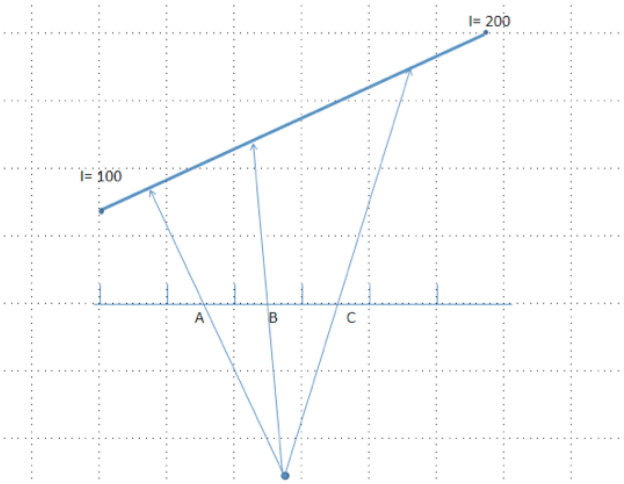
CPSC 478/578 Computer Graphics
Fall 2017
Assignment #3
Assigned: Wednesday, September 20, 2017
Due: Wednesday, October 4, 2017, 11:59pm

Note that the requirements for each question may vary depending on whether you are registered for 478 or for 578. The area addressed in this assignment is computing visibility at each pixel in an image.

Turn-in Procedure

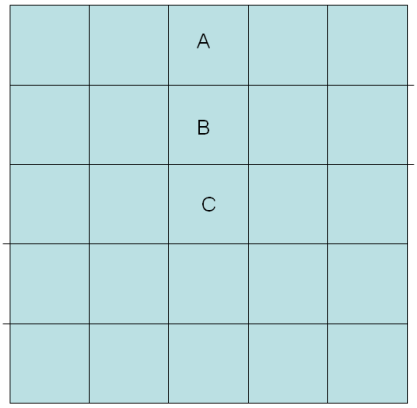
You should submit your work as a zip file using the classesv2 server. Please name your file as LastNameFirstName-Assignment3.zip
When your file is unzipped there should be subdirectories for each question named q1, q2, etc. Name your files as directed in each question. In each directory you should have:
1. The HTML and Javascript programs you have written, or pdf's of your written response (either typed directly or scanned in). For code, you should use files in the form of the samples given, rather than producing files from scratch. This will help us follow your code.
2. If the question asks you to write code to make images, provide sample images created by your program. You can save these by clicking and saving results in your browser, or by taking a screenshot.
3. A readme.{txt, doc} that lists the input used to create the images you include. You should also list the operating system (e.g. Linux, Windows 7, 8.1, 10, Mac OS 10.4.4) and browser (e.g. Firefox 40.0.2, Safari, IExplorer, Edge) that you used. If you programs fail on the machines used for grading, you may be asked to bring in your system to demonstrate that the files you submitted functioned in the environment you worked in.

Question 1 (**478 and 578**). **(20 pts)** No coding required, scan or type in your response and provide as a pdf.
a.) Consider the rendering shown below. A line segment has intensities of 100 and 200 at its two ends, and the intensity on the segment varies linearly between the end values. What is the error in estimating the intensity of pixel B by interpolating between the intensities computed for pixels A and C?

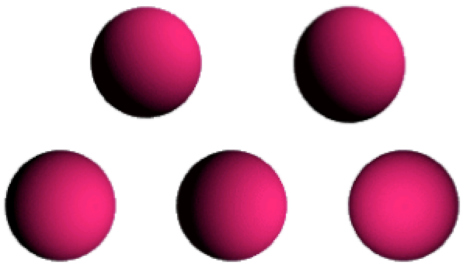


b.) Consider a virtual camera in a world coordinate system with the eye point at

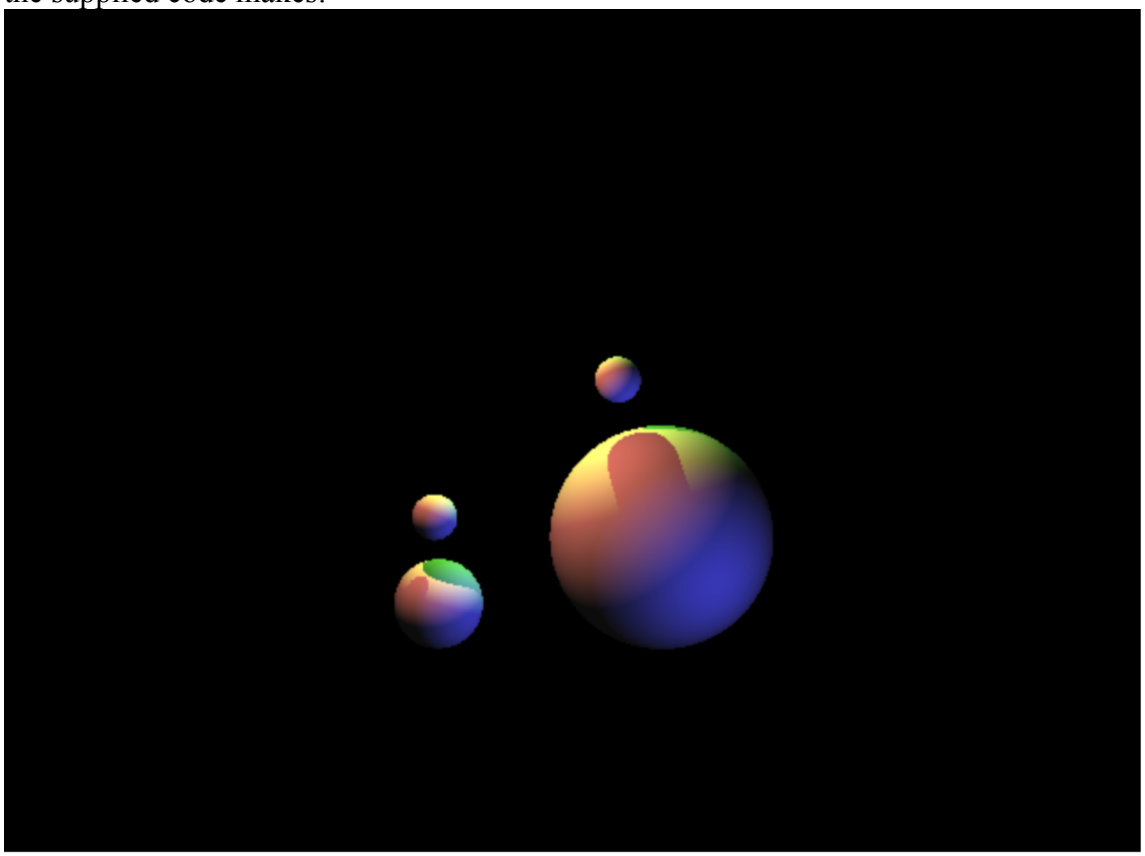
(1,10,2) and looking in the direction towards point (2, 10, -4), with an up direction given by the unit vector (0,1,0). The virtual camera has a view frustum that is 45 degrees both horizontally and vertically. A 5 by 5 pixel image is to be computed.
i. Give an equation in terms of a parameter t for all points P that lie on the ray from the camera origin through the center of the center pixel (pixel C in the diagram below) of the image.
ii. Find the angle between the rays that go through the centers of pixel A and pixel B. Is the angle between the rays that go through the centers of pixel B and pixel C the same, larger or smaller. Justify your answer.



Question 2 (478 and 578) (25 pts) Create a Javascript program that generates a 512 by 512 image. The image will be an **orthographic projection** of a shaded sphere. The eye will be at the point (0,0, infinity), and the image coordinates are from (-1,-1) on the lower left to (1, 1) on the upper right. The program should take as input a sphere diameter. A sphere diameter of one should occupy the full image (i.e. should have a diameter of 512 pixels.) The center pixel should be the point on the sphere with the normal pointing at the viewer. The program should also take as input a RGB values that give the maximum color value of a pixel on the sphere, and a vector giving the direction of the incident directional light. The program should produce an image of the sphere illuminated. Turn in two sample images, with different colors and light directions Do not use any WebGL calls for question. Provide two sample images from your program, with different sizes, colors and light direction.
Here are some examples of spheres lit from different directions:



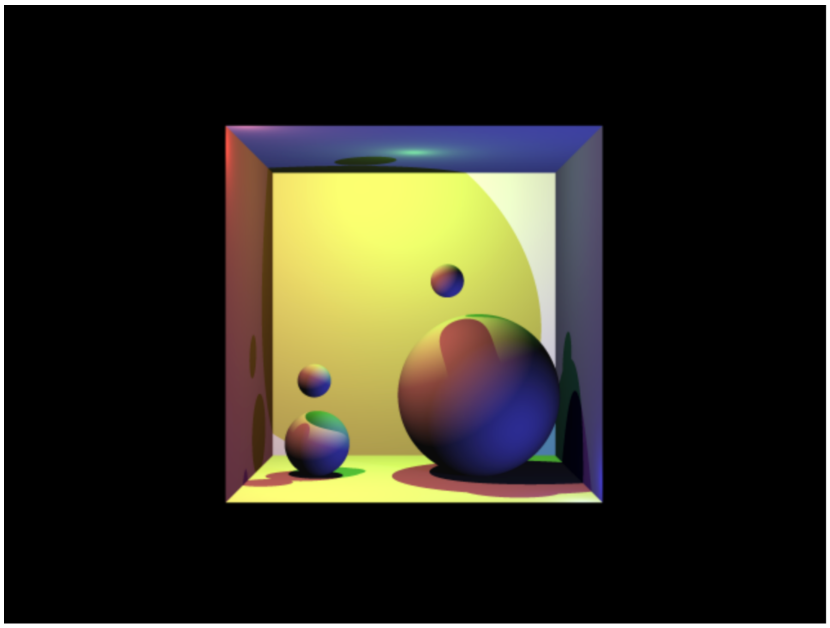
Question 3 Casting rays – antialiasing **(20 pts)**
Use the supplied code (renderme.html and files in folder js) based on the literate raytracer pages as a starting point for this question. Do not use any WebGL calls. This is the image the supplied code makes:



(478 and 578) Modify the code to do anti-aliasing to reduce the "jaggies" by computing the average of the colors computed for 4 rays cast through 4 different points in each pixel. Submit a sample image produced by your code.

Question 4 Casting rays -- Object types**. (25 pts)**
Do not use any WebGL calls for this question.

(478 and 578) Fill in the missing code tri.js and adjust the rest of the code so that there is an additional object type "triangle" with attributes point1, point2 and point3 to define where it is. See the example triangle definitions in scene-orig.js. When the triangles in the original scene are rendered you will get this:



You will need to write code to find the intersection point and normal for this object type.
ALSO: Submit an additional image rendered of another scene with visible triangles defined.

(578) Define a new cone object type. The cone attributes should be the center, the height and radius of the cone object. You may assume that the main axis of the cone is parallel to the y axis. Submit an image rendered with at least one cone visible in the image.

Question 5 (10 pts) – MOCAP
(478 and 578) Describe a project that you might use the motion capture studio at the Yale CCAM to collect data for. You won't have to implement this project, you just need to write a paragraph on how the data would be used in a game, film or other project.