CGRA354 Computer Graphics Programming T1/2024

Assignment #2: Transformations and shading (15 points)
Assigned: Monday, 18th March
Due: 23th April 2024 11:59pm

The areas addressed in this assignment include the use of Transformations, Object Instancing and Advanced Shading techniques in Computer Graphics programming.

Codebase

You are provided with a cross-platform programming framework specifically designed for the Computer Graphics assignments. The framework performs the initialization of the OpenGL pipeline allowing to focus on the assignment tasks. Comprehensive instructions for building the framework on different platforms can be found in the `README.md` file located in the source code root folder.

Turn in procedure

You should submit your work as a zip file using the ECS submission system. Please name your file as `<LastName><FirstName>-Assignment<X>.zip` where X is the assignment number. When your file is unzipped you should have:

1. The C++ code and programs you have written. You should use files in the form of the samples given, rather than producing files from scratch. This will help us follow your code.
2. Sample images created by your program. You can save these by taking a screenshot or use a screen capture tool.
3. A readme.md, txt, pdf or MS Word file that answers any questions posed, and that lists the input used to create the images you include. Alternatively you can provide a narrated screen-capture video showing the functionality of your program.
4. Any other information or supporting documentation to help us run and evaluate your submission.

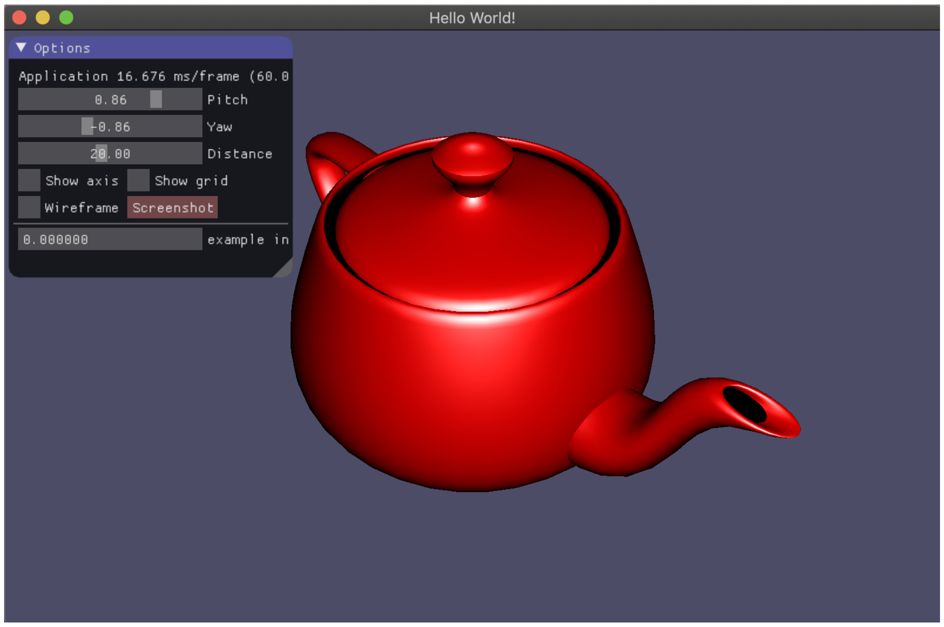
If your programs fail on the machines used for grading, you may be asked to bring in your system to demonstrate that the files you submitted functioned in the environment you worked in.

Grading rubrics

There are multiple questions in this assignment, with some question having multiple subparts corresponding to a particular tier (core, completion, challenge and writing).

Part 1: Transformations and shading (15 points)

In this exercise, you will implement the control of the camera and 3D model using keyboard and mouse input. You will be working with OpenGL commands for drawing with a basic GUI/mouse support in order to specify the transformations parameters to render with. You will apply the Phong Lighting Model and combine it with a basic texture mapping pipeline to render multiple objects with different transformations. Finally, you will calculate and display the bounding boxes for the objects.



Application 16.676 ms/frame (60.0
0.86 Pitch
-0.86 Yaw
20.00 Distance
0.000000 example in

Figure 1: The Utah teapot model rendered with Phong shading model.

Core (8 points)

You are to implement the following functionality:

- Orbital camera controls, where the camera is focused on a single point and can be controlled using rotation (yaw and pitch) and translation (distance). The camera should be controlled by both the ImGui interface (with a slider for each control) and by the mouse (mouse dragging on the x and y axis changes the yaw and pitch respectively, and mouse wheel controls the distance).
- The Phong shading model with ImGUI controls allowing the user to interactively select its parameters (ambient/diffuse colour, specular colour, and shininess). You may assume that there is only one light in the scene, either a point light at the position of the camera, or a directional light that faces the same direction as the camera.

Completion (5 points)

In this part of assignment you will draw multiple objects and extend the shading model to incorporate texturing of the objects in the scene. This will involve:

- Creating multiple objects (at least 100) with the same model but different (arbitrary selected) positions, rotations, scales and material colours (for Phong shading). This should be done by re-using the already loaded shader and mesh (meaning they should only be loaded from disk once).
- Implement texture mapping where an image is used as a texture for defining the ambient/diffuse for each model instead of a flat material colour. You may use any texture, including the one provided.

Challenge (2 point)

Calculate and display the bounding boxes for each object in the scene by:

- Calculating axis-aligned bounding boxes (AABB) for each object in your scene in model-space.
- Creating a mesh for the bounding box and displaying it using the same model-view transform as the original object.

Note: You must use the OpenGL Mathematics (GLM) facilities on this assignment.

References

[1] Wiki: Phong shading,
`https://en.wikipedia.org/wiki/Phong_shading`

[2] GLM manual,
`http://glm.g-truc.net/glm.pdf`

[3] Khronos group: Instancing,
`https://www.khronos.org/opengl/wiki/Vertex_Rendering#Instancing`

[4] Bounding box,
`https://en.wikibooks.org/wiki/OpenGL_Programming/Bounding_box`

[5] Axis-aligned bounding boxes,
`https://developer.mozilla.org/en-US/docs/Games/Techniques/3D_collision_detection`