Computer Graphics in the Scope of Informatics Engineering Education

Manuel Próspero dos Santos
Computer Science Department
Faculty of Science and Technology
New University of Lisbon
P-2825-144 Caparica
ps@di.fct.unl.pt

Abstract

At the Faculty of Science and Technology of the New University of Lisbon, the education in Informatics Engineering has been supported by a B.Sc. course and a M.Sc. course. Both of them include Computer Graphics lectures, being the corresponding curricula adapted as much as possible to the other basic scientific matters. On the other hand, the tools and teaching methods have also been accommodated according to the global resources available at the Faculty for those courses, namely the computational ones. The paper presents the methodologies used today for teaching Computer Graphics at the New University of Lisbon, gives the results of some succeeded experiments, refers the most relevant relationships with other subjects and put important questions to the future of Computer Graphics education in the scope of Informatics Engineering.
Keywords: *Computer Graphics, Teaching, Education, Informatics Engineering.*

1. Introduction

This position paper is about computer graphics education at the Faculty of Science and Technology of the New University of Lisbon (UNL), which was the first University conferring a B.Sc. degree in Informatics Engineering in Portugal. The way teaching has been done and the corresponding curricula have been chosen can be the object of a generalized discussion that will eventually lead to some improvements not only in these courses but also in similar ones. As it is well known, the computer graphics field has a multidisciplinary character and is still evolving nowadays. The impact of new products and different techniques that can be used today, such as the Internet, clearly shows another orientation a teacher has to give in order to solve some arising problems.

Although the main focus of the present paper is related with the teaching of computer graphics in B.Sc. and M.Sc. courses, the impact of this subject of study in other areas of Informatics Engineering at the UNL is also emphasized. Therefore, computer graphics education has been oriented, not only in the perspective of the end-user of a graphical application, but specially from the point of view of the underlying design and implementation. Thus, both the API and the basic programming level of a graphics system are to be covered in the courses.

2. Curricula of Computer Graphics

The next subsections of the paper describe the main contents of the different courses in computer graphics at the UNL. In addition to the indispensable textbooks, teacher's own notes are also available for all these courses. On the other hand, since the impact of the Internet has been enormous, not only this tool is filling the common lack of books and journals in our academic libraries but it is also a way to easily find very useful software products.

2.1. The Graduation Course Level

The existence of distinct modules at different levels of interest has shown to be very convenient for teaching computer graphics in the graduation course for Informatics Engineering, which is five years long. In the subsections below one can see the choices made in the real situation.

2.1.1. Fundamentals

Obviously, the first of several modules has always to have a more introductory programme. In the case of the B.Sc. course at the UNL, this module is called Computer Graphics I. It is one semester long and a mandatory subject. The topics corresponding to the covered matters are classified into some relatively diversified and well-known main areas:

- Computer Graphics definition, history and applications
- Main algorithms for output primitive raster conversion and corresponding attributes
- Clipping of lines and polygons
- Geometric primitives in graphical systems
- 2D viewing pipeline: Coordinate Systems and Window-Viewport transformations
- 2D and 3D Geometric Transformations

- Utility functions for geometric transformation composition
- Parallel and Perspective Projections
- Interactive methods for the design of 2D and 3D curves, including evaluation and representation techniques
- Interactive methods for the design of free-form surfaces, including evaluation and representation techniques
- Input logical devices and interaction techniques
- Implementation techniques for the Locator e Pick logical devices
- Main control functions and paradigms in both PHIGS and OpenGL™
- Concepts of hierarchical modeling and edition using PHIGS, OpenGL™ and VRML

2.1.2. An Advanced Course

Based on the previous knowledge of the students about graphics programming and concepts, a second module – called Computer Graphics II – covers a more restricted domain, although the main textbook is the same in both modules [1]. Due to the recent importance of 3D graphics, the chosen topics are mainly associated with photorealism and can be classified into the following groups:

- Introduction to the problem of photorealistic images
- Integration of geometric projections
- View volumes for arbitrary parallel and perspective projections
- The synthetic camera paradigm
- Normalizing Transformation and Perspective Transformation
- 3D clipping in homogeneous coordinates
- Algorithms for Hidden Line and Hidden Surface Removal
- Achromatic color and halftoning techniques
- Fundamentals of Colorimetry
- Color models
- Rules for the use and harmony of colors
- Basic notions of Fotometry
- Illumination models
- Special effects
- Shading methods for polygon meshes
- Texture mapping
- Ray-Tracing
- Radiosity
- Anti-aliasing techniques

The comparison of the curricula of these undergraduation courses, as they were stated above, may lead to some erroneous conclusions. For example, 3D graphics are seemingly postponed to the advanced course, causing an important lack in the basic one. However, this is not entirely true. As a matter of fact, what are really the significant characteristics of 3D graphics that are not covered by the topics of the first course? Note that geometric projections, 3D curves and the design of free-form surfaces, among others, were already included there. For this reason, we think the missing topics are the following ones: HLHSR algorithms, solid modeling, shading and illumination models. But these issues are presented to the students during the semester from an end-user point of view, sometimes with the help of pictures produced by well-known commercial products or extracted from recent movies. This kind of information is easily found in the Internet nowadays. If this is so, why don't we dispense with the second level course? Just because, in that way, the consequent level of students' education would not at all be adequate to an Informatics Engineering graduation. Thus, the specific topics about 3D appearing explicitly in the advanced course are treated in a programmer's point of view and not only by informing the students about the right application product to quickly produce the final result.

2.2. A Post-Graduation Course

One important goal of the M.Sc. course in Informatics Engineering is the specialization in computer science of graduated students. Due to the importance of computer graphics nowadays, this field of knowledge could not stay away.

The actual curriculum is based on the following items, whose main emphasis will be explained in section 4.:

- The role of Computer Graphics
- Main graphical representation methods
- Elements of Computational Geometry
- Geometric Modeling supporting the construction of synthetic images
- Solid modeling and representation schemes, namely B-rep, BSP, Octree and CSG
- Reference Models and API standards

- Concepts, paradigms and programming methods using PHIGS/PHIGS+, OpenGL™, Java and VRML
- Data Visualization, including isosurface generation, particle systems and volume rendering techniques

3. The Training at the Laboratory

An education in computer graphics is far from being complete without any laboratory component. Students are asked to develop some tasks whose level of difficulty depends on the kind of course they are attending: basic or advanced. The pictures included in this section are intended to provide the reader with a pictorial overview of some significant tasks.

Currently, the problems proposed to the students are a mixture between algorithmic and project oriented tasks. An effort is made in order to have all main chapters of the courses present in the final programs developed by the students. Additionally, the theoretical classes have to be synchronized with the explanation of the matters that are needed to find the most convenient solution for each problem.

All the pictures shown in the included Figures are screenshots taken from some programs developed by students at the UNL very recently. The programming activity in computer graphics has clearly benefited from the general purpose platform given by the Intel-based PC nowadays. For this reason, showing already a great evolution from the early recommendations stated in [2], at the UNL there is no longer any visible difference between a computer graphics laboratory and another one equipped for unspecified programming training.

The Microsoft's Visual C++ has been used in a Windows 95 environment [3]. Also the OpenGL™ programs were entirely created having the support of this platform [4], most of them interfacing the window system through the GLUT library [5].

Figure 1 is just one example of window-viewport transformations and planar geometric projections, with the possibility of an interactive control of the parameters.

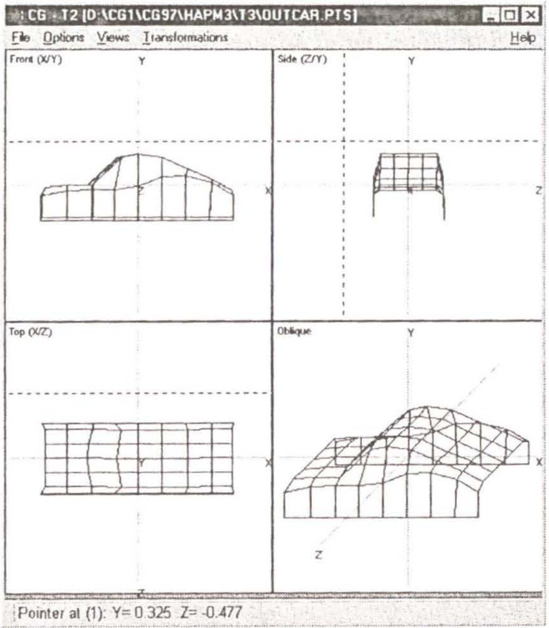

Figure 1. Window-Viewport Transformations and Planar Geometric Projections.

Different methods for designing curves are integrated in a simple editor to be implemented by the students and illustrated in Figure 2.

The fire-fighting truck depicted in Figure 3 can be moved forward and backwards under the user control. Additionally, the stairs have two kinds of independent movements: an elevation produced by a rotation transformation and a variable length achieved by shifting the three components accordingly.

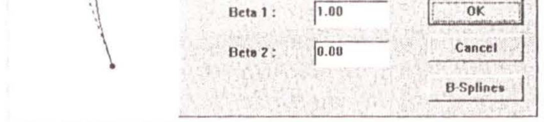

Figure 2. Interactive free-form curve editor.

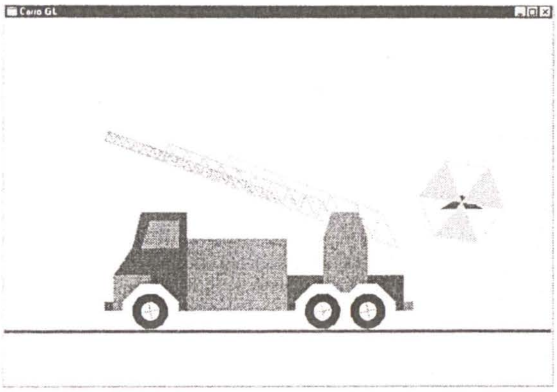

Figure 3. 2D hierarchical modeling with interactivity.

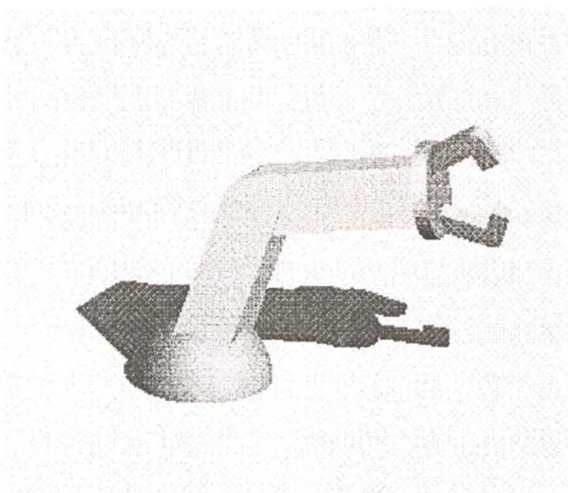

Figure 4. 3D hierarchical modeling with interactivity.

The ability of the students to create the underlying hierarchical model has shown us that not only all important concepts can be learned and practised in a 2D environment but also with additional advantages: normally the results in a 2D space have an easier interpretation and are less time consuming for debugging. The corresponding 3D situation is merely a generalization. This is illustrated in Figure 4. Therefore, the main difficult here is related to the use of illumination and projected shadows. Both Figures 3 and 4 resulted from OpenGL™ programs.

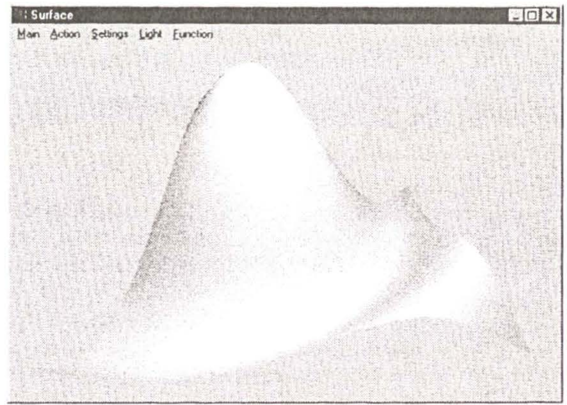

Figure 5. Surface rendering.

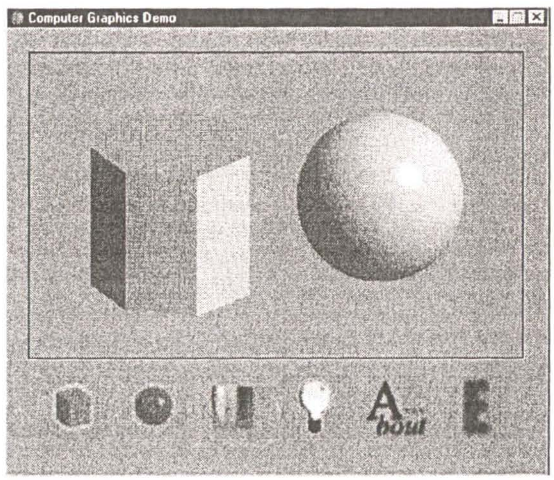

Figure 6. Illumination models and Shading.

The generation of the pictures shown in the Figures 5 and 6 was achieved by writing directly in the frame buffer, thus dispensing with the high-level functions usually found at the API level of a graphical system. This kind of exception to the use of OpenGL™ (see [6]) gives the students a good understanding of the rendering issues and allows them to program any special technique, such as the Phong shading. However, to be a feasible task in a short period of time, some restrictions had to be made: for instance, the output pipeline was simplified by fixing the viewing parameters.

Figure 7 shows us one single frame from an OpenGL™ animation program using B-splines.

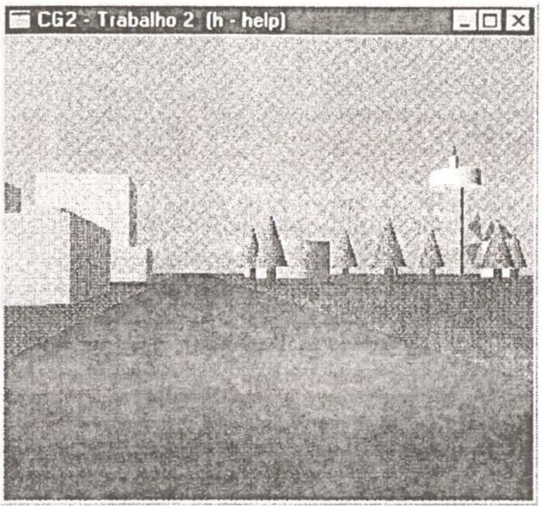

Figure 7. 3D Modeling and Animation.

Finally, it is important to refer that an event-driven programming style is present in all applications written by the students.

4. Problems and Suggested Solutions

In what concerns Computer Graphics I, the topics dealing exclusively with 2D issues are taking nearly 25% of the total time of the actual theoretical classes, being very difficult to make this value lower. One cannot forget that even what we call 3D graphics become 2D at some stage of the complete output pipeline. Nevertheless, it is obvious that the referred percentage would be much more reduced at the cost of sacrificing the explanation of several algorithms and techniques if there were no further courses available, such as Computer Graphics II.

But, in our opinion, that reduction can be achieved by other means, which are compatible with a graduation course having a strong basic kernel of subjects. Let us put the following question: should we teach a programming language (e.g. C or C++) within a computer graphics course? The answer is clearly negative, because the knowledge about programming languages is an attendance pre-requisite for any computer science student. So, why don't we try to take a step forward by moving part of the traditional computer graphics topics into other basic subjects? The main problem here is that a serious compromise must be agreed with the instructors responsible for those subjects. But the foreseen benefits are great for all parts. Consider, for instance, the case where the student is learning the concepts of recursion. What would be the difference if those studies were complemented with a comparison between recursive and iterative algorithms for filling regions of pixels? Computationally speaking, this would be a much more valuable approach than the traditional numerical examples such as the factorial function or the Fibonacci numbers. And there is a lot of additional examples from the area of computer graphics that could be useful to other fields of computer science and don't need long explanations to the new generation of students.

This solution can lead to a natural integration of different subjects of the graduation in Informatics Engineering, but requires the staff responsible for the corresponding lectures to have also a very good background knowledge about graphics. To achieve this goal, the necessary teacher's education is being started at the UNL in what concerns the preliminary programming language courses. We hope that, with the functionality present in Delphi, some basic computer graphics concepts and techniques will be easily introduced at the same time students start to learn Pascal and basic data structures.

There are also several general problems that became visible during the past experience in computer graphics education at the post-graduation level in Portugal. The most important is related with the dissimilarity of the background knowledge of the students, since they usually come from very different Universities in the country. This fact would not be serious if a suitable set of different curricula could be implemented in order to configure the course to the profile of each student. However, this methodology is only feasible when the expected number of students is considerably high. This is not the present case at the UNL and no change is foreseen at short or long term. As a matter of fact, we are talking about a global number between 15 and 20 attendants per year at this M.Sc. course.

As a consequence, the suggested strategy is to elaborate a curriculum where the following compromise is assumed. On the one hand, some basic and general issues in computer graphics have to be quickly addressed in order to uniformize the background knowledge of the students. On the other hand, the choice of the most convenient topics must include the state of the art techniques and the methodologies that can meet the wishes of a student having already a good understanding of computer graphics fundamentals.

However, no intensive programming is normally needed at the curricular part of the M.Sc. courses, since they consist of theoretical lectures and the laboratory is free to be used by the students whenever they want to see the practical aspects of some particular matters. Computational geometry topics and geometric modeling issues are considered to be a good choice from this point of view, because they can be treated at a level as high as we want. Nevertheless, a programming activity is always welcome in Informatics Engineering. Due to its declarative character, VRML was elected as the most privileged candidate to the main tool for developing skill in graphics programming either at a basic or advanced level [7, 8].

5. Conclusions

In general, the evolution of educational methods greatly results from the analysis of past experiences. But these systems are very complex and the staff responsible for teaching is not always having the knowledge about all the relevant parameters. Although many teachers conscientiously assume this fact, that is not the general rule. It thus becomes clear that the lack of helping mechanisms to solve the problem also affects those teachers who want to improve their methods based on the experience from another one.

When trying to find an adequate answer to a specific problem one must obviously consider each methodology that proved to succeed in similar situations before. However, the existence of a small difference between them, even not very significant apparently, may lead sometimes to a completely different and unexpected result. It also happens that the education and learning processes can be highly improved if some details are taken into account. For instance, following an observation by Jansen and Nieuwenhuizen in [9], the first work given to the students attending the laboratory classes in Computer Graphics I set to be a pre-programmed task instead of a simple statement about the problem to be solved. And the gratifying result was an extraordinary grow in the motivation of the students.

Computer graphics is far from being restricted to 3D graphics, no matter their increasing importance today. The way the basic undergraduate course is implemented at the UNL, as described in the paper, does not punish the student who has more difficulties to picture 3D entities and to realize, for instance, their shape and behavior related with movements, intersections, derivatives, volumes and so on.

The paper addresses other important issues. One of them is related with the direct impact the computer graphics education has had in other subjects of Informatics Engineering courses, specially in those matters connected to programming languages, algorithms and data structures. But that is not all, as we can see next. At the UNL there is no computational geometry course yet. The corresponding most significant issues are only referred in the computer graphics courses. However, we are trying to move part of them into a recently created course on complementary mathematics, named Discrete Mathematics, in the second year of the graduation studies. We hope it will be possible to extend this kind of collaboration and integration to other subjects in the near future.

Due to the rapid progress on the computer graphics field, the methods and contents of each course change every edition at the UNL. This is true even for the most basic one. The actual domain of computer graphics is so wide that it is impossible to completely embrace it in any course, especially when a small set of classes is provided. Comparing the UNL courses with the HyperGraph project [10], for example, we can observe that the major set of uncovered items concerns the hardware descriptions and not the software issues. In summary, each educational institution must choose the most relevant topics and the way they can be taught in order to contribute to the specific objectives of the graduation and post-graduation courses to be offered. Lacking for adequate evaluation methods, this is usually a very subjective and difficult task for the teachers.

In 1999, at the UNL, the introductory course for undergraduate students moved from the fourth to the third year of the graduation in Informatics Engineering, reflecting the fact that computer graphics is no longer a complementary subject. As a matter of fact, that course gives such a complete preparation to the students that it can be very useful to several subjects taught in the last two years of the graduation time period. One must note that this kind of change is only possible when the required extensive amount of background in mathematics and programming is guaranteed at the end of the second year of the computer science studies.

Acknowledgements

The following students gave their contribution to the Figures of section 3: Hugo Abreu, Paulo Matos, Telmo Cruz, Paulo Santos, Sérgio Estêvão, João Feliciano, João Canhita, Alfredo Pereira, Cristina Garcia, Pedro Valente, Pedro Lopes and José Domingos.

References

[1] J.D. Foley, A. van Dam, S.K. Feiner, J.F. Hughes. *Computer Graphics — Principles and Practice*. Second Edition in C. Addison-Wesley (1996).

[2] ACM/IEEE-CS Joint Curriculum Task Force, *Computing Curricula 1991*. (February 1991).

[3] C. Petzold. *Programming Windows*. 5th edition, Microsoft Press (1998).

[4] R. Fosner, *OpenGL Programming for Windows 95 and Windows NT*. Addison-Wesley (1996).

[5] J. Neider, T. Davis, M. Woo (OpenGL ARB). *OpenGL Programming Guide*. Second Edition, Version 1.1. Addison-

Wesley (1997).

[6] R. Wolfe. OpenGL: Agent of Change or Sign of the Times?. *Computer Graphics*. 29-31 (November 1998).

[7] A.L. Ames, D.R. Nadeau, J.L. Moreland, *The VRML 2.0 Sourcebook*. 2^{nd} edition. John Willey & Sons Inc (1997).

[8] B. Roehl, J. Couch, C. Reed-Ballreich, T. Rohaly, G. Brown. *Late Night VRML 2.0 with Java*. Ziff-Davis Press (1997).

[9] F.W. Jansen, P.R. Nieuwenhuizen. Computer Graphics Education at Delft University of Technology. *Proceedings of the Eurographics Workshop on Graphics and Visualization Education*, Oslo (1994).

[10] G. Scott Owen (project director). *HyperGraph — Teaching Computer Graphics*: A project of the ACM SIGGRAPH Education Committee, the Hypermedia and Visualization Laboratory, Georgia State University, and the National Science Foundation.
http://www.education.siggraph.org