FEM Techniques for Multiscale Visualization of Time-Dependent Flow Fields

Jens F. Acker* Jaroslaw Hron* Tobias Preusser†
Martin Rumpf‡ Stefan Turek*

Corresponding author: Jens F. Acker
EMail: jens.acker@math.uni-dortmund.de

Abstract

Multiscale visualization techniques for static as well as time-dependent flow fields are indispensable tools for the understanding and analysis of complex flow phenomena. The methods presented in [2, 18] generate multiscale visualizations and they are based on an anisotropic diffusion, resp., anisotropic transport diffusion operator. However, the presented discretizations are only given on quadtree or octree grids, which in general are impracticable in real applications for *Computational Fluid Dynamics* (CFD). In this work, we extend these methods in [2] and we discuss robust and accurate FEM discretization and iterative solver techniques on unstructured quadrilateral grids. Thus, undesirable numerical diffusion as well as numerical oscillations are avoided and efficiency of the numerical implementation is ensured. Together with improved blending strategies the effective visualization of complex flow fields in 2D is possible.

1 Introduction

The analysis and postprocessing is still one of the fundamental tasks in scientific visualization. Sophisticated multiscale methods are needed to visualize and analyze the structure of especially nonstationary flow fields for which the standard tools may fail. A huge variety of techniques for the visualization of steady as well as time-dependent flow fields in 2D and 3D has been presented during the last years. The methods currently available range from particle tracing approaches [24, 27] over texture based methods [6, 26, 3, 19, 9] to feature extraction for 3D flow fields [5, 20, 8, 10].

Recently, Westermann et al. [30] presented a level set method for flow visualization. They converted the flow into a scalar density by considering arrival times of flow surfaces. By analyzing the curvature of the level sets of the new scalar density they obtain

*Institute for Applied Mathematics and Numerics, University of Dortmund, jens.acker@math.uni-dortmund.de, jaroslav.hron@math.uni-dortmund.de, ture@featflow.de

†CeVis, University of Bremen, preusser@cevis.uni-bremen.de

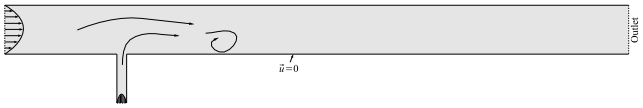
‡Institute for Numerical Simulation, University of Bonn, martin.rumpf@ins.uni-bonn.de

a separation and extraction of “principal streams” from the flow, which are then used to give a representation of the flow field. Their analysis of the level set curvatures relies on methods which are similar to surface fairing approaches, which is further supported by the use of hardware texture rendering. Jobard et al. [11] have further elaborated on the use of pixel texture hardware by extending the methodology presented by Heidrich et. al [7]. Their approach treats the handling of unsteady flows, the advection of dye and the extraction of features. A white noise is used to obtain textures which are then blended together to obtain an animated visualization of the flow. Recently [12] presented a combination of a Lagrangian and an Eulerian approach by combining the advection of particles and the texture generation from white noise. The use of programmable per pixel operations of modern texture hardware of standard PC graphics cards has been improved in the works of Weiskopf et al. [29] and van Wijk [28]. The image based flow visualization approach of van Wijk uses textures resulting from advected and diffused white noise to obtain an animated sequence at high frame-rate. Thereby the advection is completely computed by the graphics hardware using a low order Runge-Kutta scheme. Bruckschen et al. [1] presented a particle tracing approach that handles large 3D data, providing a high frame rate. Similar like all particle tracing approaches, this unfortunately does not always give sufficient insight into the structure of complex flow phenomena. For illustration we provide here three examples in which case the particle tracing approach does not lead to fully satisfying results:

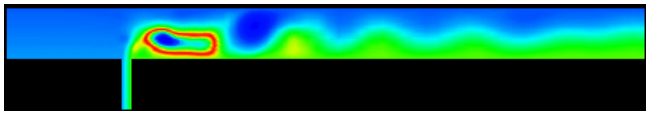
1. **Channel with perpendicular Pipe:**
   One problem of particle tracing methods lies in properly resolving the structure of convoluted and complex flows. Injecting a fast flow into a channel containing a relatively slow flow as seen in Fig. 1(b) creates a convoluted flow with a series of vortices behind the injection point. First to note is that the flow coming from the left is relatively slow and it takes time for the particles to reach the region where the flows are merging. For this reason the particles are entered only a small distance away from the perpendicular pipe. Additionally it is hard (as seen in Fig. 1(c)) to discern detailed flow features in the corresponding particle tracing because of the high particle density.

2. **Venturi Pipe:**
   Another problem arises when there are cases where globally complex flows can hide the existence of other local flow features. In our Venturi pipe example we push a flow through a pipe with a narrow region. There is a pipe attached at the narrowest point. The main flow creates a pressure drop which causes another flow from that attached pipe (see Fig. 2(a)). If particles are only put on the left boundary, the flow from the attached pipe is nearly invisible.

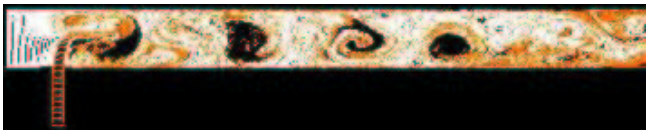
3. **Driven Cavity:**
   In this example there is a box of size 3x1 (height x width) with a tangential flow at the top (see Fig. 3(a)). This flow induces three big vortices and several small ones in the corners, depending on the Reynolds number. If a particle source is placed on top, the flow will carry most of it around the first vortex and give some particles to the vortex below and the second to the third. The differences in flow speed in these vortices are big and it takes a long time until particles get to the



(a) Flow configuration



(b) Velocity magnitude



(c) Visualization obtained via massive particle tracing

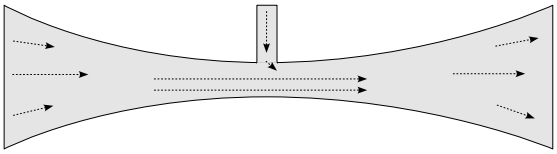
Figure 1: Snapshots for ‘Channel with perpendicular pipe’

third vortex. Until that happens, the third vortex will be invisible in the particle tracing. Only if the box is filled from the start with particles, all vortices can be seen in reasonable time (see Fig. 3(d)).

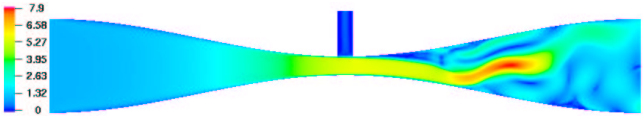
It can be concluded that particle tracing is sensitive to the right positioning of particle sources and the number of particles used. Still there is the need for tools that allow the intuitive analysis of 2D and 3D data sets, including the extraction and exploration of flow features.

In this paper we present an extension of the anisotropic transport diffusion method to complex flow fields resulting from CFD computations on arbitrary grids and use it for the Venturi pipe problem. The discretization presented in [2] was based on quadtree or octtree grids which are impracticable in real applications since vector field data coming from CFD computations is in general not given on uniform rectangular grids.

Therefore, for general unstructured meshes, we apply the discretization of the arising transport diffusion problems by the streamline-diffusion (SD) FEM scheme, and we discuss iterative solvers of type Krylov-space or multigrid schemes for the arising nonsymmetric auxiliary problems. Moreover, we analyse a corresponding balancing of the involved operators and blending strategies which are demonstrated for several test examples and which show that these numerical techniques are excellent candidates for efficient visualization methods of highly nonstationary flow with complex multiscale behaviour in space and time.



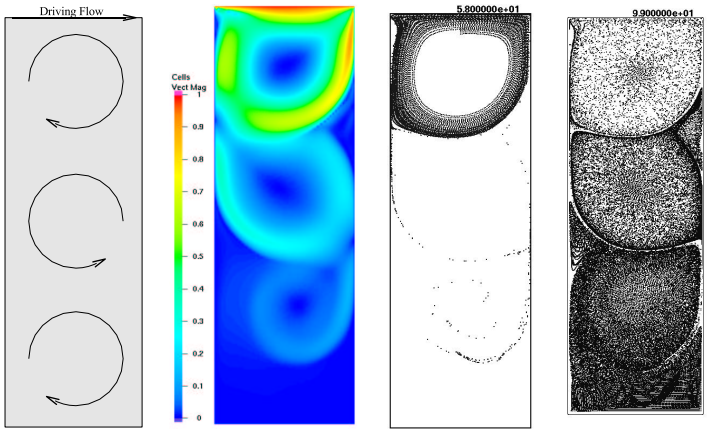
(a) Flow configuration



(b) Velocity magnitude

(c) Particle Tracing

Figure 2: Snapshots for ‘Venturi Pipe’



(a) Flow configuration

(b) Velocity magnitude

(c) Particle Tracing (top insertion)

(d) Particle Tracing (full insertion)

Figure 3: Snapshots for ‘Driven cavity’

2 The Anisotropic Transport Diffusion Method

In [2, 18], special methods which are based on anisotropic diffusion and transport anisotropic diffusion for the visualization of static and time-dependent vector fields have been presented. In this section we are going to review briefly these models, the according parameters and a blending strategy which is needed to produce a visualization of time-dependent flow fields.

2.1 The transport diffusion operator

We consider a time-dependent vector field

$$v : I \times \Omega \to \mathbb{R}^d, \qquad (s, x) \mapsto v(s, x)$$

given on a finite time-space cylinder $I \times \Omega$ where $I = [0, T]$ and $\Omega \subset \mathbb{R}^d$ for $d = 2, 3$. Here, we restrict to $d = 2$. If the vector field v is constant in time, i. e. $v(s, x) = v_0(x)$ for all $s \in I$, we can create a multiscale visualization of the flow field in form of a family of textures $\{u(t)\}_{t\in\mathbb{R}^+}$ by the following anisotropic diffusion equation:

Find $u : \mathbb{R}^+ \times \Omega \to \mathbb{R}$ *such that*

$$\begin{aligned} \partial_t u - \operatorname{div}(A(v, \nabla u)\nabla u) &= f(u) && \text{in } \mathbb{R}^+ \times \Omega, \\ A(v, \nabla u)\partial_n u &= 0 && \text{on } \mathbb{R}^+ \times \partial\Omega, \\ u(0, \cdot) &= u_0(\cdot) && \text{in } \Omega. \end{aligned} \tag{1}$$

We start this evolution with an initial image u_0 showing random white noise. Since we have assumed the vector field to be continuous, there exists a family of orthogonal mappings $B(v) \in SO(d)$ such that $B(v)e_1 = v$. And denoting the identity matrix of dimension d with Id_d, the diffusion tensor reads

$$A(v, \nabla u) = B(v) \begin{pmatrix} \alpha(\|v\|) & 0 \\ 0 & G(\|\nabla u\|)\mathrm{Id}_{d-1} \end{pmatrix} B(v)^T$$

where α is a monotone increasing function which prescribes a linear diffusion in direction of v for $\|v\| > 0$. We will choose α appropriately below. During the evolution, patterns are generated which are aligned with the flow field. The function $G(s) := \epsilon/(1 + c\, s^2)$ - well known in image processing [16] - controls the diffusion in the directions orthogonal to the flow. The function G is modelled such that the evolution performs a clustering of streamlines and thus generates coarser representations of the vector field with increasing scale t. The definition of the diffusion tensor G depends on the gradient of a regularized image $u^\sigma = u * \chi^\sigma$. This regularization is theoretically important for the well-posedness of the presented approach [13, 4]. To our experience, in the implementation this regularization can be neglected or can be replaced by a lower bound for the value of $G(\cdot)$. Additionally, for $\|v\| = 0$ there is no direction given. In this case we use an isotropic diffusion operator instead. The role of the right hand side $f(u)$ of the initial and boundary value problem (1) is to strengthen the contrast of the image during the evolution, because for $f = 0$ the asymptotic limit would be an image

of constant gray value. We set $f(u) = \rho \cdot \big((2\,u-1) - (2\,u-1)^3\big)$ with $\rho = 80$ (see Fig. 5) to increase the set of asymptotic states of the evolution. An example[1] of the multiscale evolution is shown in Figure 4, where the multiscale visualization of a flow field is displayed for the Venturi pipe problem.

Let us now suppose that the vector field varies smoothly in time. If we would consider the evolution equation separately for each fixed time $s \in I$, the resulting textures at a fixed scale $t_0 \in \mathbb{R}^+$ would not give a smooth animation of the flow in time. This is due to a lack of correlation between the line-structures of the separate textures. However, if there would be a correlation between the structure of separate textures, the resulting animation would only give an Eulerian type representation of the flow.

To obtain a Lagrangian type representation, we consider the following anisotropic transport diffusion operator for the mutiscale representation $u : \mathbb{R}^+ \times \Omega \to \mathbb{R}$

$$L_v[u] := \partial_t u + v \cdot \nabla u - \mathrm{div}(A(v, \nabla u)\nabla u)$$

and the corresponding inhomogeneous transport diffusion equation

$$\begin{aligned} L_v[u] &= f(u) && \text{in } \mathbb{R}^+ \times \Omega, \\ A(v)\partial_n u &= 0 && \text{on } \mathbb{R}^+ \times \partial\Omega \\ u(0,\cdot) &= u_0(\cdot) && \text{in } \Omega. \end{aligned} \tag{2}$$

In this equation we have identified the time s of the vector field with the scale t of the multiscale visualization. Indeed the resulting texture shows again structures aligned with streamlines which are now transported with the flow. But due to the coupling of s and t the feature scale gets coarser with increasing time, i.e. we are not able to fix a scale t_0 and typically an animation is created at this scale showing always patterns of the same size. This fact makes the use of an appropriate blending strategy unavoidable.

2.2 Balancing the parameters

In general the transport and the diffusion of the patterns of the texture u are opposite processes. Denoting a time-step of the transport diffusion equation with Δt and introducing the balance parameter $\beta > 0$ we have [2]

$$\alpha(\|v\|)(x) = \frac{\beta^2 \max(\|v(x)\|, \|v\|_{min})^2 \Delta t}{2}$$

and the following relations hold

$\beta \ll 1$	Transport dominates the model,
$\beta = 1$	Transport $\approx$ Diffusion,
$\beta \gg 1$	Diffusion dominates the model.

In our applications we always use the setting $\beta = 10$ and $\|v\|_{min} = 0.05$. Examples for other values of β can be seen in Fig. 6.

[1] This example was computed with a time step of $\Delta t = 0.005$ on a mesh with 82753 nodes.

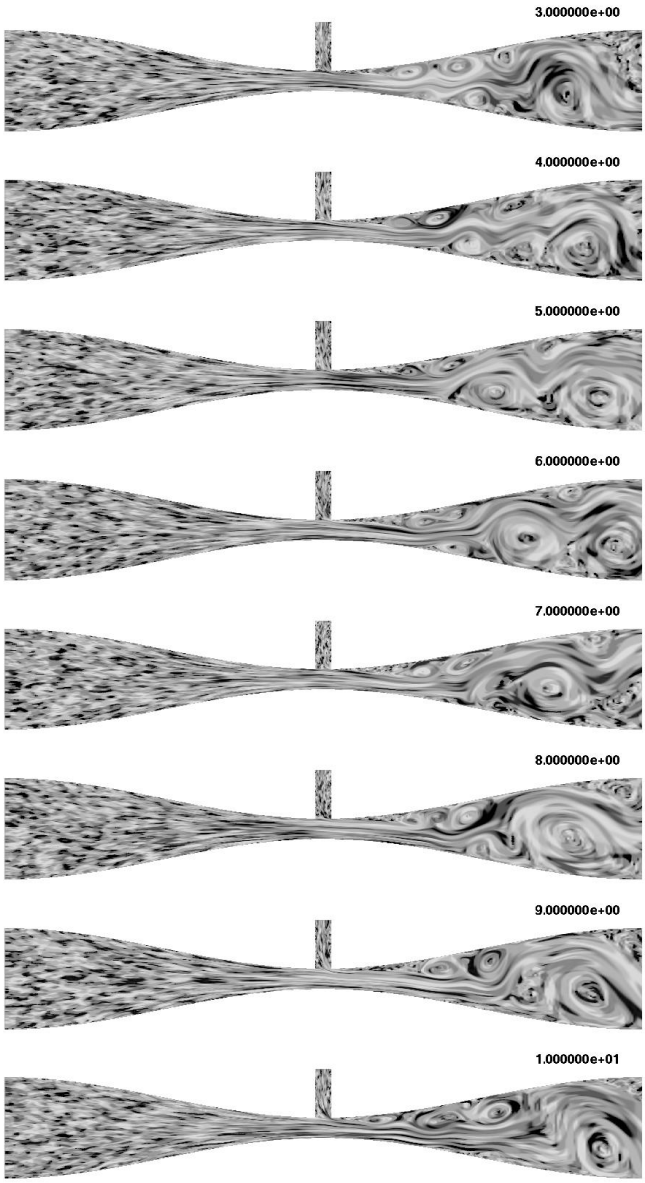


Figure 4: Multiscale visualization of the Venturi Pipe example (with transport)

2.3 Blending strategies

Blending strategies have to be used to get a visual representation of a given flow inside of a desired feature scale range. This means we have a set of solutions each started at a different time representing different feature scales which will be blended together.

(a) $\rho = 0.1$

(b) $\rho = 1$

(c) $\rho = 10$

(d) $\rho = 80$

Figure 5: Animated multiscale visualizations with different values of ρ (see page 6) based on the coupling of scale and physical time

Different blending strategies are possible, e.g. trigonometric functions, interpolating splines, etc. We are currently using a Bézier-spline based approach combined with a specialized startup phase.

At the startup phase we will bootstrap our blending from one solution to the final number n_{tot} of solutions. The current number of solutions shall be denoted with n_{akt} and the time between solution switches/reinitializations with Δt_{blend}. The solutions are handled in an array. After time Δt_{blend}, the oldest solution will be overwritten with noise and a ring shift will be carried out that brings the second oldest solution to the position of the oldest. In the startup phase a start solution containing noise is inserted

(a) $\beta = 0.1$

(b) $\beta = 1$

(c) $\beta = 10$

Figure 6: Multiscale visualizations with different values of β

at the start of the array and all other solutions are shifted one index position higher.

Our blending strategy is split into two parts. The first is handled by a function called *ParCalc* mapping the current simulation time t into the intervall $[0, 1]$, and the second by a function called *BlendCoeffCalc* that takes this mapped value and computes an array with blending weights by evaluating the n_{tot} Bernstein-polynomials of degree $n_{tot} - 1$.

$$\textit{ParCalc}(t, \Delta t_{blend}, n_{akt}, n_{tot}, flag) := \begin{cases} \frac{\mathrm{mod}\ (t, \Delta t_{blend})}{\Delta t_{blend}\, n_{tot}} + \frac{n_{tot}-1}{2\, n_{tot}} & \text{if } flag \text{ is } \textit{true} \\ \frac{t}{\Delta t_{blend}\, n_{tot}\, n_{akt}} + \frac{n_{akt}-1}{2\, n_{akt}} & \text{if } flag \text{ is } \textit{false} \end{cases}$$

with $flag$ as a boolean variable initially set to *false*. For an comparison between the use of two and four blended solutions, see Fig. 7 and Fig. 4. Is is obvious that the use of more blended solutions increases the smoothness of the transitions between visible feature scales. However, the computational time increases linearly with the number of used solutions which comes down to a tradeoff between quality and time. For preview purposes, two blended solutions are sufficient. High quality visualizations will need

more.

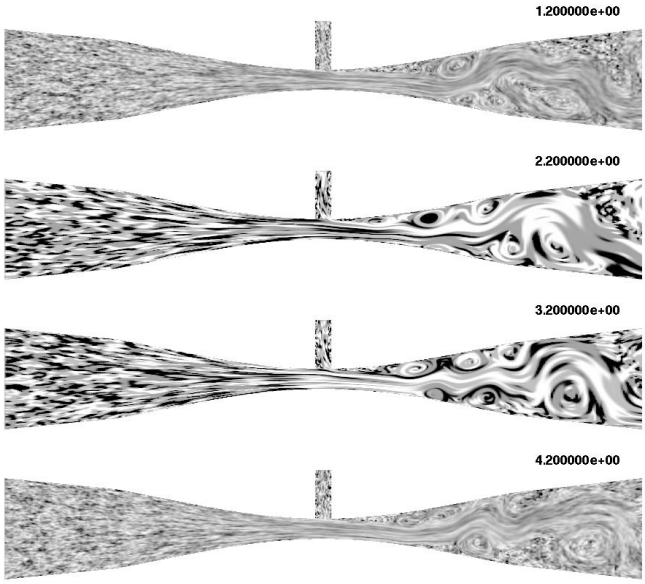


Figure 7: Multiscale visualization with only two blended solutions

3 Discretization and solver aspects

In [2] a characteristic-upwinding algorithm due to Pironneau [17] is used to discretize the transport diffusion scheme (2) on quadtree/octtree grids for $\Omega = [0,1]^d$. For the diffusive parts and the right hand side a semi-implicit scheme is applied, which results in the evaluation of A and f at the previous time steps (see eqn. (3) with $v_n := v(x, t_n)$ and the time step Δt):

$$\frac{u_{n+1} - u_n}{\Delta t} + v_{n+1} \cdot \nabla u_{n+1} - \text{div}(A(v_{n+1}, \nabla u_n)\nabla u_{n+1}) = f(u_n) \qquad (3)$$

However the application of the anisotropic diffusion visualization method on rectangular or cubical domains is often unrealistic in complex CFD applications. Moreover, vector field data typically coming from CFD simulations is rarely given on structured quadtree/octtree grids. Furthermore, the scheme introduces some numerical diffusion which can decrease the quality of the final animation. In this section we discuss a higher order discretization scheme on general meshes which leads to high quality animations, showing sharp patterns moving with the flow field.

3.1 The SD-FEM scheme

The variational formulation of equation (3) reads

$$(u_{n+1},\psi) + \Delta t(v_{n+1}\cdot\nabla u_{n+1},\psi) + \Delta t(A(v_{n+1},\nabla u_n)\nabla u_{n+1},\nabla\psi) = \\ \Delta t(f(u_n),\psi) + (u_n,\psi) \quad \forall\psi\in\mathcal{V} \quad (4)$$

with the test function space $\mathcal{V}$ and test functions ψ (see [15] for more details on variational formulations and the FEM method).

The convection part of our equation demands some kind of additional stabilization. Since the diffusion operator A is already specified, or better decomposed, in a way that allows to control the diffusion in flow direction, we replace A with a slightly modified version $\tilde{A}$:

$$\tilde{A}(v,\nabla u) = B(v)\begin{pmatrix}\alpha(\|v\|)+sd & 0\\ 0 & G(\|\nabla u\|)\mathrm{Id}_{d-1}\end{pmatrix}B(v)^T$$

This modification allows an easy implementation of the streamline-diffusion scheme. The scalar function sd is the necessary ***streamline diffusion*** added in flow direction and is computed by

$$Re_{loc} := \frac{\|v\|_{loc}\,h_{loc}}{\alpha(\|v\|)} \qquad sd := sd_{par}\,h_{loc}\frac{Re_{loc}}{1+Re_{loc}}.$$

The parameter sd_{par}, typically chosen between 0 and 2, is user-specified and h_{loc} is the local mesh width, that means defined on each mesh cell, analogously to $\|v\|_{loc}$ as local flow speed (see [21] for more details). The advantage of this scheme is that it can be easily applied on general unstructured meshes, giving sufficient robustness for treating the convection dominated parts while at the same time the amount of numerical diffusion is not too big. Moreover, since it is a *linear scheme* - in contrast to TVD methods - the resulting subproblems are linear and can be efficiently treated via standard iterative solvers. However, being a linear scheme, the SD scheme suffers potentially from spurious numerical oscillations, due to over and undershooting, and the choice for the user-specific parameter sd_{par} can be critical. In a forthcoming paper, we plan to analyze the influence of the parameter sd_{par} onto the behaviour of accuracy, robustness and efficiency of the described numerical approaches.

3.2 Efficient solvers

The previous variational formulation and its FEM discretization generates linear subproblems of the form

$$(M+\Delta t\,S)x_{n+1} = M\,x_n + \Delta t\,M\,f(x_n)$$

with the mass matrix M and the nonsymmetric stiffness matrix S. A good candidate for treating such problems are Krylov-space methods, for instance Conjugate Gradient or better BiCGStab [25] schemes. However, since we have to solve second order PDE problems, the condition number of the matrices depends on the mesh width such that

level	elements	nodes
1	20	34
2	80	107
3	320	373
4	1280	1385
5	5120	5329
6	20480	20897
7	81920	82753

Table 1: Listing of number of nodes and cells for different mesh refinement levels

the convergence rates depend on the mesh size, too. Moreover, due to the anisotropic decomposition of the diffusion operators with very different parameter ranges for diffusion in orthogonal, resp., parallel flow direction, most iterative solvers result in problems with the convergence behaviour. This can be improved by using appropriate preconditioners and, moreover, by using hierarchical multigrid schemes whose convergence rates are much less related on the condition of $M + \Delta t\ S$ for different mesh sizes.

Currently, we have implemented a special multigrid/domain decomposition method in our new FEM package FEAST. FEAST decomposes successively the domain at the coarse grid (see Fig. 8) into smaller subdomains which are locally meshed by a generalized tensor product mesh. This regular mesh structure can be used to speed up all matrix-vector operations as well as prolongation and restriction operations between multigrid levels and fast and robust smoothers will be constructed, hereby exploiting cache and pipelining effects (see [22]).

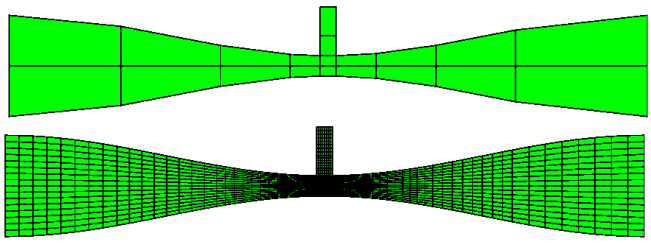


Figure 8: Coarse mesh and three times refined mesh

The following Table 2 shows exemplarily convergence results with the BiCGSTAB solver and a multigrid solver, for different time steps sizes and mesh widths. As can be seen, multigrid can perform much better, but all proposed solvers still show problems for larger time steps what is due to non-optimal preconditioners/smoothers to handle the problems with the anisotropic diffusion operator. Currently, we develop improved numerical components, which are much better suited to treat such anisotropies in a robust manner.

level	$\Delta t = 0.001$	#it	$\Delta t = 0.002$	#it	$\Delta t = 0.003$	#it
5	0.08753	6	0.08345	6	0.12957	7
6	0.08973	6	0.11818	7	0.19461	9
7	0.20600	9	0.37173	15	0.50550	21

level	$\Delta t = 0.001$	#it	$\Delta t = 0.002$	#it	$\Delta t = 0.003$	#it
5	0.00001	1	0.00008	2	0.00030	2
6	0.00003	2	0.00043	2	0.00985	3
7	0.00027	2	0.06115	5	0.60475	28

level	$\Delta t = 0.001$	#it	$\Delta t = 0.002$	#it	$\Delta t = 0.003$	#it
5	0.00001	1	0.00001	2	0.00008	2
6	0.00001	1	0.00059	2	0.00275	3
7	0.00001	1	0.00641	3	0.05587	5

Table 2: Comparing convergence rates for different refinement levels, time-steps and solvers (preconditioned (ADITRIGS) BiCG-stab (top), MG with ADITRIGS smoother, 8 smoothing steps, F-cycle (middle), preconditioned (MG/ADITRIGS) BiCG-stab (bottom). In each case the noise field was replaced with sine waves and 6 digits accuracy had to be gained.

4 Conclusions and outlooks

We have presented multiscale visualization techniques for time-dependent flow fields coming from CFD simulations on general 2D domains. The proposed models are based on PDEs with anisotropic transport diffusion operators which are linearized in time by a semi-implicit approach. The resulting problems in each time step are discretized by sophisticated streamline-diffusion FEM schemes and iterative solvers are applied to obtain accurate, robust and efficient numerical approaches on unstructured quadrilateral grids. The main features of the proposed numerical methods together with improved blending strategies and a discussion of the involved parameters has been analysed via numerical examples which are mainly coming from a CFD simulation of the 'Venturi Pipe' problem.

Currently, we are developing improved strategies for blending techniques using only those solutions which had some time to develop the desired features for the blending. Moreover, fully implicit approaches using the right hand side $f(u_{n+1})$ instead of $f(u_n)$ or using a fully nonlinear diffusion operator have to be included via fixed point defect correction schemes. At least in FEM simulations on locally refined anisotropic meshes coming from CFD simulations, such fully implicit techniques provide very robust and accurate convergence properties and have to be extended to multiscale flow visualization. Moreover, the use of the Crank-Nicholson or a related 2nd order time stepping scheme, for instance fractional-step-θ-methods (see [23]), have to be analysed which might lead to better accuracy results and hence larger time steps.

Another very important aspect is the improvement of the iterative solvers, particularly of special multigrid schemes which are able to cope with the very anisotropic differential operators and the related very ill-conditioned linear systems which have to

be efficiently solved for large time steps in implicit approaches. These fast solvers, together with improved variants of the streamline-diffusion or monotone and oscillation-free FEM-TVD techniques (see [14]) will be the key ingredients for very efficient visualization tools for complex flows which shall be applied in future 3D CFD simulations, too.

References

[1] R. Bruckschen, F. Kuester, B. Hamann, and K. I. Joy. Real-time out-of-core visualization of particle traces. In D. Breen, A. Heirich, and A. Koning, editors, *IEEE Symposium on Parallel and Large-data Visualization and Graphics (PVG 2001)*, Los Alamitos, California, 2001. IEEE Computer Society Press.

[2] D. Bürkle, T. Preusser, and M. Rumpf. Transport and anisotropic diffusion in time-dependent flow visualization. In *Proceedings Visualization '01*, 2001.

[3] B. Cabral and L. Leedom. Imaging vector fields using line integral convolution. In J. T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 263–272, Aug. 1993.

[4] F. Catté, P.-L. Lions, J.-M. Morel, and T. Coll. Image selective smoothing and edge detection by nonlinear diffusion. *SIAM J. Numer. Anal.*, 29(1):182–193, 1992.

[5] M. S. Chong, A. Perry, and B. J. Cantwell. A general classification of three-dimensional flow fields. *Phys. Fluids A*, 2(5):765–777, 1990.

[6] U. Diewald, T. Preusser, and M. Rumpf. Anisotropic diffusion in vector field visualization on euclidean domains and surfaces. *Trans. Vis. and Comp. Graphics*, 6(2):139–149, 2000.

[7] B. Heckel, G. Weber, B. Hamann, and K. I. Joy. Construction of vector field hierarchies. In *Proceedings of IEEE Visualization 1999*, pages 19–25, 1999.

[8] J. C. R. Hunt, A. A. Wray, and P. Moin. Eddies, stream and convergence zones in turbulent flow fields. Technical Report CTR-S88, Center for turbulence research, 1988.

[9] V. Interrante and C. Grosch. Stragegies for effectively visualizing 3D flow with volume LIC. In *Proceedings Visualization '97*, pages 285–292, 1997.

[10] J. Jeong and F. Hussain. On the identification of a vortex. *Journal of Fluid Mechanics*, 285:69–94, 1995.

[11] B. Jobard, G. Erlebacher, and Y. M. Hussaini. Hardware accelerated texture advection for unsteady flow visualization. In *Proceedings of IEEE Visualization 2000*, Salt Lake City, Utah, October 2000.

[12] B. Jobard, G. Erlebacher, and Y. M. Hussaini. Lagrangian-eulerian advection for unsteady flow visualization. In *Proceedings of IEEE Visualization 2001*, San Diego, California, October 2001.

[13] B. Kawohl and N. Kutev. Maximum and comparison principle for one-dimensional anisotropic diffusion. *Math. Ann.*, 311 (1):107–123, 1998.

[14] Kuzmin, D. and Turek, S. High-Resolution FEM-TVD Schemes Based on a Fully Multidimensional Flux Limiter. *J. Comput. Phys.*, (198):131–158, 2004.

[15] Matthies, G. and Tobiska, L. The Streamline-Diffusion Method for Conforming and Nonconforming Finite Elements of Lowest Order Applied to Convection-Diffusion Problems. *Computing*, (66):343–364, 2001.

[16] P. Perona and J. Malik. Scale space and edge detection using anisotropic diffusion. In *IEEE Computer Society Workshop on Computer Vision*, 1987.

[17] Pironneau, O. On the transport-diffusion algorithm and its applications to the Navier-Stokes equations. *Numer. Math.*, (38):309–332, 1982.

[18] T. Preusser and M. Rumpf. An adaptive finite element method for large scale image processing. *Journal of Visual Comm. and Image Repres.*, 11:183–195, 2000.

[19] H.-W. Shen and D. L. Kao. Uflic: A line integral convolution algorithm for visualizing unsteady flows. In *Proceedings Visualization '97*, pages 317–322, 1997.

[20] M. Tobak and P. D. J. Topology of 3D separated flow. *Ann. Rev. Fluid Mech.*, 14:61–85, 1982.

[21] S. Turek. *Efficient Solvers for Incompressible Flow Problems: An Algorithmic and Computational Approach*, volume 6 of *LNCSE*. Springer, 1999.

[22] Turek, S., Becker, Ch., and Kilian, S. Consequences of modern hardware design for numerical simulations and their realization in FEAST. In P. Amesto, P. Berger, M. Dayde, I. Duff, V. Fraysse, L. Giraud, D. Ruiz, editor, *Proceedings Euro-Par'99 Parallel Processing*, 1999. Toulouse, France, August/September 1999.

[23] Turek, S., Rivkind, L., Hron, J., and Glowinski, R. Numerical analysis of a new time-stepping θ-scheme for incompressible flow simulations. Preprint, 2005. To appear in JSC.

[24] G. Turk and D. Banks. Image-guided streamline placement. In *Proc. 23rd annual conference on Computer graphics, August 4 - 9, 1996, New Orleans, LA USA*. ACM Press, 1996.

[25] van der Vorst, H. A. Bi-CGStab: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comp.*, 13(2):631–644, 1992.

[26] J. J. van Wijk. Spot noise-texture synthesis for data visualization. In T. W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 309–318, July 1991.

[27] J. J. van Wijk. Flow visualization with surface particles. *IEEE Computer Graphics and Applications*, 13(4):18–24, July 1993.

[28] J. J. van Wijk. Image based flow visualization. In *ACM Transactions on Graphics, special issue, Proceedings ACM SIGGRAPH 2002*, San Antonio, Texas, 2002.

[29] 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 *Workshop on Vision, Modeling, and Visualization VMV '01*, pages 439–446. infix, 2001.

[30] R. Westermann, C. Johnson, and T. Ertl. A level-set method for flow visualization. In *Proceedings of IEEE Visualization 2000*, pages 147–154, 2000.