Using the Internet to Study Programming with OpenGL

Rodica Baciu

"Lucian Blaga" University of Sibiu, 14 Emil Cioran Str, Sibiu, Romania,
rodica.baciu@ulbsibiu.ro

In this paper I want to reveal the prominent part played by the Internet for me and my students in the structuring of the Computer Graphics subject matter and programming with OpenGL. I will try to present the stages in the development of the teaching process to Computer Graphics discipline as well as a review of the subject matter over the last ten years when teaching Computer Graphics represented a permanent adaptation to the students' expectations and to technological challenges.

Keywords
Computer Graphics, OpenGL, Internet, E-Learning

1. A Short History of the Computer Graphics Course

In my opinion, the actual structure of the Computer Graphics course which I have been teaching for almost ten years in the Computer Science Department at "Lucian Blaga" University of Sibiu corresponds to what it is studied in any prestigious university in the world. I think that the present structure of the course in Computer Graphics couldn't have been improved without having access to the Internet. I can say that the Internet was one of the factors that influenced the changing of the course besides hardware and software technology and the students' expectations. I will explain this statement in this paper.

The course had two main parts ten years ago: 2D graphics and 3D graphics. During courses the students were taught fundamentals of 2D and 3D graphics. At practical activities they needed to implement 3D scenes. At that moment, because the department lacked the necessary equipment, we used the 2D graphical library in Borland C++ to create 3D applications. The students were frustrated because they needed to implement graphical algorithms for viewing transformations, projections, illumination, and hidden surface removal. The students did not have enough time to study 3D graphics in a semester and many of them didn't finish their 3D projects. Their applications were not as spectacular as they would want them to be. My conclusions after that first period of Computer Graphics course in our department were:

- The fundamentals are more effectively absorbed by students when they have to implement graphical algorithms. It is impossible for students to implement graphical pipeline without fully understanding matrix transformations and homogeneous coordinates or modelling techniques. Such students will ultimately be more effective users of 3D APIs than students that only partially understand those fundamentals.
- Computer graphics had the reputation of being very difficult and very mathematical, as it originally was. A part of the students, especially those who didn't agree mathematics, were estranged from computer graphics. Because they didn't understand fundamentals they were not able to program graphical applications.

The second stage of Computer Graphics course was represented by the decision to use GDI interface of Windows for practical activities – laboratories and projects. This decision

© South-East European Research Center
(SEERC)

represented an improvement because of transition to applications based on Windows operating system. I continued to teach the students both 2D and 3D graphics. Unfortunately, GDI is a 2D interface and for 3D applications all graphical algorithms as part of an application are needed to be implemented. And this means time and knowledge. Only at the end of the course students did have all the necessary knowledge to finish a 3D project. And this was frustrating for them. The only advantage of this situation was that the best students could control all aspects of implementing a 3D application. Computer Graphics continued to represent for students a discipline requiring a lot of study and time.

In 1999 the improvement of technical equipment in our laboratories made it possible to work with OpenGL. I continued to teach my students both 2D and 3D graphics. During laboratories classes I taught both GDI and OpenGL interfaces to my students. Unfortunately, in this way the remaining time for OpenGL was not enough. On the other hand, the students' interest and enthusiasm for OpenGL was very great.

In 2000 I decided to give up teaching 2D graphics and experiment teaching only 3D graphics. The most important reason for my decision was the fact that the students showed little interest in 2D graphics and a great desire to learn 3D graphical applications programming. And my experiment proved to myself that I can teach computer graphics programming starting directly with 3D graphics and OpenGL. I have preserved the same main structure of the course since 2000.

2. The design of the Computer Graphics course

2.1 Approaches of Computer Graphics course in other universities

The computer graphics field has changed dramatically in last ten years. "OpenGL and its successors have supplanted earlier standards such as GKS and PHIGS to allow students and professionals to do significant graphical work without the need to manage basic algorithms and techniques. In addition, a new generation of inexpensive, high-powered graphics boards with built-in support for OpenGL now form the hardware side of the environment" of graphics [1]. These changes forced a lot of changes in designing Computer Graphics discipline as I showed previously. At this moment there are a lot of opinions regarding to what Computer Graphics discipline must to be [2], [3], [4], [5]. In [1] Cunningham distinguishes three kinds of Computer Graphics course approaches:

- A traditional one that "emphasizes the fundamental algorithms and techniques needed to manage the geometry and rendering directly".
- A new one that emphasizes "geometric and spatial problem representation in class discussions, by using a graphics API such as OpenGL as the basis for student projects and taking advantage of the capabilities such an API can provide, and by keeping the discussion of graphics technology at a level that can help students understand the concepts that underlie the images *without focusing on the details of how these concepts function*".
- A hybrid approach, illustrated by the well-known Angel text [6]. "In this approach, the main subjects of the course are fundamental algorithms and techniques, but the course also includes OpenGL programming and the fundamental principles are expressed using the OpenGL API to carry out class projects. This makes the topic more approachable by the computer science student, but continues to focus on the underlying techniques instead of geometric problem-solving".

Cunningham opts for the second approach. He differentiates three basic groups of people who need computer graphics knowledge: "the group who create fundamental graphics systems, the group who use develop applications and tools using graphics systems, and the group who use graphics applications and tools at a sufficiently sophisticated level that they are well served if they have a basic working knowledge of how graphics works". He argue that "the traditional first course clearly served the system development group best, while providing some service to the application and tool development group, but left out the group of high-level graphics users entirely. The new approach focuses on the application and tool developers group and has a great deal of value for high-level graphics users while not disenfranchising the system developers group."

In [7] Rosalee Wolfe formulates a group of questions regarding to the problem of choosing an approach or other: "Is it desirable to expose students to the tools that will be in prevalent use when they reach the workplace? What portion of employers will be willing to hire graduates to write code when there are $250 graphics cards that already provide equivalent functionality? Does this mean that we want to present some algorithms at a level that allows students to choose the appropriate tool instead of choosing a level that allows students to implement them? Will this depend on the student's choice of employment or further schooling upon graduation? Are there topics that are technology independent and will be important regardless of the next wave of software and hardware to hit the market?"

In [8] Lowther shows that "in contrast with Cunningham's point of view, our approach is to look into the future by incorporating modern as well as traditional topics into a comprehensive introductory course".

In [3] Kelvin Sung defines his concept of the CG design – "Integrating computer graphics concepts in moderately complex applications." He also shows "As educators, we would like to concentrate on fundamental principles and competencies where there is potential applicability throughout students' life and in their careers." I consider very valuable his following assertion: "As newer versions of the software and/or APIs are released, the knowledge students gained must continued to be valid and applicable."

In [9] Peter Shirley argues that for mature programmers taking a single graphics course, the advantages of a high-level approach are amplified.

2.2 Our approach of Computer Graphics course

The aim of the course is to help students gain a deeper insight into computer graphics concepts and OpenGL functionality, while expanding their "tool-box" of useful OpenGL techniques. The most important learning outcome of the course is that students to be able to program graphical applications using OpenGL.

The course was designed to require 84 hours in class. The course is taught with both lecture classes, to present the theory content and laboratory classes where students have hands-on contact with many of the concepts from theory. The laboratory (2 hours/week) and theory classes (3 hours/week) are alternated (students have both classes every week). Students need to develop a final graduation project. During the second part of the semester are allocated 2 hours/week in class for work to projects (14 hours/semester for projects), especially for discussions or asking questions. Certainly students need more hours of extra-class work for their projects. All laboratory activities are developed in groups of 2 or 3 students.

Both the laboratory and the theory classes attempt to follow a top-down scheme in the presentation of course content. I agree with Sung and Shirley [3], who argue that this approach is better suited for teaching mature adults than going up from foundational algorithms such as rasterization to reach 3D images with lighting effects only by the end of the course.

The course focuses from the beginning on 3D computer graphics, instead of 2D. The reasons for this decision are the growth and great availability of 3D hardware and applications, student's marked interest in them, the relatively short course time in class for to study both 2D and 3D graphics.

2.2.1 The Course structure

Table 1 Course description

Theory Theme	**Time**
Introduction. Computer Graphics. Overview of Graphics Systems. 3D Graphics Concepts.	2 hours
Part I. 3D Modelling. Polygon Surfaces. Triangulation. Curved Surfaces (Hermite Curves, Bezier Curves, Spline Curves, Hermite Surfaces, Bezier Surfaces, Spline Surfaces). Constructive Solid-Geometry Methods. Octrees Methods. Sweep Representations. Regularized Boolean Set Operations. Quadric Surfaces.	10 hours
Part II. Displaying of 3D Graphics Scenes	
Color Models. RGB Color Model. CMY Color Model. HSV Color Model. HLS Color Model. Modelling Light Intensities.	2 hours
3D Transformations. Translation. Rotation. Scaling. Shear. Transformation of Coordinate Systems.	2 hours
Viewing in 3D. Projections. Viewing Transformation. Implementing of Viewing Operations. The Viewing Pipeline.	4 hours
Hidden Surface and Hidden Line Removal. Classification of Algorithms. Techniques for Efficient Visible-Surface Algorithms. Algorithms for Visible-Surface Determination (Depth Buffer Method. Scan-Line Methods. Depth-Sorting Method. Area-Subdivision Methods).	6 hours
Illumination and Shading. Illumination Models Surface-Shading. Models for Polygons Shading. Models Related with Visible-Surface Determination. Global Illumination Models.	8 hours
Texture mapping	4 hours
Animation	4 hours
TOTAL	42 hours

Table 2 Laboratory

Laboratory theme	**Time**
Overview of 3D Graphics Scenes (VRML Scenes, VRML viewers). **Assignment:** Browsing 3D graphics scenes and discovering of 3D graphics concepts.	2 hours
Introduction to OpenGL. GLAUX, GLUT and GLU Libraries. Programming with GLAUX. **Assignment:** Moving a rectangle (up, down, left and right**)** using keyboard and mouse.	2 hours
Basic GL Operations. GL Command Syntax. OpenGL Primitives. Colours in OpenGL.	2 hours
Drawing 3D Objects using OpenGL Primitives. **Assignment:** Drawing a circle, a cylinder, a cone and a box.	2 hours
Drawing 3D Objects using GLU Primitives. **Assignment:** Drawing a circle, a cylinder, a cone, a box, a disk, and a pyramid.	2 hours
Hidden Surface Removal in OpenGL. Surface Removal based on Orientation in OpenGL. **Assignment:** Drawing a cube and clipping front faces and then back faces.	2 hours
Viewing. Projections. Clipping. **Assignment:** Moving the origin of the coordinate system. Clipping callouts of a sphere by certain clipping planes.	2 hours
Matrices in OpenGL. Transformations in OpenGL. Composition of Transformations in OpenGL. **Assignment:** Moving the arms of a robot (up-down and front-back). Moving the sun, the moon and the earth (rotation and revolution). Drawing and moving a swing.	4 hours
Parametric Curves and Surfaces in OpenGL. **Assignment:** Drawing a sinus curve by to distinct curves with 2 continuity order in the contact point. Drawing an egg by two surfaces.	2 hours
Lighting in OpenGL. **Assignment:** Lighting the objects from assignment 8.	2 hours
Display Lists in OpenGL. Bitmaps in OpenGL. Fonts in OpenGL. **Assignment:** Writing texts under objects from assignment 8.	2 hours
Texture Mapping in OpenGL. **Assignment:** Texturing objects from assignment 8.	2 hours
Test	2 hours
TOTAL	28 hours

Samples of Project Themes

- The modelling of an airplane and its shadow on the earth. The airplane can be moved, the earth is textured and the scene is lighted.
- The modelling of a car and its shadow on the earth. The car can be rotated on a road, the earth is textured and the scene is lighted.
- The modelling of a fish moving in an aquarium. The simulation of water transparency using blending.
- The modelling of a bolt screwing in a nut. The pieces are textured and lighted. The bolt can be screwed in up and down.
- The modelling of a transmission device with gears. The gears are textured and lighted and can be rotated left and right.
- The modelling of the rack and pinion animation. The pices are textured.
- The modelling of a bowl using a rotational sweep of a Bezier curve. The curve can be designed by interactive moving of control points. The bowl is lighted.
- The modelling of "The Table of Silence" (of Constantin Brâncuşi). All objects have shadows and are textured and illuminated.
- The modelling of "The Endless Column" (of Constantin Brâncuşi). All objects have shadows and are textured and illuminated.

In figure 1 are some captures from students' projects.

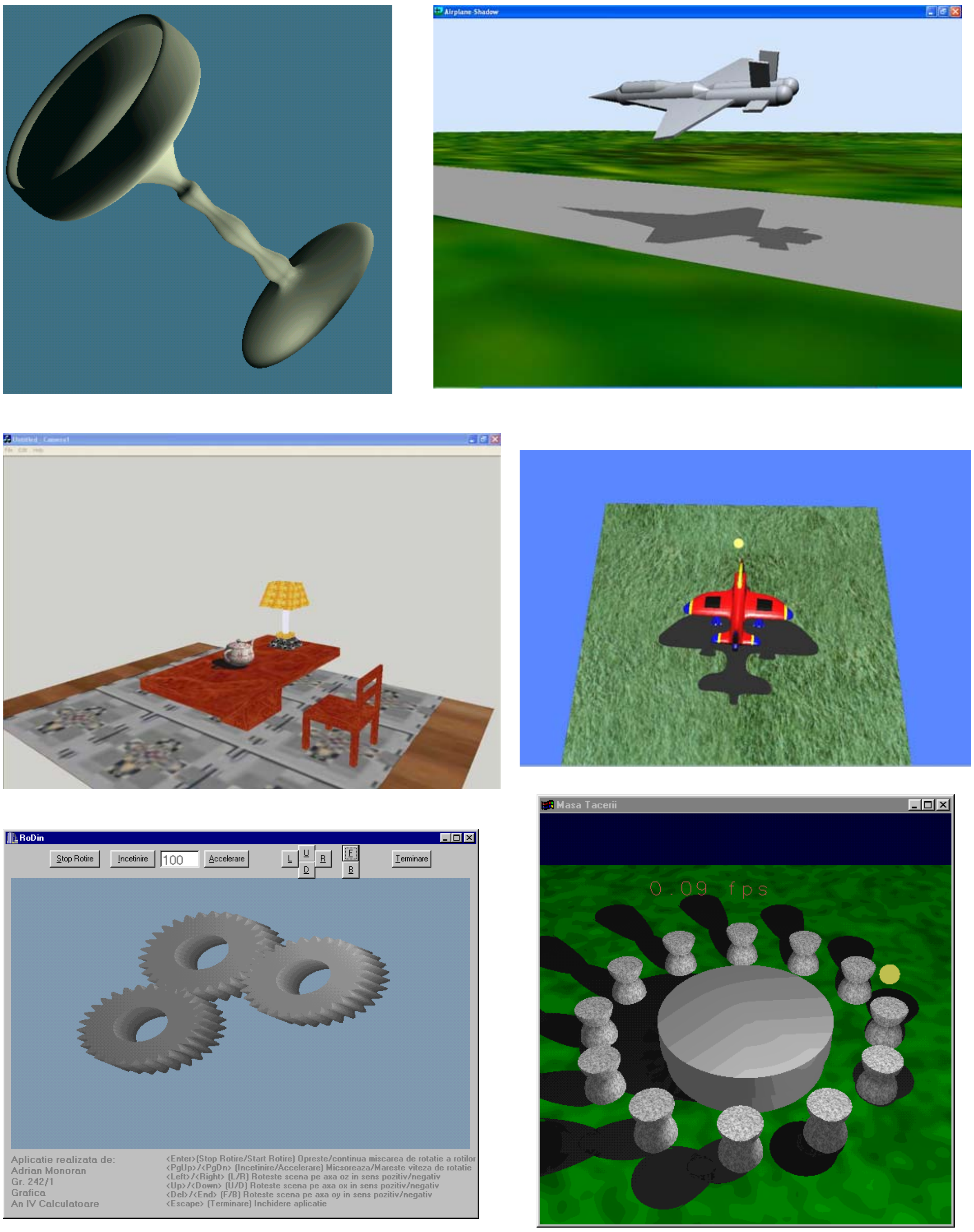
Evaluation (Grading policy)

The semester grade is based upon assignments issued in class, which might involve some small programming exercises (30%), OpenGL project (30%) and a comprehensive final examination (40%). The condition for obtaining the credits is a minimum grade of 5 for all the three components: assignment, project and exam. The final examination assesses essential theoretical knowledge of the students. Every student has to answer to few basic questions from all the chapters of the course.

3. The Students' Feedback to the Computer Graphics Course

The decision taken by me seven years ago - to give up 2D graphics – was avant-garde for that time. The students had time all the semester only to study OpenGL and 3D graphical concepts. But why OpenGL? Firstly, because "OpenGL Application Programmer's Interface (API) is easy to program and OpenGL's well defined architecture allows a coherent presentation of implementation issues" [11]. Second because students wanted to study OpenGL. The students' reaction encouraged me to stick to my decision. They have been enthusiastic because they had been learning 3D programming from the very beginning. After three laboratory classes, students were able to program 3D applications. Even if at that stage their scenes were neither illuminated nor textured it was great to see a 3D object that could be moved or rotated if you pressed arrow keys, for example.

The first contact with OpenGL is surprising for students. They are astonished about what the API can do for them and about the simple way they can program 3D applications. This increases their enthusiasm and desire to program spectacular applications. And indeed is a satisfaction to see how extraordinary projects they can do. Another reason for my students' enthusiasm is the multitude of items of information about OpenGL that can be found on the Internet.



Aplicatie realizata de:
Adrian Monoran
Gr. 242/1
Grafica
An IV Calculatoare

<Enter>(Stop Rotire/Start Rotire) Opreste/continua miscarea de rotatie a rotilor
<PgUp>/<PgDn> (Incetinire/Accelerare) Micsoreaza/Mareste viteza de rotatie
<Left>/<Right> (L/R) Roteste scena pe axa oz in sens pozitiv/negativ
<Up>/<Down> (U/D) Roteste scena pe axa ox in sens pozitiv/negativ
<Del>/<End> (F/B) Roteste scena pe axa oy in sens pozitiv/negativ
<Escape> (Terminare) Inchidere aplicatie

Figure 1 Captures from students' projects

At the end of the course every student completes an anonymous questionnaire regarding his opinion about the course. These questionnaires provide me a feedback useful in teaching the course.

Judging after student's marks and also after their opinions regarding Computer Graphics course the most attractive part of the course is programming OpenGL. The most part of the students don't agree learning theoretical part of the course. I tried to make the lectures more attractive, to make good explanations, to involve the students in the teaching process. But the students have their opinion that in the future they need to be good programmers not good theorists. Instead, my opinion is that the final exam helps them to structure their programming knowledge.

The grading policy is a compromise between my opinion and the students' opinions regarding expecting learning outcomes of the course. The student's knowledge of programming in OpenGL contributes by 60% to the final mark and their theoretical knowledge only by 40%. The best of the students obtain good marks to all kind of the examination. A great part of the students obtain very good marks for programming in OpenGL but poor marks to theoretical exam. Unfortunately there are few students who don't have a good base in programming and for this reason have poor marks to programming in OpenGL.

As a conclusion, I think is very important to be opened to student's demands but like a good parent who give his child not only he request but all he know is good for his future. And I think that for a solid formation of the future specialists only programming skills are not enough. They also need a deep cognition of fundamental aspects of Computer Graphics.

4. The Internet as a Huge Laboratory

The above mentioned ideas could not have been put into practice if my students and I had not had access to information about OpenGL. We can not say that we had access to OpenGL books in our libraries or bookstores. In fact I wrote the first Romanian book about OpenGL in 2005. My first contact with OpenGL was through Visual C++ 5.0 that contains a lot of OpenGL samples and a very good description of every OpenGL function. A great advantage was the fact that OpenGL API was free from licensing requirements, and its developers gave granted to specification [10] from the beginning. Now information about OpenGL is inexpensive and easy to obtain through the Internet. Actually the role of the Internet for practical activities of the course was huge.

As far as OpenGL learning is concerned, the Internet has become a platform opened to all users. There exists a WWW OpenGL home page [12]. Besides the sites that are at the users' disposal by specialized companies like Silicon Graphics [13], lessons and programs created by students have been developed on the Internet as well as tutorials that have been improved by users all over the world, discussions forums, and so on. For example, the NeHe site [14] contains lessons and programs related to studying OpenGL programming. The visitors of the site contribute to it with programs and own papers. Another example is Nate Robins's site [15]. He worked for Silicon Graphics and Evans Sutherland and developed a lot of OpenGL programs and an extraordinary OpenGL Tutorial. Tutorial programs demonstrate the basic OpenGL functionality by allowing the user to modify the parameters of a function and to see the effect on the scene.

In the OpenGL field the students are not only information consumers from the Internet but they have begun to contribute to improve the learning content of this area. I consider this a good sign shown by the students for OpenGL; it is also a good sign that they are satisfied with the spectacular applications OpenGL allows, too.

I have noticed a transition from the laboratory to the Internet over the last 7 years. I have also noticed, in many situations, which although in the beginning of the course students don't know anything about OpenGL, after a few laboratory classes they learn in advance laboratory topics and also start to train themselves in using the Internet.

The Internet is also a source of information regarding the way of programming the interface of their programs. In most cases, the students' projects use other interfaces than I use in laboratory classes. I use GLAUX during laboratory classes because Visual C++ installs itself this library and so I am not dependent on the laboratory room and the window and interface code is very simple. But this library can not create complex interfaces. Well, in their projects, the students use OpenGL Utility Toolkit (GLUT) programming interface API [16], GLUT based C++ user interface (GLUI) [17], or Windows API like as in NeHe applications [14]. Sometimes they choose to work with Builder C++. Their reason for choosing these libraries is the possibility to create interactive interfaces for their projects.

For students, in the OpenGL area, the Internet constitutes a source of courses, a huge library of programs, a discussions forum and a place for showing their results. The students not only use the Internet but they also transform it according to their needs. There are a lot of sites controlled by the students. The students write their own tutorials and programs, answers to questions on the Internet. In this context of e-learning, it seems that there is no place for the teacher. The teachers control the teaching activities only during laboratories or if they move as well their teaching activities on the Internet. In this respect, teachers' sites that place courses, applications, explanations, own libraries and so on at the students' disposal of others sites' visitors can be given as an example [18].

5. Conclusions

In these ten years of teaching Computer Graphics the structure of the course attended the changes of the field and the changes in the discipline structuring in the world. The students' interest for this field grew up year by year. The students consider the discipline important for their career. They use computer graphics knowledge in their license projects or in other disciplines' projects.

The above presentation revealed the significant role of the Internet in the "Computer Graphics" course design. If in interactive applications or in on-line learning systems the students' activities are controlled by the teacher there are situations like the ones presented in this paper when from a passive element of the learning process the student becomes an active element. It is he who himself finds information, finds tutorials or applications, finds answers to his questions but at a certain moment he becomes a trainer for other students, he answers to other students' questions, he shares his experience with other students or he himself creates his own training site. But we have to notice that this process takes place only because the students are interested in the OpenGL subject and this subject is a generator of satisfaction through OpenGL applications. Otherwise the students wouldn't invest time and energy in using the Internet to communicate with other students who are concerned with the same problems.

References

1 Cunningham S. Powers of 10: The Case for Changing the First Course in Computer Graphics. Thirty-first SIGCSE, Technical Symposium on Computer Science Education, March 8 to March 12, 2000, Austin, TX, pp. 46-49.

2 Wolfe R. New Possibilities in the introductory Graphics Course for Computer Science Majors. ACM Computer Graphics, May 1997, pp.35/39.

3 Angel E, Cunningham S, Shirley P, Sung K. Teaching Computer Graphics without Raster-Level Algorithms. ACM SIGCSE'06, March 1-5, 2006, Houston, Texas, USA.

4 Brown J R, Burton R P, Cunningham S, and Ohlson M. Varieties of Computer Graphics Courses in Computer Science. In Papers of the Nineteenth SIGCSE Technical Symposium on Computer Science Education. page 313, 1988.
5 Hitchner L, Cunningham S, Grissom S, and Wolfe R. Computer Graphics: The Introductory Course Grows Up. In The Proceedings of the Thirtieth SIGCSE Technical Symposium on Computer Science Education. pages 341–342, 1999.
6 Angel E. Interactive Computer Graphics: a Top-Down Approach with OpenGL. Addison-Wesley, 1997
7 Wolfe R. Open GL: Agent of Change or Sign of the Times ?. ACM Computer Graphics Nevember 1998, pp.29-31.
8 Lowther J L, Shene C K. Rendering + Modeling + Animation + Postprocessing = Computer Graphics. Proceedings of the Seventh Annual Consortium for Computing in Small Colleges: Midwest Conference, Valparaiso, Indiana, October
9 Sung K, Shirley P. A Top-Down Approach to Teaching Introductory Computer Graphics. Computer & Graphics, Vol. 28, Issue 3, PP. 383-391, June 2004 (full-length paper based on ACM SIGGRAPH 2003 Educator's Program Conference Paper).
10 OpenGL & Utility Library Specifications available at:
http://www.opengl.org/documentation/spec.html
11 Angel E. Teaching a three-dimensional computer graphic class using OpenGL. ACM SIGGRAPH Computer Graphics, vol. 31, no. 3, pp. 54-55, August 1997.
12 OpenGL home page available at:
http://www.opengl.org
13 Silicon Graphics page for OpenGL available at:
http://www.sgi.com/products/software/opengl
14 NeHe page for learning OpenGL programming available at:
http://nehe.gamedev.net/
15 Nate Robin's site for OpenGL available at:
http://www.xmission.com/~nate/tutors.html
16 GLUT programming interface API available at:
http://www.opengl.org/resources/libraries/glut
17 GLUI user interface library available at:
http://glui.sourceforge.net/
18 Nicole Deflaux Terry's OpenGL tutorial is available at:
http://www.eecs.tulane.edu/www/Terry/OpenGL