M30242 Graphics and Computer Vision 2021-22 Coursework

Part 2

UP901354

Word count: 966 (excluding headings, image labels, etc.)

Contents

Design implementations

This animation was designed to meet the specifications of the coursework assignment document – this includes functionality, object dimensions, textures, etc.

The satellite was drawn with the following dimensions:

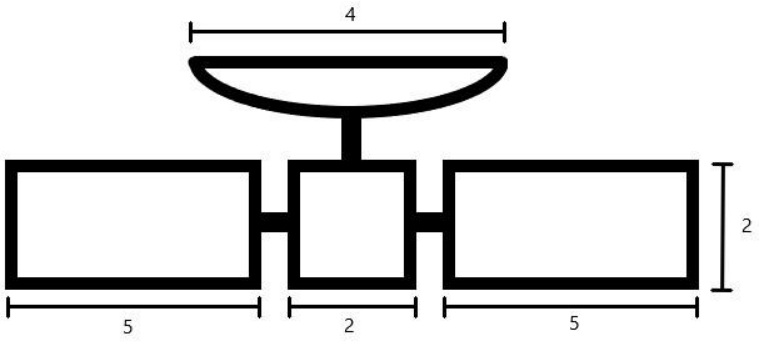

Figure 1 – Satellite dimensions

This was achieved by scaling objects – for example, the cube is drawn with dimensions 2x2x2, which is perfect for the main satellite but not for the solar panels. They are scaled by 1, 0.1 and 2.5 (x, y, z).

I modelled the scene to the specification using 3D modelling software (Blender) – this served as a concept image and helped me to visualise the scene.

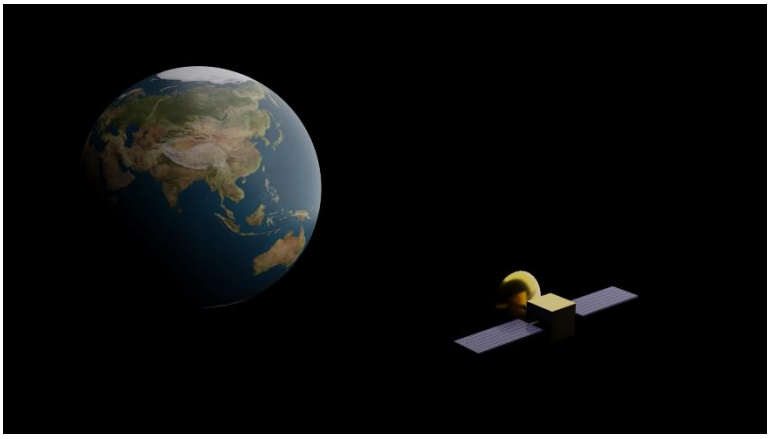

Figure 2 – Concept modelled in 3D modelling software

Drawing cubes

There are a total of 3 cubes drawn for this scene – all of these are initially drawn using the same vertex coordinates, vertex indices, and normal data. This will reduce the time it takes to set up the buffers and while, with a program of this size it may not make a difference, for a larger program (where hundreds or thousands of cubes are being drawn) it could save a substantial amount of computing power and time.

The cubes are scaled when drawn to meet the dimensions specified.

Drawing cylinders

For a cylinder of height, *h*, it is drawn by calculating the vertex coordinates for 2 circles, one at height h/2 and one at -h/2 and then drawing a triangle strip between these. As, in this animation the ends of the cylinders are not shown, the normal data is not calculated and the circles are not drawn. The vertex coordinates for the triangle strip are calculated by taking both lists of vertex coordinates (for the circles) and combining them. For example, for the 2 lists, circle1[n] and circle2[n], the strip list would look like: [circle1[1], circle1[2], circle1[3], circle2[1], circle2[2], circle2[3], …].

For the cylinders connecting both solar panels to the satellite, one object is drawn at the centre of the satellite. The length of each cylinder should be 0.7 so, after we account for the size of the satellite, the one cylinder should be 3.4 in length.

Both drawing one cylinder connecting the solar panels and not drawing the circles at the ends of the cylinders saves on the number of objects being drawn and therefore reduces the computing power needed (again, it won't affect this animation much, if at all, but if it were more complex this would help).

Drawing spheres

The x, y and z vertex values of the sphere are calculated with the following loop, with m representing rows, n representing columns and r representing radius:

```
for (var i=0; i <= m; i++) {
    for (var j=0; j <=n; j++){
        //calculate x, y and z here

        x=r * (Math.sin(i*Math.PI/m)) * (Math.cos(2*j*Math.PI/n));

        y=r * Math.cos(i*Math.PI/m);

        z=r * Math.sin(i*Math.PI/m)* Math.sin(2*j*Math.PI/n);


        sphereVertexPosition.push(x);
        sphereVertexPosition.push(y);
        sphereVertexPosition.push(z);


        // 1st row of vertices (i=0)
        // 2nd row of vertices (i=1)
        // etc...
        }
    }
```

Figure 3 – Sphere vertex calculation

Animation

When the program is initialised, variables are set for orbit radius, orbit speed, etc. The orbit radius and orbit speed are used to calculate the angle of the circle that represents the orbit at each time frame, and from this the coordinates.

The angle and radius are also used to calculate positions for the other parts of the satellite – for example, the solar panels have an orbit of 0.2 greater than the main satellite and the dish has one of 1.5 smaller.

All objects that make up the satellite are translated around at their respective orbits but are also rotated in the opposite direction by the main cubes angle – this ensures the satellite faces earth throughout the entire orbit and that the parts of the satellite stay in line.

Lighting

I have used the Phong lighting model in combination with a directional light coming from the top right of the screen (see figure 6 for an example). I tested using only a directional light but found that the left side of the earth was slightly too dark – the small amount of ambient lighting helps the viewer see the left side of the earth while not drowning out the directional light. I settled on the Phong lighting model as it provides satisfactory results while also being simple to calculate. I set the directional light at a large distance from the objects.

I also tried making the directional light slightly orange to mimic the sun, however I decided that the cleaner look of white light both looks more realistic and shows the colours of the earth better.

Difficulties

I had difficulty with setting up the directional lighting – I found that it was locked to the object rotation. This was resolved after changing the lighting coordinates to camera coordinates.

Testing

As part of testing object sizes, textures, etc., objects were drawn without any translation and with ambient lighting.

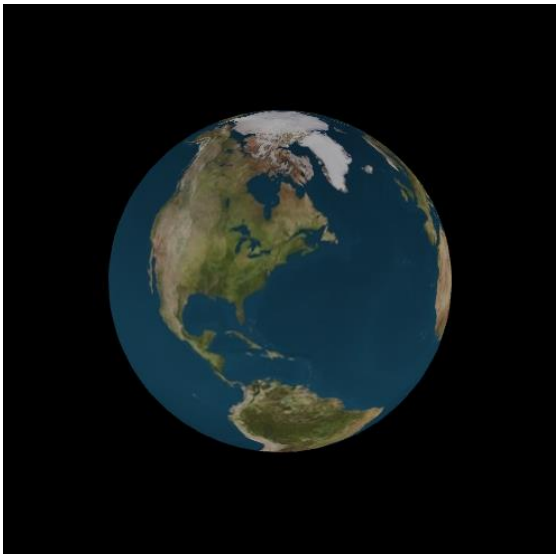

Figure 4 – Earth model

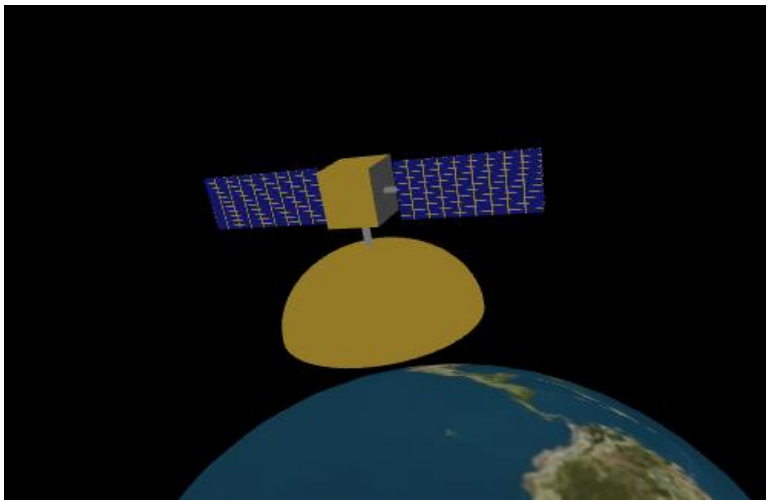

Figure 5 – Satellite model

After this, the lighting could be set up so that objects didn't appear flat (such as in figure 5 above).

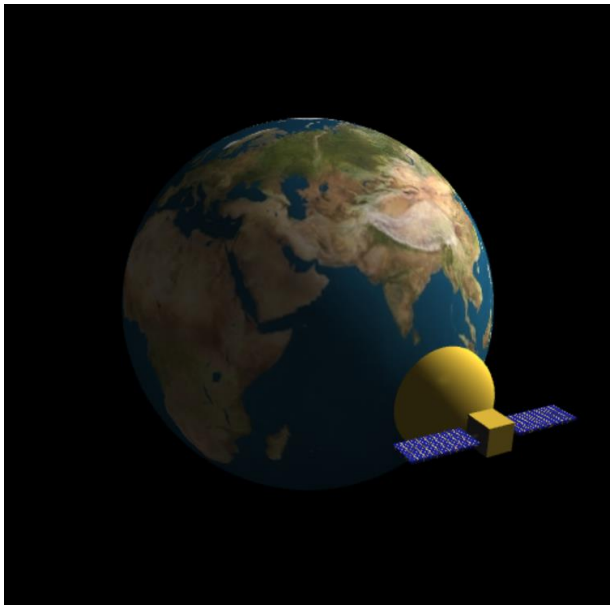

Figure 6 – Directional Lighting

For changing the orbit radius, after testing I realised I would need to set a limit so that the satellite didn't collide with the earth or go too far out of the cameras view. I also set limits on the speed based on what I thought looked sensible.

Evaluation

I would have liked to have used a hierarchical object structure, however for the small scope of the project (in terms of number of objects) it is not needed and the method I have used works well. Perhaps if this animation were to be expanded (for example, more satellites or planets added) then having a hierarchical structure to the objects would be beneficial.

I think adding shadows would increase the realism of the animation. The shadow of the earth looks like it would be fairly easy to add and the shadow of the satellite could probably be skipped, as it would look strange casting a large shadow onto the earth (due to the exaggerated size of the satellite).

Something else I would like to explore is importing models into WebGL to be used in the animation – I feel creating the models in commercial 3D software would save time.

Overall, I am happy with the quality of the result and it has helped me to recognise the advantages that WebGL has over other (less widely available) methods for creating animations.