Examiners' commentaries 2015–16

CO3355 Advanced graphics and animation – Zone A

General remarks

The general performance on this examination was high, with a significant number of strong papers; however, there were also some weaker ones. Around 9 out of 10 candidates attracted a passing mark, and a number of outstanding submissions obtained a mark of more than 90 per cent.

Although most candidates showed understanding of the relevant issues in the questions, not all managed to provide clear and concise enough explanations to earn full credit. Practising such clarity in explanations is an area worth focusing upon in revision and examination preparation, building upon the skills developed in earlier assignments and examinations.

Question 3 was the most popular with 90 per cent of candidates attempting it, closely followed by questions 2 (81 per cent) and 4 (74 per cent). Questions 5 and 1 were less popular, attracting 38 per cent and 16 per cent, respectively.

Interestingly, the most avoided questions were answered best. More specifically, answers to question 1 were mostly excellent, and on average question 5 was answered very well. These were followed by questions 4, 2 and 3 (in that order), revealing a negative correlation between the popularity and the marks gained in each question.

Comments on specific questions

Question 1

a. This part required knowledge of the major elements of a graphics system. Naming and describing them in one or two sentences was sufficient for full marks. This was answered well by most.

b. This question essentially required scaling by [s_x, s_y, s_z] followed by a translation by [t_x, t_y, t_z], which can be written as:

$$\begin{bmatrix} s_x & 0 & 0 & t_x \\ 0 & s_y & 0 & t_y \\ 0 & 0 & s_z & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Again, this was generally answered well.

c. This required candidates to: (i) construct a transformation matrix that rotates a 3D vector 30° about the y-axis and (ii) use this on a given vector. An appropriate transformation matrix can be shown as:

$$\begin{bmatrix} \cos 30^o & 0 & \sin 30^o \\ 0 & 1 & 0 \\ -\sin 30^o & 0 & \cos 30^o \end{bmatrix}$$

whereas the result of the transformation is [2.7321, 3, 0.7321].

Most derived the expression correctly and found no difficulties in its application.

d. Part (i) required an explanation of the transformation stack concept. However, some candidates misinterpreted the question and described what a **transformation** is in general, while others had problems presenting the stack functionality with clarity.

   Part (ii) used an example to test understanding of the transformation stack. Marks were awarded for use of pushMatrix() and popMatrix(), correct rotation and translation, making sure that the stack is empty after each iteration and for proper explanation. Unfortunately, only a few managed to do well here. Many had difficulties in performing the rotation and translation operations properly. Most did not manage to use the pushMatrix() and popMatrix() methods correctly.

Question 2

a. This part asked candidates to identify two ways to generate 3D models. Answers could include creation via a modelling package, a data capture device such as a laser scanner, 3D reconstruction from multiple images, and so on. Generally, candidates had some trouble identifying these. Some candidates described how 3D objects can be created in Processing i(PShape vs vertex-by-vertex), and this answer was also accepted. Others did not describe **generation**, but offered answers based on **representation** (analytic vs polygon-based). With respect to describing the term wireframe, despite this confusion, most answered well.

b. This question required knowledge of basic coordinate systems and was generally well-answered.

c. This part was about the technique of culling. Although the majority of candidates provided a satisfactory description of the technique, many failed to apply it to the cube example with clarity.

d. This question tested candidates' understanding of the Bresenham algorithm asking for: (i) its description for the simplest case as presented in the subject guide; (ii) the modifications required for a line of a different angle. The latter essentially required a change in the decision step, involving an increase of y by one (instead of x) and then deciding whether to draw $(x,y+1)$ or $(x+1,y+1)$. Most answered well in (i) but some had difficulties in (ii).

e. This required application of the Bresenham algorithm to a simple problem. Good answers included derivation of the line equation and then application of the algorithm to calculate the points accordingly. Only a few candidates followed this path, with the majority just drawing line points on the raster illustration provided.

Question 3

a. Candidates were required to identify the two fundamental properties that make triangles more appropriate for graphics calculations (i.e., them being (i) convex (not concave) and (ii) planar). Most candidates identified the second property but only a few did so with the first.

b. This part tested basic knowledge on the role of vertex and fragment shaders in the graphics pipeline and was mostly well-answered.

c. This question asked for the three types of light considered by the Phong reflection model, namely specular, diffuse and ambient light. These were identified successfully by almost all candidates.

d. Here a description of image-based lighting (IBL) was required, followed by an explanation of the steps involved in the process. Although most candidates described the generic characteristics of IBL in a satisfactory way, only a few succeeded in providing a detailed explanation of the steps of the algorithm.

e. This part asked for the functionality of the provided shader code together with a description of its main steps. The code is a fragment shader that implements the diffuse lighting equation per fragment. It first normalises the direction from the vertex to the light, then does the same for the transformed normal and afterwards calculates the dot product of the light direction and the vertex normal (equivalent to calculating the cosine of the angle between them). The result is the light intensity which can then be multiplied by the vertex colour (it has to be converted to a vector first) to get the lit colour of the fragment. The majority of candidates managed to describe the code well, but many failed to demonstrate understanding of the overall high-level functionality of the program.

Question 4

a. This part asked for the added value of texturing over plain shading and generally attracted competent answers. Nevertheless, many candidates focused on discussing two different approaches to texturing (namely, texture mapping and procedural texturing). Although correct, these descriptions were unnecessary and out of the scope of the question.

b. This only required a very short description and was generally answered well.

c. Part (i) required a description of the bump mapping technique. The majority of candidates answered well, mentioning that it does not alter geometry but many did not explain how depth is simulated.

   Part (ii) was not attempted by a number of candidates, those who did so generally provided good answers.

   Part (iii) required some intuition regarding bump and displacement mapping, in order to select the right method for each case and explain why the choice was made. The vast majority indeed made the right choices (bump mapping for orange skin and displacement mapping for waves), but some failed to explain the reasons behind their choice, especially with respect to complexity.

d. The concept of environment mapping seemed to confuse some candidates, as only very few managed to provide a good answer here.

Question 5

a. This part asked for the difference between collision and graphics shapes and was generally answered well.

b. A description of the basic categories of collision shapes was required. Overall, it was answered well, although quite a few missed the fact that, compared to the others, meshes provide a higher level of detail but also entail a higher degree of complexity.

c. This part asked for an explanation of the term restitution. Despite a few very good answers, most candidates faced difficulties here.

d. This question provided a fragment of Processing code and asked for a detailed line-by-line explanation of its functionality. The code basically creates a sphere rigid object which is positioned in a physics/gravity environment. Consequently, the sphere starts falling. Candidates performed reasonably well here.

Examiners’ commentaries 2015–16

CO3355 Advanced graphics and animation – Zone B

General remarks

The general performance on this examination was rather poor, as only about half of the candidates managed to achieve a passing mark. However, there were also a number of excellent submissions that obtained marks higher than 80 per cent.

In terms of individual popularity, questions 3 and 4 were chosen by 72 per cent of candidates, closely followed by questions 1 (67 per cent) and 2 (63 per cent). Question 5 was considerably less popular, chosen by only 20 per cent.

In terms of marks, question 1 attracted the best answers, followed by questions 2, 1 and 3 (in that order). Question 5 attracted the lowest marks.

Comments on specific questions

Question 1

a. This part asked for the identification and brief description of the five stages involved in a typical graphics pipeline. Although the vast majority of candidates had no problems in identifying the stages, not all managed to demonstrate understanding of what these stages represent.

b. Part (i) asked for a matrix expression such as the following:

$$\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$

Many candidates found this difficult.

Part (ii) covered rotation about each of the three axes; expressions such as the following can be used:

$$z\text{-axis:}\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$y\text{-axis:}\begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$x\text{-axis:}\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

As with (i), candidates faced more trouble than expected in forming these correctly.

c. Part (i) asked for an expression such as:

$$\begin{bmatrix} 0.3 & 0 & 0 & 0 \\ 0 & 0.3 & 0 & 0 \\ 0 & 0 & 0.3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Only a few candidates managed to derive a similar expression. There were, unfortunately, a number of candidates who attempted to multiply a scalar with a 3x1 or 3x3 matrix, which indicates underlying weakness in basic linear algebra and mathematics.

For part (ii), the result of this operation is [1.8, 1.8, 0.9]T.

d. Part (i) required an explanation of the transformation stack concept. However, some candidates misinterpreted the question and described what a **transformation** is in general, while others had problems presenting the stack functionality with clarity.

   Part (ii) used an example to test understanding of the transformation stack. Marks were awarded for use of pushMatrix() and popMatrix(), correct rotation and translation, making sure that the stack is empty after each iteration and for proper explanation. Unfortunately, only a few managed to do well here as many had difficulties in performing the rotation and translation operations properly. Also most did not manage to use the pushMatrix() and popMatrix() methods correctly.

Question 2

a. This question required knowledge of basic coordinate systems and was generally well-answered, though some candidates had problems in providing clear descriptions.
b. This part asked for a relatively straightforward explanation of a simple graphics algorithm – the *z-buffer* algorithm. Despite some excellent answers, most candidates had difficulties here.
c. Candidates were required to identify methods for filling a closed polygonal region with colour, such as *recursive seed fill*, *ordered seed fill* and *scanline fill*. This was mostly well-answered.
d. Candidates' understanding of the Bresenham algorithm was tested, asking for: (i) its description for the simplest case as presented in the subject guide; (ii) the modifications required for a line of a different angle. The latter essentially required a change in the decision step, involving an increase of y by one (instead of x) and then deciding whether to draw $(x,y+1)$ or $(x-1,y+1)$. Most answered well in (i) but had some difficulties with (ii).
e. This required application of the Bresenham algorithm to a simple problem. Good answers included derivation of the line equation and then application of the algorithm to calculate the points accordingly. Only a few candidates followed this path, with the majority just drawing line points on the raster illustration provided.

Question 3

a. This part tested knowledge of triangle strips and triangle fans. In general, this was answered relatively well. However, many candidates did not identify how the triangles were connected in each case (i.e. which vertices are being shared).
b. This was quite straightforward and was answered well. Nevertheless, there were some occasions where the number of colours was calculated incorrectly. For instance, in the case of 24 bits, some calculated it as $3*255=765$ or even $2^3=8$ colours instead of $2^8x2^8x2^8=2^{24}$ (=16,777,216) colours.
c. Here a description of image-based lighting was required, followed by an explanation of the steps of the process. Most had difficulties, with what seemed like rather improvised answers, showing some confusion.

d. This part asked for the functionality of the provided shader code together with a description of its main steps. The code is a fragment shader that implements the diffuse lighting equation per fragment. It first normalises the direction from the vertex to the light, then does the same for the transformed normal and afterwards calculates the dot product of the light direction and the vertex normal (equivalent to calculating the cosine of the angle between them). The result is the light intensity which can then be multiplied by the vertex colour (it has to be converted to a vector first) to obtain the lit colour of the fragment. The majority of candidates managed to describe the code well, but many failed to demonstrate understanding of the overall high-level functionality of the program.

Question 4

a. This part asked candidates to identify differences between procedural texturing and texture mapping. It was generally answered well.

b. This question required identification and short description of three types of map shapes and the coordinate systems used to implement them. Almost all candidates named map shapes and corresponding coordinate systems correctly. Some had difficulties explaining how mapping takes place in each case.

c. Part (i) required a description of the technique of displacement mapping. Although the majority of candidates described the method in a satisfactory manner, a few seemed to be confused, especially with respect to its effect on the number of polygons.

   Part (ii) required some intuition regarding bump and displacement mapping in order for candidates to select the right method for each case and explain their choice. The vast majority indeed made the right choices (bump mapping for grass as seen from a satellite but displacement mapping when seen from the ground). However, some failed to explain the reasons behind their choice, especially with respect to complexity. Also many did not manage to explain how bump mapping simulates depth.

d. The concept of environment mapping seemed to confuse candidates, as only very few managed to provide good answers and gain full marks.

Question 5

a. Candidates needed to explain some basic terms in the field of physics modelling. Only a few candidates managed to provide good answers here.

b. This part required some basic understanding of the BRigid library. Classes that construct primitive collision shapes include BBox, BSphere and BPlane, whereas compound shapes can be defined with the BCompound class. This proved challenging.

c. This question was about types of friction and how these are treated by physics engines. BRigid has only one parameter for handling friction and this is passed to the setFriction() method. Again, this proved challenging, as only a few named the friction types correctly, and the BRigid command was not identified by anyone.

d. This part provided a fragment of Processing code and asked for a detailed line-by-line explanation of its functionality. The code basically creates a sphere rigid object which is positioned in a physics/gravity environment. Consequently, the sphere starts falling. Although candidates did reasonably well in the line-by-line description, they mostly failed to show they understood the functionality of the program.