Ans. to Tut 1

Qn 1

a) The tables are as follows:

Vertex Table	Edge Table	Face Table	Attribute Table
a(0,0,0)	E1: a,b	F1: a b j f g	F1: $(k_{a,red}, k_{a,blue}, k_{a,green}) = (0.5,0.5,0.5)$ $(k_{l,r}, k_{l,b}, k_{l,g}) = ...$ …
b(10,0,0)	E2: b,j	F2: b c d j	F2 …
c(10,0,-20)	E3: j,f	F3: a g h i	F3 …
d(10,10,-20)	E4: f,g	F4: i h e d c	F4 …
e(5,15,-20)	E5: g,a	F5: j d e f	F5 …
f(5,15,0)	E6: b,c	F6: g f e h	F6 …
g(0,10,0)	E7: j,d	F7: a i c b	
h(0,10,-20)	E8: f,e		
i(0,0,-20)	E9: g,h		
j(10,10,0)	E10: a,i		
	E11: i,c		
	E12: c,d		
	E13: d,e		
	E14: e,h		
	E15: h,i		

The face table is expressed using vertices. Alternatively, it may be expressed using edges.

The attributes for each face may include for example,

ambient reflection coefficient k_a, diffusion reflection coefficient k_l, specular reflection coefficient k_s, specular reflection exponent n_s, transparency coefficient α, etc.

Note that in the Face Table, some faces are "CW" and some "CCW" as seen by us. because it should look "CCW" when seen by an observer outside. So "hidden faces" should be "CW".

b) $\hat{N} = |\overrightarrow{fe} \times \overrightarrow{fg}| = |(0,0,-20) \times (-5,-5,0)| = |(-100,100,0)| = \left(-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0\right)$

c) To find the outward normal, we can for example cross vector jd with vector jf. This gives

$$(A, B, C) = (0, 0, -20) \times (-5, 5, 0) = \begin{vmatrix} i & j & k \\ 0 & 0 & -20 \\ -5 & 5 & 0 \end{vmatrix} = (100,100,0)$$

Thus we have

100 X + 100 Y + D = 0

Put in any vertex, say *j = (10, 10, 0)* gives *D = -2000*. Thus the plane equation is

X + Y – 20 = 0

<u>Note</u>

-*X – Y + 20 = 0* is incorrect, because you should make sure that the normal (A, B, C) is computed in a way that iff

X + Y – 20 > 0

the point *(X, Y, Z)* is outside and vice versa.

d) Compute the plane equation of each face

$$A_i x + B_i y + C_i z + D_i = 0 \qquad i = 1, \ldots, 7$$

where (A_i, B_i, C_i) is the outward surface normal.

Given a point (x_p, y_p, z_p), if and only if

$$A_i x_p + B_i y_p + C_i z_p + D_i < 0 \qquad i = 1, \ldots, 7$$

then the point is inside the object (otherwise it is outside the object).

e) The above only works for convex objects. For concave objects, subdivide into m convex subparts. Then if and only if (x_p, y_p, z_p) is inside one of the m subparts, the point is inside the object.

In general, it may be difficult to subdivide a complex object into convex sub-objects. A general technique for testing whether a point is inside/outside is the inside-outside test.

f) Collision detection in a virtual reality walkthrough.

<u>Discuss</u>: Can you think of another practical or emerging application?

<u>Qn 2</u>

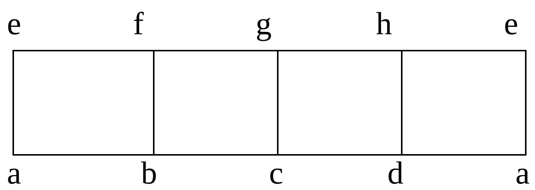
a)

vertex_table [0].X = 10; vertex_table [0].Y = 0; vertex_table [0].Z = 0; // b(10, 0, 0)
vertex_table [1].X = 10; vertex_table [1].Y = 4; vertex_table [1].Z = 0; // c (10, 4, 0)
vertex_table [2].X = 10; vertex_table [2].Y = 4; vertex_table [2].Z = 5; // g (10, 4, 5)
vertex_table [3].X = 10; vertex_table [3].Y = 0; vertex_table [3].Z = 5; // f (10, 0, 5)

edge_table [0][0] = 0; edge_table [0][1] = 1; // $\overline{bc}$
edge_table [1][0] = 1; edge_table [1][1] = 2; // $\overline{cg}$
edge_table [2][0] = 2; edge_table [2][1] = 3; // $\overline{gf}$
edge_table [3][0] = 3; edge_table [3][1] = 0; // $\overline{fb}$

face_table [0] [0] = 0; face_table [0][1] = 1; face_table [0][2] = 2;
face_table [0][3] = 3; face_table [0][4] = 0;

b)



To represent the whole object, the preferred way is to use 3 quadrilateral meshes, that is, use a separate quadrilateral mesh for both the top and the bottom face.

One example is *gluCylinder*. To close the ends of the cylinders, we use two *gluDisk* commands. Thus there are three quadrilateral meshes.

c) $(A, B, C) = \overrightarrow{cd} \times \overrightarrow{cg} = \begin{vmatrix} i & j & k \\ -10 & 21 & 0 \\ 0 & 0 & 5 \end{vmatrix} = (105, 50, 0)$

$105X + 50Y + D = 0$

Put in (X, Y, Z) = (10, 4, 0) $\Rightarrow$ D = -1250

The equation is

$105X + 50Y - 1250 = 0 \Rightarrow 21X + 10Y - 250 = 0$

d) Using mesh has the advantage over table that there is no need to specify the edge and face tables. Thus it saves memory.

Qn 3

a) The parametric form samples uniformly across the parameters. This gives a set of pixels that appear continuous when displayed in screen coordinates. On the contrary, without techniques to keep track of the slope of the curve, the non-parametric form will sample uniformly in X (or Y), which gives a non-uniform sampling across Y (or X). When displayed, this results in some pixels along the curve that are not sampled (gap in the curve).

b) Let $\left(\frac{Z}{r_z}\right)^{2/s1} = \sin^2\phi$ This gives $Z = r_z \sin^{s1}\phi$

$$\left(\left(\frac{X}{r_x}\right)^{2/s2} + \left(\frac{Y}{r_y}\right)^{2/s2}\right)^{s2/s1} = \cos^2\phi \quad \Rightarrow \quad \left(\frac{X}{r_x}\right)^{2/s2} + \left(\frac{Y}{r_y}\right)^{2/s2} = (\cos\phi)^{2s1/s2}$$

Let $\left(\frac{X}{r_x}\right)^{2/s2} = (\cos\phi)^{2s1/s2}\cos^2\theta$ and $\left(\frac{Y}{r_y}\right)^{2/s2} = (\cos\phi)^{2s1/s2}\sin^2\theta$

This gives

$X = r_x \cos^{s1}\phi \cos^{s2}\theta$ and $Y = r_y \cos^{s1}\phi \sin^{s2}\theta$

$0 \le \theta < 2\pi$ $\quad -\frac{\pi}{2} \le \phi < \frac{\pi}{2}$

Physical meaning:

Consider a globe:

ϕ latitude
θ longitude

Qn 4

```
double sgn (double t )
{
  if     (t > 0)
    return 1.0;
  else if (t < 0)
     return -1.0;
  else
     return 0.0;
```

```
}

void calculate_mesh (double s1, double s2)
// calculate the parameters of the sphere mesh
{
    double ta, tb, tc, td;

    for (int i=0; i<GRIDSIZE; i++)
        for (int j=0; j<GRIDSIZE; j++)
        {

                ta = cos(-PI/2 + i*PI/(GRIDSIZE-1));
                tb = sin(-PI/2 + i*PI/(GRIDSIZE-1));
                tc = cos(-PI + j*2 * PI/(GRIDSIZE-1));
                td = sin(-PI + j*2 * PI/(GRIDSIZE-1));

                mesh[i][j].x = 50 * sgn (ta) * pow ( fabs ( ta ), s1 )
                                * sgn (tc) * pow ( fabs ( tc ), s2 ) + 50;
                mesh[i][j].y = 100 * sgn (ta) * pow ( fabs ( ta ), s1 )
                                * sgn (td) * pow ( fabs ( td ), s2 ) + 100;
                mesh[i][j].z = 200 * sgn (tb) * pow ( fabs ( tb ), s1 ) + 200;

        }
}
```

Qn 5

a) Since the standard form of (2D) parabola is $Y^2 = 4aX$,

Non-parametric form of (3D) elliptic paraboloid:

$$X^2 + Y^2 = 4aZ$$

Parametric form:

$$X = \sqrt{4a\phi}\cos\theta$$
$$Y = \sqrt{4a\phi}\sin\theta$$
$$Z = \phi$$

Note that the idea of the parametric form is to introduce two additional independent variables (in this case θ and ϕ) and re-express X, Y, Z as functions of these additional variables to make the sampling more uniform. Hence more than one possible parametric form is viable.

Note: a closely related shape is the hyperbolic paraboloid, which has the standard form

$$X^2 - Y^2 = 4aZ$$

For more information, consult MathWorld: http://mathworld.wolfram.com/

b) Since the standard form of (2D) hyperbola is $\left(\frac{X}{a}\right)^2 - \left(\frac{Y}{b}\right)^2 = 1$,

non-parametric form of (3D) hyperboloid of two sheets is:

$$\left(\frac{X}{a}\right)^2 + \left(\frac{Y}{b}\right)^2 - \left(\frac{Z}{c}\right)^2 = -1$$

Using the same arguments as that of a sphere, but instead use $\cosh^2 x - \sinh^2 x = 1$,

Parametric form:

$$X = a(\sinh\phi)(\cos\theta)$$
$$Y = b(\sinh\phi)(\sin\theta)$$
$$Z = c(\cosh\phi)$$

Note: Other forms of hyperboloids are hyperboloid of one sheet and elliptic hyperboloid. See MathWorld: http://mathworld.wolfram.com/ for details.