

# CS535 Fall 2004 Programming Assignment 3

## Ray Tracing

Add code to assignment 2 such that when a button is pressed (or always if your code is fast enough for the graphics to flow smoothly), the scene is rendered with ray tracing instead. The rays from the camera should be set up to follow the same camera geometry as before, so that when the button is pressed, the scene will appear to be in the same place. The ray traced scene should contain four objects, the sphere, which should now be transparent and appear to be made of glass, the cube (if you find it easier, I recommend you make the cube another sphere), which should be perfectly reflective and appear to be made of mirrors, the ground plane (where the flat mesh is otherwise) and the plane at infinity. The ground plane and the plane at infinity should contain some interesting pattern. The simplest would be a colorful chessboard pattern. More interesting would be the Mandelbrot set.

Suggested steps:

1. Write code that takes a start point  $X$  on a ray and a point  $Y$ , which fully defines the ray as all the points  $X+sY$  for some scalar value  $s$  and returns the smallest positive  $s$  and the corresponding point  $X+sY$  for intersection with a plane.
2. Write a similar routine for a sphere. Object orientation may help here.
3. Write a rendering for-loop that uses the above routines to trace a ray from each pixel (Use for example  $X=COP$  and  $Y$  defined by starting at the image point and going backwards through the viewing transformations), intersect against all four objects and find the smallest positive  $s$ . When the ray hits one of the planes, use a chessboard pattern to assign the color of the pixel.
4. Write a routine that computes the reflection direction vector using the incoming ray direction  $(Y-X)/|Y-X|$  and the surface normal. Use it to change the ray definition when the ray hits the cube and sphere (set for example  $X$ =intersection point,  $Y=X$ +reflection\_direction) and recursively compute again.

Optional:

5. Write a routine that computes the refraction direction vector using incoming ray direction, the surface normal and the refraction index of the glass (note the difference between entering and leaving the glass). Use it when the ray hits the sphere (enter and leave).
6. Add more interesting patterns to the ground plane and the plane at infinity. For example the Mandelbrot set (see book to figure out how to precompute such an image and store it in memory, even cooler would be a zooming capability), or a photograph.
7. Make the objects bounce around in the scene.