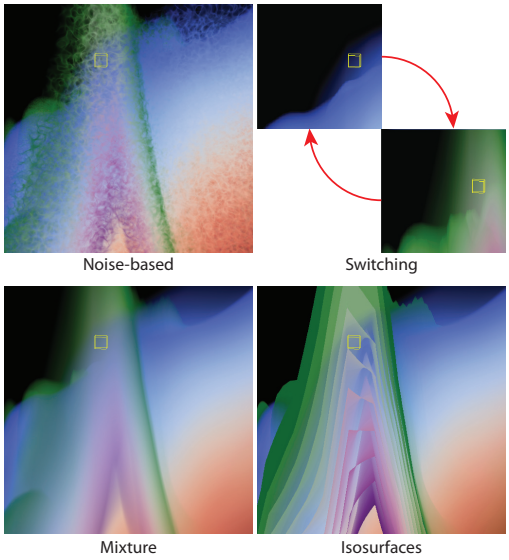

Fig. 9. Example views for the four visualization techniques compared in our study. For the "switching" method the users were able to switch between the visualization of single variables with a press of space bar. The yellow box denotes one of the locations where the participants were asked to read values from.

Fig. 10. The user interface which was used by participants to enter the values during the user study. This figure also shows the color maps that were used for temperature and specific humidity.

6.1 Task and Procedure

The participants were provided with a computer with a 22" monitor with 1680×1050 resolution at the viewing distance of approximately 60 cm. The study was conducted as a within-subjects experiment with four experimental conditions (technique) and one task per condition. The task was to determine the values of two variables within a voxel-sized 3D box. We have selected twelve boxes in different locations where both variables had non-zero opacity. To avoid favoring any particular method, we have chosen the locations of four types depending on the amount of the empty space surrounding the location. Two locations had empty space on roughly three quarters of the solid angle (*i.e.*, both variables exhibit a peak, Fig. 11a), three locations had empty space on a half of the solid angle (*i.e.* on the side of the dataset Fig. 11b), four locations with empty space on roughly a quarter of the solid angle (Fig. 11c), and three with no empty space in the surroundings (*i.e.* deep inside the data Fig. 11d). The boxes were rendered in wireframe with a distinct color (fully saturated yellow). The users were encouraged to interact with the visualization in a fixed-size 512×512 pixels window by using the mouse to rotate, pan and zoom. To enter the values, the participants used a separate dialog box with two sliders (see Fig. 10). Task completion time was measured automatically.

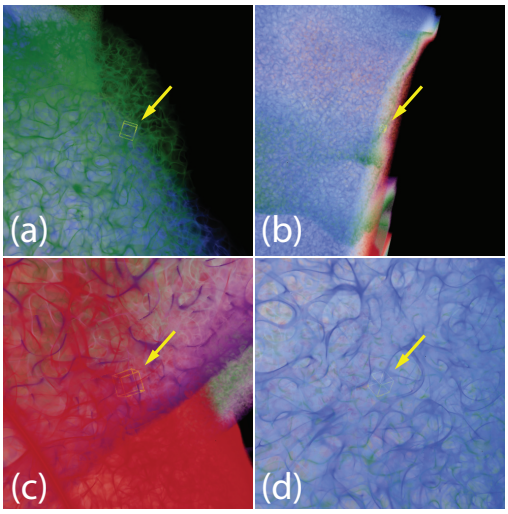

Fig. 11. Four types of locations used during the user study. (a) peak for both variables, (b) on the side of the dataset, (c) close to the empty space for both variables, (d) deep inside the datasets.

6.2 Results

The participants started each condition with a warm up phase where they familiarized themselves with the visualization method and the task. The answers and the time measured during the warm-up phase were not included into the final statistics. The participants were encouraged to rest between the conditions. To reduce the influence of learning effects, the sequence of the conditions was counter-balanced and the sequence of the locations was randomized.

Upon completion of the experiment, the users filled out a questionnaire, where they were asked to assess their subjective satisfaction by agreeing or disagreeing with the following statements on a five-point bipolar Likert scale:

- **Overview** - It was easy to gain a good overview of the data.
- **Understanding** - It was easy to understand the information displayed with this method.
- **Interpretation** - It was easy to interpret the data values within the boxes.
- **Confidence** - I am confident that my answers were correct when using this method.

Additionally, the participants were asked to rank all four methods by their own impression on how accurate and efficient the were in determining the values and by their overall personal preference.

Hypothesis: Our hypothesis was that the noise-based volume rendering performs better than the other techniques in both quantitative and qualitative assessment.

Error measures, timing and questionnaire answers were analyzed using Friedman non-parametric tests ($\alpha = .05$) for main effects and post-hoc comparisons using Wilcoxon Signed Rank tests with Bonferroni adjustments. A non-parametric test was used in all cases, because a Shapiro-Wilk test rejected normality in all cases. The evaluated measurements are summarized in the following:

- *Error measures* correspond to the absolute difference to the real value.
- *Timing* corresponds to the time from the data set appearing on screen until the participants report the determined values.
- *Subjective assessment* is evaluated by the questionnaire answers on a five-point bipolar Likert scale.
- *Preference* was assessed by assigning points to the techniques according to the order in which they were arranged by the participants.

We found a significant main effect for error ($\chi^2(3) = 38.04$, $p < 0.01$). Post-hoc comparison revealed that the error was lower for **Noise-based** than for any other technique and higher for **Mixture** than for all other techniques. No significant difference was found between the methods **Isosurfaces** and **Switching**.

We also found a significant main effect for timing ($\chi^2(3) = 31.98$, $p < 0.01$). Post-hoc comparison showed that the timing was higher for **Noise-based** than for any other method. No difference was found between **Mixture**, **Isosurfaces** or **Switching**.

There was a significant main effect for the qualitative questions *Overview* ($\chi^2(3) = 26.77$, $p < 0.01$), *Interpretation* ($\chi^2(3) = 38.13$, $p < 0.01$), *Understanding* ($\chi^2(3) = 28.00$, $p < 0.01$), and *Confidence* ($\chi^2(3) = 34.73$, $p < 0.01$). Post-hoc comparison revealed that **Mixture** was rated significantly lower than all other techniques for all questions asked. *Interpretation* and *Confidence* were rated significantly higher for the **Noise-based** method than for all other techniques. Although the mean score of the **Noise-based** method was also higher for *Overview* and *Understanding*, the difference to **Isosurfaces** and **Switching** was not significant. We found no significant difference between **Isosurfaces** and **Switching** for any question asked.

We also found a significant main effect for the rankings accuracy ($\chi^2(3) = 39.78$, $p < 0.01$), efficiency ($\chi^2(3) = 33.72$, $p < 0.01$), and preference ($\chi^2(3) = 39.18$, $p < 0.01$). Post-hoc comparison revealed that **Mixture** was again rated lower than all other techniques in all three categories. **Noise-based** was rated higher than **Switching** in accuracy and higher than other techniques in preference. The other pairwise comparisons did not reveal a significant difference.

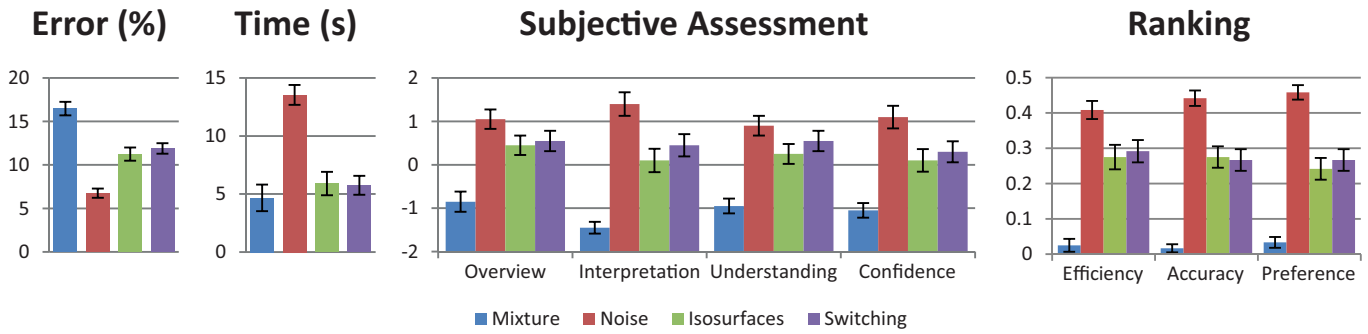

Fig. 12. The results of our user study. From left to right: average absolute difference to the real value, average task completion time, a qualitative evaluation of the methods, and overall preference of the users for choosing a certain visualization method. For error and time graphs – lower value is better, for subjective assessment and ranking – higher value is better.

7 Discussion

The study shows strong evidence that our method is superior compared to the other methods in terms of data reading accuracy. However, reading the values took participants longer with our method than with any other. This fact can be explained by two factors. Firstly, the participants have never used noise-based volume rendering before, unlike conventional volume rendering or isosurfaces. Therefore, even though there was a warm-up task before each condition, the unfamiliarity with the method might increase the time until the results were entered. Secondly, as the participants reported in the qualitative evaluation, they were much more confident that their answers were correct using the noise-based volume rendering. This confidence shows that the participants were continuously interacting with the visualization method to confirm their answer, while for other methods they have given a rough answer and then realized that further interaction will not be effective. This was also confirmed during an informal interview with each user study participant after all tasks were completed.

Additionally, the participants' answers for noise-based volume rendering has a lower error than conventional volume rendering even for univariate data. This conclusion can be made by comparing the error rates that were achieved using noise-based and switching methods. This is due to the fact that the users are able to judge the color in a thin nearly opaque band for noise-based DVR, while even for volume rendering of a single scalar field, the color of each pixel accumulates contribution from many sample points along the ray due to semi-transparency. Therefore, it may be beneficial to substitute simple volume rendering with noise-based visualization in cases where qualitative understanding of data values from the visualization is crucial for the user's task. Consequently, our method can be combined with methods requiring additional interaction such as slicing or picking in case the user needs the exact data values.

8 Expert evaluation and limitations

We have conducted an informal interview with a meteorology expert to find potential limitations of our method for its use in the practice. Overall, the expert was impressed by the basic idea and has noted that our method can be very useful for getting an impression of 3D multivariate data. However, he has also expressed a concern about the additional structure introduced by the remapping of the opacity using a noise pattern and about possible information loss by changing the opacity of the pattern. Firstly, our method may be confusing for the user without proper training. Secondly, additional structure can potentially distract the user from analyzing the actual data.

Additional measures could be taken to avoid or at least reduce the negative impact of introducing additional structure to the data. An example of such measures is the use of silhouettes to convey the structure of the data itself (see Fig. 13). Such an effect can be achieved by smoothly modulating the noise value towards zero in the vicinity of a set of isosurfaces. These silhouettes provide additional cues about the structure of the data and do not introduce aliasing because their opacity is filtered along with the opacity of the noise.

Another interesting approach to the issue of additional structure is the use of lighting-related effects, such as specular highlights and shadows that work directly on the original opacity and not on the values gathered from the noise opacity mapping function. Overall, this limitation needs to be considered when applying it to a particular visualization task or should be mitigated with additional algorithms and proper training. However, development of such algorithms and evaluation of their efficiency are out of scope of this paper. We will investigate them in future work.

9 Conclusions and Future Work

We have presented and evaluated a visualization method which is well suited for qualitative analysis of the data and which can be used to understand the dependencies between the data values directly from 3D visualizations. Our method shows significantly lower error for reading data values and higher subjective ranking than common multivariate visualization techniques. This implies that with attention to the limitations of our method, it has a high chance of being accepted for solving the tasks involving the analysis of multivariate 3D data.

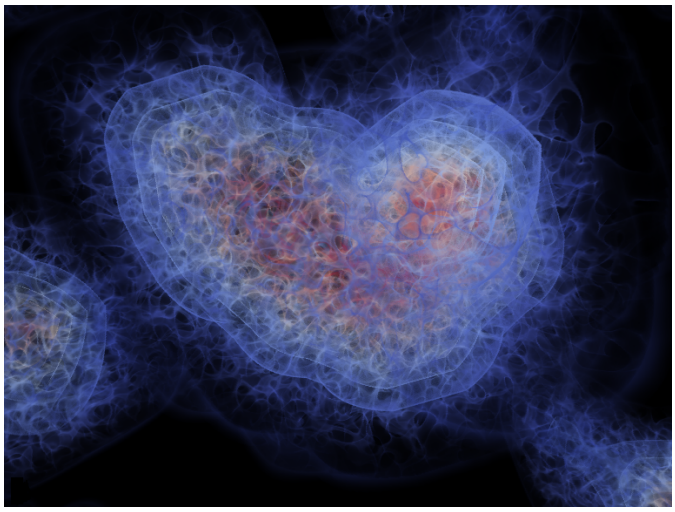

Fig. 13. An example of using silhouettes to provide additional information about the structure of the data that might be suppressed by introducing the noise pattern.

There are several directions for extending and improving our method that we will work on in the near future. Firstly, we will test how well our visualization method scales with the number of variables that are visualized simultaneously. Even though the extension of our method for more than two variables is straightforward, the perception of the concurrent visualization of three or more 3D data fields may prove to be too difficult for the users. Secondly, we will analyze how our method can be combined with other multivariate visualization approaches, such as 2D transfer functions. For example, a 2D transfer function can be used to specify the color and opacity mapping for two variables, which can be visualized with one noise pattern. It can then be combined with another noise pattern for the third variable or another pair of variables using one more 2D transfer function. And thirdly, we will explore the possibilities of using non-uniform and anisotropic noise. One option is to adapt the frequency of the noise according to data features in order to avoid over-filtering, in areas where the data is relatively homogeneous. Another option is to use the direction of the noise pattern to display additional variables or vector data similarly to [13]. However, we expect that additional measures will have to be taken for the 3D texture pattern to be well perceived by the users. It will be also interesting to experiment with time-varying noise and to evaluate its performance.

In addition, we will broaden the scope of applications for noise-based volume rendering. Currently, we are applying this method to medical tumor-ablation simulation data, to communicate a predicted cell-death area together with the uncertainty of the simulation. This may help a doctor to decide on a specific ablation protocol so that all tumor cells including all *possibly* tumorous areas are killed without harming the surrounding healthy tissue too much. Furthermore, we are experimenting with the manual evaluation of hyper-spectral imaging data. This imaging method allows to distinguish different chemical materials on-the-fly after a thorough manual adjustment. For this application area we can use our method to visualize the probabilities for other materials and therefore ease the manual adjustment process.

Acknowledgments

This research was funded by the Austrian Science Fund (FWF): P23329. Bernhard Kainz was supported by a Marie Curie Intra-European Fellowship within the 7th European Community Framework Programme (FP7-PEOPLE-2012-IEF F.A.U.S.T. 325661).

REFERENCES

[1] H. Akiba, K.-L. Ma, J. H. Chen, and E. R. Hawkes. Visualizing multivariate volume data from turbulent combustion simulations. *Computing in Science Engineering*, 9(2):76–83, Mar. 2007.

[2] S. Bergner, T. Moller, D. Weiskopf, and D. J. Muraki. A spectral analysis of function composition and its implications for sampling in direct volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12:1353–1360, 2006.

[3] W. Cai and G. Sakas. Data intermixing and multi-volume rendering. *Computer Graphics Forum*, 18(3):359–368, 1999.

[4] G. P. Compo, J. S. Whitaker, P. D. Sardeshmukh, N. Matsui, R. J. Allan, X. Yin, B. E. Gleason, R. S. Vose, G. Rutledge, P. Bessemoulin, S. Brnnimann, M. Brunet, R. I. Crouthamel, A. N. Grant, P. Y. Groisman, P. D. Jones, M. C. Kruk, A. C. Kruger, G. J. Marshall, M. Maugeri, H. Y. Mok, . Nordli, T. F. Ross, R. M. Trigo, X. L. Wang, S. D. Woodruff, and S. J. Worley. The twentieth century reanalysis project. *Quarterly Journal of the Royal Meteorological Society*, 137(654):1–28, 2011.

[5] R. Crawfis and N. Max. Multivariate volume rendering. Technical Report UCRL-JC-123623, LLNL, 1996.

[6] S. Djurcilov, K. Kim, P. Lermusiaux, and A. Pang. Visualizing scalar volumetric data with uncertainty. *Computers and Graphics*, 26:239–248, 2002.

[7] H. Doleisch, P. Muigg, and H. Hauser. Interactive visual analysis of hurricane Isabel with SimVis. Technical Report TR-VRVis-2004-058, VRVis Research Center, Vienna, Austria, 2004.

[8] K. Engel, M. Kraus, and T. Ertl. High-quality pre-integrated volume rendering using hardware-accelerated pixel shading. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware (HWWS '01)*, pages 9–16. ACM, 2001.

[9] S. Grimm, S. Bruckner, A. Kanitsar, and M. E. Gröller. Flexible direct multi-volume rendering in interactive scenes. In *Vision, Modeling, and Visualization (VMV)*, pages 386–379, Oct. 2004.

[10] H. Hagh-Shenas, S. Kim, V. Interrante, and C. Healey. Weaving versus blending: a quantitative assessment of the information carrying capacities of two alternative methods for conveying multivariate data with color. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1270 –1277, Nov.-Dec. 2007.

[11] B. Kainz, M. Grabner, A. Bornik, S. Hauswiesner, J. Muehl, and D. Schmalstieg. Ray casting of multiple volumetric datasets with polyhedral boundaries on manycore GPUs. *ACM Trans. Graph.*, 28:152:1–152:9, 2009.

[12] J. Kehrer and H. Hauser. Visualization and visual analysis of multifaceted scientific data: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):495–513, 2013.

[13] R. Khlebnikov, B. Kainz, M. Steinberger, M. Streit, and D. Schmalstieg. Procedural Texture Synthesis for Zoom-Independent Visualization of Multivariate Data. *Computer Graphics Forum*, 31(3):1355–1364, 2012.

[14] J. Kniss, S. Premoze, M. Ikits, A. Lefohn, C. Hansen, and E. Praun. Gaussian transfer functions for multi-field volume visualization. In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, VIS '03, pages 497–504, Washington, DC, USA, 2003. IEEE Computer Society.

[15] J. Kniss, R. Van Uitert, A. Stephens, G.-S. Li, T. Tasdizen, and C. Hansen. Statistically quantitative volume visualization. In *Proceedings of the 16th IEEE Visualization 2005 (VIS'05)*, VIS '05, Washington, DC, USA, 2005. IEEE Computer Society.

[16] A. Lagae and G. Drettakis. Filtering solid gabor noise. In *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, pages 51:1–51:6, New York, NY, USA, 2011. ACM.

[17] A. Lagae, S. Lefebvre, R. Cook, T. DeRose, G. Drettakis, D. S. Ebert, J. P. Lewis, K. Perlin, and M. Zwicker. State of the Art in Procedural Noise Functions. In H. Hauser and E. Reinhard, editors, *EG 2010 - State of the Art Reports*, pages 1–19. Eurographics, Eurographics Association, May 2010.

[18] A. Lagae, S. Lefebvre, G. Drettakis, and P. Dutré. Procedural noise using sparse Gabor convolution. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2009)*, 28(3):54:1–54:10, 2009.

[19] M. Levoy. Display of surfaces from volume data. *IEEE Comput. Graph. Appl.*, 8:29–37, May 1988.

[20] M. Livingston, J. Decker, and Z. Ai. Evaluation of multivariate visualization on a multivariate task. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12):2114 –2121, dec. 2012.

[21] K. Moreland. Diverging color maps for scientific visualization. In *Proceedings of the 5th International Symposium on Advances in Visual Computing: Part II*, ISVC '09, pages 92–103, Berlin, Heidelberg, 2009. Springer-Verlag.

[22] NVIDIA. *NVIDIA CUDA Programming Guide 4.0*. NVIDIA Corporation, 2011.

[23] T. Pfaffelmoser, M. Reitinger, and R. Westermann. Visualizing the positional and geometrical variability of isosurfaces in uncertain scalar fields. *Computer Graphics Forum*, 30(3):951–960, 2011.

[24] K. Pöthkow and H. C. Hege. Positional Uncertainty of Isocontours: Condition Analysis and Probabilistic Measures. *IEEE Transactions on Visualization and Computer Graphics*, 17(10):1393–1406, 2011.

[25] K. Pöthkow, B. Weber, and H.-C. Hege. Probabilistic marching cubes. *Computer Graphics Forum*, 30(3):931 – 940, 2011.

[26] F. Rößler, E. Tejada, T. Fangmeier, T. Ertl, and M. Knauff. GPU-based multi-volume rendering for the visualization of functional brain images. In *SimVis*, pages 305–318, 2006.

[27] T. Urness, V. Interrante, I. Marusic, E. Longmire, and B. Ganapathisubramani. Effectively visualizing multi-valued flow data using color and texture. In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, VIS '03, pages 115–121, Washington, DC, USA, 2003. IEEE Computer Society.

[28] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[29] H. Yu, C. Wang, R. Grout, J. Chen, and K.-L. Ma. In situ visualization for large-scale combustion simulations. *Computer Graphics and Applications, IEEE*, 30(3):45–57, 2010.