**Project 10: Advanced Shaders II**

CST 310: Computer Graphics

Connor Seimears & Ty Gehrke

Professor Citro

12/12/2023

**Loom**

<https://www.loom.com/share/12adf795deae4dd9994adf7da9d08eec?sid=18676639-1dc5-43a9-a27a-9a02704f4767>

**Theoretical Background**

Environment Mapping, Parallax Mapping, and Bump Mapping are techniques to add intricate surface details, reflections, and shading effects.

**Environment Mapping (Cube Mapping):**

* **Concept:** Simulates reflections using a cube map, wrapping the scene inside a cube surrounding the camera.
* **Mathematics:** Involves transforming reflection vectors using the cube map, providing realistic reflections on the sphere's surface.
* **Example:** Utilize a Humus website cube map, slicing it into six images corresponding to cube faces.

**Parallax Mapping:**

* **Concept:** Enhances surface depth perception by simulating displacement mapping.
* **Mathematics**: Uses height maps to displace texture coordinates, creating the illusion of surface relief.
* **Example:** Apply parallax mapping to the cube using a Humus website cube map, slicing it into six images corresponding to cube faces.

**Bump Mapping:**

* **Concept:** Creates the appearance of detailed surfaces by perturbing normals during lighting calculations.
* **Mathematics:** Involves perturbing normals based on a height map, influencing lighting calculations.
* **Example:** Implement bump mapping for the cylinder and sphere using 'Bump-Map.jpg' and 'Bump-Picture.jpg'.

**Mathematical Functions and Models**

**Environment Mapping:**

* **Functions:** Reflection vector transformations, cube map sampling.
* **Models:** Utilize spherical coordinates for the sphere's surface.

**Parallax Mapping:**

* **Functions:** Texture coordinate displacement.
* **Models:** Use height maps to perturb texture coordinates.

**Bump Mapping:**

* **Functions:** Normal perturbation based on height map.
* **Models:** Apply height map values to adjust surface normals.

**Texture Selection:**

* **Rationale:** Choose appropriate Humus cube map for realistic reflections.
* **Considerations:** Ensure bump and height maps align with the visual aesthetics of the cube and cylinder.

**Programming Concepts**

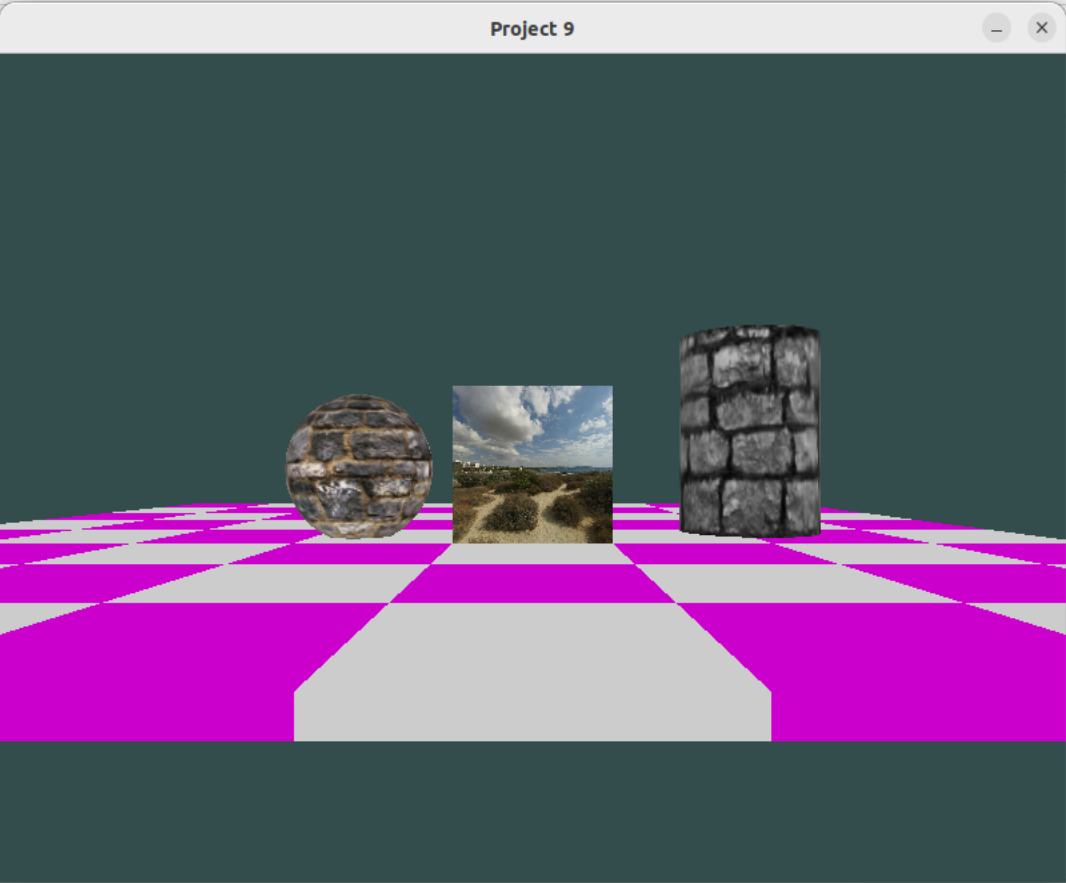
This code snippet involves several programming concepts related to graphics programming using OpenGL and C++:

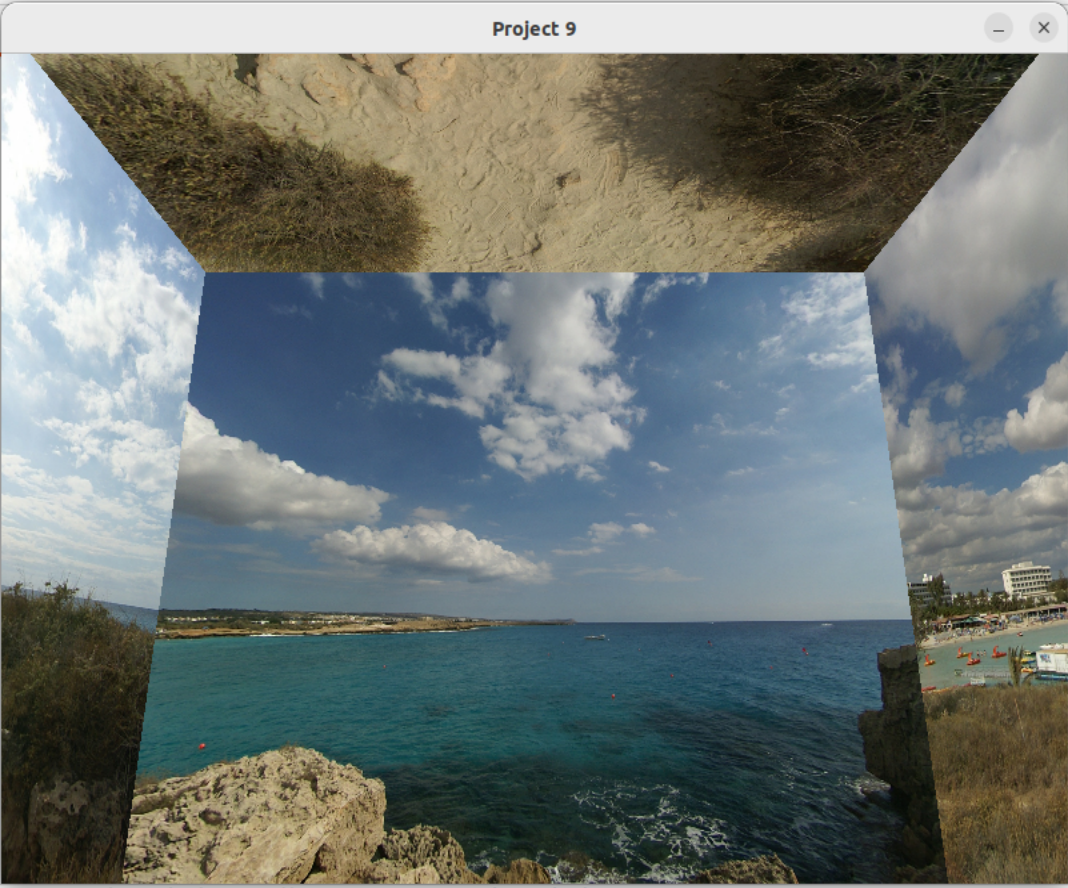
* **OpenGL:** The code utilizes the OpenGL library for graphics rendering. It sets up a window, defines shaders, manages vertices, and renders models using OpenGL functions.
* **GLFW:** GLFW is used for creating windows, contexts, handling input, and events. It's a popular library for OpenGL.
* **GLEW:** GLEW, the OpenGL Extension Wrangler Library, manages OpenGL extensions and provides access to them.
* **Shaders:** The code loads and uses vertex and fragment shaders for rendering different objects like a cube, sphere, cylinder, and a checkerboard.
* **Matrices:** Matrices from the GLM (OpenGL Mathematics) library are used for transformations, like creating view and projection matrices for the camera and models.
* **Model Loading:** It loads models from external files (OBJ format) using a model loading library or custom implementation.
* **Keyboard Input Handling:** GLFW callback functions manage keyboard input for controlling camera movement and interaction within the scene.
* **Delta Time Calculation:** The code calculates delta time for smooth camera movement, which involves handling time differences between frames.
* **OpenGL Rendering Loop:** The code operates within an infinite loop to continually update and render the scene until the window is closed.
* **Resource Management:** The code allocates and deallocates resources (like VAOs and VBOs) to efficiently manage memory usage.
* **Parallax Mapping:** This technique is used to simulate depth in textures, giving the illusion of 3D depth without adding geometry. It manipulates texture coordinates based on a height map or displacement map to create this effect.
* **Bump Mapping:** Bump mapping is a method to simulate bumps and dents on the surface of an object without changing its geometry. It uses a normal map to perturb surface normals during rendering, creating the appearance of detailed sur faces.

Overall, the code demonstrates the implementation of a simple 3D scene using OpenGL in C++, including setting up a window, managing shaders, rendering models, handling user input, texture mapping and manipulating the camera.

**Screenshots**

Program Output:





**References**

LearnOpenGL. (2023). *Cubemaps*. https://learnopengl.com/Advanced-OpenGL/Cubemaps