Monash University
CSE3313 Computer Graphics

Tutorial 1 2007

30th July 2007

Objectives

- Download and compile an OpenGL program;
- Understand the basics of GLUT: windows, frame buffer modes, event loops
- Understand how GLUT uses *callbacks* to respond to events: *display*, *resize*, *keyboard* and *mouse* events;
- Develop an overall strategy for successfully completing assignment 1: data structures, control flow, OpenGL calls required;

OpenGL Basics

On the CSE3313 web site you will find some sample OpenGL code to download (Home Page→Resources→Sample OpenGL source code: http://www.csse.monash.edu.au/~jonmc/CSE3313/Resources/SampleCode/index.html

Exercise 1.1 (download) *Download the first file,* `simple.c`.

Open the file in a text editor and take a look at the program. This is a very simple OpenGL program – it displays a white square on a black background. Notice the initialisation and setting of callback function in `main`. The final call is:

```
glutMainLoop()
```

which passes control to the GLUT event loop. This function sits in a loop and processes events until the program quits. In order for the event loop to pass control back to your own code, you need to set some *event callbacks*. In this example the only callback is the display function, `display`, set using `glutDisplayFunc`. These event callback setting functions take *pointers to functions* as parameters. Take a look at the `display` function – it is very minimal, clearing the frame buffer and then drawing a square polygon.

Compile the program and run it.

Now try to resize the screen. What happens to the picture? The problem is that we have not provided a `resize` callback to cope with the dimensions of the display window changing size.

Exercise 1.2 (resize) *Modify the program so that it correctly handles resizing of the screen window. Make sure the program maintains correct aspect ratio for the display.*

Your program should now respond correctly to any resizing of the display window. Currently there is no way to quit or interact with the program, so the next step is to add some simple keyboard interaction.

Exercise 1.3 (keyboard) *Modify the program so that it responds to the user pressing the 'q' or 'Q' keys. If either of these keys are pressed the program should quit (exit gracefully). Any other key press should be ignored.*

Exercise 1.4 (zoom) *Modify the program so that pressing the 'z' key causes the image to zoom out and the 'Z' key causes a zoom in.*

Hint: you should implement a zoom by modifying the parameters to the `glOrtho2D` call. Do not use `glScale` to perform the zoom.

Next, we will add mouse interaction, so that the user can rotate the square interactively.

Exercise 1.5 (rotate) *Modify the program so that the square rotates according to mouse movement. The rotation should be proportional to the horizontal movement of the mouse across the screen and should only take place while the left mouse button is being pressed.*

Exercise 1.6 (text) *Modify the program so that the angle of rotation of the square is displayed in the bottom left hand corner of the screen. Use raster text for this task.*

Exercise 1.7 (gasket) *Download the sample file* `gasket.c`

This program draws the Sierpinski gasket using the random point method. Have a look at the code and see if you understand how it works (run it first to see what it does).

Modify your program code so that your program now draws the gasket rather than a simple square.

Assignment 1

A data structure for the superformula needs to be developed. Recall from the assignment sheet that there are six main parameters that control the shape defined by the superformula: a, b, m, n_1, n_2, n_3. You also need to know the range for θ. The data structure for the superformula should also capture the *state* specific to each individual superformula (e.g. things like `FillState`).

Exercise 1.8 *Devise a suitable data structure for the superformula. Create a constructor(C++) or initialiser function to initialise the structure's values to sensible defaults.*

You might also want to create a setter functions/methods that change the parameters.

Exercise 1.9 *Write a function that draws the superformula based on the data structure you have developed.*

You may need to add extra parameters and a function to compute the geometric shape of the superformula.