Coursework commentaries 2016–17

CO3355 Advanced graphics and animation – Coursework assignment 1

General remarks

This coursework assignment consisted of four parts and examined hierarchical scene modelling as well as transform stacks. Part A required candidates to devise a scene and illustrate it using a hand-drawn sketch. Part B asked candidates to implement the scene with Processing, making sure that it incorporated interactivity. In Part C, candidates were asked to incorporate GLSL shaders to render the scene and develop basic effects. Finally, Part D required the implementation of a deformation effect in the vertex shader, this in turn provided an opportunity for improvisation to demonstrate what other distortion effects could be produced.

The general standard of the coursework was competent and included some excellent submissions. Most students answered the first two parts without any problems. However, many seemed to face difficulties with the GLSL implementations covered in Parts C and D.

Moreover, some submissions showed an imbalance between implementation and reporting, although to a lesser degree than previous years. While many programming attempts were excellent, they were too often not accompanied with reports of similar quality.

It is important that each report:

- answers each question clearly and provides adequate description of the implementation
- exposes the problems faced
- justifies design decisions made
- provides evidence of the results achieved (such as by using screenshots from multiple viewpoints)
- assesses and interprets the results appropriately to demonstrate their understanding and intuition

However, it is also worth keeping in mind that concision is important. More often than not a few well-focused paragraphs of discussion, together with illustrations, should be sufficient for each question.

Comments on specific questions

Part A

This question asked students to create a hand-drawn sketch of a dynamic scene for which hierarchical modelling would be appropriate. Students had to expose and describe the characteristics that would make such modelling suitable and to present the transformations that would take place when the scene was animated.

The most important aspect here was to demonstrate why the scene would benefit from hierarchical modelling. The movement of the parts/objects constituting the scene should present some form of interrelated structure.

Articulated models, consisting of rigid parts connected by joints (such as the human body) would constitute good examples. Unfortunately, some students chose scenes that consisted of objects that moved without any relationship among them. There were also many examples of well selected scenes, such as robotic arms, vehicles, solar systems, a wall clock, and so on.

The quality of drawing was not key to the assessment, as long as it conveyed the necessary information. The vast majority of answers were good in that respect.

Part B

This question required students to implement their scene in Processing, incorporating interactivity and supporting camera navigation using an appropriate library.

As expected, the quality of answers was dependent on the quality of the scene selection made in Part A. Good answers permitted the user to control aspects of the movement in real time (this included aspects such as speed, specific movements among the components, and so on) Some students did an excellent job in this respect, for example, providing the ability to turn a steering wheel, or control the rotation speed of the wheels of a vehicle.

There were a small number of students who had difficulties in integrating Peasycam, although most found it to be quite straightforward (simply by including the import and use of the corresponding Processing library).

Part C

Part C required students to write appropriate vertex and fragment GLSL shaders to render the scene and then modify the vertex shader to change the size of the object it was applied to.

Most candidates had no problem in incorporating and using the GLSL shaders. With respect to the shrink/grow effect, the goal was to change the position of each vertex, moving it closer to or further away from its centre (effectively shrinking or growing the object respectively). This was more complicated because Processing passes on to GLSL view coordinates instead of the Model ones. However, many candidates solved this problem by passing the centre of the object as a parameter to the shader or by using the normal of the vertex itself to determine the proper direction.

A commonly observed problem was when the effect was applied to objects created in the previous part. Often this resulted in the loss of proper connectivity among the parts as they were animated. Modifying the size of objects disconnected by nature, such as planet systems, was fine. However, in cases of articulated models, extra care was needed to ensure that the object parts did not lose connectivity when their size was modified.

Part D

The final section asked students to implement an appropriate vertex shader that performed object deformation. Essentially this question asked students to apply a 'damage' effect to the various objects according to a mathematical function of their choice. They were then asked to improvise thereby producing further effects.

Good attempts implemented object deformation by simply translating the vertices using any mathematical function. Object deformation could start from a plain and simple effect, such as a sinusoidal function, so that the deformation was made apparent and then built upon to achieve more sophisticated effects.

This was the least popular question and was not attempted in nearly half of the coursework submissions. However, of those students who did try, many produced sophisticated results.

Coursework commentaries 2016–17

CO3355 Advanced graphics and animation – Coursework assignment 2

General remarks

This assignment was comprised of two parts that were divided into two questions each. In the first part, students were asked to implement two lighting models using GLSL shaders, while the second part focused on procedural methods to create visual patterns for texturing.

Overall, and on par with the first assignment, the performance was satisfactory in terms of implementation and included a pleasing number of excellent submissions. However, there were many cases were the report fell short of the quality seen in the implementation. There was an increase in the number of reports lacking even a basic structure. Others were too short, omitted essential detail on implementation decisions and lacked critical evaluation. Concise and well-focused reporting is an important aspect of assignment evaluation and should be treated with due care and attention.

Comments on specific questions

Part A

Question 1

This question required students to write appropriate vertex and fragment GLSL shaders in order to implement Phong and Blinn–Phong reflection models. These were then meant to be incorporated into a Processing program providing the user with the ability to adjust the model parameters in real time. Moreover, students were asked to experiment with the parameters and demonstrate the visual results on various shapes.

In order to carry out this task, students had to understand how the specific reflection models work, especially Blinn–Phong, which is not included in the subject guide and is slightly more sophisticated. Good answers included implementation of both models, experimentation with their parameters and a demonstration of the results with screenshots from multiple viewpoints accompanied by intuitive commentary.

Although some had difficulties understanding the Blinn–Phong model, the majority of candidates provided satisfactory answers. The discussion board on the VLE provides a useful forum to help clear doubts, request Further reading, and so on.

Question 2

Here, students were asked to add a second light in the vertex shader and explore how multiple light sources could be incorporated into the scene. A neat solution would entail the use of a for loop in the vertex shader, in order to iterate through the light sources thus combining their effects on each fragment of the object.

Although this process is described briefly in the subject guide, the provision for a second light source proved challenging for many students. This question

was the least popular of the assignment and was only attempted in around half of the submissions received.

Part B

Question 1

This question required students to write code in order to generate a grid of circles. The size and number of the circles should be adjustable by the user and should incorporate lights from Part A.

Many students found it difficult to produce the circle grid. A possible solution to this would be to create a square grid and then, for the fragment processed at each point, assess whether it falls within a certain (and adjustable) radius from the centre of the corresponding square region. Whether that was the case or not would determine the colour of the fragment, effectively creating a circle within each region.

Unfortunately, some students who created the grid successfully failed to map it on an object, thus losing significant marks as this was an important part of the assessment.

Question 2

Question 2 investigated the use of noise functions, such as Perlin, Simplex and Worley, in order to make procedural patterns more interesting. Students had the opportunity to improvise with, and generate interesting effects.

This was the second least popular question and was attempted in about two thirds of the submissions. The question was open ended and left space for improvisation, with students not bound to use a specific noise generator, but rather experiment with other functions at will. The results varied greatly with respect to quality and while some effects were rather simplistic, others were very creative and impressed the examiners. Even the most rudimentary answers attracted some marks. Remember, it is always a good idea to try every question in a coursework assignment.