**Programming for Graphics: Assignment 1**

**Introduction**

This report will introduce and evaluate the graphical techniques used in the development of two applications, a clone of Atari’s famous arcade game Breakout and a three-dimensional Snake game that is viewed from an isometric perspective.

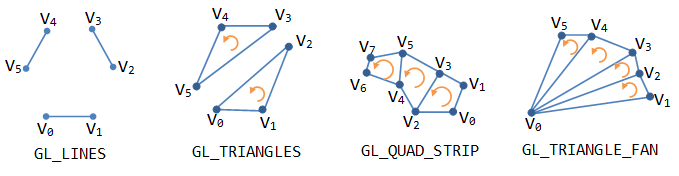
**Windowing**

The OpenGL graphics API that was used to render the graphics within the applications required a window be provided to render to. There are many libraries available that provide windowing functionality and each provide their own strengths and weaknesses.  
  
The OpenGL Utility Toolkit (GLUT) is a library that was created to provide a platform independent windowing solution. GLUT however became unmaintained and fell behind newer OpenGL releases. A separate and more active project named FreeGLUT was created as an open source alternative to GLUT and provides a similar set of functionality (Movania, 2013, p.30).

FreeGLUT however provides little to no support for iOS or Android and itself is no longer regularly updated and therefore is not suited for applications that intend to support all major desktop and mobile environments (Holy Ghost Stories, 2012).  
  
Despite its shortcomings, FreeGLUT was used to create the windows for both the Breakout and Snake applications as it is simple to setup, easy to use and thus a good choice for experimenting with and learning OpenGL.

**Graphics Primitives**

Graphics primitives are made up of vertices and can be described simply as basic geometric entities, these can range from lines and triangles to more intricate primitives such as quad strips and triangle fans (Astle, 2002, p.99).



(Geometric Primitives, 2012)

There are various methods that can be used to visualise vertex data and render primitives in OpenGL. Immediate mode requires all render calls to be enclosed between a pair of glBegin and glEnd function calls, the data is then immediately sent to the graphics processing unit for rasterization (3D Game Engine Programming, 2012). Below is a code example that draws a line primitive in immediate mode.  
  


Although straight forward, immediate mode is known to be inefficient when describing complex models and was removed from OpenGL with the release of version 3.0 (Benstead, 2014, p.51).

Vertex Buffer Objects (VBO) on the other hand store data in buffers on the graphics card as opposed to the immediate mode and vertex array techniques which store data in system memory. Using VBOs is often preferred over immediate mode as it increases performance by utilizing the GPU and reducing the amount of OpenGL API calls required by processing in batches (Benstead, 2014, pp.58-59).

Both the Breakout and Snake applications render using immediate mode as they contain only a small number of simple primitives thus the performance increase from using more advanced techniques such as VBOs and vertex arrays would be negligible.

**Transformations**

Moving objects in OpenGL requires use of what are known as coordinate transformations. Transformations allow us to move, rotate and manipulate entities within three-dimensional space (Astle, 2002, p.114).

The snake application contains cubes that move around the game world representing a snake. A Cube class was created that contains functions the perform transformations such as movement and rotations to allow for scaling and prevent code reuse. An example of the Cube class’s rotate function can be seen below.



The glRotatef function performs rotations around vectors in a counter clockwise direction by an angle measured in degrees. Clockwise rotations can be performed by providing a negative angle (Astle, 2004, p.77).

**Textured Meshes**

There are various libraries and tools such as Qt, DevIL and SOIL that assist in loading textures into OpenGL. Developer’s Image Library (DevIL) is a cross platform library that loads textures from various image types such as bmp, gif and png (Schuller, 2011, p.29). The snake application’s textures are loaded and mapped using the DevIL library and OpenGL’s glTexCoord2f function.

Texture mapping is the process of applying an image to the surface of a primitive (Astle, 2004, pp.155-156). glTexCoord2f allows you to pick points on a given texture and apply them to vertices (Michigan State University, 2016). An example usage of glTexCoord2f from the Snake application can be seen below.



**Lighting**

In regards to lighting, the Snake application features a pulsing light that changes colours over time. The glLightfv function was used to specify the light source parameters such as position and colour.

 **Bibliography**

3D Game Engine Programming (2012), *Rendering Primitives with OpenGL*. Available at: www.3dgep.com/rendering-primitives-with-opengl (Accessed: 3rd January 2017).

Astle, D. (2002), OpenGL Game Programming, Course Technology.

Astle, D. (2004), *Beginning OpenGL Game Programming*, Course PTR.

Benstead, L. (2014), *Beginning OpenGL Game Programming*, 2nd edn, Course Technology.

*Geometric Primitives* (2012) [OpenGL]. Available at: https://www3.ntu.edu.sg/home/ehchua/programming/opengl/CG\_Introduction.html (Accessed: 3rd January 2017).

Ghost Stories (2012), *The Sorry State of Game Portability*. Available at: http://www.jeffwofford.com/?p=1097 (Accessed: 3rd January 2017).

Michigan State University (2016), *Tutorial 4: Texture Mapping.* Available at: https://www.cse.msu.edu/~cse872/tutorial4.html (Accessed: 10th January 2017).

Movania, M.M. (2013), *OpenGL Development Cookbook*, 1st edn, Great Britain: Packt Publishing.

Schuller, D. (2011), *C# Game Programming: For Serious Game Creation*, 1st edn, United States: Cengage Learning.