COM4503: 3D Computer Graphics: Assignment 1 (40%)

Dr Steve Maddock

Deadline: 3pm, Wednesday 11 December

1. Introduction

The assignment will involve using modern OpenGL to render a scene. Scene graphs are required in the modelling process and animation controls are required for hierarchical models.

2. Learning outcomes

After completing this assignment, you will be able to:

- Use data structures and mathematics in representing and manipulating 3D objects
- Produce interactive software that makes use of a graphics API

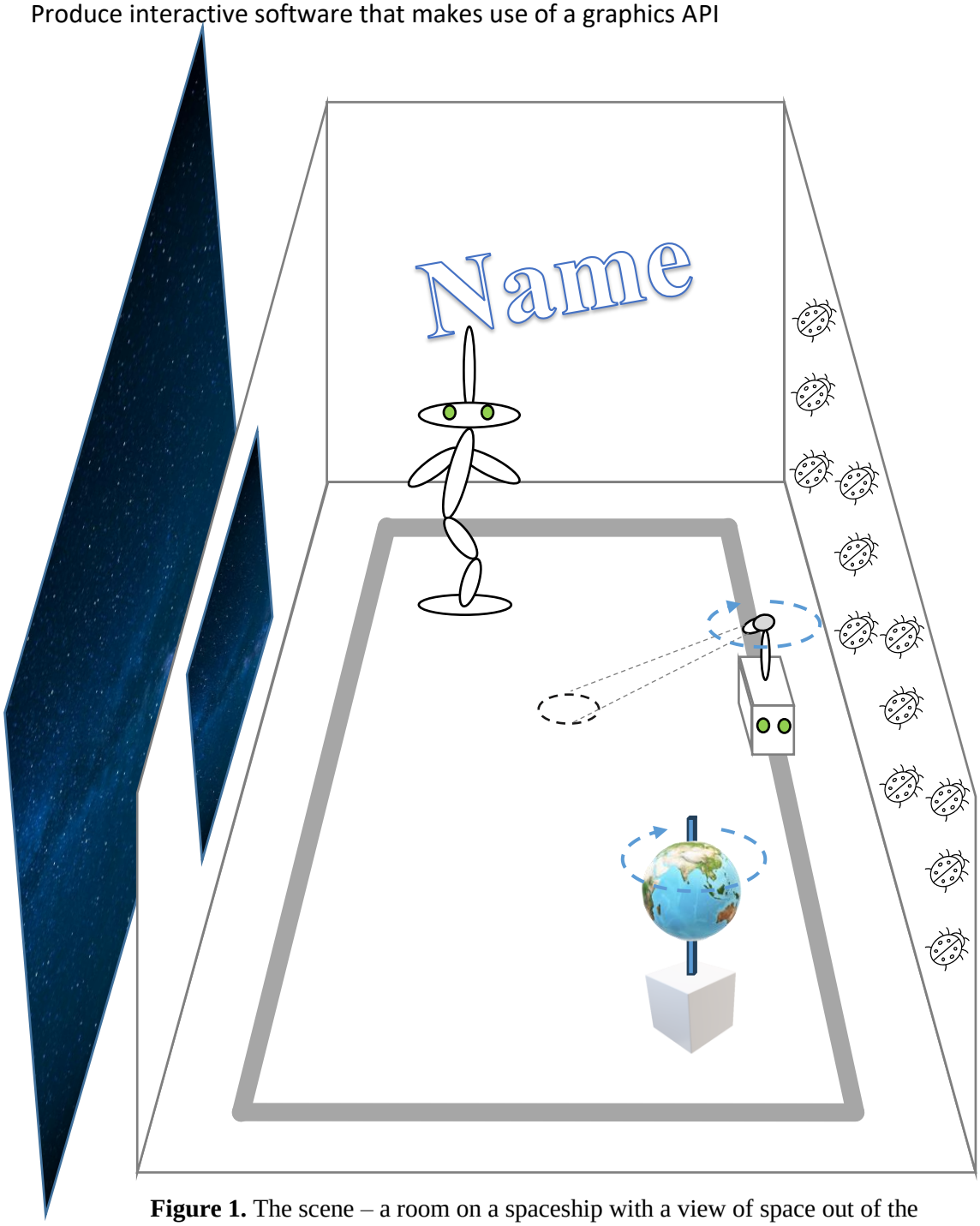

Figure 1. The scene – a room on a spaceship with a view of space out of the window. The dashed blue arrows show the rotation of the spotlight on top of the small robot (robot 2) and the rotation of the globe.

3. Requirements

Figure 1 shows a scene set in a room on a spaceship. The whole scene should be modelled using transformed flat planes, cubes and spheres. The view through the window is space (i.e. stars, etc).

You must satisfy all the following requirements.

3.1 The room

- In Figure 1, the room is made of three walls and a floor. The wall nearest to the viewer is not modelled. The room size is your choice.
- The left wall has a window in it.
- The back wall has a texture added to it. This texture includes a large shiny text message of a name. You should use part of your name for this, e.g. for me, I would choose 'Steve'. A diffuse and a specular map will be required. The specular map will make the name part appear shiny in contrast to the rest of the wall. You must create the diffuse and specular maps. These textures should be stored in files called diffuse_name.jpg and specular_name.jpg in the assets/textures folder, where your chosen name part is used in the filename so they are easy to find and check, e.g. diffuse_steve.jpg and specular_steve.jpg.
- The right wall has a repeating texture added to it. Here, a picture of a bug is used (as illustrated in Figure 2, which repeats four times in one direction and three times in the other). You should make your own choice of texture. It must repeat over the wall, i.e. the wall is not a single texture image containing multiple pictures.

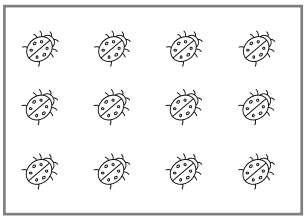

Figure 2. A bug texture in a repeating pattern.

- The floor should be textured. You choose a texture. However, it should also feature a series of thick lines in any colour you choose. This sequence of lines is the path a robot needs to follow.
- A ceiling is not shown but should be added to make the scene look better. This should be textured too. You choose the texture.

3.2 The window and the view

- The left wall in the room is a large window.
- An outside scene of space (i.e. stars, etc) can be seen through the window. Consider how you might do this: Should the scene be a texture map pasted onto the wall to look like a fake window and a scene? Or should there be a hole in the wall for the window and a texture map is pasted onto a plane that is a certain distance outside the window (as illustrated in Figure 1)? Should the window have a frame? Or should the window be the size of the whole wall with a simple small frame around the edges? Should a box of textures be added outside the window so a texture can be seen at all angles when looking out of the window? Or should a skybox be used that is outside the whole room rather than a separate plane with a texture map on it?
- Depending on the approach you choose, how does it look when the camera moves position in the room when looking out of the window? (Is it possible to stand in the room and always see a view of space outside through the window?)
- The scene outside the window should change over time whilst the program is running, e.g. the stars and planets move.
- The quality of what you produce for this part of the scene will be part of the marking. Some alternatives indicated above are more advanced than others. You must choose what to try.

3.3 The robots

- There are two robots, both modelled as scene graph hierarchies. Both are made up of transformed spheres and/or cubes.
- Robot 1, the dancing robot, is at the back left of the room in Figure 1.
- Robot 2, the small robot, is on the track on the floor at the right side of the room in Figure 1.
- Robot 1 should feature a base, three parts that form the leg and body, two arms and a head, as illustrated in Figure 1. Each of the parts can

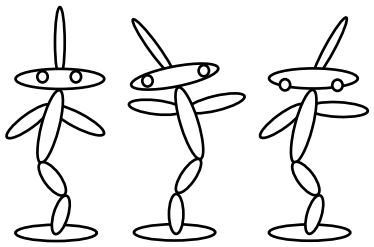

Figure 3. Some poses of robot 1, the dancing robot.

articulate to make the robot dance, as illustrated in Figure 3, which shows three poses.

- The head of robot 1 should be your own design. It must have a minimum of 4 pieces. Figure 4 gives a few possible designs. Again, this is a chance to show some creativity.

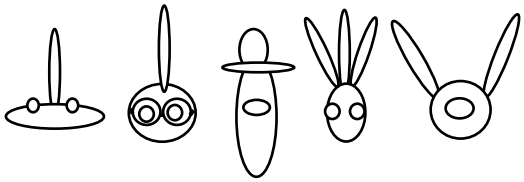

Figure 4. Some possible head designs for robot 1.

- Robot 2, the small robot, has a body, two eyes and an antenna. The antenna has a spotlight attached to it which is made of two transformed spheres, one for a small bulb and one for its casing (alternatively called its holder) – these are shaded as white and grey, respectively, in Figure 1. You can choose to vary the look of the spotlight, but it must feature a bulb and a casing for the bulb. The two together indicate which direction the spotlight is pointing in and thus where the pool of light from the spotlight should appear in the scene.
- Robot 2, the small robot, continually follows the path of grey lines on the floor, with its two eyes showing the forward direction.
- When robot 2 reaches a corner in the path, it changes direction to follow the new line.
- The hierarchy and associated transformations are more important than the quality of the individual pieces in each of the robot hierarchies. Transformed spheres and cubes must be used. I want you to demonstrate that you understand transformations and a scene graph hierarchy.
- You must texture-map both robots. You must decide on which textures to use. You cannot use the same texture(s) on each robot.
- I'll be looking for a little creativity in the animation for each robot. For example, how should robot 1 perform its dance? How should its parts move with respect to each other? Perhaps some cartoon effects could be used, e.g. head parts appearing to enlarge or move position for parts of the dance? For robot 2, when it reaches a corner, how should it turn? Should it lean a little to one side? Or spin in some way?

3.4 Spotlight

- Robot 2 has a bright white spotlight attached to the top of its antenna. The dashed lines in Figure 1 are to illustrate where this spotlight is. These dashed lines would not be seen in the real effect!!
- The spotlight continuously rotates around the top of the antenna whilst robot 2 moves. Thus, different parts of the scene will be illuminated by the spotlight as robot 2 moves. (The spotlight stops rotating when robot 2 stops moving.)
- This is an advanced requirement as you are responsible for working out how to implement a spotlight effect.

3.5 The globe

- This is made of transformed spheres (one for the globe and one for the central axis) and a transformed cube (for the pedestal).
- The globe continually rotates about the central axis.
- The texture map for the globe must look like planet Earth or a cartoon version of it. You will need to find a texture to show this or draw a cartoon version of it yourself.
- The stand for the globe also needs to be texture mapped. Try to be creative with this.

3.6 General illumination

- The scene should be illuminated with a general world light which can be positioned anywhere in the scene.
- This general world light will illuminate all parts of the scene.
- When you switch off the general light (using an interface option – see the next section), the effects of the spotlight will be much clearer.
- (You can include more than one general light if you wish.)
- You do NOT have to do shadows. Do not worry about shadow effects. (The general world light will illuminate all polygons with a normal pointing towards it and the spotlight will illuminate all objects in the direction it is pointing in which are inside its spotlight area as there are no shadow effects to show light not reaching particular points.)

3.7 User interface

- A user-controlled camera should be positioned in the scene. Use the camera that was given in the

tutorial material – the mouse can be used to change the direction the camera is pointing in, and the keys can be used to move about. Do not change the key mappings from the ones in the tutorial. If you change the key mappings, it will make it difficult to mark. It doesn't matter that the camera can move and see outside the room.

- It should be possible to turn the general light on and off (or, more creatively, dim it, i.e. reduce the intensity) from the interface.
- It should be possible to turn the spotlight (lamp bulb) on and off (or, more creatively, dim it, i.e. reduce the intensity) from the interface.
- Robot 1's dancing animation should start when robot 2 is 'near' to it and stop when robot 2 is not 'near' to it. You decide how near it needs to be. You will need to implement a proximity test (based on distance) to do this.
- You should also add buttons on the interface, so you can start and stop robot 1's dancing animation whenever you want to, irrespective of the proximity test. This will be useful for testing purposes rather than waiting for robot 2 to traverse all the way round the room!!
- It should be possible to stop and start robot 2's movement - the spotlight stops rotating when robot 2 stops moving. You need to implement interface controls to do this.
- There is no need to stop and start the globe rotation. This just continually rotates.

3.8 Animation

- Some of the animations are not straightforward and you may decide not to do them, although that would affect your marks for animation.
- Use Euler angles for the animation. Do not consider using quaternions, as this is beyond the requirements for this assignment.

4. Deliverables

- You should submit a zip file containing a copy of your program code (and any other necessary resources, e.g. image files for the textures and a readme.txt file that describes everything) via Blackboard – this can be done via the link to the assignment handout.
- You should submit whatever you have done, even if you have not completed all the requirements – for example, you might have produced a model of the scene but not done the animation. If you submit nothing, you cannot receive any marks.
- **The program MUST compile and run from the command window on a Windows PC or the terminal window on a Mac**. You should assume that the jogl environment (and paths) has already been set up (on my machine), so you do not have to include this as part of what you hand in. I won't install 'YetAnotherIDE' to make your program work; I want to compile and run the program from a command (or terminal) window using the standard javac and java commands.
- You must include appropriate comments to identify parts of the code that you wrote, e.g. /* I declare that this code is my own work */ /* Author <insert your name here> <insert your email address here> */. This could be done around major chunks of code and/or at the start of a class to identify the main changes you made.
- You can make use of all the code that I have given you in the tutorial material. However, use your comments to state which bits/chunks/files are new.
- The body of the Blackboard submission message should state that the work you have handed in is your own in addition to the code that was supplied in the tutorial material.
- The name of the main class in your program should be **Spacecraft**. That way it is easy for me to compile and run the program. (In previous years, where this has been ignored, I have wasted time for some handins trying to work out which was the main class to use.)
- *Optional*: You might like to make a short video of your animation. If you do so, *DO NOT* include this in the handin as it will be too big for Blackboard to handle – we tried using Blackboard for this in the past and it crashed the system!! Instead, put the animation on YouTube or your personal website and give the URL of the animation in a readme.txt file. Indeed, if you are thinking of a career in the graphics/games industry, then you should be adding such animation pieces to your personal website (your digital portfolio) to show off what you are capable of.

5. Marking

I will check that the program meets the requirements listed above. The program **must** compile and do some part of the work requested

even if it is not complete. Your program will be run and exercised thoroughly.

In considering the requirements, four aspects will be considered (including the quality of the work for each aspect):

- (29 marks) Modelling the scene: Each robot must be a hierarchical model. Is there a spotlight model attached to robot 2's antenna? Is there a globe? How is the room modelled? (Consider drawing scene graphs for the scene before starting to program.)
- (28 marks) Texturing: Use of texture mapping in the scene, e.g. basic texture mapping, use of diffuse and specular textures, extra texturing effects, e.g. the changing view through the window.
- (15 marks) Lighting and interface controls: all lights should behave correctly such that their effect is seen on the scene. All necessary interface controls, as described in the above specification, should be included.
- (28 marks) Robot 1 and 2 animations. Spotlight animation. Animation for the globe. Is all the animation smooth rather than jerky?

6. Unfair means

- The School's student handbooks (UG and PGT) give detailed information on the topic of unfair means and what happens if unfair means is used.
- Some students in previous years have placed solutions of their assignments on their personal world-readable websites/GitHub – where possible, they have been asked to take these down. Be careful you are not attracted to these, as using any of their code would be regarded as use of unfair means – this has happened in previous years and students have failed the module because of doing this.

7. Late submission

- Standard School rules will be applied if the work is handed in late – see UG and PGT handbooks.

Links to handbooks

UG:
https://sites.google.com/sheffield.ac.uk/comughandbook

PGT:
https://sites.google.com/sheffield.ac.uk/compgtstudenthandbook