GRAPHICS PIPELINE

COS3712 2017-10

1.1 Explain the difference between immediate mode graphics and retained mode graphics. [4]

Immediate mode graphics:

→ As vertices are generated they are sent directly to the graphics processor for rendering on the display

→ There is no memory of the geometric data. Every time a redisplay of the scene is required, the entire creation and display process must be redone.

Retained mode graphics:

→ All geometric data is computed first and stored in some data structure, which is then all sent at once to the graphics processor for display.

→ The scene can be redisplayed whenever required by resending the stored data without needing to regenerate it.

1.2 Give 2 advantages and a disadvantage of using the pipeline approach to form CGI. [4]

\+ Can provide significant performance increase when the same sequence of concurrent operations must be performed on many or large datasets, such as vertices and pixels in CG.

\+ In a pipeline, each primitive can be processed independently, leading to fast performance and reduced memory requirements, since we need not keep all objects available.

\- Cannot handle most global effects such as shadows, reflections and blending in a physically correct manner.

1.3 Name the frames in the usual order in which they occur in the WebGL pipeline [4]

1. Object (or model) coordinates
2. World (or application) coordinates.
3. Eye (or camera) coordinates.
4. Clip coordinates
5. Normalised device coordinates.
6. Window (or screen) coordinates

See p14 of summary, p 159 textbook

TRANSFORMATIONS & VIEWING

2.1 Give 3 advantages for using homogeneous coordinates in CG [3]

Any of these:

1. All affine (line preserving) transformations can be represented as matrix multiplicatio
2. We can carry out operations on points and vectors using their homogeneous coordinate representations and ordinary matrix algebra.
3. The uniform representation makes carrying out successive transformations (concatenatio far easier than in 3-D space.
4. Although we have to work in 4-D to solve 3-D problems, less arithmetic is required.
5. Modern hardware implements homogeneous-coordinate operations directly, using parallelism to achieve high-speed calculations.

*

2.2. Explain what is meant by 'non-uniform foreshortening' of objects under a perspective camera. [1]

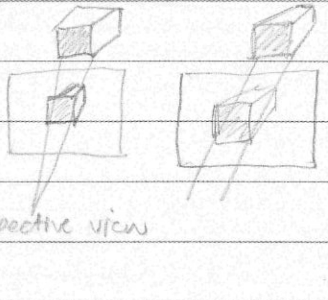
The images of objects farther from the centre of projection are reduced in size (diminution) compared with the objects closer to the COP.

2.3 The task of perspective projection is often separated into 2 steps: projection normalisation (or distortion), and orthographic projection.

2.3.1 What does each of these two steps do? [3]

- Projection normalisation converts all projections into simple orthogonal projections by distorting the objects such that the orthogonal projection of the distorted objects is the same as the desired projection of the original objects (done by applying the normalisation matrix aka projection matrix)
- The distorted objects are projected onto the plane with projectors perpendicular to the projection plane.

*



perspective view

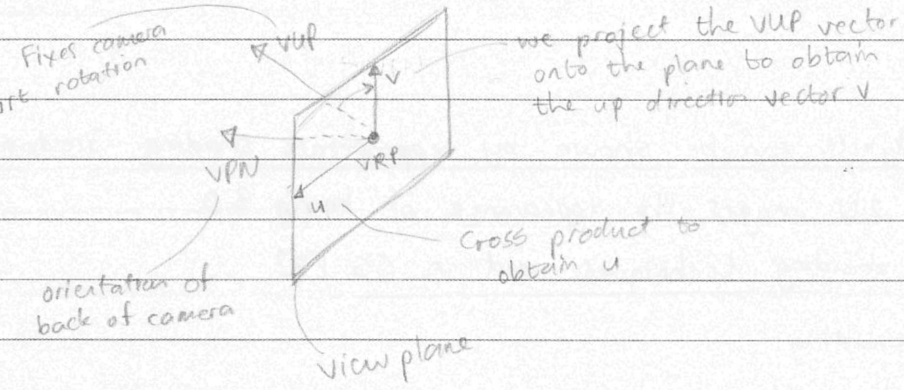
2.3.2 Give 2 advantages of using this approach to perspective projection. [2]

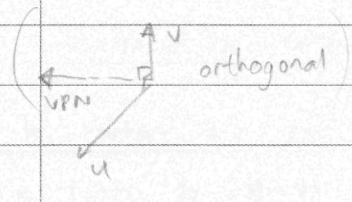
→ Both perspective and parallel views can be supported by the same pipeline

→ The clipping process is simplified because the sides of the canonical view volume are aligned with the coordinate axes.

2.4. A synthetic camera coordinate reference frame is given by a view reference point (VRP), a view plane normal (VPN) and a view-up vector (VUP). Using a diagram, show how these quantities describe the location and orientation of the synthetic camera. [4]

The VRP specifies the location of the COP, given in world coordinates.
The VPN aka n specifies the normal to the projection plane.
The VUP specifies which direction is up from the camera's perspective.





HIDDEN SURFACE REMOVAL

3.1 Hidden surface removal algorithms can be divided into two broad classes. Name and explain these two classes. [4]

1. Object space algorithms:
   - attempt to order the surfaces of objects in the scene such that rendering surfaces in a particular order provides the correct image.
   - must have all objects available so it can sort them into the correct back-to-front order.
2. Image-space algorithms:
   - work as part of the projection process and seek to determine the relationship among object points on each projector.

3.2. WebGL makes use of a z-buffer.

3.2.1. What information is stored in the z-buffer? [1]

Stores the necessary depth information as primitives are rasterized.

3.2.2 How does WebGL use this information for hidden surface removal? [3]

As primitives are rasterised, we keep track of the distance from the COP or projection plane to the closest point on each projector that has already been rendered. We update this info using the z-buffer as successive primitives are projected and filled, and ultimately display only the closest point on each projector.

3.2.3 Give 2 advantages to using this approach to hidden surface removal [2]

1. Complexity is proportional to the number of fragments generated by the rasterizer
2. Can be implemented with a small number of additional calculations over what must be done anyway without hidden surface removal.

LIGHTING & SHADING

4.1. The 3-D nature of objects can be shown by appropriate shading. Gradations or shades of colour give 2-D images the appearance of being 3-D.

4.1.1 Name the 3 major shading techniques used in CG [3]

1. Flat (or constant) shading
2. Gouraud (or smooth) shading
3. Phong shading

4.1.2 Describe the computation process for each of the techniques you listed in 4.1.1. [4]

1. The shading calculation is carried out only once for each polygon, and each point on the polygon is assigned the same shade. Shows differences in shading among adjacent polygons and stripes (Mach bands) along the edges
2. The lighting calculation is done at each vertex using the material properties and the vectors n, v and l. Each vertex will have its own colour used to interpolate a shade for each fragment. Normal at vertex = normalised average of the normals of the polygons that share the vertex.
3. Instead of interpolating vertex intensities, we interpolate normals across each polygon, thus making an independent light calculation for each fragment

4.1.3 Discuss how objects shaded by the different methods differ in appearance [4]

See previous answer.

4.2. In a simple CG lighting model we assume the specular reflection component $I_s = k_s L_s \cos^\alpha \theta$.

4.2.1. What lighting effect does the specular reflection component approximate? [1]

Specular reflection adds a highlight that we see reflected from shiny objects.

4.2.2 What does the term k_s represent? [1]

The coefficient k_s is the fraction of the incoming specular light that is reflected ($0 \leq k_s \leq 1$)

4.2.3 What effect does increasing the angle θ have? [1]

The amount of light that the viewer sees depends on θ, and the direction of the viewer. If viewing straight on, increasing the angle will reduce the amount of light.

4.2.4. What effect does increasing the value of α have? [1]

As α is increased, the reflected light is concentrated in a narrower region centred on the angle of a perfect reflector. (values 100-500 for metallic surfaces)

DISCRETE TECHNIQUES

5.1. Explain what is meant by bump-mapping. [1]

Bump mapping distorts the normal vectors during the shading process to make the surface appear to have small variations in shape, such as the bumps on a real orange.

5.2. Explain briefly the process of creating bump maps in CG [3]

The technique varies the apparent shape of the surface (without actually increasing geometric complexity) by perturbing the normal vectors as the surface is rendered. The colours that are generated by shading then show a variation in the surface properties. Unlike simple texture mapping, bump mapping will show changes in shading as the light source or objects moves.

5.3. What mapping technique computes the surroundings visible as a reflected image in a shiny object? [1]

Environment mapping or reflection mapping.

5.4. Explain briefly two methods of implementing the mapping technique described in 5.3. [4]

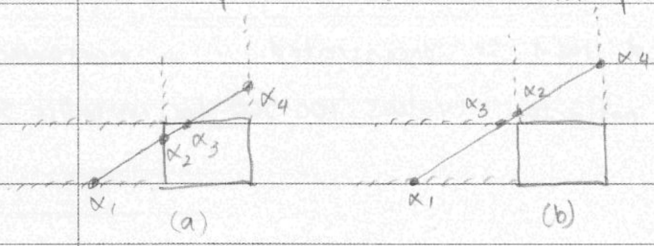
1. - 2 pass rendering: i) render the scene as "seen" from the direction of the normal of the reflecting object using sphere mapping. ii) Use this image to obtain the shades to place on the reflecting object for the second rendering of the scene.
2. Compute 6 projections, corresponding to the 6 sides of a cube, using 6 virtual cameras located at the centre of the box, each pointing in a different direction. Once we compute the 6 images, we can specify a cube map in WebGL with 6 function calls, one for each face of a cube centred at the origin.

FROM GEOMETRY TO PIXELS

6.1. Using diagrams describe briefly the Liang-Barsky clipping algorithm [8]

- Suppose we have a line segment defined by two endpoints $p_1 = (x_1, y_1)^T$, $p_2 = (x_2, y_2)^T$. We can express in matrix form: $p(\alpha) = (1-\alpha)p_1 + \alpha p_2$ (or as two scalar equations)
- As parameter α varies from 0 to 1, we move along the segment from p_1 to p_2.
  > negative values of α yield points on the line on the other side of p_1 from p_2 and vice versa for positive values of α for points the other side of p_2 from p_1.

*



- Consider the diagrams. There are 4 points where the line intersects the sides of the window. These correspond to the 4 values of parameter: α_1 (bottom), α_2 (left), α_3 (top), α_4 (right). We can order these values and determine which correspond to intersections that we need for clipping.
- In (a) α_2 and α_3 determine the clipped line segment, while in (b) notice that α_1 and α_3 before α_2, ie. the line intersects top and bottom of window before left and right, thus is entirely rejected.

6.2 Explain crossing (or odd-even) test wrt a point p inside a polygon [3]

It is a test for making inside-outside decisions. Any ray emanating from p and going off to infinity must cross an odd number of edges, while any ray from outside will cross an even no of edges if entering the polygon. This count determines inside and outside.