GRAFICA - A COMPUTER GRAPHICS TEACHING ASSISTANT

Andreas Savva, George Ioannou, Vasso Stylianou, and George Portides, University of Nicosia
Cyprus

Abstract

This paper presents an interactive software program, called *Grafica*, which is used as a tool to introduce important concepts and algorithms of computer graphics to computer science and computer engineering students. The software has been specially developed for educational purposes, and generates surfaces defined by an arbitrary topology of control points using subdivision splines. The user can apply various transformations (translation, rotation, scaling), lighting, rendering, parallel or perspective projection, geometric modelling using the subdivision method of Catmull and Clark (1978), and in general illustrate visually a large number of computer graphics concepts.

Keywords: education, teaching tools, visual learning, computer graphics concepts.

Introduction

The specialized mathematics knowledge covered in a Computer Graphics course is usually presented to students in an abstract way. Albeit, computer graphics is indeed an application of such abstract mathematics, students may find it hard to link the two together. Concepts such as 2D and 3D transformations, image projection, hidden-line removal methods, lighting, rendering, ray-tracing, and even more advance topics such as surface modelling, involve mathematics that a student will learn in a number of university courses, such as, Linear Algebra, Calculus and Physics.

Usually a student has a vague picture of the actual output of any application of such mathematics knowledge. Educators may observe that students understand abstract mathematics concepts, if and when they are assigned a software project to implement these concepts, and this understanding could still remain vague until the very last stages of the implementation. It must also be noted that in a 12-week Computer Graphics course students cannot be assigned projects involving all the areas mentioned above. However, it has been claimed that students find it very effective to understand theoretical concepts when the educator provides real-time applications resulting from the implementation of these concepts (Jackson, 2004; Taylor, Duffy, & Hughes, 2007).

A number of teaching-assistant tools have been developed throughout the years in order to assist educators in their lectures (Buchau, Rucker, Wossner. & Becker 2009; Folorunso & Akinwale 2010; Aliasgari, Riahinia, & Mojdehavar, 2010).

This paper presents a user-friendly software program, namely *Grafica* that has been developed to assist students in understanding basic concepts. This is achieved, through the real-time application of these concepts in class.

Software Development

The purpose of the development of Grafica is to illustrate a large number of basic concepts presented in Computer Graphics. A student takes such a course either the third or the fourth year of an undergraduate degree in Computer Science or Engineering, and can use this application to:

- visualise a three-dimensional (3D) scene,
- associate the scene with light sources and camera,
- load 3D models into the scene, from "model data files,"
- apply transformations to the entire scene, models, light sources and camera,
- render the scene and models with material and light,
- apply various draw options for models (points, wireframes, polygons), and
- smooth objects using the Catmull-Clark subdivision surface scheme.

Integrated Development Environment (IDE)

Microsoft Visual Studio [Microsoft Corporation, 2013], along with the C# programming language has been employed for the development of Grafica. Visual Studio is an integrated development environment. It can be used to develop console and Graphical User Interface (GUI) applications including windows forms, web and mobile applications. Additionally, the Tao Framework libraries for Open Graphics Library (OpenGL) [OpenGL, 2014] were incorporated, for gaining access to OpenGL functionality.

Open Graphic Library (OpenGL)

OpenGL is the premier environment for developing portable, interactive 2D and 3D graphics applications. OpenGL has become the industry's most widely used graphics Application Programming Interface (API), bringing thousands of applications to a wide variety of computer platforms. OpenGL fosters innovation and speeds application development by incorporating a broad set of rendering, texture mapping, special effects and other powerful visualization functions. Developers can leverage the power of OpenGL across all popular desktop and workstation platforms, ensuring wide application deployment.

Grafica – Software Design

Grafica has been developed based on the Object Oriented Programming (OOP) approach. Thus, it consists of a collection of classes that encapsulate their data members and methods while they interact with each other as directed by the application functional requirements. The main class is the *main form*, which serves as the mediator between the user and the application. The main form offers the Graphical User Interface (GUI) and hosts the necessary controls enabling the user to interact with the application. It is divided into three areas as shown in Figure 1. The *Toolbars Area*, where the functions and tools are located, the *Properties Panel Area,* which has a separate page for

each type of object, where the various properties of the objects can be viewed and modified by the user interactively, and finally the *Scene Display Area* where the scene with the various models, light sources and the camera are rendered.

Figure 1. Grafica 3D Studio User interface.

Loading Objects

Upon loading an object file the model is loaded and positioned at the centre of the scene. It must be noted that the user can load more than one model into the scene as illustrated in Figure 2.

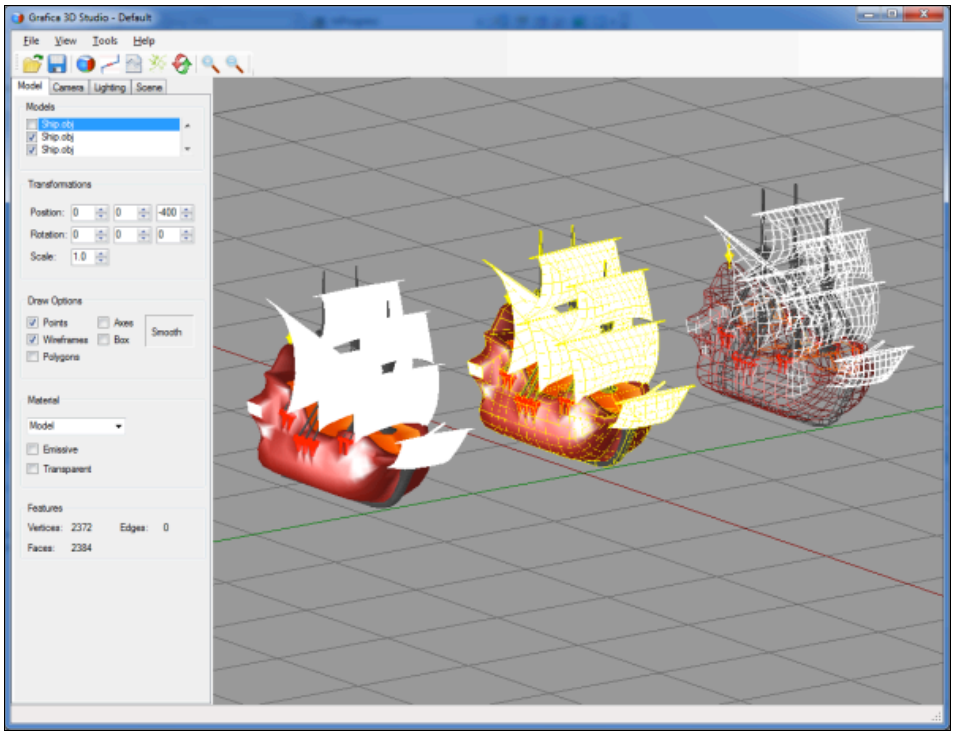


Figure 2. Scene with multiple instances of the *Ship* model.

Displaying Objects

There are three different ways of displaying an object: (a) displaying its vertices, (b) displaying its wireframe, and (c) displaying its rendered image, as shown in Figure 3. Additionally, Grafica can display objects as a combination of these.

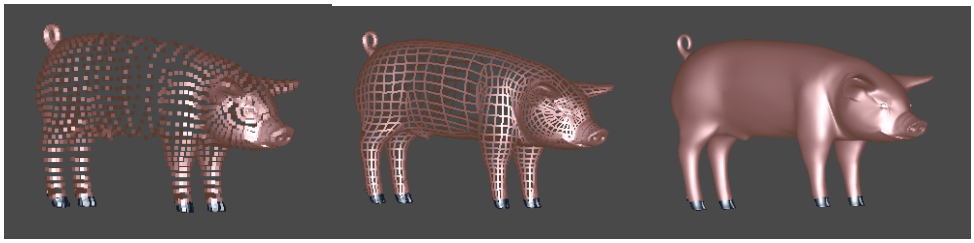


Figure 3. Different ways of displaying an object.

Material and Lighting

In the real world, objects have their own colour and are illuminated by white light, which contains an even mixture of all the colours together. If an object has a colour of red, it will mostly reflect the red photons of the white colour and absorb the rest. But if the light source is of a different colour, the object will appear darker; depending on how much light colour was absorbed. When we use lighting, we do not describe polygons as having a particular colour, but rather as being made up of materials, that have certain reflective properties. Thus, for each material, the reflective properties of ambient, diffuse and specular light sources must be defined. For example a custom material can reflect specular light very well, while absorbing most of the ambient or diffuse light or the opposite. Consequently, the material colour components actually determine the percentage of the incident light that is reflected from the surface.

Materials have two additional properties compared to light: the emission light property and the shininess. Emission property is the amount of light the model emits, while shininess property is used to define how much specular highlight must be produced on the material surface.

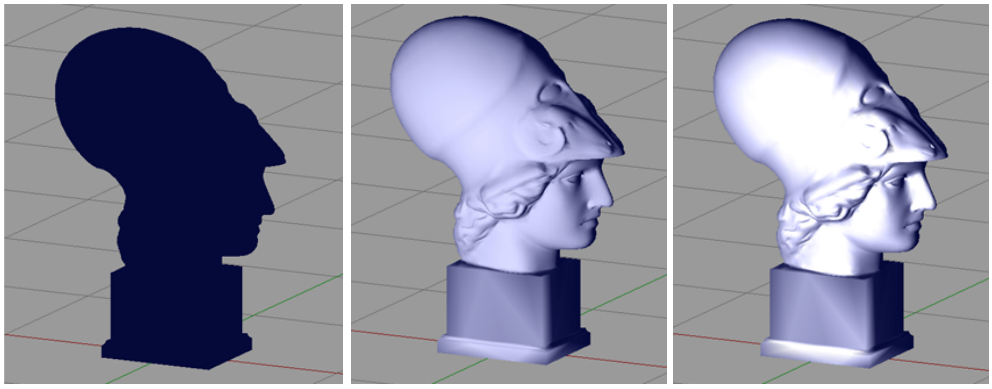
Ambient, diffuse and specular are the three types of light that OpenGL needs to approximate the real world light, in order to render and illuminate three-dimensional objects.

Ambient light doesn't come from any particular direction. Its rays of light are released from a source that is then bounced around a room or scene becoming directionless. Objects illuminated by ambient light are evenly lit on all surfaces and directions. The object shown in Figure 4(a) is illuminated only by ambient blue light. Because all vertices are evenly lit by the same colour, the object appears to look flat and two-dimensional.

Diffuse light comes from a particular direction but is reflected evenly by a surface. Even though the light is reflected evenly, the object surface is

brighter if the light is pointed directly at the surface than if the light grazes the surface from an angle. In Figure 4(b), the face is being lit by a combination of ambient and diffuse light, giving a more three-dimensional result.

Specular light is directional similar to diffuse light, but it reflects sharply giving a bright shinny spot on the surface in a particular direction. The intensity of the specular reflection depends on the model's material and the strength of the light source that contains the specular light component. In Figure 4(c), the face is being lit by a combination of ambient diffuse and specular light, making the model look realistic.



(a) (b) (c)

Figure 4. Illuminations by (a) ambient, (b) ambient and diffuse, (c) ambient, diffuse and specular light source.

Changing the position of the light source, or adding more light sources into the scene has illumination effects to models. The ambient, diffuse and specular components of the light are applied to material. An illustration is shown in Figure 5 where the light-source in the image on the left is in front of the object whereas on the image on the right is at the back of the image.

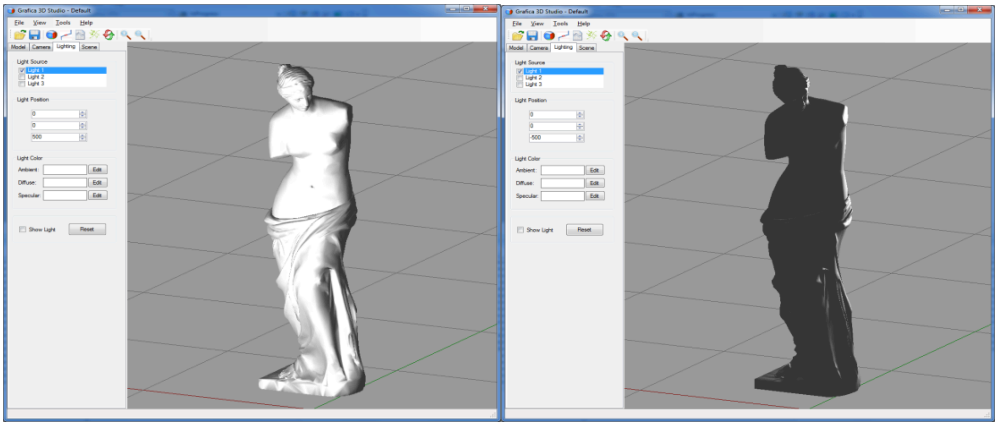
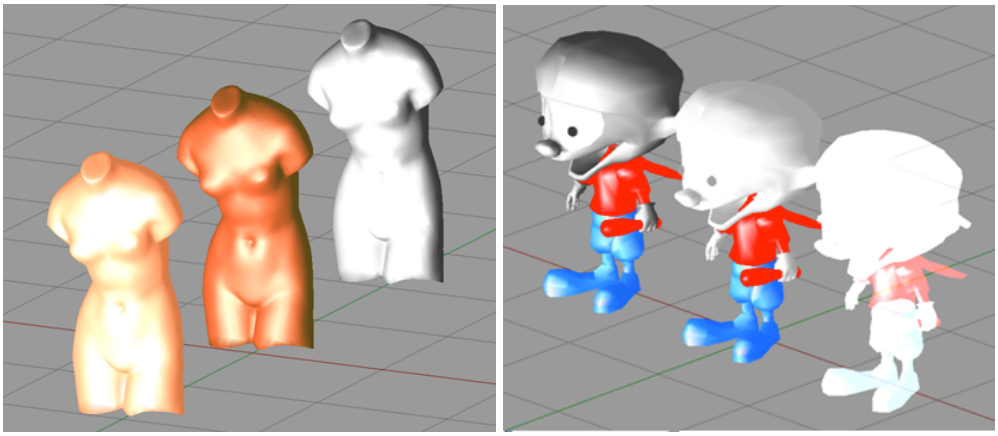


Figure 5. Moving the light-source.

Material Section options can be used for altering the material of the selected model. There is no tool for selecting a subset of faces; therefore the selected

material is applied to the entire model as illustrated in Figure 6(a). Further, emissive and transparent materials are illustrated in Figure 6(b).



(a) (b)

Figure 6. (a) Different model material.
(b) Natural, emissive and transparent material.

Transformations

Transformations can be obtained with the mouse and the keyboard. The user can *translate*, *rotate*, or *scale* individual objects, or the scene as a whole. If we recall that all the objects are loaded at the centre of the scene display area, then we can observe the transformations illustrated in Figure 7.

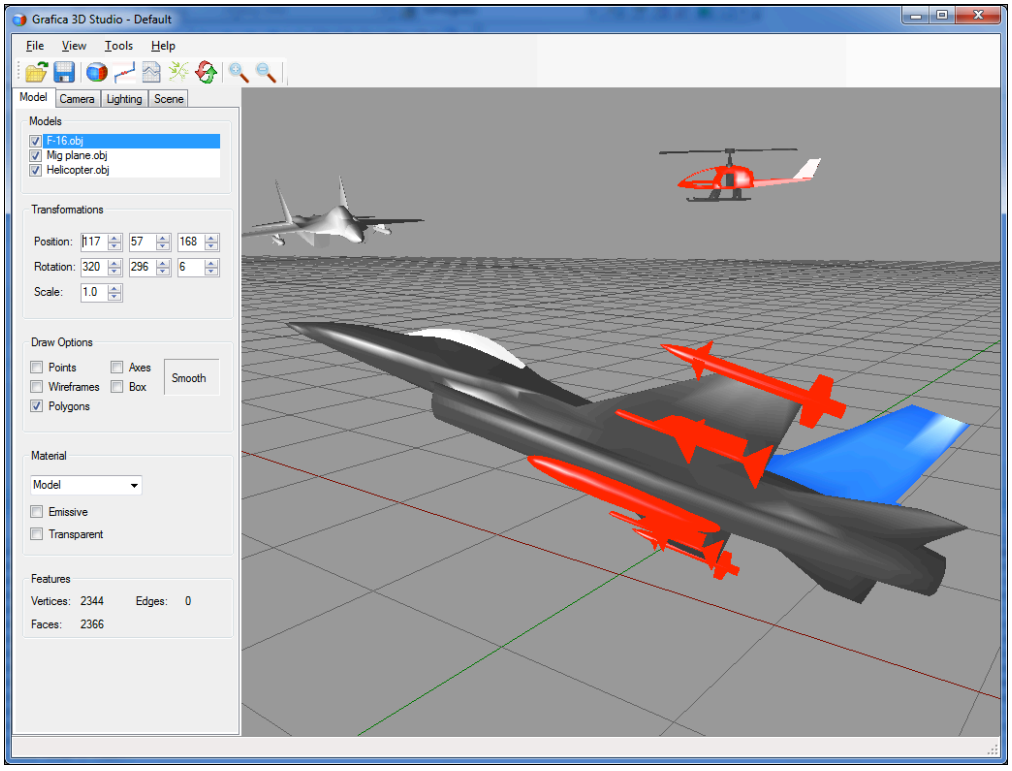


Figure 7. Object transformations.

Subdivision Surfaces – The Catmull-Clark Method

The Catmull-Clark method [Catmull & Clark 1978] is a subdivision method for generating smooth surfaces from a number of control points defining the shape of the surface. It is one of the most popular methods in surface modelling in the area of geometric design because it can generate surfaces based on an arbitrary topology of control points. In other words, the control polygon does not need to have a regular rectangular grid that is equivalent to a 2D array.

Grafica can apply the Catmull and Clark method to demonstrate smoothing of objects. An illustration is shown in Figure 8 where two objects are shown after 0, 1, 2 and 3 subdivisions (from left to right).

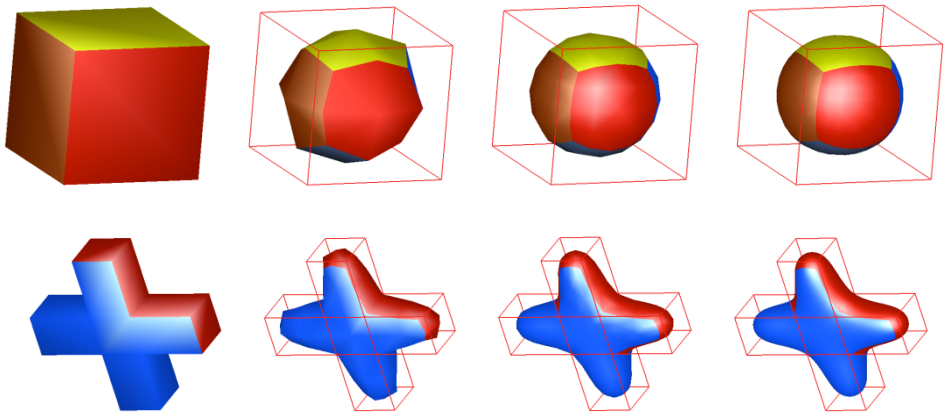


Figure 8. Two objects after 0, 1, 2 and 3 subdivisions: from left-to-right.

Employing Grafica as a Teaching Tool

The computer graphics course was introduced in the Computer Science BSc path at the University of Nicosia in the Fall 2003 semester. Since then the same lecturer has always taught it once a year. The interactive software program was introduced as a teaching tool in the last 3 years (2010 - 2013). The final examination included always 6 questions out of 8 on concepts for which Grafica was used as a teaching tool. These questions correspond to 75% of the examination's grade points. The impact of Grafica's introduction may be depicted in the grades of the students on these questions before and after the use of the software program as a teaching tool. More specifically, the average grade for these questions for all classes taught prior to the introduction of the software program was estimated to be 46.7/75. The corresponding estimate for the period investigated where Grafica was employed was 61.2. In other words, the average grade for the two questions rose by 14.5/75 grade points. These estimates may provide evidence that the use of Grafica in class assists both the lecturer in presenting material in a more efficient manner, as well as, the student in understanding abstract concepts better. The latter is also supported by positive comments given in periodic student evaluations of the course. The authors acknowledge that the efficacy of this teaching tool

needs to be validated through a survey study that may involve students and educators from other universities as well. This efficacy would quantify the level of success of the course learning outcomes.

Conclusions

In this paper some of the most important features of *Grafica* were introduced. It was impossible and impractical to cover all the functionality of the software; however, a significant sample has been presented. Particularly illustrated have been how objects are loaded into the scene, and how the user can modify their properties. The material and lighting schemes of *Grafica* were also presented.

In addition, the user can translate, rotate and scale the entire scene using the application tools. A variety of object drawing options are available, such as drawing polygons, wireframes, points and object control-polygons. The user is enabled to apply transformations on each model individually, while tools have been implemented for camera control. Finally, the user can smooth the objects using Catmull and Clark's subdivision method and save them in data files.

Grafica, as a user-friendly software, has been employed as a teaching tool for the undergraduate course of Computer Graphics. It has been observed from student examination results that the tool does have a positive effect on the grade of students on related examination questions. This in effect, may imply that with Grafica, students have a better understanding of the application of abstract mathematical concepts used in the course.

References

Aliasgari M., Riahinia, N., & Mojdehavar, F. (2010). Computer-assisted instruction and student attitudes towards learning mathematics. *Education, Business and Society: Contemporary Middle Eastern Issues, 3*(1), 6-14.

Buchau A., Rucker, W. M., Wossner, U., & Becker, M. (2009). Augmented reality in teaching of electrodynamics. *The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, *28*(4), 948-963.

Catmull, E., & Clark, J. (1978). Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, *10*, 350-355.

Folorunso, O., & Akinwale, A. T. (2010). Developing visualization support system for teaching/learning database normalization. *Campus-Wide Information Systems*, *27*(1), 25-39.

Jackson, M. (2004). Making visible: Using simulation and game environments across disciplines. *On the Horizon, 12*(1), 22-25.

Microsoft Corporation. (2013). Microsoft Visual Studio. Retrieved from http://www.visualstudio.com/en-us

OpenGL. (2014). The Industry Foundation for High Performance Graphics – From games to virtual reality, mobile phones to supercomputers. Retrieved from http://www.opengl.org

Taylor, M., Duffy, S., & Hughes, G. (2007). The use of animation in higher education teaching to support students with dyslexia. *Education + Training, 49*(1), 25-35.

Author Details

Andreas Savva savva.a@unic.ac.cy
George Ioannou gmioannou@gmail.com
Vasso Stylianou stylianou.v@unic.ac.cy
George Portides portides.g@unic.ac.cy