Last Name: ____________________ First Name: ____________ Cornell NetID: ______

CS 4620 Final, May 20, 2018
SOLUTION

1. [20 points] **Overview**

   (a) True or False?

   1. ____ In shading computations, the light direction must be normalized in order to get correct results.
   2. ____ When checking if a ray intersects some geometry, the ray direction must be normalized in order to get correct results.
   3. ____ In the Microfacet shading model, the specular highlight increases in size as the surface roughness increases.
   4. ____ In the Microfacet shading model, the peak brightness of the specular highlight increases as the surface roughness increases.
   5. ____ When interpolating transformations for animation, it's important to handle the translation component using quaternions.
   6. ____ Affine transformations preserve right angles.
   7. ____ Perspective transformations preserve similar triangles.
   8. ____ Affine transformations preserve similar triangles.
   9. ____ Perspective transformations preserve straight lines.
   10. ____ Affine transformations preserve parallel lines.
   11. ____ Perspective transformations preserve parallel lines.
   12. ____ 3D rigid body transformations commute.
   13. ____ 3D uniform scale transformations commute.
   14. ____ 3D rotations around the x axis commute.
   15. ____ To transform normals properly, you must apply the inverse transpose of the matrix used to transform positions.
   16. ____ To transform a direction by a perspective matrix, you multiply the matrix by the direction vector with a 1 in the fourth coordinate, then divide the resulting vector by its fourth coordinate.
   17. ____ To transform a direction by a perspective matrix, you multiply the matrix by the direction vector with a 0 in the fourth coordinate, then divide the resulting vector by its fourth coordinate.

   (b) Matching: for each of the following, write the number of the sentence fragment that correctly completes the statement.

   (a) A ray tracing renderer... ____
   (b) A fragment shader... ____
   (c) A vertex shader... ____
   (d) A rasterization renderer... ____
   (e) A cubic B-spline... ____

(f) A Catmull-Rom spline... ____
(g) A cubic Bézier spline... ____
(h) Euler angle interpolation... ____
(i) Linear quaternion interpolation... ____
(j) "Slerp" interpolation... ____

(1) ...processes objects one at a time.
(2) ...processes pixels one at a time.
(3) ...processes fragments one at a time.
(4) ...processes vertices one at a time.
(5) ...interpolates its control points.
(6) ...maintains C^2 continuity between segments.
(7) ...can be evaluated from its control points by the de Casteljau algorithm.
(8) ...does not have a constant axis of rotation.
(9) ...interpolates at a constant speed between endpoints.
(10) ...interpolates at a non-constant speed between endpoints.

(c) Short answer:

(a) What type of variable, if used in a GLSL shader program, must appear in both the vertex and fragment shaders? What is it used for?

(b) What two types of GLSL variables must have matching names between CPU and shader code? What are they used for?

(c) Classify the following splines as having or not having the convex hull property: Bézier, Catmull-Rom, B-Spline.

(d) Briefly explain how importance sampling can improve performance.

Solution:
a. [8 pts; -0.5 for each wrong answer, clamped at 0]
1. T
2. F
3. T
4. F
5. F
6. F
7. F
8. T

9. T
10. T
11. F
12. F
13. T
14. T
15. T
16. F
17. F

b. [10 pts, one for each match]
(a) 2
(b) 3
(c) 4
(d) 1
(e) 6
(f) 5
(g) 7
(h) 8
(i) 10
(j) 9

c. [2 pts each. 1 for saying something true; 2 for answering the question correctly.]
(a) Varying variables must appear in both vertex and fragment shaders. They let the fragment shader read values interpolated between vertices.
(b) Vertex attributes and uniform variables. Vertex attributes are values that are defined per-vertex; uniform variables are constant for all vertices and fragments.
(c) Bezier and B-Spline have the convex hull property; Catmull-Rom does not have the convex hull property.
(d) Sampling from a probability distribution that has a similar shape to the target function reduces variance and improves the convergence rate.

Last Name: ____________________ First Name: _______________ Cornell NetID: ______

2. [20 points] **Perspective**

Consider the following two projection matrices that might be used in OpenGL.

$$P_1 = \begin{bmatrix} 4/3 & 0 & 0 & 0 \\ 0 & 4/3 & 0 & 0 \\ 0 & 0 & 3 & 16 \\ 0 & 0 & -1 & 0 \end{bmatrix} \quad P_2 = \begin{bmatrix} 1/3 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 \\ 0 & 0 & 1/2 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For each of these two matrices:

(a) Does this matrix specify an orthographic or a perspective view?

(b) Where does this matrix send the point (3, 3, -4)?

(c) Where are the near and far planes?

(d) What is the vertical field of view? (This is either a distance in eye space units or an angle.)

(e) Is the point with eye space coordinates (-4, 0, -6) visible to the camera? Explain how you decided.

Hint: it is helpful to know the inverse matrices. They are (up to a scale factor, to keep the numbers simple):

$$P_1^{-1} = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & -4 \\ 0 & 0 & 1/4 & 3/4 \end{bmatrix} \quad P_2^{-1} = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & -6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Solution: [2 points each half of each part, 1 for writing some work that could be part of a solution, -0.5 for minor calculation errors.] a) P_1 is perspective, P_2 is orthographic.

b) P_1 sends the point to $\begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$

P_2 sends the point to $\begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$

c) P_1 near plane: z=-4, far plane z=-8

P_2 near plane: z=-4, far plane: z=-8

d) P_1: 2arctan(3/4) = 1.29, or 2arctan(3/4)*180/pi = 73.74 P_2: 6

e) Let v = [-4, 0, -6, 1]. P1 sends the point to [-0.8889, 0, -0.3333, 1], so it is visible to the perspective camera. P2 sends the point to [-1.3333, 0, 0, 1], so it is not visible to the orthographic camera.

3. [20 points] **Splines**

The matrix form of the quadratic Bézier spline is

$$\mathbf{p}(t) = \begin{bmatrix} 1 & t & t^2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \end{bmatrix}$$

(a) Evaluate this spline for $t = \frac{1}{2}$.

(b) What is the tangent to this spline at the two ends, in terms of the control point positions?

We would like to define a quadratic spline in which a segment is controlled by three control points $\mathbf{p}_0$, $\mathbf{p}_1$, and $\mathbf{p}_2$, and it passes through these points at $t = 0$, $\frac{1}{2}$, and 1 respectively.

(c) Construct the spline matrix for this spline. Explain your reasoning. Use the ordering $\begin{bmatrix} 1 & t & t^2 \end{bmatrix}$.

(d) Evaluate your spline for $t = \frac{1}{2}$ to show that your matrix works.

(e) Does your spline have the convex hull property?

Solution: [half credit for writing some work that could be part of the solution. -0.5 to -1 for computation errors.] a) [2 pts] $1/4p_0 + p_1 + p_2$

b) [4 pts] tangent: $(2t-2)p_0 + (2-4t)p_1 + 2tp_2$. When t=0, this evaluates to $2(p_1 - p_0)$. When t=1, this evaluates to $2(p_2 - p_1)$.

c) [8 pts] $\begin{bmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{bmatrix}$

d) [4 pts] When t=1/2, the spline evaluates to p_1.

e) [2 pts] No, not all the basis functions are positive.

4. [20 points] **GLSL**

   In Ray 1 you were provided with a shader that produced an RGB color according to the point's normal direction in world space coordinates. In this problem you will implement the same shader in GLSL. However, the normal direction you will use is the modified normal direction after normal mapping. Specifically:

   - The vertex shader should do the standard transformation to produce `gl_Position`, as well as calculating any other values you may need.
   - The fragment shader should use the interpolated uv coordinate to look up the weight vector $\{w_n, w_{t1}, w_{t2}\}$ from the texture `normalMap`. You can assume that the weight vectors are normalized.
   - The fragment shader should then compute the shifted normal according to the interpolated world space normal and world space tangents that would be used for shading. The output normal should be equal to $2*(w_n - 0.5)*$`normal` + $2*(w_{t1} - 0.5)*$`tangent1` $+2*(w_{t2} - 0.5)*$`tangent2`.
   - Lastly, the fragment shader should produce `gl_FragColor` based on the shifted normal. The output color should be `normalToColor(shiftedNormal)`, and the fragment should be fully opaque.

   Vertex shader:

```
uniform mat4 modelMatrix;
uniform mat4 viewMatrix;
uniform mat4 modelViewMatrix;
uniform mat4 projectionMatrix;
uniform sampler2D normalMap;

uniform mat3 normalMatrix; // = inverse transpose of modelMatrix

attribute vec3 position;
attribute vec3 normal;
attribute vec3 tangent1;
attribute vec3 tangent2;
attribute vec2 uv;

// Declare any varyings here:



void main() {
    // Implement vertex shader here.



}
```

Fragment shader:

```
// Declare any varyings here:





vec3 normalToColor (vec3 shiftedNormal) {
    // implementation omitted, but you
    // can assume it already works
}

void main() {
    // Implement fragment shader here.















}
```

Solution:

Vertex shader:

```
uniform mat4 modelMatrix;
uniform mat4 viewMatrix;
uniform mat4 modelViewMatrix;
uniform mat4 projectionMatrix;
uniform sampler2D normalMap;

uniform mat3 normalMatrix; // = inverse transpose of modelMatrix

attribute vec3 position;
attribute vec3 normal;
attribute vec3 tangent1;
attribute vec3 tangent2;
attribute vec2 uv;

// Declare any varyings here:
// 2 points
varying vec2 vuv;
varying vec3 vNormal;
varying vec3 vTangent1;
varying vec3 vTangent2;

void main() {
    // Implement vertex shader here.
    // 2 points
    gl_Position = projectionMatrix * modelViewMatrix * vec4(position, 1.0);

    // 1 point
    vuv = uv;
    // 2 points
    vNormal = normalize(normalMatrix * normal);
    // 2 points
    vTangent1 = normalize(modelMatrix * vec4(tangent1, 0.0));
    vTangent2 = normalize(modelMatrix * vec4(tangent2, 0.0));
}
```

Fragment shader:

```
// Declare any varyings here:
// 1 points
varying vec2 vuv;
varying vec3 vNormal;
varying vec3 vTangent1;
varying vec3 vTangent2;
```

```
vec3 normalToColor (vec3 shiftedNormal) {
    // implementation omitted, but you
    // can assume it already works
}

void main() {
    // Implement fragment shader here.

    // 2 points
    vec4 wt = texture2D(normalMap, vuv);
    // 2 points
    vec3 N = normalize(vNormal);
    vec3 T1 = normalize(vTangent1);
    vec3 T2 = normalize(vTangent2);


    // 4 points
    vec3 sNormal = 2*(wt.x-0.5)*N + 2*(wt.y-0.5)*T1+ 2*(wt.z-0.5)*T2;

    // 2 points
    gl_FragColor = vec4(normalToColor(sNormal), 1.0);
}
```

5. [20 points] **Transformation**

- (a) Which of the 2D homogeneous transformations defined by the matrices below are rigid body transformations?

$$A_1 = \begin{bmatrix} 1 & 0 & 23 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix};\ A_2 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix};\ A_3 = \begin{bmatrix} 1 & 0.5 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix};\ A_4 = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

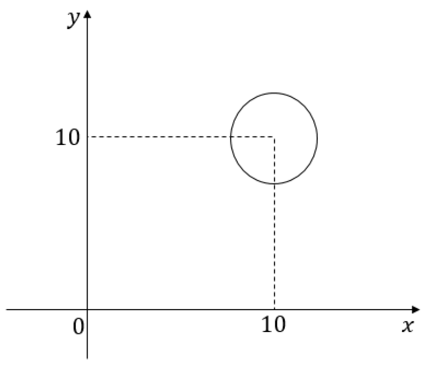
- (b) Suppose we are given a curve defined in implicit form as $C = \{\mathbf{x} \in \mathbb{R}^2 \mid f(\mathbf{x}) = 0\}$ for some function $f : \mathbb{R}^2 \to \mathbb{R}$. If we transform this curve by a linear transformation M, what is the implicit form of the transformed curve (using the same style of set notation as we did to define C above)?

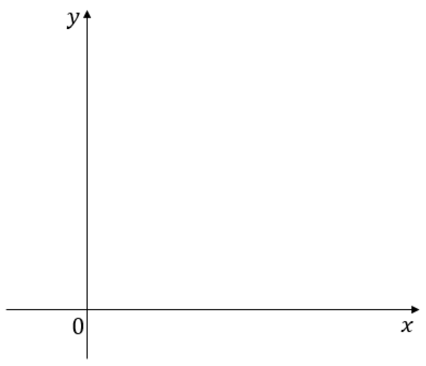
- (c) Using the matrices:

$$T = \begin{bmatrix} 1 & 0 & -10 \\ 0 & 1 & -10 \\ 0 & 0 & 1 \end{bmatrix};\ S = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix};\ R = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

suppose there is a circle in the plane, and its equation is $\{\mathbf{x} \in \mathbb{R}^2 \mid \|T\mathbf{x} - \mathbf{0}\| = 1\}$, where $\mathbf{0}$ is the origin. It is shown in the left panel of the figure below. What is the equation of the new shape after applying the transformation $T^{-1}RST$ to the circle? It's best to keep your answer in terms of matrix and vector products, without writing out the components.

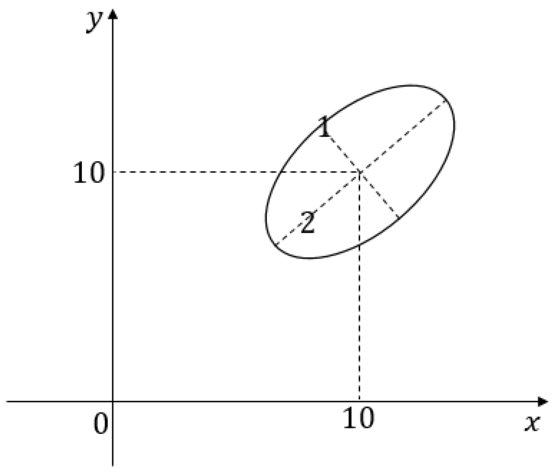
- (d) Draw the transformed shape in the right panel, marked with enough dimensions and angles to denote its position and size.





Solution:

- (a) [4 pts, graded as a 4-bit XOR] A_1 and A_4
- (b) [5 pts] $\{\mathbf{x} \in I\!R^2 \mid F(M^{-1}\mathbf{x}) = 0\}$
- (c) [5 pts] $\{\mathbf{x} \in I\!R^2 \mid \|S^{-1}R^{-1}T\mathbf{x}\| = 1\}$
- (d) [6 pts: 2 each for position, orientation, scale of the ellipse.]



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

6. [20 points] **Monte Carlo Integration**

We would like to compute the definite integral

$$\int_0^2 e^{-x}\, dx$$

with Monte Carlo integration.

(a) What is the value of the definite integral? Show your work.

(b) If we sample x from the interval $[0, 2]$ uniformly. For any $x \in [0, 2]$, what is the probability density $p(x)$ with which x gets sampled?

(c) We would like to create an estimator $g_1(x)$ for $\int_0^2 e^{-x}\, dx$ using the uniform sampling distribution. Write the expression for $g_1(x)$:

(d) Instead of sampling uniformly, we now sample x with probability density $q(x) = \frac{2-e^2}{2e^2}x + 1$. Write the expression for the estimator $g_2(x)$ for $\int_0^2 e^{-x}\, dx$ that uses this sampling distribution.

Solution:

(a)

$$\int_0^2 e^{-x} dx = 1 - e^{-2}$$

[3 points]

Last Name: ______________________ First Name: ______________________ Cornell NetID: __________

(b)

$$p(x) = \frac{1}{(2-0)} = \frac{1}{2}.$$

[5 points: 3 for indicating awareness of probability density concept, 2 for right answer.]

(c)

$$g_1(x) = \frac{e^{-x}}{p(x)} = \frac{e^{-x}}{1/2} = 2e^{-x}.$$

[6 points: 3 for attempting to compute f/p; 2 for doing it pretty much right; 1 for having normalization correct.]

(d)

$$\begin{aligned} g_2(x) = \frac{e^{-x}}{q(x)} &= \frac{e^{-x}}{\frac{2-e^2}{2e^2}x + 1} \cdot \frac{2+e^2}{e^2} \\ &= \frac{2(2+e^2)e^{-x}}{(2-e^2)x + 2e^2} \end{aligned}$$

[6 points: 3 for attempting to compute f/p; 6 for having an answer that is correct assuming q(x) was normalized. +1 extra point for trying to normalize q; +1 additional for correctly fixing the normalization.]