Last Name: ________________ First Name: ____________ Cornell NetID: ________

CS 4620 Midterm, March 20, 2018

This 90-minute exam has **??** questions worth a total of **??** points. Use the back of the pages if you need more space.

Academic Integrity is expected of all students of Cornell University at all times, whether in the presence or absence of members of the faculty. Understanding this, I declare I shall not give, use, or receive unauthorized aid in this examination.

Signature: ________________________ Date __________

1. [20 points] **Meshes**

Suppose we are given the geometry shown in the left side of Figure **??**, and we are tasked with collapsing the red edge. This means combining its neighboring vertices into a new vertex. The resulting geometry is given on the right side of the figure. The new vertex's position will be the average position of the two old vertices.

Suppose also that each vertex in this mesh is associated with a normal vector. The new vertex's normal vector will be parallel to the average of the normals of the old vertices.

Below is a buffer that is filled with the positions of each of the vertices of the original mesh:

```
-1.0, 0.0, 0.0,
0.0, 1.0, 0.0,
0.0, 0.2, 0.0,
0.0, -0.2, 0.0,
0.0, -1.0, 0.0,
1.0, 0.0, 0.0
```

We are also given the following normal buffer:

```
-5/13, 0.0, 12/13,
0.0, 5/13, 12/13,
0.0, 8/17, 15/17,
0.0, -8/17, 15/17,
0.0, -5/13, 12/13,
5/13, 0.0, 12/13
```

Finally, we also have an index buffer. Each grouping of 3 entries describes a single triangle in the original mesh. An entry with value i selects the ith point in the position buffer above, and the ith vector in the normal buffer above.

```
0, 2, 1,
0, 3, 2,
0, 4, 3,
1, 2, 5,
2, 3, 5,
3, 4, 5
```

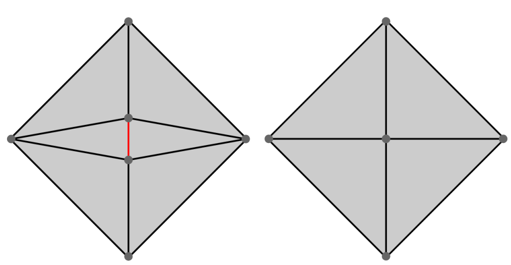


Figure 1: Left: the original mesh structure, with the edge to be removed highlighted in red. Note that this mesh is planar. Right: the resulting mesh structure.

(a) Write new position, normal, and index buffers that correspond to the mesh shown on the right side of Figure **??**. For vertices that remain in the mesh, leave the data from the original mesh in the same order. When adding new data to the position and normal buffers, add it at the end of the buffer. When multiple orderings of indices are possible, chose the ordering that puts the lowest index first. The triangles should be oriented such that the front side is facing out of the page. Use 0-based indexing.

Position buffer:

Normal buffer:

Index buffer:

(b) Suppose that we are to perform similar edge-collapsing operations many times on a large mesh. Name a mesh storage scheme that would allow us to do this more efficiently than the indexed structure defined above.

2. [15 points] **Cubemap**

Similar to the written question in A4 shaders, imagine we have a cube map with cross-like texture, shown in the figure below.

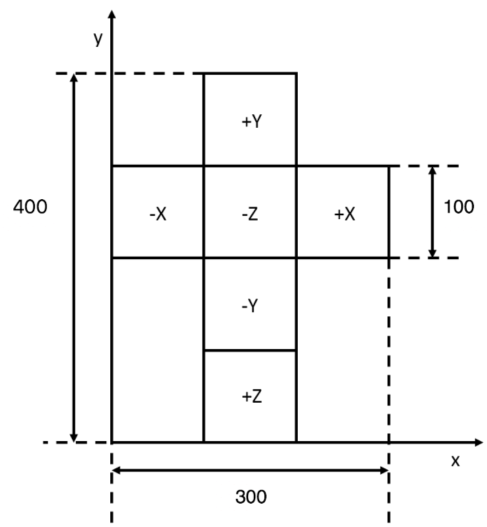


Figure 2: cube map with cross like texture.

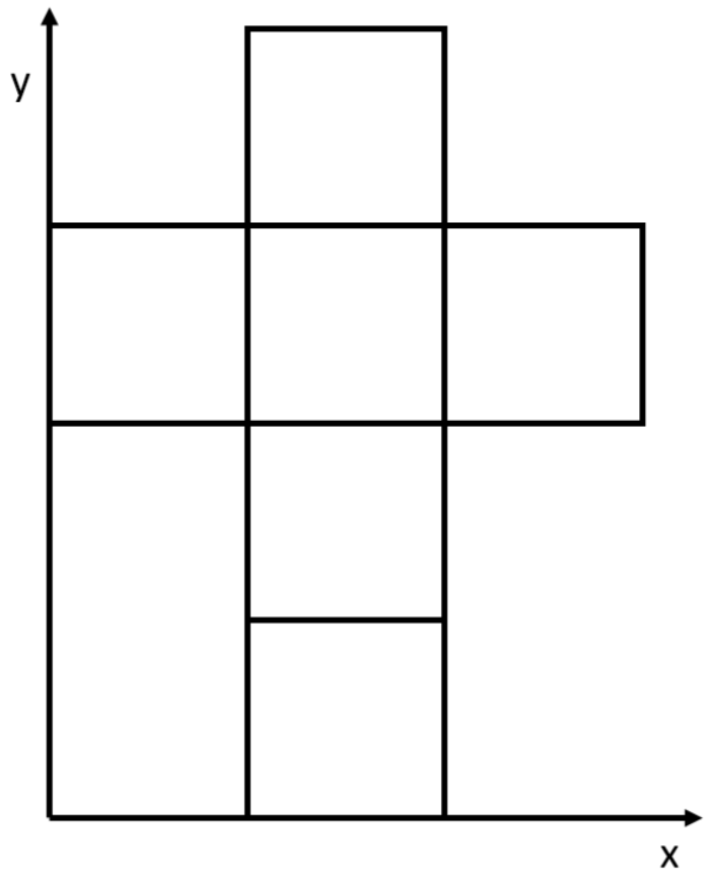
Suppose we have a perspective camera positioned at $(0, 0, -25)$ looking towards $(0, 0, 0)$. A square mirror surface 100 by 100 units in size is centered at $(0, 0, 0)$ with its normal pointing towards $-z$. The scene is rendered using the cube map as an environment map.

- (a) Point A $(25, 0, 0)$ lies on the specular reflecting square. Imagine a viewing ray that intersects the square at A and is specularly reflected. What is the 3D unit vector that will be used to look up in the cube map? At what pixel coordinates will the texture in Figure 2 be sampled?

- (b) Answer the same two questions for Point B $(25, 25, 0)$.

- (c) Answer the same two questions for Point C $(50, 0, 0)$ on the edge of the mirror.

- (d) Which part of cube map is reflected by the specular square? Shade in the region in the figure below, marking dimensions so that we can tell exactly which part is shaded. Briefly explain the reasoning behind your answer.



3. [15 points] **Transformation Stack**

   Consider how an object is transformed in the programming part of the Scene assignment. The object's canonical-to-world transformation is composed of 5 different transforms in a "stack." Here is a recap of the transformations.

   - T A translation.
   - R_x A rotation around the x-axis.
   - R_y A rotation around the y-axis.
   - R_z A rotation around the z-axis.
   - S An axis-aligned scaling.

   The stack of transformations is combined in order so that the final transformation is $M = TR_xR_yR_zS$.

   The transformations are controlled by three kind of manipulators: translation, rotation, and scale. There are three manipulators of each kind, controlling transformations in the x, y, and z direction or about the x, y, and z axes.

   For the following question, give your answers using the five transformations described above.

   - (a) Consider a point **o** on the object, written in the coordinate system of the object. What is **o** written in world coordinates?

   - (b) Consider a point **p** on the x-axis translation manipulator, written in the coordinate system of the manipulator itself. What is **p** written in world coordinates?

   - (c) Consider a point **q** on the y-axis rotation manipulator, written in the coordinate system of the manipulator itself. What is **q** written in world coordinates? What is the axis of rotation in world coordinates?

4. [15 points] **Normal interpolation**

   (a) Explain why we want to interpolate vertex normals for shading.

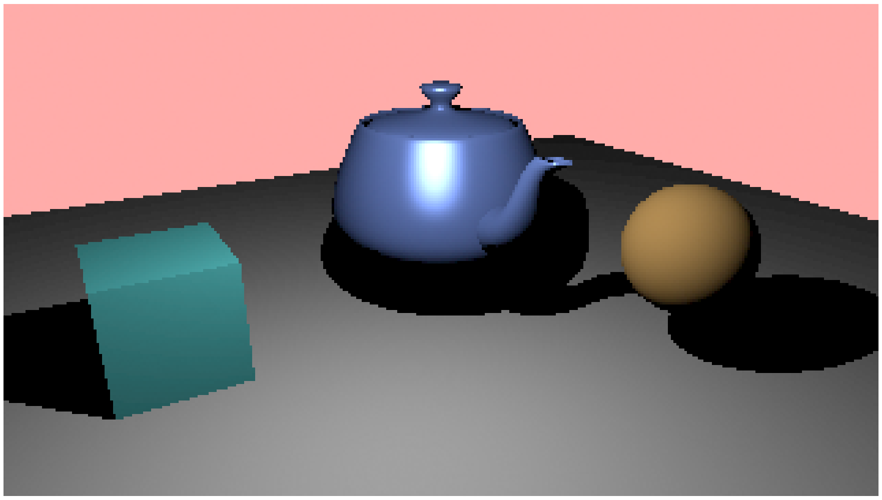
   (b) True or False?
   For any point on a face ABC, the normal we use for shading is a convex linear combination of the vertex normal vectors $\mathbf{n}_a$, $\mathbf{n}_b$, and $\mathbf{n}_c$.[1]

   (c) Suppose we have a triangle mesh with shared vertex normals, which contains two triangles PQR and RQS that share an edge. What are the barycentric coordinates of the midpoint of the edge QR in each of these two triangles?

   (d) Prove using barycentric coordinates that the interpolated normal at any point on the edge QR has the same value when calculated in triangle PQR using the vertex normals $\mathbf{n}_p$, $\mathbf{n}_q$, and $\mathbf{n}_r$ or in triangle RQS using the vertex normals $\mathbf{n}_r$, $\mathbf{n}_q$, and $\mathbf{n}_s$.

[1] Reminder: a convex linear combination is a linear combination in which the weights are positive and sum to 1.

5. [15 points] **Ray Tracing Bugs**



The image here is a correct ray traced rendering of a scene consisting of a single light, a box, a teapot mesh, a sphere, and a plane (approximated by a box with a small height). The teapot has Microfacet Beckmann shading, and all other surfaces have Lambertian. We have modified the ray tracer to introduce several single-line bugs, one for each image you see on the next page. For each image, 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) Example: Bug ii: Ray intersection is not properly determining the first object hit.

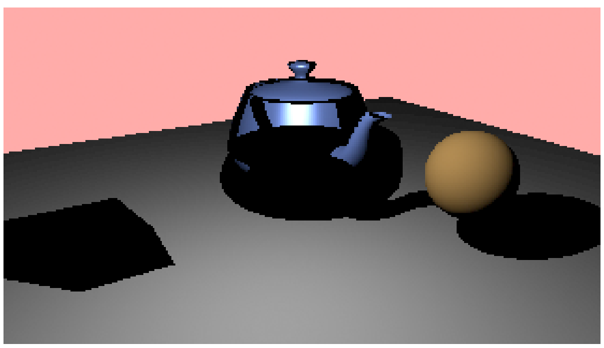
(b)

(c)

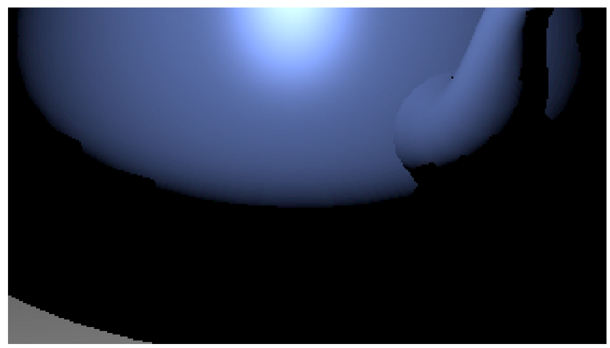
(d)

(e)

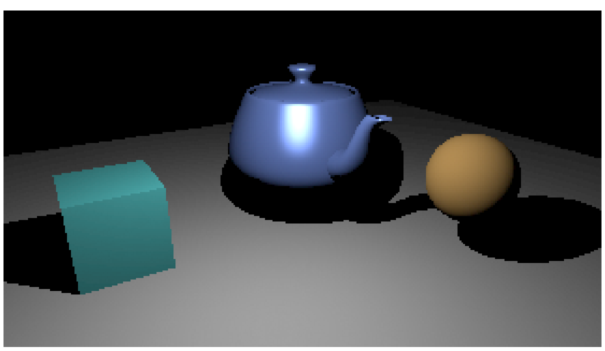
(f)



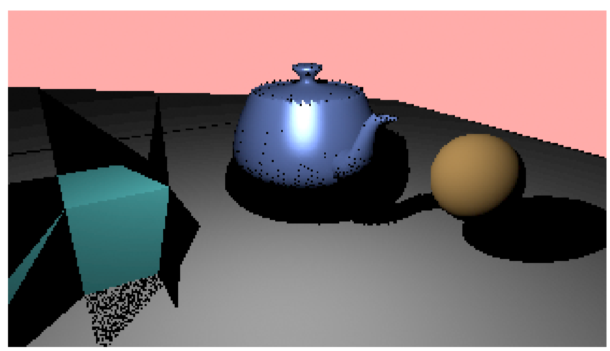
(a)



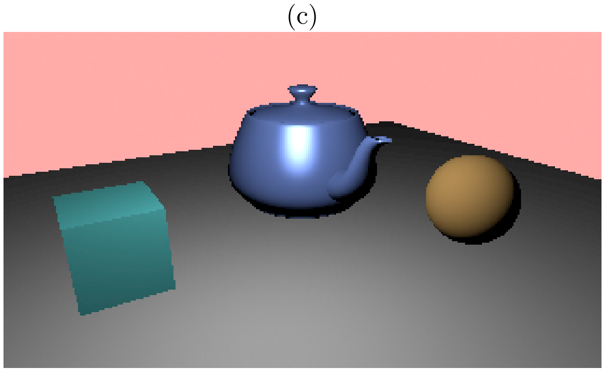
(b)



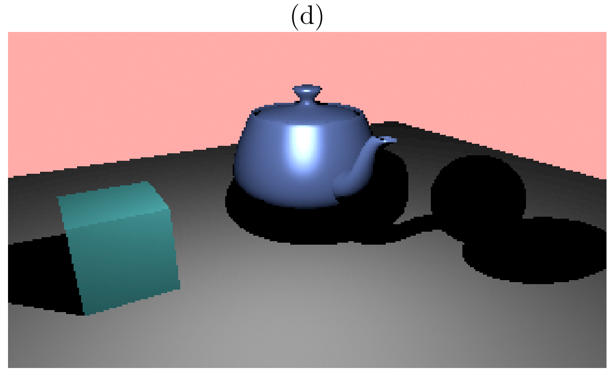
(c)



(d)



(e)



(f)