Last Name: _____ First Name: _____ Cornell NetID: _____

CS 4620 Midterm, October 21, 2014
SOLUTION

1. [20 points] **Ray Intersection**.

   Consider a scene with a sphere and a triangle. The sphere's center is at $(0, 0, 0)$ and its radius is 1. UV coordinates of the sphere are the same as in assignment 1.[1]

   The triangle has vertices $(2, 0, 0)$, $(0, 2, 0)$, and $(0, 0, 2)$. The camera (ray origin) is at $(\sqrt{3}, \sqrt{3}, \sqrt{3})$ and the ray direction is $(-\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}})$.

   Please answer the following questions. The answers can be unreduced—they can be in terms of square roots, fractions, and (inverse) trigonometric functions.

   (a) Does the ray intersect the sphere? If yes, what data (position, normal, and t) would be put in the intersection record? If not, why?

   (b) Does the ray intersect the triangle? If yes, what data would be put in the intersection record? If not, why?

   (c) Which intersection record, if any, would be used in a subsequent shading computation? Why?

[1] The North pole of the earth is at $(0, 1, 0)$. The South pole is at $(0, -1, 0)$. And points on the Greenwich meridian have coordinates $(0, y, z)$ with $z > 0$. The texture coordinate u depends only on longitude, with u=0 at longitude 180° West and u=1 at 180° East. v depends only on latitude, with v=0 at the South pole and v=1 at the North pole.

Last Name: _____ First Name: _____ Cornell NetID: _____

Solution: Q1. Yes. Position: $(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})$. Normal: $(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})$. t: 2.
(+3 for position, +3 for normal, +2 for t)

Q2. Yes. Position: $(\frac{2}{3}, \frac{2}{3}, \frac{2}{3})$. Normal: $(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})$. t: $3 - \frac{2}{\sqrt{3}}$.
(+3 for position, +3 for normal, +2 for t)

Q3. The hit on the triangle, because $t_{triangle} < t_{sphere}$.
(+2 for correct answer, +2 for decent explanation.)

Last Name: _____ First Name: _____ Cornell NetID: _____

2. [16 points] **Transformation Matrices**. Consider the following matrices representing transformations on 3D points in homogeneous coordinates.

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 2 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad D = \begin{bmatrix} \frac{\sqrt{3}}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 0 \\ \frac{-1}{2} & 0 & \frac{\sqrt{3}}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(a) Describe the transformation of each one of the matrices above (e.g. X represents a rotation by Y degrees, etc.), using the categories: scale, translation, rotation, projection.

Solution: A: Scale along y by a factor of $\frac{1}{2}$. Or scale by $(1, \frac{1}{2}, 1)$
B: Scale by $\frac{1}{2}$ and translation by $(t_x, t_y, t_z) = (1.5, 1, 0.5)$
C: Projection to $x - z$ plane
D: Rotation 30° CCW around the y-axis
(Each +1 for category, +1 for details.)

(b) What matrix would you use to rotate 30° **clockwise around the y-axis**.

Solution: $E = D^-1 = \begin{bmatrix} \frac{\sqrt{3}}{2} & 0 & \frac{-1}{2} & 0 \\ 0 & 1 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{\sqrt{3}}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ (+1 for choosing D, +1 inverting it. -0.5 for truly minor calculation errors. Just "D" is fine if they lost a point for writing "CW" in the previous part.)

(c) Suppose you wanted to *undo* the effect of BA. Write a single matrix that can achieve such a result.

Last Name: _____ First Name: _____ Cornell NetID: _____

Solution: $(BA)^{-1} = A^{-1} * B^{-1} = F$ where $F = \begin{bmatrix} 1 & 0 & 0 & -1.5 \\ 0 & 2 & 0 & -2 \\ 0 & 0 & 1 & -0.5 \\ 0 & 0 & 0 & 0.5 \end{bmatrix}$ (+2 for wanting to invert BA; +1 for attempting a valid calculation; +1 for numbers. Also fine to put in reduced form with 0.5 in the corner.)

(d) Is it possible to write a matrix to *undo* C? (explain, in your own words, why that is the case)

Solution: No. We projected to a subspace and lost the information in one of the dimensions (y-axis) which cannot be recovered. Matrix C is not invertible.
(+1 for "no"; +1 for valid explanation; just "not invertible" is sufficient.)

Question:	1	2	3	4	5	Total
Points:	20	16	24	20	20	100
Score:						

3. [24 points] **Meshes**

Suppose we are given the geometry (an octahedron) shown in Figure 1, and the following position buffer containing floating point data:

{0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, -1.0, 0.0, 0.0, 0.0, 0.0, -1.0, 0.0, -1.0, 0.0}

Here are three possible index buffers (containing integer data) that correspond to the octahedron. For each, please state whether the specification will give the correct output. If not, please explain what went wrong and fix it by crossing out and rewriting some of the numbers.

1. {0, 1, 2, 0, 2, 3, 0, 3, 4, 0, 4, 1, 2, 0, 1, 3, 0, 2, 4, 0, 3, 1, 0, 4}

2. {2, 0, 1, 2, 3, 0, 2, 5, 3, 2, 1, 5, 4, 0, 1, 4, 1, 5, 4, 5, 3, 4, 3, 0}

3. {1, 2, 0, 1, 0, 4, 1, 4, 5, 1, 5, 2, 3, 4, 0, 3, 0, 2, 3, 2, 5, 3, 5, 4}

Now suppose we take this same mesh and duplicate a few of the vertices. Our position buffer now becomes:

{0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, -1.0, 0.0, 0.0, 0.0, 0.0, -1.0, 0.0, -1.0, 0.0, 0.0, -1.0, 0.0, 0.0, -1.0, 0.0, 0.0, -1.0, 0.0}

And our index buffer is:

{0, 4, 6, 1, 6, 7, 2, 7, 8, 3, 8, 5, 9, 6, 4, 10, 7, 6, 11, 8, 7, 12, 5, 8}

Let's add texture coordinates to our new mesh. Below are two different texture coordinate buffers. For each, please match it with the corresponding mesh layout in UV-space (Figure 2), as well as the corresponding final renderings (Figure 3).

4. {0.5, 1.0, 0.5, 1.0, 0.5, 1.0, 0.5, 1.0, 0.0, 0.5, 1.0, 0.5, 0.25, 0.5, 0.5, 0.5, 0.75, 0.5, 0.5, 0.0, 0.5, 0.0, 0.5, 0.0, 0.5, 0.0}

5. {0.25, 1.0, 0.25, 1.0, 0.75, 1.0, 0.75, 1.0, 0.0, 0.5, 1.0, 0.5, 0.25, 0.5, 0.5, 0.5, 0.75, 0.5, 0.0, 0.0, 0.5, 0.0, 0.5, 0.0, 1.0, 0.0}

The answer for each part on this page consists of a number (to describe which UV layout corresponds) and a letter (to describe which pair of renderings corresponds). For instance, "UV Layout 1 and Rendering A" could be an answer.

Solution: Part 1:

Q1. The last 4 triangles are duplicated. To correct this, the last 12 entries should be changed to:

{... 5, 2, 1, 5, 3, 2, 5, 4, 3, 5, 1, 4}

Any cycling of these indices within any group of 3 is also correct; it is also correct if groups of 3 are rearranged. (6: +3 for identifying the right ones to change; +2 for subbing in 5; +1 for getting orientations correct. -0.5 for minor errors. Consistently reversing winding order is OK. Just "correct" is 2/6.)

Q2. The winding order on the last 4 triangles is incorrect. To correct this, the last 12 entries should be changed to:

{... 4, 1, 0, 4, 5, 1, 4, 3, 5, 4, 0, 3}

Any cycling of these indices within any group of 3 is also correct; it is also correct if groups of 3 are rearranged. (6: +4 for identifying the right ones to change; +1 for somehow permuting them; +1 for getting orientations correct. -0.5 for minor errors. Consistently reversing winding order is OK.)

Q3. This index buffer is correct. (4: +4 for saying so. up to +2 for saying something plausible in attempting to change it. Consistently reversing winding order is OK.)

Part 2:

Q4: UV layout 2 and Render C.

Q5: UV layout 3 and Render A.

(each 4: +2 for correct layout, +2 for correct rendering)

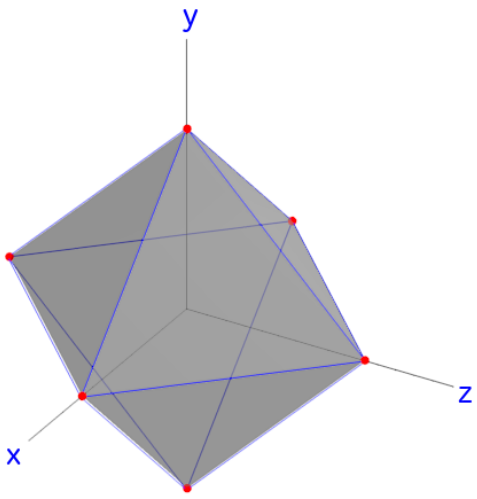
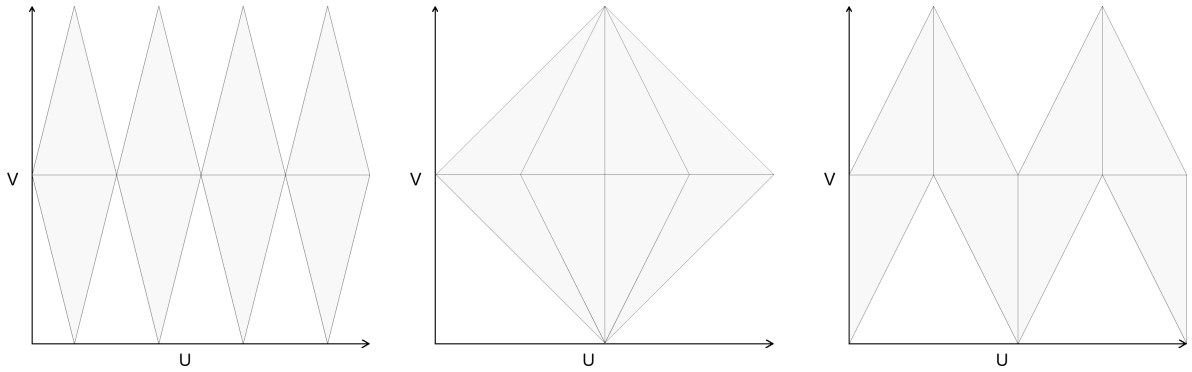


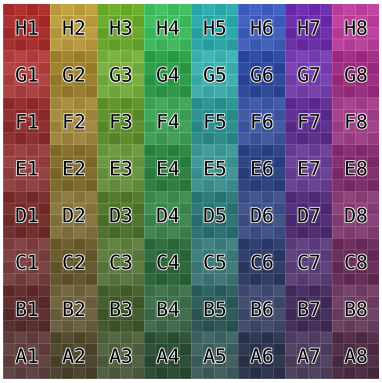
Figure 1: The target mesh configuration.



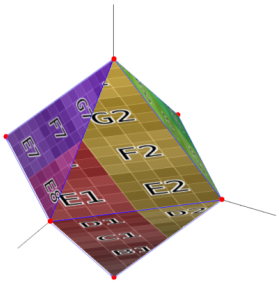
(a) UV layout 1. (b) UV layout 2. (c) UV layout 3.

Figure 2: Layouts of the mesh in UV-space.

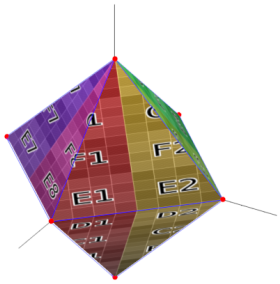
Last Name: _____ First Name: _____ Cornell NetID: _____



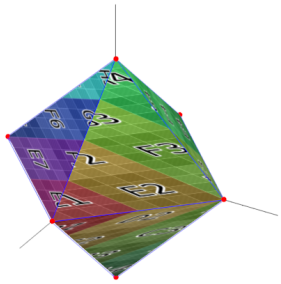
(a) The texture used in the following renderings.



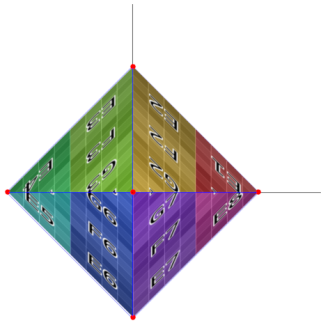
(b) Render A.



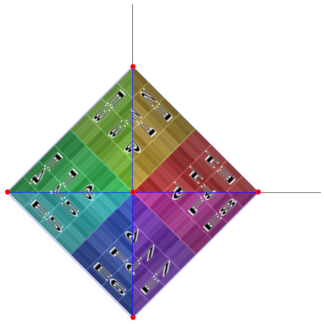
(c) Render B.



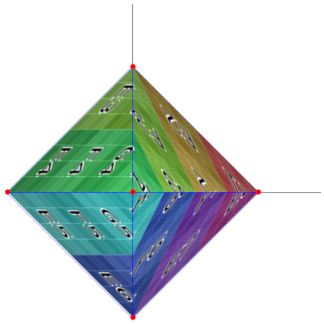
(d) Render C.



(e) Render A, top-down view.



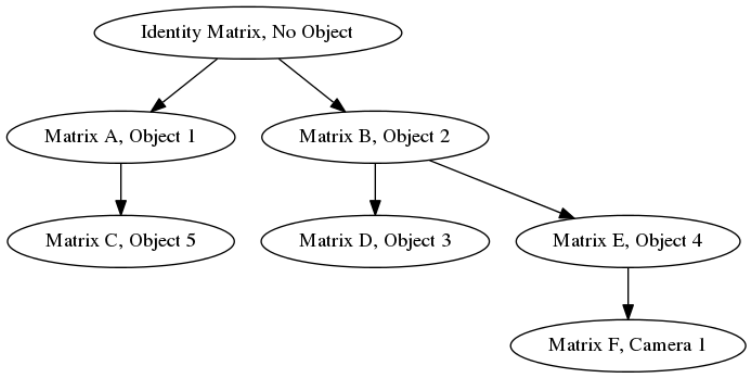
(f) Render B, top-down view.



(g) Render C, top-down view.

Figure 3: Views of the rendered mesh. Pictures in the same column have the same UV layout but a different camera location; the bottom row is a top-down view of the mesh.

4. [20 points] **Scene hierarchy** Consider the following simple scene graph. Each node is given with its corresponding local transformation matrix and object pair.



(a) Fill in the blank: In the standard scene graph model, the matrix D takes coordinates in _____'s frame to coordinates in _____'s frame. Solution: (Object 3, Object 2) (+2 fore each blank; 2/4 for having them swapped.)

(b) What is the associated M_{model} matrix, that brings object-space coordinates to world-space coordinates, for...

(a) Object 1: _____ (b) Object 3: _____

(c) Camera 1: _____

Solution:
(a) A (+2)
(b) BD (+2; 1/2 for reverse order)
(c) BEF (+2; 1/2 for reverse order)

(c) Suppose we are currently using Camera 1, which is a perspective camera that uses the perspective projection matrix M_{per1} and the viewport transformation M_{vp}. Unfortunately, all of our geometry is currently described in Object 5's frame. Say we wanted to view our geometry on a screen. How should we transform $\vec{p}$ (given in Object 5's frame) such that $\vec{p}$ ends up on our screen? Your solution should be of the form $X\vec{p}$ where X is a composition of several matrices. Additionally, include a sentence or two explaining how $X\vec{p}$ gets us screen coordinates.

Solution: $M_{vp}M_{per1}F^{-1}E^{-1}B^{-1}AC\vec{p}$. The resulting vector is of the form $\langle x_{screen}, y_{screen}, z_{canonical}, 1\rangle$. (5: +0.5 for M_{vp}; +1 for M_{per}; +1 for $F^{-1}E^{-1}B^{-1}$; +1 for AC; +0.5 for having node

transforms in the right order ("$(BEF)^{-1}$" is OK); +1 for saying something true and related to the question in the explanation.

(d) Suppose during the coordinate change described in the previous question, we'd like to translate point p by 5 units in the $+y$ direction. This is somewhat ambiguious, because we don't know which frame this transformation takes place in. How should we transform $\vec{p}$ (again as an expression $X\vec{p}$) if we want this to occur in...

i. In world space?

ii. In Object 2's space?

Solution: Let

$$X = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

a) $M_{vp}M_{per1}F^{-1}E^{-1}B^{-1}XAC\vec{p}$

b) $M_{vp}M_{per1}F^{-1}E^{-1}XB^{-1}AC\vec{p}$

(5: +1 for defining the matrix (ok to give precise description rather than matrix); +2 for getting it in the right place each time.)

5. [20 points] **Ray tracer bugs**

Look at each of the five images in Figure 5 that were produced by a Ray I ray tracer with various bugs. For each one choose one of the three possible explanations for the bug:

i. The error is caused by a problem with ray generation.

ii. The error is caused by a problem with ray intersection.

iii. The error is caused by a problem with shading computations.

For each choice, back it up with an example of an error that would cause the observed symptoms. There is no right or wrong explanation; only plausible and implausible ones. But when there is a clearly plausible cause, very far-fetched explanations will not make full credit. Shadow computations and texture operations count as part of shading. Computing surface normals counts as part of ray intersection. The first one is done as an example.

(a) iii. The problem is that the texture is flipped along the v axis, which could be caused by the texture mapping code failing to reverse the y coordinate when indexing into the texture image.

(b)

(c)

(d)

(e)

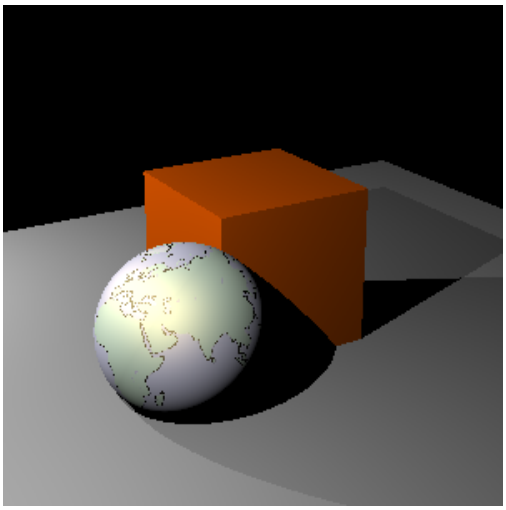


Figure 4: The correct rendering

Solution:

(a) already solved

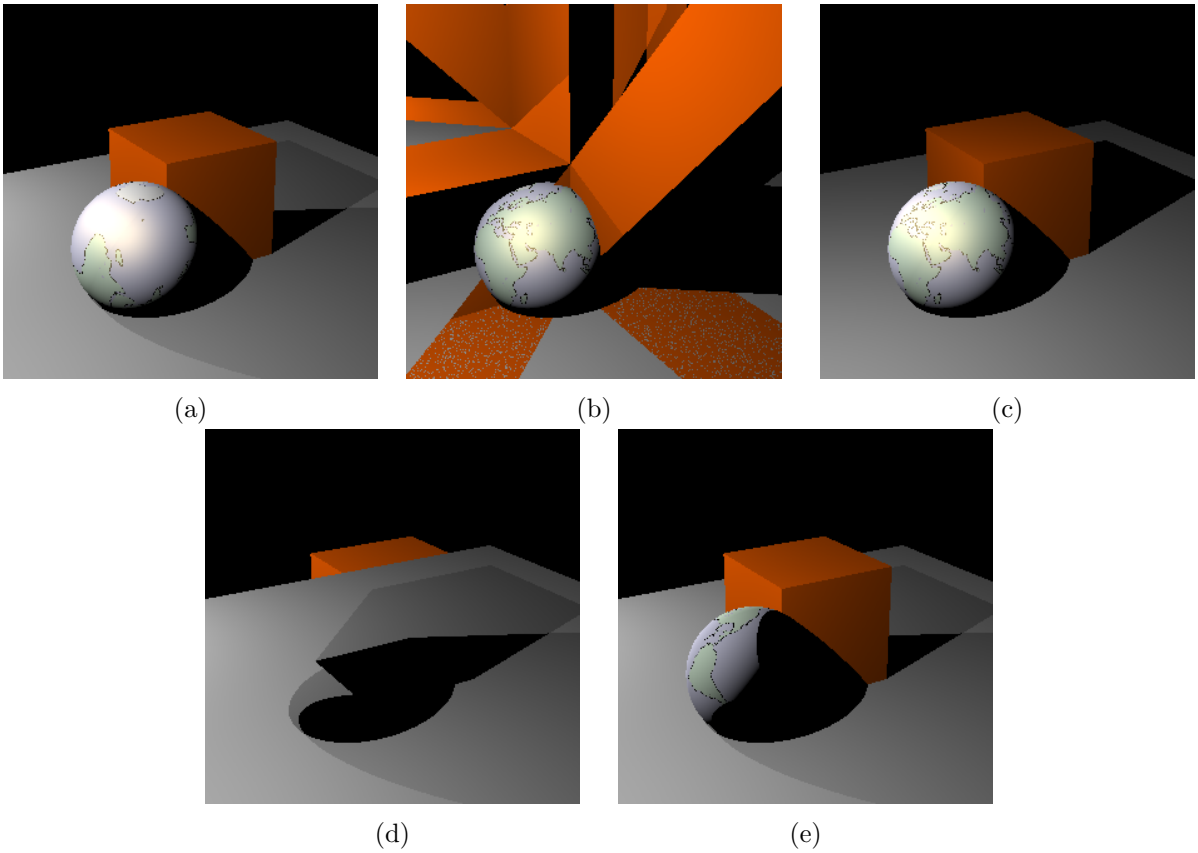
(b) Ray intersection (ii): Problem with triangle intersection, bad boundary check for alpha/beta/gamma.

(c) Shading computation (iii): Problem when iterating over lights in Lambertian shader (they don't need to say Lambertian).

(d) Ray intersection (ii): Not saving the closest intersection when intersecting the ray with the scene.

(e) Ray intersection (ii): Returning wrong root for sphere intersection.

(Each worth 5: +3 for correct image, +2 for plausible explanation. Different image with plausible alternate explanation is OK; 3/5 for far-fetched but remote possibility. Plan to accepty various creative answers for the last one.)



(a) (b) (c)

(d) (e)

Figure 5: Images produced by introducing single-statement bugs into the program.