THIS PAPER IS NOT TO BE REMOVED FROM THE EXAMINATION HALLS

UNIVERSITY OF LONDON **CO3355 ZA**

BSc Examination

COMPUTING AND INFORMATION SYSTEMS AND CREATIVE COMPUTING

Advanced Graphics and Animation

Date and Time: Friday 19 May 2017: 10.00 – 12.15

Duration: 2 hours 15 minutes

There are FIVE questions on this paper. Candidates should answer **THREE** questions. All questions carry equal marks and full marks can be obtained for complete answers to **THREE** questions. The marks for each part of a question are indicated at the end of the part in [.] brackets.

Only your first **THREE** answers, in the order that they appear in your answer book, will be marked.

There are 75 marks available on this paper.

A hand held calculator may be used when answering questions on this paper but it must not be pre-programmed or able to display graphics, text or algebraic equations. The make and type of machine must be stated clearly on the front cover of the answer book.

1. Maths, projections and transformations.

(a) Provide the formula for the Euclidean distance between points **a** and **b**, defined below:

$$a = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ 1 \end{bmatrix} \text{ and } b = \begin{bmatrix} b_1 \\ b_2 \\ 0 \\ 1 \end{bmatrix}$$

[2]

(b) Consider a vector $[0.5, 2, 1, 1]^T$. Calculate the result of transforming it by the matrix below and describe the kind of transform performed.

$$\begin{bmatrix} 2 & 0 & 0 & 1 \\ 0 & 2 & 0 & 1 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

[6]

(c) Let an object in 3D space, centered at the origin, a uniform scaling transformation S and a translation transformation T. Consider the following cases:

a. First scale the object by S then translate it by T
b. First translate the object by T then scale it by S

Now for each of the following propositions, state whether it is always true, always false or it depends on the transformations and justify your answer in a sentence.

i. *In neither case has the object moved from the origin.*
ii. *The object is the same size in both cases.*
iii. *The object is in the same position in both cases.*
iv. *The object has moved more in case a.*

[10]

(d) Identify an advantage of each of the two basic types of projections (perspective and parallel). How will a set of parallel lines appear in each?

[4]

(e) Given the vector $[x_a, y_a, z_a, 1]^T$ calculate the result of its orthographic projection on the plane: z=-1.

[3]

2. Animation

(a) What is the problem that the technique of *layering* solves in traditional animation? Describe briefly how it does this.

[5]

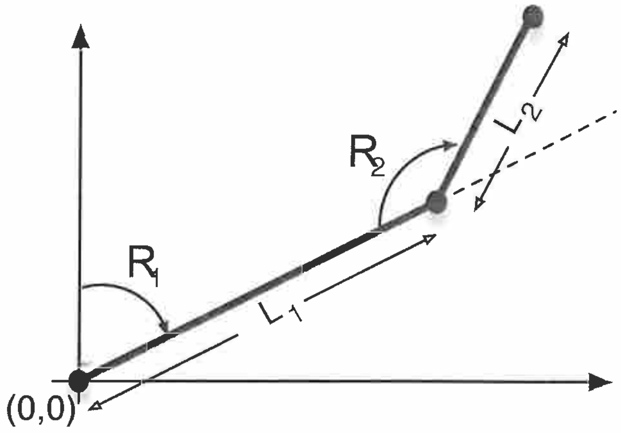
(b) Consider an animation depicting the motion of a ball encoded so that in between two consecutive key frames, there are 9 interpolated ones. Let two consecutive key frames A and B where the ball is positioned at $p_A=[4, 5, 9]^T$ and $p_B=[7, 8, 2]^T$ respectively. Perform linear interpolation to calculate the position of the ball at the frame appearing right before B. Show your work.

[5]

(c) When representing curves, how does the Hermite form differ from the Bezier one?

[2]

(d) How is the position of a bone in skeletal animation calculated using forward kinematics? Derive the formula using the following figure that depicts a skeleton consisting of 2 bones with lengths L_1 and L_2 respectively.



[7]

(e) For a triangle with angles **A**, **B** and **C** and lengths of respectively opposite sides **a**, **b** and **c**, the law of cosines states that $\mathbf{c}^2 = \mathbf{a}^2+\mathbf{b}^2 - 2\mathbf{ab}\cos\mathbf{C}$.

Consider again the previous figure and let $L_1=4$ and $L_2=2$ and the desired position of the end of the second bone to be (-5, 0). Use the law of cosines to solve this simple inverse kinematics problem by determining the joint angles θ_1 and θ_2 that correspond to R_1 and R_2. Be sure to check that the desired position is within reach first.

[6]

3. Shading

(a) What is meant by the term *depth cueing*?

[3]

(b) Explain why Phong shading is more computationally expensive than Gouraud shading.

[4]

(c) Explain the concept of ray-tracing.

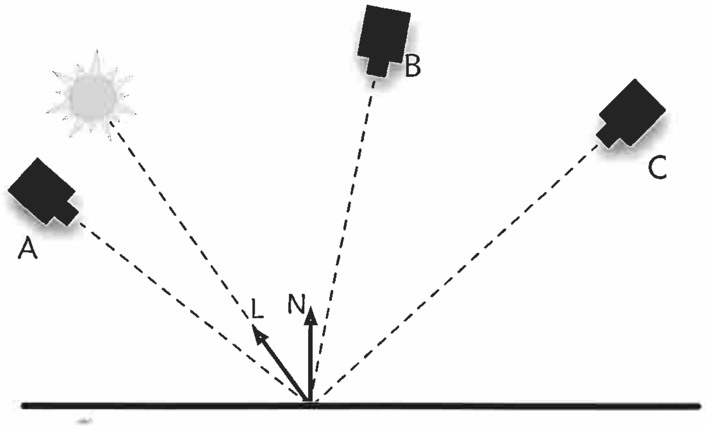
[5]

(d) In the following scene, 3 cameras (A, B and C) are looking at a point on a surface, illuminated by one light source. The illumination is calculated using the Phong lighting model.

(i) Which camera sees the brightest diffuse illumination?

(ii) Which camera sees the brightest specular illumination?

Justify your answer in each case.



[6]

(e) The following listing is the source code of a GLSL shader. Identify the type of shader and explain what it does and how it does it.

```
uniform float time;
varying vec4 vertColor;

void main() {
    float r = 0.5 * (1.0+sin( time / 10.0 )) ;
    float g = vertColor.z;
    float b = vertColor.y;
    gl_FragColor = vec4(r, g, b, 1.0);
}
```

[7]

4. Texturing

(a) Explain what a *map entity* is in terms of texture mapping. Give examples of commonly used map entities.

[6]

(b) What is the name of the coordinate system fixed to a specific object? Why is it useful for texture mapping?

[4]

(c) Imagine a scene with two identical looking bumpy objects, produced using bump mapping for the first and texture mapping for the second. For each of the following methods discuss whether it would cause the texture-mapped object to look less realistic than the bump-mapped one.

i. Moving the light
ii. Moving the object
iii. Moving the viewpoint

[6]

(d) Following is the code of a vertex shader that is part of a Processing application that performs texture mapping.

```
uniform mat4 transform;
uniform mat4 texMatrix;
attribute vec4 position;
attribute vec4 color;
attribute vec2 texCoord;
varying vec4 vertColor;
varying vec4 vertTexCoord;
void main() {
        gl_Position = transform * position;
        vertColor = color;
        vertTexCoord = texMatrix * vec4(texCoord, 1.0, 1.0);
}
```

(i) Explain what the code does.

[5]

(ii) Now consider the complementing fragment shader code below. Fill in the commands missing from the main() method so that texture mapping is performed correctly.

```
uniform sampler2D texture;
varying vec4 vertColor;
varying vec4 vertTexCoord;
void main() {
}
```

[4]

5. Rendering and Programming

(a) What is a varying variable in a GPU program? Give three examples of how you might use varying variables.

[6]

(b) Make a drawing that illustrates an example of a scene where Painter's algorithm would fail to represent correctly. Explain why.

[6]

(c) What is a depth buffer? How does it differ from a framebuffer? Explain how a depth buffer can be used in real-time graphics.

[5]

(d) For each of the following tasks state whether the use of a vertex or a fragment shader is more appropriate. Justify your answer.

i. Edge detection on images.
ii. Skinning.
iii. Phong shading.
iv. Displacement mapping.

[8]

END OF PAPER