Teaching and Applying Computer Graphics using the Visualization Toolkit

Paulo Dias, Joaquim Madeira, Beatriz Sousa Santos

Dept. of Electronics, Telecommunications and Informatics / IEETA
University of Aveiro
Campus Universitário de Santiago, P-3810-193 Portugal

Abstract

In the last two years, we have been using the Visualization Toolkit (VTK) as a tool for teaching and applying Computer Graphics, for Computer Engineering students who chose to attend the elective “3D Modeling and Visualization” course.

Students have 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 are also required to develop a visualization application based on VTK.

We present first the motivation for using VTK and the main features of the “3D Modeling and Visualization” course. Afterwards, we describe some of the most successful projects developed by our students. Then, we globally analyze the effectiveness of using VTK, and present the results of the questionnaire handed out to the students who attended the course in the last semester.

Key words: Computer Graphics Education, Visualization Toolkit, VTK

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 [7,9,12,2,11,14]. In the past, a bottom-up approach was normally used, where students had to build all necessary code (almost) from scratch. Later, many educators switched to a top-down approach, based on using a higher-level API such as OpenGL or

Email addresses: paulo.dias@det.ua.pt (Paulo Dias), jmadeira@det.ua.pt (Joaquim Madeira), bss@det.ua.pt (Beatriz Sousa Santos).

Preprint submitted to Elsevier 11 March 2007
Java3D [16], with less relevance being given to raster-level algorithms. More recently, courses designed to take advantage of available GPUs and shading languages have also started to be offered [3,15].

Although classical CG textbooks, based on the traditional bottom-up approach (e.g., [6]), do remain useful, advances in hardware, graphical libraries and more recent API-based CG textbooks [13,8,1,4] offer both students and educators the possibility of exploring advanced concepts and developing useful course projects, e.g., for data visualization.

Two years ago, when planning our lectures in the Computer Graphics area for the first semester of 2005/2006, and given the interest shown by prospective students, we decided to offer:

- “3D Modeling and Visualization”, an elective course for students already having some CG background, and presenting them more advanced concepts and offering the possibility of working with de facto CG and Scientific Visualization standard libraries.
- Specialization courses in CG, Visualization and Geometric Modeling, for M.Sc. students, which had to be accompanied by an integrated “CG Laboratory” course of 4 hours per week, providing them with additional hands-on experience. 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.

Since we consider that the top-down approach is more adequate for such 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 Visualization Toolkit, 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 [17], and would have to acquire some knowledge on Data Visualization, perhaps today’s most important application area of CG.

VTK [13] 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 modeling, polygon reduction, mesh smoothing, cutting, contouring, and Delaunay triangulation.
The first results of using VTK as a tool for teaching and applying CG, during the first semester of 2005/2006, were presented earlier at the EUROGRAPHICS 2006 Education Programme [5]. In this paper we report on the experience of teaching the “3D Modeling and Visualization” course during the last two years.

In what follows, we first present the main features of the “3D Modeling and Visualization” course, and then describe some of the most successful projects developed by our students. Afterwards, we globally analyze the effectiveness of using VTK, and present the results of the questionnaire handed out to the students who attended the course in the last semester. Finally, we will present some conclusions.

2 3D Modeling and Visualization

The “3D Modeling and Visualization” course (3DMV) was introduced for the first time in the first semester of 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, given the strong interest shown by prospective students, and we considered that the students would benefit from additional exposure to advanced topics in the Computer Graphics area; moreover, medical imaging and data visualization are long-established research areas in our department, with relevance in many graduation and post-graduation projects and/or R&D activities.

Students attending the MV3D course mostly have an introductory background on the fundamental CG concepts, obtained in their third year Human-Computer Interaction course, but usually have no experience in using any CG API. Therefore, the main topics addressed throughout the course are:

(2) Introduction to OpenGL (Lab)
(3) Geometric Modeling (Polygonal meshes and free-form curves and surfaces)
(4) Techniques conducing to higher realism (Ray-tracing, radiosity, textures)
(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 is 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 encompassed 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 the data; visualization and reslicing of medical data.

In addition to the work carried out during their lab classes, each group of two students is required to develop a visualization mini-project, corresponding to about four weeks of after-class work. Most of the projects imply that the students have 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 university departments are usually made in order to detect data visualization needs where VTK could be helpful. Several data sets have been 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 values within an industrial oven).

The main idea is to give the students real data sets to visualize, and thus increase their motivation. For some of the problems and data sets, visualiza-
tion 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 of directly participating in the specification of their features, instead of being limited to the use of existing commercial application software with limited possibilities.

3 Examples of Student’s Visualization Projects

We will now present some of the most successful visualization applications developed by the students that attended the 3DMV course in 2005/2006 and 2006/2007. The examples described encompass the application areas of medical visualization, physical simulation, acoustic data visualization, engineering and virtual reality.

A — Brain data visualization

One of the most interesting projects carried out in 2005/2006 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 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 physicians 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).
Fig. 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.

Fig. 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 pre-processed 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 clusters and patterns (Figure 3).

![Two modes of dipole representation.](image)

Fig. 3. Two modes of dipole representation.

B — Visualization of water pressure and velocity around a vessel’s hull

The goal of this project, also carried out in 2005/2006, 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 [13]. 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 4.

![Visualization of water pressure and velocity around a vessel’s hull](image)

C — Acoustic data visualization

An interesting set of projects carried out in 2006/2007 consisted in developing applications to visualize the propagation of sound waves through time, given
Fig. 4. Visualization of velocity and pressure around a vessel’s hull.

time-varying data sets containing the results of previously computed sound propagation simulations.

2D sound wave propagation

The application allows the analysis of 2D simulation data from an acoustic modeling software, which allows injecting a sound impulse at an arbitrary point of a given environment and uses a numerical model based on finite differences in the time domain (FDTD) to simulate the interaction among neighboring points and, thus, the propagation of sound waves.

The user can analyze the data in one of two modes: the Sound Wave window (see Figure 5) presents, for a given instant in time $t$, the sound wave values, as well as the associated zero-valued iso-lines; as an alternative the Maximum Values window shows the maximum values recorded so far, for time $t$. As usual, the user can navigate forward or backwards in time.

Fig. 5. Sound wave representation for $t = 39$: representing the 2D propagation, as well as the zero-valued iso-lines.
3D acoustic pressure

Given the results of 3D acoustic simulation, namely pressure values and frequency response for a given 3D environment, at successive iteration steps, the second application allows visualizing (1) sound propagation through time, and (2) the frequency response of the 3D model considered.

In both cases, 3D time-varying data is simultaneously depicted in two representations, using the same color scale: one allows slicing through the model volume using two orthogonal plane widgets, the other shows the iso-surfaces.

Fig. 6. Acoustic pressure in a cubical volume at \( t = 29 \): representation using orthogonal planes (left) and iso-surfaces (right).

D — Visualization of the Oporto underground tunnels

In addition to the usual digitalization of building façades or monuments, 3D laser scanners can also be used to acquire a point cloud describing other man-made constructions. In order to assess particular features, the “artescan” company\(^1\) of Aveiro, Portugal, was charged with scanning the surface of some tunnels of the Oporto underground network (“Metro do Porto”), in order to obtain precise tunnel cross-sections.

Given an example of such tunnel point cloud data, a group of students developed, in 2006/2007, a visualization application allowing the representation of the scanned data in various ways: as point cloud, 3D Delaunay triangulated surface or Gaussian splattered volume. In all of these representations the user can interact with a plane widget to define appropriate tunnel cross-sections (see Figure 7).

\(^1\) See www.artescan.net
Another project developed in 2005/2006 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 8.

Fig. 8. User wearing the HMD (left), and his view of the game (right).
4 Using VTK: an assessment

Since it is not usual to choose VTK as a tool supporting CG lab classes, we decided to assess the effectiveness of using VTK taking into account three different points of view: (1) comparing the evaluation results of the OpenGL and VTK projects developed by our students; (2) analyzing the answers given to a questionnaire handed out to the students in 2006/2007; and (3) collecting our impressions on the use of VTK, given the questions and reactions from the students during the VTK lab classes.

4.1 Evaluation of the OpenGL and VTK projects

In addition to developing and implementing their 3DMV projects: one using OpenGL, the other using VTK, students were required to write a report describing the main steps of the work carried out, as well as present their work to their colleagues. Their projects were then individually evaluated by the authors.

Instead of directly comparing the evaluation results of all the projects, we decided to perform a relative comparison and group the students according to Table 1.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Weak evaluation in both OpenGL and VTK</td>
<td>20%</td>
<td>10%</td>
</tr>
<tr>
<td>Average evaluation in both OpenGL and VTK</td>
<td>30%</td>
<td>10%</td>
</tr>
<tr>
<td>Significantly worse evaluation in VTK than in OpenGL</td>
<td>—</td>
<td>40%</td>
</tr>
<tr>
<td>Significantly better evaluation in VTK than in OpenGL</td>
<td>20%</td>
<td>20%</td>
</tr>
<tr>
<td>Excellent evaluation in both OpenGL and VTK</td>
<td>30%</td>
<td>20%</td>
</tr>
</tbody>
</table>

Table 1
Analysis of the evaluation results of the OpenGL and VTK projects.

While, for 2005/2006, the results in Table 1 seem to validate our initial idea that using VTK would not be an excessive load for most students, and would be an excellent way of further motivating interested students, this is no longer true for the results obtained in 2006/2007: although a comparable group of students performed better, 40% of the students performed significantly worse when using VTK, than when using OpenGL.

Clearly, the results in Table 1 are inconclusive. Although a possible explanation might be the excessive workload of some of the latter students, further work
is needed here: to try to better identify global student shortcomings, as well as particular questions and needs regarding the use of VTK.

4.2 Questionnaire handed out in 2006/2007

In the last semester a questionnaire was handed out to the students at the end of their 3DMV course: the main goal was to gather their own opinion regarding the work carried out when developing the two projects, as well as additional information on their use of OpenGL and VTK, and on their experience previous to the course (80% of the students declared having average or above average programming experience; 80% of the students had never used a CG programming API before).

Regarding the OpenGL project, 87% of the students considered themselves satisfied with the applications they had developed, and two-thirds classified their work as being above average. This overall evaluation is quite similar to our own evaluation of the work of our students.

Regarding the VTK project, the main conclusions that can be extracted from the answers to the questionnaire are:

- 80% of the students considered themselves satisfied with the applications they had developed, and, again, two-thirds classified their work as being above average.
- although there is no significative difference, students globally dedicated slightly less time to the VTK project than to the OpenGL project.
- although there is no significative difference, students globally declared having slightly less difficulty in developing their VTK project than their OpenGL project.
- students considered that, on average, the information available for VTK was mostly insufficient, and poorer that the information available for OpenGL.

Regarding our own evaluation of the work carried out by the students, their answers to the questionnaire revealed higher-expectations that did not match our later evaluation of their VTK projects.

Moreover, we were quite surprised to find out that students seemed to have dedicated slightly less time, and found it slightly less difficult, to their VTK projects, in comparison to the OpenGL projects. In our opinion, the VTK projects would have required more time and effort for the students to attain above than average results. Such an attitude by the 2006/2007 students might explain why the global evaluation in Table 1 is worse than for the 2005/2006 students.
4.3 Our own impression

In spite of the unexpected results in Table 1, our overall evaluation on the use of VTK is encouraging. We were positively surprised by the quality of the visualization applications developed by some of our students, since the time allocated to the final VTK projects was relatively reduced. Using VTK allowed us to propose challenging and motivating tasks, and students were able to develop relatively complex applications in a short time, when compared with other low-level APIs. This was certainly rewarding for most students.

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 [10], 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 did not succeed in developing satisfactory work.

Nevertheless, most students particularly appreciated the use of a higher-level tool 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 that might be proposed to some of them later.
5 Conclusion

We have presented our experience regarding the use of VTK, in the last two years, in the context of the elective “3D Modeling and Visualization” course at the University of Aveiro. In spite of the global results in 2006/2007 being somewhat less than expected, when compared to 2005/2006, our overall evaluation on the use of VTK is encouraging.

The object-oriented structure of a higher-level toolkit like VTK and its modularity, and the visualization and interaction functionalities available, are of advantage for most of the students, who are able to develop complex visualization applications in a relatively short time.

However, it is also clear for us that the learning-curve of VTK might delay or even prevent the progress of some students, and we still have to devise appropriate strategies to mitigate this problem.

Since we intend to keep using VTK in our CG courses, both at undergraduate and graduate level, that will be our next challenge.

6 Acknowledgments

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

A word of gratitude is also due to our colleagues: José Maria Fernandes, José Rocha Pereira, Filipe Teixeira-Dias, Guilherme Campos, Augusto Silva and José Silvestre Silva, who provided us with some of the data sets used in the various assignments. Thanks are also due to the 3D laser digitalization company “artescan”, of Aveiro, for providing us with the Oporto underground tunnels data.

References


