University of California San Diego
Department of Computer Science and Engineering
CSE167: Introduction to Computer Graphics
Fall Quarter 2018
Midterm Examination #1
Thursday, October 24th, 2019
Instructor: Dr. Jürgen P. Schulze

Name:_____

Your answers must include all steps of your derivations, or points will be deducted.

This is closed book exam. You may not use electronic devices, notes, textbooks or other written materials.

All coordinate system are right-handed.

Good luck!

*Do not write below this line*

| Problem | Max. | Points |
|---|---|---|
| 1 | 10 | |
| 2 | 10 | |
| 3 | 10 | |
| 4 | 10 | |
| 5 | 10 | |
| 6 | 10 | |
| 7 | 10 | |
| 8 | 10 | |
| **Total** | 80 | |

**1. True or False (10 Points)**

True or False: Circle one choice only for each part. (1 point each)

T / F: The inverse of a 3D rotation matrix is simply its transpose.

T / F: The surface normal is the vector that is parallel to the surface.

T / F: Vector $(1, 1, 1)^T$ is a unit vector.

T / F: In rotations with Euler angles the order of rotations is always first about the x axis, then the y axis, then the z axis.

T / F: An affine transformation is any transformation that preserves collinearity and ratios of distances.

T / F: The law of associativity always applies to a sequence of affine transformations.

T / F: The light rays emanating from a perfect point light source are never parallel, unless lenses or mirrors are involved.

T / F: Spotlights as defined in this course are directional lights with three additional parameters.

T / F: Generally, in a typical OpenGL application, GLSL fragment shaders are executed many more times than vertex shaders.

T / F: The tesselation level of a sphere is related to the number of triangles it is rendered with.

**2. Affine Transformations (10 Points)**

1. Write 4x4 matrices for the following affine transformations in 3D space: (3 points)
    1. translate by (3, -1, 4)

    2. uniform scaling down by a factor of 4

    3. rotate 90 degrees about the y-axis

2. You have a 3D model that has been translated via its model matrix such that its center is at (1, 2, -2) (meaning that its model matrix is a translation matrix). We want to change the position/orientation of the model by applying an affine transformation to update the model matrix.
    a. Show the order of the matrix multiplications to update the model matrix in order to orbit the model about the origin (Let R be a rotation matrix) (hint: we want as an example X = YZ for the appropriate matrices) (2 points)

    b. Show the order of matrix multiplications to update the model matrix in order to spin the model about the point (1, 2, -2) (Let R be a rotation matrix) (2 points)

    c. Show the order of matrix multiplication to update the model matrix to translate the model closer to the origin as in programming assignment 2. (1 point)

    d. Write the translation matrix used to translate the model closer to the origin as in programming assignment 2 (2 points)

**3. Coordinate System Transformation (10 Points)**

Given a point p with *camera space* coordinates p = (2, 1, 1).

In addition, camera space has its origin at $(2, \frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}})$ in *world space*, and the basis vectors of camera space have *world coordinates* $(0,1,0)$, $\left(\frac{1}{\sqrt{2}}, 0, \frac{1}{\sqrt{2}}\right)$, $\left(\frac{1}{\sqrt{2}}, 0, \frac{-1}{\sqrt{2}}\right)$.

What are the *world space* coordinates of p?

**4. GLSL Shaders (10 Points)**

Given the following shader code:

```
gl_Position = projection * view * model * vec4(position, 1.0f);
FragPos = vec3(model * vec4(position, 1.0f));
Normal = mat3(transpose(inverse(model))) * normal;
```

Please describe:

1. In homework project 2, where would you most likely use this piece of code? Be specific. (2 points)

2. Given that M: object-to-world matrix, C: camera matrix, P: projection matrix, D: viewport matrix

   a) Find the corresponding variables in the first line of the above shader code. (2 points)

   b) Describe what coordinate systems the following variables from the above shader code are in: “position”, “gl_Position”, “FragPos” and “Normal”. (4 points)

   *position:*

   *gl_Position:*

   *FragPos:*

   *Normal:*

3. Given the following two lines of C++ code:

```
A) model = glm::rotate(model, angle, axis);
B) model = glm::rotate(angle, axis) * model;
```

Which of the lines of code above (A or B) makes a 3D model:

a) spin about its own origin (1 point)?

b) rotate about the center of the world coordinate system (1 point)?

**5. Lighting (10 Points)**

a) Name two differences between the concepts of global Illumination and local Illumination? (4 points)

b) Explain the difference between faceted and smooth shading, and explain how both the Gouraud and Phong shading methods work, and their limitations. (4 points)

c) Explain the difference between Phong shading and the Phong illumination model. (2 points)

**6. Diffuse Reflection (10 Points)**

As derived in class, in the Phong Illumination Model diffuse reflection is is calculated as:

$$c_d = c_l k_d (\mathbf{n} \cdot \mathbf{L})$$

1. Briefly describe each of the five variables in this equation: (5 points)

2. Diffuse reflection is described by “Lambert’s cosine law”. Explain why. (2 points)

3. Name three examples of real-world objects which are mostly diffuse reflectors (with little or no specular reflection) (3 points).

**7. Lights (10 Points)**

Everything below assumes the lighting model we discussed in class.

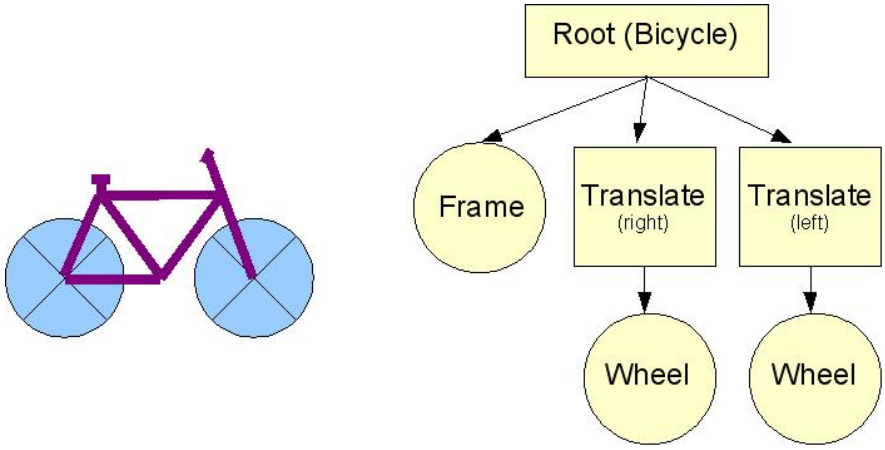
a) Name two differences between directional lights and point lights. (2 points)

b) Name the three additional parameters spotlights have compared to point lights. (3 points)

c) How do the three distance attenuation options for point lights we covered in class differ from one another? (3 points)

d) Why are there multiple options for distance attenuation? (2 points)

**8. Scene Graph (10 Points)**




1. The figure above illustrates the scene graph for a bicycle that is not moving. Given the partial source code below, add the code that is missing to build the scene graph for the bicycle. Use the function `addChild(__)` to add one scene graph node to another, assuming that it is correctly defined for all `Transform` nodes. (5 points)

```
root = new Transform(…);
frame = new Frame();
wheel = new Wheel();
translateRight = new Transform(…);
translateLeft = new Transform(…);
```

2. Assume that a draw method has been defined for all scene graph node types, and it takes a 4x4 model matrix as its only parameter. Write the pseudo code that will render the entire bicycle, assuming that the draw methods were written correctly. (3 points)

3. Now you want to rotate the wheels of the bicycle by adding scene graph nodes in the right places. What changes are necessary to the scene graph? Insert them them into the scene graph diagram at the top, and briefly explain below. (2 points)