IndeGS: Index Supported Graphics Data Server for CFD Data Postprocessing

Christoph Brochhaus, Thomas Seidl
Data Management and Data Exploration Group
RWTH Aachen University, Germany
{brochhaus,seidl}@informatik.rwth-aachen.de

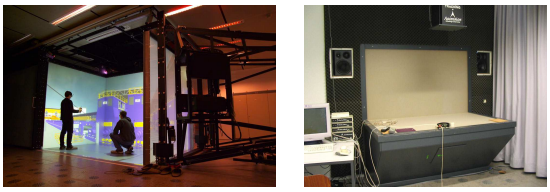
ABSTRACT

Virtual reality techniques particularly in the field of CFD (computational fluid dynamics) are of growing importance due to their ability to offer comfortable means to interactively explore 3D data sets. The growing accuracy of the simulations brings modern main memory based visualization frameworks to their limits, inducing a limitation on CFD data sizes and an increase in query response times, which are obliged to be very low for efficient interactive exploration. We therefore developed "IndeGS", the **ind**ex supported **g**raphics data **s**erver, to offer efficient dynamic view dependent query processing on secondary storage indexes organized by "IndeGS" offering a high degree of interactivity and mobility in VR environments in the context of CFD postprocessing on arbitrarily sized data sets.

Our demonstration setup presents "IndeGS" as an independent network component which can be addressed by arbitrary VR visualization hardware ranging from complex setups (e.g. CAVE, HoloBench) over standard PCs to mobile devices (e.g. PDAs). Our demonstration includes a 2D visualization prototype and a comfortable user interface to simulate view dependent CFD postprocessing performed by an interactive user freely roaming a fully immersive VR environment. Hereby, the effects of the use of different distance functions and query strategies integrated into "IndeGS" are visualized in a comprehensible way.

1. INTRODUCTION

In recent years, numerical simulations in the area of fluid dynamics offer a very high level of accuracy and reproducibility and replace tedious and expensive experiments which often strongly depend on environmental conditions. Simulations are an acknowledged method, both in industrial development and in research, in the domains of e.g. physics, automotive engineering and analysis of aerodynamic forces. These methods are generally described as *computational fluid dynamics* (CFD). During a post-processing step certain features and results of the CFD data sets are extracted out of the simulation results by request of an interactive user, who usually is an expert in the field of CFD. The results are commonly visualized in virtual reality environments, e.g. the *HoloBench*, a stereo projection table composed of two right-angled projection surfaces, or two- to six-sided rear projection systems called *CAVE* (cf. figure 1), or even small mobile devices like PDAs. They offer a high degree of interactivity by letting users immerse into the visualized objects. Common post-processing tasks include isosurface extraction ("display regions with a temperature of exactly 125°C") or particle tracing ("display the path of object located at starting position (x_0, y_0, z_0) over time"). State-of-the-art frameworks using standard PCs, which allow for lower costs and a high degree of scalability, perform these tasks by storing the CFD data sets in main memory and quickly extracting interesting features by completely scanning the data set.



a) CAVE b) HoloBench

Figure 1: Examples of VR Hardware

With increasing CPU powers, computers are able to produce larger and larger simulation data sets that quickly exceed main memory limitations, calling for an effective and efficient organization of the CFD data sets with external memory access structures. Another negative effect of the immense data sets, which are often many gigabytes in size, is the increasing response time of post-processing queries. New access methods, presented in the following chapters, avoid slowing down the user's flow of work by speeding up processing time until a result is presented and ready for visual inspection.

We developed the index based graphics data server "IndeGS" offering secondary storage methods to break the main memory limitation. Furthermore, it offers various dynamic view dependent access methods to deliver a good and quick first impression of the results enabling the user to change view and post-processing parameters "on the fly" during query processing with immediate response from the system. In our demonstration setup we offer the ability to explore the effects of different access methods and dynamic user behaviors during the course of post-processing.

VLDB '07, September 23-28, 2007, Vienna, Austria.

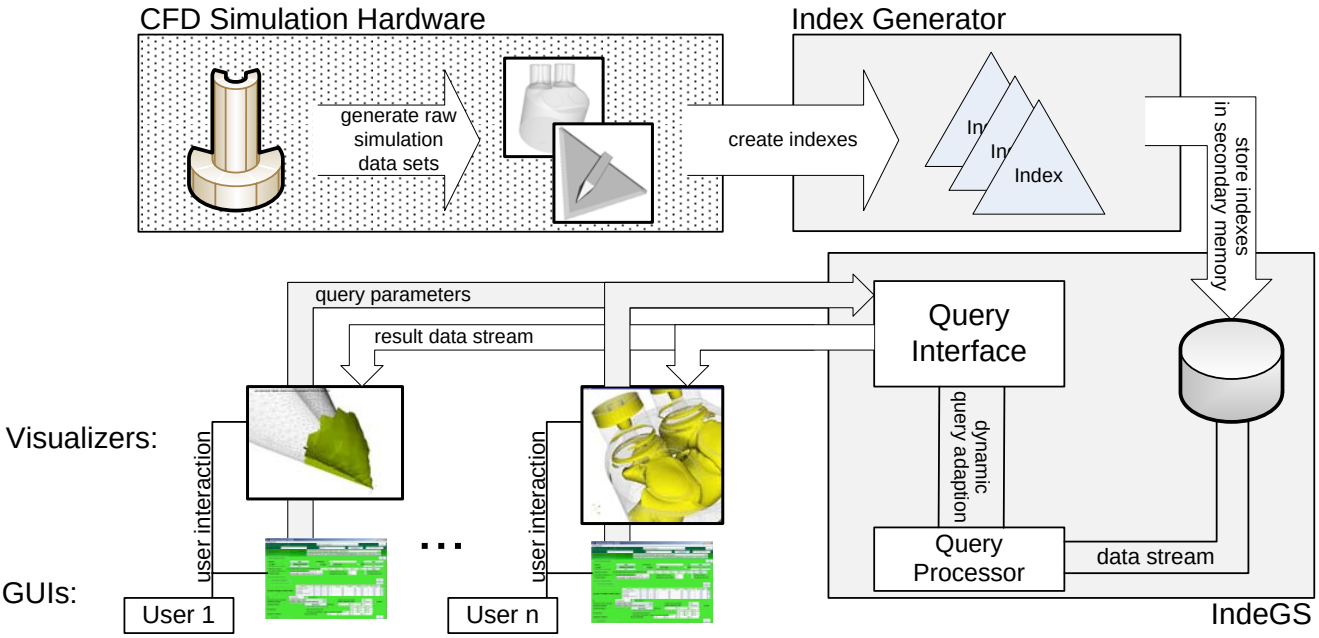


Figure 2: System Architecture

2. SYSTEM ARCHITECTURE

Figure 2 shows the system architecture of our demonstration setup. The dotted area shows the CFD simulation components which are not part of our demonstration. These simulators generate huge amounts of raw data by performing numerical simulations of interactions of fluids or gases on complex surfaces of arbitrary level of detail. From these raw data sets, our index generator creates secondary storage based indexes which are described in section 3.1.

The central element of our demonstration is the graphics data server "IndeGS". Queries are received by the query interface, which interacts with the query processor. The query processor accesses secondary storage indexes and extracts the results via access methods described section 3.2. Result data will then be streamed to the client visualizer which initiated the query. These clients run independently of "IndeGS" on different instances in the network. They can be complex hardware configurations like a *CAVE* or a *HoloBench*, but they can also run on standard PCs or mobile devices like PDAs. For our mobile demonstration setup, we concentrated on the use of notebooks. In contrast to complex VR setups, where the user's position and movements are acquired by tracking devices, the user of our demo controls the view parameters with standard input hardware like keyboard and mouse. Postprocessing type and parameters are entered via a GUI. During query processing, "IndeGS" reacts to changes regarding view point and direction and adapts the result stream to maintain view dependency. At any time during query execution, postprocessing can be restarted with different parameters at user's request. Typical post-processing queries are *isosurface extraction* (e.g. "create isosurface with temperature = 125°C"), *range queries* (e.g. "display cells with temperature between 100°C and 120°C") and arbitrary combinations e.g. with *geometrical selection* ("display cells with temperate = 125°C and humidity = 70% only in certain region of data set"). Data sets available in our demonstration setup are a simulation of a fuel injection into a combustion engine cylinder and simulated aerodynamic flows over the surface and in the surrounding of a delta wing airplane.

3. DESCRIPTION OF "INDEGS"

For the purpose of efficient CFD post-processing on data sets of almost arbitrary size, we developed our graphics data server "IndeGS" which can be integrated in any virtual reality framework via a clearly defined communication. "IndeGS" is connected to the freely available and powerful toolkit *ViSTA* [7] used for integrating VR technology in technical and scientific applications. A more detailed explanation of the techniques used in "IndeGS" can be found in [3]. The experiments conducted there prove the efficiency of "IndeGS" in the field of CFD post-processing on different real-world data sets.

3.1 Index Structure

To cope with the immense data sets generated during CFD simulations, "IndeGS" organizes and stores the data in secondary index structures derived from well-known spatial data structures like the R-Tree family (R-Tree [4], R^*-Tree [1] , X-Tree [2]), which have proven to be appropriate for indexing spatial data and which use minimum bounding rectangles (MBRs) that serve as directory nodes during tree traversal. The data itself is stored on leaf node level. For the detailed discussion of the techniques realized in "IndeGS", we refer to [3].

The CFD data sets handled by "IndeGS" are based on the open standard *VTK* [6]: they consist of cells (tetrahedra, pyramids etc.) which are defined by their corner points. Each corner point carries additional scalar information (temperature, pressure, density etc.), depending on the executed simulation. For example, the delta wing data set consists of 3 time steps with 15 million cells, 4,5 million data points

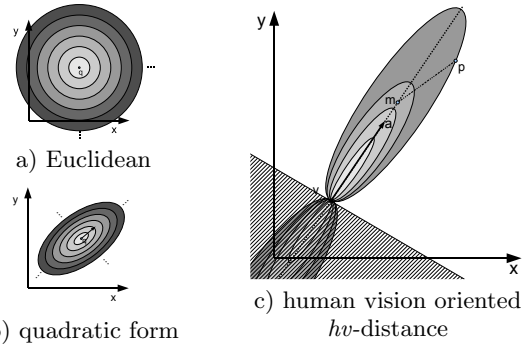


Figure 3: Distance Function Isolines Overview

and 8 scalar values each, resulting in a file size of approx. 2.5 GB.

3.2 Query Processing

To lower costly operation times of complex VR hardware and to increase efficiency of expert users performing CFD post-processing, it is of utmost importance to reduce query times and to allow the user to get a quick first impression of the query result. The access methods offered by "IndeGS" show the following benefits: (1) results close to the viewer are presented rapidly, (2) results in direct line of sight are ranked with higher priorities, (3) changing view position and view direction are reflected by dynamic query adaption. Queries q are specified by the following parameters:

$q :=$ (view point v, view direction a, box b,
scalar range sc_1, ..., scalar range sc_n).

v is the exact position of the viewer in or around the visualized data set and a the view direction specified as a three-dimensional vector. Box b specifies the geometrical range of the query. Usually, this box covers the complete range of the three dimension, as no results shall be omitted from the result set, except in the case of "geometrical selection" post-processing. Ranges of the scalar values sc_i can be specified to meet the requirements of each query. In the case of isosurface extraction for a certain scalar, the corresponding range sc_i has to be set to the designated value. A cell is considered *active*, if all its values intersect with the corresponding ranges specified in q.

The first two benefits from above are realized by ranking and streaming all result cells depending on the users position and its view direction in the VR environment. Cells in the proximity and in the line of sight of the user are thereby delivered before cells which contribute less to the overall impression of the complete result set. The ranking based on k-nearest neighbor search using priority queues is presented in [5], which perform a recursive walk through the underlying spatial data structure by calculating priority/distance approximations (*MINDIST*) for MBRs in directory nodes and exact priorities/distances for objects in leaf nodes based on distances like Euclidean distance etc.. The result set is incrementally streamed to the visualizer, while intermediate MBRs and cells are kept in a queue according to their priorities. The query point is set to the users standpoint and the nearest neighbor search is performed until all results cells have been streamed to the visualizing component of the VR framework. Besides depth-first search, our "IndeGS" offers a variety of distance functions to be used for ranking result cells, which we discuss in following.

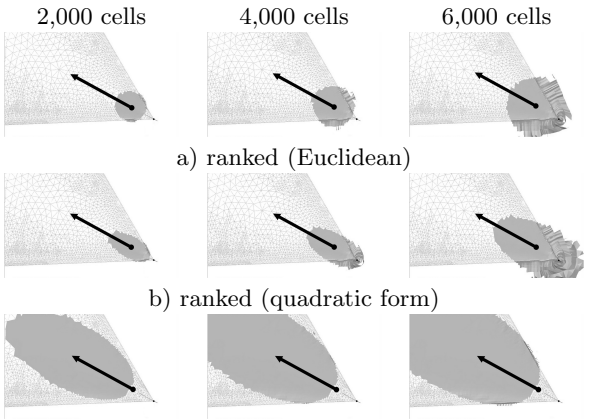


Figure 4: Distance function examples

3.2.1 Distance Functions

The most common distance function is the Euclidean distance (cf. figure 3a). Combined with query processing techniques from above, result cells are visualized in a growing spherical manner around the user's standpoint v. Thus, the standpoint v of the user is taken into account and cells closer to the user are presented before cells farther away. To increase the speed of appearance of result cells in the direct line of sight and improve the "first impression" of the result at the same time, we also consider quadratic form distance functions, where the shape of the isoline ellipsoid is stretched in direction of the axis of sight (cf. figure 3b).

To further integrate the view direction and in particular the view orientation into our system, we developed a new distance function reflecting characteristics of human vision (referred to by us as *hv*-distance). Growth of the result set is accelerated in direction of the line of sight and at the same time delayed in the area of human peripheral vision. The corresponding isolines are illustrated in figure 3c): the distance of p is defined by the radius of the ellipsoid which crosses p and v with center point m, which lies on the line of sight a starting at v. Elements behind the user (located in the hatched region defined by the hyperplane running through v and perpendicular to a) are treated separately, after all cells in front have been streamed to the visualizer. A user standing at point v with line of sight a will first be presented elements directly in front of her or him close to the a axis, and elements farther away from v and a will appear later (peripheral vision). For the exact mathematical derivation of the corresponding distance definition, we refer the reader to [3]. The required definition of an appropriate *MINDIST*-function can also be found there. The *MINDIST*-functions of both quadratic form and *hv*-distance function each apply a customized gradient technique to find the location on the MBR which has the least distance to the query point.

Thus "IndeGS" offers access methods with different degrees of view dependency: depth-first traversal of the index delivers all result cells in random order. Ranking with Euclidean distance incorporates the user's standpoint. With quadratic form distance functions, the view axis is considered, and with the *hv*-distance function, the view orientation is also taken into account during ranking.

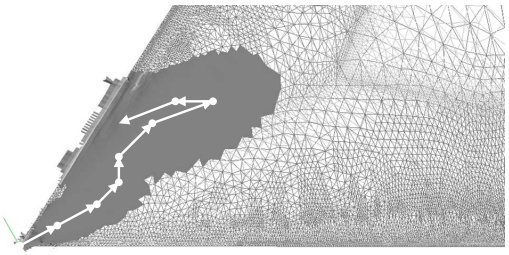


Figure 5: Dynamic Query Execution Example

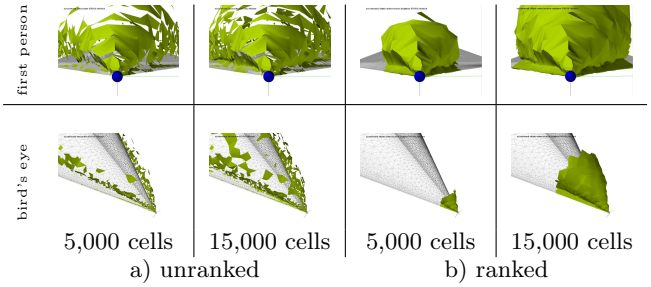


Figure 6: Query processing "delta wing tip"

Figure 4 shows an example of an isosurface extraction with a scalar value of 0.2 for *mach number* from the delta wing data set at different stages during query processing (after 2,000, 4,000, 6,000 cells). The screenshots show a section of the tip of the delta wing from a bird's eye view with the user's position and view direction depicted by the arrows.

3.2.2 *Dynamic Query Processing*

Another important aspect of CFD post-processing is the dynamic behavior of users: the change of the user's standpoint and view orientation has a direct influence on the priorities of the result objects. As "IndeGS" is using a priority queue for ranking elements, this queue will be reorganized if certain view parameters change, depending on the distance function used. The Euclidean distance is robust against changes of view direction. The calculated priorities of elements in the queue only change, when the user/query point moves. Quadratic form and *hv*-distance functions are influenced also by a view direction change. When these changes happen during query execution, the queue will be be rearranged to guarantee a correct ranking and smooth query processing. As a user might trigger many queue rearrangements by moving around in the VR environment and rapidly changing view directions, a careful examination and optimization of these rearrangements is inevitable. We therefore developed techniques to avoid that queue rearrangements noticeably decelerate query processing by only triggering rearrangements if the change of view parameters exceeds a certain threshold. This leads to a lower number of rearrangements, but at cost of the quality of the ranking, which will not be correct until the queue is reorganized. Other heuristics implemented in "IndeGS" in order to keep the average queue sizes low are described in [3].

Figure 5 shows the same isosurface extraction as in figure 4 from a bird's eye view, but with a moving user (depicted by the arrows). It can be clearly observed that the result set is growing in front of the user, whenever a new query is triggered (here: one arrow represents one view modification). This screenshot shows query execution up to 9,000 out of $\approx 150,000$ result cells.

4. DEMONSTRATION BENEFITS

With the help of our demonstration setup, it is possible to examine and display the effects of the use of different access methods offered by "IndeGS" in the context of CFD post-processing in a comprehensible and vivid way. By including a visualization client running on standard PC hardware, the use of "IndeGS" and its functionality can be easily demonstrated without being constrained to expensive and complex VR hardware. The user can start arbitrary post-processing queries on different data sets and experience how "IndeGS" reacts to dynamic query updates from freely chosen perspectives. "IndeGS" itself reports information about query processing in a text window including block reads, number of result cells, processing times and index parameters.

Figure 6 shows more screenshots of an isosurface extraction on the delta wing data set at two different stages of query processing (after 5,000 and 15,000 returned cells) for unranked and ranked result streams and from two different perspectives.

Figures 4 to 6 show, that our demonstration enables the exploration of the potentials of "IndeGS" in a vivid and comprehensible way by performing CFD post-processing on example sets and examining the effects of different access methods.

5. ACKNOWLEDGMENTS

The authors would like to thank Christian Bischof and Marc Wolter from the *Center for Computing and Communication* of the RWTH Aachen University for providing access to their virtual reality infrastructure and for supporting the integration of "IndeGS" in the *ViSTA* framework. We also thank Christian Klaus and Dennis Meichsner for their valuable work on the basic implementation of the graphics data server.

6. REFERENCES

[1] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. In *SIGMOD Conference*, pages 322–331, 1990.

[2] S. Berchtold, D. A. Keim, and H.-P. Kriegel. The X-Tree: An Index Structure for High-Dimensional Data. In *VLDB Conference*, pages 28–39, 1996.

[3] C. Brochhaus and T. Seidl. Efficient Index Support for View-Dependent Queries on CFD Data. In *SSTD Conf.*, pages 57–74, 2007.

[4] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *SIGMOD Conf.*, pages 47–57, 1984.

[5] G. R. Hjaltason and H. Samet. Ranking in Spatial Databases. In *SSD*, pages 83–95, 1995.

[6] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit*. Kitware Inc., 2004.

[7] T. van Reimersdahl, T. Kuhlen, A. Gerndt, J. Heinrichs, and C. Bischof. ViSTA - a multimodal, platform-independent VR-Toolkit based on WTK, VTK, and MPI. In *IPT Workshop*, 2000.