

# Preface

This book is the result of teaching computer graphics for one and two semester, year two/three undergraduate and postgraduate lecture courses in Computer Graphics. Throughout the book, theory is followed by implementation using C/C++ and complete programs are provided with suggestions for change to enhance student understanding. During 30 years of university teaching the author has become aware of the frustration that many students suffer, of code fragments that ‘never quite work’ and programs that on a different system require system dependent additions! With this in mind all the programs given have been tested using MS C++ v6 and most have been tested using Solaris 4.2 and Borland C++ v5.

There are a number of texts which give a more in depth approach to the OpenGL pipeline and repetition has therefore been avoided by referencing such texts for the interested reader. The objective is to get students immersed in graphics applications as rapidly as possible, to develop confidence, which in turn leads to experimentation, which is so vital to the enthusiastic programmer. Theory and practice have been developed in parallel so that in many cases the reader begins to understand the strengths and weaknesses of a particular algorithm.

After a rapid ‘getting started’ introduction we look at the structure of bit map (.bmp) files as a precursor to understanding audiovisual files (.avi). This work forms a foundation for later sections on image processing and texturing. These are very simple file structures that can be converted from other image file formats using commercially available software packages. The chapter introducing image processing covers edge detection, enhancement and data capture from CAT scans. Theory and practice can be quite different and some processing appears more of an art than a science due to the variability in the image quality and the nature of the image itself. The example of edge finding on CAT scans where different slices may have well defined edges and other more fuzzy edges due to gray hair is not always apparent to the eye. We address these problems with alternative solutions with varying degrees of success to enable readers to comprehend that algorithmic development is still an inexact science for such applications.

In Chapter 4 we move to the first chapter, which might be considered to be computer graphics with all the mathematics that is required. I do not encourage students to skip over these areas of understanding for the ‘black box’ approach will only get you so far and gaps in knowledge at an early stage will come to haunt one later on. The toil of getting to grips with material will be amply repaid as students grow in knowledge – although I am aware that many avoid the joys of mathematical rigor!

Early on in computer graphics students are often heard to say; “why is there no error message and nothing on the screen?”. Knowing where one is in 3D space has been another difficult area to communicate. Conceptually compressing a 3D model onto a 2D screen does not come easily to many students’ minds and here we introduce viewing. In OpenGL the viewing process is concerned with two operations; the mapping an object into a viewing volume for screen display and called the PROJECTION process, while the orientation and size operations on the object is referred to as the MODELVIEW process. Being in the wrong place, whether viewer or object can leave nothing on a screen and much confusion as how to proceed. The author maintains that there is little alternative to debugging than printing out a few well-chosen coordinates to see if they lie in the field of view! With practice one becomes aware of location, but in the early stages 3D space seems like learning to fly in cloud – few reference points and a degree of panic if you don’t trust your instruments (coordinate values in our case).

Lighting begins to bring realism to displays and in this section we cover the different forms of lighting available and see the effects as we move objects in relation to a light source. We also link the early work on the transformations as we see how illumination varies across a surface and adjust the surface properties to produce highlights.

In Chapter 7 we return to bit map images and use them to cover surfaces with pictures, initially on simple flat planes and then develop linear mapping methods for more complex surfaces. We introduce the concept of mapping onto terrain with a given height profile to simulate the idea of mountains and valleys. Multiple textures are used to simulate scenery as seen through windows and the changes apparent from movement in a room are implemented. An introduction to 3D texturing is included and finally reconstruction of 3D volumes from 2D CAT scan slices.

Object orientation has played a lesser part in the text up to this section and we now use the concept to build artifacts based on work in the previous chapters. Objects such as cylinders and ‘stretched cubes’ are textured with an appropriate image and result in building blocks for furniture. The tables built are then placed within the context of say a room to introduce students to applications of furniture and room decoration. The writing rather than only using objects has proved fruitful in student projects, where an understanding of mass manufacture has led students to consider in some depth the flexibility required in design for object use to be economic.

The final chapter considers the development of curves and surfaces using Splines and Bezier interpolation and fractals. A section on fractals explores the role they can play in adding objects to scenery and outlining complex shapes.

## Approach

This book sets out to solve problems by complete example rather than discursively hint of how it may be done. We live in an era of mass education where as ever, the very able thrive without much need of teaching but where the large mass of students

have less access to staff due to funding constraints. Some believe technology will overcome these obstacles and the author believes that one contribution to the solution of the problem is doing by example. However, a black box recipe approach leaves students vulnerable when circumstances change and thus I have included theory followed by worked examples. The initial implementation sometimes produces inefficient code and I have left this deliberately in order that students can follow where algorithms originate from theory. Good program design and efficient code comes from an in depth understanding that is the result of a long process of assimilation.

## Supporting Material

Support materials may be found may be found on the accompanying Springer web site <http://www.springer.com/978-1-84800-022-3>

They include:

- Program code
- Data files
- OpenGL include files
- OpenGL link libraries from the Mesa site.

Robert Whitrow  
Department of Computing  
Communications Technology and Mathematics  
London Metropolitan University, UK



<http://www.springer.com/978-1-84800-022-3>

OpenGL Graphics Through Applications

Whitrow, R.

2008, XV, 330 p., Softcover

ISBN: 978-1-84800-022-3