INTERACTIVE VISUALIZATION TECHNIQUES FOR A VIRTUAL REALITY BASED ANALSYIS OF SIMULATION RESULTS

Paul Benölken, Holger Graf, Joachim Rix
Fraunhofer Institute for Computer Graphics

Rundeturmstr. 6
64283 Darmstadt, Germany
paul.benoelken, holger.graf, joachim.rix@igd.fhg.de,

ABSTRACT

In this paper we present an overview of existing state of the art visualization techniques for the interactive analysis of results from numerical simulations and measurements. We describe the basic concepts and key ideas behind these different visualization methods in this paper. The potential of these techniques for an efficient integration into a virtual reality environment will be investigated. Furthermore we present our first demonstrator for visualizing multiparametric data and give an outlook on our plans for further exploiting and developing these techniques in an upcoming project.

KEYWORDS: interactive visualization, direct volume rendering, LIC, virtual environments, multi parametric visualization

1 Introduction

During the past decade numerical simulations have become more and more important for predicting different properties of new products as well as of various kinds of natural phenomena. Many applications of such simulations are found in different natural sciences as well as in the engineering domain (CAE), where the development of new prototypes is based on numerical models. Due to the increasing availability of high performance computing hardware more complex and realistic simulations can be run in shorter time periods. Moreover, the results of these computations have become very close to real world tests and measurements so that the need for creating hardware prototypes can be reduced. Finally, as the testing of physical prototypes is still indispensable, these tests can be prepared and accomplished in a more efficient way by simulating this process (Computer Aided Testing, CAT).

As such computations and measurements mostly result in large sets of scalar or vector data appropriate visualization tools are required for a proper analysis, interpretation and presentation.

In this paper we will give an overview of different states of the art visualization techniques for the interactive exploration and analysis of simulation results. Methods for visualizing scalar and vector fields using direct volume rendering and line integral convolution will be presented and compared with more conventional methods as isosurfaces, vector glyphs, particle traces or streamlines. Furthermore we will show some interaction paradigm based on virtual reality technologies and discuss their potential for an intuitive and comprehensive exploration of complex simulation results. Finally we will present our first demonstrator for visualising multi-parametric data sets and give an outlook on our future work.

2 Scalar Field Visualization

Scalar fields in 3D space (volume data) occur in many different application fields as simulation results as well as in meteorology, geology and medical applications. In general, the following steps are required for visualizing volume data:

- Data preparation (conversion, re-sampling, completition, data reduction)
- Mapping (classification, scalar value – color/opacity assignment, surface extraction)
- Shading (choice of shading model and parameters)
- Rendering (mapping graphical primitives onto the screen)

Different strategies are used for visualizing volume data which are divided into decomposition methods, surface extraction and direct volume rendering techniques.

2.1 Decomposition

Decomposition techniques subdivide the volume data set into points, slices or make use of volumetric primitives as cubes or spheres where scalar data are mapped onto colors and opacities. The color mapping is implemented by using indexed lookup tables. Cutting planes where a predefined portion of the data set is extracted and displayed using color mappings are very frequently used. Combined with an interactive positioning, this technique allows a quite intuitive interpretation of the data set.

2.2 Surface Extraction

Surface extraction methods, which are referred as volume modeling, approximate a surface contained within the volume data with polygonal primitives. A surface is defined by applying a segmentation function to the volume data. Based on a segmentation criterion, parts of the volume data are marked as part of a region or not. The surface is the region where the segmentation function changes its value (e.g. from 0 to 1). Typical representatives of such surfaces are isosurfaces where a predefined threshold value (iso value) is used for subdividing the data set. Here the surface represents all positions of a specific value in the data set so that topological and geometrical properties of value distributions can be investigated. The most important method for generating isosurfaces is the Marching Cubes algorithm [25].

Both isosurfaces as well as cutting planes can be implemented and rendered very efficiently by using conventional graphics hardware accelerators. For this reason they are very frequently used together with color mappings for an interactive visualization of volume data. Nevertheless both methods lack in their ability in giving a complete overview of the whole data set structure. In the case of isosurfaces, only the data values specified with the iso-value are shown. The same is true for cutting planes where only the data values within the specified data slide can be seen. Furthermore, finding relevant iso-values is not trivial and requires a deeper knowledge of the data set.

2.3 Direct Volume Rendering

During the past decade direct volume rendering has become a very popular and powerful technique for visualizing 3D scalar data. The direct mapping of semitransparent voxel data onto the projection plane allows the visualization of overlapping structures so that more information can be obtained from a single picture. One drawback of this technique has always been the insufficient rendering time. Therefore several approaches have been published for achieving better performance results ([18], [19], [23]). The most promising techniques for obtaining interactive frame rates are hardware supported 3D or 2D texture mapping ([21], [20, [16], [5]) and cell projection techniques ([24], [2], [11]). In the following subsection we will shortly present and discuss these methods.

2.3.1 3D Texture Mapping

The idea behind the 3D texture mapping approach is to interpret the voxel data as a 3D texture. At the beginning of this algorithm, introduced by [24] the voxel data is loaded into the texture memory. Afterwards parallel polygons are computed, slicing the volume orthogonal to the viewers direction. Then texture values are assigned to each polygon point by trilinear interpolation. Finally the polygons are drawn back to front into the frame buffer using the OpenGL blending functions. The algorithm exploits the OpenGL texture mapping and trilinear interpolation supported by the 3D texture hardware of today's graphic cards. In the case that only 2D textures are supported by the hardware the algorithm is modified as follows:

Instead of a single polygon set, three sets of polygonal slices (one for each principal axis) are computed in object space. Afterwards a bilinear interpolation is used for mapping the texture data onto the polygons. Then the polygons are transformed with the current model view matrix and rendered front to back where the polygon set which is closest to the current viewing direction is used.

Volume rendering based on texture mapping strongly depends on the texturing and rasterization ability of the graphics hardware. One problem is that the image quality is related to the number of volume slices and thus to the number of polygons which are drawn and alpha blended into the frame buffer what in turn influences the overall performance. Furthermore due to the limited texture memory larger data sets have to be split up into several volume bricks which have to be sorted and rendered back to front for achieving correct images. This leads to the effect that volume data has to be re-loaded from the main-memory into the texture buffer which again reduces the performance. Finally, texture based volume rendering is mainly useful for regular grid structures. However, many numerical simulation results are defined on structured- or irregular grids. One approach for overcoming this limitation is to resample or scan convert the data onto a regular grid. The problem with this procedure is that a much higher number of cells is needed for presenting the same information otherwise artifacts in the final image are unavoidable. Figures 2 and 3 show some examples for texure based volume rendering.

2.3.2 Projected Tetrahedra

With their Projected Tetrahedra (PT) algorithm Shirley and Tuchman [24] introduced a new method which accelerates dramatically the rendering of unstructured tetrahedral grids. The algorithm visualizes the volume data defined at each vertex of a tetrahedral grid by decomposing the projected profile of each tetrahedron into triangles and finally renders these partially transparent triangles which is done very quickly by today's graphics hardware. In the first stage of the original PT algorithm all cells are sorted according to their visibility. Next each tetrahedron is classified according to its projection in image space by testing the plane equations of each side and splitted into triangles. Afterwards colors and opacities are computed for each triangle vertex. Finally the triangles are rendered by using the graphics hardware. Although this algorithm considerably improves the rendering performance it has two critical parts:
One handicap is the visibility ordering of the tetrahedrons which has to be performed for each change of the viewing direction. Therefore several algorithms for a fast visibility ordering [31][32][33] have been published. Nevertheless, as these methods still result in a considerable load of the host CPU further approaches have been made for new hardware supported visibility ordering algorithms [29][30]. Another drawback of this algorithm has been the linear approximation of colors and opacity between vertices which sometimes leads to artifacts. For this reason some improvements based on 2D and 3D texture mappings have been published in [28] and [2] for a more accurate rendering of colors and opacities.
Consequently direct volume rendering based on the PT algorithm and its further improvements seems to be a very efficient and powerful technique for the interactive visualization of tetrahedral grids. Other grid structures can be converted into tetrahedral meshes which makes this technique suitable for a VR based exploration of volume data sets resulting from numerical simulations.

2.3.3 VR Integration

The integration of direct volume rendering into a virtual reality environment has been subject to several research activities. A distributed approach based on the Lacroute's VolPack implementation of the shear warp factorisation [18] was published in [7]. This implementation allowed a collaborative visualization of pre-classified volume data sets in distributed VR- and desktop environments while the classification was done in a preprocessing step. However, the VolPack library only supported orthogonal projections so that usage was limited to VR environments with a single projection wall. Furthermore the library does not support the integration of polygonal data which is required in many applications for getting a complete understanding of the whole data set.

Another approach for integrating direct volume rendering into VR environments is described in [1]. This implementation is based on 3D Texture mapping and supports VR based interaction mechanisms which allows the interactive assignment of color and opacity transfer functions via 3D floating menus. The floating menus are activated and manipulated by a 3D "laser" beam which is casted from the tracked mouse position into the pointed direction.

A solution based on the Studierstube environment is reported by Wohlfarter et al. in [3]. Studierstube is an augmented reality framework based on OpenInventor. It supports two-handed interaction by using a tracked personal interaction panel (PIP) in the one hand and a tracked pen in the other. The interaction panel is a transparent palette onto which virtual menus and controls are displayed. The pen is used to point to objects and for manipulating menus and objects. The visualization of volume data is based on the OpenGL Volumizer API which has been integrated into the Studierstube environment.

The applications described in [1] and [3] support a complete and intuitive investigation of volume data in a VR environment by using different interaction paradigm. Nevertheless, as both are based on 3D Texture mapping they are more or less restricted to volume data defined on regular grid structures. Anyhow, as our aim is the interactive exploration of volume data resulting from numerical simulation which is typically defined on structured or irregular meshes, we suggest the integration of the above described PT algorithm into the Studierstube environment.

3 Visualizing 3D vector fields

The analysis of three dimensional vector fields is subject to many applications and research fields including the analysis of fluid flows in gases and liquids as well as the exploration of electromagnetic phenomena. 3D vector fields result from numerical simulations as in the CFD domain, from experiments, or real world measurements. For visualizing vector fields, the same steps as described in chapter 2 are required: Data preparation, mapping and rendering. The data preparation step includes filtering of noise resulting from measurements as well as data reduction and selection for handling large data sets as they usually result from high performance computations.

3.1 Common Techniques

One of the most popular elementary method for visualizing vector fields are vector glyphs. These graphical primitives which usually are shaped as arrows are commonly used for visualizing the magnitude and direction of vectors at selected points in a vector field. In many applications arrow glyphs are combined with color mappings for visualizing additional information or cutting planes for eliminating ambiguities. Very frequently vector glyphs are built from 3D primitives as cones and cylinders which are shaded for enhancing the 3D perception.

Streamlines, pathlines and streaklines are used for visualizing the path of single particles in a flow field. In general streamlines are defined as the tangential curves of the vector field at a fixed time whereas pathlines describe the movement of a particle over the time in a stationary vector field. The positions of all particles which have passed a fixed point at different times is described by streaklines. The line computations are done using numerical integration algorithms as the 4th order Runge-Kutta method.

Stream surfaces can be seen as a 3D extension to streamlines where adjacent streamlines are joined together to form a surface so that each point on such a surface is tangential to the specified flow field. Another method for visualizing flow fields are stream objects which are inserted into the flow field at fixed time intervals on a predefined starting position. Over the time the stream objects change their size and shape according to the flow characteristics.

The above mentioned methods allow the visualization of different properties at selected points of a vector field. However, for avoiding occlusions and ambiguities their usage has to be limited on a quite small portion of the vector field. In [27] stream arrows are suggested for overcoming the occlusion problem. This method maps transparent arrow-shaped textures onto stream surfaces so that both the flow directions as well that what lies behind the stream surface can be shown. Nevertheless, for avoiding confusing images the number of such stream arrows still has to be kept small. Two examples of flow visualizations based on vector glyphs, streamlines, cuttingplanes and isosurfaces are shown in figure 4 and 5.

3.2 Exploring Local Properties

The inspection of local properties as for example vortices in turbulent flows is another important topic in the area of vector field visualization. A well known method in this field is the visualization of critical points. Critical points are locations in a vector field where the velocity becomes the zero vector. They are uniquely identified and classified by computing the eigenvalues of the jacobian matrices. Based on this classification critical points can be displayed by using different icons which show their characteristics (saddle point, focus, center, node, etc.). Another method is to examine the jacobian matrix for a given number of samples around the critical point. The decomposition of the matrix into symmetric and asymmetric parts gives the local shear and rotational information which are mapped onto geometrical properties of polygons so that their size and shape illustrate the local flow properties. A further method for showing local vector field properties are local flow probes. A special probe was presented by [34] which is also derived from the jacobian matrix. Here flow field properties are presented by mapping direction, velocity, acceleration, curvature, rotation, shear, and convergence/divergence to different geometric parts of some complex glyphs which are placed at pre-selected locations in the vector field.

In addition to these methods, flow ribbons are very frequently used for visualizing 3D flows. Flow ribbons are generated, by connecting adjacent pathlines, which start from a particular point in the flow field, via triangulation.

3.3 Visualizing Global Structures

The above described methods display vector field properties at particular positions whereas an overview of the entire vector field structure can be obtained by using global methods. In this context streamlines play an important role as each point in the vector field is traversed by exactly one streamline. However, the display of streamlines is restricted to a limited number of positions within a flow field. Otherwise, this method would result into confusing and meaningless images without any identifiable lines and structures. A main feature of global visualization techniques is therefore to display all lines in such a way that the relevant structures of a vector field are shown. In the following we will describe two popular techniques for visualizing global vector field characteristics.

The Line Integral Convolution (LIC) method was introduced by [17] and is a powerful and effective technique for visualizing the global structure of vector fields. It allows the imaging of the whole vector field displaying at once the detailed characteristics of the field. This is done by convoluting a white noise input texture with the vector field. Figure 7 shows the result of a LIC based visualization of a circular vector field. An example for mapping a LIC texture onto a surface is shown in figure 6.

Over the last years a number of extension and improvements of the basic LIC technique were made. Algorithms have been published for a fast LIC (FLIC) [13] as well as extensions that allow mapping of flat LIC images onto curvilinear surfaces in 3D space [9]. Many other enhancements were made to achieve real time visualization for applying LIC to unsteady flows, for handling time-varying data and for using LIC to illustrate the shape of arbitrary surfaces[14].

With the Integrate & Draw (ID) method [26] the concept of LIC images is further developed. But instead of using noise textures a randomly selected gray value is assigned to each streamline. Streamline integration as well as the selection of starting points is done in the same way as in [13] whereas the calculation of mean values for assigning values to multiple chosen pixels has been modified so that the blurring effects of the FLIC algorithm are avoided. With the ID method images with more contrast can be produced. Furthermore this algorithm is computational faster than the FLIC method. The time saving for computing an output image is in general about 50%.

3.4 VR based Exploration

Over the last few years the interactive exploration of 3D vector fields in a virtual reality environment has been the subject of several research projects and developments. Based on the upcoming VR technology various applications have been created for the interactive visualization of 3D flow data. Commonly used are the methods described in section 3.1. Here stereoscopic presentations of vector glyphs, path-, stream-, streamlines or surfaces as well as combinations of these techniques are used to support the inspection of 3D vector fields. Very popular are particle traces where the path of geometrical primitives along a trajectory is visualized so that the impression of inserting material into the flow is given. Nevertheless, as the computation of such trajectories is mainly based on numerical solutions of differential equations they are in general very time consuming. Therefore such computations are either performed on massive parallel high performance clusters or shifted into a pre-processing step which in turn limits the interactive selection of starting points. Furthermore as these methods tend to produce complex presentations with overlapping geometry they are restricted to rather coarse spatial resolution.

In order to overcome these restrictions some further approaches have been investigated. An implementation of a 3D Line Integral Convolution has been presented in [5]. This 3D LIC method is based on 3D Texture mapping. Beside different clipping mechanism an interactive assignment of colors and opacities is used for visualizing the internal flow structure. For improving the flow perceptions 3D LIC animations are generated. In order to overcome the occlusion problem Fuhrmann et. al.[25] suggest the usage of dashtubes for 3D flow visualization. In general, dashtubes are streamlines whose cylindrical shape is extruded along the flow direction. For reducing the occlusion and showing the flow direction their geometry is displayed with an animated, opacity mapping texturing. This application has been integrated into the Studierstube environment. Magic lenses and magic boxes are applied as interacting techniques for inspecting dense areas of flow and for focussing on regions of interest.

4 Visualization of Multiparametric Data

Another important issue is the visualization of multiparametric data sets where each value in a data set is identified by an n-tuple of independent variables, with $n \geq 3$. In many publications, the terms multiparametric and multidimensional are used synonymously. For clarification, as multiparametric data sets we define collections of scalar values which have a uniquely defined position in a multidimensional observation space. Multiparametric data set occur in different measurements or numerical simulation where a physical entity is modeled by a function with n input parameters. The aim is now to visualize such data sets a in the context of a maximal number of parameter values for getting a more efficient and intuitive presentation of correlation's and other characteristics as maxima and minima.

One problem in visualizing such data sets is that in computer graphics we are limited to 3 dimension, even if we are using stereo viewing techniques. Another problem lies in the human ability to understand more than 3 dimensions. Various methods have been developed in the past to present multiparametric data and to overcome these restrictions.

One method for visualizing multiparametric data sets is given by mapping data values onto small graphic primitives, which are referred as glyphs or icons. Very commonly glyphs are used for assembling several parameter values into a single presentation. In general, the different data values are coded into the glyph position, size, shape, color or texture. Examples for using glyphs are given in [35] and [34]. The usage of such glyphs or icons is strongly related to the specific application area. Therefore visualization tools should offer methods for controlling the appearance and the data mapping of glyphs. For this reason an architecture of a visualization environment has been suggested in [15], that supports a user controlled creation and usage of icons.

Another representation method is given by using parallel coordinates. Parallel coordinates are defined by a parallel and equidistant assembly of coordinate axes, which represent the value ranges of the different parameters. In this approach, points of the n-dimensional parameter space are mapped onto line segments which connect the different parameter values at the coordinate axes. Extruded parallel coordinates and 3D parallel coordinates are extensions to this method which have been published in [8]. Extruded parallel coordinates are build, by first creating a planar parallel coordinate system. Afterwards the coordinate system is moved along a trajectory, which defines a spatial axis. Each point of this axis is traversed by exactly one line segment, so that different data sets with the same parameter settings can be displayed. 3D parallel coordinates are used by replacing the single coordinate axes by parallel planes, where each plane is defined by two parameter axes. These planes can be placed arbitrarily in 3D space. A further possibility is to combine and link these planes with polylines or polygon surfaces for showing dependencies of selected data values. By using this method, the number of simultaneous visualized parameters, could be efficiently increased.

4.1 The C.A.T.Viewer

The C.A.T.Viewer is our first demonstrator which has been implemented within the BMBF project iViP [4] for analyzing multiparametric data sets. The aim of this implementation was to build a graphical analysis tool for supporting the data exploration in the domain of Computer Aided Testing, CAT. This application is based on a modular structure which allow the extension of the existing functionality by integrating new code as modules. A visualization pipeline similar to the one described in [36] can be build up by connecting the modules via selected ports. After starting the pipeline execution the results are visualized in the viewer window. The principle layout of the C.A.T.Viewer pipeline is shown in Figure 1.

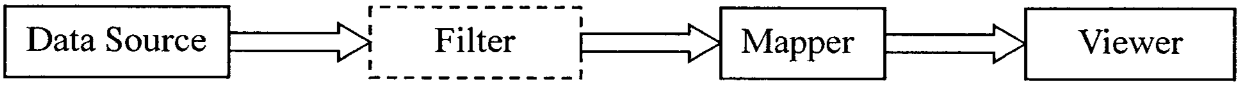

Fig. 1 The C.A.T.Viewer Visualization Pipeline

As a data source several modules are available for loading data sets as well as for performing calculations e.g. the numerical evaluation of multidimensional functional models at discrete grid point. We have implemented some modules for evaluating different functional models on 3D regular grids. The evaluation is done by arbitrarily mapping three of n parameters onto the three principal coordinate axes whereas the remaining n-3 parameters can be manipulated by the user interface. In a further release, we will extend this procedure by integrating animation functionality's for varying different parameter setting. After these computations the grid data is transferred via selected ports to various mapper-modules where the data is mapped onto geometric structures as glyphs, isosurfaces, cutting- and contourplanes. Further modules exist for performing color mappings and data probing. Figure 9 shows the visualization of a regular grid resulting from the evaluation of a functional model with 7 parameters.

A filtering process is required for visualizing (multiparametric) data sets from measurements. Such data sets frequently contain noise or measurement errors. In many cases a deeper knowledge of the original measurement process is required for obtaining interpretable subsets of the whole measurement data. If the measurement was not performed over the whole parameter range, interpolations have to be performed for further processing. Based on these considerations we implemented a module for filtering and visualizing measurement data from a motor testbed. The results are shown in Figure 8. For the future we plan to integrate some more advanced visualization techniques as 3D parallel coordinates. Furthermore, we intend to extend the C.A.T.Viewer with VR interaction techniques based on the Studierstube framework.

5 Outlook

For the analysis and optimization of digital models different steps as data preparation, simulation and evaluation have to be performed. However in today's CAE systems, these steps are accomplished within different time slots allowing no interaction with the simulation and no visual real-time feedback from the simulation engine.

VR as an user centered man machine interface is currently only used as a visualization tool during the evaluation stage. This by no means makes full use of the powerful benefits of this technology. The provision of 3D-display technology, interaction mechanisms and real-time visualization would efficiently support a solution approach to the problems of data preparation and finally efficient result evaluation. VR might enable the user to steer through the simulation results and directly interact with the data sets and results.

The aim of the EU funded project ViSiCADE is therefore to efficiently integrate existing VR-technology like virtual table, interaction- and visualization components into a platform that enables a seamless integration of CAE simulation tasks into VR. As a new man-machine interface this will be an efficient environment that enables the engineer to set up and control FEM based simulations in a more convenient way and finally evaluate and interact with the complex 3D output data.

An appropriate link between interaction mechanism and simulation engine will offer the capability to interactively move and modify the boundary conditions with a related adaptation of the analysis model for the simulation. On the basis of a hierarchical approach we will be able to display the results of the simulation engine while moving the boundary conditions in real-time. Hence, rapid preliminary design optimisation will be conducted on simpler analysis models in real-time. These will be coupled to the 3D geometry used in detailed modelling. Our hierarchical approach will start an analysis from coarse models for a preliminary design evaluation and result in local models that enable the designer to clarify design details as required.

Within this projects we intend to explore and further develop the visualization methods described in the previous sections for displaying the results and enhancing the evaluation stage. To allow an immediate visual feedback it is envisaged to integrate and further develop direct volume rendering techniques based on the PT algorithm. This method seems to be a good candidate for an interactive visualization of structured and irregular grids. Further investigations and developments are planned for a VR based exploration of flow data. Promising techniques in this area are methods like 3D LIC which may be combined with the principles and interaction paradigms presented with the dashtube visualization. For achieving real time aspects, the capabilities of today's programmable graphics boards will be exploited so that the visualization is mainly performed by the graphics hardware whereas the CPU time is left for the simulation and interaction. Finally, based on the Studierstube framework we will develop new interaction mechanisms for interactively assigning and manipulating boundary conditions as well as for steering the simulation.

Images

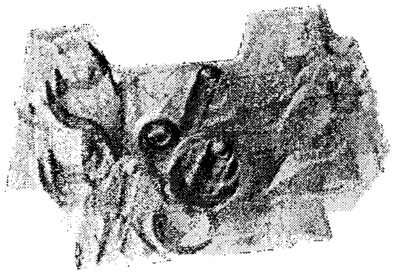

Fig. 3: Direct volume rendering
a CT-scan of a motor block

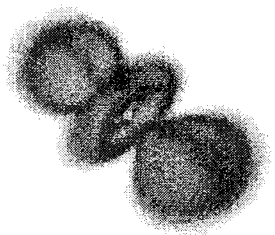

Fig. 2: Visualizing the probability of presence of a
hydrogen atom with direct volume rendering

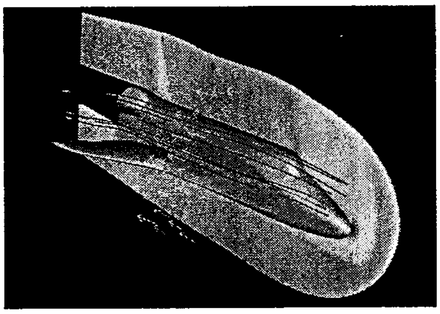

Fig. 5: Visualization of the flow around the space
shuttle with streamlines, cuttingplane and
arrow glyphs.

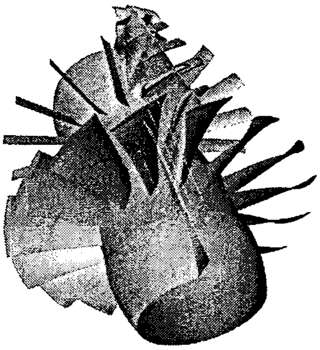

Fig. 4: Flow visualization in a turbine
with cuttingplanes and isosurfaces

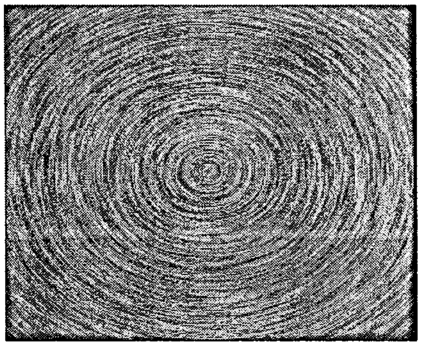

Fig. 6: Visualizing a circular vector field
with LIC

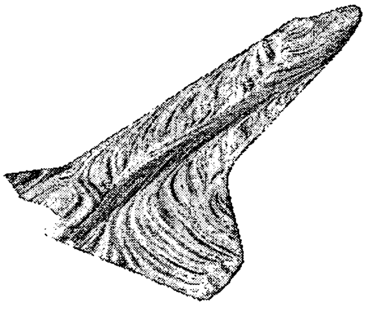

Fig. 7: Mapping of a LIC texture onto
a surface

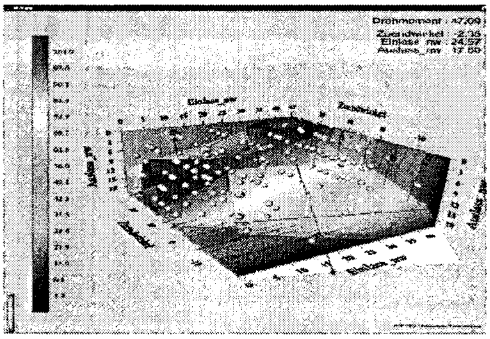

Fig. 9: Evaluation of a function with 7 parameters
on a regular grid using glyphs, isosurfaces,
cuttingplanes, conturlines and data probes
for visualization.

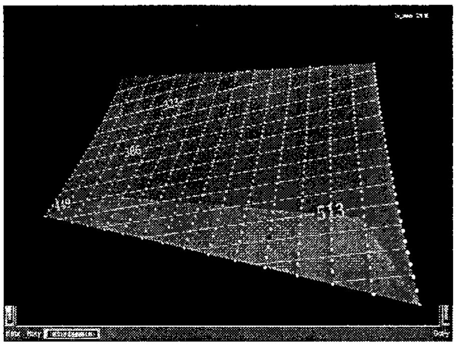

Fig. 8: Heightfield visualization of a multiparametric
data set from a motor testbed with contourlines
and glyphs

REFERENCES

1. Schulze J.P, Woessner U, Walz S.P, Lang U (2001) Volume Rendering in a Virtual Environment, Proceedings of 5th IPTW and Eurographics Virtual Environments, Springer Verlag, 2001.
2. Roettger S, Kraus M, Ertl T (2000) Hardware-Accelerated Volume and Isosurface Rendering Based On Cell-Projection, Proceedings of IEEE Visualization '00. pp. 109-116.
3. Wohlfahrter W, Encarnacao L.M, Schmalstieg D(2000) Interactive Volume Exploration on the StudyDesk, Proceedings of the IPTW 2000
4. Benölken P, Maurer H, Ott T (2000) iViP subproject 4.4 - virtual test bench. ProductData Journal Vol.7 (2000), Nr. 2, pp. 38-41.
5. Rezk--Salama C, Hastreiter P, Teitzel C, Ertl, T (1999) Interactive Exploration of Volume Line Integral Convolution Based on 3D-Texture Mapping, Proc. Visualization '99,pp. 233-240.
5. Westermann R, Ertl T (1998) Efficiently Using Graphics Hardware in Volume Rendering Applications. Computer Graphics vol 32, pp. 169-179.
7. Benölken P, Niemeier R, Lang U (1998) Collaborative Volume Rendering in a Distributed Virtual Reality Environment. In: Eurographics Workshop Proceeding Series , pp. 47-1 -47-10, Stuttgart, 16.-22. June 1998.
8. Wegenkittl, R.; Löffelmann, H.; Gröller, E. (1997) „Visualizing the Behavior of Higher Dimensional Dynamical Systems". Proceedings Visualition '97, IEEE Computer Society Press, Los Alamitos, pp. 119-125.

9. MAO,X.,KIKUKAWA,M.,FJUITA,N.,AND IMAMIYA, A. (1997) Line Integral convolution for Arbi-trary 3D Surfaces through Solid Texturing. In in Proceedings of Eighth Eurographics Workshop on Visualization in Scientific Computing.
10. Wong, P. C.; Bergeron, R. D. (1997) 30 Years of Multidi-mensional Multivariate Visualization. in: Nielson, G.M.; Hagen, H.; Müller, H.: Scientific Visualizati-on. IEEE Computer Society Press, Los Alamitos, pp. 3-33.
11. Westermann R, Ertl T (1997) Rendering and Re-Sampling Unstructured Volume Data by Polygon Drawing. Technical Report 17/1997. Universität Erlangen-Nürnberg.
12. Cabral B, Leedom C (1995) Highly parallel vector visualization using line integral convolution. In Proceedings of the Seventh Siam Conference On Parallel Processing for Scientific Computing, volume 7, pages 803-807.
13. Stalling D, Hege H (1995) Fast and Resolution Independent Line Integral Convolution. Proceedings Siggraph '95, Los Angeles, pp. 249-256, 1995.
14. FORSSEL,L.K.,AND S., C. Using Line Integral Convolution for Flow Visualization: Curvilinear Grids, Variable Speed Animation, and Unsteady Flows. IEEE Transactions on Visualization and Computer Graphics 1, 2 (June 1995).
15. Foley J, Ribarsky B (1994) Next-generation Data Visualization Tools. In Scientific Visualization, Advances and Challenges, chapter 7, pages 103-127.
16. Wilson O, Van Gelder A, Wilhelms J (1994) Direct Volume Rendering via 3D Texures. Technical Report UCSC-CRL-94-19. University of California, Santa Cruz.
17. Cabral B, Cam N, Foran J (1994) Accelerated Volume Rendering and Tomographic Reconstruction with 3D Texture Hardware. In ACM Symposium on Volume Visualization '94. pp. 91 – 98
18. Lacroute, P.G, Levoy M (1994) Fast volume rendering using a shear warp factorisation of the viewing transformation. Proceedings SIGGRAPH. pp. 451-458.
19. Laur D, Hanrahan P (1993) Hirarchical Splatting : A Progressive Refinement Algorithm for Volume Rendering. In ACM Computer Graphics, Proc SIGGRAPH '93, pp. 285-288.
20. Cullip T.J, Neumann U (1993) Accelerated Volume Reconstruction with 3D Texture Hardware. Technical Report TR93-027. University of North Carolina, Chapel Hill N.C.
21. Cabral B, Leedom L.(1993) Imaging vector fields using Line Integral Convolution. Proceedings Siggraph '93, pp. 263-272, Computer Graphics 27, 1993.
22. van Wijk, J. J., van Liere, R. D. (1993) Hyperslice. Pro-ceedings Visualization '93, IEEE Computer Society Press, Los Alamitos, pp. 119-125.
23. Danskin J. Hanrahan P (1992) Fast Algorithms for Volume Ray Casting. In ACM Workshop on Volume Visualization '92. pp. 91-98
24. Shirley P, Tuchman A (1990). A Polygonal Approximation to Direct Scalar Volume Rendering. San Diego Workshop on Volume Visualization, Computer Graphics, 24(5): pp. 63–70.
25. Fuhrmann A, Groeller E (1998) Real-Time Techniques For 3D Flow Visualization. Proceedings of IEEE Visualization.
26. Perez Riscet, C. (1998) Visualizing 2D Flows: Integrate and Draw. Proceedings 9. Eurographics Workshop on Visualization in Scientific Computing. pp. 53-67.
27. Löffelmann H, Mroz L, Gröller E, and Purgathofer W (1997) Stream arrows: enhancing the use of stream surfaces for the visualization of dynamical systems. Published in The Visual Computer 3(8), pp. 359-369.
28. Engel K, Kraus M , Ertl T (2001) High-Quality Pre-Integrated Volume Rendering Using Hardware- Accelerated Pixel Shading. In Eurographics / SIGGRAPH Workshop on Graphics Hardware '01 , Annual Conference Series, page 9. Addison-Wesley Publishing Company, Inc.
29. Krishnan S, Silva C, Wie B (2001) A Hardware-Assisted Visibility-Ordering Algorithm With Applications to Volume Rendering to appear at Joint Eurographics-IEEE TCVG Symposium on Visualization VisSym01, Ascona, Switzerland.
30. King D, Wittenbrink C, Wolters H, (2001) An Architecture for Interactive Tetrahedral Volume Rendering Volume Graphics Workshop 2001, Stony Brook, NY, USA.
31. Williams P, (1992) Visibility ordering meshed polyhedra. ACM Transaction on Graphics,11(2):103-126.

32. Silva C, Mitchell J, Williams P, (1998) An interactive time visibility ordering algorithm for polyhedral cell complexes. In ACM / IEEE Symposium on Volume Visualization, pp. 15-22. San Francisco, CA.
33. Comba J, Klosowski J, Max N, Mitchell J, Silva C, Williams P (1999) Fast polyhedral cell sorting for interactive rendering of unstructured grids. Computer Graphics Forum, 18(3):369-376.
34. de Leew, W.L, van Wijk, J,J, (1993) A Probe for Local Field Visualization. Proceedings Visualization '93, IEEE Computer Society Press, pp. 39-45, Los Alamitos.
35. Kerlick G. D, (1990) Moving Iconic Objects in Scientific Visualization, IEEE Visualization '90 Proceedings, pp 124-129.
36. Schroeder W, Martin K, Lorensen B, (1998) The Visualization Toolkit. 2nd Edition. Prentice-Hall Inc. ISBN 0-13-954694-4.