Using VTK as a Tool for Teaching and Applying Computer Graphics

P. Dias, J. Madeira and B. Sousa Santos

Dept. of Electronics, Telecommunications and Informatics / IEETA, University of Aveiro
Campus Universitário de Santiago, P-3810-193 Aveiro, Portugal
Email: paulo.dias@det.ua.pt

Abstract

During the first semester of 2005/2006 we used the Visualization Toolkit (VTK) as a tool for teaching and applying Computer Graphics, both for Computer Engineering students who chose to attend the 3D Modeling and Visualization *course, and for M.Sc. students specializing in Computer Graphics.*
In both cases, students had not only to use VTK in about half of their lab classes, in order to accomplish some tasks and gain some knowledge on VTK's features and functionalities, but they were also required to develop a visualization application based on VTK.
We describe first the motivation for using VTK in these two different scenarios, as well as the main course topics where we used the toolkit. Afterwards, we present some of the most successful projects developed by our students. Finally, we state some conclusions.

Categories and Subject Descriptors (according to ACM CCS): K.3.2 [Computer and Information Science Education]: Computer Science Education; I.3.4 [Computer Graphics]: Graphics Utilities; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

1. Introduction

In the last few years we have been witnessing a discussion on how to better teach Computer Graphics (CG) to students in different areas [GBK*95,HS00,Paq05,ACSS06]. In the past, a bottom-up approach was normally used, where students had to build all necessary code (almost) from scratch. More recently, many educators have switched to a top-down approach, based on using a higher-level API such as OpenGL or Java3D, with less relevance being given to raster-level algorithms. Although classical graphics textbooks, based on the traditional bottom-up approach (e.g., [FvDF*94]), do remain useful, advances in hardware, graphical libraries and more recent API-based CG textbooks [SML98, HB04, Ang06] offer both students and educators the possibility of exploring advanced concepts and developing useful course projects, e.g., for data visualization.

When planning our courses in the Computer Graphics area for the first semester of 2005/2006 we were faced with two distinct challenges:

- About 20 Computer Engineering students entering their last year, with some background on CG fundamentals, showed their interest in the elective course *3D Modeling and Visualization*, which should present them with more advanced CG concepts and offer them the possibility of working with *de facto* CG and Scientific Visualization standard libraries.
- At the same time, we had decided to offer to M.Sc. students specialization courses in CG, Visualization and Geometric Modeling, which had to be accompanied by a CG Lab course, providing them with some hands-on experience.

Since we consider that the top-down approach is more adequate to these courses, we decided to introduce a well-known visualization library, the Visualization Toolkit (VTK). In both cases, we would be teaching students already possessing some basic CG knowledge and, also important, having some object-oriented programming experience; thus, we decided to base part of the lab classes on the VTK library, and also require them to develop a visualization application based on VTK. In this way, students would have to use a higher-level API, in addition to the traditional OpenGL, and would have to acquire some knowledge on Data Visualization, perhaps today's most important application area of CG.

VTK [SML98] is an open-source, freely available toolkit for 3D computer graphics, image processing, and visualization. It uses a higher-level of abstraction than other rendering libraries, like OpenGL, making it much easier to create graphics and visualization applications. In addition, it also offers a wide variety of visualization algorithms including scalar, vector, tensor, texture and volumetric methods; and advanced modeling techniques like implicit modelling, polygon reduction, mesh smoothing, cutting, contouring, and Delaunay triangulation.

In what follows, we briefly present the contents of the courses taught, and describe some of the most successful projects developed by our students. Finally, we will comment on the results and present some conclusions.

2. 3D Modeling and Visualization

The *3D Modeling and Visualization* course (3DMV) was introduced for the first time in 2005/2006 as a final year elective for Computer Engineering students, corresponding to 2 hours of lecture classes and 2 hours of lab classes, per week.

The 3DMV course was offered, since we considered that interested students would benefit from additional exposure to advanced topics in the Computer Graphics area; moreover, medical imaging is a long-established research area in our department with relevance in many graduation and post-graduation projects and/or R&D activities. Students had only an introductory background on the fundamental CG concepts obtained in their third year Human-Computer Interaction course, but had no experience in using any CG API.

The main topics addressed throughout the course were:

1. Review of Computer Graphics fundamentals
2. Introduction to OpenGL (Lab)
3. Geometric Modeling (Polygonal meshes and free-form curves and surfaces)
4. Techniques conducing to higher realism
5. Introduction to Volume Visualization (Surface extraction and Direct Volume Rendering)
6. Introduction to VTK (Lab)

Regarding the lab classes, the first half of the semester was dedicated to OpenGL, the second half to VTK. OpenGL was used to illustrate the CG and Geometric Modeling concepts taught during the lectures, and to provide the students with their first hands-on experience using a CG API.

The fundamentals of VTK were introduced during lab classes and consolidated using a sequence of practical exercises developed for each class. The six classes used to introduce VTK were on the following topics:

1. **First examples, interactors, cameras and lighting:** compiler configuration; visualization of a simple cone; using interactors and different interaction techniques; using several cameras and light sources.
2. **Actor properties, multiple actors and renderers, transformations, shading and textures:** modifying actor properties (color, opacity, etc.); managing multiple actors in the same or in multiple renderers in the same window; using transformations to change location and orientation; applying different shading techniques and textures.
3. **Observers and callbacks, glyphing and picking:** callbacks and event management; simple examples of glyphing, picking of objects and coordinate visualization.
4. **Widgets, implicit functions, contouring and probing:** introduction to the use of widgets; definition and visualization of quadrics using contouring; probing of a quadric with a plane and visualization of the resulting isolines.
5. **Visualization of 2D images, visualization and clipping of polygonal data:** manipulation and visualization of 2D images; importing VRML polygonal data and simple manipulation of the resulting polydata objects.
6. **Visualization of non-structured grids and volumetric data:** creation and manipulation of a simple non-structured grid; association of scalar information to data; visualization and reslicing of medical data.

In addition to the work carried out during their lab classes, each group of two students was required to develop a visualization mini-project, corresponding to about four weeks of after-class work. Most of the projects implied that the students had to visualize data from different sources using VTK, and create tools and widgets using the library to provide additional visualization capabilities (such as slicing or probing).

Contacts with other departments of the university were made, in order to detect data visualization needs where VTK could be helpful. Several data sets were proposed to our students ranging from electromagnetic radiation data from antennas, to medical data (brain or lung imaging) and physical processes (water flow around a ship's hull or temperature within an industrial oven).

The main idea was to give the students real data sets to visualize, and thus increase their motivation. For some of the problems and data sets, visualization tools already exist, but don't seem to completely satisfy the final users: our colleagues showed an increased interest when interviewed on the issue, since they saw the possibility of influencing the design of the developed applications and directly participating in the specification of their features, instead of being limited to the use of existing commercial application software with limited possibilities.

We will now present some of the most successful visualization applications developed by the 19 students that attended the 3DMV course.

A — Brain data visualization

One of the most interesting projects consisted in developing an application to visualize, in an integrated way, data originating from different brain imaging and signal modalities, namely, MRI and SPECT data, EEG data, and electrical dipole data. The latter two modalities provide time-varying data sets.

The work was divided into three sub-tasks, each one allocated to two students. Given the common platform (VTK), each group implemented the visualization of a different type of data. The entire work was integrated into a single application, which allows the user to easily switch between different data sets and visualization methods.

Volumetric MRI and SPECT data

One group of students had to visualize, in the same window, previously registered MRI (Magnetic Resonance Imaging) and SPECT (Single Photon Emission Computed Tomography) data. This simultaneous visualization provides doctors with important information about the location and intensity of brain activity.

A surface extracted from the MRI data is presented, as well as SPECT data shown as red surfaces. The interface gives the user the possibility to activate up to three cutting-planes (horizontal, coronal and sagittal) to visualize cross-sections of the registered data (Figure 1).

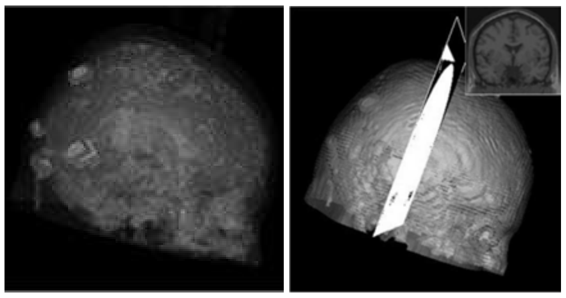

Figure 1: *MRI and SPECT data (left) with different opacity values, and coronal plane with corresponding slice (right).*

EEG data

The EEG (Electroencephalogram) measures the electrical brain activity through time, using several electrodes placed on the head of a patient.

Using a mesh model of the human head, a second group of students was asked to visualize the location of the electrodes, as well as the EEG signal values on the patient's head. Electrode locations are represented as white spheres with an associated label. The color at each mesh vertex is defined by the signal value of the closest electrode. This results in a final representation with different colors associated to the electrical potential variations on the patient's head (Figure 2).

Since temporal information is available, the user can also navigate through different acquisition times and observe the evolution of the EEG data.

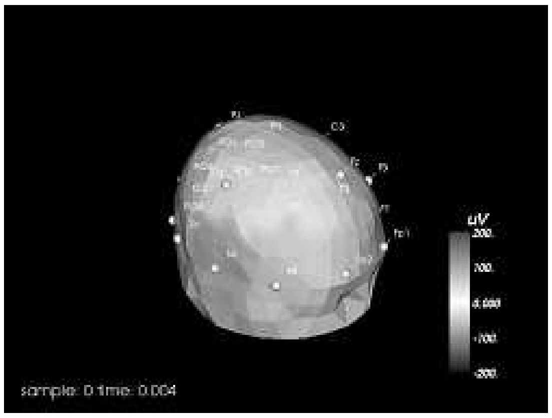

Figure 2: *Mapping EEG data on a model of a patient's head.*

Sources of electrical activity (dipoles)

A third group of students had to represent the estimated location of the sources of electrical activity within the brain (dipoles). The dipoles are represented as arrows within a surface model of the patient's head. The data is read from preprocessed files describing the sampling frequency, the location and the orientation of the dipoles. In addition to these features the application can also display, with varying colors, different groups of dipoles. As for the EEG data, the user can also go forward or backward in time.

Given the large discrepancy between the magnitudes of the dipoles, two visualization modes are available. In the first, the length of the arrow depends on the dipole magnitude; in the second, all arrows are shown with the same size to simplify the analysis and detection of groups and patterns (Figure 3).

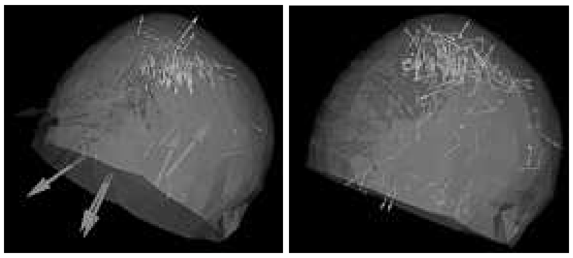

Figure 3: *Two modes of dipole representation.*

B — Football game visualization in a VR environment

Another mini-project developed in the scope of the elective 3DMV course was *footVR*: the goal was to visualize the dynamics of a football game using a Virtual Reality (VR) environment under development at our laboratory.

The application reads the logfile from a simulation of a robotic football game and allows the user to visualize the game in the VR environment. The developed software allows watching the game (robots/players are represented as simple triangular prisms) either in a desktop, by controlling the viewing angle (there is an option to follow automatically the ball), or in the VR environment composed of a Head Mounted Display (HMD) and a tracker. In the latter, motion of the user's head is registered and the camera parameters updated accordingly, as shown in Figure 4.

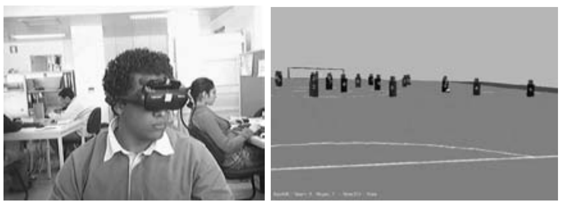

Figure 4: *User wearing the HMD (left), and his view of the game (right).*

C — 3D triangulation of point clouds

This mini-project goes beyond direct data visualization, since students were asked to apply additional VTK features, namely, the Delaunay triangulation algorithm.

Starting with 3D point clouds acquired with a prototype 3D scanner [DMS04], the first objective of the project was to visualize them with VTK. Additionally, students also had to use the triangulation algorithms implemented in VTK to triangulate a point cloud, resulting in a polygonal model of the acquired data. A snapshot of the final application is presented in Figure 5.

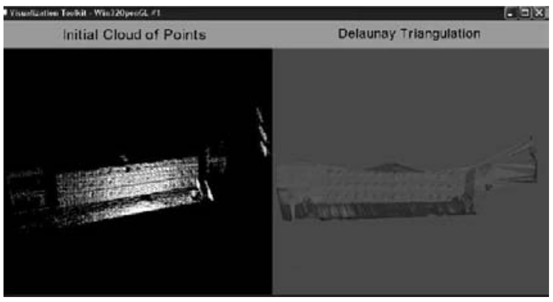

Figure 5: *3D point cloud (left) and final triangulated model (right).*

D — Visualization of water pressure and velocity around a vessel's hull

The objective of this mini-project was to develop visualization tools to analyze the water flow around a vessel's hull. The data consisted of the coordinates of sampled points (unstructured grid), as well as pressure and velocity values.

Hedgehogs were used to represent velocity data and pressure was converted to a structured grid through splatting [SML98]. The final application gives the user the possibility to manipulate a cutting-plane where the pressure data is displayed through color mapping. A view of the final visualization is presented in Figure 6.

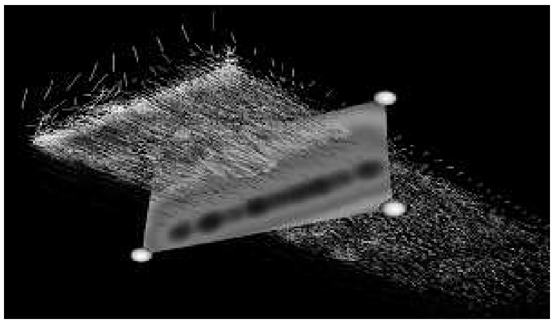

Figure 6: *Visualization of velocity and pressure around a vessel's hull.*

3. CG Lab Course for M.Sc. Students

Also during the first semester of 2005/2006, VTK was used in a different context: in the scope of a post-graduation course that integrated a Computer Graphics profile offered at M.Sc. level. Within this profile, five courses were proposed to the students, three mandatory and two elective courses. Electives were homogenization courses meant to give a basic education in Computer Graphics or Human-Computer Interaction.

Four of the courses were based on lectures:

1. Computer Graphics Topics or Human-Computer Interaction Topics (2 hours/week): basic courses on Computer Graphics or Human-Computer Interaction for students with no previous background on these areas.
2. Geometric Modeling (2.5 hours/week): an introductory course addressing free-form modeling using Bézier, B-Spline, and NURBS curves and surfaces, as well as introducing the use of polygonal mesh models.
3. Visualization (2.5 hours/week): an introductory course addressing data characteristics, visualization techniques and algorithms for 1D, 2D and 3D scalar data, evaluation of visualizations, as well as an introduction to the human visual system and perceptual aspects relevant to Visualization.

In addition to these lecture-based courses, there was an integrated laboratory course of 4 hours/week, meant for the students to apply and develop the concepts acquired throughout the other courses. Given the audience, M.Sc. students that are supposed to be more independent than graduation students, the laboratories were organized as to introduce a series of CG tools and libraries, progressing from the SVG and VRML languages to VTK. The laboratory course was organized as follows:

1. SVG [FGH04] — 2 hours
2. VRML [ANM97] — 4 hours
3. OpenCV [Ope01] — 2 hours
4. OpenGL and GLUI [WNDS99] — 6 hours
5. VTK [SML98] — 8 hours
6. Project development — ca. 12 hours

During the lab classes, and associated to the presentation of the main features of each tool/library, students had to accomplish some illustrative tasks, mostly involving some programming. Additionally, there were homework assignments for each one of the presented tools. This model required a significant effort by the students, but gave them the possibility of acquiring a broader overview of some currently used CG tools.

VTK was the last tool presented, since it was considered the most complete and also the one that requires more background and effort to be fully understood and used. The final assignment (i.e., project) proposed to each student was also based on VTK, and the last lab classes (ca. 12 hours) consisted mostly in the supervised design and development of each student's assignment.

Note that this final VTK project was individual work, and whenever possible linked to the student's interests. We present in what follows two of the most successful projects.

A — Medical data visualization

This project had two main purposes: on the one hand, to explore the capabilities of VTK to visualize medical data originating from different sources through the same user interface, and, on the other hand, to test the possibility to integrate VTK within a GUI. In order to attain the former objective, two different types of data sets were given to the student: CT (Computer Tomography) lung data with segmented air bubbles and MRI brain images. Given the background knowledge of the student the second objective was attained using FOX Toolkit [vdZ05].

The final application that can visualize MRI as well as CT lung data is shown in Figure 7. Among other options, it is possible to activate up to three orthogonal planes and select a point on each one of the separate cutting-planes, thus automatically updating the position of the other planes.

Some preliminary experiments were also done with the introduction of a pseudo-haptic feedback [LBE04] in the application: the speed of the cursor is modified as a function of the value of the data below it. This option was interesting to detect the crossing of the border between lung and air bubbles in the CT lung data set.

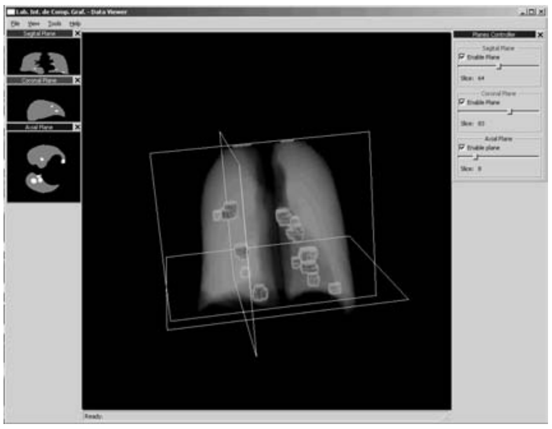

Figure 7: *The user interface developed using FOX Toolkit.*

B — Reconstruction and visualization of 3D brain structures from MRI data

The aim of this project was the reconstruction and 3D visualization of the hippocampus extracted from MRI data. The segmentation of the area of interest from the MRI data set was performed previously, and was not an objective of the work.

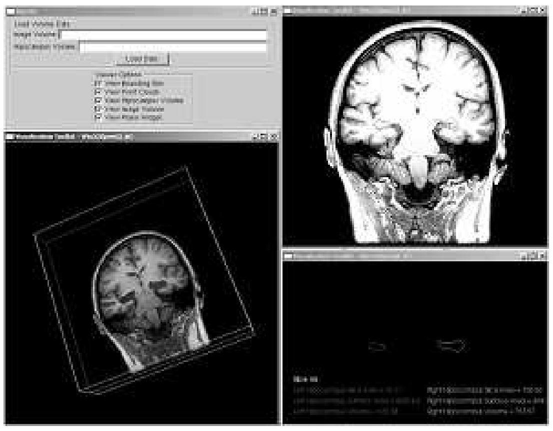

Figure 8: *The user interface for the hippocampus reconstruction and visualization application.*

The student developed a 3D reconstruction algorithm based on the one of Christiansen and Sederberg [CS78], to create a model of the hippocampus from the segmented contours in different slices. Once the polygonal model is constructed, the right and left hippocampus are represented using two different colours, as shown in Figure 8.

4. Conclusion

We briefly presented our experience regarding the use of VTK in the context of two Computer Graphics courses at the University of Aveiro. Our overall evaluation is encouraging. We were even positively surprised by the quality of the work achieved by some of our students (mainly in the *3D Modeling and Visualization* course, where the time allocated to the final work was reduced). The use of VTK gave us the possibility to propose challenging and motivating tasks, and students were able to develop relatively complex applications in a relatively short time, when compared with low-level APIs. This was certainly rewarding for the students.

Students particularly appreciated the use of a higher-level tool such as VTK, which allows developing working prototypes and provides some degree of interaction and appropriate visualization functionalities. For the most successful projects, students were also asked to write a short paper describing the main features of their work, to be published in the internal journal of our department. Despite the fact this additional work was asked for after the conclusion of the semester, almost all students agreed. This exercise was, after all, a nice introduction to more challenging research projects.

For the students, the object-oriented structure of VTK and its modularity were also important advantages. However, many students complained about the lack of good manuals to help them use VTK. The available documentation generated automatically with Doxygen is very often insufficient to clearly understand the features of the classes used, and examples are missing for many functions.

Even with the help of the user's guide [Kit03], it is often difficult to understand at first how VTK classes behave: this is certainly a strong limitation of the toolkit, which does not recommend its use by students with less programming experience or reduced knowledge of the object-oriented paradigm. In some way, using VTK can even be frustrating for a student, since final solutions to some programming or development difficulties are often very short (a few lines of code), but difficult to attain. A possible way to mitigate this problem might consist in providing the students with a set of additional code examples, to help them more easily understand VTK.

The above mentioned shortcomings of VTK force students to a somewhat important effort, during their first contact with the toolkit, in order to overcome first difficulties. For students with a low motivation this was a major drawback, and a few didn't succeed in developing a satisfactory work.

A first analysis of the effectiveness of using VTK in the 3DMV course can be done comparing the evaluation results of the OpenGL and VTK mini-projects: about 20% of the students were weak in both APIs, 20% performed significantly better in VTK than in OpenGL, and another 30% had excellent results in both APIs. This seems to validate our initial idea that using VTK would not be an excessive load for most students, and would be an execellent way of further motivating interested students.

Overall, the experience was conclusive and we intend to keep using VTK in our CG courses, both at undergraduate and graduate level. During 2006/2007 we will conduct a more formal assessment of the educational effectiveness of using VTK.

5. Acknowledgements

First of all, we wish to thank all the students who developed the projects presented in this paper.

A word of gratitude is also due to our colleagues: José Maria Fernandes, José Rocha Pereira, Filipe Teixeira-Dias, Augusto Silva and José Silvestre Silva, who provided us with some of the data sets used in the various assignments.

We thank Samuel Silva for his help in preparing the final version of this paper.

References

[ACSS06] ANGEL E., CUNNINGHAM S., SHIRLEY P., SUNG K.: Teaching computer graphics without raster-level algorithms. In *Panel Session at SIGCSE 2006* (Houston, Texas, USA, 2006).

[Ang06] ANGEL E.: *Interactive Computer Graphics: A Top-Down Approach using OpenGL*, 4th ed. Addison-Wesley, 2006.

[ANM97] AMES A. L., NADEU D. R., MORELAND J. L.: *The VRML 2.0 Sourcebook*, 2nd ed. John Wiley & Sons, Inc, 1997.

[CS78] CHRISTIANSEN H. N., SEDERBERG T. W.: Conversion of complex contour line definitions into polygonal element mosaics. *Computer Graphics 13* (1978), pp. 187–192.

[DMS04] DIAS P., MATOS M., SANTOS V.: 3D reconstruction of real-world scenes using a low-cost 3D range scanner. In *Proc. 4th Conf. on Construction Applications of Virtual Reality (CONVR2004)* (Lisbon, Portugal, 2004), pp. 121–128.

[FGH04] FROST J., GOESSNER S., HIRTZLER M.: *Learn SVG: The Web Graphics Standard*. Self Publishing, 2004.

[FvDF*94] FOLEY J., VAN DAM A., FEINER S., HUGHES J., PHILLIPS R.: *Introduction to Computer Graphics*. Addison-Wesley, 1994.

[GBK*95] GRISSOM S., BRESENHAM J., KUBITZ B., OWEN S., SCHWEITZER D.: Approaches to teaching computer graphics. In *Proc. SIGCSE 1995* (1995), pp. 382–383.

[HB04] HEARN D., BAKER M. P.: *Computer Graphics with OpenGL*, 3rd ed. Prentice-Hall, 2004.

[HS00] HITCHNER L. E., SOWIZRAL H. A.: Adapting computer graphics curricula to changes in graphics technology. *Computers & Graphics 24* (2000), pp. 283–288.

[Kit03] KITWARE: *The VTK User's Guide*. Kitware Inc., 2003.

[LBE04] LÉCUYER A., BURKHARDT J. M., ETIENNE L.: Feeling bumps and holes without a haptic interface: the perception of pseudo-haptic textures. In *Proc. SIGCHI 2004* (Vienna, Austria, 2004), pp. 239–246.

[Ope01] OPENCV: *OpenCV Reference Manual*. Intel Corp., 2001.

[Paq05] PAQUETTE E.: Computer graphics education in different curricula: Analysis and proposal for courses. *Computers & Graphics 29* (2005), pp. 245–255.

[SML98] SCHROEDER W., MARTIN K., LORENSEN B.: *The Visualization Toolkit — An Object Oriented Approach to 3D Graphics*, 2nd ed. Prentice-Hall, 1998.

[vdZ05] VAN DER ZIJP J.: Fox toolkit. *http://www.fox-toolkit.org* (online March/2005).

[WNDS99] WOO M., NEIDER J., DAVIS T., SHREINER D.: *OpenGL Programming Guide*, 3rd ed. Addison-Wesley, 1999.