

Developing a 3D Graphics Application

**2009**

Alex J Davison

DV003874

Module Name: Introduction to Programming 3D Applications

Module Number: CE00056-1

Tutor: Dr Len Noriega

Title Image [2009]

Contents

Page Number

[Introduction 3](#_Toc228361600)

[Graphics programming 3](#_Toc228361601)

[Films 3](#_Toc228361602)

[Computer games 4](#_Toc228361603)

[Engineering uses 5](#_Toc228361604)

[OpenGL API implementation with in graphical applications 5](#_Toc228361605)

[Objectives of the application 6](#_Toc228361606)

[File format 6](#_Toc228361607)

[Graphical application 7](#_Toc228361608)

[Met the objective? 8](#_Toc228361609)

[Reading object(s) 8](#_Toc228361610)

[Texture/Lighting 8](#_Toc228361611)

[Hidden face elimination 9](#_Toc228361612)

[Movement of camera 9](#_Toc228361613)

[Change texture/lighting on input 9](#_Toc228361614)

[Control object using input 9](#_Toc228361615)

[Five object scene based on a file 10](#_Toc228361616)

[Scene graph 10](#_Toc228361617)

[Reflection and evaluation 10](#_Toc228361618)

[Bibliography 12](#_Toc228361619)

# Introduction

This report will cover an introduction to the world of graphics programming with the use of references to current graphics applications that are uses in films or computer games which in there own right a graphic application and other uses of graphics programming.

A graphics application has been created to show how graphics programming works. The graphics application will be used to show how to create an object and/or scene. The object scene data will be stored in a file and then read in by the graphic application, the graphics application will the display the object/scene. A justification of design decisions taken will be given in this report in regard to file layout.

This report will give an outline of the objective of the graphics application with an explanation of the functions of the application, this will also include any possible objectives set but not met in the in the graphics application.

The report will cover in brief the relation of the OpenGL API implementation to 3D computer graphics concepts which were discussed in the lectures and recommended reading given by Mr Bob Hobbs and Dr Len Noriega between 12th January 2009 and 31st March 2009.

Finally this report will critical evaluate and reflect upon the graphics application made.

# Graphics programming

Graphics programming is used when physics, maths and the use of computer are used a 2D image based upon 3D virtual world. To quote Edward Angel “Computer graphics is concerned will all aspects of producing picture pr images using a computer.” Interactive Computer Graphics [2009]

Graphics programming can be used for many applications such as films, games, engineering uses etc below are three brief description of examples of graphics programming being used in real life.

## Films



Figure 1 (Left) & 2 (Right)

Animation(2009) Spider-Man (2009)

Many films in today’s cinemas have some form of graphical programming in them, some films are entirely created using graphical programming, with actors voices used to bring the images alive. As shown in figure one and two graphical programming was used to achieve figure one and figure two.

Figure one is completely made from a graphical application, to make the animation of the whole scene. This animation is like that of the Coca-Cola polar bear commercial created by Rhythm and Hues in 1993, where graphical programming was used to make unreal events seem real. Like that of a polar bear drinking a bottle of Coca-Cola. These animations where made as stated by Isaac V. Kerlow [2009] “...produced entirely on 32-bit Amiga and Macintosh microcomputers.” showing the potential of graphical programming at an early stage if computer technology.

Figure two on the other hand is composed of two images, one which made using graphical programming and another using the two actors whom are real. The effect of spider-man hitting a hole into the sand man, with all the sand disbursing out of the back of the sand man’s body is part of the graphical programming. The graphical programming was used to simulate realism of the sand. Computer would be used to calculate the path of grains of sands. Not only the path but also other things such as the light source(s) and whether the grain of sand should be dark or light and other aspects such as these.

## Computer games



Figure 3 (Left) & 4 (Right)

Tom Clancy H.A.W.X (2009) Grid (2009)

Currently computer games are one of the best examples of graphical programming as they try to achieve realism in all aspects of the game, using physics and maths to achieve the realism in movement and images. Textures are given to objects/scene to make the game appear as real as possible. Computer games use Newton’s laws of motion[[1]](#footnote-2):

“Law I

A boy tends to remain at rest or continue to move in a straight line constant velocity unless it is acted upon by an external force. This is the concept of inertia.

Law II

The acceleration of a body is proportional to the resultant force acting on the body, and this acceleration is in the same direction as the resultant force.

Las III

For every force acting on a body (action) there is an equal and opposite reacting force (reaction) in which the reaction is collinear to the acting force.”

Figure three and four shows two computer games Tom Clancy's H.A.W.X (figure three) and Grid (figure four). Both computer games use Newton’s laws to portray realism. Such things as collision detection are implemented so when two cars crashing Newton’s laws are used to calculate the resulting actions of the cars. Even an aircraft’s missile hitting another object results in a change within the computer game using Newton’s laws. These are calculated using physics, maths and computations made by the computer.

## Engineering uses

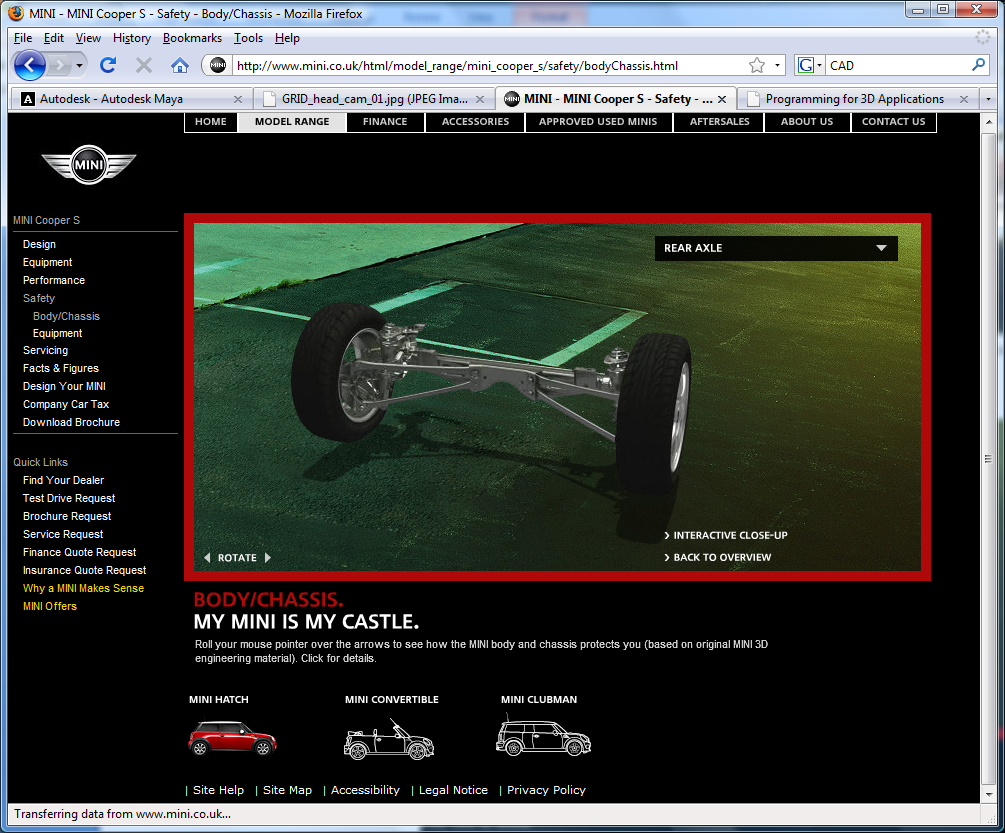


Figure 5 (Left) & 6 (Right)

Mini Frame (2009) Valve (2009)

Engineers use CAD (Computer Aided Design) applications to visualize a product or in some cases to test the product before prototyping this is also indicated by the first chapter of Donald Hearn []. CAD applications use graphics to a product but also show how a product such as suspension unit as show in figure five might break under extreme conditions. Not only can it show how things might break by simulating the materials the product is made of but also how the product might function and look at proven by the valve in the figure six.

# OpenGL API implementation with in graphical applications

OpenGL provides methods that are prewritten and that use mathematical calculation to work out lighting effects, the viewing frustum (the area in which we can see), shading, object placing etc many of the things that we take for granted in the really world. These methods are needed as computer does not know of such concepts that are in the real world but also predefined methods help to speed up the process of writing 3D applications.

The predefined methods in OpenGL are useful as programmers do not have to rewrite large amounts of code over and over again to achieve the same effect. The methods don’t only use math but also physics to achieve the really world visualisation for the 2D image that is produced.

An API (Application Programming Interface) allows for the code to be written by the programmer and then checked by the computer itself to ensure that the code will compile, however, the computer cannot check for logical errors made by the programmer.

As the mathematical elements and the complex physics of the elements have be compiled in the OpenGL methods that allows values to be passed into the OpenGL methods. This will is all done in an API by the programmer, it is down to the programmer to put all the elements together by the use of loops, if statements switch statements etc and the logical thinks of the programmer to make a graphical application.

The combination of OperGL methods and an API such as Microsoft Visual Studio 2009 allows for graphical applications to be made. Not only are the use of the OpenGL methods needed and the use of the API needed but also the knowledge of such concepts of the viewing pipeline needed by the programmer to achieve such graphical applications in figure one to six.

# Objectives of the application

The objectives of the graphical application are to produce a graphical application which does the following:

The graphics application should read in at least one object from a file, with another file for for the texture to be applied to the object. The object should use back face culling to remove any unneeded facets, so that from the viewing point that they unneeded facets can’t be seen. However the viewing point should be moveable back face culling should be recalculated after the camera is moved. The object should have lighting and texture applied to it. The object also should also be movable use keyboard input.

Once the single object has been setup it should be easy to setup four additional objects to create a scene, using a scene graph to manage the objects and read them in. All these criteria can be found in the Assignment [2009]. All these should be implemented within the graphical application. The main objective is to be able to write a application similar to a screensaver, however, with interactive elements to the application.

# File format

For the 3D application the file format was kept simple to keep it easy to error handle as the object was defined by firstly drawing the object on a piece of paper and working out the all the points, winding order and normalisation which was hard to understand. Understanding the new concepts meant that the file format was keep simple to for it to understood, this would also make it easy to change if needed.

The format that was used for the scene was as shown below in figure seven.

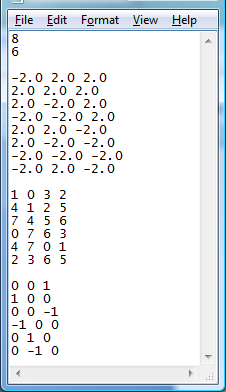


Figure 7

The first number in figure seven defines the number of points of object being read in, this number is used to initialize the variable for the points of the object. The second number defines the number of facets that the object has. The next set of figures defines the positions of the eight points of the objects in relation to the world origin.

The next lot of numbers made up of zero to seven which defines the winding order of the, the facets below that is the normalisation for the facets so that the facets has a position of up for lighting purposes. The normalisation is used to calculate whether the facet should be darker or lighter depending on the lighting within the scene.

# Graphical application

Below is a breakdown of the graphical application that was made, with the application objectives and whether the objectives have been met. An example of the graphical application can be found in figure eight.

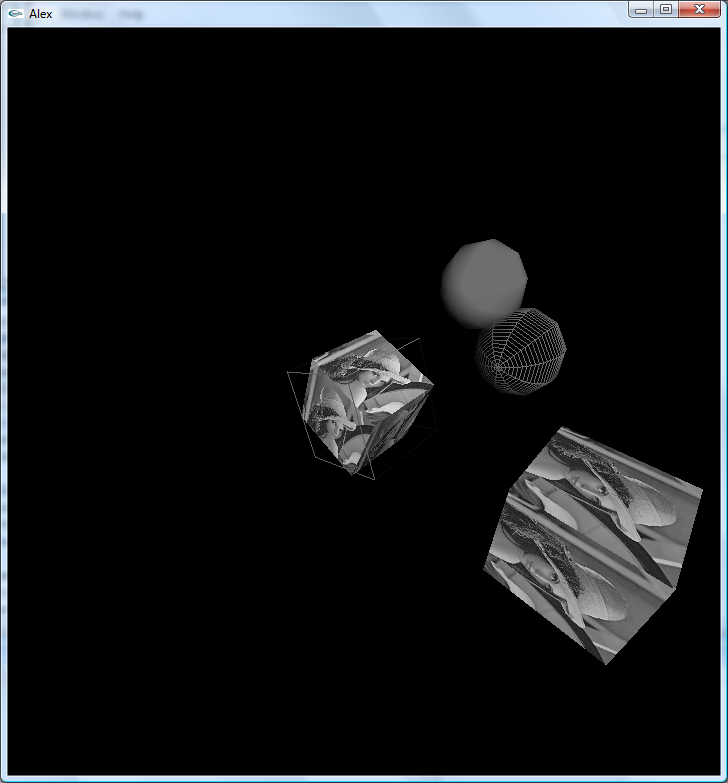


Figure 8

## Met the objective?

Below is a summary of whether the objectives have been met.

### Reading object(s)

It was possible to read in a cube in to the graphical application. The cube was read in from a text file that is shown in figure seven. This was a great feat as reading in from a file in the C language had been accomplished before. This achievement was well earned as this part of the project had been very difficult. However it was not possible for a larger object to have been read in.

A larger object would have been nice but this was not possible as inexperience and lack of time hampered the efforts to achieve such a feat. Having managed to read in the simple cube has given a bigger insight in to how a bigger object would be read in. All that would be needed would have been a large number of vertex coordinates and facets winding numbers.

The simple cube, however, was a great achievement after all the time spent on this area of the graphical application. It took many hours and days to implement the seemly easy task which turned into a nightmare.

Hobbs B (2009) exercises helped with the implementation of reading in an object, along with the lecture give by Nick Dyson [2009] and where the main bases for the method used to input the file data into the graphical application.

The cube can be seen in figure eight in the middle and also a duplicate on the right hand side of the picture near the bottom.

### Texture/Lighting

A texture was added to the cube, the texture is read in from a file, the texture was from a raw file format. The image was in greyscale which was 512 by 512pixels big, which was perfect for the cube. A coloured image would have been preferred and could have implemented with the use of Dr Noriega L (2009) exercise, however due to lack of time the coloured image was not achieved.

Three lighting effects were applied to the scene with great success, as problems plagued efforts to apply the effect. The errors that did occur where fixed as soon as hidden faces where removed. It is easy to see that the lights are working because as the cube in the middle rotates around the facets get lighter and darker meaning that lighting is implemented. The location of the light is not known and colours aren’t idea however this was research but not implemented as time was short any changes made seemed to create more errors.

Dr Noriega L (2009) helped along with Angel E (2009), Wright Jr R. S. & Lipchak B (2004) and Hill Jr F. S. (2001) with the implementation of texturing. Lighting was implemented with the use of Dr Noriega L (2009) exercise being implemented on to the graphical application.

### Hidden face elimination

The hidden facets of the cubes and other object in the scene where removed from being drawn. This was achieved with great success and solved some issues that cropped up during the production of the graphical application. This was one of the simpler aspects of the graphical application to implement.

Hidden face removal was aided by the details about the subject give in Angel E (2009) and Wright Jr R. S. & Lipchak B (2004).

### Movement of camera

The movement of the camera was achieved with great success. The camera movement was achieved beyond expectations as the camera rotates around the point of origin of the world coordinates using the “q” and “e” key on the keyboard. This is was achieved using maths never used by the programmer. More camera movement would have been added had there been more time.

### Change texture/lighting on input

The ability to enable the lighting was easy and the texture was a much harder, however the ability to change the texture and/or the lighting of an object in the scene using input was not achieved. The ability to change the light/texture wasn’t achieved due to the lack of experience and time, however, it would have been easy by changing the colour array used to define the colour of light, this would have meant that the colour of the light would have change from grey to another colour at the push of a button.

Even after the help by Dr Noriega L (2009), Angel E (2009), Wright Jr R. S. & Lipchak B (2004) and Hill Jr F. S. (2001) with the implementation of texturing and. Dr Noriega L (2009) exercise it was still not possible to implement this objective .

### Control object using input

It is possible to move the cube from the centre of world, it is possible to move the cube in four different directions in the x and z axis. This was made with great success, not only does that cube move but it also rotas about its own axis. There are also two spheres that cross one another’s paths continuously. The spheres were meant to show collision detection, however, this was not possible to achieve this given the complex nature. What was achieved, was done so independently making it one of the great successes.

The controls are as follows:

* “1” – Slow cube rotation of the cube
* “2” – Slow cube rotation of the cube
* “r” – Rest spheres positions
* “l” – Change direction of cube in the middle
* “+” – Enlarge spheres
* “-“ – Decease size of spheres
* “w” – move cube negative in the z axis
* “a” – move cube positive in the z axis
* “s” - move cube negative in the x axis
* “d” - move cube positive in the x axis

### Five object scene based on a file

Having read in one cube successfully it was not possible for more data to be read from the file and then be displayed but it was possible to hard code five objects. Three cubes and two spheres are defined within the world. Two of the objects are technically the same cube as the cube is read in twice. The cubes that are read in are the same cube that is why they have the same texture. Having spent a lot of time on the one object and hoping to get collision detection with the two spheres both collision and reading in five objects was unsuccessful. The reading in of the one object was a great success and will help with future 3D applications.

### Scene graph

Understanding why scene graphs are used is easy and the theory behind the scene is understandable but how to code a scene graph was too complex for the scene graph to be implemented. The scene graph was one of the most difficult aspects to grasps out of all the assignment therefore was not achieved. However a lot of time was spent researching the scene graph and aspects surrounding the scene graph but still with no success.

# Reflection and evaluation

After the 3D application had been declared finished it was apparent that from lack of experience of not having done much of this type of work before that the application was lacking in many areas. Compared to other members of the course the graphical application seemed simple and very minimalistic. However it is believed that most apart from two have of the items needed have been covered.

If there had been more time then much more would have been cover and achieved. A lot of

time was spent on understand the basic concepts of the 3D graphical application, however achieving this would have been easier having had more time to try working examples and piecing bits of code together.

The lectures and staff where very useful, even after attend extra lectures in addition to those mandatory to the course some of the concepts still seem obscure. Such things as linking list or better known as scene graphs also pointers and others where cover and over again but still not fully understood this highlighted areas of weakness. Areas of strength were also highlighted so over all the 3D application was a success.

Overall there was a great success on starting to learn new concepts, however, hopefully over time the subject will become easier to understand and implement as it will become second nature.

# Bibliography

|  |
| --- |
| Angel E (2009) *Interactive Computer Graphics a Top-Down Approach Using OpenGL* Fifth Edition Pearson Education [*Page 485 - 492*] |
| Angel E (2009) *Interactive Computer Graphics a Top-Down Approach Using OpenGL* Fifth Edition Pearson Education [*Page 262-264*] |
| Angel E (2009) *Interactive Computer Graphics a Top-Down Approach Using OpenGL* Fifth Edition Pearson Education [*Page* ]  Edward Angel “Computer graphics is concerned will all aspects of producing picture pr images using a computer.” Interactive Computer Graphics [2009] |
| Animation (2009) [Online] Available from: <http://images.autodesk.com/adsk/files/top_10_reasons_to_buy_maya.pdf> [20th April 2009] |
| David M. Bourg [2002]  “Law I  A boy tends to remain at rest or continue to move in a straight line constant velocity unless it is acted upon by an external force. This is the concept of inertia.  Law II  The acceleration of a body is proportional to the resultant force acting on the body, and this acceleration is in the same direction as the resultant force.  Las III  For every force acting on a body (action) there is an equal and opposite reacting force (reaction) in which the reaction is collinear to the acting force.” |
| Dr Noriega L (2009) An Introduction to LightingCE00056-1 [Lecture Notes] *An Introduction to Lighting* Introduction to 3D applications Stafford University ST18 0AB 9th March 2009. |
| Dr Noriega L (2009) Texture MappingCE00056-1 [Lecture Notes] *Texture Mapping* Introduction to 3D applications Stafford University ST18 0AB 16th March 2009. |
| Grid (2009) [Online] Available from:  <http://www.codemasters.com/images/microsites/grid/GRID_head_cam_01.jpg> [20th April 2009] |
| Hill Jr F. S. (2001) *Computer Graphics Using Open GL* Second Edition Alan Apt [*Pages 439 - 442*] |
| Hobbs B (2009) Surface RenderingCE00056-1 [Lecture Notes] *Surface Rendering (Wire Frame Mesh)* Introduction to 3D applications Stafford University ST18 0AB 19th January 2009. |
| Isaac V. Kerlow [2009] “...produced entirely on 32-bit Amiga and Macintosh microcomputers.” |
| Mini Frame (2009) [Online] Available from: <http://www.mini.co.uk/html/model_range/mini_cooper_s/safety/bodyChassis.html> [20th April 2009] |
| Nick Dyson [2009] 20/03/2009 9am |
| Spider-Man (2009) Available from:  <http://www.reelingreviews.com/spider-man3pic.jpg> [20th April 2009] |
| Title Image (2009) Available from:  <http://larryfire.files.wordpress.com/2008/06/26_pixar_lg.jpg> [20th April 2009] |
| Tom Clancy H.A.W.X (2009) [Online] Available from: <http://blog.al.com/techcetera/2008/04/TomClancyHAWX2.jpg> [20th April 2009] |
| Valve (2009) Available from: <http://www.swagelok.com/images/cmi/imageLibrary/CAD%20image.jpg> [20th April 2009] |
| Wright Jr R. S. & Lipchak B (2004) *OpenGL Superbible* Third Edition Sams [*Page116 - 118*] |
| Wright Jr R. S. & Lipchak B (2004) *OpenGL Superbible* Third Edition Sams [*Page453 - 457*] |

1. Newton’s Laws are taken from David M. Bourg [2002] [↑](#footnote-ref-2)