UOW
AUSTRALIA

SCIT

School of Computing & Information Technology

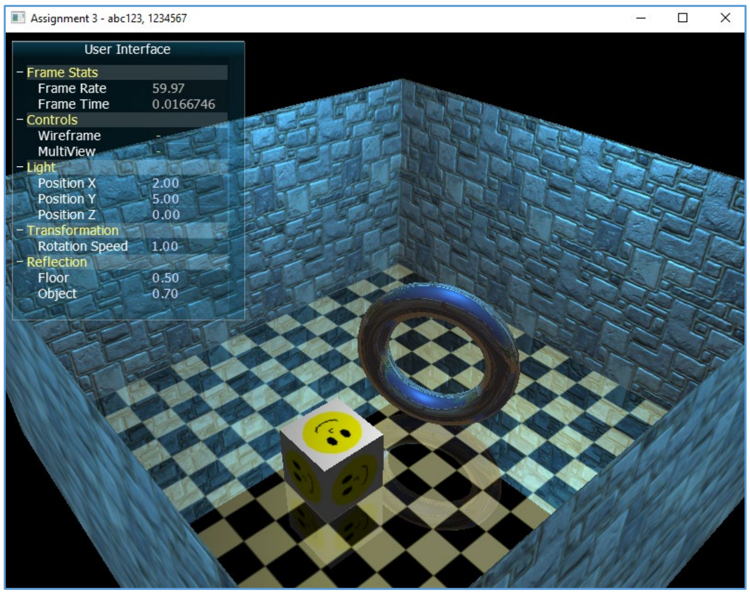
CSCI336 – Interactive Computer Graphics
Spring 2023

Assignment 3

Due on Friday, 27th October 2023 at 17:00

Task

Write an OpenGL program that displays a textured 3D scene with lighting and a Graphical User Interface (GUI). Figure 1 shows the scene (a working program will be demonstrated in one of the lectures). Put your UOW username and ID number in the window title.



```
- Frame Stats
    Frame Rate         59.97
    Frame Time         0.0166746
- Controls
    Wireframe          -
    MultiView          -
- Light
    Position X         2.00
    Position Y         5.00
    Position Z         0.00
- Transformation
    Rotation Speed     1.00
- Reflection
    Floor              0.50
    Object             0.70
```

Figure 1: 3D scene.

- Scene geometry (2 marks)

  Construct a room with four walls and a floor (no ceiling), as shown in Figure 1. Create the room by specifying vertex positions, surface normals, texture coordinates, etc.

  The room should contain two objects, loaded from two different model files: a torus and a cube

  Your program should have a movable camera that the user can control using keyboard and mouse input. To keep camera rotation separate from interaction with the GUI, the camera should only rotate when the right mouse button is pressed and held.

- Lighting and textures (3 marks)

  The room should be lit with a point light source. Provide an interface element for the user to move the light's position.

  Use different images to texture the wall, floor and cube.

- Animation (1 mark)

  The torus should rotate. Provide an interface element for the user to control its rotation speed.

- Normal mapping (2 marks)

  Render the walls of the room using normal mapping to create the appearance of bumps on the surface.

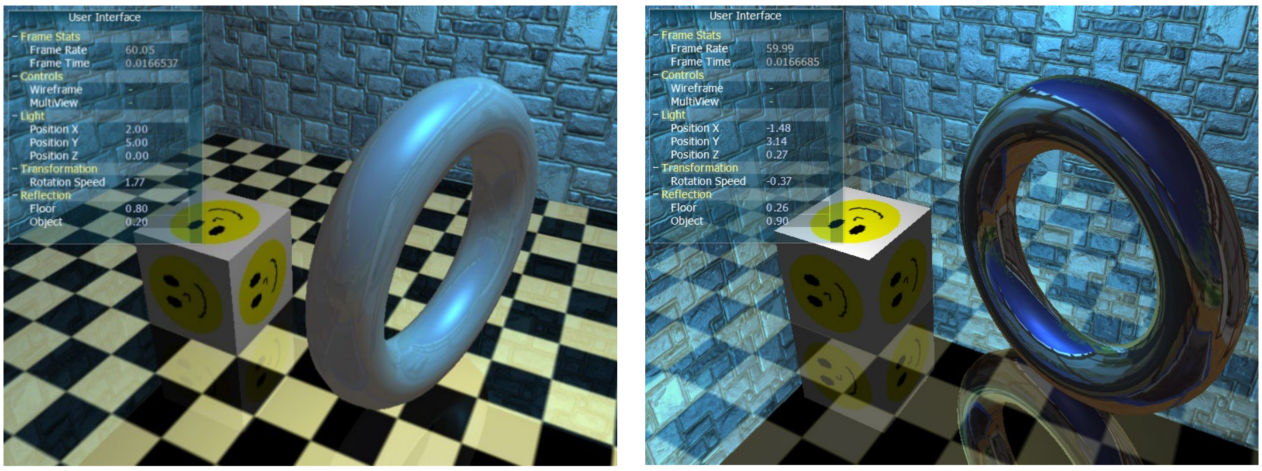
- Reflective surface (2 marks)

  The floor of the room should a reflective surface. Use the stencil buffer and blending for this. Provide an interface element for the user to control the amount of reflection.

- Cube environment mapping (2 marks)

  Render the torus using cube environment mapping. Provide an interface element for the user to control the amount of reflection.

The screenshots in Figure 2 below show the light at different positions, different rotation speeds, and different amount of reflection for the floor and the torus.



```
User Interface
- Frame Stats
    Frame Rate        60.05
    Frame Time        0.0166537
- Controls
    Wireframe         -
    MultiView         -
- Light
    Position X        2.00
    Position Y        5.00
    Position Z        0.00
- Transformation
    Rotation Speed    1.77
- Reflection
    Floor             0.80
    Object            0.20
```

```
User Interface
- Frame Stats
    Frame Rate        59.99
    Frame Time        0.0166685
- Controls
    Wireframe         -
    MultiView         -
- Light
    Position X        -1.48
    Position Y        3.14
    Position Z        0.27
- Transformation
    Rotation Speed    -0.37
- Reflection
    Floor             0.26
    Object            0.90
```

(a) (b)

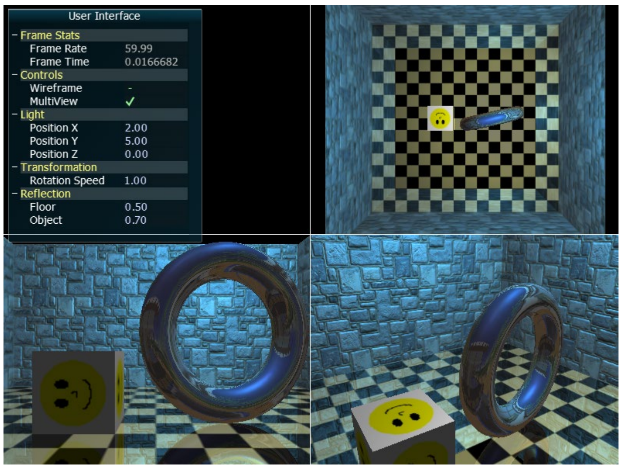
Figure 2: Different properties.

- Multiple views (3 marks)

  Provide an interface element for the user to toggle between a single view and multiple views.

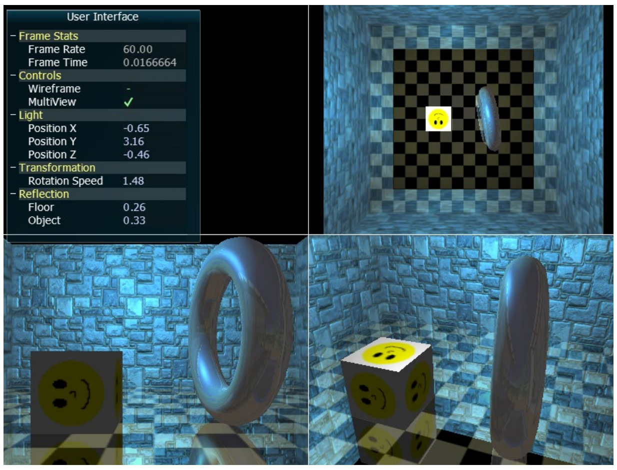
  The multiple views all use perspective projection. The top-right view is a top-down view, the bottom-left view is a front view and the bottom-right view is a view from the moving camera. Note that the bottom-right view changes based on keyboard and mouse input.

The screenshots in Figure 3 below show examples of multiple views using different properties. The bottom-left view shows the camera at different locations.



```
User Interface
- Frame Stats
    Frame Rate        59.99
    Frame Time        0.0166682
- Controls
    Wireframe         -
    MultiView         ✓
- Light
    Position X        2.00
    Position Y        5.00
    Position Z        0.00
- Transformation
    Rotation Speed    1.00
- Reflection
    Floor             0.50
    Object            0.70
```

(a)



```
User Interface
- Frame Stats
    Frame Rate        60.00
    Frame Time        0.0166664
- Controls
    Wireframe         -
    MultiView         ✓
- Light
    Position X        -0.65
    Position Y        3.16
    Position Z        -0.46
- Transformation
    Rotation Speed    1.48
- Reflection
    Floor             0.26
    Object            0.33
```

(b)

Figure 3: Multiple views.

Screenshots

In your submission, include screenshots demonstrating your working program and the features that were implemented. Save the screenshots using one of the common image formats, i.e. bmp/jpg/png.

Instructions and Assessment

Zip all source files (.cpp, .h, .vert and .frag) and **screenshots** (.bmp/.jpg/.png) into a single file. Submit this via Moodle by the due date and time (**do NOT zip your entire project file** as this can be very large and **do NOT use .rar format**). If not submitted on Moodle, the assignment will not be marked.

If you use the .obj and .bmp files from the labs, you do not have to include these with your submission.

The assignment must be your own work. If asked, you must be able to explain what you did and how you did it. Marks will be deducted if you cannot correctly explain your code.

NOTE: The mark allocations shown above are merely a guide. Marks will be awarded based on the overall quality of your work. Marks may be deducted for other reasons, e.g., if your code is too messy or inefficient, if you cannot correctly explain your code, etc.

For code that does not compile, does not work or for programs that crash, the most you can get is half the marks (i.e. 7.5 marks or less). It is better to comment out sections of your code that do not work, and include a note for the marker.