Q1 (20 marks)

Show the difference between perspective projection and orthogonal projection methods by discussing their transformation matrices and rendering results. Also describe how to set up the respective projections using OpenGL.

Solution:

Perspective projection:

It is a set of transforms uesd to simulate the human eyes look at a scene. It also can be treated the projection as being viewed through a camera viewfinder. The position, orientation, aspect ratio, near and far planes will control the behavior of the projection transformation. In perspective projection, the objects in the distance are appears smaller than objects close by. This method shows distant objects as smaller to provide additional realism. The mathematical transformation matrix is

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{d} & 0 \end{pmatrix}$$

The *OpenGL* function is *glPerspective(fov, aspect, near, far).*

Orthogonal projection:

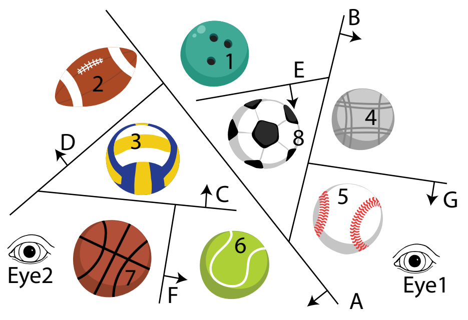
It is a set of transforms used to show profile, detail or precise measurements of a three dimensional object. Parallel lines in 3D world in an orthographically projected image are of the same scale regardless of whether they are far or near to the viewpoint. If the normal of the viewing plane is parallel to the Z axis, the mathematical transformation matrix is

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

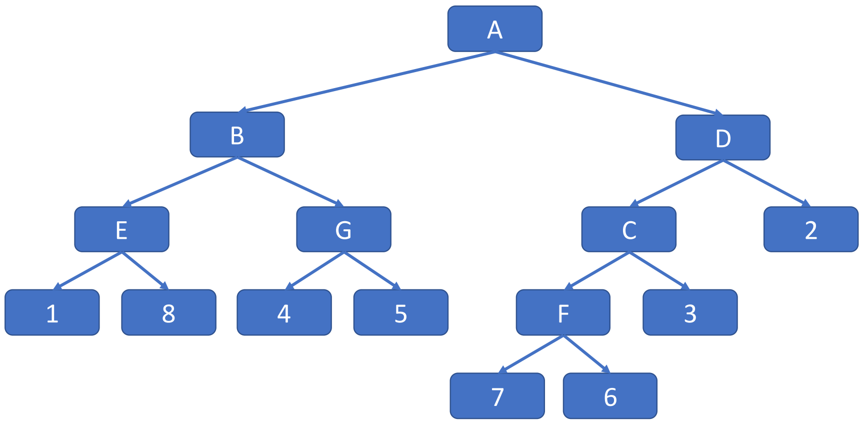
The *OpenGL* function is *glOrtho(left, right, bottom, top, near, far).*

Q2 (20 marks)

Consider a scene with 7 objects (numbered 1 to 7) as shown in the figure below. Construct the corresponding BSP tree. Based on this BSP tree, give the rendering sequence of the objects from the viewpoints Eye1 and Eye2, respectively.



Solution:



Rendering Sequence for each eye:
Eye 1: $2 \to 3 \to 7 \to 6 \to 1 \to 8 \to 4 \to 5$
Eye 2: $5 \to 4 \to 8 \to 1 \to 6 \to 7 \to 3 \to 2$

Q3 (20 marks)
(a) Explain how to detect whether or not a planar boundary face of an object is front-facing.
(b) Describe the Z-buffer algorithm briefly, and explain how it is used for hidden surface removal.

Solution:
(a) Assume the direction of the view is los, the normal vector of the boundary face is pv. Then the direction of the face can be decided by the value: $a = dot(los, pv)$. If $a < 0$, which means that the angle between los and pv is among $(90°, 270^{circ})$, so that the face is front-facing in current view.

(b) The Z-buffer algorithm is used for visual surface determination. First of all, we allocate Z-buffer. Then, set the initial value of it to be at infinity. After that, loop over all of the objects and rasterize the current object. Then, check the z value of each covered pixel(x,y). If it is less than the Z-buffer, set the Z-buffer to z value of the pixel and write pixel.
In the rendering process, the depth/Z value of each pixel is checked against an existing depth value. If the current pixel is behind the pixel in the Z-buffer, the pixel is rejected, otherwise it is shaded and its depth value replaces the one in the Z-buffer. Z-buffer supports dynamic scenes easily, and is currently implemented efficiently in graphics hardware.

Q4 (20 marks)
(a) Describe the three basic components of Phong illumination equation;
(b) Describe how the Flat shading method and Gouraud shading method work;
(c) Describe the pros and cons of the Phong shading method.

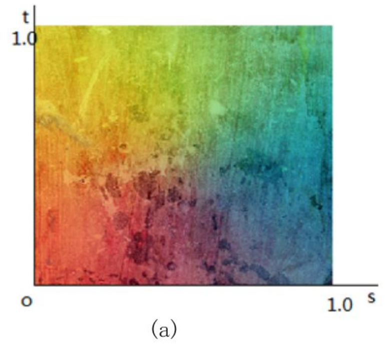
Solution:

(a) The three basic components of Phong illumination are: diffuse reflection, specular reflection and ambient reflection.
Diffuse reflection is also called Lambertian reflection. It appears equally bright from all viewing directions and its intensity is view-direction independent. Its reflected intensity depends only on direction of light source.
Specular reflection is the highlight observed on a shiny, glossy or mirror-like surface, such as the surface of glass or plated metal. Its appearance changes as the viewpoint moves. The intensity of specular reflection at a particular point of the surface depends on the viewing direction.
Ambient reflection is often modeled by a constant term in a simple reflection equation. It is purely fictitious.

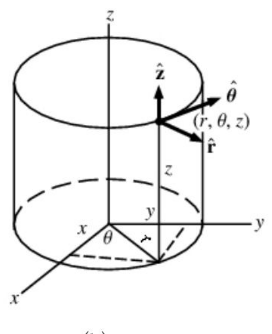
(b) Flat shading method: A single intensity is calculated for each surface polygon. All the pixels in that surface are filled with the same color.
Gouraud shading method: Use Phong's illumination equation to compute color intensities at all vertices; Color intensities of pixels along each edge are linearly interpolated from those at vertices; Color intensities of pixels within a span on a scanline are linearly interpolated from the color intensities at the two endpoints.

(c)Pros of the Phong shading method: Phong shading method can be used to provide better highlight rendering result than Gourand shading.
Cons of the Phong shading method: This method has to calculate normals for each inside positions, which require more time.

Q5 (20 marks)
(a) Consider a texture pattern defined in a unit square $[0,1]^2$ and a cylindrical surface $\mathcal{S}$, as depicted in the figure below.



(a)



(b)

The cylindrical coordinates of a point on $\mathcal{S}$ can be represented as $(r\cos\theta, r\sin\theta, z)$, where r is the radius of the cylindrical surface and $0 \le \theta \le 2\pi, 0 \le z \le 1$. Suppose we want to wrap the texture on $\mathcal{S}$. Derive a mapping which assigns a texture coorindates in $[0,1]^2$ to a point on $\mathcal{S}$.
(b) In OpenGL, antialiasing can be controlled by the second parameter (GL TEXTURE MIN FILTER/GL TEXTURE MAX FILTER) of the texture function glTexParameteri(). Please explain the different effects of GL NEAREST and GL LINEAR? Which one is more suitable for anti-aliasing?

Solution:

(a) The texture coordinate is represented as (s, t), where $0 \leq s, t \leq 1$. The cylindrical coordinate is represented as $(r\cos\theta, r\sin\theta, z)$, where $0 \leq \theta \leq 2\pi, 0 \leq z \leq 1$. The mapping relationship pf coordinates between the texture and the side face is $(s, t) = (\frac{\theta}{2\pi}, z)$.

(b)GL NEAREST means to return the value of texture element that is nearest (in Manhattan distance) to the center of the pixel being textured.

GL LINEAR will return the weighted average of the four texture elements that are closest to the center of the pixel being textured.

GL LINEAR is more suitable for anti-aliasing.