Collin Barnwell/crb331

Project B: BLIMP WORLD!

Description of Goals

The goal of this project for me was, somewhat unsurprisingly, to learn how to make a ray tracer. The going was rough at the beginning as a struggled through the first steps, but once I got my first shape to appear on screen, I starting cruising. I now feel like I understand how all ray tracing technologies are implemented, even if I haven't implemented them myself yet.

I definitely think it was useful to implement Phong shading again since last quarter, as it was much easier and I now think I understand it very well.

I knew this project would be hairier than anything we had done in the past, so I made it a major goal to stay organized this time (I believe I succeeded). Also, (I am somewhat embarrassed), this was the first of the graphics projects that I did in an object-oriented manner. My code is much cleaner and, as a result, even though this project is much more complex than any of my other projects, it is not as many lines and thus a little bit more maintainable. It was good to finally understand how I can write graphics programs in JavaScript in this way.

User Guide/Instructions

- The Space Bar will recalculate the picture.
- The arrow keys move the camera parallel to the ground plane.
- 'W' and 'S' move the camera up and down.
- 'A' and 'D' rotate the camera direction about the vertical axis.
- 'Q' and 'E' rotate the camera about the horizontal axis.
- 'Z' toggles antialiasing; by default it is OFF.
- Play with the fields below the canvas to adjust the light sources or turn them on/off. Make sure you click somewhere else after clicking one of the checkboxes or editing the fields (to deselecct them), then press "SPACE".
- You can also switch scenes with the radio buttons below the canvas. Make sure you click somewhere else after clicking the radio button (to deselect it), then hit "SPACE".
- EVERY TIME YOU MAKE A CHANGE, YOU MUST PRESS "SPACE" FOR THE PICTURE TO GET RETRACED.
- Hit 'F1' to hide or display these instructions (or 'H', or '1').
- Enjoy!

Code Guide

I used classes for everything that I could (Rays, Cameras, Scenes, Shapes, Cylinders, Spheres, Cubes , Cones, Planes, etc, which helped the organization a lot. I made most of the tracing functions functions of the Camera class, and checked each shape with its own personalized 'findHit' function. All of this was in the camRays.js file. The week1.js file was more or less just a wrapper that called all of those functions for each pixel, while the controls.js dealt with all the external events, controls, and

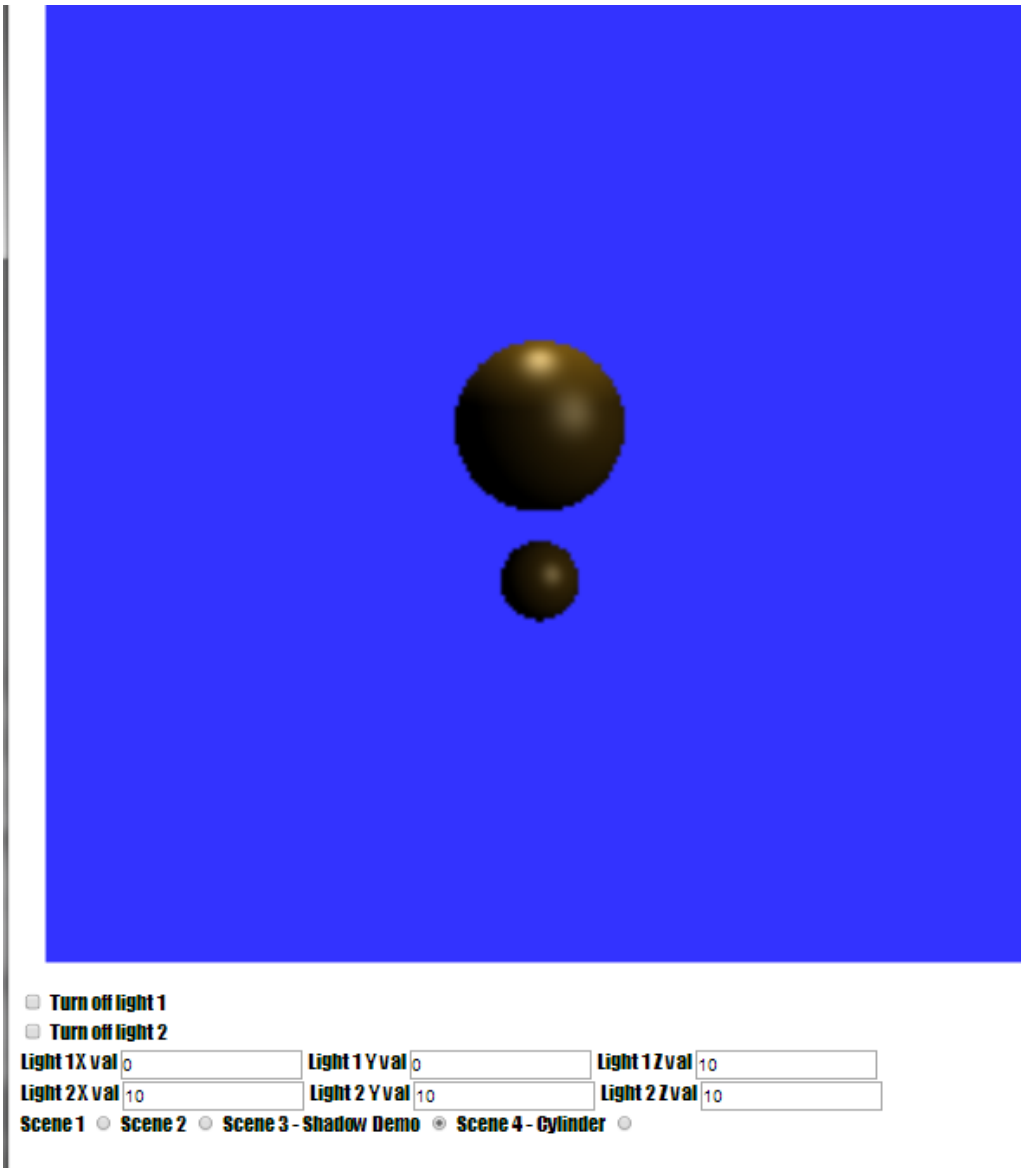
settings. I also had some global variables for antialiasing, camera position, look at point, up vector, and light position; all of which were directly affected by user input. In the lib directory, I included the glMatrix and jQuery libraries, as well as the libraries provided by the book last quarter.

Also, this site was really helpful for tracing other shapes and getting a feel for solving implicitly: https://www.cl.cam.ac.uk/teaching/1999/AGraphHCI/SMAG/node2.html#eqn:rectray

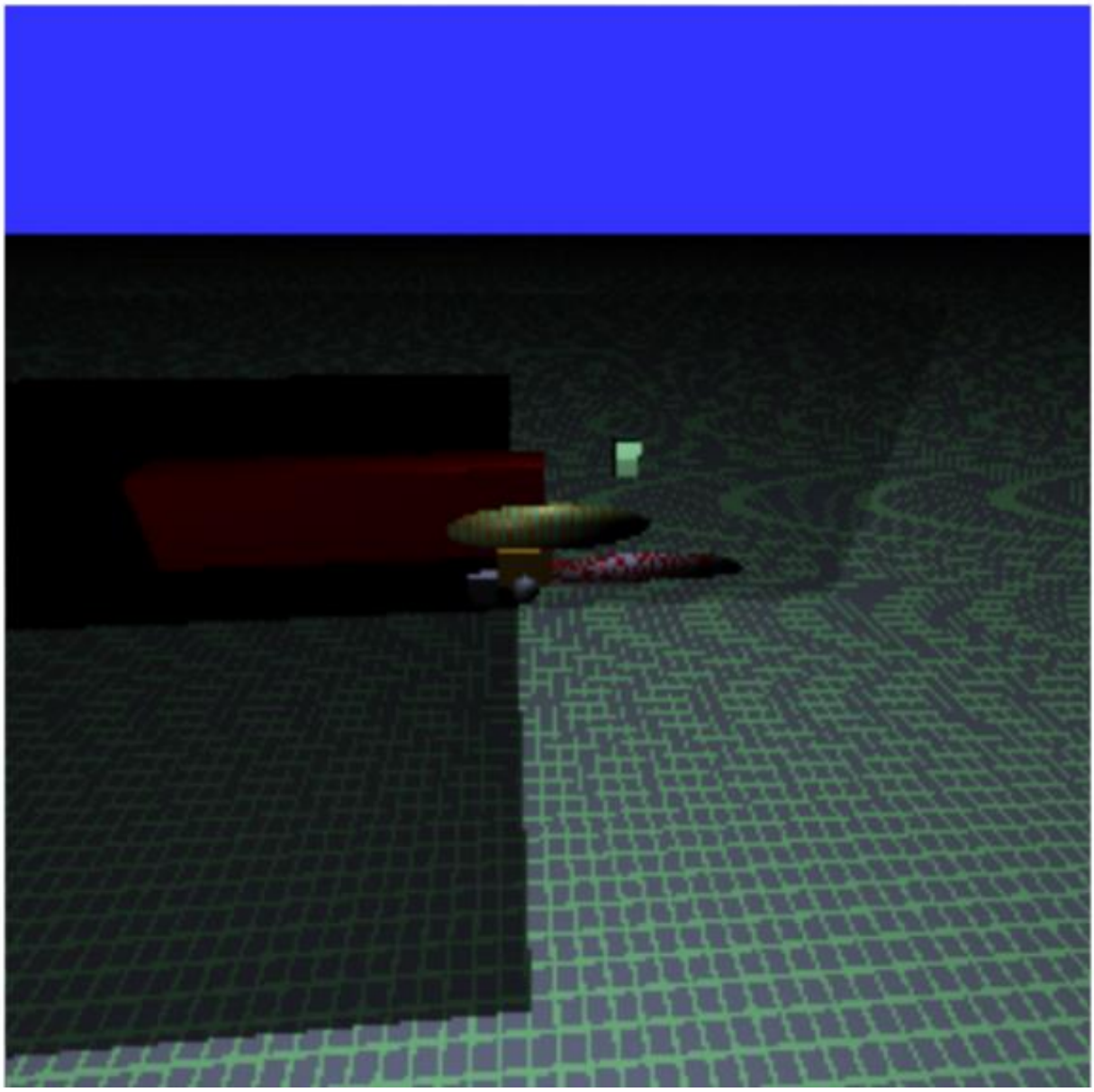
Results

Shadows:

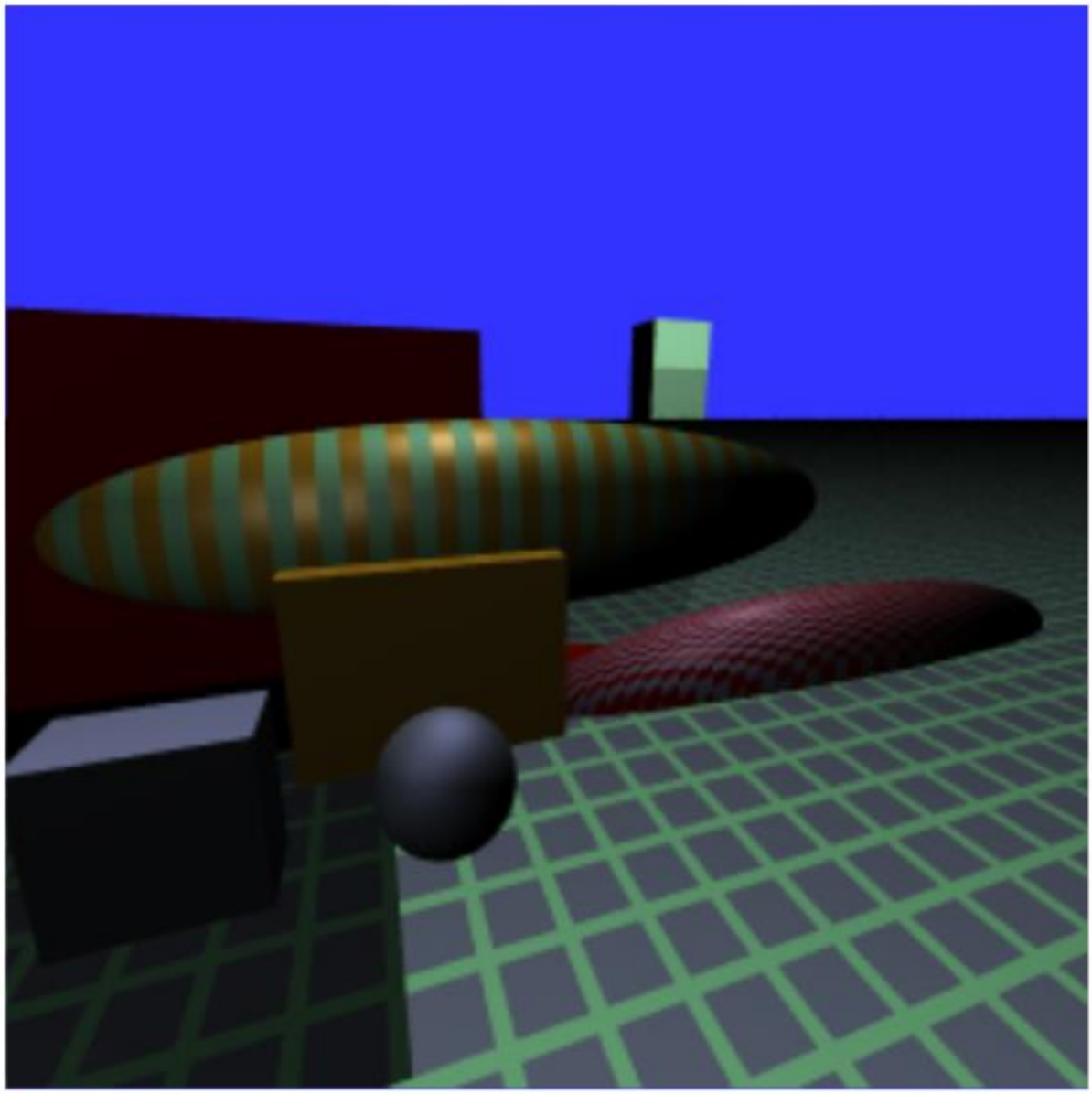
In Scene 3 (Shadow demo) you can see the specular highlights and diffuse lighting from the light above disappear, because the smaller sphere is underneath the larger one. However, ambient light and diffuse/specular light from the other light can still be seen on the smaller sphere.



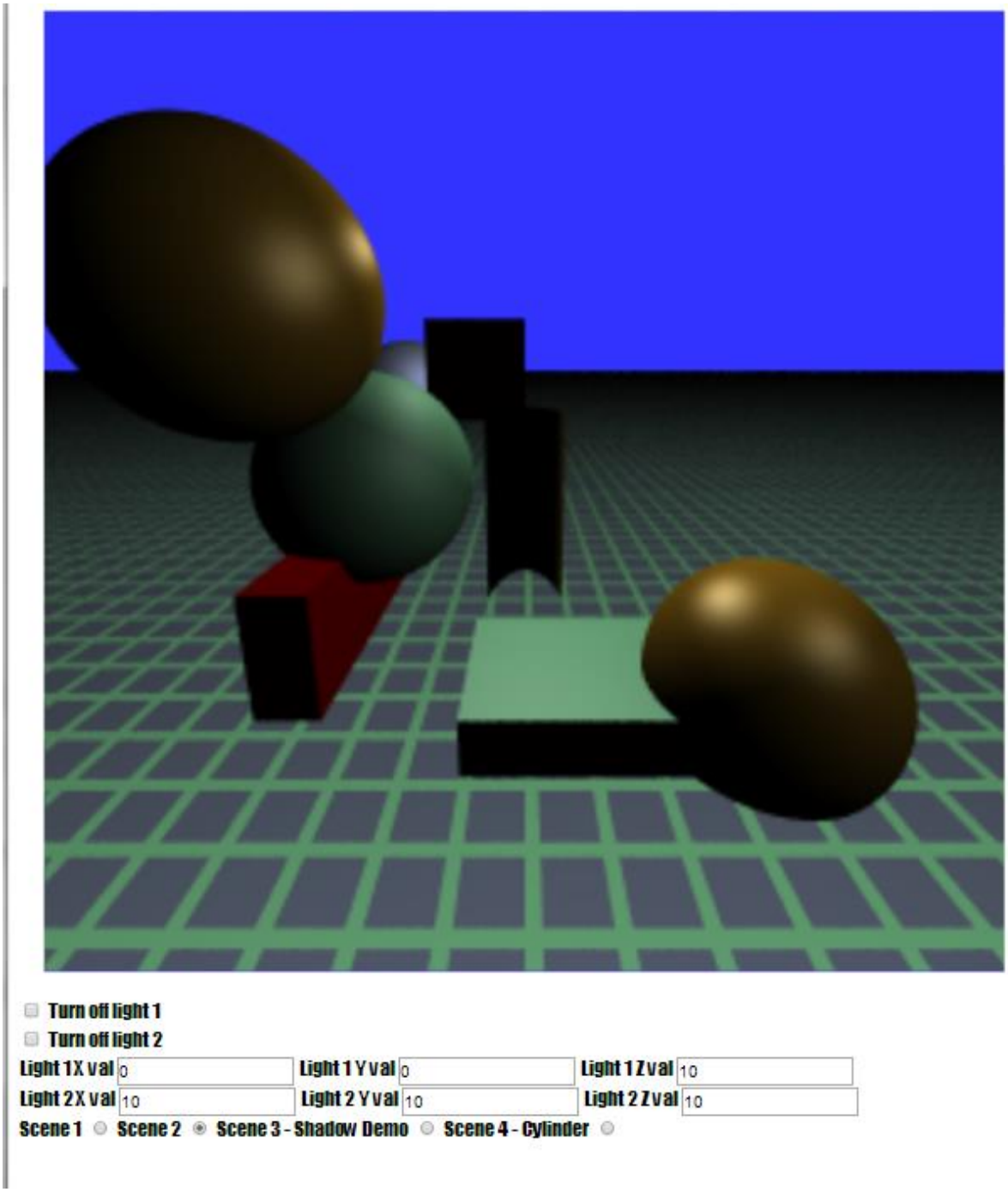
Large overlapping shadows cast on opposite sides of the block wall:



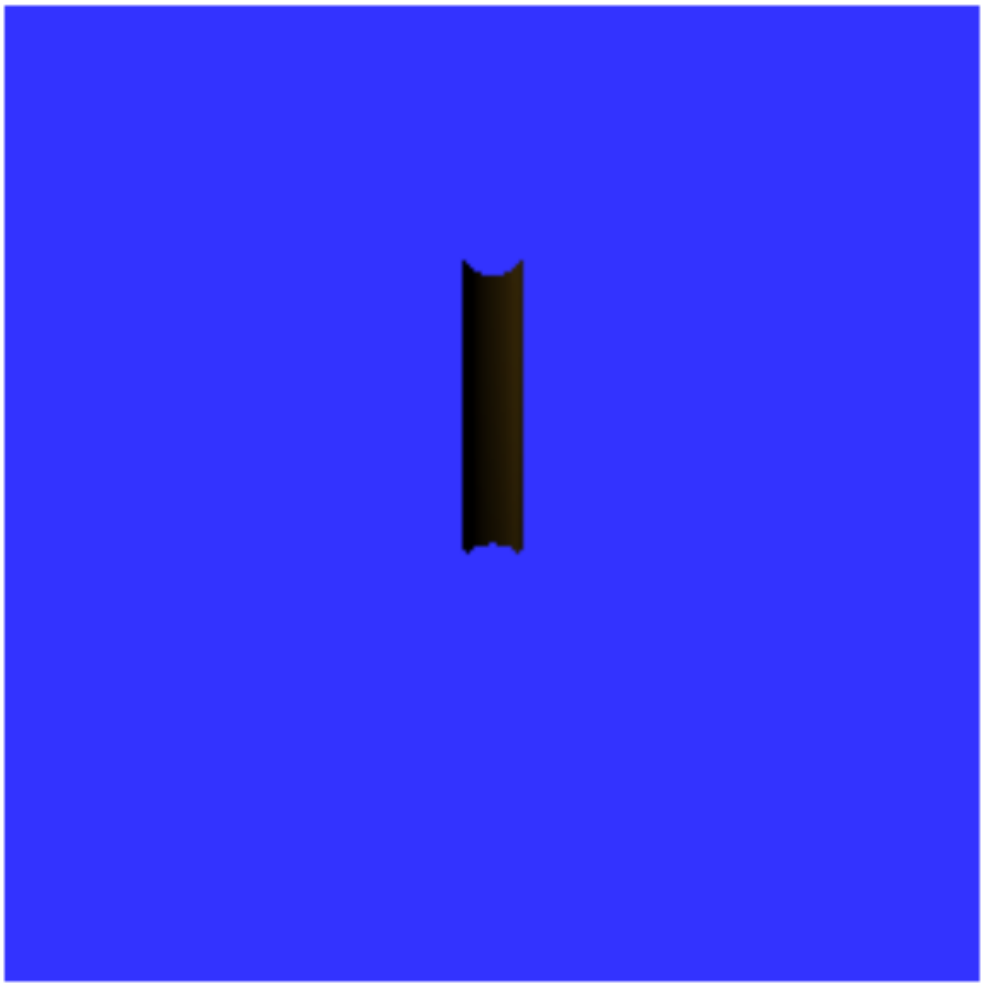
2 Procedural materials, squeezed/stretched spheres: there is a checkerboard pattern on the lower right one, and a striped pattern on the upper left one. This is also the only picture so far to be shown with antialiasing ‘on’.



Scene # 2: 4 more spheres and 4 more boxes. There is also a dark-looking cylinder in the center of the picture; but don't worry if you can't see it, because it has its own scene!

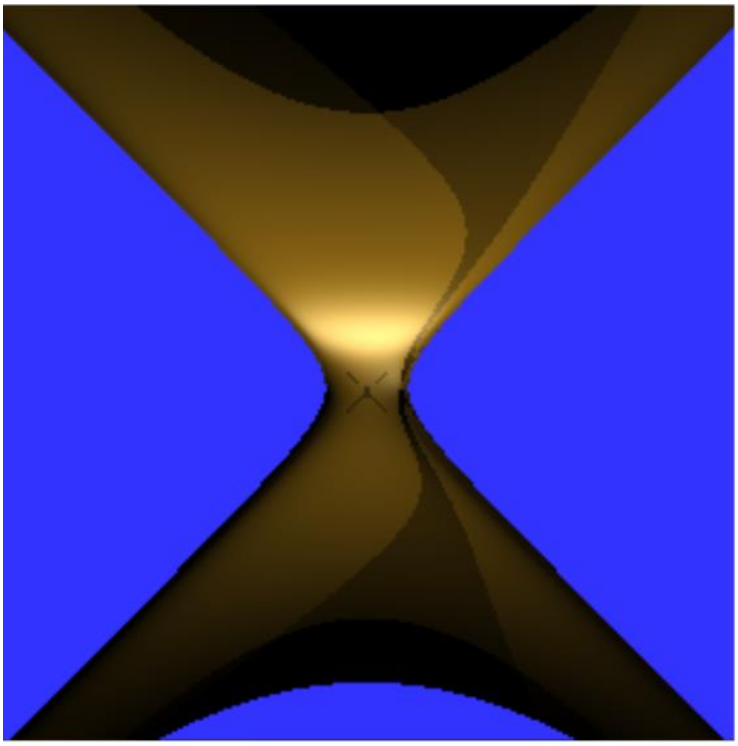


The much-anticipated cylinder scene:

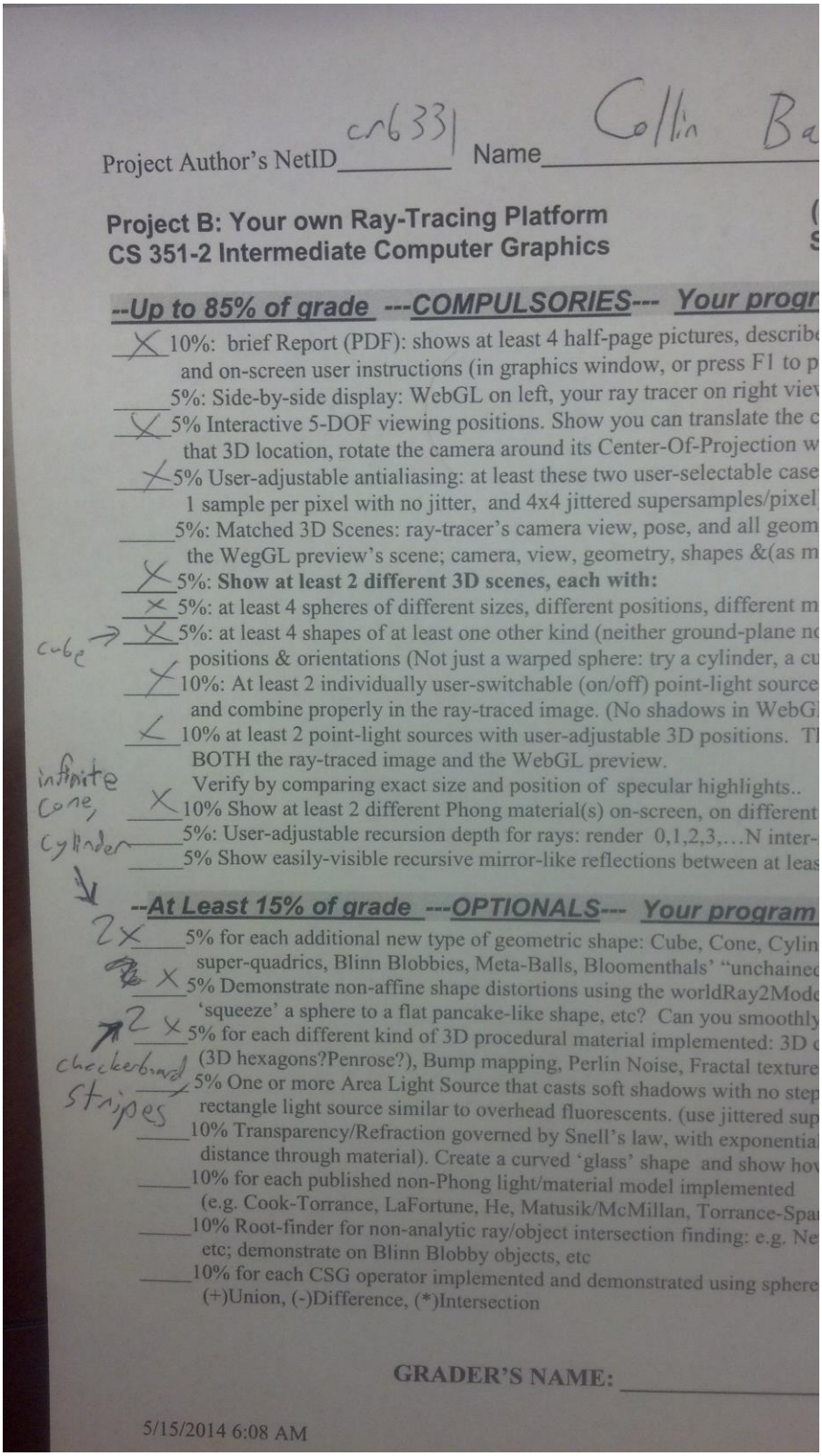


Turn off light 1
Turn off light 2
ht 1X val 0 Light 1 Y val 0 Light 1 Z val 10
ht 2X val 10 Light 2 Y val 10 Light 2 Z val 10
ene 1 Scene 2 Scene 3 - Shadow Demo Scene 4 - Cylinder

Also: An infinite cone!



Chosen options (90% total)



Project Author's NetID cr6331 Name Collin Ba

Project B: Your own Ray-Tracing Platform
CS 351-2 Intermediate Computer Graphics

--Up to 85% of grade ---COMPULSORIES--- Your progr

X 10%: brief Report (PDF): shows at least 4 half-page pictures, describe and on-screen user instructions (in graphics window, or press F1 to p

____ 5%: Side-by-side display: WebGL on left, your ray tracer on right viev

X 5% Interactive 5-DOF viewing positions. Show you can translate the c that 3D location, rotate the camera around its Center-Of-Projection w

X 5% User-adjustable antialiasing: at least these two user-selectable case 1 sample per pixel with no jitter, and 4x4 jittered supersamples/pixel

____ 5%: Matched 3D Scenes: ray-tracer's camera view, pose, and all geom the WegGL preview's scene; camera, view, geometry, shapes &(as m

X 5%: **Show at least 2 different 3D scenes, each with:**

X 5%: at least 4 spheres of different sizes, different positions, different m

cube → X 5%: at least 4 shapes of at least one other kind (neither ground-plane nc positions & orientations (Not just a warped sphere: try a cylinder, a cu

X 10%: At least 2 individually user-switchable (on/off) point-light source and combine properly in the ray-traced image. (No shadows in WebG

X 10% at least 2 point-light sources with user-adjustable 3D positions. TI BOTH the ray-traced image and the WebGL preview.

infinite cone, cylinder Verify by comparing exact size and position of specular highlights..

X 10% Show at least 2 different Phong material(s) on-screen, on different

____ 5%: User-adjustable recursion depth for rays: render 0,1,2,3,…N inter-

____ 5% Show easily-visible recursive mirror-like reflections between at leas

--At Least 15% of grade ---OPTIONALS--- Your program

2X ____ 5% for each additional new type of geometric shape: Cube, Cone, Cylin super-quadrics, Blinn Blobbies, Meta-Balls, Bloomenthals' "unchained

X 5% Demonstrate non-affine shape distortions using the worldRay2Mode 'squeeze' a sphere to a flat pancake-like shape, etc? Can you smoothly

2 X 5% for each different kind of 3D procedural material implemented: 3D c (3D hexagons?Penrose?), Bump mapping, Perlin Noise, Fractal texture

checkerboard stripes 5% One or more Area Light Source that casts soft shadows with no step rectangle light source similar to overhead fluorescents. (use jittered sup

____ 10% Transparency/Refraction governed by Snell's law, with exponentia distance through material). Create a curved 'glass' shape and show hov

____ 10% for each published non-Phong light/material model implemented (e.g. Cook-Torrance, LaFortune, He, Matusik/McMillan, Torrance-Spa

____ 10% Root-finder for non-analytic ray/object intersection finding: e.g. Ne etc; demonstrate on Blinn Blobby objects, etc

____ 10% for each CSG operator implemented and demonstrated using sphere (+)Union, (-)Difference, (*)Intersection

GRADER'S NAME: _____

5/15/2014 6:08 AM