Vision, Modeling, and Visualization (2012)
M. Goesele, T. Grosch, B. Preim, H. Theisel, and K. Toennies (Eds.)

Spectral Analysis of Higher-Order and BFECC Texture Advection

R. Netzel, M. Ament, M. Burch, and D. Weiskopf

Visualization Research Center (VISUS), University of Stuttgart, Germany

Abstract

We present a spectral analysis of higher-order texture advection in combination with Back and Forth Error Compensation and Correction (BFECC). Semi-Lagrangian texture advection techniques exhibit high numerical diffusion, which acts as a low-pass filter and tends to smooth out high frequencies. In the spatial domain, numerical diffusion leads to a loss of details and causes a blurred image. To reduce this effect, higher-order interpolation methods or BFECC can be employed separately. In this paper, we combine both approaches and analyze the quality of different compositions of higher-order interpolation schemes with and without BFECC. We employ radial power spectrum diagrams for different advection times and input textures to evaluate the conservation of the spectrum up to fifth-order polynomials. Our evaluation shows that third-order backward advection delivers a good compromise between quality and computational costs.

1. Introduction

Nowadays advection methods are widely used for flow visualization in different domains, e.g., combustion engine design, aircraft engineering, or earth sciences. In particular, advection plays an important role in the popular flow visualization approach based on semi-Lagrangian advection. Regardless of the specific method used, the aim is to numerically solve the advection equation so that important features, such as edges or the spectrum of the transported image, are conserved. However, due to discretization and interpolation, error is introduced while solving the equation. This can lead to numerical diffusion and consequently to a loss of the feature properties. Especially, iterative methods that are efficient for time-dependent flow visualization implicitly implement a low-pass filter.

Our objective is to better understand the quality of texture advection for flow visualization. In particular, we assess backward and BFECC strategies as well as their combination with interpolation of higher-order. To this end, we propose an evaluation method that uses radial power spectrum diagrams to compare different time steps of the advection processes. We also evaluate a new method that combines higher-order interpolation with the BFECC approach. Furthermore, we investigate the influence of space discretization by testing different spatial resolutions of the texture.

2. Related Work

A simple method to obtain an impression about the trend of a flow is to put glyphs, like arrows, into a given image that show the direction of the flow at their respective positions. The problem here is the size of the glyphs. The bigger they are the better they can be recognized, but the level of detail of the flow that can be illustrated decreases. An alternative approach generates a texture that describes the flow with the needed level of detail. For this purpose, Line Integral Convolution (LIC) [CL93] was introduced. This method takes a noise image and filters it along the stream lines of an underlying vector field. Another technique for texture generation is Spot Noise [vW91]. Thereby white spots are randomly placed and smeared according to a vector field. Similarly, flows can also be visualized by a source that emits dye, that is then transported by the flow [LTH08, Wei04]. Furthermore, advection methods can be used to imitate high resolution fluids [Ney03].

Texture advection methods adopt one of the following computational views. In the Lagrangian view, the computation area is densely covered with particles that move according to a vector field [WSEE05]. In contrast, the Eulerian approach adopts a view of stationary locations at which the change of properties is observed. Typically, those properties are stored in a static grid, often even a uniform grid

DOI: 10.2312/PE/VMV/VMV12/087-094

that covers the domain. Finally, the semi-Lagrangian view combines the Lagrangian and Eulerian approaches: properties are stored on a static grid as in the Eulerian approach, whereas transport is computed by moving positions on the grid in a Lagrangian manner [vW02, JEH02, WBE05]. This leads to a transport from time step to time step.

Furthermore, the techniques can be categorized into forward or backward strategies. Forward means that the field is integrated forward in direction of the field vectors to calculate the new positions of particles [vW02, YNBH11]. Backward integration calculates the previous position from where properties are transported [MB96, Sta99, JEH02, Wei04]. A combination of both strategies is also possible [KLLR05, SFK*08] to increase the accuracy of the advection.

In this paper, we restrict ourselves to semi-Lagrangian advection because it is highly popular for interactive flow visualization, e.g., see [vW02, JEH02, Wei04]. We do not consider more complex high-quality advection methods that come with much higher computational costs than the semi-Lagrangian methods, such as area sampling techniques [LTH08], finite-volume approaches [KSW*12], or blending of forward and backward advected images in the context of water colorization [BNTS07].

For the integration of a vector field, there is a variety of numerical methods. A simple one is the Euler method, which computes the next location according to first-order approximation. Second-order approximation is implemented by the Leap-Frog method [HLW03]. The Runge-Kutta method is popular because it shows even better results than the Leap-Frog approach at reasonable computational cost [BW96]. We employ fourth-order Runge-Kutta throughout this paper.

Our paper addresses the problem of quantifying the quality of texture advection by spectral analysis. One of the few related papers is by Yu *et al.* [YNBH11], who demonstrate that their Lagrangian method preserves the spectrum of the input texture. Weiskopf [Wei09] regards the spectral appearance of an exponential filter kernel in the context of LIC flow visualization. However, so far no one has compared grid-based and BFECC advection methods in frequency space. This paper is one step toward closing this gap in the literature.

3. Advection

Advection is the process of passive transport of a scalar quantity $\varphi(\boldsymbol{x},t)$ within a vector field $\boldsymbol{v}(\boldsymbol{x},t)$. In the Eulerian frame, the process of advection is described by the continuity equation:

$$\frac{\partial \varphi}{\partial t} + \nabla \cdot (\varphi \boldsymbol{v}) = 0. \tag{1}$$

The scalar quantity φ describes the distribution of the advected material, e.g., the density of smoke or, as in our case, the color intensities $\boldsymbol{\varphi} = (c_R, c_G, c_B)$ of an RGB texture. As

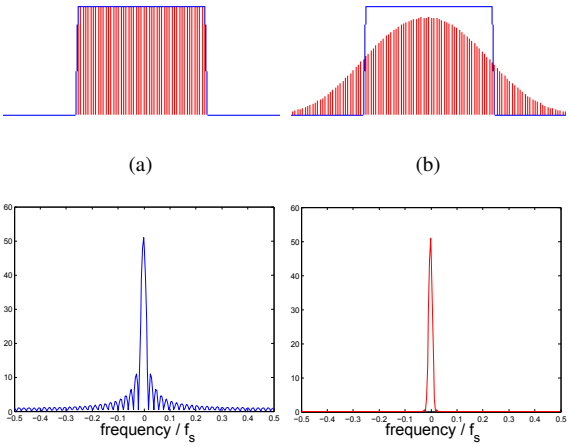


Figure 1: *Numerical diffusion in one dimension. (a) Analytical box function (blue) and discrete samples (red). (b) semi-Lagrange advection after 50 time steps. In the spatial domain, the discrete samples are smoothed across the sharp boundary of the analytic solution. (c) Fourier transformation of the box function. (d) The Fourier transformation of the numerical solution after 50 time steps shows significant damping of high frequencies.*

the channels are independent of each other, we obtain three independent advection equations:

$$\frac{\partial \boldsymbol{\varphi}_i}{\partial t} + \nabla \cdot (\boldsymbol{\varphi}_i \boldsymbol{v}) = 0, \qquad i = 1,2,3, \tag{2}$$

one for each channel. The numerical solution of Equation (2) typically introduces an error term $\boldsymbol{e}_i(\boldsymbol{x},t) \neq 0$ on the right-hand side:

$$\frac{\partial \boldsymbol{\varphi}_i}{\partial t} + \nabla \cdot (\boldsymbol{\varphi}_i \boldsymbol{v}) = \boldsymbol{e}_i, \qquad i = 1,2,3. \tag{3}$$

Depending on the solution method, the error term is dominated by different effects, such as numerical diffusion or dispersion [KK03]. In this paper, we focus on numerical diffusion, which is usually dominant in Eulerian and semi-Lagrangian-based advection algorithms as the consequence of repeated, usually linear, interpolation operations. Figure 1 depicts the impact of numerical diffusion in a 1D example of an advected box function, using the semi-Lagrange method. In this case, a modified partial differential equation is actually solved, i.e., an advection–diffusion equation:

$$\frac{\partial \boldsymbol{\varphi}_i}{\partial t} + \nabla \cdot (\boldsymbol{\varphi}_i \boldsymbol{v}) = \underbrace{D\Delta\boldsymbol{\varphi}_i + \boldsymbol{f}_i}_{=\boldsymbol{e}_i}, \qquad i = 1,2,3, \tag{4}$$

where D is a diffusion coefficient and $\boldsymbol{f}_i$ denotes other sources of errors. We assume that $|\boldsymbol{f}_i| \ll |D\Delta\boldsymbol{\varphi}_i|$, which is a known fact for semi-Lagrangian-based methods [SFK*08] on which we focus in the following sections. In the special case where the vector field $\boldsymbol{v}$ is solenoidal, e.g., if $\boldsymbol{v}$ is the solution of the incompressible Navier-Stokes equations, Equa-

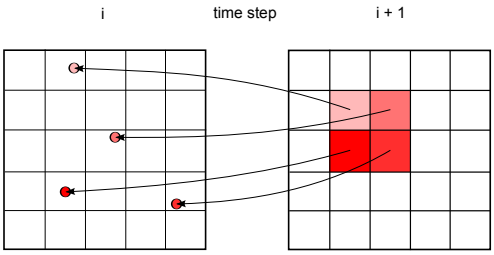


Figure 2: *Backward integration with interpolation of intensity values.*

tion (4) simplifies and yields:

$$\frac{\partial \varphi_i}{\partial t} + \boldsymbol{v} \cdot \nabla \varphi_i = D\Delta \varphi_i + \boldsymbol{f}_i, \qquad i = 1,2,3. \quad (5)$$

However, independent of the properties of the vector field, the additional diffusion term implements a low-pass filter and the magnitude of the diffusion coefficient determines the amount of smoothing. Depending on the algorithm, the diffusion coefficient may be a function of different quantities and its magnitude can vary significantly. While it is possible to find analytic expressions for the diffusion coefficient for simple techniques, such as the semi-Lagrange method in one dimension [BMF07], it is difficult to derive an analytic expression in the general case, especially when limiters and higher-order interpolation schemes are combined in two or more dimensions as in our case. Therefore, we employ spectral analysis to quantify the amount of numerical diffusion by studying the conservation of high frequencies over the advection time.

4. Advection Techniques

In this section, common basic techniques are described that are used to compute advection and we briefly discuss the basic concept of higher-order interpolation. In this paper, we focus on semi-Lagrangian methods in conjunction with BFECC. We illustrate our approch for 2D examples, but the extension to 3D is straightforward.

4.1. Ground Truth

For our spectral analysis and comparison, we require a ground truth that does not exhibit numerical diffusion. Thus, to compute the color of each pixel after a finite advection time $t = T$, we integrate a full path line backward in time to determine the location of each pixel in the original image at advection time $t = 0$ [MB96]:

$$\boldsymbol{x}(0) = \boldsymbol{x}(T) + \int_T^0 \boldsymbol{v}(\boldsymbol{x}(t),t)dt. \quad (6)$$

4.2. Backward

Semi-Lagrangian techniques rely on iterative backward integration and interpolation. Backward integration can be specified by inverting the vectors to compute the intensity that would be advected to a certain location:

$$\boldsymbol{x}(t_i) = \boldsymbol{x}(t_{i+1}) + \int_{t_{i+1}}^{t_i} \boldsymbol{v}(\boldsymbol{x}(t),t)dt. \quad (7)$$

In contrast to some variants of forward integration, backward integration does not produce holes in the image of the current time step. There is also no need for scattering intensity values around adjacent pixels, but an interpolation of the transported intensity is required, which is shown in Figure 2.

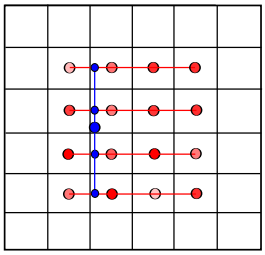


Figure 3: *Higher-order interpolation with polynomial of third degree. First, for each row, a polynomial in x-direction is constructed, and intermediate values are retrieved. Next, these values are used to construct a polynomial in y-direction.*

4.3. BFECC

The Back and Forth Error Compensation and Correction method [DL03, DL07] is a combination of a forward and backward technique and consists of four steps. A first image is generated by backward integration. This image is then used to perform forward integration, which results in a second image. A third image is computed by calculating the difference between the first and the second one. It represents the error that has occurred during the back- and forward integration. In the last step, the intensity values of the original image are modified by adding half of the calculated error to the corresponding intensities of the original image. Based on the corrected image, a backward step is performed to obtain the final image.

4.4. Higher-Order Interpolation

For the quality of the result of an advection method, it is of great importance to handle resampling of intensities with high accuracy. Although bilinear interpolation is attractive due to its high performance, it exhibits high numerical diffusion and causes strong blurring of details. To increase the precision of the interpolation, we employ higher-order interpolation with polynomials of degree three and five. This is depicted in Figure 3. For a polynomial of n-th degree, $(n+1)^2$ points are required in 2D around the position $\boldsymbol{x} = (x,y)^T$ that should be interpolated. Then, for each row in the resulting square, a polynomial is interpolated and evaluated

at a position that is equal to the x-component of $\boldsymbol{x}$, generating $n+1$ intermediate data points. These points are used to interpolate another polynomial and evaluated it at a position that is equal to the y-component of $\boldsymbol{x}$. This leads to the final intensity value at position $\boldsymbol{x}$.

4.5. BFECC with Higher-Order Interpolation

BFECC and higher-order interpolation are orthogonal concepts that have been employed separately in previous approaches to reduce numerical diffusion. However, to the best of our knowledge, a combination of both techniques has not been discussed before in the literature. Therefore, we substitute all bilinear interpolations that are commonly used in BFECC methods with higher-order interpolation schemes of degrees three and five. In the following section, we analyze and quantify the impact on quality for both concepts separately and in combination.

5. Spectral Analysis

A spectral analysis of vector fields using texture advection techniques is appropriate, since a filtering is accomplished that modifies the spectrum. For example, LIC methods, which can produce good representations of vector fields, perform an explicit filtering along streamlines.

In the following, first, the generation of radial power spectrum diagrams is described in detail. After that, a brief description of how to handle spectral evaluation is provided.

5.1. Radial Power Spectrum Diagram

The process for analyzing the spectrum of an image is executed by us in four steps. First, we apply a Hamming window on the input image to avoid the artificial generation of spurious high frequencies from discrete Fourier transform. Next, a radial power spectrum diagram (RPSD) is computed. To this end, we calculate for each radius the average power along a concentric circle located around the zero-frequency-origin of the Fourier-transformed image. This is depicted in Figure 4. At last, we calculate the radius of the circle that covers 99% of the power, and highlight this border in the diagram. This frequency radius can be regarded as a highly aggregated indicator of spectral conservation: the closer the radius of the advected image to the original image, the higher the quality.

5.2. Process of Analysis

One can imagine the process as a top-down approach motivated by the well-known mantra of Shneiderman [Shn96]: "Overview first, zoom and filter, then details-on-demand". Here, the 99% frequency radius and the RPSD can be used for a first glance, since they are aggregated representations of the 2D frequency domain. As a next step, the 2D frequency domain can be observed in detail, and finally the spatial domain as well if necessary.

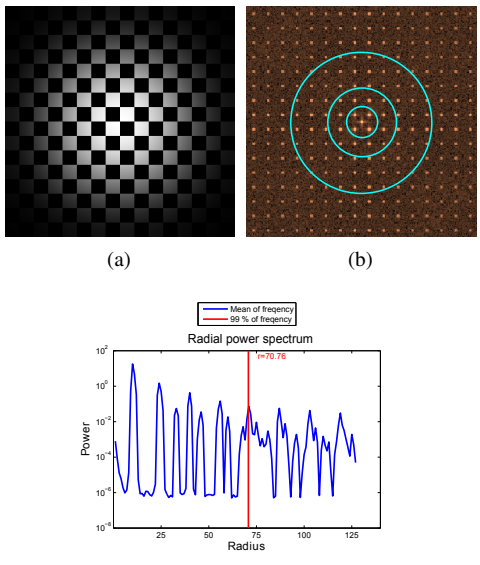


Figure 4: *(a) Input image, (b) frequency-transformed image with some concentric circles, (c) RPSD with average power at circles of varying radius.*

A bottom-up approach is also feasible. Here, someone could learn how observed texture properties in the original spatial domain could manifest in the frequency domain.

5.3. Parameter Choices for Analysis

For the evaluations in this paper, we used two input images, a checkerboard and a Gaussian filtered white noise image to cover a wide range of different spectra. The checkerboard image is a very regular pattern with high and distinct peaks in the spectrum, whereas the noise image is smoothly varying in the Fourier domain and is typically employed for LIC. Both textures were advected with an artificial circular and a realistic Navier-Stokes vector field obtained by numerical simulation. For the integration of the vector fields, we chose the time step size $\Delta t = 0.05$, which is small enough to make the numerical error for integration with fourth-order Runge-Kutta negligible. We processed 500 time steps and evaluated the advection after 25, 50, 100, 250, and 500 steps. Additionally, we investigated the influence of the resolution of input images on the advection. For this purpose, we generated a Gaussian filtered white noise image of size 64×64. We further modified this image in frequency space by cutting off all frequencies located at a distance of more than 32. Upsampling was done by zero padding in the Fourier domain three times to generate textures of size 128×128, 256×256, and 512×512.

6. Results

In this section, the results of our evaluation of backward and BFECC methods in combination with different interpolation strategies for texture advection are presented to demonstrate the amount of spectral change caused by blurring. This blurring effect manifests itself in the form of decreasing 99% frequency radius in the RPSD for increasing time. A small radius indicates that low frequencies are dominant in a texture, which corresponds to blurring. Larger radii indicate an inclusion of higher frequencies, which corresponds to better preservation of the spectrum. Hence, also image features are better conserved and blurring is less pronounced. This is shown in Figure 5 for a circular vector field and a checkerboard texture with resolution 256×256.

A Gaussian filtered noise image with the same resolution was used for Figure 6. By comparing the radii of the RPSDs, calculated after advection over 50 and 500 time steps, it can be seen that by increasing the interpolation order, the radii also increase and the blurring effect is reduced. The BFECC methods further increase the radii and thereby decrease blurring compared to backward methods.

Table 1 and Table 2 contain the radii for all time steps. Here, one can see a decrease of the radii with progressing advection for all methods and also increasing radius values for increasing interpolation order. Table 3 contains the radii for the checkerboard and the noise images after 100 time steps of advection with the Navier-Stokes vector field. Here, one can also see that the higher-order strategies perform better. An evaluation based on the Navier-Stokes vector field with more then 250 time steps was not possible due to artifacts that were produced by the BFECC methods. These artifacts are depicted in Figure 7. The artifacts cause spurious high frequency, which leads to an artificial increase of the radii. This can also be seen in Figure 8(a) and 8(b).

Now we consider the convergence speed of methods that use interpolation of different orders. The input images for the advection are generated by upsampling the Gaussian filtered noise image as described in Section 5.3. The images are advected 500 time steps. Figure 9 shows convergence plots for all combinations of methods. These plots represent the ratio of the 99% frequency radii of the individual advection techniques and the ground truth. One can see that higher-order interpolation leads to faster convergence to 1 with increasing resolution. Also, BFECC influences the convergence positively compared to the backward plots. Nevertheless, the BFECC strategies cause artifacts. However, the influence of the artifacts decreases with higher resolutions. This can be seen in the ratio values of the BFECC curve with fifth-order interpolation, which are greater than one in the beginning.

The presence of artifacts that introduce high frequencies is also noticeable in the RPSD. This is the case in Figure 8(b). Here, the average power rises again at the end of the plot.

Table 1: *Radii for containment of 99% of frequency for all tested methods and time steps. The checkerboard image and the circular vector field are used.*

	Time Steps	25	50	100	250	500
	Ground Truth	55.1	54.7	54.9	55.3	55.4
Backward	bilinear	24.7	24.4	23.5	11.9	13.1
	3rd order	39.3	38.4	24.9	24.7	24.5
	5th order	39.6	39.5	39.1	38.1	25.5
BFECC	bilinear	39.0	30.4	24.5	24.4	24.0
	3rd order	39.4	39.0	32.5	24.9	24.7
	5th order	39.7	39.5	39.2	38.3	25.9

Table 2: *Radii for containment of 99% of frequency for all tested methods and time steps. The Gaussian filtered noise image and the circular vector field are used.*

	Time Steps	25	50	100	250	500
	Ground Truth	118.6	118.9	118.1	118.7	118.6
Backward	bilinear	83.7	59.7	42.5	25.1	13.1
	3rd order	92.6	76.0	58.9	43.9	36.3
	5th order	98.1	85.8	70.8	56.3	49.8
BFECC	bilinear	93.5	75.5	56.0	38.9	29.4
	3rd order	114.6	107.4	92.3	57.6	43.8
	5th order	118.5	115.2	107.9	87.7	64.3

The 2D frequency images (Figure 8(d)) reveal some obvious peaks at the high-frequency boundaries that are reflected in the original texture (Figure 7(c)). This top-down analysis can also be inverted to a bottom-up approach. In any case, a look at the spatial domain is recommended for final analysis because the aggregated view in the RPSD could hide some effects through blurring caused by transport.

The computation speed is another important issue. Table 4 contains the average FPS rates over an advection time of 100 steps. The values were measured on a system containing an Intel i7 CPU with 3.4 GHz. Furthermore, all methods were

Table 3: *Radii for containment of 99% of frequency for all tested methods after 100 time steps for the Gaussian filtered noise image and the checkerboard image. The Navier-Stokes vector field is used.*

		Checkboard	Noise
	Ground Truth	120.3	123.5
Backward	bilinear	72.3	90.4
	3rd order	89.6	99.9
	5th order	101.8	108.5
BFECC	bilinear	96.3	96.1
	3rd order	110.1	115.1
	5th order	119.3	121.5

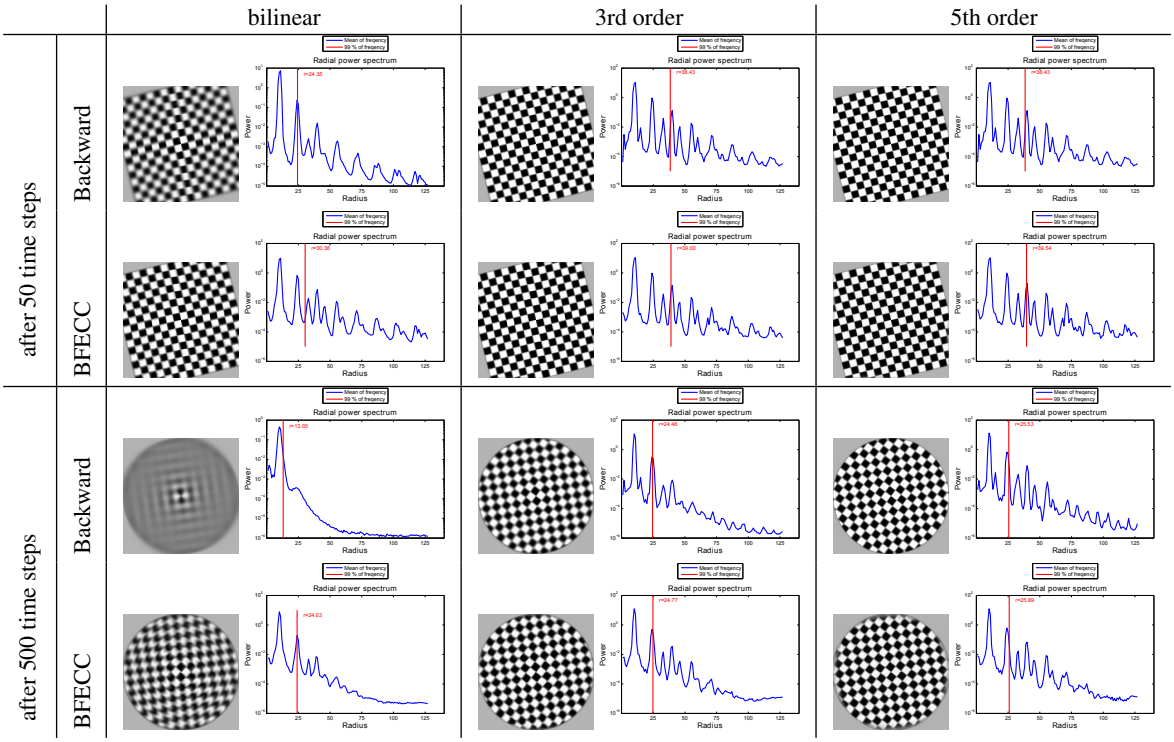


Figure 5: *Advection images and RPSDs after 50 and 500 time steps for backward and BFECC based methods with different interpolation strategies. The checkerboard is used as input image. The vector field is circular.*

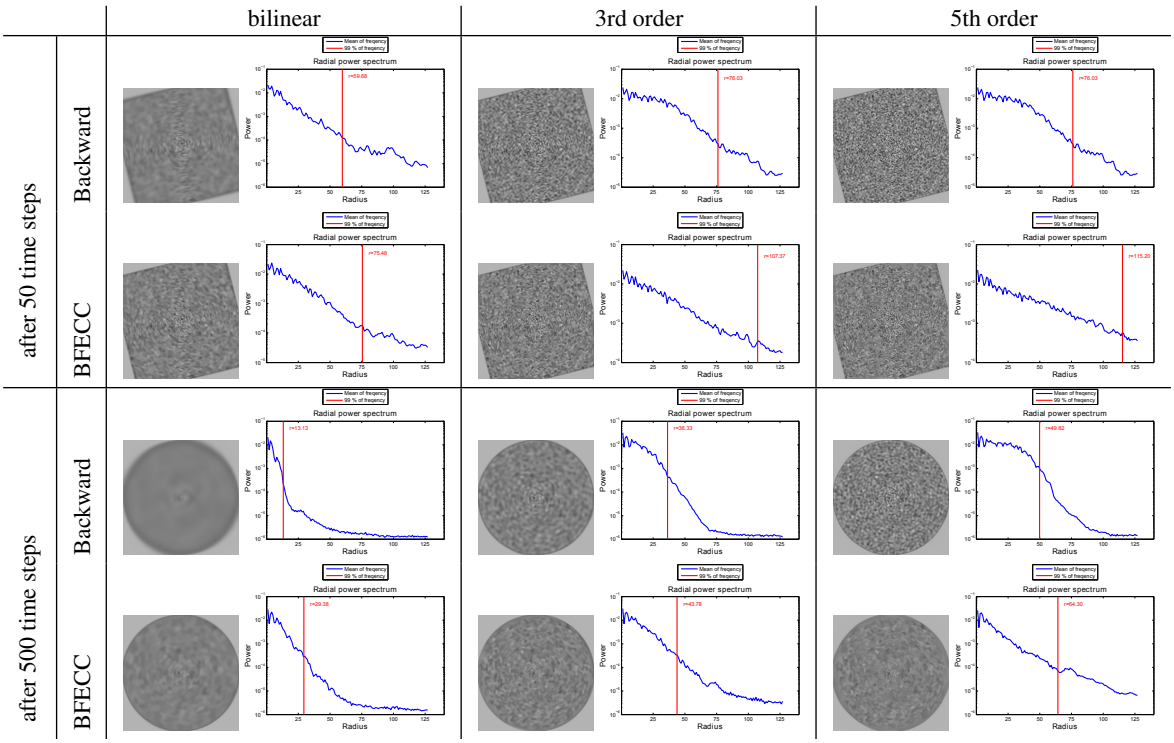
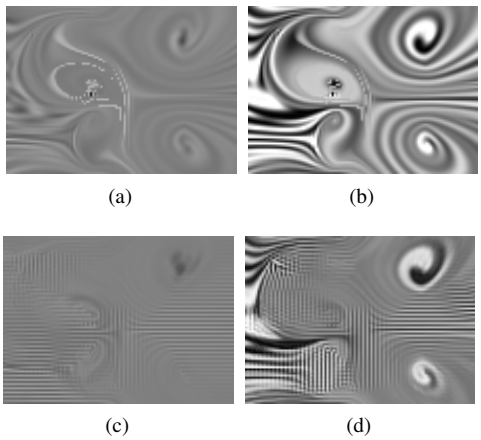
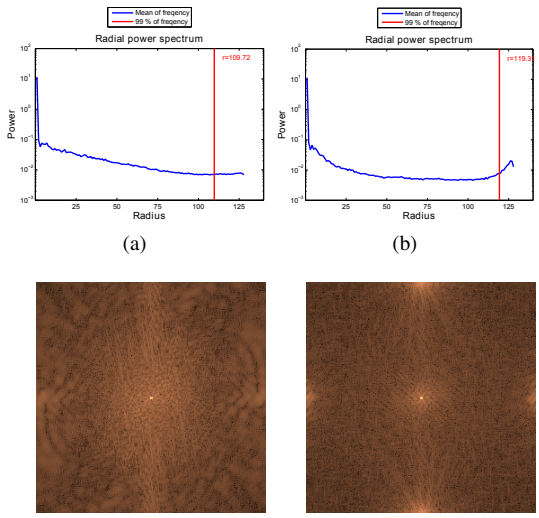


Figure 6: *Advection images and RPSDs after 50 and 500 time steps for backward and BFECC based methods with different interpolation strategies. The Gaussian filtered white noise image is used as input image. The vector field is circular.*



(a) (b) (c) (d)

Figure 7: *Apertures of advection images after 1000 time steps with 5rd order interpolation, applied to noise (left) and checkerboard image (right). Images (a) and (b) are based on a backward method and have no artifacts. Images (c) and (d) are based on BFECC and show artifacts.*



(a) (b) (c) (d)

Figure 8: *Frequency images for backward fifth-order (c), and BFECC fifth-order (d) interpolation, based on Gaussian filter noise image after 1000 time steps of advection. A vector field in manner of Navier-Stokes was used, see Figure 7. Images (a) and (b) are the corresponding radial power spectrum diagrams.*

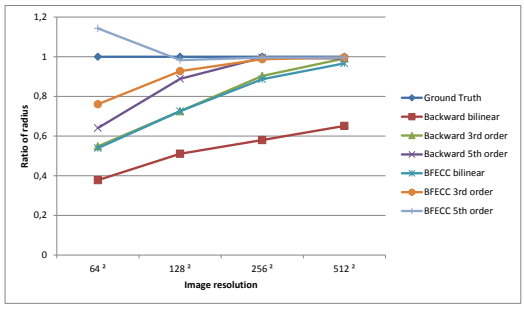


Figure 9: *Convergence speed of methods towards the ground truth over different image resolutions. Values are ratios between the actual radius and the ground truth radius after 500 time steps of advection.*

implemented in a single threaded version, without any focus of optimization. Table 4 shows a few interesting properties. First, by using BFECC the FPS rate drops by a factor of about 4. Considering the number of computation steps, described in Section 4.3, this drop is justified. The next thing that can be noticed is a decreasing FPS rate with increasing order of interpolation. Comparing the FPS rates of the bilinear and third-order interpolation, a decrease by a factor of about 6 arises and for fifth-order by a factor of 10. This is a greater loss then expected regarding the number of necessary texture lookups. We believe that these effects are caused by the cache misses.

A collection of more detailed evaluations can be found on http://www.vis.uni-stuttgart.de/texflowvis.

7. Conclusion

We have proposed an approach to analyze the quality of texture advection techniques. Our approach is based on radial power spectrum diagrams and is complemented by visual inspection of the advected images in the spatial domain. Furthermore, the 2D frequency image can be inspected to identify effects that occur during the advection. The advantage of the highly aggregated measure of the 99% frequency radius is that it can be applied to many advection images without producing too much data. As shown, it can be used for advected images for multiple time steps. Here, we have identified the dependency between decreasing radii and blurring of the advected images. Furthermore, we have shown the influence of the resolution of an input image for advection by regarding the convergence speed for all tested combinations of methods.

On the side of flow visualization techniques, we have introduced a combination of higher-order interpolation and BFECC that can be used to reduce the effect of blurring.

All over, our evaluation has shown that the BFECC approach with third-order interpolation generates highly accu-

Table 4: *Measurements of FPS rates*

	bilinear	3rd order	5th order
Backward	106.6	15.7	9.5
BFECC	24.1	4.2	2.4

rate advection textures, but suffers from low FPS rates. The backward integration with fifth-order interpolation tends to result in similar texture quality and higher FPS rates. In general, this approach exhibits the best tradeoff between computation speed and texture quality. Notice that the frame rates were obtained here from single threaded versions of the different techniques. Much higher rates can easily be achieved by multi-threaded implementations.

Acknowledgements

This work was funded by German Research Foundation (DFG) within project WE 2836/4-1. We thank Stephan Starke for thorough implementation work and initial comparisons. We also thank all reviewers for their constructive comments.

References

[BMF07] BRIDSON R., MÜLLER-FISCHER M.: Fluid simulation. In *ACM SIGGRAPH 2007 Course Notes* (2007), pp. 1–81. 3

[BNTS07] BOUSSEAU A., NEYRET F., THOLLOT J., SALESIN D.: Video watercolorization using bidirectional texture advection. *ACM Transaction on Graphics 26*, 3 (2007), Article 104. 2

[BW96] BUTCHER J., WANNER G.: Runge-Kutta methods: Some historical notes. *Applied Numerical Mathematics 22* (1996), 113–151. 2

[CL93] CABRAL B., LEEDOM L. C.: Imaging vector fields using line integral convolution. In *ACM SIGGRAPH '93* (1993), pp. 263–270. 1

[DL03] DUPONT T. F., LIU Y.: Back and forth error compensation and correction methods for removing errors induced by uneven gradients of the level set function. *Journal of Computational Physics 190*, 1 (2003), 311–324. 3

[DL07] DUPONT T. F., LIU Y.: Back and forth error compensation and correction methods for Semi-Lagrangian schemes with application to level set interface computations. *Mathematics of Computation 76* (2007), 647–668. 3

[HLW03] HAIRER E., LUBICH C., WANNER G.: Geometric numerical integration illustrated by the Störmer/Verlet method. *Acta Numerica 12* (2003), 399–450. 2

[JEH02] JOBARD B., ERLEBACHER G., HUSSAINI M.: Lagrangian-Eulerian advection of noise and dye textures for unsteady flow visualization. *IEEE Transactions on Visualization and Computer Graphics 8*, 3 (2002), 211–222. 2

[KK03] KARNIADAKIS G. M., KIRBY R. M.: *Parallel Scientific Computing in C++ and MPI*. Cambridge University Press, 2003. 2

[KLLR05] KIM B., LIU Y., LLAMAS I., ROSSIGNAC J.: Flowfixer: Using BFECC for fluid simulation. In *Workshop on Natural Phenomena* (2005), pp. 51–56. 2

[KSW*12] KARCH G. K., SADLO F., WEISKOPF D., MUNZ C.-D., ERTL T.: Visualization of advection-diffusion in unsteady fluid flow. *Computer Graphics Forum 31*, 3 (2012), 1105–1114. 2

[LTH08] LI G.-S., TRICOCHE X., HANSEN C.: Physically-based dye advection for flow visualization. *Computer Graphics Forum 27*, 3 (2008), 727–734. 1, 2

[MB96] MAX N., BECKER B.: Flow visualization using moving textures. In *ICAS/LaRC Symposium on Visualizing Time-Varying Data* (1996), pp. 77–87. 2, 3

[Ney03] NEYRET F.: Advected textures. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2003), pp. 147–153. 1

[SFK*08] SELLE A., FEDKIW R., KIM B., LIU Y., ROSSIGNAC J.: An unconditionally stable MacCormack method. *Journal of Scientific Computing 35*, 2-3 (2008), 350–371. 2

[Shn96] SHNEIDERMAN B.: The eyes have it: a task by data type taxonomy for information visualizations. In *IEEE Symposium on Visual Languages* (1996), pp. 336–343. 4

[Sta99] STAM J.: Stable fluids. In *ACM SIGGRAPH '99* (1999), pp. 121–128. 2

[vW91] VAN WIJK J. J.: Spot noise texture synthesis for data visualization. *Computer Graphics (Proceedings of SIGGRAPH '85) 25*, 4 (1991), 309–318. 1

[vW02] VAN WIJK J. J.: Image based flow visualization. *ACM Transactions on Graphics 21*, 3 (2002), 745–754. 2

[WBE05] WEISKOPF D., BOTCHEN R. P., ERTL T.: Interactive visualization of divergence in unsteady flow by level-set dye advection. In *Conference on Simulation and Visualization* (2005), pp. 221–232. 2

[Wei04] WEISKOPF D.: Dye advection without the blur: A level-set approach for texture-based visualization of unsteady flow. *Computer Graphics Forum 23* (2004), 479–488. 1, 2

[Wei09] WEISKOPF D.: Iterative twofold line integral convolution for texture-based vector field visualization. In *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, Möller T., Hamann B., Russell R., (Eds.). Springer, 2009, pp. 191–211. 2

[WSEE05] WEISKOPF D., SCHRAMM F., ERLEBACHER G., ERTL T.: Particle and texture based spatiotemporal visualization of time-dependent vector fields. In *IEEE Conference on Visualization* (2005), pp. 639–646. 1

[YNBH11] YU Q., NEYRET F., BRUNETON E., HOLZSCHUCH N.: Lagrangian texture advection: Preserving both spectrum and velocity field. *IEEE Transactions on Visualization and Computer Graphics 17*, 11 (2011), 1612–1623. 2