Using VTK as a tool for teaching and applying Computer Graphics

Paulo Dias, Joaquim Madeira and Beatriz Sousa Santos
paulo.dias, jmadeira, bss@det.ua.pt

DETI / IEETA — University of Aveiro
Campus Universitário de Santiago, 3810-193 Aveiro, Portugal

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 to 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

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 [Grissom95, Hitchner00, Paquette05, Angel06a]. 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, e.g., based on using a higher-level API 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., [Foley94]), do remain useful, advances in hardware, graphical libraries and more recent API-based CG textbooks [Schroeder98, Hearn04, Angel06b] 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, and who had previously been taught the fundamentals of CG, 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 Geometrical 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, 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 [Schroeder98] is an open source, freely available software system 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, the library 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 present briefly the contents of the courses taught, and show some of the most successful projects developed by our students. Finally, we will comment on the results and give some conclusions.

2. 3D Modeling and Visualization

2.1 Description

The “3D Modeling and Visualization” course (3DMV) was introduced for the first time in 2005/2006 as a final year elective in the Computer Engineering course, 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 previously had only an introduction to the fundamental CG concepts in their third year Human-Computer Interaction course, where a brief introduction to Computer Graphics is given; but had no experience in using any CG API.

The main topics addressed throughout the course are:

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 the VTK library 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:

- First examples, interactors, cameras and lighting.
- Actors properties, multiple actors and renderers, Shading, textures and transformations.
- Observers and Callbacks. Glyphing and picking.
- Widgets, Implicit functions, contouring and probing.
- Visualization of 2D images, visualization and clipping of polygonal data.
- Visualization of non-structured grids and volumetric data.

In addition to the work carried out during their lab classes, each group of two students were required to develop a visualization mini-project, corresponding to about four weeks of after-class work, whose theme had to be chosen before the end of the semester. Most of the projects implied the students to visualize data from different sources using VTK, and to 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 needs of data visualization where the VTK would be helpful. Several data 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 to visualize, and thus increase their motivation. For some of the problems and data sets visualization tools already exist, but they seem not 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 of directly interacting with their features, instead of being limited to the use of an existing commercial software with limited possibilities.

2.2 Results of the practical assignments

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 coming from different brain imaging and signal modalities. The work was divided in 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 and visualization methods available.

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 is 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).

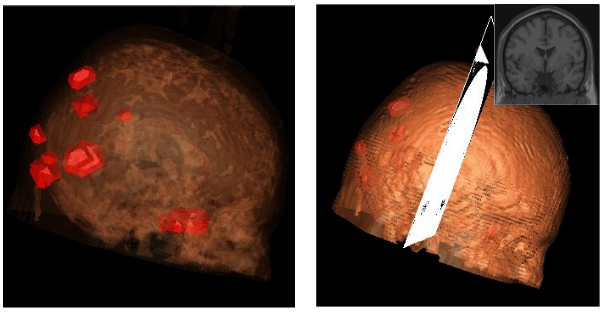

Figure 1: MRI and SPECT data with different opacity and coronal plane with respective slice.

EEG signal

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

The second group was asked to visualize the location of the electrodes registered with the patient's head. Electrode locations are represented as white sphere with an associated label. The color at each mesh vertex is defined by the signal value of the closest electrode. This result 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 the different acquisition times and see the evolution of the EEG signal.

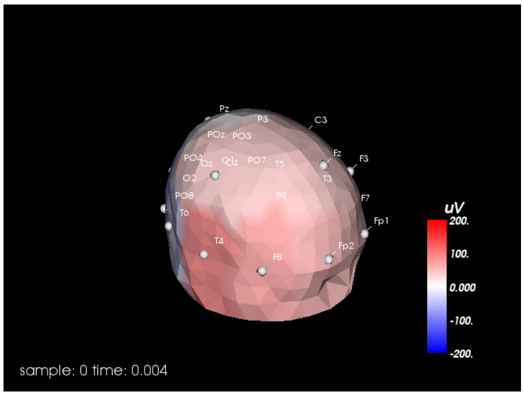

Figure 2: Mapping EEG signal on the patient's head

Electrical activity sources using dipole vectors

The third group of students had to represent the location of dipoles (a estimation of the location of the sources of electrical activity in the brain) as vectors. The dipoles are represented as arrows within a representation of the head of the patient. The data corresponding to the dipoles is read from pre-processed files with the sampling frequency, the location and the orientation of the dipoles. In addition to these features the application can also display with a different color different groups of dipoles. As for the EEG signal, 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).

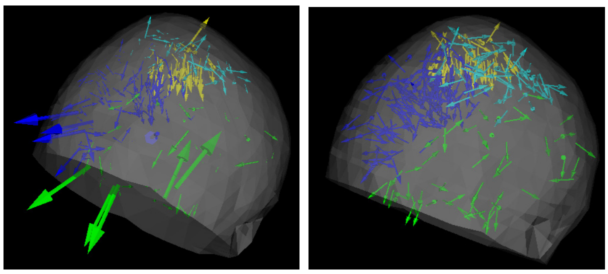

Figure 3: Two modes of dipole representation.

B – Football game visualization in a VR environment

Another project developed in the scope of the elective 3DMV course was "footVR", whose goal was to visualize the dynamics of a football game using a 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 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.

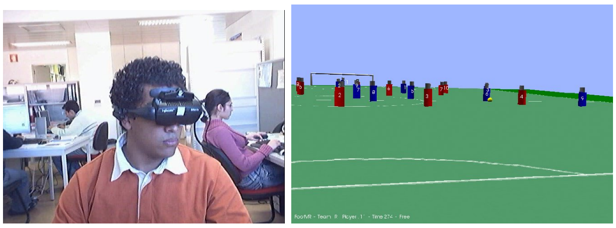

Figure 4: User wearing the HMD, and projected view.

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 [Dias04], the objective of the project was, first, to visualize point clouds using VTK. Additionally, they also had to use the triangulation algorithms implemented in Toolkit to triangulate the point cloud resulting in a polygonal model of the acquired area. A snapshot of the final application is presented in Figure 5.

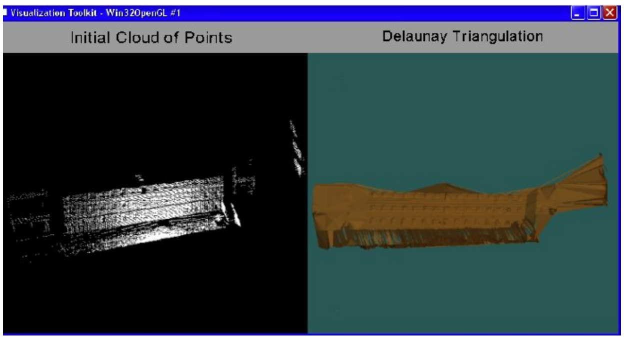

Figure 5: 3D point cloud (left) and final triangulated model (right)

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

The objective of this mini-project was to develop visualization tools to for analyzing 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 and pressure was converted to a structured grid through splatting. The final application gives the user to 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.

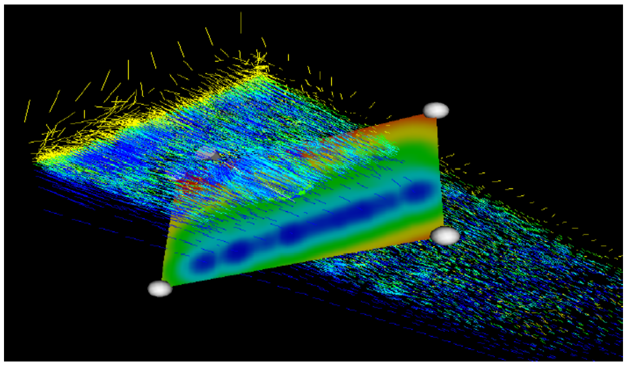

Figure 6: Visualization of velocity and pressure around a vessel's hull.

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 MSc 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.

Three of them were based on lectures:

- Computer Graphic Topics or Human-Computer Interaction Topics (2 hours/week): which were basic courses on Computer Graphics or Human-Computer Interaction for students that had had no previous introduction to the area
- Geometric Modeling (2.5 hours/week): free-form modeling using Bézier, B-Spline, and NURBS curves and surfaces; introduction to modeling using polygonal meshes.
- Visualization (2.5 hours/week): an introductory course on the subject addressing data characteristics, visualization techniques and algorithms for 1D, 2D, 3D scalar data, evaluation of visualizations as well as an introduction to the human visual system and perceptual aspects.

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, MSc 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 following libraries were presented during the laboratories:

- SVG [Frost04] (2 hours)
- VRML [VRML96] (4 hours)
- OpenCV [OpenCV01] (2 hours)
- OpenGL and GLUI [Woo99] (6 hours)
- VTK [Schroeder98] (8 hours)

During the lab classes, and associated to the presentation of the main features of each 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 of the most currently used CG tools.

VTK was the last tool presented, since it was considered as the more complete and also the one that requires more background and effort to be fully understood and used. The final assignment 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 coming 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 datasets 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 the FOX Toolkit.
The final application that can load MRI as well as CT lung images is shown in Figure 7. Among other options, it is possible to activate up to three orthogonal planes and select points in each one of the separate cut planes, updating the position of the other planes automatically.
Some preliminary experiments were also done with the introduction of a pseudo-haptic feedback [Lécuyer04] in the application: the speed of the cursor is modified in 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.

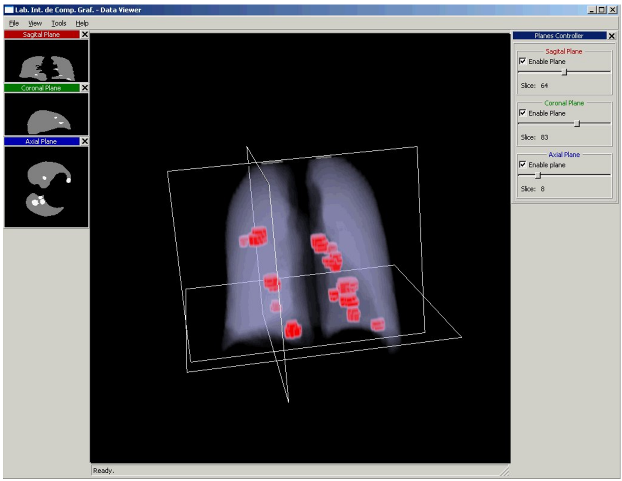

Figure 7: Main aspects of the user interface developed using FOX Toolkit.

B - Reconstruction and Visualization of 3D brain structures from MRI data.

In this project the student had to reconstruct and visualize in 3D the hippocampus extracted from MRI data. The segmentation of the area of interest from the MRI data set was performed previously, and was not the objective of the work, which emphasized the reconstruction and visualization of the hippocampus.

The student developed a 3D reconstruction algorithm based on the one of Christiansen and Sederberg [Christiansen78], to reconstruct the hippocampus from the segmented contours in different slices. Once the polygonal model is constructed, the right and left hippocampus are represented with two different colours, as shown in Figure 8.

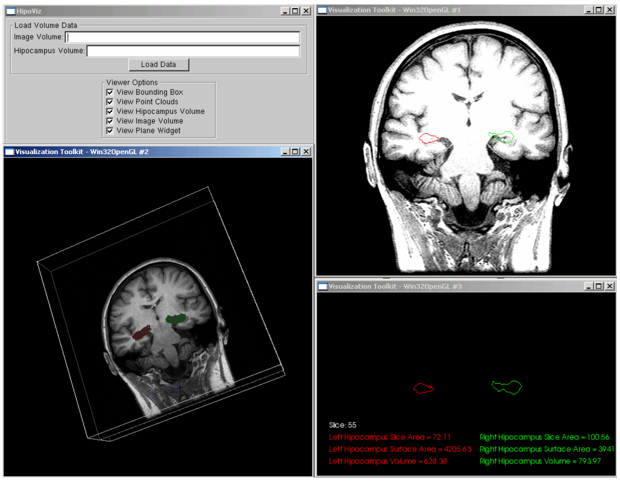

Figure 8: Aspects of the user interface for the hippocampus reconstruction and visualization application.

4. Conclusion

We briefly presented our experience with 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).

Students also appreciated the use of a higher-level tool that allows developing working prototypes providing a certain degree of interaction and appropriate functionalities, in a rather short time. 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 at the end a nice introduction toward more challenging research projects.

The object-oriented structure of VTK and its modularity was also an important advantage in learning and using this tool. However, many students complained about the lack of good manuals to help in the use of 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 [Kitware03], 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 achieve.

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

Still the experience was conclusive and we intend to keep using VTK in our CG courses, both at graduation and post-graduation level.

5. Acknowledgments

A word of gratitude is 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 also wish to thank all the students who developed the projects presented in this paper.

6. References

[Angel06a] E. Angel, s. Cunninggham, P. Shirley, K. Sung. "Teaching Computer Graphics Without Raster-Level Algorithms". Panel Session at Technical Symposium on Computer Science Education (SIGCSE 2006). Houston, Texas USA March 1-5, 2006.

[Angel06b] E. Angel, "Interactive Computer Graphics: A Top-down Approach using OpenGL" (4th Edition), Addison Wesley, 2006.

[Christiansen78] H. N. Christiansen, T. W. Sederberg, "Conversion of Complex Contour Line Definitions into Polygonal Element Mosaics", Computer Graphics, 13, 1978, pp. 187-192.

[Dias04] P. Dias, M. Matos, V. Santos "3D Reconstruction of Real World Scenes Using a Low-cost 3D range scanner". In Proceedings of the 4th Conference of Construction Applications of Virtual Reality - CONVR2004. ADETTI/ISCTE, Lisboa, Portugal. 14-15 September 2004, pp.121-128

[Foley94] James Foley, Andries van Dam, Steven Feiner, John Hughes and Richard Phillips, "Introduction to Computer Graphics", Addison Wesley, 1994.

[Frost04] J. Frost, S. Goessner, M. Hirtzler, "Learn SVG" Self. Publishing May 2004.

[Grissom95] S. Grissom, J. Bresenham, B. Kubitz, S. Owen, D. Schweitzer. "Approaches to Teaching Computer Graphics", Proceedings of the 26th SIGCSE technical symposium on Computer science education, pp 382-383, March 1995.

[Hearn04] Donald Hearn and M. Pauline Baker, "Computer Graphics with OpenGL, 3rd Edition", Prentice Hall 04.

[Kitware03] Kitware Inc, The VTK User's Guide, Kitware Inc, 2003.

[Lécuyer04] A. Lécuyer, J.M. Burkhardt, and L. Etienne, "Feeling bumps and holes without a haptic interface: the perception of pseudo-haptic textures", in Proc. of the SIGCHI conference on Human factors in computing systems, pp.239-246, 2004.

[OpenCV01] Intel Corporation, "OpenCV Reference Manual", 2001.

[Paquette05] E. Paquette, "Computer Graphics education in different curricula: analysis and proposal for courses", Computers & Graphics, 29(2), pages 245-255, Elsevier, 2005.

[Schroeder98] Schroeder, W., K. Martin, B. Lorensen, The Visualization Toolkit- An Object Oriented Approach to 3D Graphics, 2nd ed., Prentice Hall, 1998

[VRML96] The Virtual Reality Modeling Language Specification. Version 2.0, ISO/IEC WD 14772. August 1996.

[Woo99] M. Woo, J. Neider, T. Davis, D. Schreiner, “OpenGL Programming Guide”, OpenGL ARB, 3rd Edition, Addison-Wesley, 1999.