Spring 5-1-2011

Interactive OpenGL Animation

Lusha Zhang

Interactive OpenGL Animation

A Capstone Project Submitted in Partial Fulfillment of the
Requirements of the Renée Crown University Honors Program at
Syracuse University

Lusha Zhang
Candidate for B. S. Degree
and Renée Crown University Honors

May 2011

Honors Capstone Project in Computer Science

Capstone Project Advisor: _____
Ernest Sibert

Honors Reader: _____
Marjory Baruch

Honors Director: _____
James Spencer, Interim Director

Date: _____

Abstract

This is an interactive animation created using C language, with support from the OpenGL, GLUT and the SDL frameworks. The concept behind this animation is to utilize the functions provided by OpenGL to achieve visual effects involving lighting, texture, motion, modeling and interaction.

Based on experiment and research on various libraries, the Simple DirectMedia Layer (SDL) framework was chosen to implement an audio frame buffer to play background music. The animation involved significant mathematical calculation in terms of motion and modeling. The animation is an underwater scene that contains randomly generated bubbles, a dolphin and a whale. Textures were applied to surrounding surfaces that construct the underwater environment. Lighting was optimized to make all the objects in the scene visible and genuine. Timing was adjusted to obtain smooth rendering. The dolphin and the whale were constructed vertex-wise using display lists.

In terms of interaction, the animation allows users to traverse the scene using keyboard input. Users can also choose to play and stop background music, which was composed using GarageBand.

The animation has met the proposed requirements. The OpenGL application programming interface (API) provides a basic and easy-to-use library for 2D and 3D graphic development. It constitutes the foundation of higher-level platforms such as Blender and Autodesk Maya.

This report is presented as a “system requirements specification,” augmented with explanations of various technical issues and certain implementation consideration.

Contents

1. Introduction

1.1 Purpose

The purpose of this document is to describe the specific functionalities and features of an OpenGL Animation. The methods used to implement the interaction, animating, modeling and lighting involved in this animation will be explained in this document, so that it can be referred to future research. The document will also explore the mathematical approach of modeling and animating in OpenGL.

1.2 Scope

The OpenGL Animation will allow users to traverse the scene. Constraints will be applied to avoid rendering issues, such as users going out of the boundaries and infinite loops caused by the improper use of display functions.

1.3 Definitions

- Display List: a group of OpenGL commands stored for future implementation. When a display list is called, the commands in the list are executed in its predefined order.
- Viewport: a rectangular area used to project a 3D scene to the position of a virtual camera. A viewport is a region used to display a portion of a scene.

- Modelview Matrix: the matrix that contains both modeling and viewing transformations that place the viewer at the origin with the view direction aligned with the negative Z axis.
- Projection Matrix: a matrix that handles the transformation from the eye coordinates to the clip coordinates--from a 3D scene into a 2D image.
- Identity Matrix: a matrix that represents the identity transformation.
- Frustum: the field of view of a virtual camera that defines how far the viewer can see in a 3D space.
- Normal Vector: a vector that is perpendicular to a surface.
- Mipmaps: pre-calculated, optimized collections of images that accompany a main texture in order to increase rendering speed.

2. General Description

2.1 Perspective

The goal of this animation is to create a generally pleasing, tranquil environment, which contains three purposes. The first purpose of the designer is to investigate the random generator. Water bubbles need to be randomly generated in terms of location and rising speed. To investigate modeling, mathematical calculation is applied at the vertex level. The second purpose is to conduct interaction with the user. The third part is to smoothly animate the objects.

2.2 Functionality

- Have a window interface
- Allow users to traverse the scene smoothly through keyboard commands
- Have a right-click pop-up menu
- Load a default .wav file
- Allow users to choose to play or stop the background music
- Allow users to quit the animation
- Provide error checking
- Keep the viewpoint within the scene

2.3 User Characteristics

The viewer, who will watch the animation and traverse the scene, must have access to computers with Mac OS X 10.5 or later. The viewer must also be able to open the executable file and follow the instructions stated in the Readme.txt file.

2.4 General Constraints

Users cannot cross the boundaries of the environment. The bubbles are randomly generated within the boundaries of the environment.

2.5 Dependencies

The following frameworks will be used to support graphics and audio implementation:

- OpenGL.framework

- GLUT.framework
- SDL.framework
- AppKit.framework

3. Specific Requirements

3.1 Functional Requirements

3.1.1 Viewport Initialization

The Viewport size must be initialized with proper width and height. The frustum must have settings for the user to see the scene clearly without distortion.

3.1.2 Camera Initialization

The virtual camera must be initialized at a location at which the viewer can see the whole setting.

3.1.3 Water Surface

The water surface must be constructed using proper geometry and must have proper texture applied.

3.1.4 Floor Surface

The floor surface must be constructed using proper geometry and must have proper texture applied.

3.1.5 Surrounding Water

The surrounding water walls must be constructed using proper geometry and must have proper texture applied.

3.1.6 Play Music

The animation provides a right-click pop-up menu, which contains the option “Play Music.” It allows the viewer to turn on background music.

3.1.7 Stop Music

The right-click pop-up menu provides the option to the viewer to stop playing background music.

3.1.8 Bubbles

Creating bubbles must include two parts--modeling and animating.

- Solid sphere must be chosen with proper radius and division to create a bubble. Each bubble must have material applied. The ambient light, diffusion, specular reflection, shininess and emission settings must reflect the genuine appearance of water bubbles in an aesthetic way.
- The location of bubbles must be randomly generated, but must not be outside of the scene. The bubbles must rise from the bottom to the water surface smoothly.

3.1.9 Load Texture

This function must load a .raw file into the buffer and bind the corresponding texture mipmap before applying texture to objects.

3.1.10 Traversing

Keyboard interaction must allow the viewer to traverse the scene using keyboard commands.

- **Rotation**

  Left and right arrow keys must allow the viewer to turn left and right, respectively, at a conveniently chosen angle and to rotate smoothly every time.

- **Parallel Movement**

  Up and down arrow keys must allow the viewer to move forward and backward, respectively, by a certain distance each time.

3.1.11 Display Function

The display function must coordinate all the models and settings, and is called in every GLUT loop to render each frame.

3.1.12 Set Lighting

Lighting settings must make all the objects in the scene visible and optimize the visual presentation of colors and texture. It must be included in the display function and initialization.

3.1.13 Timer

A timer function must maintain the redisplaying of the scene to be smooth.

3.1.14 Error Checking

The animation must be able to check for the following conditions that would cause errors or rendering issues:

- Before playing background music, the animation must check if the .wav file is loaded correctly. If the file is not found, the error checking function generates an error message and terminates the program.
- Functions that handle the rotation and parallel movement of the virtual camera must verify that the new location of the viewer is within the boundaries of the scene before making a move. If the viewer is about to be out of bounds, the traverse functions must set the viewer at the nearest boundary.

3.1.15 Audio Mixer

- Audio mixer initialization must specify the settings for OpenGL animation with 22Khz, 16-bit and 2 channels.
- The mixer function must load a .wav file, convert it to fit the initial settings and build an audio converter.
- When reaching the end of the music, the mixer must free the loaded data.

3.1.16 Right-Click Pop-up Menu

Right clicking must retrieve a menu offering the following options:

- Quit the animation
- Play background music
- Stop background music

3.1.17 Animating the Dolphin and the Whale

- **The Dolphin and the Whale Display List**

  The display lists must store the vertex and normal coordinates of a dolphin and a whale.

- **Drawing functions**

  The functions must call the display list and apply proper texture to the model.

- **Swim functions**

  The functions act like a pilot, leading the animated objects to follow predesigned routes with suitable speeds and rotation.

3.1.18 Animating Submarine

- **Submarine Display List**

  The display lists must store the modeling commands to construct the submarine using fundamental geometric shapes.

- **Animating function**

  This function must manipulate the submarine to move back and forth.

3.2 User Interface:

3.2.1 Software Interface:

The user will interact with our software through a GLUT window.

3.2.2 Hardware Interface:

A sound card is needed for playing the .wav file.

3.2.3 Communication Interface:

- **GLUT Window:**

  A viewport with specified width and height for the viewer to traverse the scene.

- **Pop-up Menu:**

  A right-click menu to allow the user to quit the animation, play background music and stop the music.

3.3. Performance requirements:

The animation must meet all the functional requirements stated in 3.1. Rendering must be optimized to cause no slowdown of the animation. Keyboard interaction must be accurate. The animation must work with any .wav file.

3.4 Design Constraints:

3.4.1 Software Constraints:

The following Xcode Frameworks will support the animation:

- The SDL Framework
- OpenGL 2.0 and later
- The GLUT Framework
- Mac OS X version 10.5 or later

3.4.2 Hardware Constraints:

User computers must meet the minimum graphics requirements.

- 512 MB of internal RAM at least
- 125 MB video RAM at least
- Mouse
- Keyboard
- 1.25 GHz G4 PowerPC or faster processor
- Audio card to play music

3.4.3 Traversing Constraints:

- Users are not allowed to cross the boundaries of the scene.

3.5 Attributes:

3.5.1 Rendering Issues:

The animation was created under rigorous testing to avoid rendering slowdown. Users can traverse the scene smoothly.

Background music is played at suitable frequency and channel settings to avoid twisted sound effects. Functions must be called in suitable order to avoid an infinite loop and system slowdown.

3.5.2 Maintainability:

Code will be commented well and support more implementation.

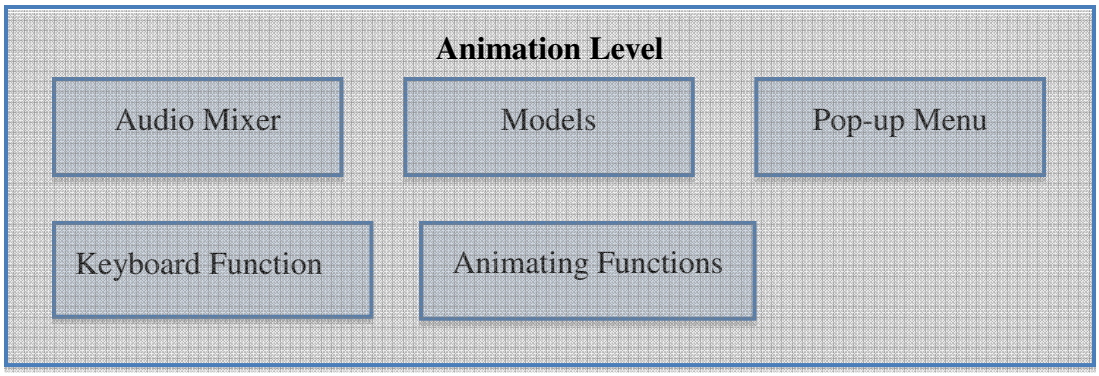
3.5.3 Transferability:

The animation will have the ability to work on Mac OS X 10.5 or later.

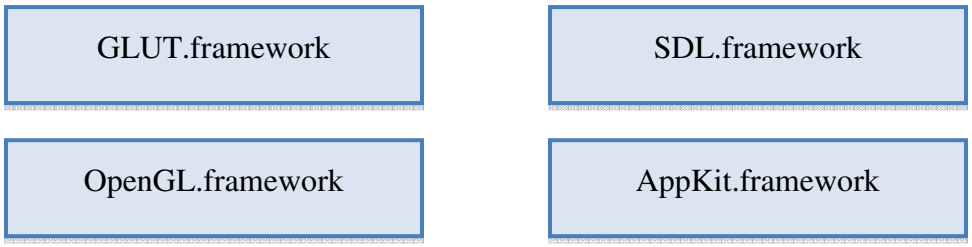
4. Appendix:

Architecture Diagram:

This architecture diagram is included for further understanding of the different layers of the animation and how they will work together.



Application Framework



References:

http://www.libsdl.org/intro.en/usingsound.html

http://pyopengl.sourceforge.net/documentation/manual/gluNewQuadric.3G.xml

http://wiki.libsdl.org/moin.cgi/SDL_LoadWAV

http://usa.autodesk.com/adsk/servlet/index?siteID=123112&id=9469002

http://www.opengl.org/about/overview/#1

http://developer.apple.com/library/mac/#documentation/MusicAudio/Conceptual/CoreAudioOverview/CoreAudioFrameworks/CoreAudioFrameworks.html

http://www.meandmark.com/sdlopenglpart1.html

http://gpwiki.org/index.php/C:How_to_set_up_your_SDL_Build_Environment#MacOS_X:_Xcode

http://glprogramming.com/red/chapter07.html

http://www.songho.ca/opengl/gl_projectionmatrix.html

http://glprogramming.com/red/appendixe.html

http://glprogramming.com/red/chapter02.html#name5

http://www.teachenglishinasia.net/asiablog/asian-water-lilies-and-lotus-flowers

http://itc.blogs.com/greenstream/blogs_about_water/

http://www.cs.uregina.ca/Links/class-info/405/WWW/Lab3/

http://lcs.syr.edu/faculty/baruch/CIS425625/Lectures/Lectures.html