Project 4: Render Your Scene w/ Primitives

Course: CST-310 TR1100A Dr. Citro

Date: October 6th, 2024

Students: Dayana Gonzalez Cruz, Yulissa Valencia

Github Link : https://github.com/DayanaGCruz/CST-310-YSW

Project Description
Part 1.
The purpose of this project is to identify three scenes and photograph them from two varying angles. In the following assignment, one will be tasked with drawing out a number of primitive shapes in the scene. Following that, it will be replicated using OPENGL primitives. Other assignments will include adding lighting and colors with OPENGL.

All pictures were personally taken and not retrieved from the internet or otherwise.

Part 2.
A photographed scene was chosen by the instructor. The geometrical composition of the scene and the difficulties involved with rendering it were discussed. Finally, a hand-drawn geometrical representation of the objects in the scene was drawn, and the original photograph was showcased side-by-side with the drawing.

Three Beginning Scenes

A.
Approach: This scene was chosen because it has many simple and block shapes present in the foreground and background. It has dynamic, complex lighting and colorful tones. The window view has more shapes to include also.

1. 
2. 

B.

Approach: This scene was chosen as it has natural lighting, many tiny shapes in the form of tree leaves and mandarins as well as a center shape of a swing with smaller compositional shapes, and other shapes in back to draw out, too.

3.  4. 

C.

Approach: The decision to select this scene is that is has many larger objects with smaller compositional objects and features a variety of colors. There is lighting present but it is less complicated than some of the other scenes.

5.  6.

Instructor-Selected Picture:

Objects in scene:

Foreground

- Potted tree
  - → Pot : Cylinder
  - → Trunk: Cylinder
  - → Leaves: Triangles
- Geometrically patterned rug → Rectangular prism

Mid-ground

- Desk chair
  - → Chair : Cuboids
  - → Swivel Mechanism : Cylinders
- Air conditioner
  - → Base : rectangular prism
  - → Holes : Rectangular prisms
- Whiteboard → rectangular prism
- Bookshelf → rectangular prisms
- Computer Desk → rectangular prism

- Orchid
  - → Pot : Cylinder
  - → Stem : Cylinder
- Display monitor → rectangular prism
- Window → rectangular prism
- Window Blinds → Many rectangular prisms
- Television → rectangular prism
- Printer → rectangular prism
- Bookshelf → Many rectangular prisms
- Potted shrub
  - → Pot : Cylinder
  - → Stem : Cylinder
  - → Shrub : Sphere
- Dog treat bag → Pentagonal Prism
- Pig toy → Sphere
- PC Tower → Rectangular prism and spheres
- Keyboard
  - → Base : Rectangular prism
  - → Keys : Many cylinders
- Seashell light
  - → Shell : Many cylinders and spheres
  - → Light : Sphere
- Candle → Cylinder

Main objects:

- Potted tree
- Computer Desk
- Display monitor
- PC tower
- Bookshelf
- Orchid
- Television
- Air Conditioner

Key characteristics:

- Strong separation between flooring color and pattern and white wall
- Angles in floor, wall, and ceiling planes
- Lighting from window being sliced seen in left side of photo
- Colorful geometric rug
- Light reflections on some reflective objects (displays)
- Text and patterns on some objects (candle, dog treat bag, rug, etc.)

Rendering Difficulty Ranking (Decreasing):

1. Potted Tree
→ Representing the curved nature of the tree trunk and the sheer number of leaves requires a good algorithm to procedurally
Generate.

2. Seashell Light
→ Complex curved surfaces with intricate light refraction patterns make this challenging to render realistically.

3. Chair
→ Involves both smooth and rigid shapes with different materials (e.g., cushion vs. metal legs), which may also require accurate physics simulation.

4. Orchid
→ The delicate, thin petals and organic curvature of flowers demand detailed surface modeling and accurate shading techniques.

5. Keyboard
→ The numerous keys and fine texturing make this more difficult.

6. Air Conditioner
→ Typically boxy with simple geometric shapes, but the grille patterns and reflective surfaces add some complexity.

7. PC Tower
→ The rectangular form is simple, but additional features like ventilation holes, lighting, and texturing bring added difficulty.

8. Window + Blinds
→ The blinds introduce multiple slats requiring careful shadow and light management.

9. Bookshelf→ A bookshelf may have numerous objects (books, décor), varying in size, color, and shape, which adds to its overall complexity.

10. Pig Toy
→ Its organic yet simple shape may involve attention to detail on the small components like the legs and face.

11. Display Monitor

→ Mostly rectangular with clean lines, but reflections on the screen add complexity.

12. Potted Shrub
→ Similar to the potted tree, but with fewer leaves and less complex branch structures, making it easier to handle procedurally.

13. Whiteboard
→ Simple rectangular shape with little to no reflection.

14. Printer
→ A boxy shape with minor surface details like buttons, trays, and texturing.

15. Geometrically Patterned Rug
→ A flat surface, but complex patterns and potential textural variation make it slightly harder than other flat objects.

16. Dog Treat Bag
→ The crinkled surface, transparency, and printed designs on the bag are more challenging than other small objects.

17. Candle
→ Simple cylindrical shape. Some text pattern with the label but very far away.

18. Television
→ A simple rectangular object with minimal surface detail, although rendering reflective glass adds difficult.

Image:

Hand-drawn Image:

PROJECT 4

Challenges Translating from Hand-art to OpenGL
Translating from hand-drawn primitives to OpenGL opened a door to a realm of new complexities. A drawing is still a 2D representation; hence, being translated into 3D primitives in OpenGL, new complexities emerged with the location and scale of the object in relation to the other objects in the scene. For instance, the joints of complex objects which are composed of a plethora of smaller primitives are difficult to orient.

1. **Object: Television**
    a. Mathematical characteristics:
        i. Dimensions:
            1. Width (X): 7.43 m
            2. Height (Y): 0.387 m
            3. Depth (Z): 4.26 m
    b. Primitives used: GL_TRIANGLES

c. Transformations: The TV is translated to the coordinates (-8.4428, -6.1356, 10.277) in the scene to position it in its final location. The television is rotated by -33.288 degrees around the z-axis, which titles it to face towards the left.
d. Shader(s): Vertex shader, handles the transformations for the television (translation, rotation, scaling) and applies the model, view, and projection matrices to position the television in the scene. Fragment shader applies the black color to the TV.

2. **Object: Computer Desk**
   a. Mathematical characteristics:
      i. Width (X): 6.44 m
      ii. Height (Y): 2.7 m
      iii. Depth(Z): 0.94 m
   b. Primitives used: GL_TRIANGLES
   c. Transformations: The object is translated to (-0.28708, -3.9235, 4.3692) to position it in the scene.No rotation is applied. The object is scaled by factors 3.222 (X), 1.352 (Y), and 0.470 (Z).
   d. shader(s)**:**Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid color to simulate the object color

Computer Desk Leg 1

a. Mathematical characteristics:
   i. Width (X): 0.161 m
   ii. Height (Y): 0.256 m
   iii. Depth(Z): 3.23 m
b. Primitives used: GL_TRIANGLES
c. Transformations: The object is translated to (2.1107, -2.9647, 2.7022) to position it in the scene. Rotated by -4.0704° around the X-axis and 91.758° around the Z-axis. The object is scaled by factors -0.080 (X), -0.128 (Y), and 1.615 (Z), which mirrors the object along the X and Y axes.

d. shader(s):Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid color to simulate the object color

Computer Desk Leg 2

a. Mathematical characteristics:
    i. Width (X): 0.165 m
    ii. Height (Y): 0.284 m
    iii. Depth(Z): 2.94 m
b. Primitives used: GL_TRIANGLES
c. Transformations: The object is translated to (-2.5854, -2.8338, 2.5128) to position it in the scene. Rotated by 6.0588° around the X-axis and 83.719° around the Z-axis. The object is scaled by factors -0.083 (X), -0.142 (Y), and 1.471 (Z), which mirrors the object along the X and Y axes.
d. shader(s):Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid color to simulate the object color

3. **Object: Display Monitor**
    a. Mathematical characteristics:
        i. Width (X): 3.38 m
        ii. Height (Y): 0.204 m
        iii. Depth(Z): 1.79 m
    b. Primitives used: GL_TRIANGLES
    c. Transformations: The object is translated to (0.55044, -4.7396, 6.4128) to position it in the scene. No rotation is applied. The object is scaled by factors 1.688 (X), 0.102 (Y), and 0.896 (Z).
    d. shader(s):Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid color to simulate the object color

Display Monitor Stand

a. Mathematical characteristics:
    i. Width (X): 0.448 m
    ii. Height (Y): 0.162 m
    iii. Depth(Z): 1.04 m
b. Primitives used: GL_TRIANGLES
c. Transformations: The object is translated to (0.55044, -4.8898, 5.4713) to position it in the scene. No rotation is applied. The object is scaled by factors -0.224 (X), 0.081 (Y), and 0.521 (Z).
d. shader(s):Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid color to simulate the object color

4. **Object: PC tower**
    a. Mathematical characteristics:
        i. Width (X): 0.975 m
        ii. Height (Y): 1.36 m
        iii. Depth(Z): 2.13 m
    b. Primitives used: GL_TRIANGLES
    c. Transformations: The object is translated to (-2.1612, -4.1724, 5.9142) to position it in the scene. No rotation is applied. The object is scaled by factors 0.487 (X), 0.681 (Y), and 1.065 (Z).
    d. shader(s): Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid color to simulate the object color.
5. **Object: Bookshelf**
    a. Mathematical characteristics:
        i. Width (X): 4.03 m
        ii. Height (Y): 6.06 m
        iii. Depth (Z): 0.154 m
    b. Primitives used: GL_TRIANGLES

c. Transformations: The object is translated to (-9.9239, -5.6566, 4.86) to position it in the scene. Rotated by -90° around the X-axis and -31.504° around the Z-axis. The object is scaled by factors 2.013 (X), -3.028 (Y), and 0.077 (Z).
d. shader(s): Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid color to simulate the object color.

6. **Object: Chair**
    a. Mathematical characteristics:
        i. Width (X): 1.66 m
        ii. Height (Y): 0.548 m
        iii. Depth (Z): 2.43 m
    b. Primitives used: GL_TRIANGLES
    c. Transformations: The object is translated to (-0.15615, -1.6445, 3.5727) to position it in the scene. Rotated by 89.73° around the X-axis. The object is scaled by factors 0.832 (X), 0.274 (Y), and 1.217 (Z).
    d. shader(s): Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid color to simulate the object color.
7. **Object: Potted Shrub**
    a. Mathematical characteristics:
        i. Width (X): 1.82 m
        ii. Height (Y): 1.63 m
        iii. Depth (Z): 1.16 m
    b. Primitives used: GL_TRIANGLES
    c. Transformations: The object is translated to the coordinates (8.469, 0.20914, 1.2832) to position it correctly in the scene. No rotation is applied. The object is scaled by factors -0.910 (X), -0.814 (Y), and 0.582 (Z), indicating mirroring along the X and Y axes.
    d. shader(s): Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid color to simulate the object color.

8. **Object: Whiteboard**
    a. Mathematical characteristics:
        i. Width (X): 5.29 m
        ii. Height (Y): 0.341 m
        iii. Depth (Z): 0.753 m
    b. Primitives used: GL_TRIANGLES
    c. Transformations: The whiteboard is translated to the coordinates (7.2669, -6.7356, 2.4243) to position it correctly in the scene. It is slightly rotated by 0.2064 degrees around the Z-axis. The whiteboard is scaled by factors 2.643 (X), 0.171 (Y), and 1.880 (Z) to match its dimensions.
    d. shader(s): Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid color to simulate the object color.
9. **Object: Printer**
    a. Mathematical characteristics:
        i. Dimensions:
            1. Width (X): 1.68 m
            2. Height (Y): 1.55 m
            3. Depth (Z): 0.753 m
    b. Primitives used: GL_TRIANGLES
    c. Transformations: The printer is translated to the coordinates (-8.7055, -4.7787, 4.1802) in the scene. It is rotated by -30.187 degrees around the z-axis towards the left. The printer is scaled by the factors 0.842(X), 0.776(Y), and 0.377(Z).
    d. shader(s): Vertex shader handles the transformations and applies the model, view, and projection matrices. Fragment shader applies a solid black color to simulate the printer color.
10. **Object: Rug**
    a. Mathematical characteristics:
        i. Dimensions:
            1. Width (X): 7.11 m
            2. Height (Y): 1.55 m

3. Depth (Z): 0.753 m

b. Primitives used: GL_TRIANGLES

c. Transformations: The rug is translated to the coordinates (-0.28708, -0.32484, 1.0759) to position it correctly in the scene. No rotation is applied. The rug is scaled by factors 3.554 (X), 2.586 (Y), and 0.056 (Z) to match its dimensions.

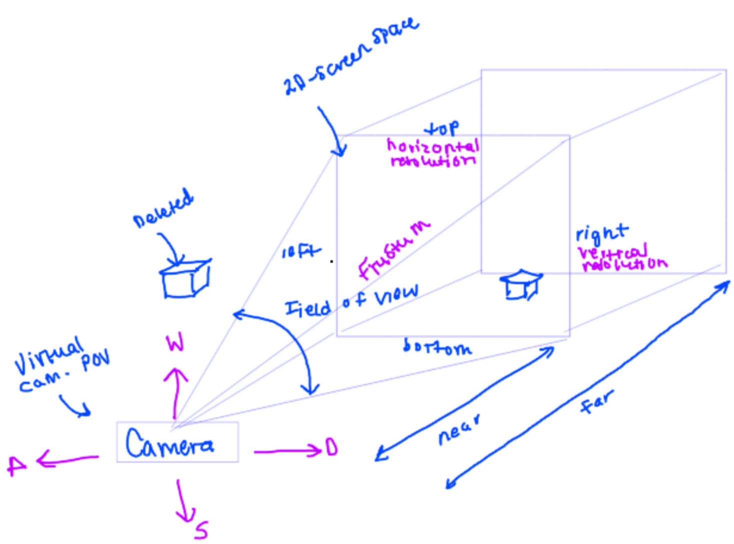
d. shader(s): Vertex shader for handling of transformations and fragment shader for color.

Position and Use of Virtual Camera:

The virtual camera is perspective-based rather than orthogonal. Objects seem smaller the farther away that they are. Moreover the camera responds to user input through the mouse and WASD keys, changing its position and orientation.

Diagram:

The following diagram illustrates how a virtual camera in 3D space projects objects onto a 2D screen. Positioned at the bottom left, the camera captures objects within its field of view, which is shown as a conical projection toward the screen. The 3D objects are projected onto the near plane, where the screen space is represented in terms of horizontal and vertical resolution. The perspective effect of the camera makes objects further from the camera appear smaller. The near and far planes define the visible range of depth, and the camera's tilt or rotation affects how the objects are aligned on the screen, demonstrating how the virtual camera creates the illusion of depth and perspective in a 3D scene. The triangle of which the virtual camera is the apex represents the frustum, what is visible to the camera. That which is outside the frustum is not rendered and is discarded. The WASD shows that the virtual camera can move with user input.



Screenshots of Execution of Code:

Note: *Program logic is bugged in terms of loading and rendering materials from .obj and .mtl files passed and will hopefully be resolved when lighting and textures are added. .webm file is submitted alongside assignment for better clarity.*

Below screenshot depicts what the model's materials look like in the blender environment.