SUBJECT CODE: CS 6620

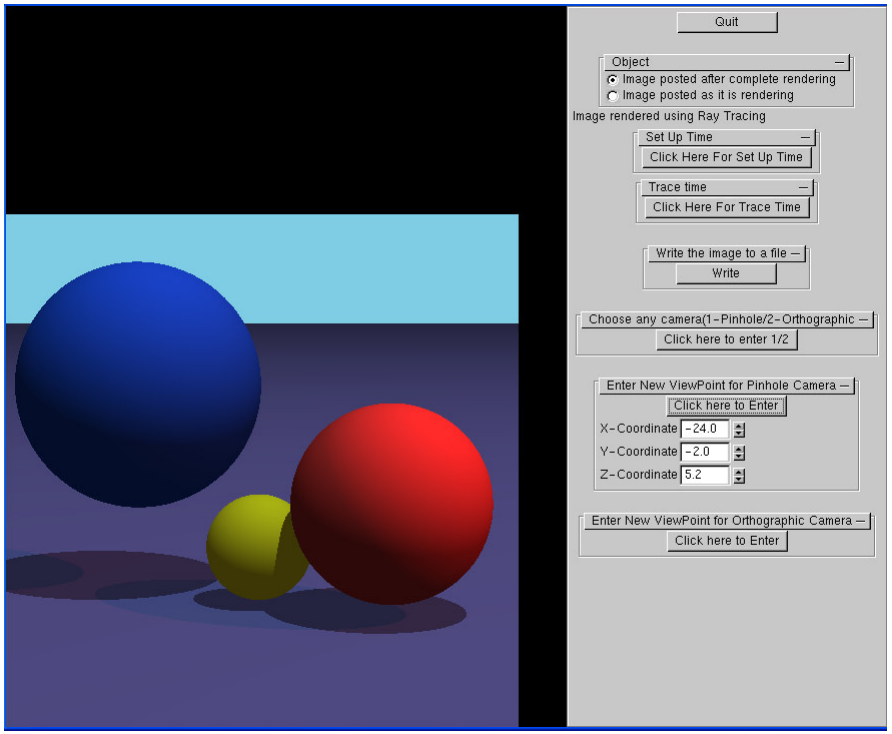
SUBJECT: ADVANCED COMPUTER GRAPHICS II

LEENA KORA
EMAIL:leenak@cs.utah.edu

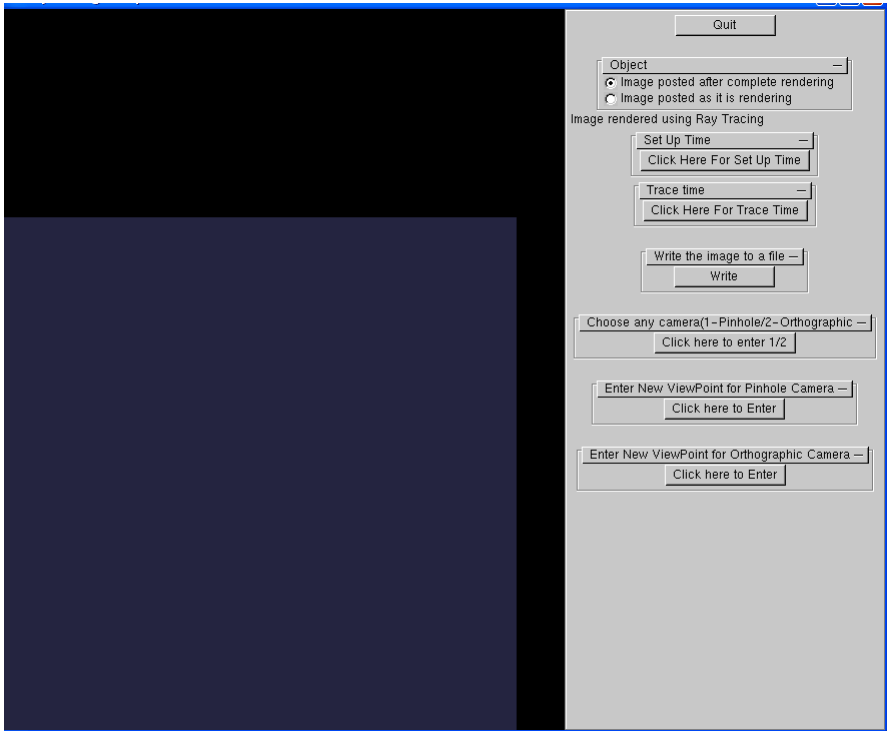
Unid: u0527667

PROGRAM 2

PART 1(Required Image1)



PART 1(Required Image2)

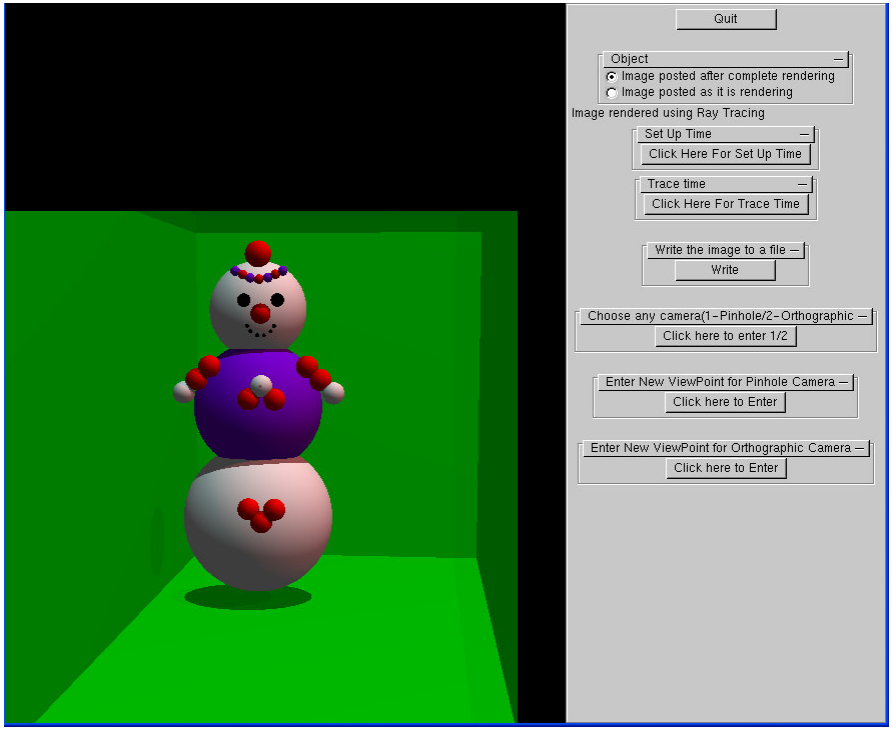


PART 2

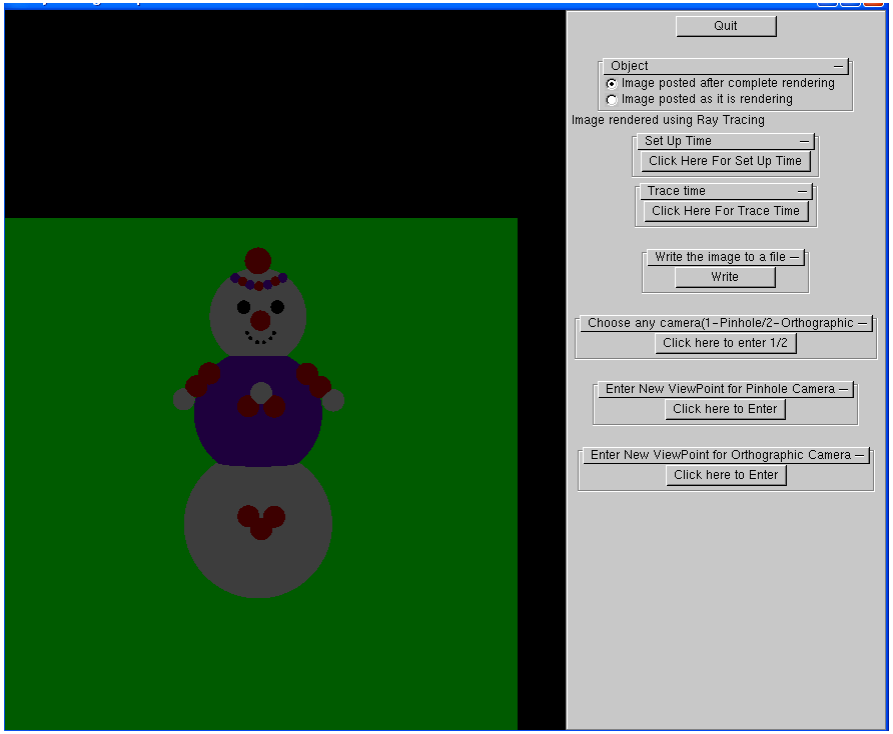
Link to my code: http://www.cs.utah.edu/~leenak/Code2.txt

PART 3(Creative Image(s))

- I first created all the spheres adjusting their positions to implement my "Snowman".
- Then decorated it with some colored balls.
- Then I created a "Cool Cooler" to my snowman to prevent it from melting by using 'Plane' objects.
- I got the below image when I measured the distance between the hit point on the object and the position of the light source by calculating the length of the vector, representing the difference between hit point and position of the light, after normalization of the vector.



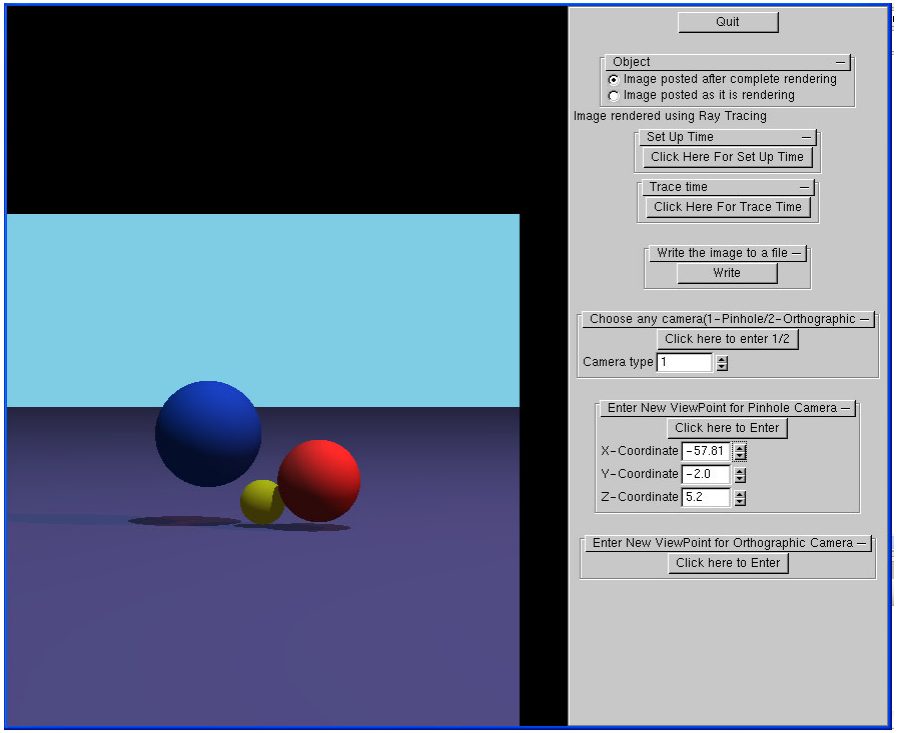
I got the below image when I measured the distance between the hit point on the object and the position of the light source by calculating the length of the vector, representing the difference between hit point and position of the light, before normalization of the vector.



PART 3(Description of my design choices)

- I created all the necessary base classes like camera, object, light, material and background.
- Then I created subclasses of the above items. I decided to follow the standard object oriented concepts such as creating data members as private. It requires more coding like creating methods to access the members of base class by subclasses.
- I created all the subclasses in separate header files.
- I decided to go for creating primitive class for items such as sphere and plane.
- I though we were going to use only Lambertian material properties to color the objects, so I added Ka and Kd(I used Kb notation) contants in material base class itself.
- I implemented the shadow rays in Lambertian material class's member funtion called color_mateiral().
- Then I created a scene class that had pointers to all type of the objects in scene.
- I extented my viewer using spinner and click button objects for making my viewer interactive. I did use gliu.h for the interface.

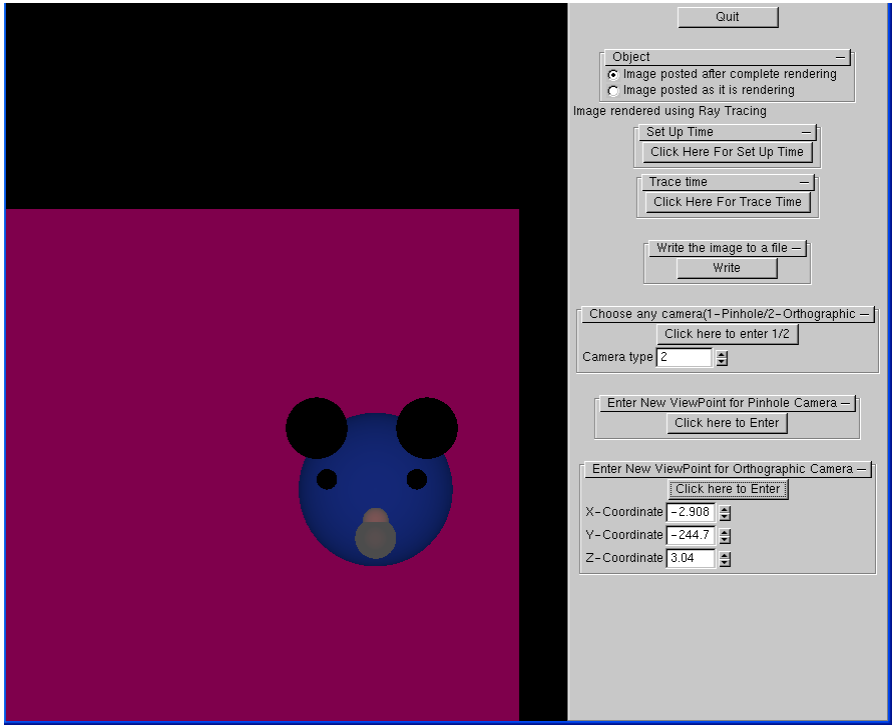
PART 3(Extra credit 1)



I have extended my integrated viewer by allowing the users to change the viewpoint interactively through entering the new point or clicking on the spinner object. I have also allowed the user to switch between the orthogonal camera and pinhole camera..

PART 3(Extra credit 3)

The below snapshot which has a "teddy face" in it shows my implemention of orthogonal camera.


Quit
Object
Image posted after complete rendering
Image posted as it is rendering
Image rendered using Ray Tracing
Set Up Time
Click Here For Set Up Time
Trace time
Click Here For Trace Time
Write the image to a file
Write
Choose any camera(1-Pinhole/2-Orthographic
Click here to enter 1/2
Camera type 2
Enter New ViewPoint for Pinhole Camera
Click here to Enter
Enter New ViewPoint for Orthographic Camera
Click here to Enter
X-Coordinate -2.908
Y-Coordinate -244.7
Z-Coordinate 3.04