COSC363 Computer Graphics
Lab03: Illumination And Texture Mapping

Welcome to an alien world! This lab takes you to Planet-X where an alien is preparing a spaceship for lift off.

I. Spaceship.cpp:

The program displays the model of a spaceship (Fig. 1). The arrow keys can be used to rotate the scene and to move the camera up or down. The names of functions used for drawing various parts of the scene are also given in Fig.1 .

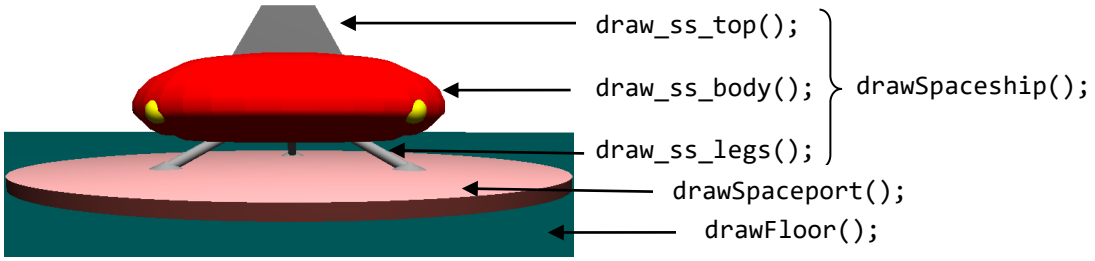


Fig. 1.

1. The floor plane is modelled using a single large quad. Such models are not suitable for displaying spotlights (see Slide [03]:27). A highly tessellated disc is therefore used as the base ("Spaceport") for the spaceship model. The scene uses GL_LIGHT0 as the primary light source. In the `initialise()` function, select a second light source GL_LIGHT1, and define it as a spotlight with properties as follows.

```
glEnable(GL_LIGHT1);
glLightfv(GL_LIGHT1, GL_DIFFUSE, white);
glLightf(GL_LIGHT1, GL_SPOT_CUTOFF, 40);
glLightf(GL_LIGHT1, GL_SPOT_EXPONENT, 10);  //See slides [03]:24-26
```

The default value for the diffuse colour of GL_LIGHT0 is white, and hence not explicitly set by the user. However, all remaining lights GL_LIGHT1… GL_LIGHT7 have a default value {0, 0, 0, 1}. The light source's colour must therefore be specified by the user as given above.

The function `draw_ss_body()` contains the code for modelling the surface of the spaceship's body, and two lights (represented by yellow spheres) on the surface. We will use GL_LIGHT1 as a spotlight at one of these positions. Include the following code at the beginning of this function, after the comment line "----Define spotlight position, direction here----".

```
float light1_pos[4] = { 21.21, 10, 21.21, 1 };
float light1_dir[3] = { 0.5, -0.7071, 0.5 };
glLightfv(GL_LIGHT1, GL_POSITION, light1_pos);
glLightfv(GL_LIGHT1, GL_SPOT_DIRECTION, light1_dir);
```

Note that the spotlight is directed towards the floor in +x, -y, +z directions. In a similar manner, attach the light source GL_LIGHT2 to the second light's position

on the spaceship model. The two spotlight positions and directions are symmetrical about the yz-plane. Fig.2 shows the output of the program with spotlights.

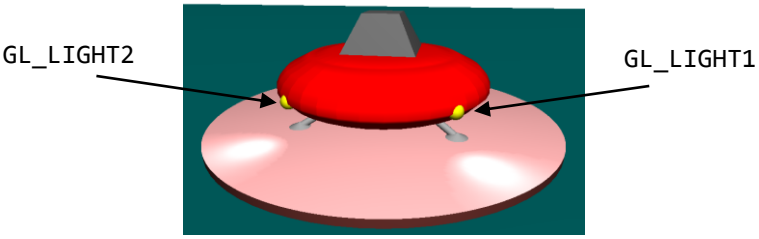


Fig. 2.

2. We will now make the top part ("dome") of the spaceship transparent to reveal the alien at the controls. Transparency is usually defined using an interpolative blending equation. Specify the blending equation (see Slide [03]:32) in the `initialise()` function. Blending must be enabled only when drawing the top part of the spaceship. Modify the code in the function `drawSpaceship()` as follows:

```
draw_ss_body();
draw_ss_legs();
draw_ss_alien();
glEnable(GL_BLEND);
   draw_ss_top();
glDisable(GL_BLEND);
```

Also, modify the colour value used in the `draw_ss_top()` function with the addition of an opacity factor of 0.6 (see Slide [03]:32). Fig. 3 shows the updated display of the spaceship scene.

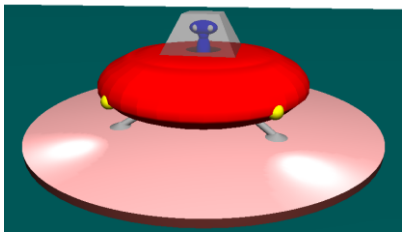


Fig. 3.

3. The program uses two textures "Rock.tga" and "Stars.tga". The header file `loadTGA.h` contains the code for loading texture images in tga format. We will map the first texture "Rock.tga" to the floor plane. In the `drawFloor()` function, enable texturing and bind the first texture for mapping:

```
glEnable(GL_TEXTURE_2D);
glBindTexture(GL_TEXTURE_2D, txId[0]);
```

Assign texture coordinates to the four vertices of the quad. (see Fig. 4). Remember to disable texturing after drawing the floor plane.

```
glDisable(GL_TEXTURE_2D);
```

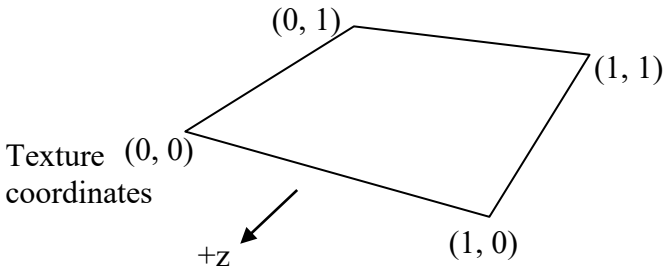


Fig. 4.

Note that the texture mapped image appears blurry due to the large size of the floor plane (Fig. 5a). Increase the number of repetitions of the texture to 10 in both 's' and 't' directions (see Slide[04]:23). The supplied texture can be seamlessly tiled along both directions. The output with the tiled texture is shown in Fig. 5b.

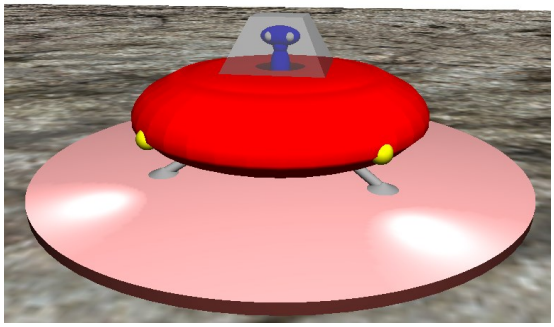


Fig. 5a

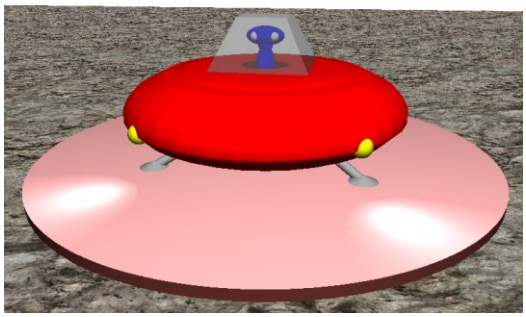


Fig. 5b

Take-home exercise: Define a set of vertical quads along the edges of the floor plane and map the second texture (Stars.tga) to the quads to create a night-sky background (Fig. 6). This texture can also be seamlessly tiled in both directions. The program may be further extended to create the animation of the spaceship lift off!

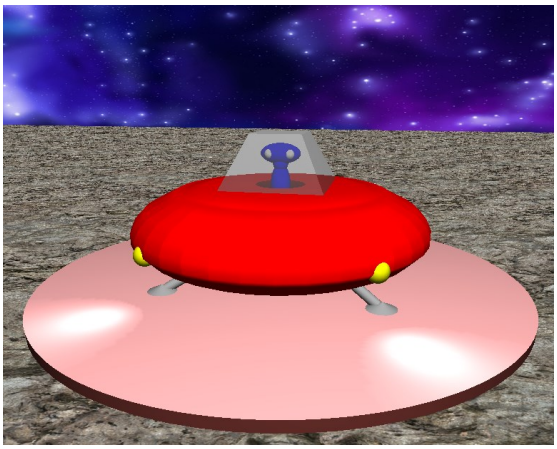
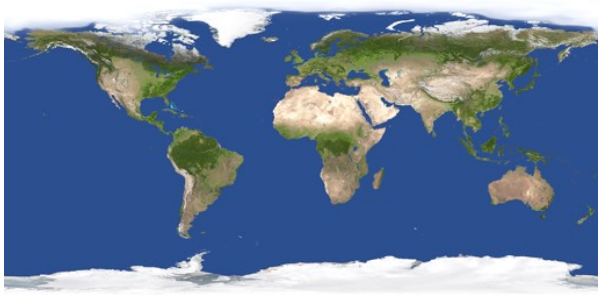


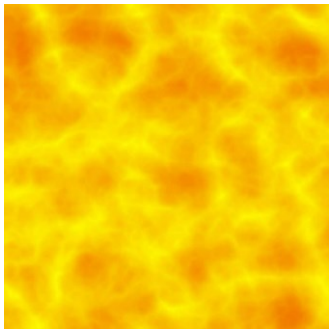
Fig. 6.

II. Earth.cpp

The program "Earth.cpp", uses the header file `loadBMP.h` to load two texture images in BMP format (Fig (7)).



Earth.bmp



Sun.bmp

Fig. 7.

The program gives an example of texturing spheres defined as quadric surfaces using implicitly (auto) generated texture coordinates. Quadric surfaces have a two-parameter representation, and the values of the parameters can be directly mapped to texture coordinates *s*, *t*. Texture coordinates for such surfaces can be automatically generated using the function call

```
gluQuadricTexture (q, GL_TRUE);
```

This mode is already enabled inside the function `initialise()`. The program generates the display of the Earth spinning about its axis at a fixed distance from the Sun (Fig. (8)). The sphere representing the Sun is positioned at the origin, and the Earth at a distance of 20 units along the x-axis. Please go through the transformations applied to the sphere models in the `display()` function.

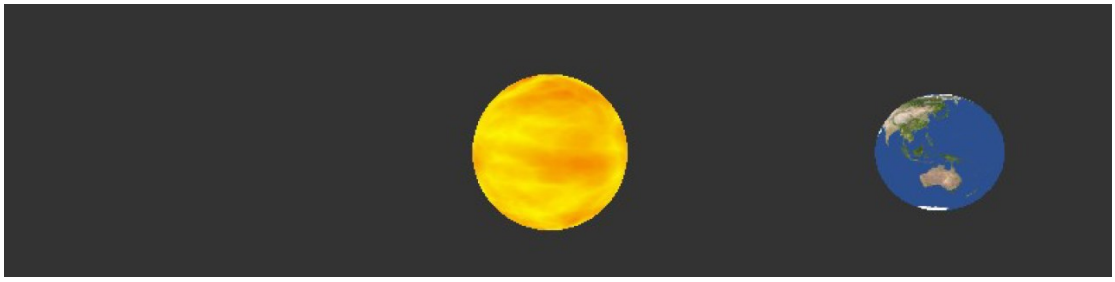


Fig. 8

A light source GL_LIGHT0 is already enabled inside the `initialise()` function. Specify its position as the origin (position of the Sun) in the `display()` function. Modify the environment parameter of the Earth texture to GL_MODULATE in the display() function, so that shadows are visible on Earth's textured surface (Fig (9)):

```
glTexEnvi(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);
```

Finally, revolve the Earth around the Sun (Fig. (9)).

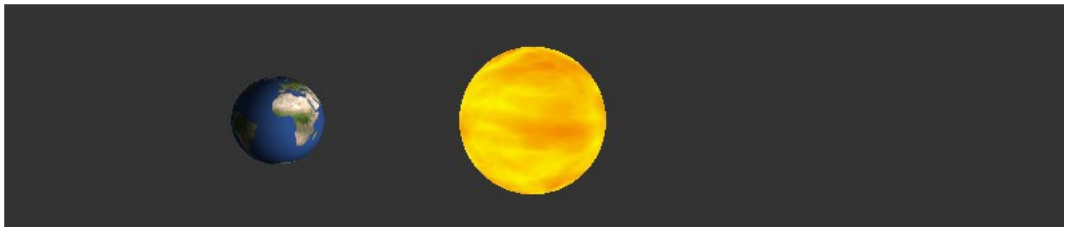


Fig. (9)

Refs:
[3]: COSC363 Lecture Slides, Lec03_Illumination.pdf
[4]: COSC363 Lecture Slides, Lec04_TextureMapping.pdf

III. Quiz-03

The quiz will remain open until **11:55pm, 24-Mar-2023**.

A quiz can be attempted only once. A question within a quiz may be attempted multiple times. However, a fraction of the marks (25%) will be deducted for each incorrect answer.