Final Project Proposal

Rendering of a Vampire's Study

Written by

Luiza Nogueira Costa

ID 40124771

COMP 371, Section BL, Winter 2022

February 16th, 2022

INTRODUCTION

This project proposal falls under the category of real time rendering using OpenGL, particularly within the rendering-focused section of related topics. It assumes a level of expertise comparable to an introduction to computer graphics course and the availability of external libraries and open-source assets or 3D object models for its completion.

MOTIVATION

As a lifelong fan of the horror genre, particularly of gothic tales in the vein of Bram Stoker's Dracula, the importance of elements like the setting, scenery, and visual atmosphere in the pursuit of storytelling have always intrigued me. The use of lighting and texture in particular can increase the depth of the tale being told – dark clouds seen through a foggy window, candlelight that barely illuminates the way through a corridor, water leaking down stone walls in a dungeon cell, or even the appearance of fresh blood on the floor. Being able to translate such sensorial experiences through computer graphics seems to me an important step in advancing the field of virtual reality and interactive storytelling, which is why I would like to challenge myself into trying my hand at it by means of this proposed final project.

DESCRIPTION

The aim of this project is to render an interactive scene in OpenGL depicting a vampire's study. The user will be able to employ the first-person camera and keyboard controls to walk around the room and explore items belonging to this mysterious character. 2D textures will be used for rendering assets such as the walls, the floors, and other scene objects. A foggy window will also be rendered, from which dim afternoon light shines in as the sun presumably sets outside; the colored rays will emanate across the room and project onto the floors. Further light sources will include flickering candles added into the darker corners of scenery, which will be used to create the ominous lighting effect fitting for a horror story. Lastly, a shallow puddle of blood will be seen upon the floor, featuring a wet shader that reflects the environment around it and further blood dripping from a glass spilled over the table and onto the puddle.

Example images showing various complexities of graphics for similar scenes found in horror-focused animated movies and video games have been included in the appendix section of this proposal; however, to keep the project within the scope of the course and timeline, the intention is to retain matters simple, although thorough, and focus on applying the topics covered in lectures in a more holistic fashion.

OBJECTIVES

The proposed objectives to be concluded for the project are as follows:

1. Set up of the OpenGL environment for the rendering of the 3D scene. Creation of a room with solid walls and floors, followed by the inclusion of base objects to populate the scene and their respective 2D textures.

The initial stage of the project creation will involve the student setting up the mainframe for running the scene. This work relates to the content discussed in the first three labs pertaining to the course. It constitutes the bulk of importing, scaling, and displaying the objects in the scene. The assets used will be primarily taken from open-source libraries [1], as the modelling of so many .obj files from scratch using tools like Blender would pose an additional skill requirement on this objective that may not be achievable within the timeframe of this project. Though this list may change depending on availability, these will

include: a desk, bookshelves, a chair, an armchair, curtains, a glass window, candles, a dimming fireplace, and additional decorations pertaining to the gothic theme. Textures will be added following the standard 2D texture incorporation.

2. Implementation of a first-person camera that allows the scene to be explored using cursor and keyboard input.

To create the first-person camera view [2], the mouse input must be confined to the center of the window once the application starts, so that the program may record its current position within the rendering loop to keep track of any movements performed from one frame to the other. Then, the effect of looking around the scene can be replicated by rotating the coordinates around its y-axis (for left-right movements) and x-axis (for up-and-down movements). Keyboard input will be used to simulate motion around the 3D space, which can be done with the glutKeyboardFunc() method and keyboard functions that track when certain keys are pressed and released. To reach a point (a, b) in the plane, for example, we may translate the world to (-a, -b). Combining the rotation and translation, followed by storing the new camera position, will result in the first-person camera behavior.

3. Implementation of atmospheric lighting in the form of rays passing through the glass window.

The intent of this objective is to have a window object in the scene from which sunrays emanate, travel past the particles in the air, and project onto the floor, which can be simulated by employing real-time volumetric light scattering and projective texture mapping. As described in this progress report for a similar project [3], generic refraction through glass may be combined with projective texture mapping and Blinn-Phong shading with falloff, followed by multipass rendering, to produce the desired outcome. Coloring can be added to the light rays to simulate the late-afternoon appearance.

4. Addition of further cinematic lighting through candlelight simulation.

The simplified rendering of candlelight without the implementation of a full particle-simulation procedure may be achieved through the use of directional and point lights [4] stemming from the candlewick, on top of which a bloom effect may be applied [9]. The intensity of the light source may be adjusted dynamically through the rendering loop by having its value altered with each frame displayed, thus recreating the flickering effect without need of true particles being generated. A more complex, though more realistic alternative is to render an already animated candle frame-by-frame [10][11] and apply the lighting effects over it.

5. Rendering of blood atop surfaces through a wet shader.

In order to have puddles of blood rendered above surfaces such as the ground and the desk, the first step involves the creation of the reflection of the scene above the liquid. This can be achieved by transferring the scene image into a cubemap [5], which is then passed to the puddle objects as a texture so they may be used in the reflection rendering [6]. Alternatively, screen space reflections may be used [15]. The goal is to create the reflective effect of a shallow accumulation of water along with the viscous red coloration of fresh blood. As an improvement, additional real-time rendering of blood droplets can be implemented using a blood shader in GLSL [13][14].

APPENDIX :

Picture 1. Image of the Tremere Chantry in Vampire: The Masquerade – Bloodlines [7].

Picture 2. Image of the Tremere Chantry in Vampire: The Masquerade – Bloodlines [7].

Picture 3. Screenshot of light passing through a window in Resident Evil Village [8].

Picture 4. Screenshot of a pool of blood in Resident Evil Village [8].

BIBLIOGRAPHY

[1] "Free 3D models," *Free3D*. [Online]. Available: https://free3d.com/3d-models/obj?page=2. [Accessed: 06-Feb-2022].

[2] R. Thakur, "Setting up a first person camera in opengl to move around in your 3D world," *Setting up a first person camera in OpenGL to move around in your 3D world*, 24-Feb-2018. [Online]. Available: https://thepentamollisproject.blogspot.com/2018/02/setting-up-first-person-camera-in.html. [Accessed: 06-Feb-2022].

[3] "Rays of light through stained glass windows - github pages." [Online]. Available: https://mdzahidh.github.io/cs148/assets/best/visuals/01-report.pdf. [Accessed: 06-Feb-2022].

[4] "Multiple lights," *LearnOpenGL*. [Online]. Available: https://learnopengl.com/Lighting/Multiple-lights. [Accessed: 06-Feb-2022].

[5] "Cubemaps," *LearnOpenGL*. [Online]. Available: https://learnopengl.com/Advanced-OpenGL/Cubemaps. [Accessed: 06-Feb-2022].

[6] "How can I make a 'wet surface'/'shallow puddle' shader in ..." [Online]. Available: https://gamedev.stackexchange.com/questions/153619/how-can-i-make-a-wet-surface-shallow-puddle-shader-in-unity. [Accessed: 06-Feb-2022].

[7] "Vampire: The masquerade – bloodlines," *Wikipedia*, 08-Jan-2022. [Online]. Available: https://en.wikipedia.org/wiki/Vampire:_The_Masquerade_%E2%80%93_Bloodlines#Accolades. [Accessed: 06-Feb-2022].

[8] "Resident evil village," *Wikipedia*, 06-Feb-2022. [Online]. Available: https://en.wikipedia.org/wiki/Resident_Evil_Village. [Accessed: 06-Feb-2022].

[9] "OpenGL 3D game tutorial 47: Bloom Effect - YouTube." [Online]. Available: https://www.youtube.com/watch?v=LyoSSoYyfVU?__ref=vk.api. [Accessed: 14-Feb-2022].

[10] "Skeletal animation," *LearnOpenGL*. [Online]. Available: https://learnopengl.com/Guest-Articles/2020/Skeletal-Animation. [Accessed: 14-Feb-2022].

[11] "3D Animated Candle," *TurboSquid*. [Online]. Available: https://www.turbosquid.com/3d-model/free/animated/candle. [Accessed: 14-Feb-2022].

[12] "Water drop 3b – physically based wet surfaces," *Sébastien Lagarde*, 08-Sep-2013. [Online]. Available: https://seblagarde.wordpress.com/2013/04/14/water-drop-3b-physically-based-wet-surfaces/. [Accessed: 14-Feb-2022].

[13] "Real-time blood in GLSL 2," *YouTube*. [Online]. Available: https://www.youtube.com/watch?v=7lpKlbArXCM&ab_channel=MattiasLagergren. [Accessed: 15-Feb-2022].

[14] "Rendering of Water Drops in Real-Time Ines Stuppacher." (2007). Supan, Peter. [Online]. Available: "https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.102.7409&rep=rep1&type=pdf"

[15] "OpenGL Tutorial - Screen Space Reflections," *YouTube*. [Online]. Available: https://www.youtube.com/watch?v=K2rs7K4y_sY&ab_channel=NullPointer. [Accessed: 15-Feb-2022].