Lighting and Textures

Due date: Tuesday, March 8

The goal of this assignment is to implement a simple system for setting up lights in a 3d scene, as well as explore texturing surfaces.

1 What your program should do

There are three distinct parts to this assignment. One is setting up an environment with textured objects and a simple camera manipulation interface. Feel free to use any textures you want (you can convert any images to tga format using ImageMagick convert tool, GIMP or Photoshop). The second is a user interface for positioning lights in this environment, and the last one is an interface for choosing material colors for objects.

The current mode of the user interface is selected from a GLUT popup menu, which should contain the following items: camera, light positions, material diffuse, material specular.

Textured scene. Your scene should contain at least the following objects:

- A texture-mapped sphere, with the whole surface covered by the texture (do not use `glutSphere`, write your own OpenGL code generating polygons and texture coordinates for the sphere).
- A texture-mapped closed cylinder, with texture applied to the whole surface, including caps (again, write your own function for the cylinder).
- An object of complex shape (you can use glutTeapot), with a texture mapped to it using projective mapping; the texture used for this object should have an alpha channel.
- A texture-mapped ground plane.
- A cube surrounding the scene with texture map applied on 5 sides; suitable textures can be downloaded from the link posted on the web page.

Position the objects so that all are visible at once. Lighting should be enabled for objects and the ground plane, but not for the surrounding cube. Pressing M should toggle mipmaps for all textures, pressing F toggles the filters between linear and nearest neighbour.

Implement a simple interface for changing the view: clicking on a point in the scene moves it to the center of the image (we will discuss at the lecture how to do this using `gluLookAt` and `gluUnproject` functions), up and down arrows move the camera closer and further away.

Light positioning interface. It should be possible to place 4 point light sources in the scene. Pressing a digit i from 1 to 4 enables manipulation of a light; a small sphere should be displayed at the light source position for enabled lights. The following operations should be supported:

- space bar turns the current light on and off;
- clicking close to the sphere with the mouse and dragging moves the light parallel to the screen plane;
- Pressing shift and moving the mouse up and down moves the light along a line perp. to the screen. Horizontal component of the mouse motion has no effect in this case;
- Pressing up and down arrows changes the light intensity.

Initially, only the first light is turned on. Pressing S toggles lights on and off. Note that the direction of motion of a light in the world coordinates depends on the camera position and orientation: You need to infer the direction of motion from the camera position, look-at point, and the up vector.

Material specification interface. In the lower left corner of your image, display four horizontal bars, three with the gradient of a primary color (red, green or blue) from 255 to 0, and one to set shininess for specular reflection (no effect for diffuse mode). The bars can be drawn as three OpenGL rectangles, setting up a simple orthographics camera just for this purpose (the camera is switched to a perspective view when the scene is drawn). The gradient of colors is obtained by assigning suitable colors to the vertices of each polygon.

When one of the two material menu items is selected, clicking on a point inside one of the rectangles affects the corresponding parameter of the material property of the current object (e.g. if the current mode is material diffuse, clicking on the red bar changes the red component of the diffuse coefficient). The space bar switches between objects.

2 What to turn in

Just the source code and a working executable of your program; please make sure you include all textures your program uses.