An integrated 3D approach for handling geometry and simulation data in planning processes

Peter Apian-Bennewitz

Fraunhofer Institute for Solar Energy Systems, ISE
Thermal and Optical Systems
Solar Building Group
D-79100 Freiburg, Germany
apian@ise.fhg.de
http://ise.fhg.de/radiance

ABSTRACT

We present the philosophy, implementation and experience with *rshow*, a tool for integrated display of three-dimensional data during the planning and architectural design process. In a compact and expansible interface, it combines the building geometry with simulation data, e.g. daylighting, thermal analysis or airflow data. One main idea is to display the inherently three-dimensional building form as truly 3D data which can be interpreted as intuitively as possible. Scalar or vector simulation data can be added into this 3D visualization, resulting in a homogeneous and consistent summary of available design criteria.

To implement this technically, the core display engine was designed to run effectively on standard computers in the NT range as well as high-end UNIX visualization workstations. The response to typical, frequently occurring user tasks is fast enough on all systems to give the user sufficient 'feeling' for the 3D data sets. [1]

1 INTRODUCTION

The planning and construction of buildings relies increasingly on computer generated numerical data. These ensure both a high standard and accelerate the planning process. Typical examples of computer simulations include *stress analysis* of concrete and steel structures, *thermal calculations* of walls and interior spaces, *lighting* with artificial sources and daylight in office spaces or museums, computational fluid dynamics of *air flows* in interior spaces and *acoustics* of theatres and halls. Each of these areas uses its own set of programs, which generally are becoming faster and easier to use, but all of them generate an increasing amount of data.

The *Solar Building* group at Fraunhofer ISE has accumulated field experience with simulation tools over 10 years, during which it became apparent that the *visualization* of simulation data plays a key role: During the planning and consultancy phase, the interactive use of simulation programs becomes faster because results are interpreted more easily. Communication with others in a project is easier since results are clear and key points easy to grasp. And finally the same high quality visualization may influence decisions by the client side. This is especially useful when presenting advanced concepts or materials.

In the following we describe the *rshow project*, which was originally started in 1993 to visualize daylight distributions in interior spaces. Based on in-house experience and feedback by international users, it was largely rewritten during 1997/98 to extend visualization to a broader set of data. The design philosophy of *rshow* will be outlined, together with examples of different data types and their visualization. Comparisons to similar projects are sketched briefly.

2 INTERFACE PROFILE AND DESIGN PHILOSOPHY OF *RSHOW*

The *rshow project*'s aim is to write a software tool for displaying data to engineers in a way which is flexible, quick and easy to handle. This is determined by the following parameters:

- Workflow
- Type of data to be displayed
- Display method

We will proceed along this list to explain *rshow*'s principles.

2.1 Envisioned Workflow

Rshow is conceived of as the front end for handling simulation data during planning. It is a tool to analyze performance aspects where the geometric building shape is defined. Therefore no effort

[1] To be published at *Second European Conference on Product and Process Modelling, 19-21 Oct. 1998, BRE Watford, UK*

Type	Definition Area	Data
scalar	surfaces	temperature
	surfaces	irradiance *
vector	sphere	light scattering functions (*BRTF*) *
vector	volumes	air flow, sheer forces
tensor	volumes	sheer tensor

Tab. 1: Datatypes commonly found in building simulation. Those marked with * are used in examples in the text.

has been made to implement CAD features for manipulating the building shape.

It is used in addition to CAD programs, since these are very rarely adapted to the needs of simulation programs, mainly since simulations need additional parameters per surface or volume, which are not accessible from the CAD GUI.

Rshow was not written as a plug-in to CAD programs, as existing software which does (e.g. SiView [Sch97]) was found too limited in their methods of displaying data.

The parameters and material properties needed for numerical analysis are fed to the simulation program by *rshow*, which in turn displays the results in the three-dimensional context of the building envelope. Making this loop easy and quick to use creates a working environment where different strategies, methods or materials can be tested, selected and checked with ease.

Some numerical algorithms, especially for lighting analysis, are implemented internally in *rshow*, while other calculations are done externally by other programs. Data for the latter case is exchanged by files or, if the operating systems permits, by pipe.

Rshow is not primarily intended for laypeople, as it does not hide technical aspects of simulations and parameters from the user. Our experience with simulation programs was that none of them combine robustness, speed and precision in an automatic way to ensure consistent results when used without a careful and experienced check on their numerical parameters.

2.2 Available Types of Data

The discipline of scientific visualization has developed a spectrum of methods for displaying data, most of which are related to computational fluid dynamics (airflows) and stress analysis. These data undergo considerable processing in which surfaces or vector fields are generated for display. However, most data found in building simulation are defined as scalar quantities on surfaces (Table 1), which poses comparatively few problems in terms of scientific visualization ([Men90]). We will focus on scalar quantities in this paper, as they are frequently used. In this category, lighting quantities (e.g. irradiance and radiance) will be taken as primary examples.

2.3 Structure of rshow

For displaying three-dimensional data with a graphical user interface (GUI) on UNIX and NT platforms, we chose the following components and libraries:

- Language: ANSI-C for flexibility, standardization and speed
- 3D library: Open-GL, the de-facto industry standard
- GUI: Tcl/Tk for cross-platform support (UNIX, Mac, NT)

Note that higher level 3D libraries or toolkits (e.g. *Inventor* offered by Silicon Graphics Inc.) were bot chosen, as they were considered too be to inflexible or proprietary.

Rshow itself is divided into two main parts: core functions (window handling, device handling) and modular drawing functions. The latter handle different types of data and display styles, linking their own GUI routines to the core GUI. This modular approach offers extensibility and flexibility.

Displaying potentially complex three dimensional geometry at interactive update rates needs computer power which traditionally has been the exclusive domain of high-end workstations. As part of the current trend to bring 3D graphics to PC systems, the hardware costs fell significantly. *Rshow* is intended to run on systems varying from midrange PC systems to high-end workstations. Interactive update rates are achieved for all platforms by limiting display features depending on hardware speed.

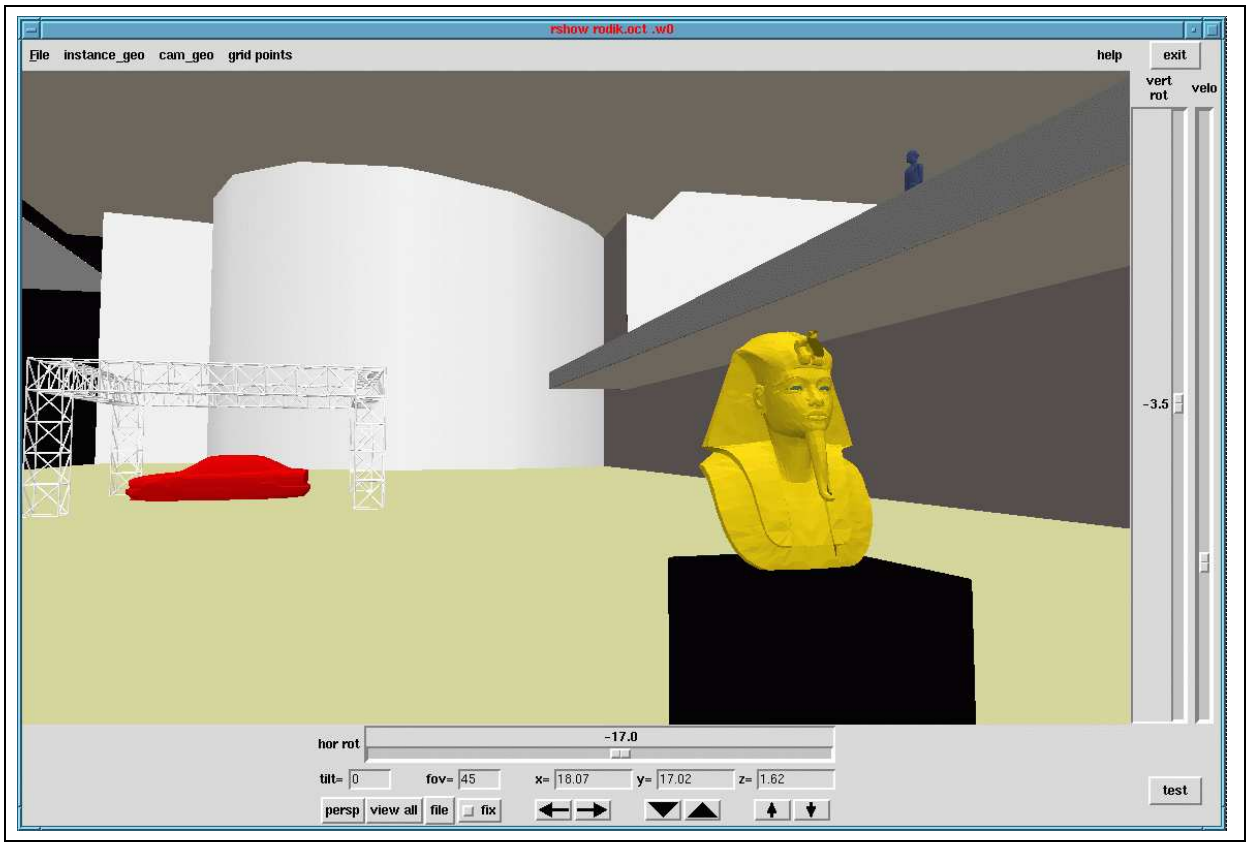

Fig. 1: The core window of *rshow*: Pull down menus are located at top, a navigation bar is located below the image. The sliders adjust line-of-sight, the arrow buttons provide lateral movement. Extension input devices are used if available. Data courtesy Ipas

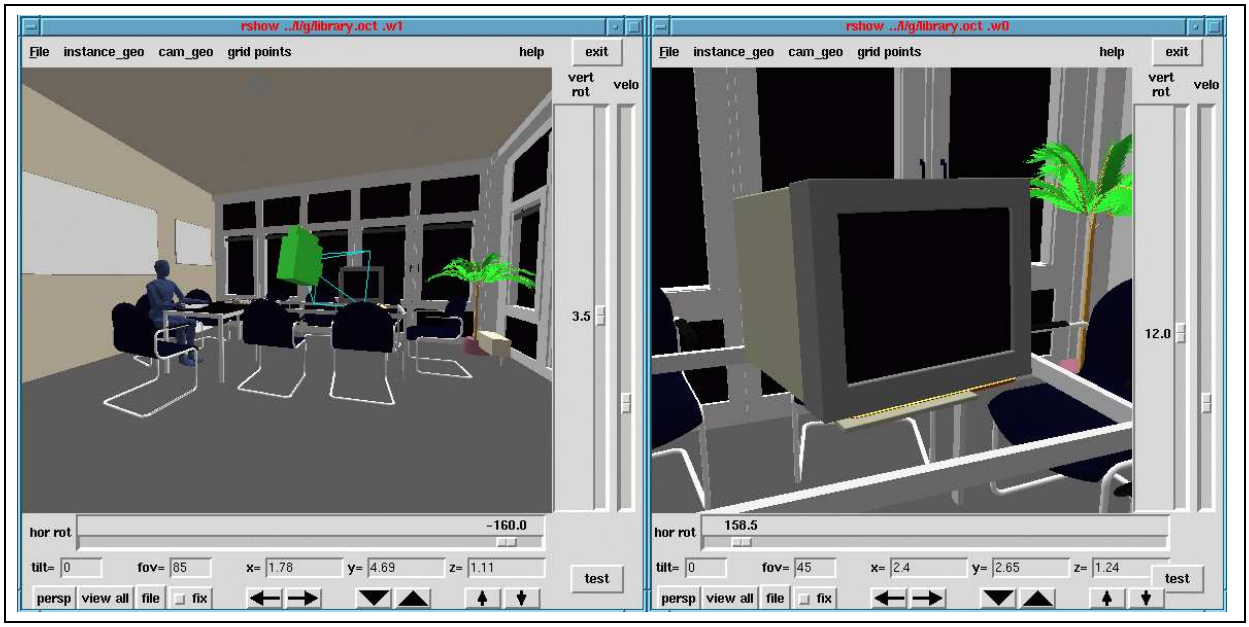

Fig. 2: Display in multiple windows offer different views and display modes. The camera model displayed in the left window represents the view position in the right window, a feature most often used to navigate in complex spaces.

Rshow uses additional input devices were available (see below). Free maneuvering in 3D space was found to be easier with additional input devices, especially if they offer 6 axis of freedom (e.g. SpaceMouse). Furthermore such a configuration allows a fast two-handed method of working, one hand for navigation, the other one for selecting surfaces, data values or options.

Regarding *Virtual Reality*, mostly associated with helmet-mounted-display (HMD), data-gloves and other immersive technologies, *rshow* took a conservative approach as these technologies are not frequent at user sides. However, an implementation at core level is seen as straight forward.

Primary development environments are Linux Pentium-PC, OnyxII by Silicon Graphics and Hewlett-Packard HP777 workstations. Beta releases are frequently made available on the web at **http://ise.fhg.de/radiance/pabs-toolbox/rshow/newrshow.html**.

2.4 Core Features

The core implements a multi-window, single-process, full 3D and extensible drawing environment in which the routines for data display are embedded. Fig. 1 shows the GUI as defined by the core tcl/tk functions, to which the data display functions add their own specific pull-down or pop-menu menus.

Multi-window display is made possible by a modular approach to drawing and GUI tcl/tk routines and offers multiple views of the same geometric data set with different display modes possible (Fig. 2). *Rshow* implements an internal event distribution for external events, window resizing, changes in geometry and adaptive refinement. Each window has a 3D-camera associated with it, specifying view position, angles, type and auxiliary parameters. It can be seen as a view into the virtual 3D world.

The "single-process" *rshow* does not "fork" processes, as the NT operating system doesn't offer process communications as powerful as UNIX. Consequently task switching, e.g. between different drawing windows, is implemented internally.

Under UNIX, extension devices are handled through the standard X-window protocol, using X's very powerful abstraction layer for input extensions. In practice this includes economic joysticks on PCs running Linux as well as high-end 6-axis devices like SpaceBall or SpaceMouse on Silicon Graphics Onyx machines. Additionally, devices not supported by the X-server could be handled by the *rshow* I/O module through serial- or other I/O ports.

The display process requires optimization to achieve interactive update rates, which is potentially handled in the core functions with culling, display list compiling and display mode switching.

2.5 Examples for Data Display

Rshow modules for data display consist of two parts: C-code and tcl/tk scripts, which attach to the core of the program. The display possibilities of these modules are practically unlimited, as they use Open-GL commands directly to define their 3D objects.

Typically these objects are stored in Open-GL display-lists, which are called by the redrawing function defined in the modules. However, its up to the modules to use display-lists or not.

The drawing subroutine written in C communicate with the tcl/tk scripts of the GUI for that module. If a user changes the drawing style or other parameters of an object, this change is handled locally inside the module. Among other structures, this is crucial for an extensible program.

We illustrate the ideas with three examples: The basic building shape forms the backbone of visualization, irradiance levels are one of the simpler examples of visualization and the visualization of light scattering is slightly more complex.

2.5.1 Geometric building shape

The building shape is read from files generated by CAD programs and consists of so called geometric primitives, such as polygons, cylinders or spheres. Some CAD formats allow more complex types like non-uniform-rational-b-splines (NURBS) [Fro98]. These primitives are broken down into Open-GL polygons suitable for the display hardware, which draws them according to preselected styles.

Four parameters for drawing the building shape seem relevant for visualization, as shown in the examples.

- Solid, shaded, textured display

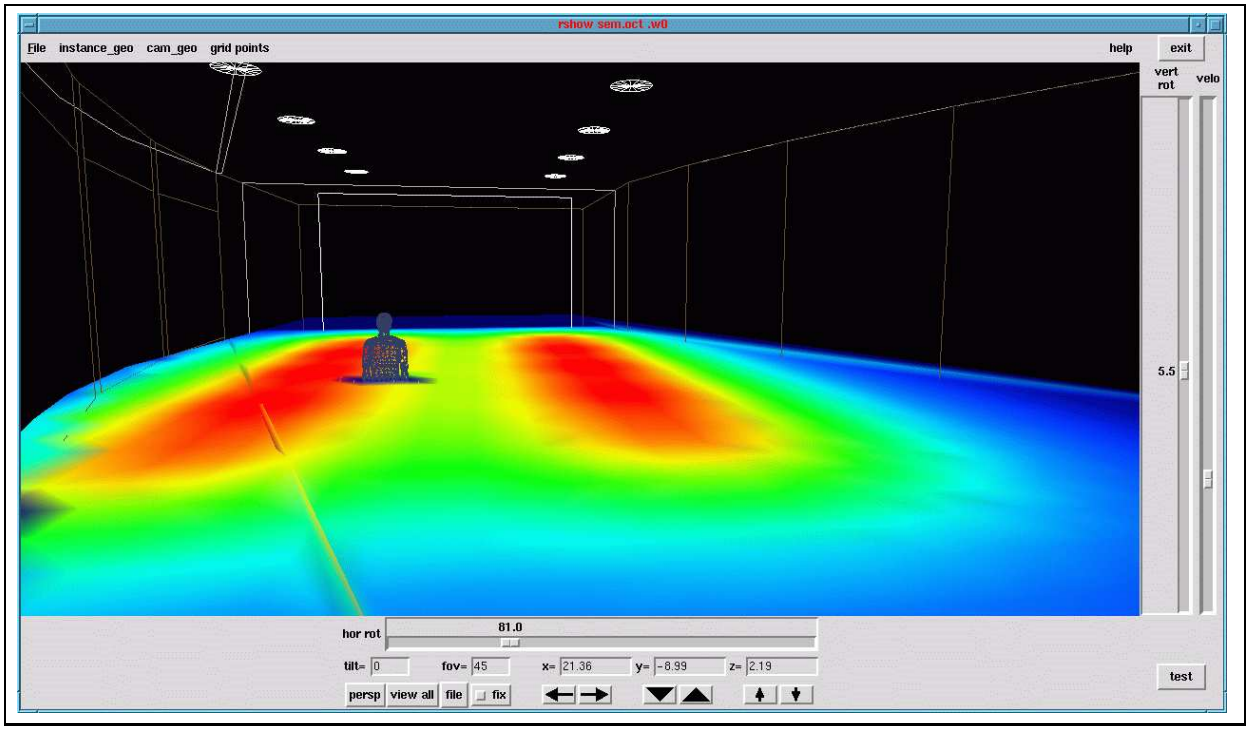

Fig. 3: Illumination levels at 1m height in a planned seminar room. Building geometry displayed as wireframe. Seconds later the user can press a button and start the rendering shown in Fig.4. The orientation of the plane for illuminance calculations is freely chosen by the user.

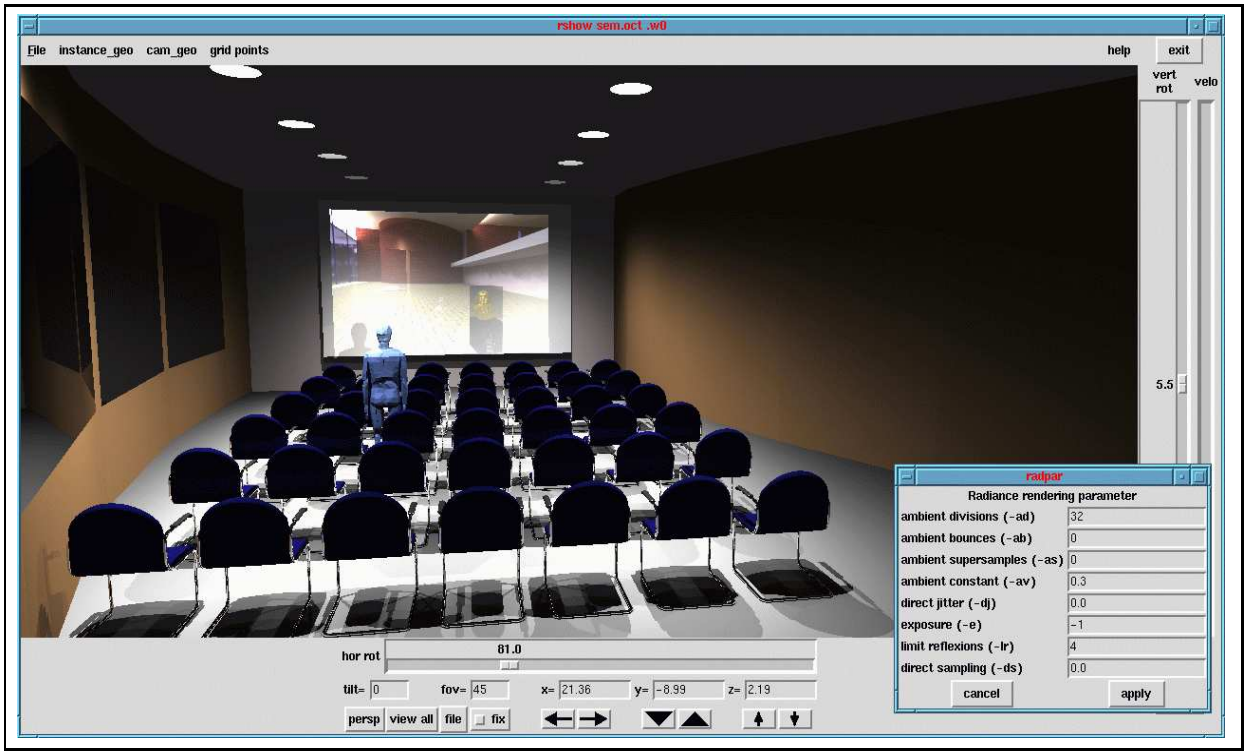

Fig. 4: Raytraced image of seminar room, using the RADIANCE raytracing engine to display luminance values during the *rshow* session from Fig.3. Note the partially visible projected image.

- Wireframe for speed
- Clipping planes for display of internal structures (cutaway view)
- Translucent for display of internal structures

2.5.2 Luminance and Illuminance display

Traditional lighting analysis consisted of checking irradiance levels on working surfaces to comply to legal requirements and experience. Newer studies of visual comfort use radiance and luminance as key values to identify glare, veiling reflections and other parameters.

Fig. 3 shows an *rshow* screen with a classical example of illuminance values mapped to a false-color scale. The wavelength dependent irradiance values typically calculated by the light simulation program are converted to one illuminance value, which is mapped to a color.

A luminance calculation with different light sources is shown in Fig. 4. Adaptive raytracing is used to get a physically valid image of the light distribution in the scene. These calculations are started on demand of the user either within the *rshow* window in foreground mode (adaptive refinement) or in batch mode, optionally across a network of machines.

As in the irradiance simulation, the calculations are done internally in *rshow* using routines of the Radiance Synthetic Image System ([WLS98]). Combined with an easy to use display of simulation parameters, this visualization makes high precision lighting calculations as easy as pressing a button.

2.5.3 Daylighting (BRTF display)

New materials are available for daylighting (e.g. switchable materials, like thermo-chrome) [jou94], but not yet common practice. Sometimes the implementation of the concept and the selection of a new building material is therefore not intuitive and helped considerably by visualization. Fig. 6 and Fig. 5 show a translucent material as an example.

Generally, light scattering is characterized by the bidirectional-reflectance-transmittance-function (*BRTF*). At a point of the material surface it is given as: $BRTF(\vec{x}_{out}, \vec{x}_{in})$. The two vectors specify the outgoing and incidents light directions (see [AB95] for an introduction to *BRTF*s) For our discussion here its simply a quantity depending on four angles. The incoming light direction depends on time of day, building location and orientation. The visualization has to show the *BRTF* values for a user selectable incoming direction by displaying $BRTF_f(\theta_{out}, \phi_{out})$. This data is very similar to a candlepower-distribution-curves found with lamps.

Fig. 6 combines a display of measured material characteristic with a display of the resultant illuminance distribution, which depend on additional properties. To see both in one image has advantages: It verifies the congruence between measured data and the mathematical model which is used for light simulation. And it shows the influence of other parameters, like geometry of the building or wall reflectance more clearly.

Similar arguments hold true for artificial lighting and candlepower-distribution-curves.

The data $BRTF_f(\theta_{out}, \phi_{out})$ is displayed as a surface defined by triangulation, a standard method which was used at ISE outside *rshow* before. Displaying the distribution inside the building shape handles all geometric transformation for window orientation, sun position, time of day etc.

The engineer can focus on the interaction between the material and the building geometry, selecting different materials from a database and testing their light output for different times of day and seasons. Light calculations for the interior spaces, in which the properties of walls etc., are taken into account are started next.

3 CONCLUSION

The framework of this new program to visualize data related to the planning of buildings was presented and the ideas demonstrated with examples. Other fields, e.g. airflow simulation, may use their own concepts within *rshow*.

Consistency of results is greatly enhanced by combining CAD geometric data with display of numerical calculations. Using one graphical frontend for a variety of different simulation tools is user friendlier and faster for the user.

The conceptual openness to other data types and the direct use of OpenGL make *rshow* more flexible and extendable than other programs. This allows the engineer to model and visualize different aspects of the cross-linked design process in a modular and consistent step-by-step way.

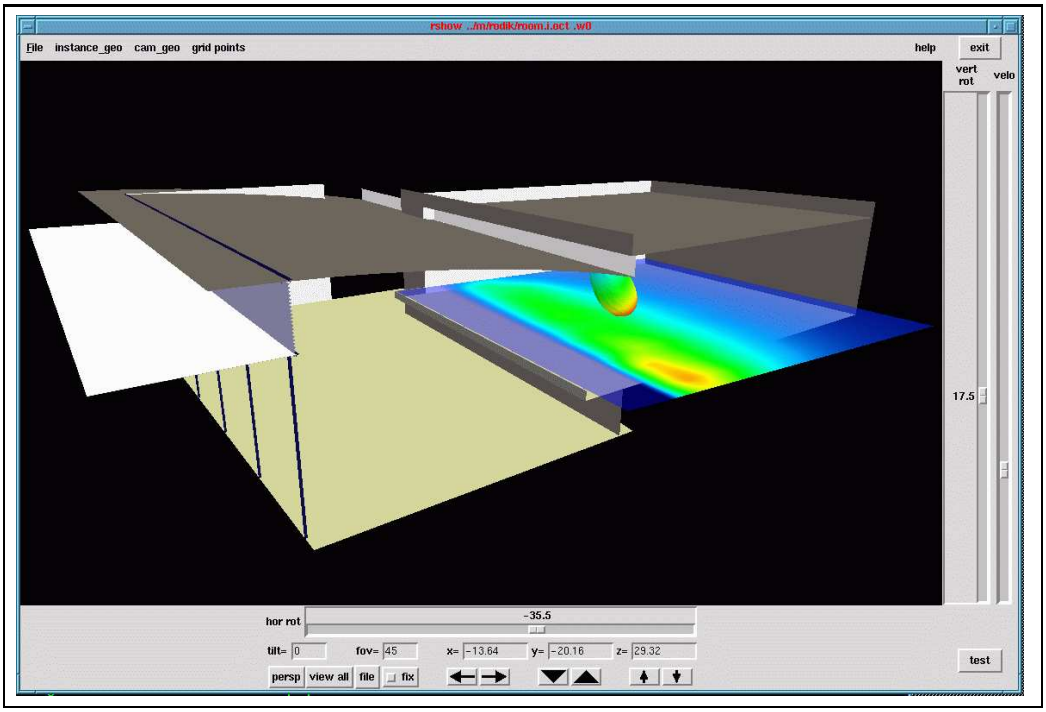

Fig. 5: Cutaway view of planned museum building. The rear space is partially lighted by a celestory band of translucent material as shown in the detailed view Fig. 6. Note: This is an *rshow* example, not a proposition for a working daylighting application.

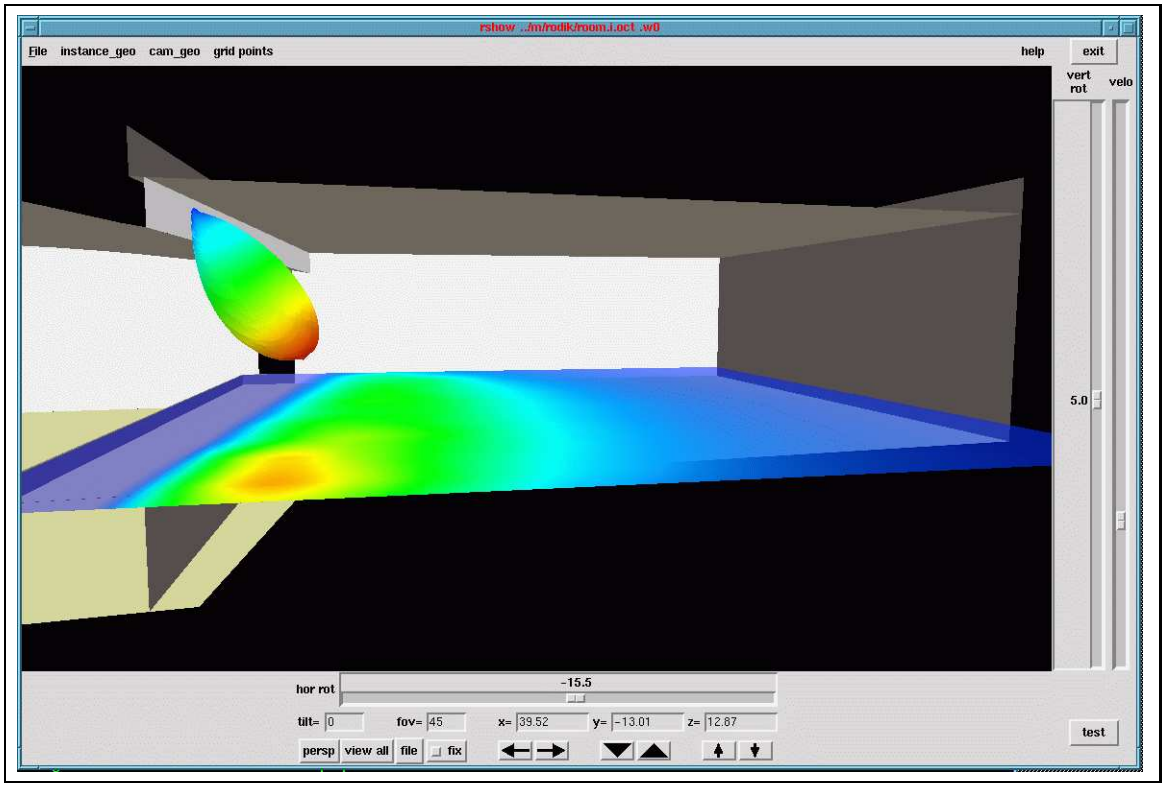

Fig. 6: Closer view at rear section of museum. The scattering characteristic (*BRTF*), which characterizes the translucent window material, was measured at Fraunhofer-ISE and is displayed as surface defined polar coordinates. As explained in the text, this is similar to candlepower distribution curves. Illuminance levels 1m above floor-level are displayed as a falsecolored plane.

4 FURTHER WORK

Further work concentrates mainly on adding modules as necessary and enhancing display speed by OpenGL refinement.

ACKNOWLEDGMENTS

Many thanks to Prof. Luther, head of Fraunhofer-ISE, for the internal grant which started the rewrite of *rshow*; to Greg Ward, author of Radiance, for discussions and help with Radiance's inner working; to Priv. Doz. Dr. Volker Wittwer, head of department, for continuous support, to B. Holder for support in purchasing the SGI VGX320 which brought the original *rshow* to life; and to all co-workers at Fraunhofer ISE for making the place nice to work at.

REFERENCES

[AB95] Peter Apian-Bennewitz. *Messung und Modellierung von lichtstreuenden Materialien zur Computer-Simulation von Tageslichtbeleuchtung.* PhD thesis, Universität Freiburg, Fraunhofer Institut für solare Energiesysteme, D-79100 Freiburg, November 1995.

[Fro98] Thomas Froese. Step data standards and the construction industry. http://www.civil.ubc.ca/~ tfroese, 1998.

[jou94] joule2 particpants. Characterization of glazings materials for daylighting applications, final report. Technical report, The Comission of the European Communities, CNRS ENTPE, Rue Audin, 69518 Vaulx-en-Velin, France, 1994.

[Men90] Raul H. Mendez. *Visualization in Supercomputing.* Springer Verlag, 1990.

[Sch97] Hans-Joachim Schmidt. Lichtsimulation leicht gemacht. In *Drittes Symposium Innovative Lichttechnik in Gebäuden*, Regensburg, 1997. OTTI Technologie-Kolleg, Ostbayerisches Technologie Transfer Institut.

[WLS98] Greg Ward Larson and Rob Shakespeare. *Rendering with Radiance.* Morgan Kaufmann, 1998.