Examiners' commentary 2018–2019

CO3355 Advanced graphics and animation – Zone A

General remarks

This year, the general standard of performance was slightly lower compared to last year, with about 70 percent of the candidates achieving a passing mark.

In terms of individual popularity, Question 2 was the most popular with about 71 per cent of the candidates attempting it, closely followed by Question 3 (67 per cent). Questions 1 and 4 were answered by 57 per cent of the candidates, while 48 per cent attempted Question 5.

Nevertheless, Question 5 attracted the best answers, followed by Questions 1 and 3, which were also answered very well on average. Answers to Questions 2 and 4 were less satisfactory.

As in previous years, most candidates showed appreciation of the relevant issues in questions, but many failed to provide clear and concise explanations. Practising such clarity in explanations is an area worth focusing on in revision and examination practise, and it should build upon the skills developed in earlier assignments and examinations.

Comments on specific questions

Question 1: Maths and transformations

a. Part (a) tested basic knowledge of related mathematics, and the majority of candidates answered this well. Some common issues in each case were:
   i. Performing calculations to extract the vector from the origin, though this was not necessary.
   ii. Adding the two vectors.
   iii. Not computing the vector magnitude properly.
b. This part gave a vector and asked candidates to calculate its coordinates after a counterclockwise rotation around the origin. Although this is a straightforward calculation, some candidates had problems with the rotation direction, calculating the opposite (clockwise) one. It is worth noting that the subject guide keeps it clockwise despite the reversal of the y axis. In addition, candidates are advised to verify their result visually to make sure it is the intended one.
c. Part (c) asked candidates to calculate the angle between two pairs of vectors and was generally answered well. Note that in (i) it sufficed to state that the angle was zero, without needing to show any calculations, as the two vectors were in the same direction.
d. This part examined candidate's understanding of basic geometric transformations. Candidates had to identify the components of a 2D matrix operation, making sure they respected the order of their application, that is from right to left. It sufficed to say that first the vector was translated by [-1 -1], then scaled by [2, 3] and, finally, translated by [1 1], while the effect of the combined operation was a scaling by [2, 3] about a fixed point ([1, 1]).

   While most candidates identified the transforms correctly, sometimes the order of their application was either incorrect or unclear. With respect to the purpose of the combined operation, very few candidates managed to provide a clear explanation. Some identified the similarity with a transformation stack and those who provided a clear justification for this earned full marks.

Question 2: Model to screen

a. This was a bookwork question and was mostly well answered.

b. Part (b) asked candidates to consider why we would consider using orthographic projection. Pointing out that parallel lines are kept parallel on screen and mentioning applications such as engineering and architecture would yield full marks.

c. The operation described in part (c) was culling. It is important as it reduces the number of rendered polygons and, thus, computational complexity.

d. Part (d) examined hidden surface rendering techniques.

   i. A good answer would explain the theory (bookwork – see subject guide p.35) with simple reference to the surfaces shown in Figure 1. However, the majority of candidates failed to provide a concise and clear explanation of this as many thought that the surface representation in the figure was problematic and had to be solved by the depth buffering algorithm.

   ii. A concise explanation of the z-buffer algorithm, using surfaces *A*, *B* and *C* would attract full marks. A common mistake here was the impression that z-buffering considers the depth of each surface or object (and not each pixel).

   iii. The image would look the same, regardless of the order in which the primitives are rendered, as z-buffer solves the visibility problem. Despite some good responses here, the majority of candidates did not manage to correctly justify their answer.

e. The final part of this question tested candidate's generic knowledge on established 3-dimensional test models. Any justified answer could work here, as long as it was supported and well explained. Characteristics might include: number of polygons (large to test detail or precision and/or speed or low for quick and easy testing of material, animation, textures or lighting, and so on), convex or concave curvature, manifold joints, imperfections to make it more realistic, multiple surfaces suitable for testing indirect/global illumination, and so on. Unfortunately, many candidates discussed generic testing processes instead of model objects and were awarded partial marks. Despite that, there were also some excellent answers.

Question 3: Graphics programming

a. This question asked for a quick contrast of vertex and fragment shaders and was mostly answered well.

b. Here, candidates had to name three types of operations/effects fragment shaders are suitable for. Surprisingly, this proved difficult, with many candidates failing to provide correct examples.

c. Part (c) examined physics vs graphics representations. It would suffice to point out that a graphics shape needs to be more detailed in order to satisfy the eye, while a physics one does not require this level of detail to make calculations more efficient. This was mostly well answered.

d. Part (d) examined three types of shapes: (i) triangles, (ii) triangle strips and (iii) triangle fans. Although most candidates seemed to understand their differences, many did not realise that the origin of the screen coordinate system is at the top left corner.

e. Part (e) tested basic GLSL programming.

i. A good answer here would state that the code loads a shader from file "fshader.glsl" creates a sphere, positioned in the centre of the frame and applies the shader on it.

ii. Here a very simple fragment shader needed to be specified, such as the following example:

```
void main() {
    gl_FragColor = vec4(1.0,0.0,0.0,1.0);
}
```

iii. An answer to this one could be using the following shader:

```
uniform float t;
void main() {
    gl_FragColor = vec4(abs(sin(t)),0.0,0.0,1.0);
}
```

and setting the variable inside the Processing program:

```
sh.set("t", millis() / 1000.0);
```

Question 4: Lighting and display

a. In this bookwork question about *tone mapping*, candidates had to briefly discuss how dynamic range affects image quality and how tone mapping tries to address that. Only very few provided a complete answer.

b. In part (b), candidates were asked to provide a brief definition of *flat shading*. This was mostly well-answered. See pp.67–68 of the subject guide.

c. Here, candidates had to craft an ideal situation where flat shading would be perfectly accurate. Any justified and well explained example could gain full marks. For example, flat shading could be accurate if there was infinite computing power, and could use "infinite" resolution for the model. Another example could be having the light source at infinity, so that the light vector was constant, while the viewer was at infinity and polygons represented actual surface, not an approximation. Although many candidates described situations where it would be somewhat adequate (such as when low complexity is a must), only a few did indeed craft situations where it would yield full representational accuracy.

d. This part of the question asked candidates to describe the steps involved when calculating the colour of a pixel using *Phong* shading. Most candidates had problems with this. Although most managed to provide some generic elements, it was rare to see clear steps and, especially, an informative diagram.

e. Very few candidates provided a good answer to part (e). Phong shading assumes a somewhat smooth basic shape. If that is not the case, it can make the edges look somewhat round instead of shading the faces in different light tones. That is because the vertex normals might not be representative of the surface itself, and interpolation would smoothen the edge, making it less realistic.

Question 5: Texturing

a. This part asked candidates to identify and briefly describe any three coordinate systems, as long as they are somehow involved in texture mapping. Although there were some very good answers, in most of them the connection with texture mapping was rather vague.

b. This was a bookwork (subject guide, p.75) question about texture mapping with a planar map shape, and was mostly well answered.

c. Part (c) asked for an example of a scene where it would make sense to choose to use *displacement mapping* over *bump mapping*. The answer should justify the need for increased detail at the cost of higher complexity. Moreover, it should demonstrate understanding of how the two methods differ. Most answers managed to provide these successfully.

d. Part (d) asked about the usefulness of randomness in computer graphics. Almost all candidates managed to achieve full marks.