Introduction to Computer Graphics

Assignment Five

“Let there be light”

Due date: November 17, 11:59 PM

Credit 10 points

This assignment may be done in pairs if you prefer.

IMPORTANT: All parts of all assignments should be understood for the exams, regardless of whether your partner did it.

In this assignment we will improve “pp-world” even more. The same rules regarding input and output and program exit from the previous assignments apply. In addition, do not forget to keep the same behavior regarding having a default parallelepiped if none are specified in the command line.

Grad students

Add a sphere command. Parameters are x,y,z, and radius. Implement an approximation of a sphere as a collection of polygons. You can use any collection of polygons that resembles a sphere. Try to use a simple strategy! The main requirement is that a sphere must be clearly distinguishable from a stretched/rotated box. You can give your spheres default colors.

Scaling of a sphere with x,X,y,Y,z,Z should scale the sphere uniformly. (Don’t create ellipsoids, unless you want to experiment with them, in which case, go for it, but let us know that you are doing so in the README. Notice that a proper job would either require outputting commands to construct ellipsoids, or perhaps an “ellipsoid” command. Maintaining ellipsoid capability for assignment 6 may provide some real entertainment!).

Rotating a sphere can be implemented as a NOP, but translation should work. The user should be able to add a default sphere with the menu. Picking should work on spheres.

End grad student part
In the following, you can use the naïve version of color introduced in class (diagonal model). Also, R,G, or B values that are greater than 255 should be set to 255. OpenGL probably does this for you. (This is called clipping—yes, another use of the word).

Implement the following command to add ambient light:

```plaintext
ambient [ r ] [ g ] [ b ]
```

(r,g,b) is the color of a perfect white diffuse reflector. Thus even if there are no lights, your world is not black. The default ambient light should be (50, 50, 50).

Implement a command to add a single point source. The command should be:

```plaintext
light [ x ] [ y ] [ z ] [ r ] [ g ] [ b ]
```

You can assume that it is distant enough so that the $1/r^2$ fall off can be ignored. If you want to allow more than one point source, that is fine (note this in the README). You will need to compute its direction relative to a polygon, which you can do simply based on its center computed by the average of the vertices. (Simply averaging vertices of a general polygon is not the best “center”, but for parallelograms and triangles it works fine).

Add a command:

```plaintext
specular [ v ] [ n ]
```

v is an integer from 0 to 100 which expresses a percentage of the light to be added via the Phong model, and n is the exponent in the Phong model. The command applies to every surface of the preceding object (box or sphere). Apply the specularity to the entire surface, based on the center of the surface computed by the average of the vertices.

Add a command:

```plaintext
mirror [ m ]
```

[m] is an integer from 0 to 100 which expresses a percentage of the light to be reflected. The command applies to every surface of the preceding box, or the preceding sphere. Add a bunch of white to the surface (proportional to [m]) to indicate to the user that this surface is a mirror. For example, if v is 50, add (100, 100, 100) to the color of the surface. **IMPORTANT:** The mirror will not act like a mirror until assignment 6.

Include menu items “specular strength”, “specular sharpness”, and “mirror”. Each of these will have four sub-menu settings or numbers going from “none” to “some” to “lots”, to “max”. The user’s choice will apply to the selected object. Further the keyboard entries v/V, n/N, m/M should decrease/increase the amount of specular percentage, specular sharpness, and mirror percentage of the selected object.

**Extra credit**

If you would like to improve on the program, be sure to explain what you did in the README file, and it will be considered for extra credit.
Deliverables

You must electronically submit a README containing any relevant information, but at a minimum, your name; an executable (called a5); and a src directory containing source files and a Makefile which can be used to build the executable.

The program must compile and run on one of the graphics machines (gr01, ..., gr08). Put in the README file the machine which you have verified this on.

The turnin name is cs433_hw5.