COMPUTER GRAPHICS (NOV 2018)

Q.P.Code: 60317

Q.1) Attempt any five from the following:- **(20 M)**

a) Compare Raster and Random Scan Techniques. **(5 M)**

Ans:

Random Scan	Raster Scan
1. It has high resolution.	1. Its resolution is low
2. It is more expensive.	2. It is less expensive.
3. Any modification if needed is easy	3. Modification is tough.
4. solid pattern is tough to fill.	4. Solid pattern is easy to fill.
5. Refresh rate depends on resolution.	5. Refresh rate does not depend on the picture.
6. Only screen with view on an area is displayed.	6. Whole screen is scanned.
7. Beam penetration technology comes under it.	7. Shadow mark technology comes under this.
8. It does not use interlacing method.	8. It uses interlacing.

b) What are the disadvantages of DDA algorithm? **(5 M)**

Ans:

- Floating point arithmetic in DDA algorithm is still time consuming.
- The algorithm is orientation dependent. Hence end point accuracy is poor.
- Although DDA is fast, the accumulation of round-off error in successive additions of floating point increment, however can cause the calculation pixel position to drift away from the true line path for long line segment.
- Rounding-off in DDA is time consuming.

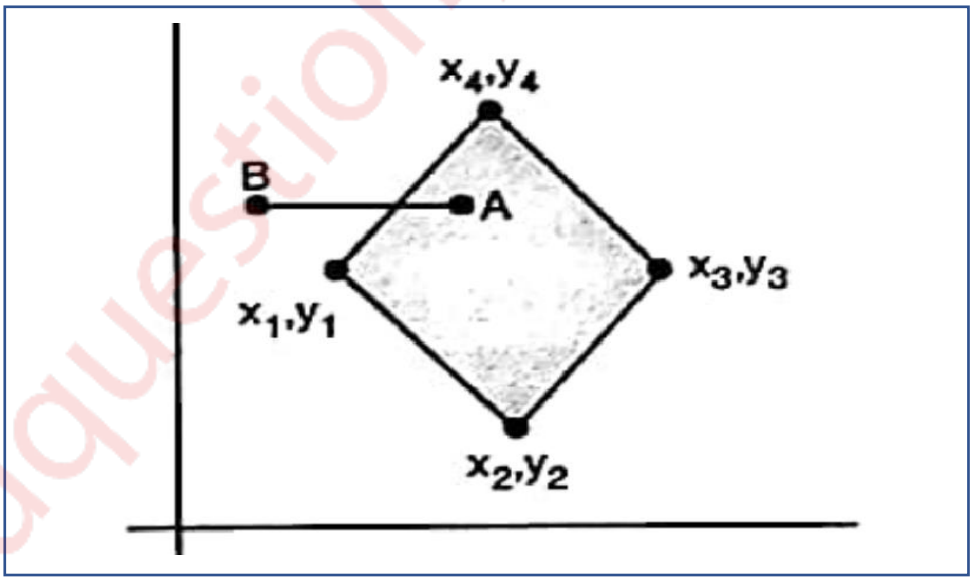
- Because of round off, errors are introduced and causes the calculated pixel position to drift away from the true line path.
- Because of floating point operation the algorithm is time consuming.

---

c) Explain inside outside test used in filling algorithm. (5 M)

Ans:

- One method of doing this is to construct a line segment between a point in question i.e. point to test whether inside or outside, and a point which is surely outside the polygon. But ow we are going to find out a point which is surely outside the polygon? It is very easy to find out that point.
- For example, pick a point with an x co-ordinate smaller than the smallest co-ordinate of the polygons vertices and the y co-ordinate will be any y value, or for simplicity we will take y value same as the y value of point in question.
- In this case point A is one, which we want to check i.e. whether point A is inside polygon or not.
- As we are using arrays to store vertices of polygon we can easily come to know the vertex which is having lowest value of x and that is x_1. So we have to select a point smaller than x_1 and generally we select same value of y as that of point A, in order to draw straight line. But even if you select any y value for outside point, it will not make any difference.



- Then count how many intersections are occurring with polygon boundary by this line till the point in question i.e. 'A', id reached. If there are an odd number of intersections then the point is outside the polygon.

---

d) What are aliasing and antialiasing? Explain any one antialiasing method. (5 M)

Ans:

- In computer graphics, the process by which smooth curves and other lines become jagged because the resolution of the graphics device or file is not high enough to represent a smooth curve.
- In the line drawing algorithms, we have seen that all rasterized locations do not match with the true line and we have to select the optimum raster locations to represent a straight line. This problem is severe in low resolution screens. In such screens line appears like a stair-step, as shown in the figure below. This effect is known as aliasing. It is dominant for lines having gentle and sharp slopes.

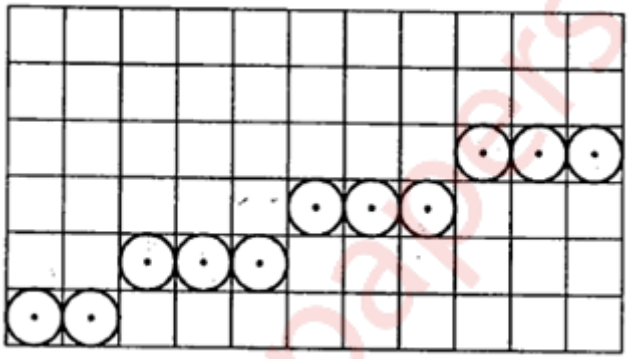


Fig. Aliasing effect

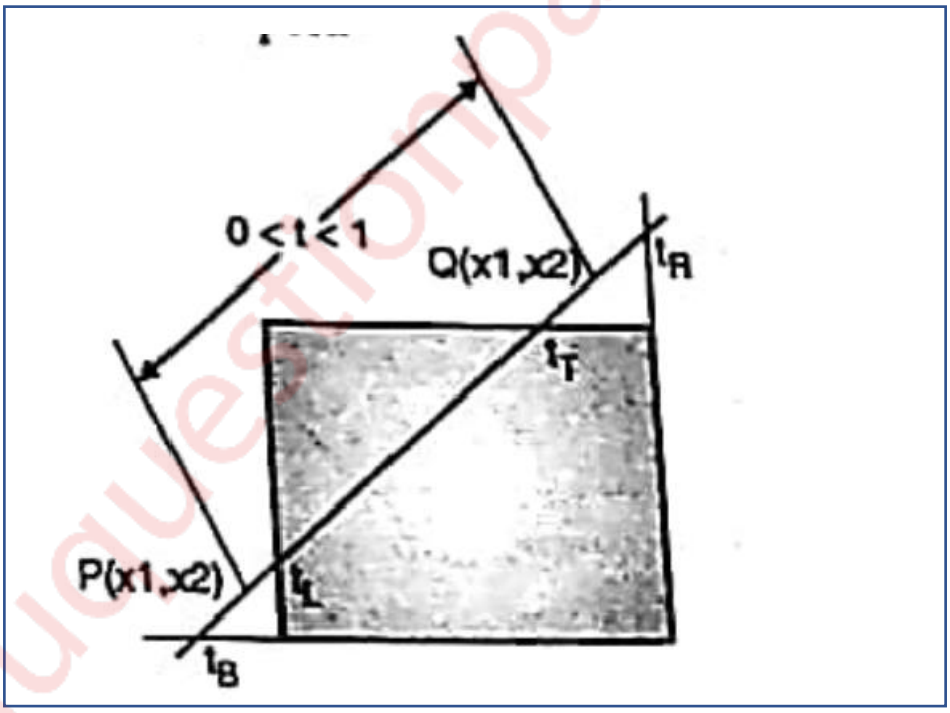
- The aliasing effect can be reduced by adjusting intensities of the pixels along the line. The process of adjusting intensities of the pixels along the line to minimize the effect of aliasing is called antialiasing.
- **Antialiasing** is a term for techniques that are designed to mitigate the effects of aliasing.
- The idea is that when a pixel is only partially covered by a shape, the colour of the pixel should be a mixture of the colour of the shape and the colour of the background.
- When drawing a black line on a white background, the colour of a partially covered pixel would be grey, with the shade of grey depending on the fraction of the pixel that is covered by the line.
- The technique of anti-aliasing is:
- **Prefiltering:** This is technique that determines pixel intensity based on the amount of that particular pixel coverage by the object in the scene i.e. It computes pixel colours depending on the objects coverage.
- It means how much part or fraction of that pixel is covered by the object and depending on that, it sets the value of the pixel. It requires large number of calculations and approximations. Prefiltering generates more accurate antialiasing effect. But due to its high complexity of calculations it is not used.

Q.2)

a) Explain Liang Barsky line clipping algorithm. Apply the algorithm to clip the line with coordinates(35, 60) and (80, 25) against window (xmin, ymin) = (10, 10) and (xmax, ymax) = (50, 50). (10 M)

Ans:

- The Liang-Barsky algorithm uses the parametric equation of a line and inequalities describing the range of the clipping box to determine the intersections between the line and clipping box. With these intersections it knows which portion of the line should be drawn. This algorithm is more efficient than Cohen-Sutherland. The ideas for clipping line of Liang-Barsky and cyrus-Beck are the same. The only difference is Liang-Barsky algorithm has been optimized for an upright rectangular clip window. So we will study only the idea of Liang-Barsky.
- Liang and Barsky have created an algorithm that uses floating-point arithmetic but finds the appropriate ends points with at most four computations. This algorithm uses the parametric equations for a line and solves for inequalities to find the range of the parameter for which the line is in the viewport.



- Let P(X1, Y1), Q(X2, Y2) be the line which we want to study. The parametric equation of line segment from gives x-values and y-values for every point in terms of a parameter that ranges from 0 to 1. The equations are

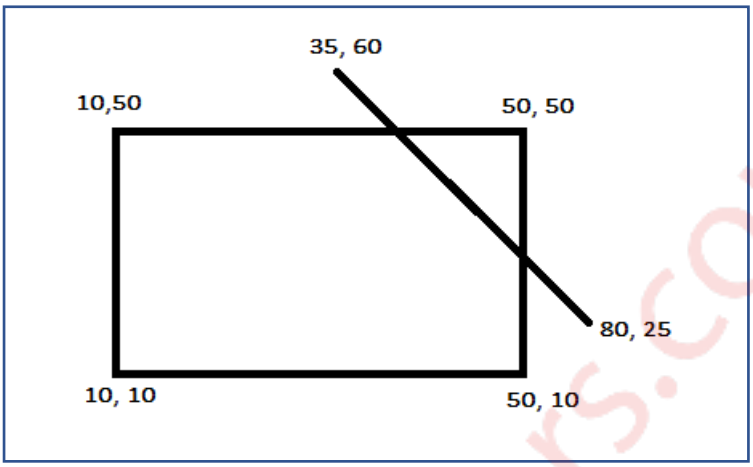
  X = X1 + (X2 – X1)*t = Y1 + dY*t

  And

  Y = Y1 + (Y2 – Y1)*t = X1 + dX*t

We can see that when t=0, the point computed is P(x1, y1) and when t =1, the point computed is Q(x2, y2).

- Lets first draw the window and line as shown in fig.



Given things are XL = 10, YB = 10, XR = 50, YT = 50

Lets call a line AB with its coordinates as X1 = 35, Y1 = 60, X2 = 80, Y2 = 25

Now we have to find Dx and Dy as

Dx = X2 – X1 = 80 – 35 = 45

Dy = Y2 – Y1 = 25 – 60 = -35

Lets calculate the values of parameters P and Q as

P1 = -Dx = -45

P2 = Dx = 45

P3 = -Dy = -(-35) = 35

P4 = Dy = -35

Now,

Q1 = X1 – XL = 35 -10 = 25

Q2 = XR – X2 = 50 - 80 = -30

Q3 = Y1 – YB = 60 -10 = 50

Q4 = YT – Y2 = 30 - 25 = 5

Now lets find P,

P1 = Q1/P1 = 25/(-45) = (-5/9)

P2 = Q2/P2 = -30/(45) = (-2/3)

P3 = Q3/P3 = 50/(35) = (10/7)

P4 = Q4/P4 = 5/(-35) = (-1/7)

Now,

t1 = Max(-5/9, 10/7, 0) = 10/7

t2 = Min(-2/3, -1/7, 1) = -2/3

Now,

X1' = X1 + Dx*t1

= 35+45*(10/7)

= 99.29

Y1' = Y1 + Dy*t1

= 60+(-35)*(10/7)

= 10

And with t2 it will be

X2' = X1 + Dx*t2

= 35+45*(-2/3)

= 5

Y2' = Y1 + Dy*t2

= 60+(-35)*(-2/3)

= 83.33

From this we will come to know that a point (5, 83.33) is an intersection point with respect to the edge of the window boundary. So we need to discard the line from point (35, 60) to (5, 83.33) and consider line (5, 83.33) to (80, 25).

---

b) Derive the matrix for 2D rotation about an arbitrary point. (10 M)

Ans:

- **Rotation about an Arbitrary Point:-**
- To rotate an object about an arbitrary point, (Xp ,Yp) we have to carry out three steps:
    - Translate point (Xp, Yp) to the origin.
    - Rotate it about the origin and,
    - Finally, translate the centre of rotation back where it belongs (See figure 1.).

- we have already seen that matrix multiplication is not commutative, i.e. multiplying matrix A by matrix B will not always yield the same result as multiplying matrix B by matrix A. Therefore, in obtaining composite transformation matrix, we must be careful to order the matrices so that they correspond to the order of the transformations on the object. Let us find the transformation matrices to carry out individual steps.

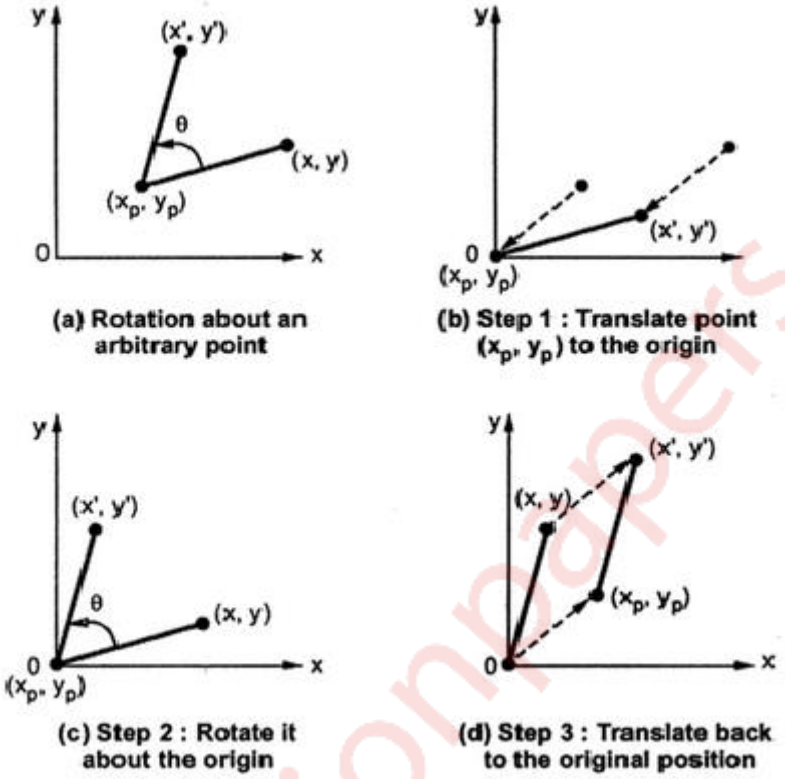


Fig. 1

The translation matrix to move point (x_p, y_p) to the origin is given as, x_p

$$T_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xp & -yp & 1 \end{bmatrix}$$

The rotation matrix for counterclockwise rotation of point about the origin is given as,

$$R = \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The translation matrix to move the center point back to its original position is given as,

$$T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

Therefore the overall transformation matrix for a counterclockwise roatation by an angle Θ about the point (xp, yp) is given as,

$$T_1 * R * T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xp & -yp & 1 \end{bmatrix} * \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

$$= \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ -xpcos\Theta + ypsin\Theta & -xpsin\Theta - ypcos\Theta & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

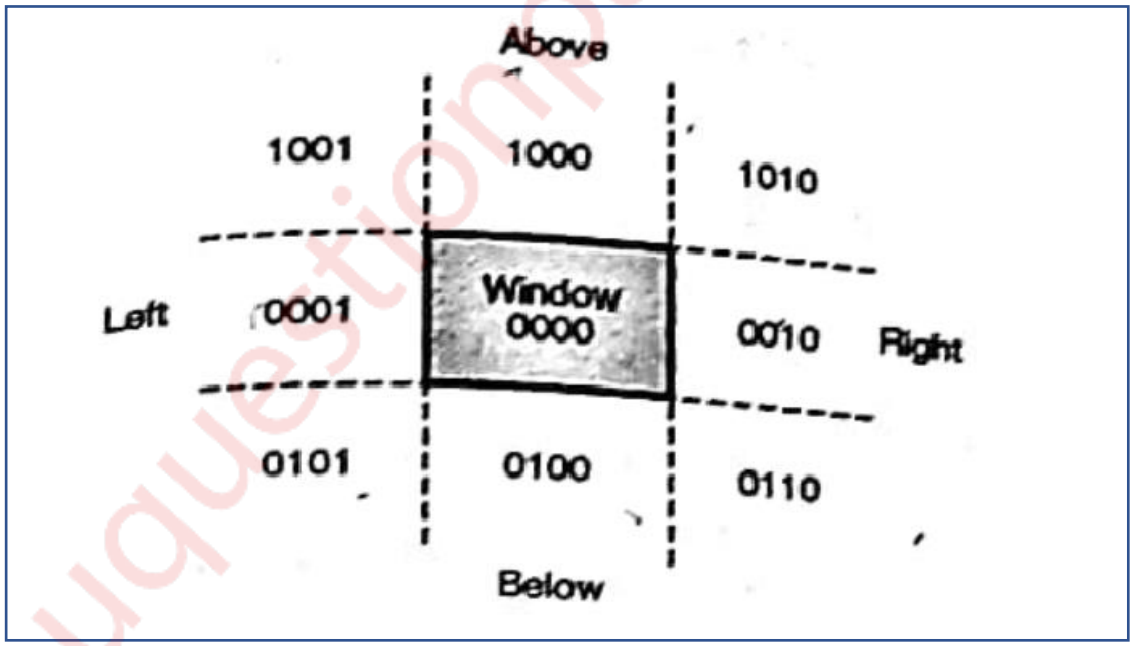
$$= \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ -xpcos\Theta + ypsin\Theta + xp & -xpsin\Theta - ypcos\Theta + yp & 1 \end{bmatrix}$$

Q.3)

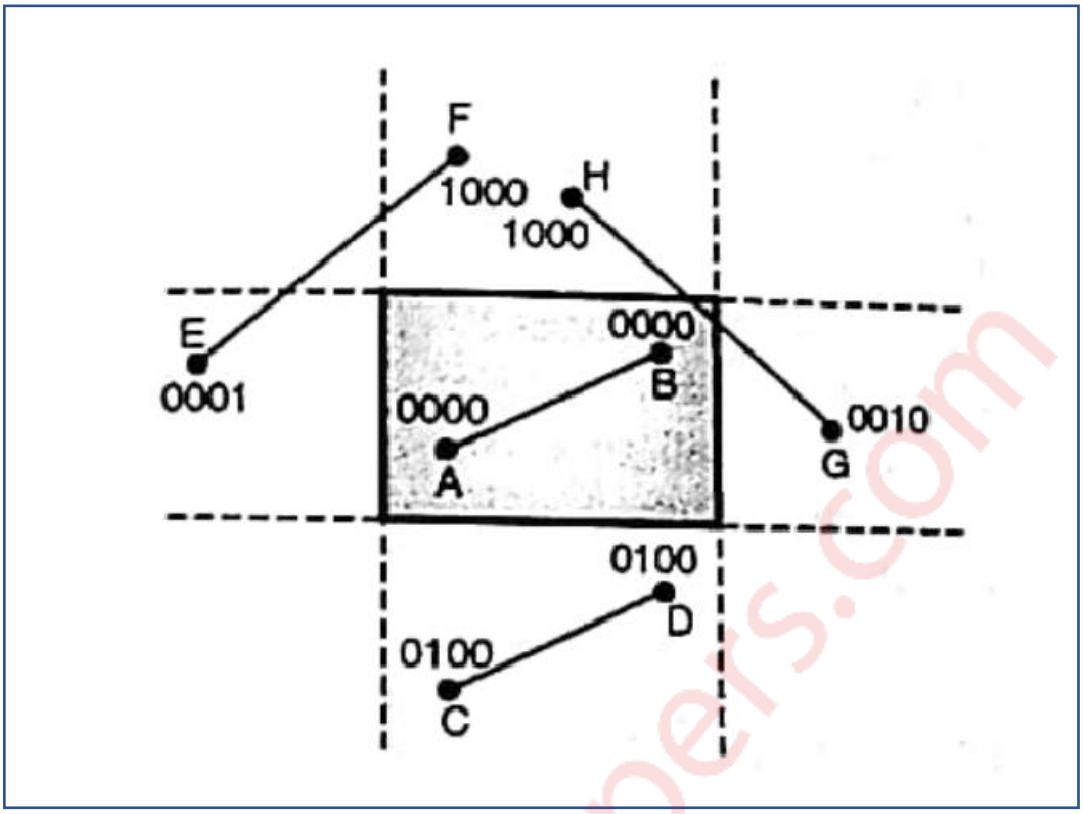
a) Explain the Cohen-Sutherland line clipping algorithm with suitable example. (10 M)

Ans:

- It is one of the popular line clipping algorithm. This algorithm immediately removes the lines which are lying totally outside the window.



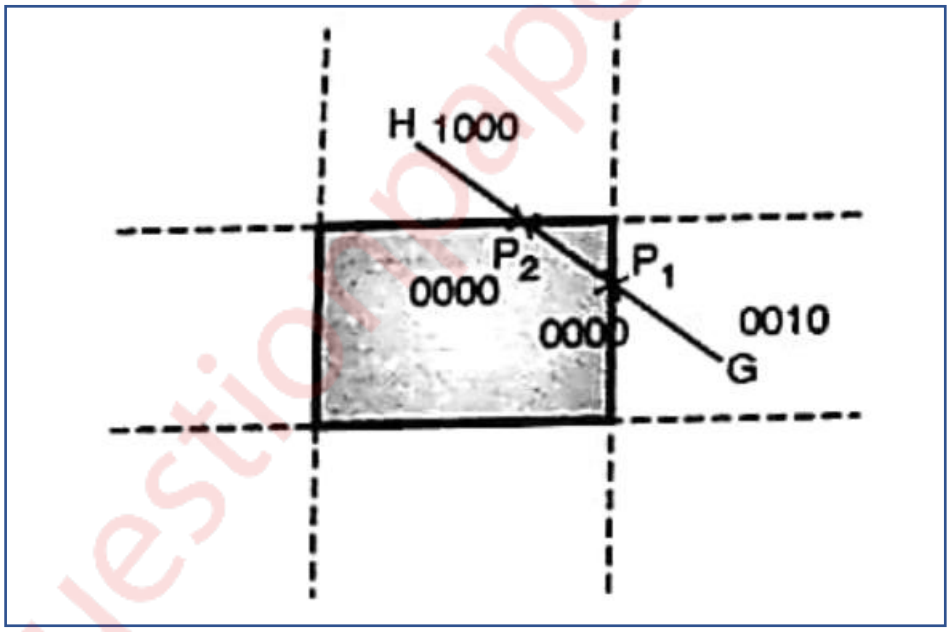
- End point of each line is assigned a four bit binary code which is called ass out code. We can give abbreviated name to this four bit code as, “ABRL” where A = above, B = below, R = right, L = left.



- The cohen Sutherland algorithm tells that if the line is totally on one side of the window then immediately we have to discard that line. Similarly if the line is totally inside the window directly display that line without clipping.
- **Case 1:**
    - Consider a line AB. Here both end points of AB are surely inside the window. Therefore the out code for end point A will be
      out code(A) = ABRL = 0000, out code(B) = ABRL = 0000
    - The algorithm says that if the out codes of both endpoints of a line are 0000. Then the line is fully visible. So we should not clip the line.
- **Case 2:**
    - Consider a line CD. Here endpoints of CD are surely outside the window. The out code for endpoint C will be, out code(C) = ABRL = 0100, outcode(D) = ABRL = 0100
    - Now here for line CD, the outcodes of both endpoints of a line are not zero, so we have to perform logical AND operation of both outcodes. Outcode(C) = 0100 AND Outcode(D) = 0100 = 0100.
    - If the result of AND operation is nonzero, then the line is surely outside the window. So clip that line. Here line CD is clipped as its AND result is nonzero.
- **Case 3:**
    - But if the logical AND result is zero, then we cannot make form statement about a line, whether it should be visible or partially visible. Consider line EF and GH. For line EF outcode of E will be 0001 and outcode of F will be 1000. As both outcodes are not zero, so we have to perform logical AND.

Outcode(E) = 0001 AND outcode(F) = 1000 = 0000

- The AND result is zero. It means the line EF is not lying completely on any side of the window.
  Similarly for line GH,
  Outcode(G) = 0010 AND outcode(H) = 1000 = 0000
- If we observe both lines GH and EF, the AND result of both line is zero. But line GH is partially visible and line EF is fully invisible. So, far such line we have to edge of the window the line is intersecting and then we have to find the intersection point. To find out this we can make use of outcode.
- Consider line EF. Outcode of E is 0001 and F is 1000. Here we have to compare the first bit of both outcodes..
- First bit of E is 1 and first bit of F is 0. That means point E of line EF is lying outside the left boundary of window. So we have to find out intersection point of line EF with left boundary of window. Call that intersection as P_1. Now we are having two lines as EP_1 and P_1F.



- As point E is surely outside the left boundary of window so we are clipping the line EP_1. Now we will concentrate an P_1F. now we have to find outcode of P_1, which will be 1000. Here both outsides are nonzero i.e. P_1= 1000 and F = 1000 so, logical AND will be nonzero i.e. 1000, it means P_1F is lying completely on one side of window i.e. above the window, so discard the line. In this we are eliminating whole line.
- The same method is used for partially visible lines also such as GH.

b) What is meant by parallel and perspective projection? Derive matrix for matrix projection. (10 M)

Ans:

Parallel Projection:

i. In parallel projection, Z coordinate is discarded and parallel lines from each vertex on the object are extended until they intersect the view plane.

ii. The point of intersection is the projection of the vertex.

iii. We connect the projected vertices by line segments which correspond to connections on the original object.

iv. A parallel projection preserves relative proportions of objects.

v. Accurate views of the various sides of an object are obtained with a parallel projection. But not a realistic representation.

vi. Parallel projection is shown below in figure 30.

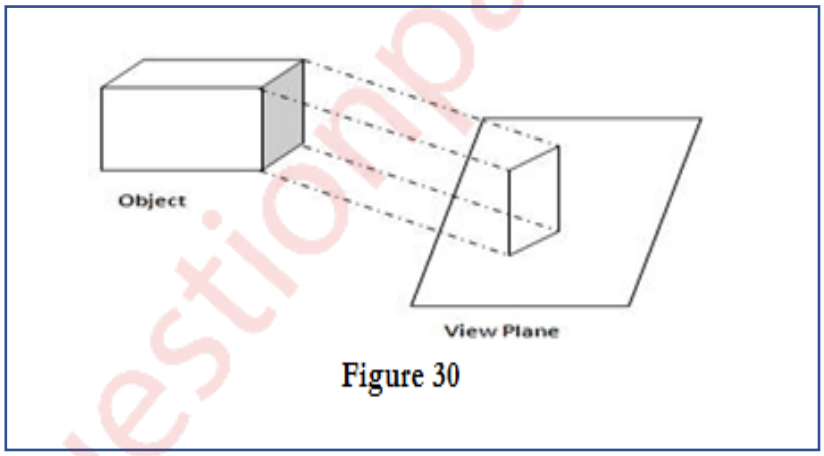


Figure 30

Perspective Projection:

i. In perspective projection, the lines of projection are not parallel.

ii. Perspective Projection transforms object positions to the view plane while converging to a center point of projection.

iii. In this all the projections are converge at a single point called the “center of projection” or “projection reference point”.

iv. Perspective projection produces realistic views but does not preserve relative proportions.

v. Projections of distant objects are smaller than the projections of objects of the same size that are closer to the projection plane.

vi. Perspective projection is shown below in figure 31.

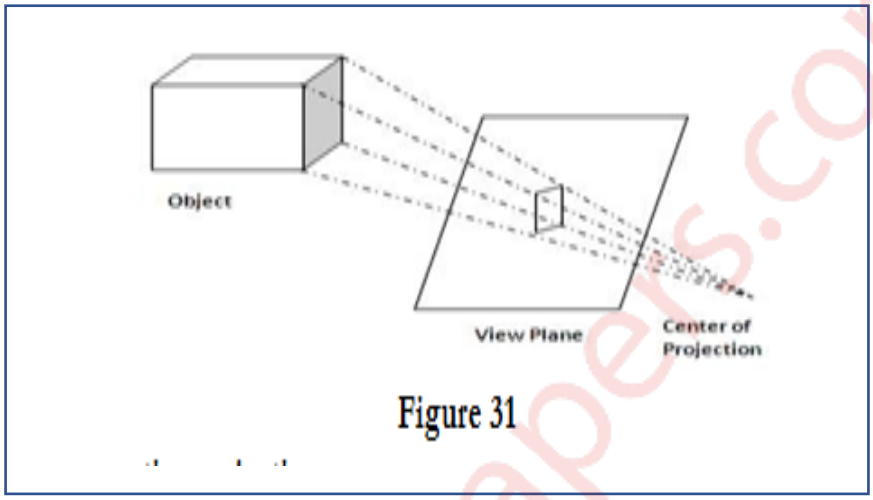
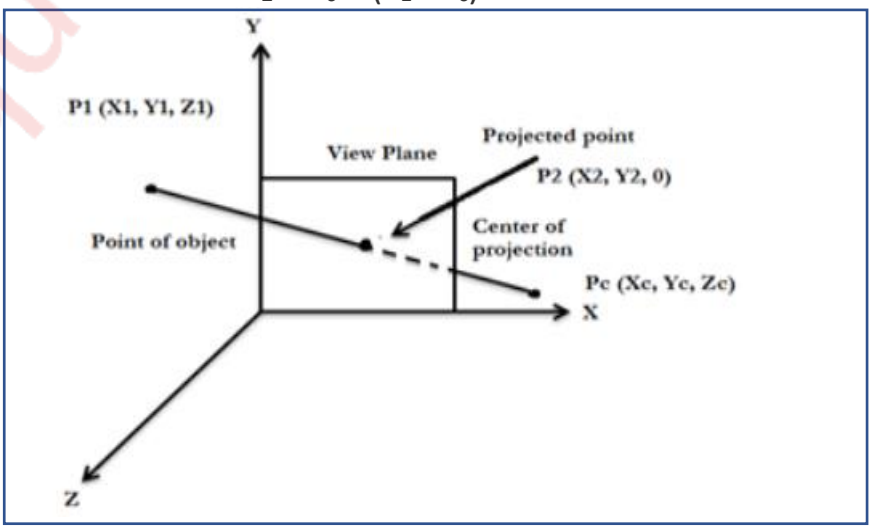


Figure 31

Matrix for perspective projection:

Let us consider the center of projection is at $P_c(X_c,Y_c,Z_c)$Pc(Xc,Yc,Zc) and the point on object is $P_1(X_1,Y_1,Z_1)$P1(X1,Y1,Z1), then the parametric equation for the line containing these points can be given as

$$X_2 = X_c + (X_1 - X_c)U$$
$$Y_2 = Y_c + (Y_1 - Y_c)U$$
$$Z_2 = Z_c + (Z_1 - Z_c)U$$



For projected point Z2 is 0, therefore the third equation can be written as

$0 = Z_C + (Z_1 - Z_C)U$

$U = -Z_C/Z_1 – Z_2$

Substituting the value of U in first two equations we get,

$X_2 = (X_C - Z_C) * (X_1 - X_C) / (Z_1 - Z_C)$

$= X_CZ_1 – X_CZ_C – X_1Z_C + X_CZ_C/Z_1 – Z_C$

$= X_CZ_1 – X_1Z_C / Z_1 – Z_C$

$Y_2 = (Y_C - Z_C) * (Y_1 - Y_C) / (Z_1 - Z_C)$

$= Y_CZ_1 – Y_CZ_C – Y_1Z_C + Y_CZ_C / Z_1 - Z_C$

The above equation can be represeented in the homogeneous matrix form as given below,

$$[X_2\ Y_2\ Z_2\ 1] = [X_1\ Y_1\ Z_1\ 1] \begin{bmatrix} -ZC & 0 & 0 & 0 \\ 0 & -ZC & 0 & 0 \\ XC & YC & 0 & 1 \\ 0 & 0 & 0 & -ZC \end{bmatrix} \begin{bmatrix} -ZC & 0 & 0 & 0 \\ 0 & -ZC & 0 & 0 \\ XC & YC & 0 & 1 \\ 0 & 0 & 0 & -ZC \end{bmatrix}$$

Here, we have taken the center of projection as PC(XC, YC, ZC), if we take the center of projection on the negative Z axis such that

X = 0

Y = 0

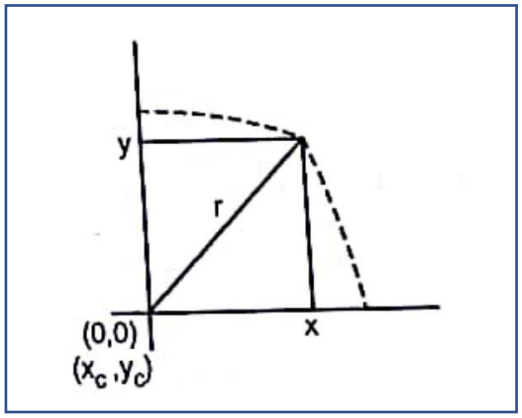
$Z = -Z_C$

Q.4)

a) Specify midpoint circle algorithm. Using the same, plot the circle whose radius is 8 units and center is at (10, 10). (10 M)

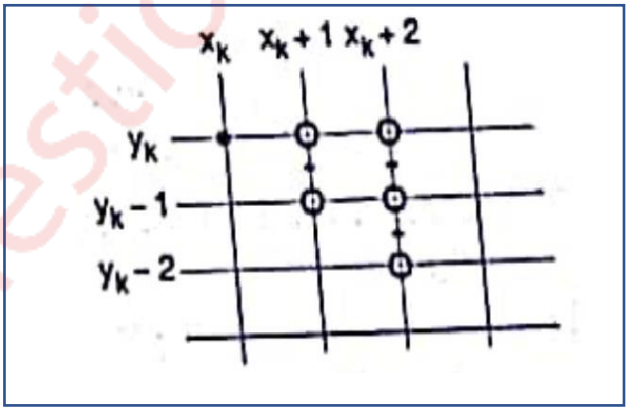
Ans:

- Here we have to determine the closest pixel position to the specified circles path at each step. For this we have to accept radius 'r' and center point x_c and y_c.
- Then each calculated point is to be moved to proper position by adding x_c and y_c to corresponding x and y-value. In first quadrant we are moving in x-direction with starting point as x=0 to ending point x=y.
- To apply a midpoint method, we define a circle function.



$F_{circle}(x,y) = x^2+y^2-r^2$ which is same as $r^2=x^2+y^2$ which is derived from equation of circle, $(x-x_c)^2+(y-y_c)^2=r^2$. But here we are considering x_c, y_c as (0, 0)
Hence, $x^2+y^2 = r^2$.

- Any point (x, y) on the boundary of the circle with radius r satisfies the equation $F_{circle}(x,y) = 0$. If point is inside the circle function is negative and if point is outside circle then circle function is positive.
  i.e. if
  $f_{circle}(x,y) < 0$, then x, y is inside the circle boundary.
  $= 0$, then x, y is on circle boundary.
  > 0, then x, y is outside the circle boundary.
- Thus circle function is the decision parameter in this algorithm. Assuming that we have just plotted x_k, y_k point.
- Now we have to determine whether next point $(x_k + 1, y_k)$ is closer or $(x_k + 1, y_k -1)$ is closer to actual circle.



- Our decision parameter (P_k) is circle function.
  $P_k = f_{circle}(x_k + 1, y_k)$ or
  $P_k = f_{circle}(x_k + 1, y_k - 1)$
- Therefore we will take midpoint of these two point as,
  $P_k = f_{circle}(x_k + 1, y_k – 1/2)$
  $= f_{circle}(x_k + 1)^2 +(y_k – 1/2)^2 - r^2$ from circle equation.
- If $P_k < 0$, then it means this point $(x_k + 1, y_k – 1/2)$ is inside circle. So we have to plot pixel $(x_k + 1, y_k)$.

After this we have by 1.

i.e. $P_{knew} = f_{circle}(x_k + 1, y_k - 1/2)$

$= (x_k + 2)^2 + (y_k - 1/2)^2 - r^2$

If we solve this we will get

$P_{knew} = P_k + (2\,x_k + 1)$

i.e. when, earlier we have not changed row.

- If $P_k > 0$, then we will select $y_k - 1$ for displaying. Like this we will select the point to be displayed.

  $P_{knew} = f_{circle}(x_k + 2, y_k - 3/2)$

  $= (x_k + 2)^2 + (y_k - 3/2)^2 - r^2$

  If we solve this we will get

  $P_{knew} = P_k + (2\,x_k - 2\,y_k + 1)$

  i.e. when, earlier we have not changed row.
- Similarly the initial decision parameter is obtained by evaluating the circle function at start position $(x_0, y_0) = (0, r)$

  Now calculate the decision parameter.

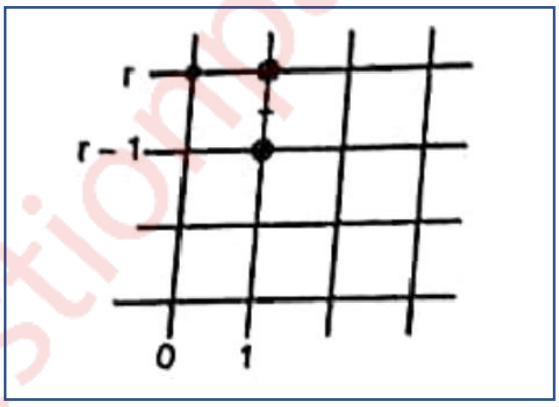
  $P_0 = f_{circle}(x_k + 1, y_k - 1/2)$

  But $x_k = 0$ and $y_k = r$

  $x_k + 1 = 1$ and $y_k - ½ = r - ½$

  $P_0 = f_{circle}(1, r - ½) = 1^2 + (r - ½)^2 - r^2$

  $P_0 = 5/4 - r$



If radius r is integer we will round it ass $P_0 = 1 - r$.

- Given: $r = 8$, $x_c = 10$, $y_c = 10$

  As stated in algorithm,
  - We first calculate the points assuming the centre co-ordinates (0, 0)
  - At the end we translate the circle.

  Now,

  $P_0 = 1 - r$

  $= 1 - 8 = -7$

  $P_0 < 0$

$X_{k+1} = x_{k+1} = x_k + 1 = 0 + 1 = 1$

$Y_{k+1} = y_k = 8$

$P_{k+1} = P_k + 2 * x_{k+1} + 1 = -7+2*1+1 = -4$

Continue this process till (x=y) i.e. till angle 45, where x and y becomes equal.

X	Y
10	18
11	18
12	17
13	16
14	15
14	15
16	14

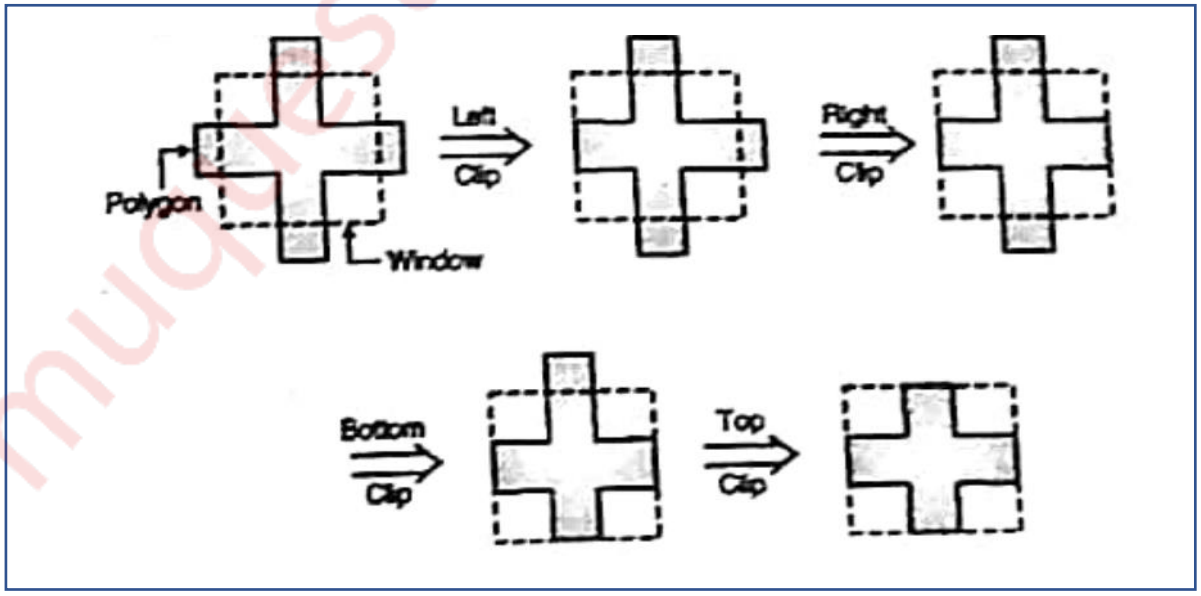
---

b) Explain any one polygon clipping algorithm. (10 M)

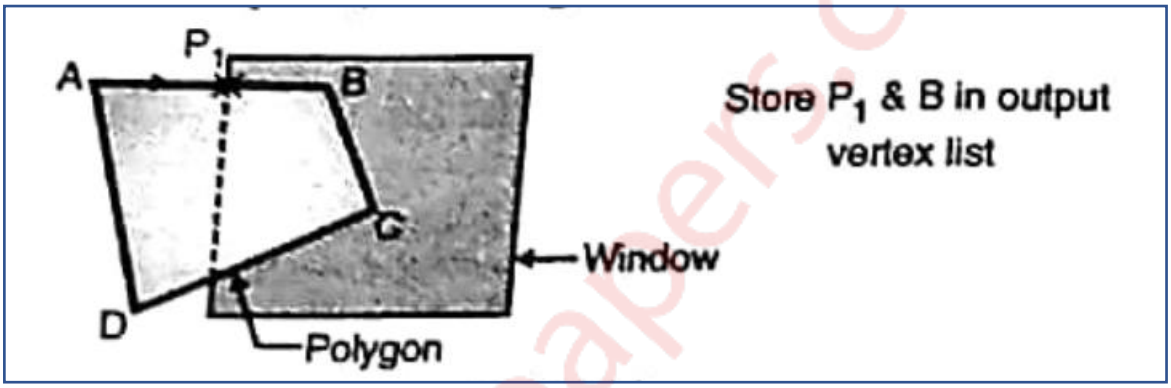
Ans:

Sutherland Hodgman polygon clipping algorithm:

- We can clip a polygon by clipping whole polygon against each boundary edge of the window. We know that to represent a polygon we need a set of vertices.
- This left clip procedure generates new set of vertices which indicates left clipped polygon. Against this new set of vertices is passed to the right boundary clipper procedure. Again we will get new set of vertices. Then we will pass this new set of vertices to bottom boundary clipper and lastly to top boundary clipper procedure.

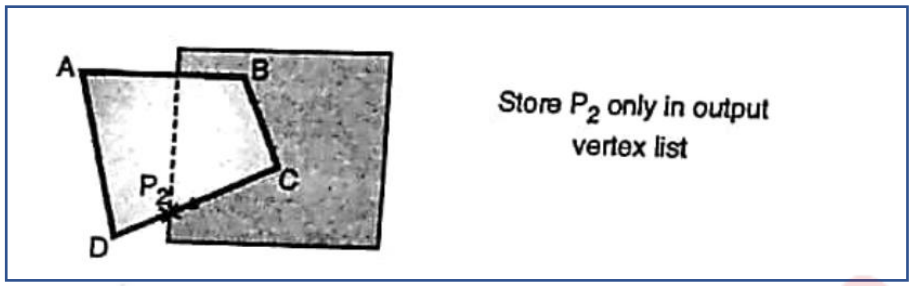


- At the end of every a new set of vertices is generated and this new set or modified polygon is passed to the next clipping stage.
- After clipping a polygon with respect to all four boundaries we will get final clipped polygon.
- When we clip a polygon w.r.t any particular edge of the window four different cases, as each pair of adjacent polygon vertices is passed to a window boundary clipper.
- **Case 1:** If the first vertex is outside the window boundary and the second vertex is inside the window, then the intersection point of polygon with boundary edge of window and the vertex which is inside the window is stored in a output vertex list.

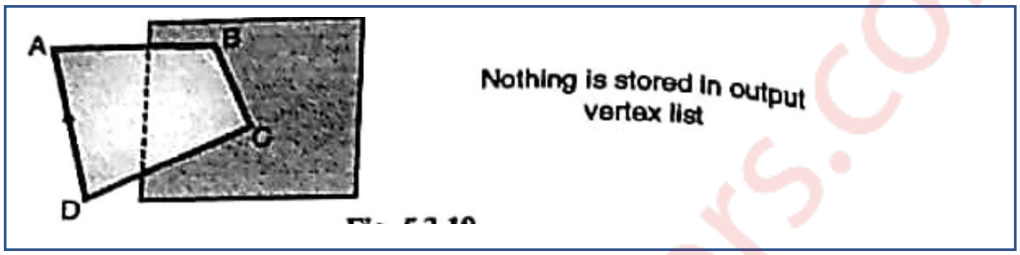


- For edge AB instead of storing vertex A and B are storing P_1 and B in output vertex list.
- **Case 2:** If both, first and second vertices of a polygon are lying inside the window, then we have to store the second vertex only in output vertex list.

- **Case 3:** If the first vertex is inside the window and second vertex is outside the window i.e. opposite to case 1, then we have to store only intersection point of that edge of polygon with window in output vertex list.



- **Case 4:** If both first and second vertices of a polygon are lying outside the window then no vertex is stored in vertex list.



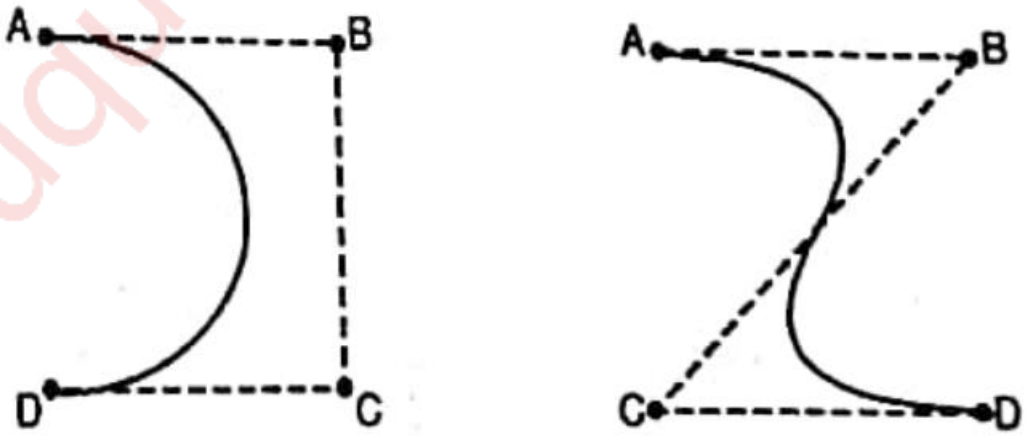
- Once all vertices have been considered for one clip window boundary, the output list of vertices is clipped against the next window boundary.

Q.5)

a) Explain Bezier curve with its properties and construct. (10 M)

Ans:

- It is a different way of specifying a curve, rather same shapes can be represented by B-spline and Bezier curves. The cubic Bezier curve require four sample points, these points completely specify the curve.



- The curve begins at the first sample point and ends at fourth point. If we need another Bezier curve then we need another four sample points. But if we need two Bezier curves connected to each other, then with six sample points we can achieve it. For this, the third and fourth point of first curve should be made same as first and second point of curve.
- The equation for the Bezier curve are as follows:

$$X = X_4a^3 + 3X_3a^2(1-a) + 3X_2a(1-a)^2 + X_1(1-a)^3$$

$$Y = Y_4a^3 + 3Y_3a^2(1-a) + 3Y_2a(1-a)^2 + Y_1(1-a)^3$$

$$Z = Z_4a^3 + 3Z_3a^2(1-a) + 3Z_2a(1-a)^2 + Z_1(1-a)^3$$

- Here as the value of 'a' moves from 0 to 1, the curve travels from the first to fourth sample point. But we can construct a Bezier curve without referencing to the above expression. It is constructed by simply taking midpoints.
- Properties of Bezier curve are as follows:
    1. The basic function are real in nature.
    2. Bezier curve always passes through the first and last control points i.e. curve has same end points as the guiding polygon.
    3. The degree of polynomial defining the curve segment is one less than the number of defining polygon point.
    4. The curve generally follows the shape of the defining polygon.
    5. The direction of the tangent vector at the end points is the same as that of the vector determined by first and last segments.
    6. The curve lies entirely within the convex hull formed by four control points.
    7. The curve exhibits the variation diminishing property. This means that the curve does not oscillate about any straight line more than the defining polygon.
    8. The curve is invariant under an affine transformation.

---

b) Explain Gourard and Phong Shading along with their advantages and disadvantages. (10 M)

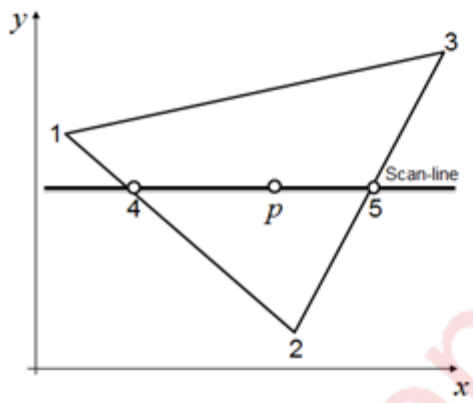
Ans:

Gouraud Shading:

1. Gouraud surface shading was developed in the 1970s by Henri Gouraud.
2. It is the interpolation technique.
3. Intensity levels are calculated at each vertex and interpolated across the surface.

4. Intensity values for each polygon are matched with the values of adjacent polygons along the common edges.
5. This eliminates the intensity discontinuities that can occur in flat shading.
6. To render a polygon, Gouraud surface rendering proceeds as follows:
    - Determine the average unit normal vector at each vertex of the polygon.
    - Apply an illumination model at each polygon vertex to obtain the light intensity at that position.
    - Linearly interpolate the vertex intensities over the projected area of the polygon

Illumination values are linearly interpolated across each scan-line as shown in figure 41.



$$I_4 = \frac{y_4 - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y_4}{y_1 - y_2} I_2$$

$$I_5 = \frac{y_5 - y_2}{y_3 - y_2} I_3 + \frac{y_3 - y_5}{y_3 - y_2} I_2$$

$$I_p = \frac{x_5 - x_p}{x_5 - x_4} I_4 + \frac{x_p - x_4}{x_5 - x_4} I_5$$

Figure 41

Advantages: Removal of discontinuities associated with the constant shading model.

Disadvantages: Highlighted surfaces are sometimes displayed with anomalous shape and the linear intensity interpolation can use bright or dark intensity interpolation can use bright or dark intensity strips. This effect can be reduced by sung phong shading method.

Phong Shading:

1. A more accurate interpolation based approach for rendering a polygon was developed by Phong Bui Tuong.
2. Basically the Phong surface rendering model is also called as normal-vector interpolation rendering.
3. It interpolates normal vectors instead of intensity values.
4. To render a polygon, Phong surface rendering proceeds as follows:
5. Determine the average unit normal vector at each vertex of the polygon.
6. Linearly interpolate the vertex normal over the projected area of the polygon.

7. Apply an illumination model at positions along scan lines to calculate pixel intensities using the interpolated normal vectors as shown in figure 42

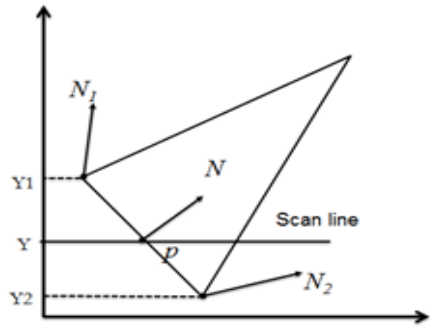


Figure 42

$$N = \frac{y - y_2}{y_1 - y_2} N_1 + \frac{y_1 - y}{y_1 - y_2} N_2$$

Advantages: It is very effective in dealing with specular highlights.

Disadvantages: This technique is relatively time-consuming since the illumination model is evaluated at every point using interpolated normal vectors.

Q.6) Write short notes on (20 M)

a) Depth Buffer method (10 M)

Ans:

- Another way to handle hidden liens and surfaces is z buffers. It is also called as depth buffer algorithm. Here we are sorting the polygons according to their position in space. And then in frame buffer itself. We are sorting polygons which are closer to viewer. We know that frame buffer is used to store images which we want to display on monitor. Here for visibility test we are making use of z buffer along with frame buffer.
- The z buffer is a large array to hold all the pixels of display z buffer is somewhat similar to frame buffer. In frame buffer we are having arrays to store x and y co-ordinates of an image. Similarly z buffer contains z co-ordinates of pixels which we want to display.
- When there is nothing to display on monitor i.e. frame buffer is empty, at that time we have t initialize z buffer elements to a very large negative values. A large negative values on z axis represents a point beyond which there is nothing i.e. setting background colour. $z_{buffer}(x, y) = z_{initialvalue,}$
- If the new surfaces has z value greater than z_{buffer} then it lies in front. So we have to modify the contents of $z_{buffer}(x, y)$ by new z values and set the pixel value at (x, y) to the colour of the polygon at (x, y).
  i.e. if ($z(x, y) > z_{buffer}(x, y)$)
  {

$z_{buffer}(x, y) = z(x, y)$
put pixel (x, y, polygon-colour)
}

- if the new value of new surface is smaller than $z_{buffer}(x, y)$ then it lies behind some polygon which was previously entered. So new surface should be hidden and should not be displayed. The frame buffer and z_{buffer} should not be modified here. Here the comparison should be carried out by using pixel by pixel method.
- **Advantages:**
  - It is easy to implement.
  - As z buffer algorithm processes one object at a time, total number of objects can be large.

- **Dis-advantages:**
  - It requires lots of memory as we are storing each pixels z value.
  - It is a time consuming process as we are comparing each and every pixel.

---

b) Halftone and Dithering techniques.

Ans:

Half toning

1. Many displays and hardcopy devices are bi-level
2. They can only produce two intensity levels.
3. In such displays or hardcopy devices we can create an apparent increase in the number of available intensity value.
4. When we view a very small area from a sufficient large viewing distance, our eyes average fine details within the small area and record only the overall intensity of the area.
5. The phenomenon of apparent increase in the number of available intensities by considering combine intensity of multiple pixels is known as half toning.
6. The half toning is commonly used in printing black and white photographs in newspapers, magazines and books.
7. The pictures produced by half toning process are called halftones.
8. In computer graphics, halftone reproductions are approximated using rectangular pixel regions, say 2*2 pixels or 3*3 pixels.
9. These regions are called halftone patterns or pixel patterns.

Dithering Techniques

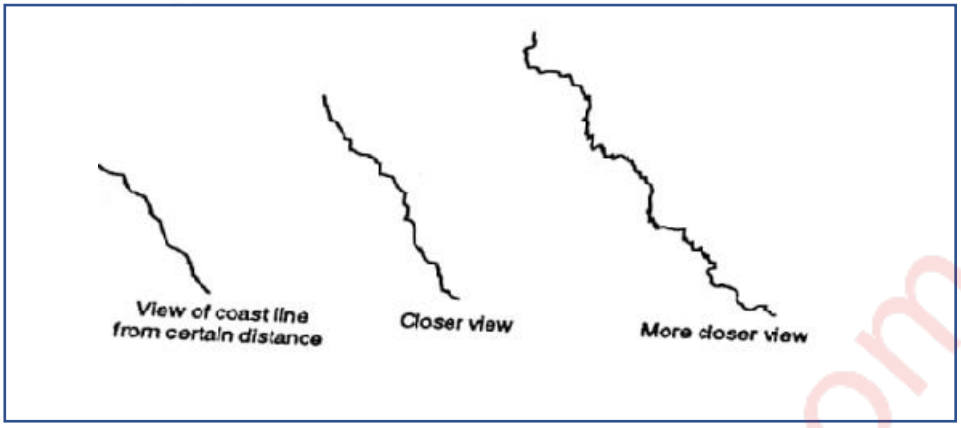
- Dithering refers to techniques for approximating halftones without reducing resolution, as pixel grid patterns do.
- The term dithering is also applied to halftone approximation method using pixel grid, and something it is used to refer to color halftone approximations only.
- Random values added to pixel intensities to break up contours are often referred as dither noise.
- Number of methods is used to generate intensity variations.
- Ordered dither methods generate intensity variations with a one-to-one mapping of points in a scene to the display pixel.
- To obtain n2n2 intensity levels, it is necessary to set up an n*n dither matrix DnDn whose elements are distinct positive integers in the range of 0 to n2−1n2−1.

---

c) Fractals

Ans:

- A fractal is defined as a rough or fragmented geometric shape that can be split into parts, each of which is approximately a reduced-size reproduction of the complete shape based on the property known as self- similarity. It was derived from the Latin word fractus which means broken or fractured. Natural objects can be realistically described using fractal geometry methods. Example.- cloud, mountains, trees, stone etc.
- Fractal methods use procedures rather than equations to model objects. so it uses procedural modelling. The major characteristic of any procedural model is that the model is not based on data ,but rather on the implementation of the procedure following a particular set of rules.
- A fractal combines the following characteristic:
- Its parts have the same form or structure as a whole, except that they are at a different scale and may be slightly deformed.
- Its form is extremely irregular or fragmented, and remains so, whatever the scale of examination.
- It is formed by iteration i.e. the procedure is used repeatedly(recursively)
- Example: if P0 = (X0, Y0, Z0) is a selected initial position, the successive levels P1 = F (P0), P2 = F (P1), …… Pn = F (Pn-1) are generated by a transformation function F.
- Fractional dimensions.



- Imagine that one object is made up of clay. If we break that objects in terms of line or line segments, then we will give the dimensions D_1= 1. If the object is broken into a plane, then we will give the dimension D_t = 2. And if the object is broken into 3D objects like cube, sphere etc. then we will give the transformation as D_t = 6. Here the variables D_t is called topological dimensions.

---

d) Koch curve

Ans:

- The Koch curve can be drawn by dividing line into 4 equal segments with scaling factor 1/3 and middle two segments are so adjusted that they form adjacent sides of an equilateral triangle as shown in the Fig. 16(a) .This is the first approximation to the koch curve.
- To apply the second approximation to the Koch curve we have to repeat the above process for each of the four segments. The resultant curve is shown in Fig. 16(b).

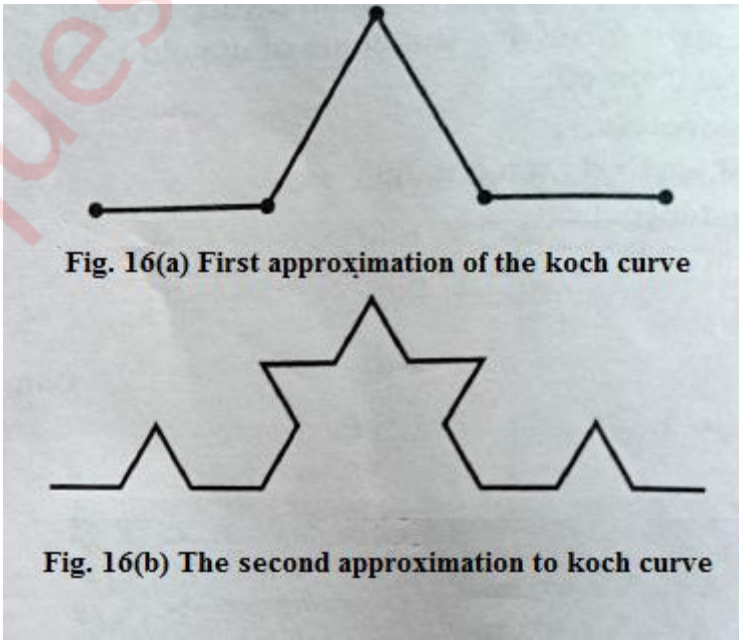


Fig. 16(a) First approximation of the koch curve

Fig. 16(b) The second approximation to koch curve

- The resultant curve has more wiggles and its length is 16/9 times the original length.
- From the above figures we can easily note following points about the koch curve :
    - Each repetition increases the length of the curve by factor 4/3.
    - Length of curve is infinite.
    - Unlike Hibert's curve, it doesn't fill an area.
    - It doesn't deviate much from its original shape.
    - If we reduce the scale of the curve by 3 we find the curve that looks just like the original one; but we must assemble 4 such curves to make the originals, so we have

$$4 = 3^D$$

Solving for D we get

$$D = \log_3 4 = \log 4 / \log 3 = 1.2618$$

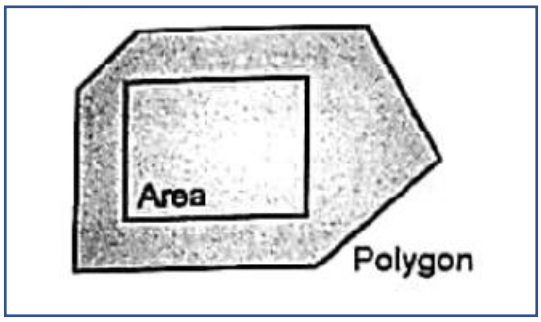
- Therefore for koch curve topological dimension is 1 but fractal dimension is 1.2618.
- From the above discussion we can say that point sets, curves and surfaces which give a fractal dimension greater than the topological dimension are called fractals The Hilbert's curve and koch curves are fractals, because their fractal dimensions (respectively, 2 and 1.2618) are greater than their topological dimension which is 1.

---

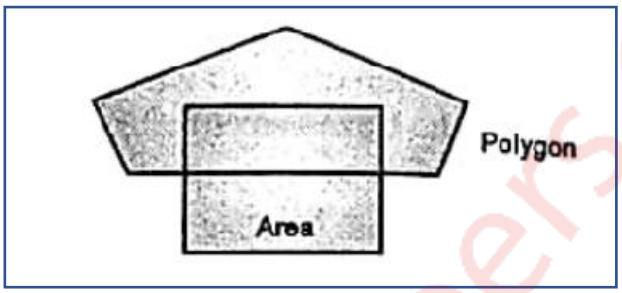
e) Area subdivision method

Ans:

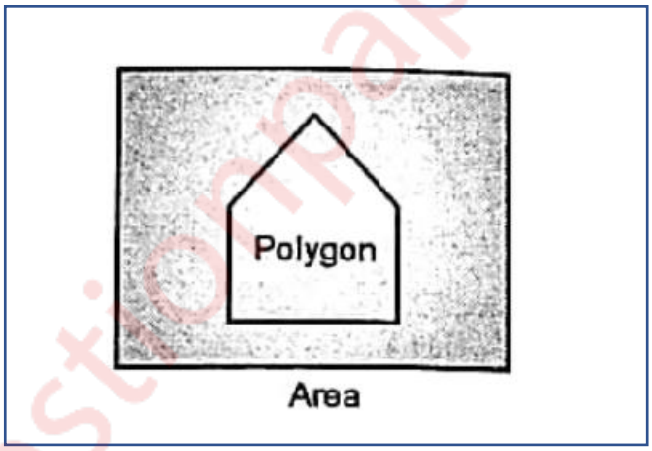
- The area-subdivision method takes advantage by locating those view areas that represent part of a single surface.
- Divide the total viewing area into smaller and smaller rectangles until each small area is the projection of part of a single visible surface or no surface at all.
- Continue this process until the subdivisions are easily analysed as belonging to a single surface or until they are reduced to the size of a single pixel.
- The algorithm is a recursive procedure based on a 2-step strategy.
    1. Decide which polygons overlap the given area on the screen.
    2. Which polygons are visible in that area.
- Categories of polygons are:
- **Surrounding polygon:** In this case a polygon completely covers the given screen area



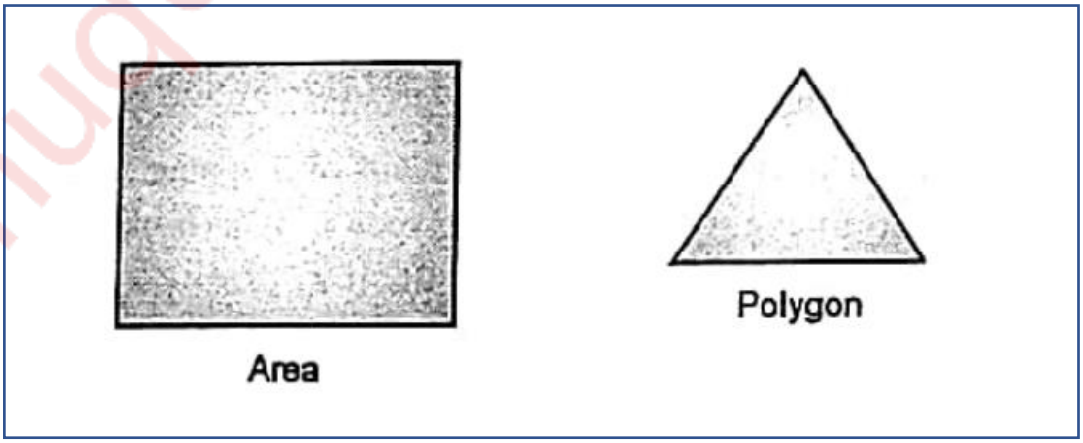
- **Intersection Polygon:** In this case a polygon is getting intersected with the screen area i.e. the polygon is partially inside the screen area.



- **Contained Polygon:** In this case a polygon is lying completely inside the given screen area.



- **Disjoint polygon:** Whenever the polygon is lying completely outside the given screen area, it is called as disjoint polygon.



COMPUTER GRAPHICS (MAY 2018)

Q.P.Code: 21848

Q.1) Attempt any five from the following:- (20 M)

a) Explain CSG method for solid modelling. (5 M)

Ans:

- Another technique for solid modelling is to combine the volumes occupied by overlapping three-dimensional objects using Boolean set operations.
- This modelling technique is called Constructive Solid Geometry (CSG). It creates a new volume by applying Boolean operators such as union. intersection, or difference to two specified objects.
- The Fig. (10), Fig. (11), Fig. (12) show the example for forming new shapes using Boolean set operations The Fig. 10 (a) shows that two rectangular blocks are placed adjacent to each other. We can obtain the combined object with the union operation as shown in Fig. 10 (b).

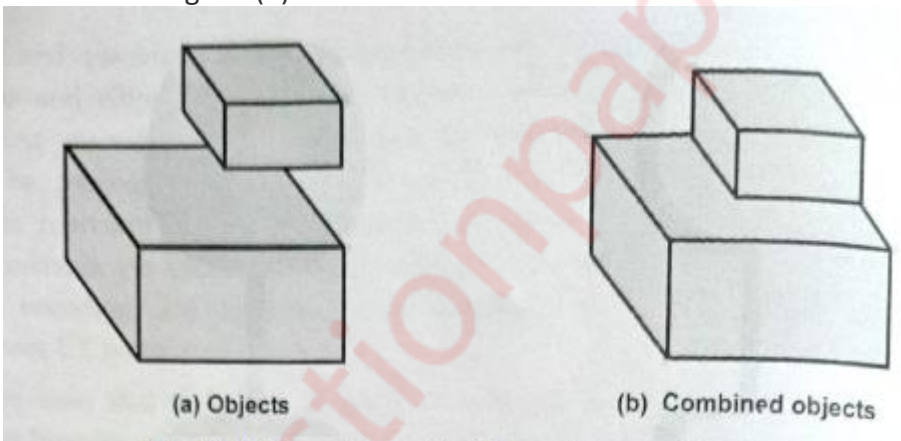


Fig. (10) Combined object by union operator.

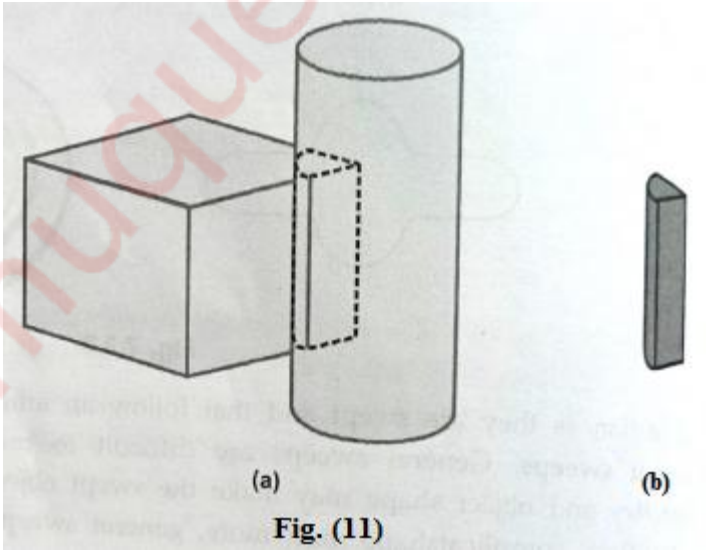


Fig. (11)

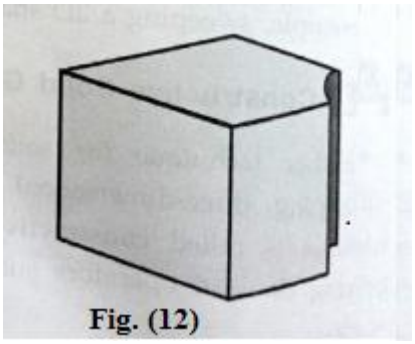


Fig. (12)

- The Fig.(11) shows the result of intersection operation obtained by overlapping cylinder and cube.
- With the difference operation, we can obtain the resulting solid as shown in Fig. (12).
- The CSG method Uses three dimensional objects such as blocks, pyramids, cylinders, cones, spheres, and closed spline surfaces to generate other solid objects In this method, an object is stored as a tree with operators at the internal nodes and simple primitives at the leaves.
- Some nodes represent Boolean operators, whereas others represent operations such as translation, rotation, and scaling. It is important to note that Boolean operations are not, in general, communicative Therefore the edges of the trees must be in proper order.

---

b) What is aliasing and Explain any one antialiasing method. (5 M)

Ans:

- In computer graphics, the process by which smooth curves and other lines become jagged because the resolution of the graphics device or file is not high enough to represent a smooth curve.
- In the line drawing algorithms, we have seen that all rasterized locations do not match with the true line and we have to select the optimum raster locations to represent a straight line. This problem is severe in low resolution screens. In such screens line appears like a stair-step, as shown in the figure below. This effect is known as aliasing. It is dominant for lines having gentle and sharp slopes.

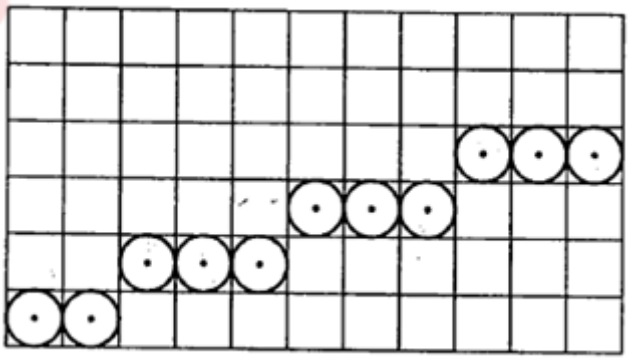


Fig. Aliasing effect

- The aliasing effect can be reduced by adjusting intensities of the pixels along the line. The process of adjusting intensities of the pixels along the line to minimize the effect of aliasing is called antialiasing.
- The technique of anti-aliasing is:
- **Prefiltering:** This is technique that determines pixel intensity based on the amount of that particular pixel coverage by the object in the scene i.e. It computes pixel colours depending on the objects coverage.
- It means how much part or fraction of that pixel is covered by the object and depending on that, it sets the value of the pixel. It requires large number of calculations and approximations. Prefiltering generates more accurate antialiasing effect. But due to its high complexity of calculations it is not used.

---

c) Compare Raster Scan and Random Scan displays. (5 M)

Ans:

Random Scan	Raster Scan
1. It has high resolution.	1. Its resolution is low
2. It is more expensive.	2. It is less expensive.
3. Any modification if needed is easy	3. Modification is tough.
4. solid pattern is tough to fill.	4. Solid pattern is easy to fill.
5. Refresh rate depends on resolution.	5. Refresh rate does not depend on the picture.
6. Only screen with view on an area is displayed.	6. Whole screen is scanned.
7. Beam penetration technology comes under it.	7. Shadow mark technology comes under this.
8. It does not use interlacing method.	8. It uses interlacing.

---

d) Prove that two successive rotations are additive i.e. R1(Θ_1)*R2(Θ_2) = R(Θ_1+Θ_2). (5 M)

Ans:

Lets assume that $\Theta_1 = \Theta_2 = 90^0$

It means that first we will perform rotation of Θ_1 i.e. 90^0 in anticlockwise direction and then on that resultant rotation matrix we will perform again anticlockwise rotation by angle Θ_2

By performing two times of rotation by Θ_1 and Θ_2 we will get results as if roation by (Θ_{1+} Θ_2).

$$R(\Theta_1) = \begin{vmatrix} \cos 90 & \sin 90 \\ -\sin 90 & \cos 90 \end{vmatrix} = \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix}$$

$$R(\Theta_2) = \begin{vmatrix} \cos 90 & \sin 90 \\ -\sin 90 & \cos 90 \end{vmatrix} = \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix}$$

Now if R(Θ_1)* R(Θ_2)

$$\text{then, } \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix} * \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix} = \begin{vmatrix} -1 & 0 \\ 0 & -1 \end{vmatrix}$$

now R(Θ_1 + Θ_2) i.e. R($90^0 + 90^0$) = R(180^0)

$$\text{hence, } R(180^0) = \begin{vmatrix} \cos 180 & \sin 180 \\ -\sin 180 & \cos 180 \end{vmatrix} = \begin{vmatrix} -1 & 0 \\ 0 & -1 \end{vmatrix}$$

Hence, R1(Θ_1)*R2(Θ_2) = R(Θ_1+Θ_2).

Q.2)

a) Explain Bresenham line drawing algorithm with proper mathematical analysis and identify the pixel positions along a line between A(10,10) and B(18,16) using it. (10 M)

Ans:

- There is one more algorithm to generate lines which is called as Bresenham's line algorithm. The distance between the actual line and the nearest grid location is called error and it is denoted by G.
- The beauty of Bresenham's algorithm it is implemented entirely with integer numbers and the integer numbers are much more faster than floating-point arithmetic. But in previous section we have seen that we are checking slope with 0.5 i.e. floating point variable.
- Consider P as height of pixel or error. Our condition is whether P>0.5 or not,
  If P > 0.5 ……(1)

- But here 0.5 is floating point so we have to convert floating point to integer. For that we will multiply both sides by 2.
  2P-1 > 0 …….(2)
- In this equation we have removed fractional part 0.5. But it still it may contain fractional part from the denominator of slope. Because every time we are updating P by,
  P = P + slope or P = P+ slope-1

  i.e. here we are adding slope to P.

  But slope is nothing but Dy/Dx.

- Now,
  2PDx – Dx > 0
- Now we will define G = 2PDx – Dx …….(3)
- As G = 2PDx – Dx

  G + Dx = 2PDx

  G + Dx / 2Dx = P

- Now if we consider P = P + slope, then
  We will get G + Dx / 2Dx = G + Dx / 2Dx + Dy/Dx
  G = G + 2Dy ……..(4)
- Similarly if we have P = P + slope – 1
  We will get G = G + 2Dy – 2Dx …….(5)
- So, we can use test G > 0 to determine when a row boundary Is crossed by a line instead of P > 0.5
- Put initial value of P as Dy/Dx to find initial value of G.
  G = 2Dx (Dy / Dx) – Dx
  G = 2Dy – Dx
  - For each column we check G. If it is we move to the next row and add (2 Dy – 2 Dx) to G, because it is equivalent to P = P + slope -1, otherwise we will keep the same row and add (2 Dy) to G.

  - Given A(10, 10) B(18, 16)

    A(x1, y1) = (10, 10)
    B(x2, y2) = (18, 16)
    Dx = x2-x1 = 18 – 10 = 8
    Dy = y2-y1 = 16 – 10 = 6
    Now, lets find decision factor G,
    G = 2Dy – Dx = 2(6) – 8 = 4
    First plot point (18, 16)
    Now increase x by 1

Hence, x = 19
Here G = 4 i.e. positive so increase y by 1 and update G
As, G = G+2(Dy - Dx)
= 4+2(6 - 8) = 0
Now plot point (19, 17)
Similarly finding different values,

A	B
18	16
19	17
20	18
21	18
22	19
23	20
24	21
25	21
26	22
27	23
28	24

---

b) Explain the steps for 2D rotation about arbitrary point and provide a composite transformation for the same. (10M)

Ans:

- **Rotation about an Arbitrary Point:-**
- To rotate an object about an arbitrary point, (Xp ,Yp) we have to carry out three steps:
    - Translate point (Xp, Yp) to the origin.
    - Rotate it about the origin and,
    - Finally, translate the centre of rotation back where it belongs (See figure 1.).
- we have already seen that matrix multiplication is not commutative, i.e. multiplying matrix A by matrix B will not always yield the same result as multiplying matrix B by matrix A. Therefore, in obtaining composite transformation matrix, we must be careful to order the matrices so that they correspond to the order of the transformations on the object. Let us find the transformation matrices to carry out individual steps.

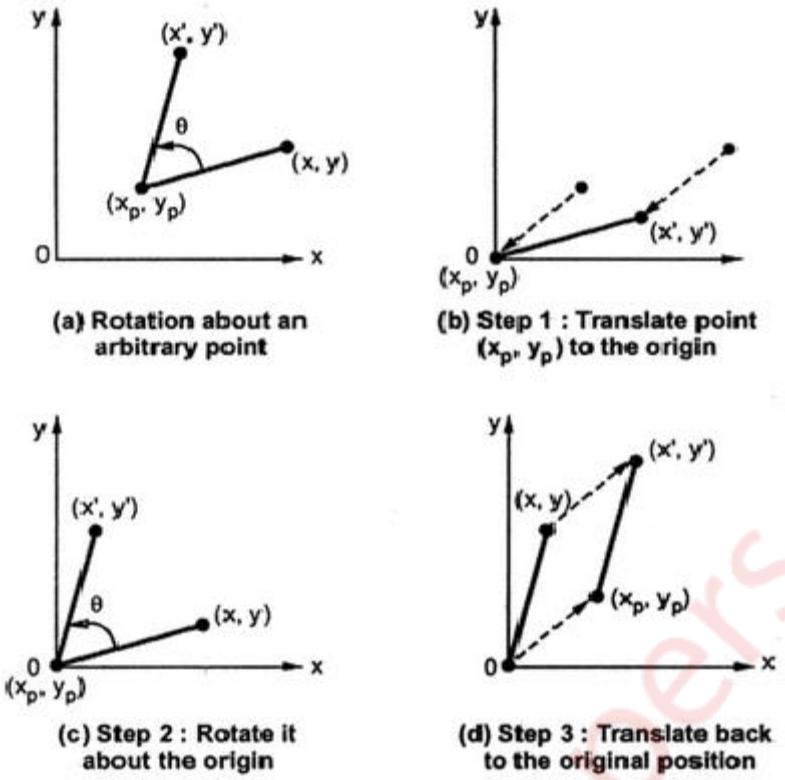


Fig. 1

The translation matrix to move point (x_p, y_p) to the origin is given as, x_p

$$T_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\text{xp} & -yp & 1 \end{bmatrix}$$

The rotation matrix for counterclockwise rotation of point about the origin is given as,

$$R = \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The translation matrix to move the center point back to its original position is given as,

$$T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

Therefore the overall transformation matrix for a counterclockwise roatation by an angle Θ about the point (xp, yp) is given as,

$$T_1*R*T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xp & -yp & 1 \end{bmatrix} * \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

$$= \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ -xpcos\Theta + \text{ypsin}\Theta & -xpsin\Theta - \text{ypcos}\Theta & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

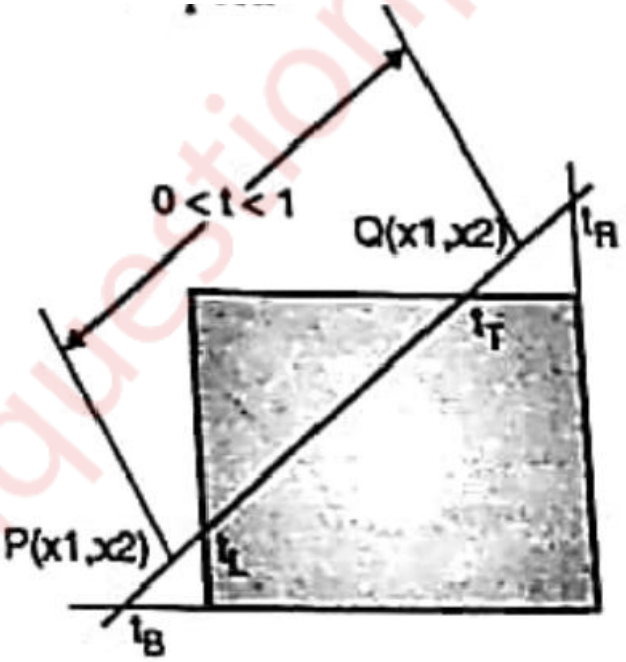
$$= \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ -xpcos\Theta + ypsin\Theta + xp & -xpsin\Theta - ypcos\Theta + yp & 1 \end{bmatrix}$$

Q.3)

a) Explain Liang Barsky line clipping algorithm. Apply the algorithm to clip the line with coordinates(30, 60) and (60, 20) against window (xmin, ymin) = (10, 10) and (xmax, ymax) = (50, 50). (10 M)

Ans:

- The Liang-Barsky algorithm uses the parametric equation of a line and inequalities describing the range of the clipping box to determine the intersections between the line and clipping box. With these intersections it knows which portion of the line should be drawn. This algorithm is more efficient than Cohen-Sutherland. The ideas for clipping line of Liang-Barsky and cyrus-Beck are the same. The only difference is Liang-Barsky algorithm has been optimized for an upright rectangular clip window. So we will study only the idea of Liang-Barsky.
- Liang and Barsky have created an algorithm that uses floating-point arithmetic but finds the appropriate ends points with at most four computations. This algorithm uses the parametric equations for a line and solves for inequalities to find the range of the parameter for which the line is in the viewport.



- Let P(X1, Y1), Q(X2, Y2) be the line which we want to study. The parametric equation of line segment from gives x-values and y-values for every point in terms of a parameter that ranges from 0 to 1. The equations are

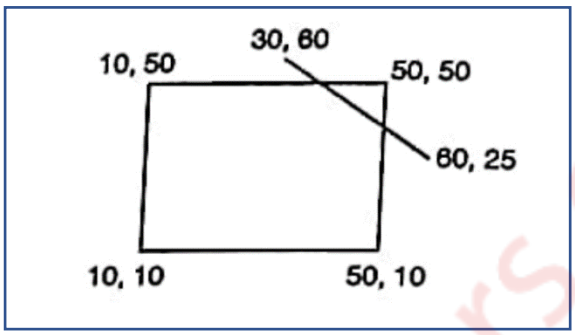
  X = X1 + (X2 – X1)*t = Y1 + dY*t

  And

$$Y = Y1 + (Y2 – Y1)*t = X1 + dX*t$$

We can see that when t=0, the point computed is P(x1, y1) and when t =1, the point computed is Q(x2, y2).

- Lets first draw the window and line as shown in fig.



Given things are XL = 10, YB = 10, XR = 50, YT = 50

Lets call a line AB with its coordinates as X1 = 30, Y1 = 60, X2 = 60, Y2 = 25

Now we have to find Dx and Dy as

$$Dx = X2 – X1 = 60 – 30 = 30$$

$$Dy = Y2 – Y1 = 25 – 60 = -35$$

Lets calculate the values of parameters P and Q as

$$P1 = -Dx = -30$$

$$P2 = Dx = 30$$

$$P3 = -Dy = -(-35) = 35$$

$$P4 = Dy = -35$$

Now,

$$Q1 = X1 – XL = 30 -10 = 20$$

$$Q2 = XR – X2 = 50 -30 = 20$$

$$Q1 = Y1 – YB = 30 -10 = 50$$

$$Q1 = YT – Y1 = 30 -10 = -10$$

Now lets find P,

$$P1 = Q1/P1 = 20/(-30) = (-2/3)$$

P2 = Q2/P2 = 20/(30) = (2/3)

P3 = Q3/P3 = 50/(35) = (20/7)

P1 = Q4/P4 = -10/(-35) = (2/7)

Now,

t1 = Max(-2/3, 10/7, 0) = 10/7

t2 = Min(2/3, 2/7, 1) = 2/7

Now,

X1' = X1 + Dx*t1

= 30+30*(10/7)

= 72.71

Y1' = Y1 + Dy*t1

= 60+(-35)*(10/7)

= 10

And with t2 it will be

X2' = X1 + Dx*t2

= 30+30*(2/7)

= 38.57

Y2' = Y1 + Dy*t2

= 60+(-35)*(2/7)

= 50

From this we will come to know that a point (38.57, 50) is an intersection point with respect to the edge of the window boundary. So we need to discard the line from point (30, 60) to (38.57, 50) and consider line (38.57, 50) to (60, 25).

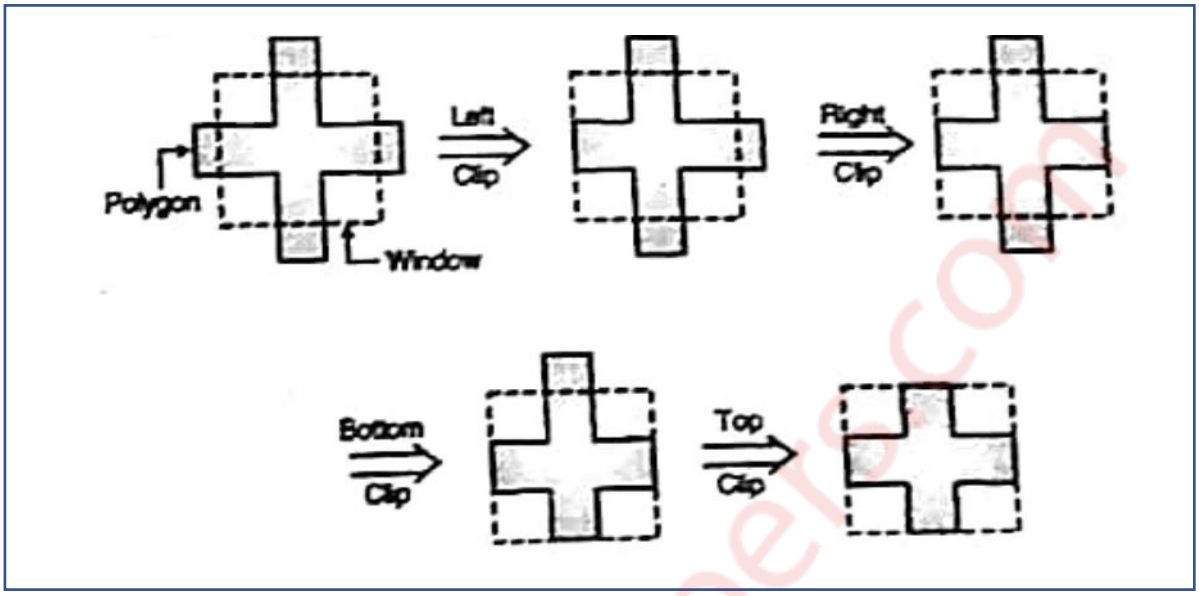
---

b) Explain Sutherland Hodgman polygon clipping algorithm with suitable example and comment on its shortcoming. (10 M)

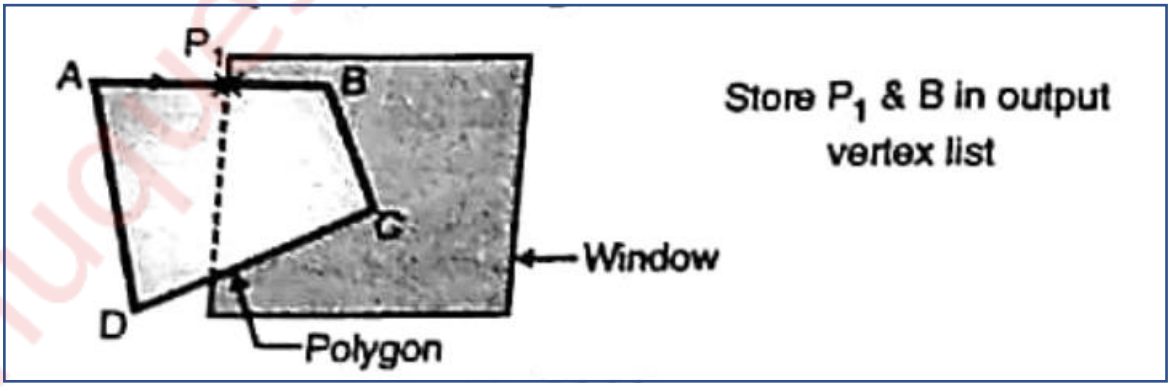
Ans:

- We can clip a polygon by clipping whole polygon against each boundary edge of the window. We know that to represent a polygon we need a set of vertices.
- This left clip procedure generates new set of vertices which indicates left clipped polygon. Against this new set of vertices is passed to the right boundary clipper

procedure. Again we will get new set of vertices. Then we will pass this new set of vertices to bottom boundary clipper and lastly to top boundary clipper procedure.



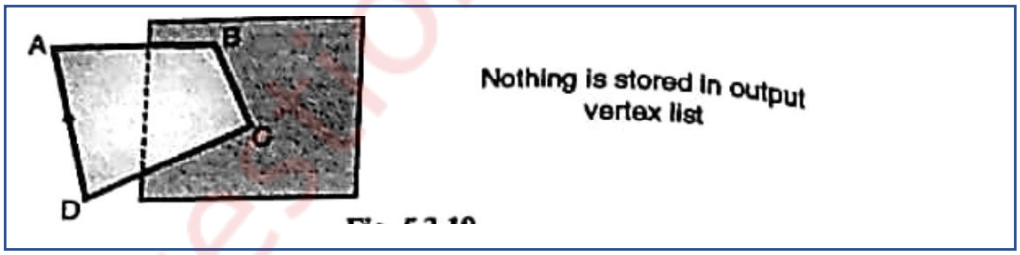
- At the end of every a new set of vertices is generated and this new set or modified polygon is passed to the next clipping stage.
- After clipping a polygon with respect to all four boundaries we will get final clipped polygon.
- When we clip a polygon w.r.t any particular edge of the window four different cases, as each pair of adjacent polygon vertices is passed to a window boundary clipper.
- **Case 1:** If the first vertex is outside the window boundary and the second vertex is inside the window, then the intersection point of polygon with boundary edge of window and the vertex which is inside the window is stored in a output vertex list.



- For edge AB instead of storing vertex A and B are storing P_1 and B in output vertex list.
- **Case 2:** If both, first and second vertices of a polygon are lying inside the window, then we have to store the second vertex only in output vertex list.

- **Case 3:** If the first vertex is inside the window and second vertex is outside the window i.e. opposite to case 1, then we have to store only intersection point of that edge of polygon with window in output vertex list.

- **Case 4:** If both first and second vertices of a polygon are lying outside the window then no vertex is stored in vertex list.



- Once all vertices have been considered for one clip window boundary, the output list of vertices is clipped against the next window boundary.

Q.4)

a) What is window and viewport? Derive the window to viewport transformation and also identify the geometric transformation involved. (10 M)

Ans:

Window:

1. A world-coordinate area selected for display is called a window.
2. In computer graphics, a window is a graphical control element.
3. It consists of a visual area containing some of the graphical user interface of the program it belongs to and is framed by a window decoration.
4. A window defines a rectangular area in world coordinates. You define a window with a GWINDOW statement. You can define the window to be larger than, the same size as, or smaller than the actual range of data values, depending on whether you want to show all of the data or only part of the data.

Viewport:

1. An area on a display device to which a window is mapped is called a viewport.
2. A viewport is a polygon viewing region in computer graphics. The viewport is an area expressed in rendering-device-specific coordinates, e.g. pixels for screen coordinates, in which the objects of interest are going to be rendered.
3. A viewport defines in normalized coordinates a rectangular area on the display device where the image of the data appears. You define a viewport with the GPORT command. You can have your graph take up the entire display device or show it in only a portion, say the upper-right part.

Window to viewport transformation:

1. Window-to-Viewport transformation is the process of transforming a two-dimensional, world-coordinate scene to device coordinates.
2. In particular, objects inside the world or clipping window are mapped to the viewport. The viewport is displayed in the interface window on the screen.
3. In other words, the clipping window is used to select the part of the scene that is to be displayed. The viewport then positions the scene on the output device.

1. Using this proportionality, the following ratios must be equal.

$$\frac{xv - xv_{min}}{xv_{max} - xv_{min}} = \frac{xw - xw_{min}}{xw_{max} - xw_{min}}$$

By solving these equations for the unknown viewport position (xv, yv), the following becomes true:

$$xv = S_x xw + t_x$$

$$yv = S_y yw + t_y$$

The scale factors (Sx, Sy) would be:

$$S_x = \frac{xv_{min} - xv_{min}}{xw_{max} - xw_{min}}$$

$$S_y = \frac{yv_{min} - yv_{min}}{yw_{max} - yw_{min}}$$

And the translation factors (Tx, Ty) would be:

$$t_x = \frac{xw_{max}xv_{min} - xw_{min}xv_{max}}{xw_{max} - xw_{min}}$$

$$t_y = \frac{yw_{max}yv_{min} - yw_{min}yv_{max}}{yw_{max} - yw_{min}}$$

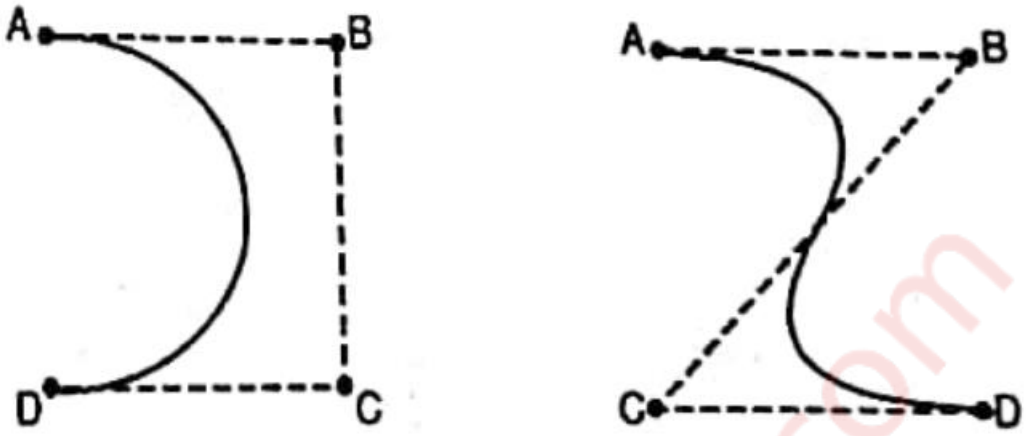
1. The position of the viewport can be changed allowing objects to be viewed at different positions on the Interface Window.
2. Multiple viewports can also be used to display different sections of a scene at different screen positions. Also, by changing the dimensions of the viewport, the size and proportions of the objects being displayed can be manipulated.
3. Thus, a zooming affect can be achieved by successively mapping different dimensioned clipping windows on a fixed sized viewport.
4. If the aspect ratio of the world window and the viewport are different, then the image may look distorted.

---

b) Explain what is meant by Bezier curve? State the various properties of Bezier curve. (10 M)

Ans:

- It is a different way of specifying a curve, rather same shapes can be represented by B-spline and Bezier curves. The cubic Bezier curve require four sample points, these points completely specify the curve.



- The curve begins at the first sample point and ends at fourth point. If we need another Bezier curve then we need another four sample points. But if we need two Bezier curves connected to each other, then with six sample points we can achieve it. For this, the third and fourth point of first curve should be made same as first and second point of curve.
- The equation for the Bezier curve are as follows:

$$X = X_4a^3 + 3X_3a^2(1-a) + 3X_2a(1-a)^2 + X_1(1-a)^3$$

$$Y = Y_4a^3 + 3Y_3a^2(1-a) + 3Y_2a(1-a)^2 + Y_1(1-a)^3$$

$$Z = Z_4a^3 + 3Z_3a^2(1-a) + 3Z_2a(1-a)^2 + Z_1(1-a)^3$$

- Here as the value of 'a' moves from 0 to 1, the curve travels from the first to fourth sample point. But we can construct a Bezier curve without referencing to the above expression. It is constructed by simply taking midpoints.
- Properties of Bezier curve are as follows:
    1. The basic function are real in nature.
    2. Bezier curve always passes through the first and last control points i.e. curve has same end points as the guiding polygon.
    3. The degree of polynomial defining the curve segment is one less than the number of defining polygon point.
    4. The curve generally follows the shape of the defining polygon.
    5. The direction of the tangent vector at the end points is the same as that of the vector determined by first and last segments.
    6. The curve lies entirely within the convex hull formed by four control points.
    7. The curve exhibits the variation diminishing property. This means that the curve does not oscillate about any straight line more than the defining polygon.
    8. The curve is invariant under an affine transformation.

Q.5)

a) What is meant by parallel and perspective projection? Derive matrix for oblique projection. (10 M)

Ans:

Parallel Projection:

i. In parallel projection, Z coordinate is discarded and parallel lines from each vertex on the object are extended until they intersect the view plane.

ii. The point of intersection is the projection of the vertex.

iii. We connect the projected vertices by line segments which correspond to connections on the original object.

iv. A parallel projection preserves relative proportions of objects.

v. Accurate views of the various sides of an object are obtained with a parallel projection. But not a realistic representation.

vi. Parallel projection is shown below in figure 30.

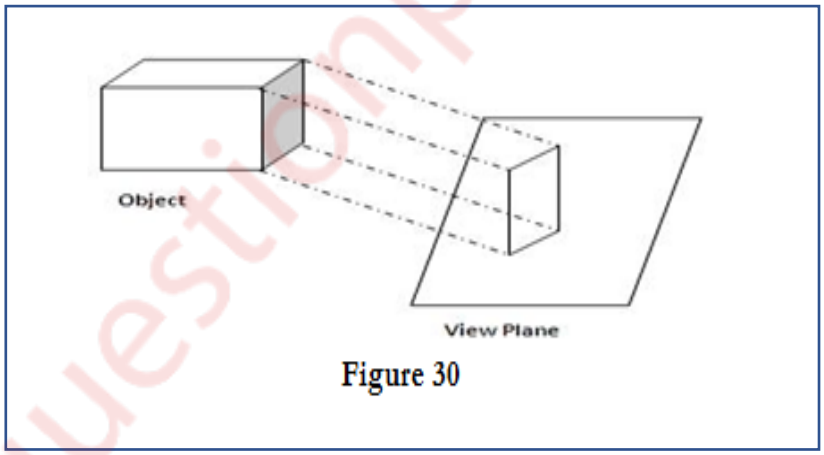


Figure 30

Perspective Projection:

i. In perspective projection, the lines of projection are not parallel.

ii. Perspective Projection transforms object positions to the view plane while converging to a center point of projection.

iii. In this all the projections are converge at a single point called the “center of projection” or “projection reference point”.

iv. Perspective projection produces realistic views but does not preserve relative proportions.

v. Projections of distant objects are smaller than the projections of objects of the same size that are closer to the projection plane.

vi. Perspective projection is shown below in figure 31.

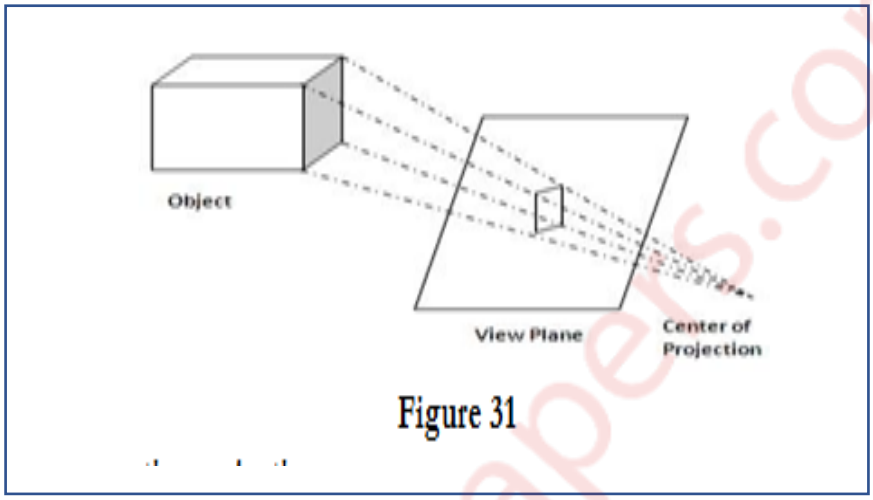
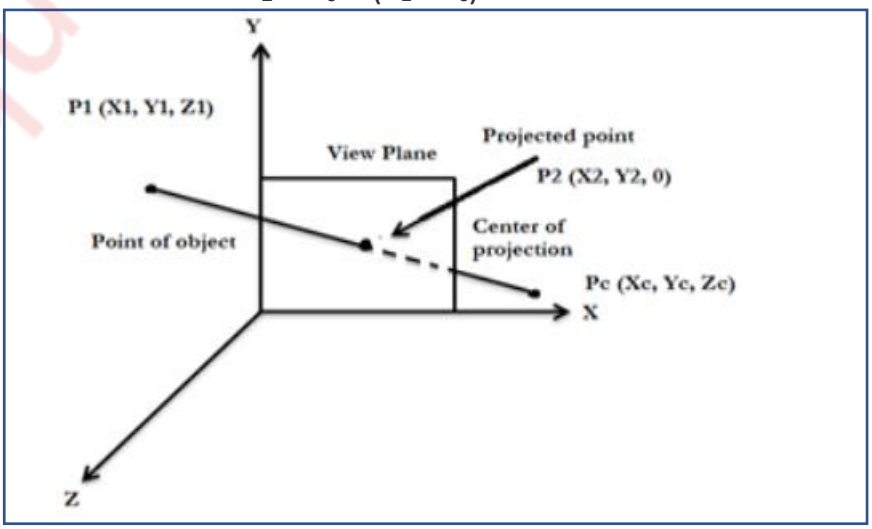


Figure 31

Matrix for perspective projection:

Let us consider the center of projection is at $P_c(X_c,Y_c,Z_c)$Pc(Xc,Yc,Zc) and the point on object is $P_1(X_1,Y_1,Z_1)$P1(X1,Y1,Z1), then the parametric equation for the line containing these points can be given as

$$X_2 = X_c + (X_1 - X_c)U$$
$$Y_2 = Y_c + (Y_1 - Y_c)U$$
$$Z_2 = Z_c + (Z_1 - Z_c)U$$



For projected point Z2 is 0, therefore the third equation can be written as

$0 = Z_C + (Z_1 - Z_C)U$

$U = -Z_C/Z_1 – Z_2$

Substituting the value of U in first two equations we get,

$X_2 = (X_C - Z_C) * (X_1 - X_C) / (Z_1 - Z_C)$

$= X_CZ_1 – X_CZ_C – X_1Z_C + X_CZ_C/Z_1 – Z_C$

$= X_CZ_1 – X_1Z_C / Z_1 – Z_C$

$Y_2 = (Y_C - Z_C) * (Y_1 - Y_C) / (Z_1 - Z_C)$

$= Y_CZ_1 – Y_CZ_C – Y_1Z_C + Y_CZ_C / Z_1 - Z_C$

The above equation can be represeented in the homogeneous matrix form as given below,

$$[X_2\ Y_2\ Z_2\ 1] = [X_1\ Y_1\ Z_1\ 1] \begin{bmatrix} -ZC & 0 & 0 & 0 \\ 0 & -ZC & 0 & 0 \\ XC & YC & 0 & 1 \\ 0 & 0 & 0 & -ZC \end{bmatrix} \begin{bmatrix} -ZC & 0 & 0 & 0 \\ 0 & -ZC & 0 & 0 \\ XC & YC & 0 & 1 \\ 0 & 0 & 0 & -ZC \end{bmatrix}$$

Here, we have taken the center of projection as PC(XC, YC, ZC), if we take the center of projection on the negative Z axis such that

X = 0

Y = 0

$Z = -Z_C$

---

b) Explain Z buffer algorithm for hidden surface removal. (10 M)

Ans:

- Another way to handle hidden lines and surfaces is z buffers. It is also called as depth buffer algorithm. Here we are sorting the polygons according to their position in space. And then in frame buffer itself. We are storing polygons which are closer to viewer. We know that frame buffer is used to store the images which we want to display on monitor. Here for visibility test we are making use of z buffer along with frame buffer.
- The z buffer is a large array to hold all the pixels off display. Z buffer is somewhat similar to frame buffer. In frame buffer we are having arrays to store x and y co-ordinates of an image. Similarly z buffer contains z co-ordinates of pixels which we want to display. It helps to sort the polygons by comparing their z position. In

simple terms it keeps track of nearest z coordinate of those points which are seen from the pixel (x, y).

- When there is nothing to display on monitor i.e. frame buffer is empty, at that time we have to initialize z buffer elements to a v. A large negative value on z axis represents a point beyond which there is nothing i.e. setting background color.

  $Z_{buffer}(x, y) = z_{initial}$

- Polygons will be entered one by one into frame buffer by the display file interpreter. During this process, for each pixel (x, y) that lies inside the polygon we have to calculate z (x, y) for the pixel (x, y). Then we have to compare the z position of the polygon point with the z_{buffer} (x, y) value and decide whether the new surface is in front of or behind the current contents of frame buffer.
- If the new surface has z value greater than z_{buffer} value, then it lies in front. So we have to modify the contents of z_{buffer} (x, y) by new z value and set the pixel value at (x, y) to the color of the polygon at (x, y).

  i.e. if($z(x,y) > z_{buffer}(x,y)$)
  {
  $z_{buffer}(x,y) = z(x,y)$
  put pixel (x,y,polygon-colour)
  }

- If the z value of new surface is smaller than z_{buffer} (x, y) then it lies behind some polygon which was previously entered. So the new surface should be hidden and should not be displayed. The frame buffer and z_{buffer} should not be modified here. Here the comparison should be carried out by using pixel by pixel method.

Advantages:

- It is easy to implement.
- As z buffer algorithm processes one object at a time, total number of objects can be large.
- It does not require any additional technique.

Disadvantages:

- It requires lots of memory as we are storing each pixel's z value.
- It is a time consuming process as we are comparing each and every pixel.

Q.6) Write short notes on (20 M)

a) Koch Curve: (5 M)

Ans:

- The Koch curve can be drawn by dividing line into 4 equal segments with scaling factor 1/3 and middle two segments are so adjusted that they form adjacent sides of an equilateral triangle as shown in the Fig. 16(a) .This is the first approximation to the koch curve.
- To apply the second approximation to the Koch curve we have to repeat the above process for each of the four segments. The resultant curve is shown in Fig. 16(b).

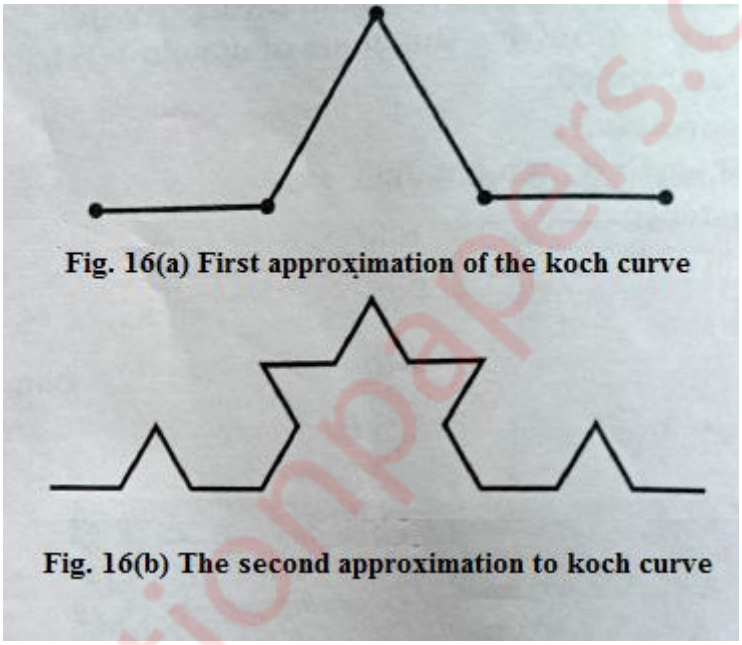


Fig. 16(a) First approximation of the koch curve

Fig. 16(b) The second approximation to koch curve

- The resultant curve has more wiggles and its length is 16/9 times the original length.
- From the above figures we can easily note following points about the koch curve :
    - Each repetition increases the length of the curve by factor 4/3.
    - Length of curve is infinite.
    - Unlike Hibert's curve, it doesn't fill an area.
    - It doesn't deviate much from its original shape.
    - If we reduce the scale of the curve by 3 we find the curve that looks just like the original one; but we must assemble 4 such curves to make the originals, so we have

$$4 = 3^D$$

Solving for D we get

$$D = \log_3 4 = \log 4 / \log 3 = 1.2618$$

- Therefore for koch curve topological dimension is 1 but fractal dimension is 1.2618.
- From the above discussion we can say that point sets, curves and surfaces which give a fractal dimension greater than the topological dimension are called fractals The Hilbert's curve and koch curves are fractals, because their fractal dimensions (respectively, 2 and 1.2618) are greater than their topological dimension which is 1.

---

b) Sweep representation and Octree representation (5 M)

Ans:

- Sweep representations are used to construct three dimensional objects from two dimensional shape .There are two ways to achieve sweep: Translational sweep and Rotational sweep. In translational sweeps, the 2D shape is swept along a linear path normal to the plane of the area to construct three dimensional object. To obtain the wireframe representation we have to replicate the 2D shape and draw a set of connecting lines in the direction of shape, as shown in the figure (8)

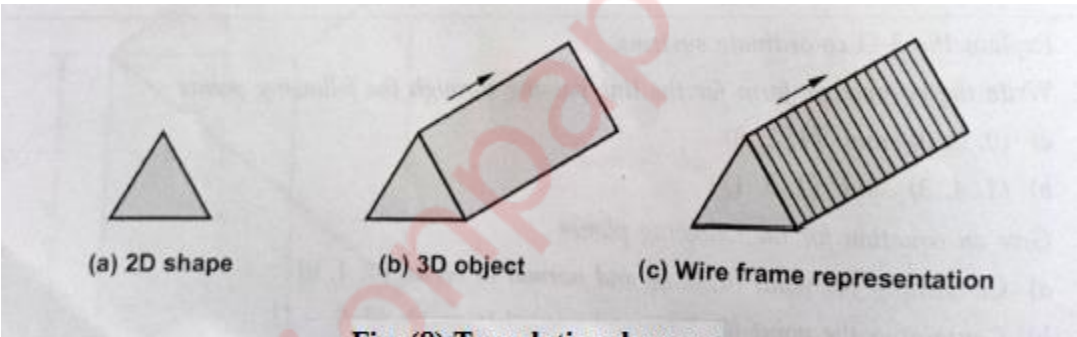


Fig. (8) Translational sweep

- In general we can specify sweep constructions using any path. For translation we can vary the shape or size of the original 2D shape along the sweep path. For rotational sweeps, we can move along a circular path through any angular distance from 0° to 360°.
- These sweeps whose generating area or volume changes in size, shape or orientation as they are swept and that follow an arbitrary curved trajectory are called general sweeps .General sweeps are difficult to model efficiently for example, the trajectory and object shape may make the swept object intersect itself, making volume calculations complicated.
- **Octree Representation:** Octrees are the hierarchical structure used to represent solid objects in some graphics systems. The tree structure of octrees is organized so that each node corresponds to a region of three dimensional space.
- Octrees are in turn derived from quad-trees, an encoding scheme used for representing two dimensional images. The fundamental idea behind both the quad-tree and octree is the divide and conquer power of binary subdivision. A quad-tree is derived successively dividing a 2D plane into quadrants. After first

subdivision, each node in the quad-tree has four data elements, one for each of the quadrants in the region. These quadrants can be of two types homogeneous and heterogeneous.

- The quadrant is said to be homogeneous quadrant the colour information for that quadrant is stored in the corresponding data element of the node. In addition, a flag is set in the data element to indicate that the quadrant is homogeneous.
- The heterogeneous quadrant contains pixels of different colours. If some of them are heterogeneous then they are again subdivided and the process is repeated until all quadrants are homogeneous or until a predetermined cut-off depth is reached.

---

c) Gouraud and phong shading (5 M)

Ans:

Gouraud Shading:

1. Gouraud surface shading was developed in the 1970s by Henri Gouraud.
2. It is the interpolation technique.
3. Intensity levels are calculated at each vertex and interpolated across the surface.
4. Intensity values for each polygon are matched with the values of adjacent polygons along the common edges.
5. This eliminates the intensity discontinuities that can occur in flat shading.
6. To render a polygon, Gouraud surface rendering proceeds as follows:
    - Determine the average unit normal vector at each vertex of the polygon.
    - Apply an illumination model at each polygon vertex to obtain the light intensity at that position.
    - Linearly interpolate the vertex intensities over the projected area of the polygon

Illumination values are linearly interpolated across each scan-line as shown in figure 41.

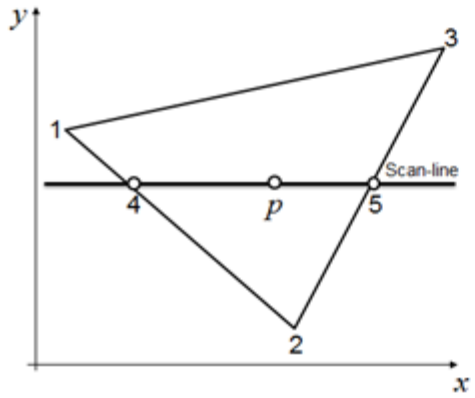


Figure 41

$$I_4 = \frac{y_4 - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y_4}{y_1 - y_2} I_2$$

$$I_5 = \frac{y_5 - y_2}{y_3 - y_2} I_3 + \frac{y_3 - y_5}{y_3 - y_2} I_2$$

$$I_p = \frac{x_5 - x_p}{x_5 - x_4} I_4 + \frac{x_p - x_4}{x_5 - x_4} I_5$$

Phong Shading:

1. A more accurate interpolation based approach for rendering a polygon was developed by Phong Bui Tuong.
2. Basically the Phong surface rendering model is also called as normal-vector interpolation rendering.
3. It interpolates normal vectors instead of intensity values.
4. To render a polygon, Phong surface rendering proceeds as follows:
5. Determine the average unit normal vector at each vertex of the polygon.
6. Linearly interpolate the vertex normal over the projected area of the polygon.
7. Apply an illumination model at positions along scan lines to calculate pixel intensities using the interpolated normal vectors as shown in figure 42

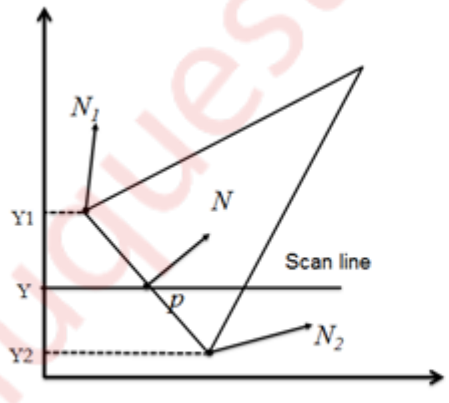


Figure 42

$$N = \frac{y - y_2}{y_1 - y_2} N_1 + \frac{y_1 - y}{y_1 - y_2} N_2$$

d) Halftoning and Dithering. (5 M)

Ans:

Half toning

1. Many displays and hardcopy devices are bi-level
2. They can only produce two intensity levels.
3. In such displays or hardcopy devices we can create an apparent increase in the number of available intensity value.
4. When we view a very small area from a sufficient large viewing distance, our eyes average fine details within the small area and record only the overall intensity of the area.
5. The phenomenon of apparent increase in the number of available intensities by considering combine intensity of multiple pixels is known as half toning.
6. The half toning is commonly used in printing black and white photographs in newspapers, magazines and books.
7. The pictures produced by half toning process are called halftones.
8. In computer graphics, halftone reproductions are approximated using rectangular pixel regions, say 2*2 pixels or 3*3 pixels.
9. These regions are called halftone patterns or pixel patterns.

Dithering Techniques

- Dithering refers to techniques for approximating halftones without reducing resolution, as pixel grid patterns do.
- The term dithering is also applied to halftone approximation method using pixel grid, and something it is used to refer to color halftone approximations only.
- Random values added to pixel intensities to break up contours are often referred as dither noise.
- Number of methods is used to generate intensity variations.
- Ordered dither methods generate intensity variations with a one-to-one mapping of points in a scene to the display pixel.
- To obtain n2n2 intensity levels, it is necessary to set up an n*n dither matrix DnDn whose elements are distinct positive integers in the range of 0 to n2−1n2−1.

COMPUTER GRAPHICS (DEC 2019)

Q.P.Code:76033

Q.1) Attempt any five from the following:- (20 M)

a) Define the following terms: Resolution, Aspect ratio, Phosphorescence and fluorescence. (5 M)

Ans:

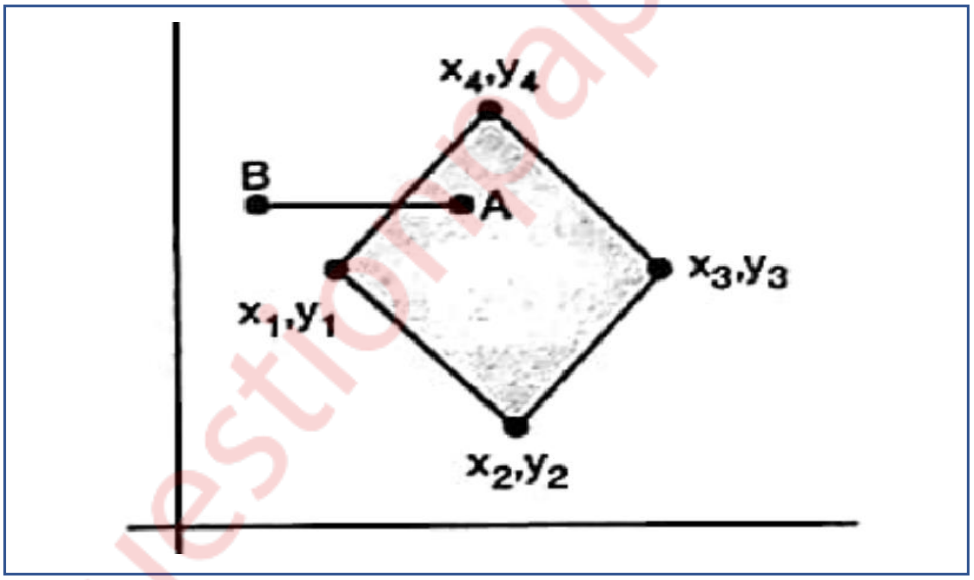
- **Resolution:** In computers, resolution is the number of pixels (individual points of colour) contained on a display monitor, expressed in terms of the number of pixels on the horizontal axis and the number on the vertical axis. The sharpness of the image on a display depends on the resolution and the size of the monitor. The same pixel resolution will be sharper on a smaller monitor and gradually lose sharpness on larger monitors because the same number of pixels are being spread out over a larger number of inches.
- **Aspect Ratio:** An aspect ratio is an attribute that describes the relationship between the width and height of an image. Aspect ratio is expressed by the symbolic notation: X:Y. The values of X and Y are not the actual width and height of the image, but describe the relationship between them. Aspect ratio is generally used to determine the relative horizontal and vertical sizes of computer graphics. For example, if a computer graphic has an aspect ratio of 3:1, this means the width of the graphic is three times of the height of the image. Aspect ratio plays an important role in resizing. During resizing, the aspect ratio must remain same in order to keep the image undistorted. A distorted aspect ratio leads to stretching of the image.
- **Phosphorescence:** Phosphorescence is luminescence that occurs when energy is supplied by electromagnetic radiation, usually ultraviolet light. The energy source kicks an electron of an atom from a lower energy state into an "excited" higher energy state; then the electron releases the energy in the form of visible light (luminescence) when it falls back to a lower energy state.
- **Fluorescence:** Fluorescence occurs when electrons move from their ground state to an excited state. These electrons keep the same spin as in the ground state, but when they return to the ground state they emit energy. This energy has a longer wavelength than the originally absorbed energy. If this longer wavelength is within the visible spectrum, then we can see a glowing light.

b) What is the purpose of Inside-Outside test, explain anyone method.

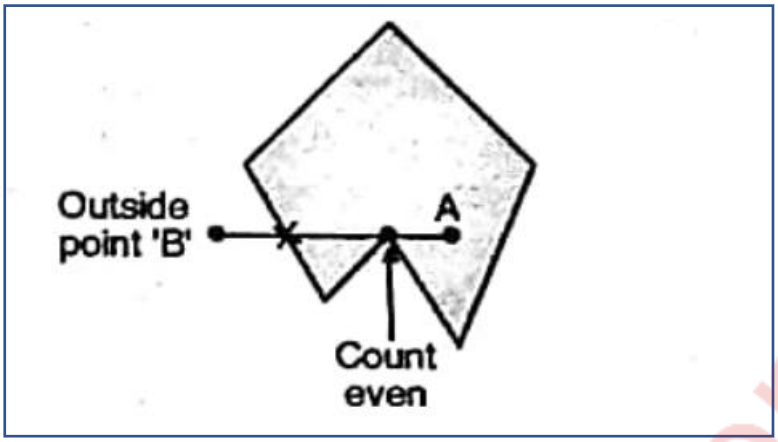
(5 M)

Ans:

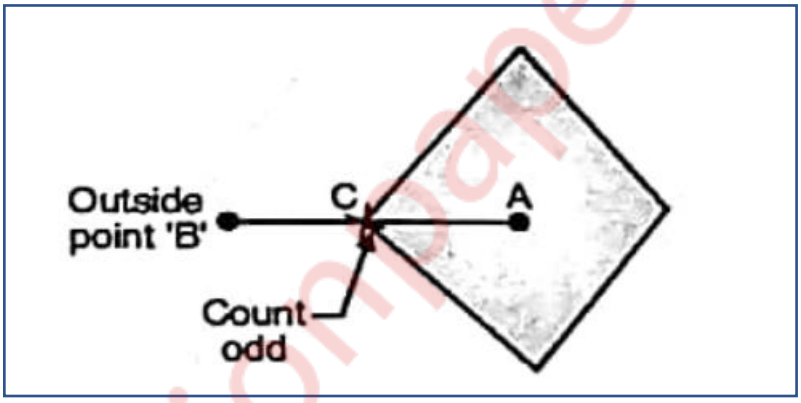
- One method of doing this is to construct a line segment between a point in question i.e. point to test whether inside or outside, and a point which is surely outside the polygon. But ow we are going to find out a point which is surely outside the polygon? It is very easy to find out that point.
- For example, pick a point with an x co-ordinate smaller than the smallest co-ordinate of the polygons vertices and the y co-ordinate will be any y value, or for simplicity we will take y value same as the y value of point in question.
- In this case point A is one, which we want to check i.e. whether point A is inside polygon or not.
- As we are using arrays to store vertices of polygon we can easily come to know the vertex which is having lowest value of x and that is x_1. So we have to select a point smaller than x_1 and generally we select same value of y as that of point A, in order to draw straight line. But even if you select any y value for outside point, it will not make any difference.



- Then count how many intersections are occurring with polygon boundary by this line till the point in question i.e. 'A', is reached. If there are an odd number of intersections then the point is outside the polygon.
- This is called even-odd method to determine interior points of polygon.
- But this even odd test fails for one case i.e. when the intersection point is vertex. To handle this case we have to make few modifications we must look at other end points of two segments of a polygon which meet at this vertex.
- If these points lie on the same side of the constructed line AB, then the intersection point counts as an even number of intersection.



- But if they lie on opposite sides of the constructed line AB, then the intersection point is counted as a single intersection.



- In this case the constructed line AB intersects polygon in a vertex ‘C’. here the other points of both the line segments which meets at vertex ‘C’ are lieing in opposite sides of this constructed line Ab. So the intersection at vertex ‘C’ is counted as odd count i.e. 1. The total number of C intersection made by this line with polygon is 1 i.e. odd and as it is odd therefore the point ‘A’ is inside.

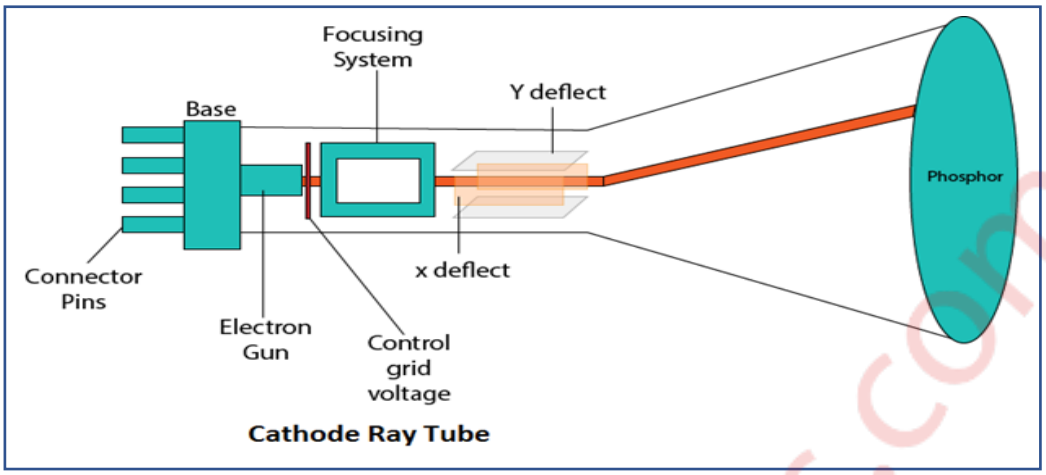
---

c) Draw the diagram of CRT and explain is working. (5 M)

Ans:

- CRT stands for Cathode Ray Tube. CRT is a technology used in traditional computer monitors and televisions. The image on CRT display is created by firing electrons from the back of the tube of phosphorus located towards the front of the screen.

- Once the electron heats the phosphorus, they light up, and they are projected on a screen. The color you view on the screen is produced by a blend of red, blue and green light.



- Main Components of CRT are:
- **Electron Gun:** Electron gun consisting of a series of elements, primarily a heating filament (heater) and a cathode. The electron gun creates a source of electrons which are focused into a narrow beam directed at the face of the CRT.
- **Control Electrode:** It is used to turn the electron beam on and off.
- **Focusing system:** It is used to create a clear picture by focusing the electrons into a narrow beam.
- **Deflection Yoke:** It is used to control the direction of the electron beam. It creates an electric or magnetic field which will bend the electron beam as it passes through the area. In a conventional CRT, the yoke is linked to a sweep or scan generator. The deflection yoke which is connected to the sweep generator creates a fluctuating electric or magnetic potential.
- **Phosphorus-coated screen:** The inside front surface of every CRT is coated with phosphors. Phosphors glow when a high-energy electron beam hits them. Phosphorescence is the term used to characterize the light given off by a phosphor after it has been exposed to an electron beam.

---

d) What do you understand by Control Points, Degree of Continuity, Local and Global control w.r.t Curve Generation. (5 M)

Ans:

- **Control Points:** In computer-aided geometric design a control point is a member of a set of points used to determine the shape of a spline curve or, more generally, a surface or higher-dimensional object.
- **Degree of Continuity:**

Q.2)

a) Explain DDA Line drawing algorithm and Plot the points for line AB (A(10, 15) B(5, 25)) using it. (10 M)

Ans:

- Digital differential analyser is based on incremental method. The slope intercept equation for a straight line is, y = mx + b, where m is slope and b is y-intercept. We can determine the value of m as,

$$M = (y2 - y1)/(x2 - x1) = Dy / Dx$$

- For lines with slope(m) is <1 i.e. lines with positive gentle slopes, we are moving in x-direction by uniform steps of calculating the corresponding y-value by using

$$Dy/Dx = m$$

$$Dy = m \cdot Dx$$

$$(y_a + 1) - (y_a) = m.[(x_a + 1) - (x_a)]$$

But as we are moving in x-direction by 1 unit i.e. distance between two columns;

$$(x_a + 1) - (x_a) = 1$$

$$(y_a + 1) - (y_a) = m(1)$$

$$y_a + 1 = m + y_a$$

$$\text{i.e. } y_{new} = \text{slope} + y_{old}$$

- Similarly when the slope is > 1 i.e. lines with positive steep slopes, we are moving in y-direction by uniform steps and calculating the corresponding x-value.

$$Dx/Dy = m$$

$$\text{i.e. } Dx = Dy/m$$

$$(x_a + 1) - (x_a) = (y_a + 1) - (y_a) / m$$

$$(x_a + 1) - x_a = 1/m$$

$$X_a + 1 = 1/m + x_a$$

$$\text{i.e. } x_{new} = 1/\text{slope} + x_{old}$$

- When the slope(m) is = 1 then as x changes, values of y also changes.

$$\text{i.e. } Dy/Dx = m$$

$$\text{but } m = 1$$

$$Dy = Dx$$

$$Y_{new} = y_{old} + 1 \text{ and } x_{new} = x_{old} + 1$$

- Given line A(10, 15) and B(5, 25)

X_1 =10, Y1 = 15 and X2 = 5, Y2 = 25

Dx = X2-X1 = (5-10) = -5

Dy = Y2-Y1 = (25-15) = 10

|Dx|<|Dy|

$X_{increment}$ = Dx/steps = 5/5 = 1

$Y_{increment}$ = Dy/steps = 10/25 = 0.4

$x_{new} = x_{old} + x_{increment} = 10+1=11$

$y_{new} = y_{old} + y_{increment} = 15+0.4 = 15.4$

now loop steps are tabulated as follows:

I	X	Y	Plot
1	10	15	10,15
2	11	15.4	11,15
3	12	15.8	12,16
4	13	16.2	13,16
5	14	16.6	14,17
6	15	17.0	15,17
7	16	17.4	16,17
8	17	17.8	17,18
9	18	18.2	18,18
10	19	18.6	19,19

---

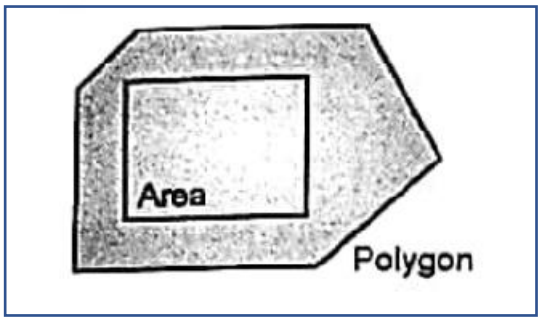
b) Explain area subdivision algorithm for hidden surface removal. (10 M)

Ans:

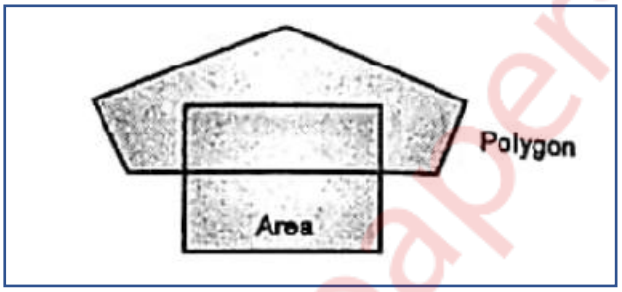
- The area-subdivision method takes advantage by locating those view areas that represent part of a single surface.
- Divide the total viewing area into smaller and smaller rectangles until each small area is the projection of part of a single visible surface or no surface at all.
- Continue this process until the subdivisions are easily analysed as belonging to a single surface or until they are reduced to the size of a single pixel.
- The algorithm is a recursive procedure based on a 2-step strategy.

1. Decide which polygons overlap the given area on the screen.
2. Which polygons are visible in that area.

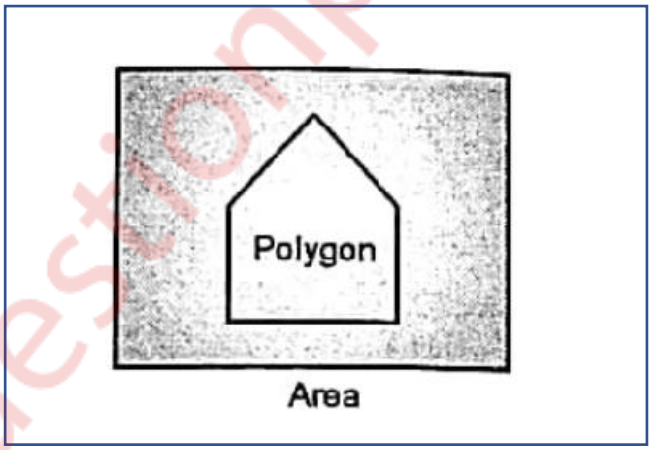
- Categories of polygons are:
- **Surrounding polygon:** In this case a polygon completely covers the given screen area



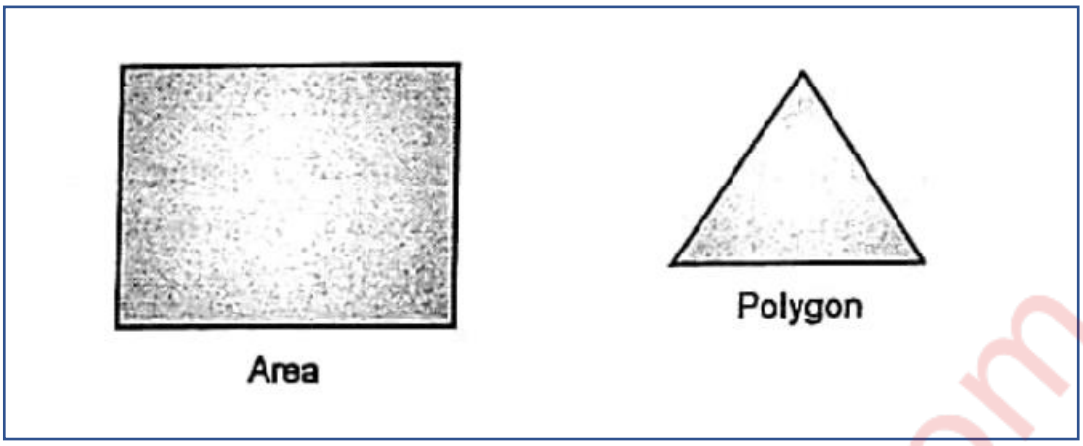
- **Intersection Polygon:** In this case a polygon is getting intersected with the screen area i.e. the polygon is partially inside the screen area.



- **Contained Polygon:** In this case a polygon is lying completely inside the given screen area.



- **Disjoint polygon:** Whenever the polygon is lying completely outside the given screen area, it is called as disjoint polygon.



Q.3)

a) What is aliasing, how it affects the appearance of an object. Explain any two anti-aliasing methods. (10 M)

Ans:

- In computer graphics, the process by which smooth curves and other lines become jagged because the resolution of the graphics device or file is not high enough to represent a smooth curve.
- In the line drawing algorithms, we have seen that all rasterized locations do not match with the true line and we have to select the optimum raster locations to represent a straight line. This problem is severe in low resolution screens. In such screens line appears like a stair-step, as shown in the figure below. This effect is known as aliasing. It is dominant for lines having gentle and sharp slopes.

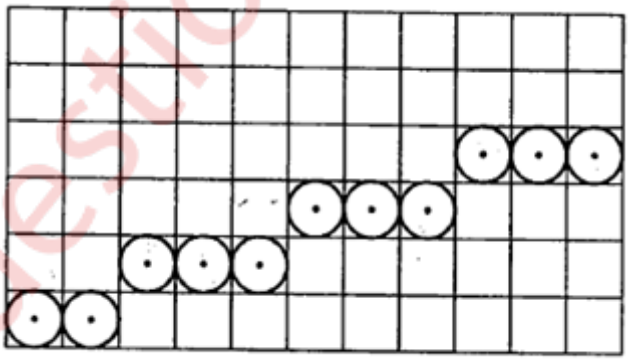


Fig. Aliasing effect

- The **aliasing effect** can be reduced by adjusting intensities of the pixels along the line. The process of adjusting intensities of the pixels along the line to minimize the effect of aliasing is called antialiasing.
- **Antialiasing** is a term for techniques that are designed to mitigate the effects of aliasing.

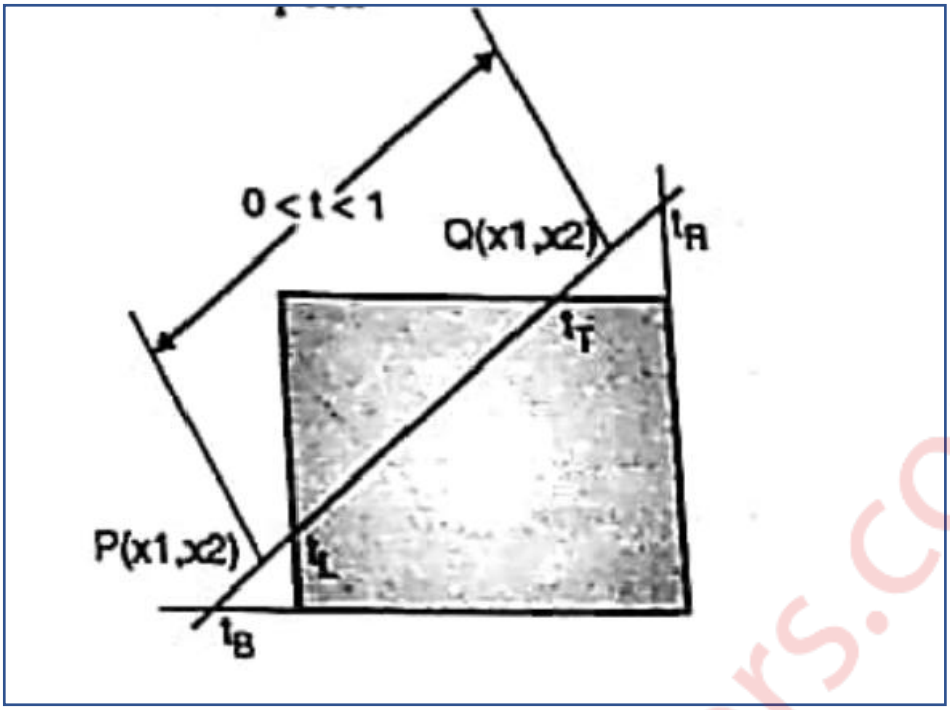
- The idea is that when a pixel is only partially covered by a shape, the colour of the pixel should be a mixture of the colour of the shape and the colour of the background.
- When drawing a black line on a white background, the colour of a partially covered pixel would be grey, with the shade of grey depending on the fraction of the pixel that is covered by the line.
- The technique of anti-aliasing is:
- **Prefiltering:** This is technique that determines pixel intensity based on the amount of that particular pixel coverage by the object in the scene i.e. It computes pixel colours depending on the objects coverage.
- It means how much part or fraction of that pixel is covered by the object and depending on that, it sets the value of the pixel. It requires large number of calculations and approximations. Prefiltering generates more accurate antialiasing effect. But due to its high complexity of calculations it is not used.
- **Postfiltering**: Super sampling or postfiltering is the process by which aliasing effects in graphics are reduced by increasing the frequency of the sampling grid and then averaging the results down. This process means calculating a virtual image at a higher spatial resolution than the frame store resolution and then averaging down to the final resolution. It is called postfiltering as the filtering is carried out after sampling.

---

b) Explain Liang Barsky line clipping algorithm, what is benefit over Cohen Sutherland algorithm? Clip the line with co-ordinates (5, 10) and (35, 30) against window (x_{min}, y_{min}) = (10, 10) and (x_{max}, y_{max}) = (20, 20).
(10 M)

Ans:

- The Liang-Barsky algorithm uses the parametric equation of a line and inequalities describing the range of the clipping box to determine the intersections between the line and clipping box. With these intersections it knows which portion of the line should be drawn. This algorithm is more efficient than Cohen-Sutherland. The ideas for clipping line of Liang-Barsky and cyrus-Beck are the same. The only difference is Liang-Barsky algorithm has been optimized for an upright rectangular clip window. So we will study only the idea of Liang-Barsky.
- Liang and Barsky have created an algorithm that uses floating-point arithmetic but finds the appropriate ends points with at most four computations. This algorithm uses the parametric equations for a line and solves for inequalities to find the range of the parameter for which the line is in the viewport.



- Let P(X1, Y1), Q(X2, Y2) be the line which we want to study. The parametric equation of line segment from gives x-values and y-values for every point in terms of a parameter that ranges from 0 to 1. The equations are

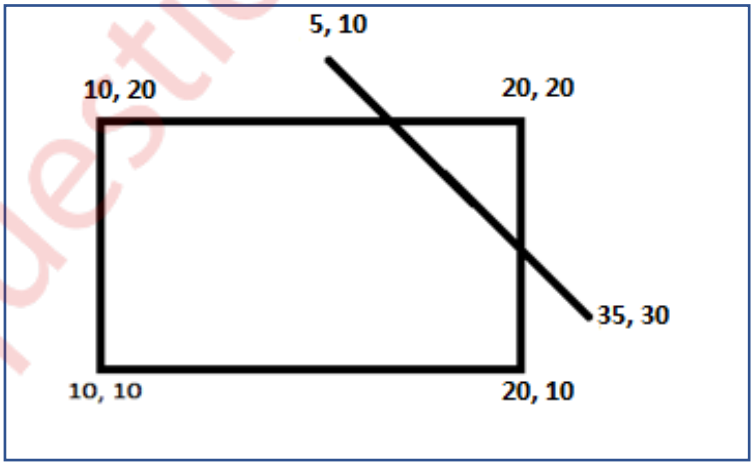
  X = X1 + (X2 – X1)*t = Y1 + dY*t

  And

  Y = Y1 + (Y2 – Y1)*t = X1 + dX*t

  We can see that when t=0, the point computed is P(x1, y1) and when t =1, the point computed is Q(x2, y2).

- Lets first draw the window and line as shown in fig.



Given things are XL = 10, YB = 10, XR = 20, YT = 20

Lets call a line AB with its coordinates as X1 = 5, Y1 = 10, X2 = 35, Y2 = 30

Now we have to find Dx and Dy as

Dx = X2 – X1 = 35 - 5 = 30

Dy = Y2 – Y1 = 30 – 10 = 20

Lets calculate the values of parameters P and Q as

P1 = -Dx = -30

P2 = Dx = 30

P3 = -Dy = -20

P4 = Dy = 20

Now,

Q1 = X1 – XL = 5 -10 = -5

Q2 = XR – X2 = 20 - 15 = 5

Q3 = Y1 – YB = 10 -10 = 0

Q4 = YT – Y2 = 20 - 10 = 10

Now lets find P,

P1 = Q1/P1 = -5/(-30) = (1/6)

P2 = Q2/P2 = 15/(30) = (1/2)

P3 = Q3/P3 = 0/(-20) = (0)

P4 = Q4/P4 = 10/(20) = (1/2)

Now,

t1 = Max(1/6, 0, 0) = 1/6

t2 = Min(1/2, 1/2, 0) = 1/2

Now,

X1’ = X1 + Dx*t1

= 5+30*(1/6)

= 10

Y1’ = Y1 + Dy*t1

= 10+(20)*(1/6)

= 13.33

And with t2 it will be

X2’ = X1 + Dx*t2

= 5+30*(1/2)

= 20

Y2’ = Y1 + Dy*t2

= 10+(20)*(1/2)

= 20

From this we will come to know that a point (20, 20) is an intersection point with respect to the edge of the window boundary. So we need to discard the line from point (5, 10) to (5, 83.33) and consider line (20, 20) to (35, 30).

Q.4)

a) What is shading? Explain Gourard and Phong Shading with their pros and cons. (10 M)

Ans:

Gouraud Shading:

1. Gouraud surface shading was developed in the 1970s by Henri Gouraud.
2. It is the interpolation technique.
3. Intensity levels are calculated at each vertex and interpolated across the surface.
4. Intensity values for each polygon are matched with the values of adjacent polygons along the common edges.
5. This eliminates the intensity discontinuities that can occur in flat shading.
6. To render a polygon, Gouraud surface rendering proceeds as follows:
    - Determine the average unit normal vector at each vertex of the polygon.
    - Apply an illumination model at each polygon vertex to obtain the light intensity at that position.
    - Linearly interpolate the vertex intensities over the projected area of the polygon

Illumination values are linearly interpolated across each scan-line as shown in figure 41.

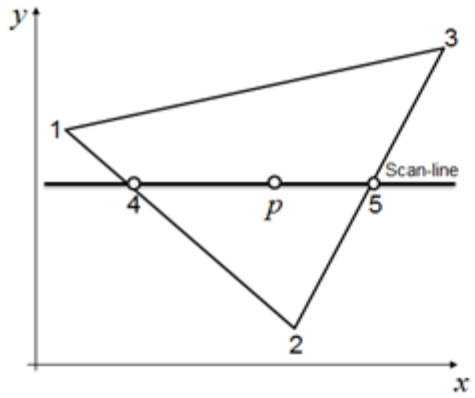


Figure 41

$$I_4 = \frac{y_4 - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y_4}{y_1 - y_2} I_2$$

$$I_5 = \frac{y_5 - y_2}{y_3 - y_2} I_3 + \frac{y_3 - y_5}{y_3 - y_2} I_2$$

$$I_p = \frac{x_5 - x_p}{x_5 - x_4} I_4 + \frac{x_p - x_4}{x_5 - x_4} I_5$$

Advantages: Removal of discontinuities associated with the constant shading model.

Disadvantages: Highlighted surfaces are sometimes displayed with anomalous shape and the linear intensity interpolation can use bright or dark intensity interpolation can use bright or dark intensity strips. This effect can be reduced by sung phong shading method.

Phong Shading:

1. A more accurate interpolation based approach for rendering a polygon was developed by Phong Bui Tuong.
2. Basically the Phong surface rendering model is also called as normal-vector interpolation rendering.
3. It interpolates normal vectors instead of intensity values.
4. To render a polygon, Phong surface rendering proceeds as follows:
5. Determine the average unit normal vector at each vertex of the polygon.
6. Linearly interpolate the vertex normal over the projected area of the polygon.
7. Apply an illumination model at positions along scan lines to calculate pixel intensities using the interpolated normal vectors as shown in figure 42

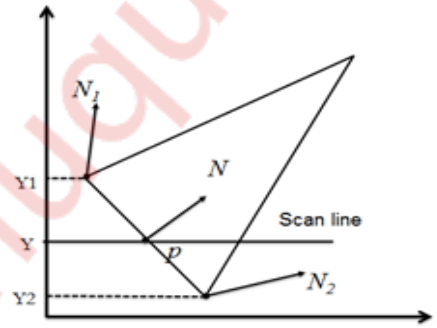


Figure 42

$$N = \frac{y - y_2}{y_1 - y_2} N_1 + \frac{y_1 - y}{y_1 - y_2} N_2$$

Advantages: It is very effective in dealing with specular highlights.

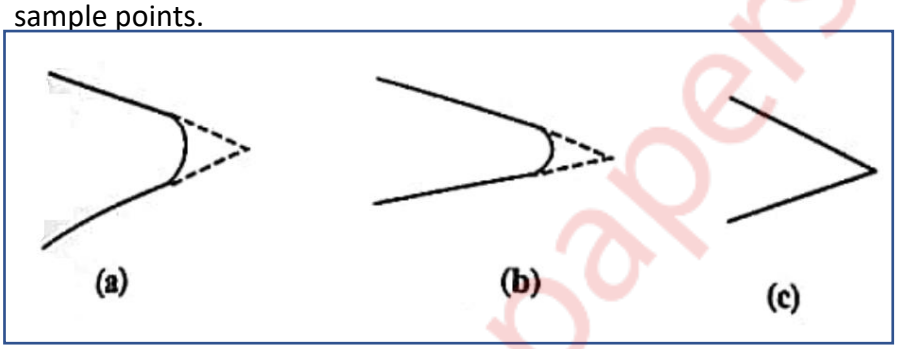
Disadvantages: This technique is relatively time-consuming since the illumination model is evaluated at every point using interpolated normal vectors.

---

b) Explain what is meant by B spline curve? State the various properties of spline curve. (10 M)

Ans:

- A set of blending functions which takes this approach is called B-splines. Basically spline means a strip, which we have to move around the sample points. Generally the B-spline blending functions were designed to eliminate sharp corners in the curve and the curve does not usually pass through the sample points. But if we need sharp corners we can produce it by using many identical sample points.



- Fig(a) shows the curve using one sample point. Fig(b) shows the curve using two identical sample points. It means we are calling the blending function for same sample point twice. And fig(c) shows if three identical sample points are used the curve will be forced to pass through the sample points.
- **Properties of B-spline Curve:**
  - The sum of the B-spline basic functions for any parameter value u is 1 i.e. $\sum_{i=1}^{n+1} N_{i,k}(u) = 1$
  - Except for k=1 each basis function has precisely one maximum value.
  - The maximum order of the curve is equal to the number of vertices of defining polygon.
  - The degree of B-spline polynomial is independent of the number of vertices of defining polygon.
  - B-spline allows local control over the curve surface because each vertex affects the shape of a curve only over a range of parameter values where its associated basic function is nonzero.
  - The curve exhibits the variation diminishing property. The curve does not oscillate about any straight line move often then its defining polygon.
  - The curve is generally follows the shape of defining polygon.

- Any affine transformation can be applied to the curve by applying it to the vertices of defining polygon.
- The curve line within the convex hull of its defining polygon.

Q.5)

a) Explain Scan line polygon fill algorithm with the help of suitable diagrams. (10 M)

Ans:

- Boundary fill algorithm and flood fill algorithm is defined at pixel level.
- Scan line fill algorithm is defined at geometric level i.e. coordinates, edges, vertices etc.
- The algorithm starts with first scan line and proceeds line by line to the last scan line.
- It checks whether every pixel on that scan line satisfies inside point test or not i.e. it checks which points on that scan line are inside the polygon.
- This method avoids the need for seed point.
- Consider this example of convex polygon to explain this algorithm.

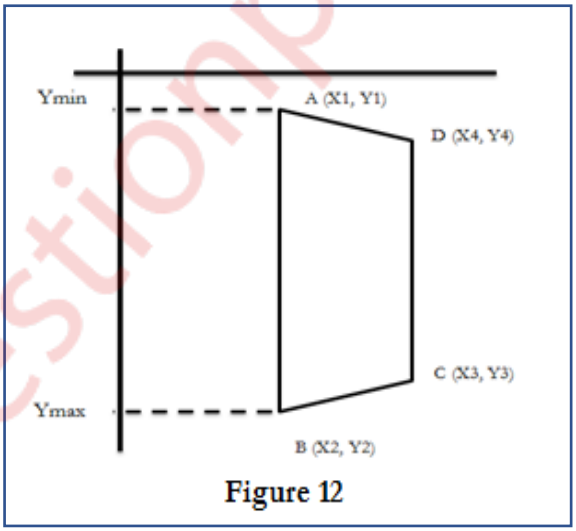


Figure 12

- As shown in figure 12, algorithm begins with first scan line i.e. Ymax and proceed line by line towards the last scan line i.e. Ymin.
- Here we are considering only the first and line scan and not individual edges.
- For each edge of polygon we are storing 5 attributes along with the slope.
- For AB we are storing:
    - Xmax: X2

- Xmin: X1
- Ymax: Y2
- Ymin: Y1

- Likewise we find for AD, CD and BC.

Edge	Ymax	Xmax	Ymin	Xmin	Slope
AB	Y2	X2	Y1	X1	M1
AD	Y4	X4	Y1	X1	M2
CD	Y3	X3	Y4	X4	M3
BC	Y2	X2	Y3	X3	M4

- In this we have to select only those edges which are getting intersected by scan line.
- For this we are using Ymax of particular edge to find out whether it is intersecting by scan line or not
- Now sort the Ymax array.

EDGE	Ymax	Xmax	Ymin	Xmin	SLOPE
AB	Y2	X2	Y1	X1	M1
BC	Y2	X2	Y3	X3	M4
CD	Y3	X3	Y4	X4	M3
AD	Y4	X4	Y1	X1	M2

- So from the table we can find out that edge AB and BC are intersection of scan line.
- Every time we are decreasing the scan line by 1, from Ymax to Ymin of polygon. Refer figure13

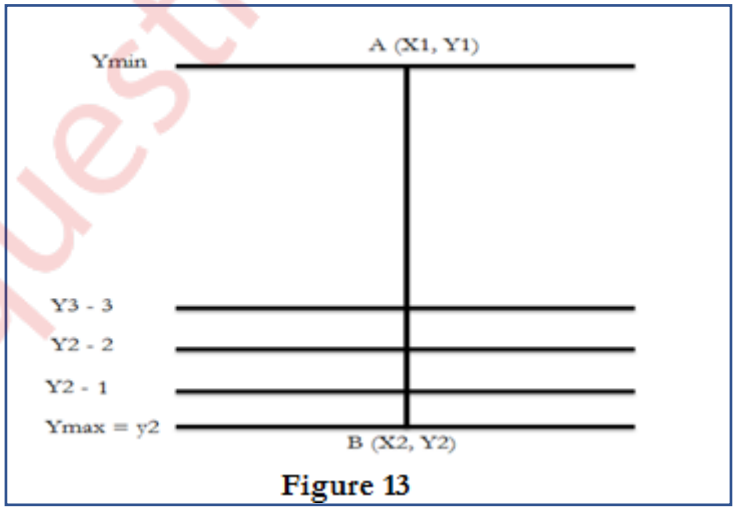


Figure 13

- Now we have to find the corresponding X value for the intersection point.

  Slope (m) = DY/DX = Change in Y / Change in X.

= Ynew – Yold / Xnew – Xold

= 1 / Xnew – Xold

Xnew – Xold = 1 / m

Therefore, Xnew = Xold + 1/m.

- Like this we are finding intersecting of scan line with every edge of polygon.
- Once we have found the intersecting points with both edges i.e. selected edges like AB and BC, then join these two intersecting points by solid line and continue.
- Again decrease Y by 1 and find out new intersection points for those edges till one of the edges gets over i.e. the scan line goes below Ymin of one edge.

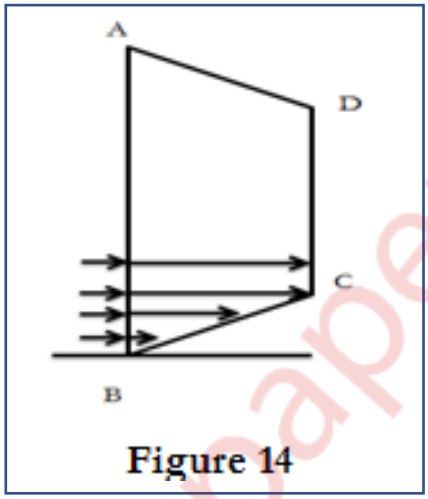


Figure 14

- In this case edge BC gets over earlier, so discard edge BC and select next edge from sorted table, which is CD and continue the process.
- The process continues till scan line reaches to Ymin of the polygon.

---

b) Explain the step for 2D reflection w.r.t. line y=mx and also derive a composite transformation matrix. (10 M)

Ans:

- To rotate an object about an arbitrary point, (Xp ,Yp) we have to carry out three steps:
  - Translate point (Xp, Yp) to the origin.
  - Rotate it about the origin and,
  - Finally, translate the centre of rotation back where it belongs (See figure 1.).
- we have already seen that matrix multiplication is not commutative, i.e. multiplying matrix A by matrix B will not always yield the same result as multiplying matrix B by matrix A. Therefore, in obtaining composite transformation matrix, we must be careful to order the matrices so that they correspond to the order of the

transformations on the object. Let us find the transformation matrices to carry out individual steps.

- The translation matrix to move point (x_p, y_p) to the origin is given as, x_p

$$T_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xp & -yp & 1 \end{bmatrix}$$

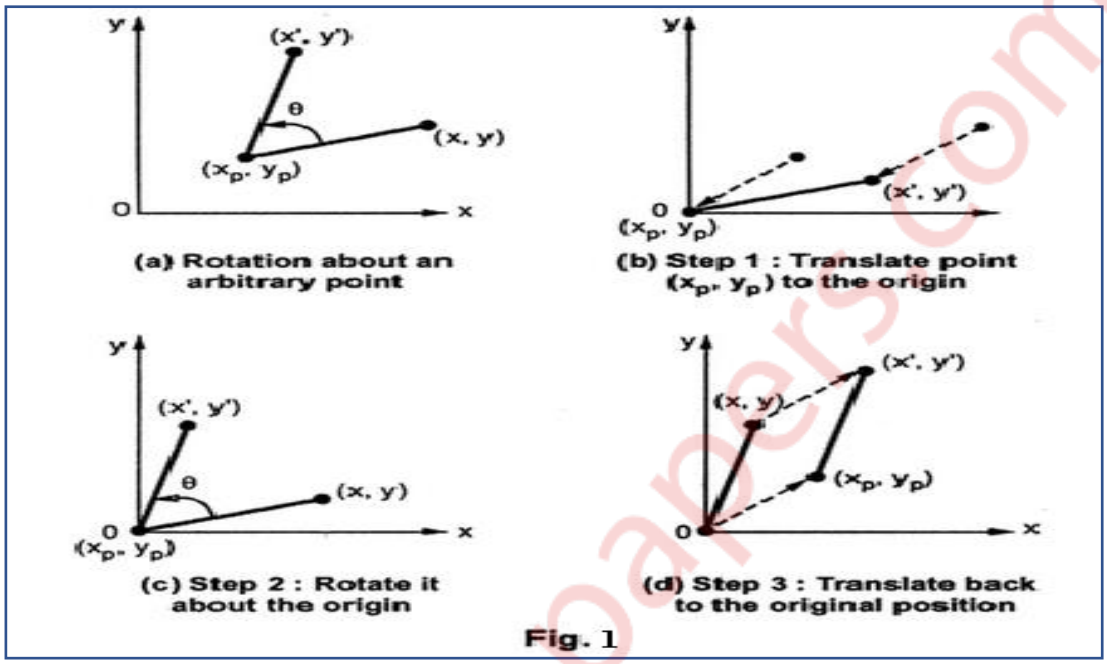


Fig. 1

The translation matrix to move point (x_p, y_p) to the origin is given as, x_p

$$T_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xp & -yp & 1 \end{bmatrix}$$

The rotation matrix for counterclockwise rotation of point about the origin is given as,

$$R = \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The translation matrix to move the center point back to its original position is given as,

$$T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

Therefore the overall transformation matrix for a counterclockwise roatation by an angle Θ about the point (xp, yp) is given as,

$$T_1*R*T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xp & -yp & 1 \end{bmatrix} * \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

$$= \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ -xpcos\Theta + ypsin\Theta & -xpsin\Theta - ypcos\Theta & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

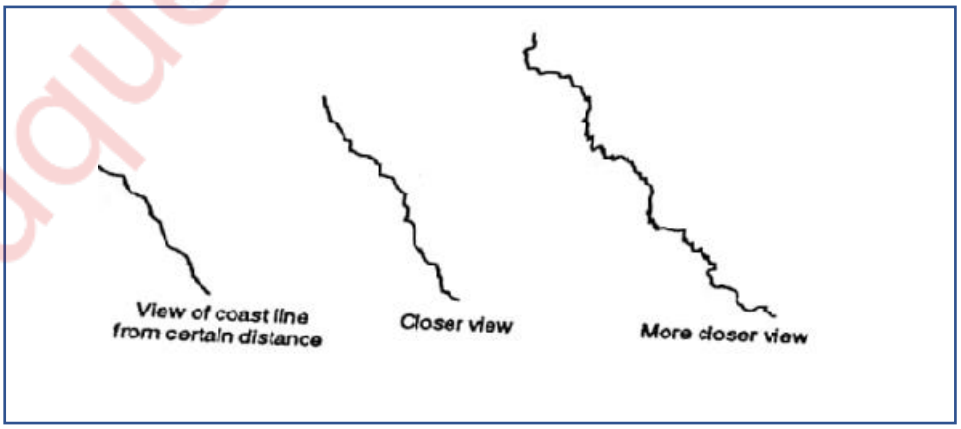
$$= \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ -xpcos\Theta + ypsin\Theta + xp & -xpsin\Theta - ypcos\Theta + yp & 1 \end{bmatrix}$$

Q.6) Write short notes on:

a) Fractals:

Ans:

- A fractal is defined as a rough or fragmented geometric shape that can be split into parts, each of which is approximately a reduced-size reproduction of the complete shape based on the property known as self- similarity. It was derived from the Latin word fractus which means broken or fractured. Natural objects can be realistically described using fractal geometry methods. Example.- cloud, mountains, trees, stone etc.
- Fractal methods use procedures rather than equations to model objects. so it uses procedural modelling. The major characteristic of any procedural model is that the model is not based on data ,but rather on the implementation of the procedure following a particular set of rules.
- A fractal combines the following characteristic:
- Its parts have the same form or structure as a whole, except that they are at a different scale and may be slightly deformed.
- Its form is extremely irregular or fragmented, and remains so, whatever the scale of examination.
- It is formed by iteration i.e. the procedure is used repeatedly(recursively)
- Example: if P0 = (X0, Y0, Z0) is a selected initial position, the successive levels P1 = F (P0), P2 = F (P1), …… Pn = F (Pn-1) are generated by a transformation function F.
- Fractional dimensions.



- Imagine that one object is made up of clay. If we break that objects in terms of line or line segments, then we will give the dimensions D_1= 1. If the object is broken into a plane, then we will give the dimension D_t = 2. And if the object is broken into 3D objects like cube, sphere etc. then we will give the transformation as D_t = 6. Here the variables D_t is called topological dimensions.

---

b) Sweep representation and CSG method.

Ans:

- Sweep representations are used to construct three dimensional objects from two dimensional shape .There are two ways to achieve sweep: Translational sweep and Rotational sweep. In translational sweeps, the 2D shape is swept along a linear path normal to the plane of the area to construct three dimensional object. To obtain the wireframe representation we have to replicate the 2D shape and draw a set of connecting lines in the direction of shape, as shown in the figure (8)

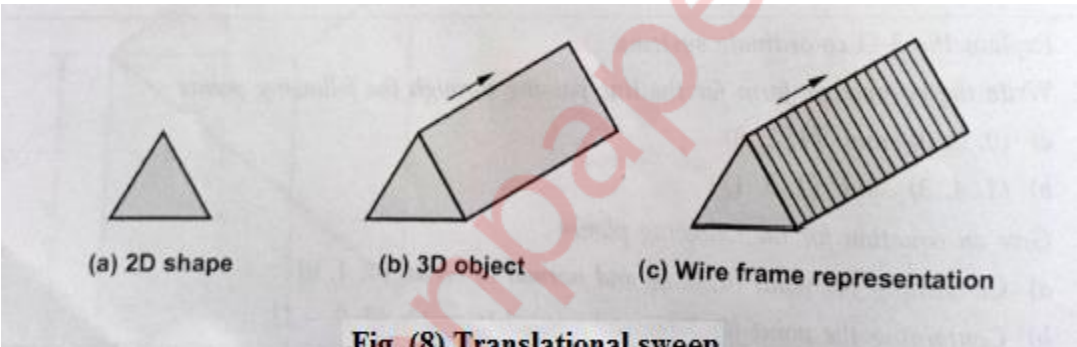


Fig. (8) Translational sweep

- In general we can specify sweep constructions using any path. For translation we can vary the shape or size of the original 2D shape along the sweep path. For rotational sweeps, we can move along a circular path through any angular distance from 0° to 360°.
- These sweeps whose generating area or volume changes in size, shape or orientation as they are swept and that follow an arbitrary curved trajectory are called general sweeps .General sweeps are difficult to model efficiently for example, the trajectory and object shape may make the swept object intersect itself, making volume calculations complicated.

- **CSG Method:** Another technique for solid modelling is to combine the volumes occupied by overlapping three-dimensional objects using Boolean set operations.
- This modelling technique is called Constructive Solid Geometry (CSG). It creates a new volume by applying Boolean operators such as union. intersection, or difference to two specified objects.

- The Fig. (10), Fig. (11), Fig. (12) show the example for forming new shapes using Boolean set operations The Fig. 10 (a) shows that two rectangular blocks are placed adjacent to each other. We can obtain the combined object with the union operation as shown in Fig. 10 (b).

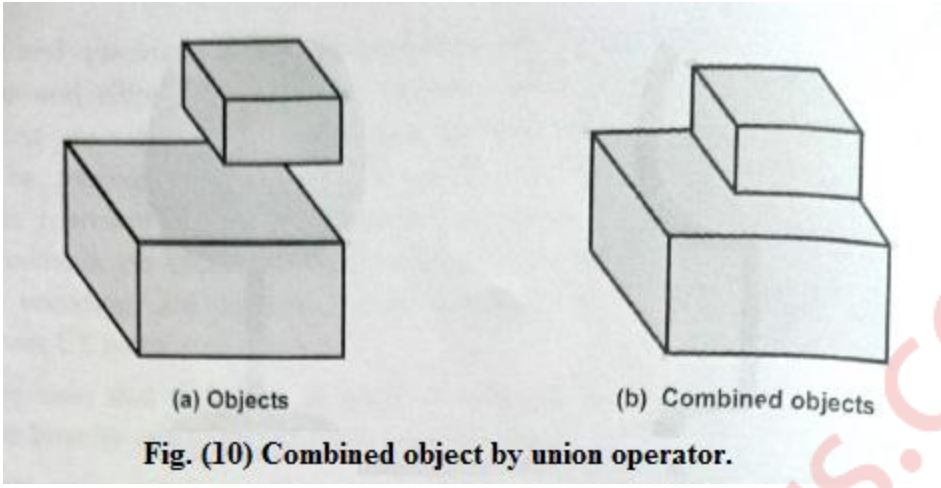


Fig. (10) Combined object by union operator.

- The Fig.(11) shows the result of intersection operation obtained by overlapping cylinder and cube.
- With the difference operation, we can obtain the resulting solid as shown in Fig. (12).
- The CSG method Uses three dimensional objects such as blocks, pyramids, cylinders, cones, spheres, and closed spline surfaces to generate other solid objects In this method, an object is stored as a tree with operators at the internal nodes and simple primitives at the leaves.

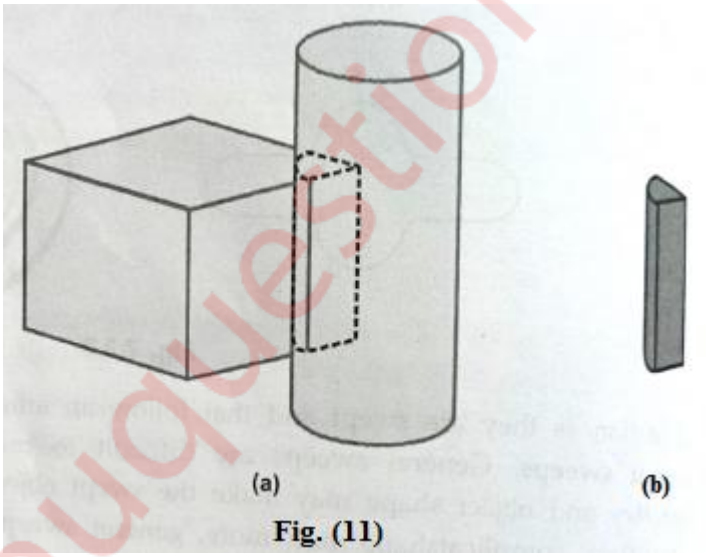


Fig. (11)

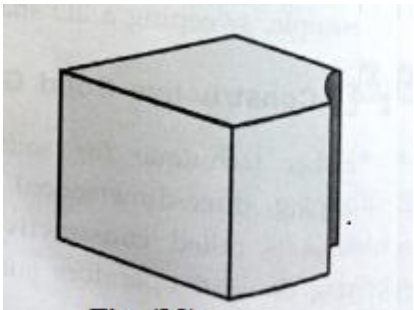


Fig. (12)

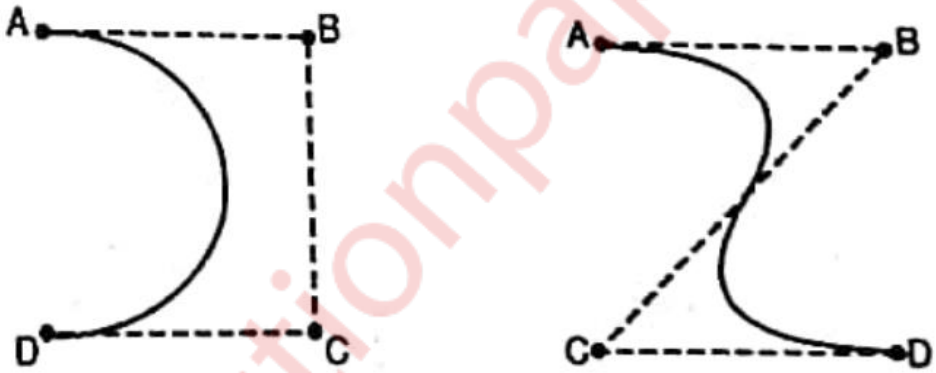
- Some nodes represent Boolean operators, whereas others represent operations such as translation, rotation, and scaling. It is important to note that Boolean operations are not, in general, communicative Therefore the edges of the trees must be in proper order.

---

c) Bezier Curve and the properties. (10 M)

Ans:

- It is a different way of specifying a curve, rather same shapes can be represented by B-spline and Bezier curves. The cubic Bezier curve require four sample points, these points completely specify the curve.



- The curve begins at the first sample point and ends at fourth point. If we need another Bezier curve then we need another four sample points. But if we need two Bezier curves connected to each other, then with six sample points we can achieve it. For this, the third and fourth point of first curve should be made same as first and second point of curve.
- The equation for the Bezier curve are as follows:

$$X = X_4a^3 + 3X_3a^2(1-a) + 3X_2a(1-a)^2 + X_1(1-a)^3$$

$$Y = Y_4a^3 + 3Y_3a^2(1-a) + 3Y_2a(1-a)^2 + Y_1(1-a)^3$$

$$Z = Z_4a^3 + 3Z_3a^2(1-a) + 3Z_2a(1-a)^2 + Z_1(1-a)^3$$

- Here as the value of 'a' moves from 0 to 1, the curve travels from the first to fourth sample point. But we can construct a Bezier curve without referencing to the above expression. It is constructed by simply taking midpoints.
- Properties of Bezier curve are as follows:
    1. The basic function are real in nature.
    2. Bezier curve always passes through the first and last control points i.e. curve has same end points as the guiding polygon.
    3. The degree of polynomial defining the curve segment is one less than the number of defining polygon point.
    4. The curve generally follows the shape of the defining polygon.
    5. The direction of the tangent vector at the end points is the same as that of the vector determined by first and last segments.
    6. The curve lies entirely within the convex hull formed by four control points.
    7. The curve exhibits the variation diminishing property. This means that the curve does not oscillate about any straight line more than the defining polygon.
    8. The curve is invariant under an affine transformation.

---

d) Halftone and Dithering. (10 M)

Ans:

Half toning

1. Many displays and hardcopy devices are bi-level
2. They can only produce two intensity levels.
3. In such displays or hardcopy devices we can create an apparent increase in the number of available intensity value.
4. When we view a very small area from a sufficient large viewing distance, our eyes average fine details within the small area and record only the overall intensity of the area.
5. The phenomenon of apparent increase in the number of available intensities by considering combine intensity of multiple pixels is known as half toning.
6. The half toning is commonly used in printing black and white photographs in newspapers, magazines and books.
7. The pictures produced by half toning process are called halftones.
8. In computer graphics, halftone reproductions are approximated using rectangular pixel regions, say 2*2 pixels or 3*3 pixels.
9. These regions are called halftone patterns or pixel patterns.

Dithering Techniques

- Dithering refers to techniques for approximating halftones without reducing resolution, as pixel grid patterns do.
- The term dithering is also applied to halftone approximation method using pixel grid, and something it is used to refer to color halftone approximations only.
- Random values added to pixel intensities to break up contours are often referred as dither noise.
- Number of methods is used to generate intensity variations.
- Ordered dither methods generate intensity variations with a one-to-one mapping of points in a scene to the display pixel.
- To obtain n2n2 intensity levels, it is necessary to set up an n*n dither matrix DnDn whose elements are distinct positive integers in the range of 0 to n2−1n2−1.

COMPUTER GRAPHICS (MAY 2019)

Q.P.Code: 21849

Q.1) Attempt any five from the following:- (20 M)

a) What is aliasing antialiasing (5 M)

Ans:

- In computer graphics, the process by which smooth curves and other lines become jagged because the resolution of the graphics device or file is not high enough to represent a smooth curve.
- In the line drawing algorithms, we have seen that all rasterized locations do not match with the true line and we have to select the optimum raster locations to represent a straight line. This problem is severe in low resolution screens. In such screens line appears like a stair-step, as shown in the figure below. This effect is known as aliasing. It is dominant for lines having gentle and sharp slopes.

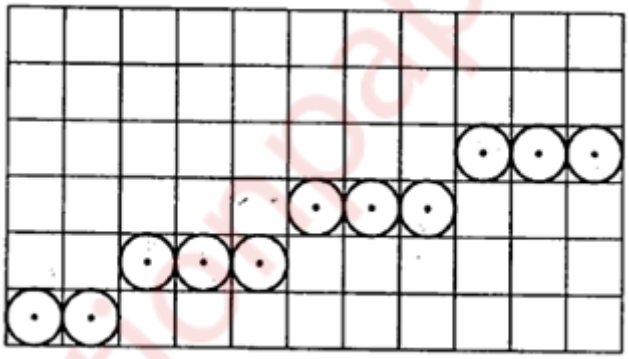


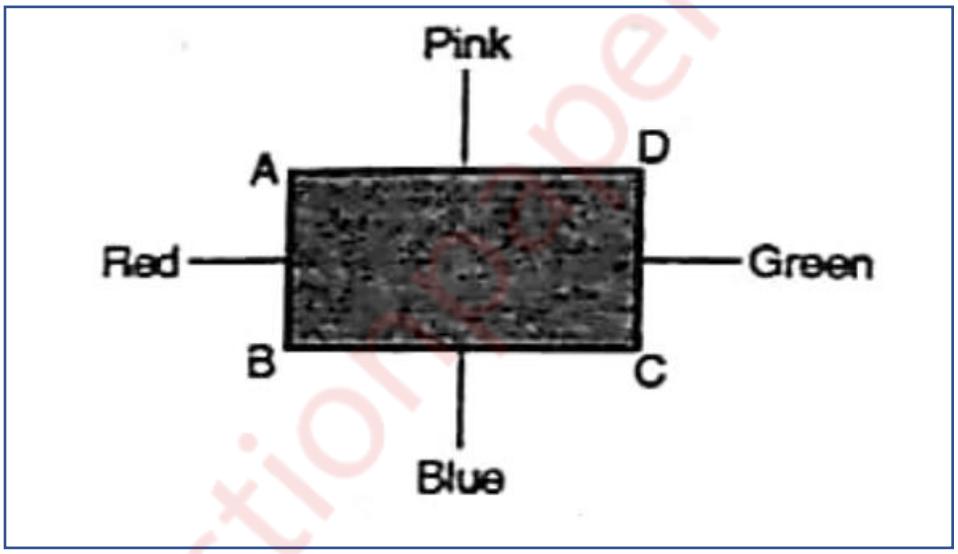
Fig. Aliasing effect

- The aliasing effect can be reduced by adjusting intensities of the pixels along the line. The process of adjusting intensities of the pixels along the line to minimize the effect of aliasing is called antialiasing.
- **Antialiasing** is a term for techniques that are designed to mitigate the effects of aliasing.
- The idea is that when a pixel is only partially covered by a shape, the colour of the pixel should be a mixture of the colour of the shape and the colour of the background.
- When drawing a black line on a white background, the colour of a partially covered pixel would be grey, with the shade of grey depending on the fraction of the pixel that is covered by the line.

b) Write the flood fill approach for 8 connected method. (5 M)

Ans:

- The limitations of boundary fill algorithm are overcome in flood fill algorithm. Like boundary fill algorithm this algorithm also begins with seed point which must be surely inside the polygon.
- Now instead of checking the boundary colour this algorithm checks whether the pixel is having the polygons original colour i.e. previous or old colour.
- If yes, then fill that pixel with new colour and uses each of the pixels neighbouring pixel as a new seed in a recursive call. If the answer is no i.e. the colour of pixel is already changed then return to its caller.
- Sometimes we want to fill an area that is not defines within a single colour boundary. Here edge AB, BC, CD, DA are having red, blue, green and pink colour, respectively.



- We can paint such areas by replacing a specified interior colour instead of searching for a boundary colour value. Here we are setting empty pixel with new colour till we get any coloured pixel.
- Flood fill and boundary fill algorithms are somewhat similar. A flood fill algorithm is particularly useful when the region or polygon has no uninformed coloured boundaries.
- The flood fill algorithm is sometimes also called as seed fill algorithm or forest fill algorithm. Because it spreads from a single point i.e. seed point in all direction.
- The following procedure illustrates a recursive method for flood fill by sing 8-connected method.

```
f-fill (X, Y, newcolour)
{
    Current = getpixel(X, Y);
    If(Current != newcolor)
    {
```

```
                Putpixel(X, Y, newcolour);
                f-fill(X-1, Y, newcolour);
                f-fill(X+1, Y, newcolour);
                f-fill(X, Y-1, newcolour);
                f-fill(X, Y+1, newcolour);
                f-fill(X+1, Y+1, newcolour);
                f-fill(X-1, Y+1, newcolour);
                f-fill(X-1, Y-1, newcolour);
                f-fill(X+1, Y-1, newcolour);
        }
}
```

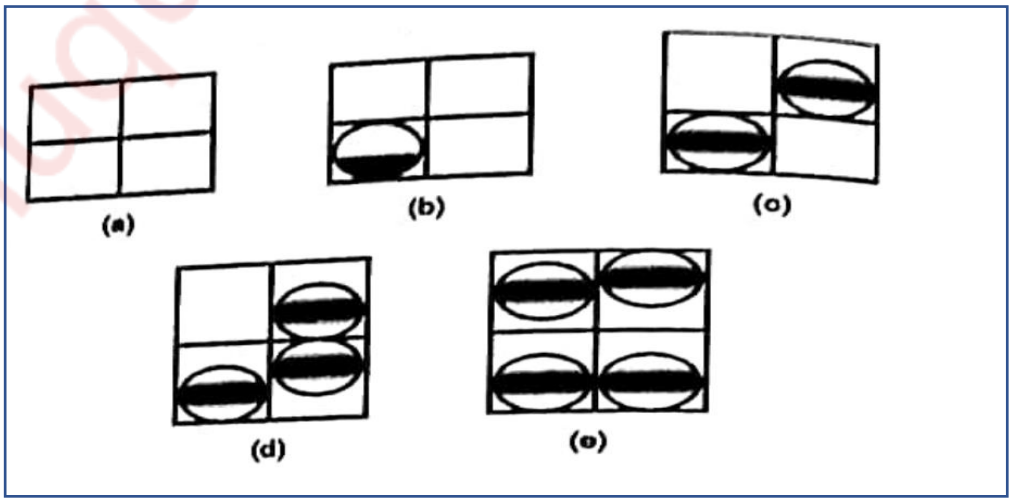
---

c) Explain the concept of halftoning with example. (5 M)

Ans:

Half toning

- Many displays and hardcopy devices are bi-level
- They can only produce two intensity levels.
- In such displays or hardcopy devices we can create an apparent increase in the number of available intensity value.
- When we view a very small area from a sufficient large viewing distance, our eyes average fine details within the small area and record only the overall intensity of the area.
- The phenomenon of apparent increase in the number of available intensities by considering combine intensity of multiple pixels is known as half toning.
- The half toning is commonly used in printing black and white photographs in newspapers, magazines and books.
- The pictures produced by half toning process are called halftones.
- In computer graphics, halftone reproductions are approximated using rectangular pixel regions, say 2*2 pixels or 3*3 pixels.



- These regions are called halftone patterns or pixel patterns.
- For bi-level systems one can represent shading intensities with a halftoning method that converts the intensity of each point on a surface into a regular pixel grid that can display a number of intensity level. The number of intensity level with this method depends on hoe many pixels we include on the grid.
- A graphics package employing a halftone technique displays a scene by replacing each position in the original scene an (n * n) grid pixels that are turned on. Such a technique of turns on two level systems into one with the five possible intensities.

---

d) Prove that two successive rotations are additive. (5 M)

Ans:

Lets assume that $\Theta_1 = \Theta_2 = 90^0$

It means that first we will perform rotation of Θ_1 i.e. 90^0 in anticlockwise direction and then on that resultant rotation matrix we will perform again anticlockwise rotation by angle Θ_2

By performing two times of rotation by Θ_1 and Θ_2 we will get results as if roation by $(\Theta_{1+} \Theta_2)$.

$$R(\Theta_1) = \begin{vmatrix} \cos 90 & \sin 90 \\ -\sin 90 & \cos 90 \end{vmatrix} = \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix}$$

$$R(\Theta_2) = \begin{vmatrix} \cos 90 & \sin 90 \\ -\sin 90 & \cos 90 \end{vmatrix} = \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix}$$

Now if $R(\Theta_1) * R(\Theta_2)$

then, $\begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix} * \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix} = \begin{vmatrix} -1 & 0 \\ 0 & -1 \end{vmatrix}$

now $R(\Theta_1 + \Theta_2)$ i.e. $R(90^0 + 90^0) = R(180^0)$

hence, $R(180^0) = \begin{vmatrix} \cos 180 & \sin 180 \\ -\sin 180 & \cos 180 \end{vmatrix} = \begin{vmatrix} -1 & 0 \\ 0 & -1 \end{vmatrix}$

Hence, $R1(\Theta_1) * R2(\Theta_2) = R(\Theta_1 + \Theta_2)$.

Q.2)

a) Plot the points for midpoint ellipse with rx = 3 and ry =5 for region 1. (10 M)

Ans:

Given: rx = 3 and ry =5 for region 1

(xc, yc) =(0, 0)

Plot first point as (x, y) where x = 0 & y= ry, hence y =5

Initial decision parameter is calculated by

dp -1 = $(rx)^2 – (ry)^2 (-ry + 1/4)$

= $5^2 -3^2 (-5 + 1/4) = 271/4$

Now, we have to check whether we have crossed region – 1 or not

$2 (ry)^2 x < 2 (rx)^2 y$

$2 (3)^2*0 < 2 (5)^2*5$

$0 < 250$

As $2 (ry)^2 x < 2 (rx)^2 y$ it means point lies in region – 1

Here dp – 1 = 271/4

Hence, increase x by 1 and decrease by 1

(x, y) = (1, 4)

Every time we have to check whether we have crossed the region or not. If yes, then find out the new initial value of region-2 and proceed along y-axis till y becomes zero.

At the end i.e. when y becomes 0 we will get following table of x any y co-ordinates.

At last replicate these points to rest of the quadrants to get full ellipse.

X	Y
0	5
1	4
2	3
3	2
3	1
3	0

b) Explain the steps for 2D rotation about arbitrary point. (10 M)

Ans:

- **Rotation about an Arbitrary Point:-**
- To rotate an object about an arbitrary point, (Xp ,Yp) we have to carry out three steps:
    - Translate point (Xp, Yp) to the origin.
    - Rotate it about the origin and,
    - Finally, translate the centre of rotation back where it belongs (See figure 1.).
- we have already seen that matrix multiplication is not commutative, i.e. multiplying matrix A by matrix B will not always yield the same result as multiplying matrix B by matrix A. Therefore, in obtaining composite transformation matrix, we must be careful to order the matrices so that they correspond to the order of the transformations on the object. Let us find the transformation matrices to carry out individual steps.

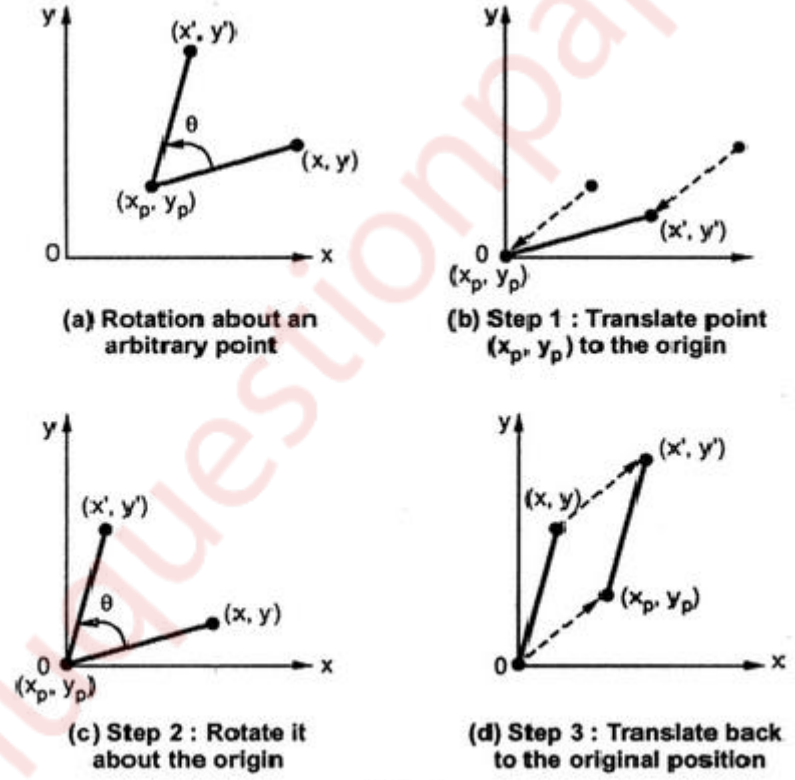


Fig. 1

The translation matrix to move point (x_p, y_p) to the origin is given as, x_p

$$T_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xp & -yp & 1 \end{bmatrix}$$

The rotation matrix for counterclockwise rotation of point about the origin is given as,

$$R = \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The translation matrix to move the center point back to its original position is given as,

$$T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

Therefore the overall transformation matrix for a counterclockwise roatation by an angle Θ about the point (xp, yp) is given as,

$$T_1 * R * T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xp & -yp & 1 \end{bmatrix} * \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

$$= \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ -xpcos\Theta + ypsin\Theta & -xpsin\Theta - ypcos\Theta & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xp & yp & 1 \end{bmatrix}$$

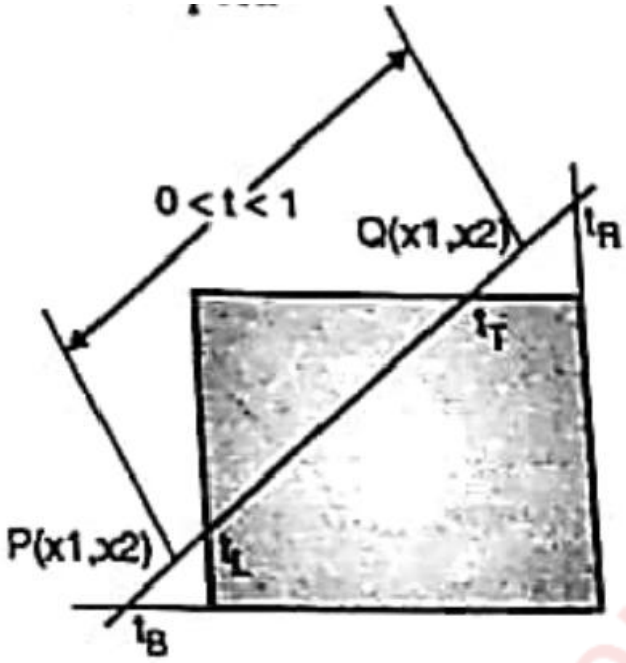
$$= \begin{bmatrix} cos\Theta & sin\Theta & 0 \\ -sin\Theta & cos\Theta & 0 \\ -xpcos\Theta + ypsin\Theta + xp & -xpsin\Theta - ypcos\Theta + yp & 1 \end{bmatrix}$$

Q.3)

a) Explain Liang Barsky line clipping algorithm. Apply the algorithm to clip the line with coordinates(30, 60) and (60, 25) against window (xmin, ymin) = (10, 10) and (xmax, ymax) = (50, 50). (10 M)

Ans:

- The Liang-Barsky algorithm uses the parametric equation of a line and inequalities describing the range of the clipping box to determine the intersections between the line and clipping box. With these intersections it knows which portion of the line should be drawn. This algorithm is more efficient than Cohen-Sutherland. The ideas for clipping line of Liang-Barsky and cyrus-Beck are the same. The only difference is Liang-Barsky algorithm has been optimized for an upright rectangular clip window. So we will study only the idea of Liang-Barsky.
- Liang and Barsky have created an algorithm that uses floating-point arithmetic but finds the appropriate ends points with at most four computations. This algorithm uses the parametric equations for a line and solves for inequalities to find the range of the parameter for which the line is in the viewport.



- Let P(X1, Y1), Q(X2, Y2) be the line which we want to study. The parametric equation of line segment from gives x-values and y-values for every point in terms of a parameter that ranges from 0 to 1. The equations are

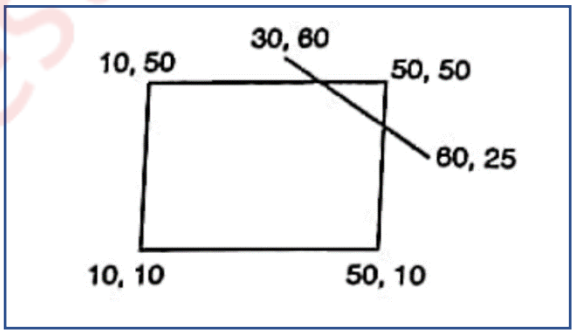
  X = X1 + (X2 – X1)*t = Y1 + dY*t

  And

  Y = Y1 + (Y2 – Y1)*t = X1 + dX*t

  We can see that when t=0, the point computed is P(x1, y1) and when t =1, the point computed is Q(x2, y2).

- Lets first draw the window and line as shown in fig.



Given things are XL = 10, YB = 10, XR = 50, YT = 50

Lets call a line AB with its coordinates as X1 = 30, Y1 = 60, X2 = 60, Y2 = 25

Now we have to find Dx and Dy as

Dx = X2 – X1 = 60 – 30 = 30

Dy = Y2 – Y1 = 25 – 60 = -35

Lets calculate the values of parameters P and Q as

P1 = -Dx = -30

P2 = Dx = 30

P3 = -Dy = -(-35) = 35

P4 = Dy = -35

Now,

Q1 = X1 – XL = 30 -10 = 20

Q2 = XR – X1 = 50 -30 = 20

Q3 = Y1 – YB = 60 -10 = 50

Q4 = YT – Y1 = 50 -60 = -10

Now lets find P,

P1 = Q1/P1 = 20/(-30) = (-2/3)

P2 = Q2/P2 = 20/(30) = (2/3)

P3 = Q3/P3 = 50/(35) = (10/7)

P4 = Q4/P4 = -10/(-35) = (2/7)

Now,

t1 = Max(-2/3, 10/7, 0) = 10/7

t2 = Min(2/3, 2/7, 1) = 2/7

Now,

X1’ = X1 + Dx*t1

= 30+30*(10/7)

= 72.71

Y1’ = Y1 + Dy*t1

= 60+(-35)*(10/7)

= 10

And with t2 it will be

X2’ = X1 + Dx*t2

= 30+30*(2/7)

= 38.57

Y2' = Y1 + Dy*t2

= 60+(-35)*(2/7)

= 50

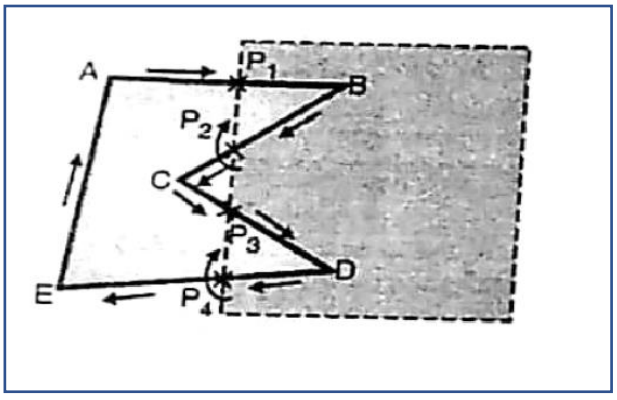
From this we will come to know that a point (38.57, 50) is an intersection point with respect to the edge of the window boundary. So we need to discard the line from point (30, 60) to (38.57, 50) and consider line (38.57, 50) to (60, 25).

---

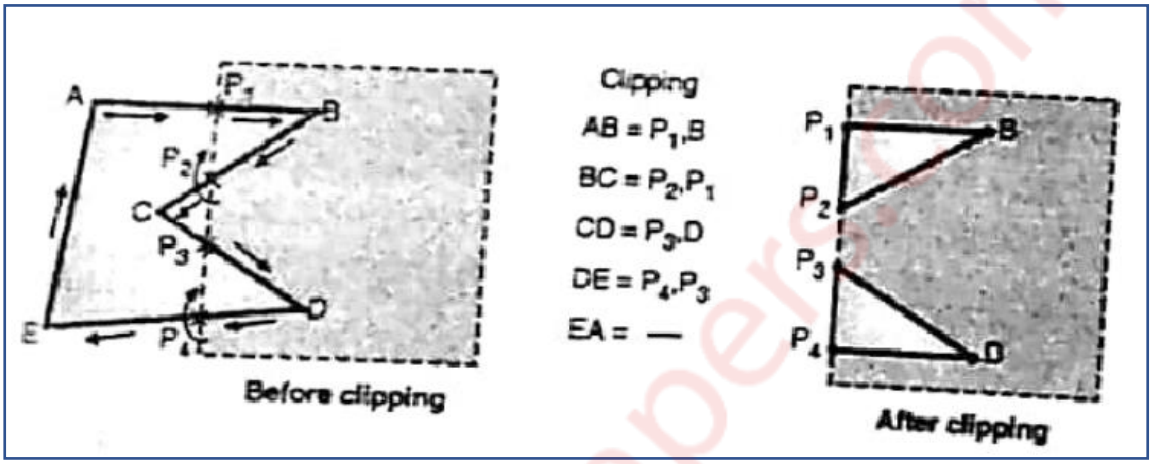
b) Explain Weiler Artherton polygon clipping algorithm with suitable example. (10 M)

Ans:

- This algorithm is one of the generalized polygon clipping algorithm. In Sutherland-Hodgeman algorithm, we always follow the path of polygon. But in this case sometime we are selecting polygon path and sometime window path. When to follow which path, that will depend on polygon processing direction, and whether a pair of polygon processing direction, and whether a pair of polygon vertices currently being processed represents an outside to inside pair or an inside to outside pair.
- We can follow polygon processing path either clockwise or anticlockwise. When we are following a path of polygon in clockwise direction at that time we have to use following rules.
- We have to follow the polygon boundary, same as that of Sutherland-hodgeman algorithm, if the vertex pair is outside to inside.
- We have to follow the window boundary in clockwise direction, if the vertex pair is inside to outside.
- Consider the concave polygon as shown in fig.
- Let us process the polygon in clockwise path. Here our vertex list will be the set of all vertices i.e. {A, B, C, D}.



- For edge Ab, as we are moving from outside to inside so store intersection point P1 and second point i.e. B. as we are following polygon in clockwise direction, so we have to consider next edge as BC. Here we are moving from inside to outside. So we are storing only intersection point P2. so we have to store that. It means we are storing P1 and P2. For next edge i.e. CD, again we are storing intersection point P3 and next point D. for edge DE we are storing intersection point P4 and following boundary in clockwise direction i.e. storing P3. For edge EA, as both points are outside, no vertex is stored.



- Here in Weiler-Atherton algorithm we are maintaining as such two different vertex list in single list. Here after clipping, the vertex list will be {P1, B, P2, P1, P3, D, P4. P3}. We have to draw edges by joining vertices as P1 to B, B to P2 to P1. Now there is no need to form an edge between P1 to P3. We have to continue with edge P3 to D, D to P4 and P4 to P3. We may use some logic here such as, when any vertex is stored twice in vertex list then don't form edge from that vertex to next vertex. Now we will not have edge P1P3 and edge P3P1. That's why we are sating, we are maintaining two sub array in single array of vertex.

Q.4)

a) What is window and viewport? Derive the matrix for viewport transformation. (10 M)

Ans:

Window:

1. A world-coordinate area selected for display is called a window.
2. In computer graphics, a window is a graphical control element.
3. It consists of a visual area containing some of the graphical user interface of the program it belongs to and is framed by a window decoration.

4. A window defines a rectangular area in world coordinates. You define a window with a GWINDOW statement. You can define the window to be larger than, the same size as, or smaller than the actual range of data values, depending on whether you want to show all of the data or only part of the data.

Viewport:

1. An area on a display device to which a window is mapped is called a viewport.
2. A viewport is a polygon viewing region in computer graphics. The viewport is an area expressed in rendering-device-specific coordinates, e.g. pixels for screen coordinates, in which the objects of interest are going to be rendered.
3. A viewport defines in normalized coordinates a rectangular area on the display device where the image of the data appears. You define a viewport with the GPORT command. You can have your graph take up the entire display device or show it in only a portion, say the upper-right part.

Window to viewport transformation:

1. Window-to-Viewport transformation is the process of transforming a two-dimensional, world-coordinate scene to device coordinates.
2. In particular, objects inside the world or clipping window are mapped to the viewport. The viewport is displayed in the interface window on the screen.
3. In other words, the clipping window is used to select the part of the scene that is to be displayed. The viewport then positions the scene on the output device.

1. Using this proportionality, the following ratios must be equal.

$$\frac{xv - xv_{min}}{xv_{max} - xv_{min}} = \frac{xw - xw_{min}}{xw_{max} - xw_{min}}$$

By solving these equations for the unknown viewport position (xv, yv), the following becomes true:

$$xv = S_x xw + t_x$$

$$yv = S_y yw + t_y$$

The scale factors (Sx, Sy) would be:

$$S_x = \frac{xv_{min} - xv_{min}}{xw_{max} - xw_{min}}$$

$$S_y = \frac{yv_{min} - yv_{min}}{yw_{max} - yw_{min}}$$

And the translation factors (Tx, Ty) would be:

$$t_x = \frac{xw_{max}xv_{min} - xw_{min}xv_{max}}{xw_{max} - xw_{min}}$$

$$t_y = \frac{yw_{max}yv_{min} - yw_{min}yv_{max}}{yw_{max} - yw_{min}}$$

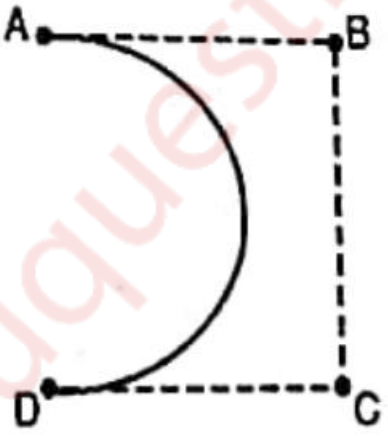
1. The position of the viewport can be changed allowing objects to be viewed at different positions on the Interface Window.
2. Multiple viewports can also be used to display different sections of a scene at different screen positions. Also, by changing the dimensions of the viewport, the size and proportions of the objects being displayed can be manipulated.
3. Thus, a zooming affect can be achieved by successively mapping different dimensioned clipping windows on a fixed sized viewport.
4. If the aspect ratio of the world window and the viewport are different, then the image may look distorted.

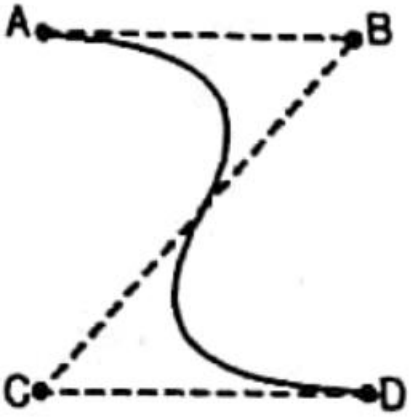
---

b) Explain what is meant by Bezier curve? State the various properties of Bezier curve. (10 M)

Ans:

- It is a different way of specifying a curve, rather same shapes can be represented by B-spline and Bezier curves. The cubic Bezier curve require four sample points, these points completely specify the curve.





- The curve begins at the first sample point and ends at fourth point. If we need another Bezier curve then we need another four sample points. But if we need two Bezier curves connected to each other, then with six sample points we can achieve it. For this, the third and fourth point of first curve should be made same as first and second point of curve.

- The equation for the Bezier curve are as follows:

$$X = X_4a^3 + 3X_3a^2(1-a) + 3X_2a(1-a)^2 + X_1(1-a)^3$$

$$Y = Y_4a^3 + 3Y_3a^2(1-a) + 3Y_2a(1-a)^2 + Y_1(1-a)^3$$

$$Z = Z_4a^3 + 3Z_3a^2(1-a) + 3Z_2a(1-a)^2 + Z_1(1-a)^3$$

- Here as the value of 'a' moves from 0 to 1, the curve travels from the first to fourth sample point. But we can construct a Bezier curve without referencing to the above expression. It is constructed by simply taking midpoints.
- Properties of Bezier curve are as follows:
    1. The basic function are real in nature.
    2. Bezier curve always passes through the first and last control points i.e. curve has same end points as the guiding polygon.
    3. The degree of polynomial defining the curve segment is one less than the number of defining polygon point.
    4. The curve generally follows the shape of the defining polygon.
    5. The direction of the tangent vector at the end points is the same as that of the vector determined by first and last segments.
    6. The curve lies entirely within the convex hull formed by four control points.
    7. The curve exhibits the variation diminishing property. This means that the curve does not oscillate about any straight line more than the defining polygon.
    8. The curve is invariant under an affine transformation.

Q.5)

a) What is meant by parallel and perspective projection? Derive matrix for perspective projection. (10 M)

Ans:

Parallel Projection:

i. In parallel projection, Z coordinate is discarded and parallel lines from each vertex on the object are extended until they intersect the view plane.

ii. The point of intersection is the projection of the vertex.

iii. We connect the projected vertices by line segments which correspond to connections on the original object.

iv. A parallel projection preserves relative proportions of objects.

v. Accurate views of the various sides of an object are obtained with a parallel projection. But not a realistic representation.

vi. Parallel projection is shown below in figure 30.

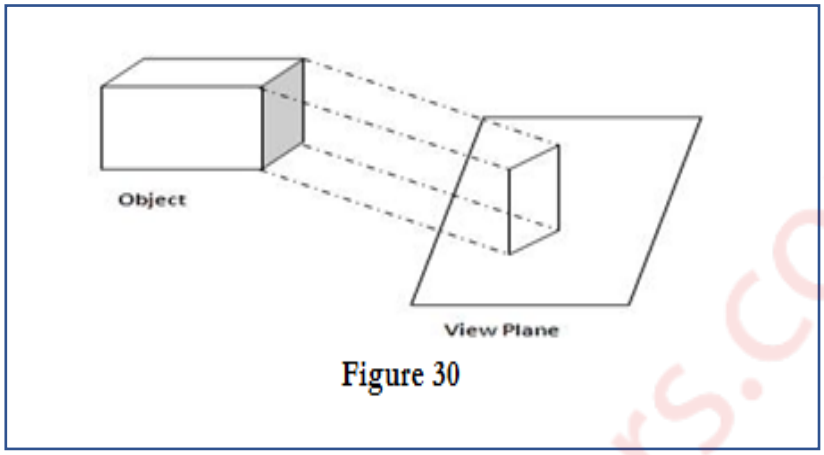


Figure 30

Perspective Projection:

i. In perspective projection, the lines of projection are not parallel.

ii. Perspective Projection transforms object positions to the view plane while converging to a center point of projection.

iii. In this all the projections are converge at a single point called the “center of projection” or “projection reference point”.

iv. Perspective projection produces realistic views but does not preserve relative proportions.

v. Projections of distant objects are smaller than the projections of objects of the same size that are closer to the projection plane.

vi. Perspective projection is shown below in figure 31.

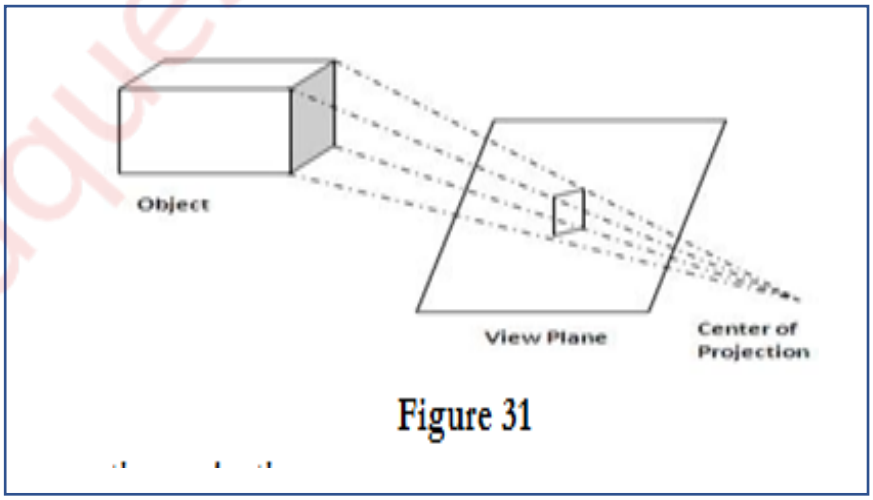
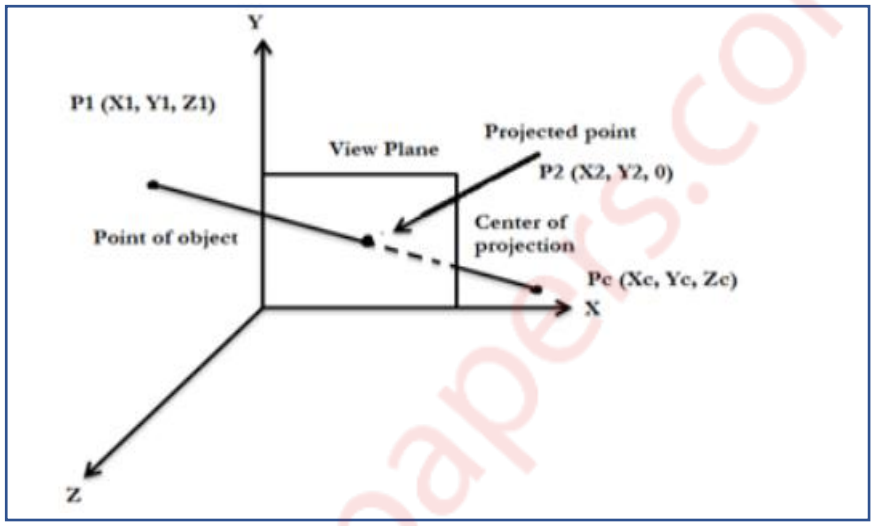


Figure 31

Matrix for perspective projection:

Let us consider the center of projection is at $P_c(X_c,Y_c,Z_c)$Pc(Xc,Yc,Zc) and the point on object is $P_1(X_1,Y_1,Z_1)$P1(X1,Y1,Z1), then the parametric equation for the line containing these points can be given as

$$X_2 = X_c + (X_1 - X_c)U$$
$$Y_2 = Y_c + (Y_1 - Y_c)U$$
$$Z_2 = Z_c + (Z_1 - Z_c)U$$



For projected point Z2 is 0, therefore the third equation can be written as

$0 = Z_C + (Z_1 - Z_C)U$

$U = -Z_C/Z_1 - Z_2$

Substituting the value of U in first two equations we get,

$X_2 = (X_C - Z_C) * (X_1 - X_C) / (Z_1 - Z_C)$

$= X_CZ_1 - X_CZ_C - X_1Z_C + X_CZ_C/Z_1 - Z_C$

$= X_CZ_1 - X_1Z_C / Z_1 - Z_C$

$Y_2 = (Y_C - Z_C) * (Y_1 - Y_C) / (Z_1 - Z_C)$

$= Y_CZ_1 - Y_CZ_C - Y_1Z_C + Y_CZ_C / Z_1 - Z_C$

The above equation can be represeented in the homogeneous matrix form as given below,

$$[X_2\ Y_2\ Z_2\ 1] = [X_1\ Y_1\ Z_1\ 1] \begin{bmatrix} -ZC & 0 & 0 & 0 \\ 0 & -ZC & 0 & 0 \\ XC & YC & 0 & 1 \\ 0 & 0 & 0 & -ZC \end{bmatrix} \begin{bmatrix} -ZC & 0 & 0 & 0 \\ 0 & -ZC & 0 & 0 \\ XC & YC & 0 & 1 \\ 0 & 0 & 0 & -ZC \end{bmatrix}$$

Here, we have taken the center of projection as PC(XC, YC, ZC), if we take the center of projection on the negative Z axis such that

X = 0

Y = 0

$Z = -Z_C$

---

b) Explain Z buffer algorithm for hidden surface removal. (10 M)

Ans:

- Another way to handle hidden liens and surfaces is z buffers. It is also called as depth buffer algorithm. Here we are sorting the polygons according to their position in space. And then in frame buffer itself.
- We are sorting polygons which are closer to viewer. We know that frame buffer is used to store images which we want to display on monitor. Here for visibility test we are making use of z buffer along with frame buffer.
- The z buffer is a large array to hold all the pixels of display z buffer is somewhat similar to frame buffer. In frame buffer we are having arrays to store x and y co-ordinates of an image. Similarly z buffer contains z co-ordinates of pixels which we want to display.
- When there is nothing to display on monitor i.e. frame buffer is empty, at that time we have t initialize z buffer elements to a very large negative values. A large negative values on z axis represents a point beyond which there is nothing i.e. setting background colour. $z_{buffer}(x, y) = z_{initialvalue,}$
- If the new surfaces has z value greater than z_{buffer} then it lies in front. So we have to modify the contents of $z_{buffer}(x, y)$ by new z values and set the pixel value at (x, y) to the colour of the polygon at (x, y).
  i.e. if ($z(x, y) > z_{buffer}(x, y)$)
  {
  $z_{buffer}(x, y) = z(x, y)$
  put pixel (x, y, polygon-colour)
  }
- if the new value of new surface is smaller than $z_{buffer}(x, y)$ then it lies behind some polygon which was previously entered. So new surface should be hidden and should not be displayed. The frame buffer and z_{buffer} should not be modified here. Here the comparison should be carried out by using pixel by pixel method.
- For example, in figure (e) shown below, among three surfaces, surface S1 has the smallest depth at view position (x, y) and hence highest z value. So it is visible at that position.

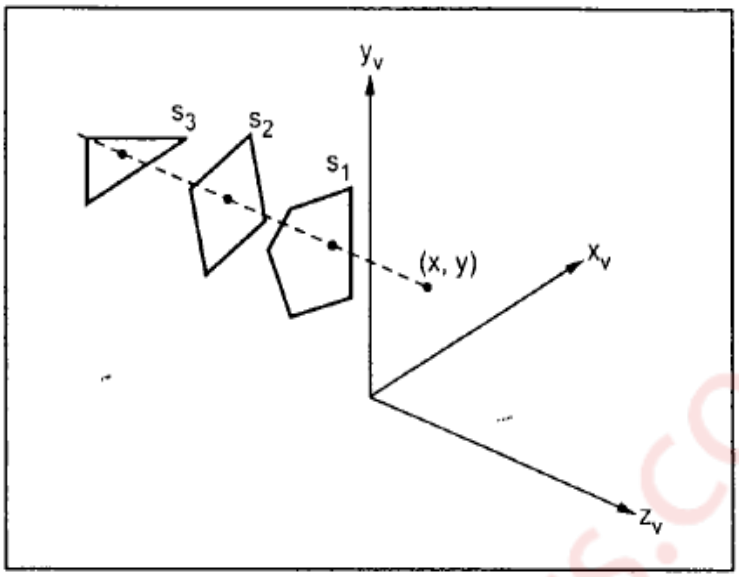


Fig. (e)

- To calculate z-values, the plane equation

  Ax + By + Cz + D =0

  is used where (x, y, z) is any point on the Plane, and the coefficient A, B, C and D are constants describing the spatial properties of the Plane.

- Therefore, we can write

  Z = (-AX -BY – D) / C

  Note, if at (x, y) the above equation evaluates to z1, then at (x + Δx, y) the value of z. is,

  Z1 = A / C (Δx)

  Only one subtraction is needed to calculate z(x + 1. y), given z(x, y). Since the quotient A/C is constant and Δx = 1. A similar incremental calculation can be performed to determine the first value of z on the next scan line, decrementing by B/C for each Δy.

  - **Advantages:**
    - It is easy to implement.
    - As z buffer algorithm processes one object at a time, total number of objects can be large.
  - **Dis-advantages:**
    - It requires lots of memory as we are storing each pixels z value.
    - It is a time consuming process as we are comparing each and every pixel.

Q.6) Write Short notes on **(20 M)**

a) Koch Curve

Ans:

- The Koch curve can be drawn by dividing line into 4 equal segments with scaling factor 1/3 and middle two segments are so adjusted that they form adjacent sides of an equilateral triangle as shown in the Fig. 16(a) .This is the first approximation to the koch curve.
- To apply the second approximation to the Koch curve we have to repeat the above process for each of the four segments. The resultant curve is shown in Fig. 16(b).

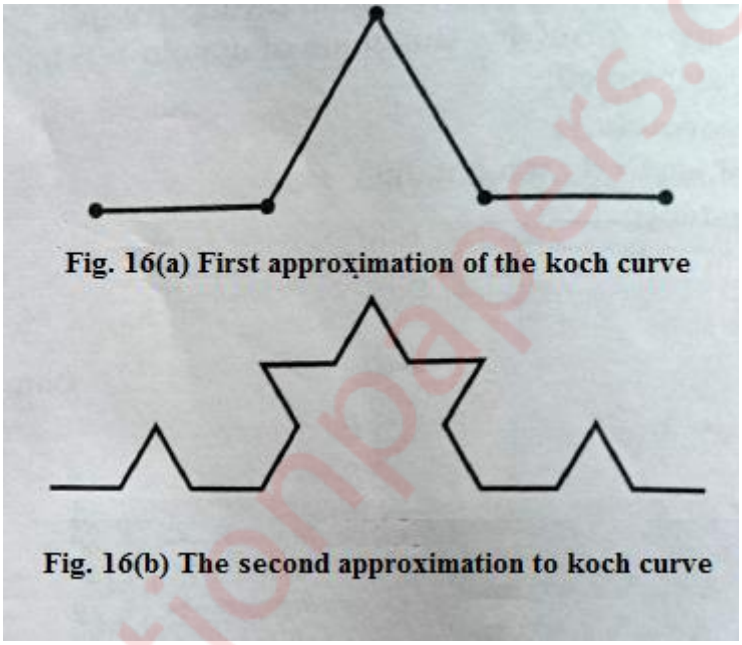


Fig. 16(a) First approximation of the koch curve

Fig. 16(b) The second approximation to koch curve

- The resultant curve has more wiggles and its length is 16/9 times the original length.
- From the above figures we can easily note following points about the koch curve :
    - Each repetition increases the length of the curve by factor 4/3.
    - Length of curve is infinite.
    - Unlike Hibert's curve, it doesn't fill an area.
    - It doesn't deviate much from its original shape.
    - If we reduce the scale of the curve by 3 we find the curve that looks just like the original one; but we must assemble 4 such curves to make the originals, so we have

$$4 = 3^D$$

Solving for D we get

$$D = \log_3 4 = \log 4 / \log 3 = 1.2618$$

- Therefore for koch curve topological dimension is 1 but fractal dimension is 1.2618.
- From the above discussion we can say that point sets, curves and surfaces which give a fractal dimension greater than the topological dimension are called fractals The Hilbert's curve and koch curves are fractals, because their fractal dimensions (respectively, 2 and 1.2618) are greater than their topological dimension which is 1.

---

b) Sweep representation

Ans:

- Sweep representations are used to construct three dimensional objects from two dimensional shape .
- There are two ways to achieve sweep:
  - Translational sweep
  - Rotational sweep.
- In translational sweeps, the 2D shape is swept along a linear path normal to the plane of the area to construct three dimensional object. To obtain the wireframe representation we have to replicate the 2D shape and draw a set of connecting lines in the direction of shape, as shown in the figure (8)

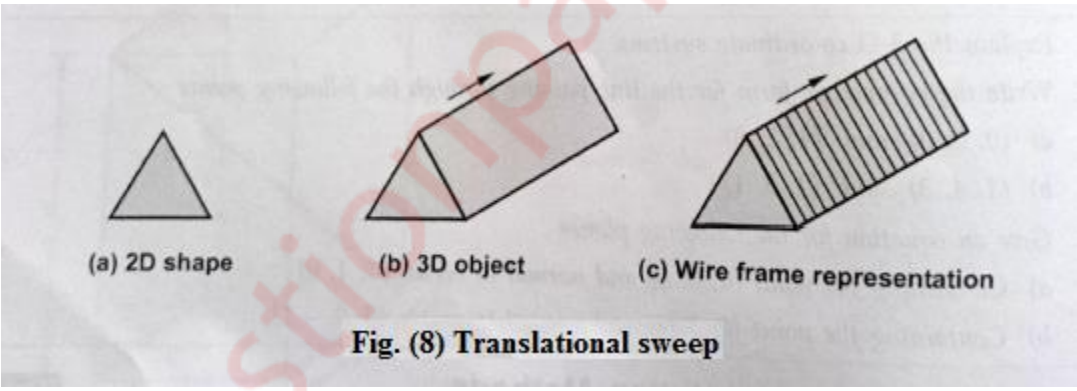
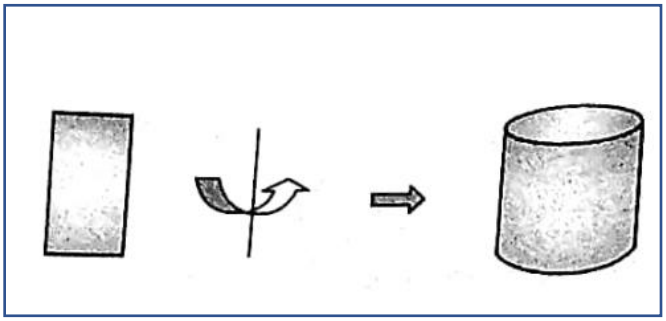


Fig. (8) Translational sweep

- In general we can specify sweep constructions using any path. For translation we can vary the shape or size of the original 2D shape along the sweep path. For rotational sweeps, we can move along a circular path through any angular distance from 0° to 360°.
- These sweeps whose generating area or volume changes in size, shape or orientation as they are swept and that follow an arbitrary curved trajectory are called general sweeps. General sweeps are difficult to model efficiently for example, the trajectory and object shape may make the swept object intersect itself, making volume calculations complicated.
- In rotational sweep, the 2D is rotated about an axis of rotation specified in the plane of 2D shape to produce three dimensional objects.



- Sweep representation are widely used in compute vision. However, the generation of arbitrary objects rather difficult using this technique.

---

c) Gouraud and Phong shading

Ans:

Gouraud Shading:

1. Gouraud surface shading was developed in the 1970s by Henri Gouraud.
2. It is the interpolation technique.
3. Intensity levels are calculated at each vertex and interpolated across the surface.
4. Intensity values for each polygon are matched with the values of adjacent polygons along the common edges.
5. This eliminates the intensity discontinuities that can occur in flat shading.
6. To render a polygon, Gouraud surface rendering proceeds as follows:
    - Determine the average unit normal vector at each vertex of the polygon.
    - Apply an illumination model at each polygon vertex to obtain the light intensity at that position.
    - Linearly interpolate the vertex intensities over the projected area of the polygon

Illumination values are linearly interpolated across each scan-line as shown in figure 41.

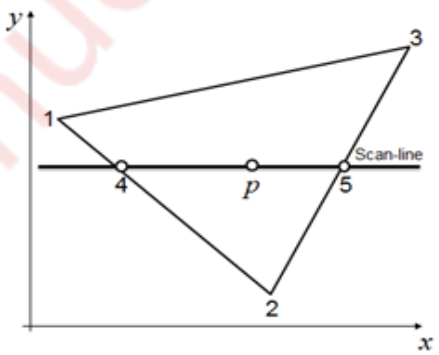


Figure 41

$$I_4 = \frac{y_4 - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y_4}{y_1 - y_2} I_2$$

$$I_5 = \frac{y_5 - y_2}{y_3 - y_2} I_3 + \frac{y_3 - y_5}{y_3 - y_2} I_2$$

$$I_p = \frac{x_5 - x_p}{x_5 - x_4} I_4 + \frac{x_p - x_4}{x_5 - x_4} I_5$$

Phong Shading:

1. A more accurate interpolation based approach for rendering a polygon was developed by Phong Bui Tuong.
2. Basically the Phong surface rendering model is also called as normal-vector interpolation rendering.
3. It interpolates normal vectors instead of intensity values.
4. To render a polygon, Phong surface rendering proceeds as follows:
5. Determine the average unit normal vector at each vertex of the polygon.
6. Linearly interpolate the vertex normal over the projected area of the polygon.
7. Apply an illumination model at positions along scan lines to calculate pixel intensities using the interpolated normal vectors as shown in figure 42

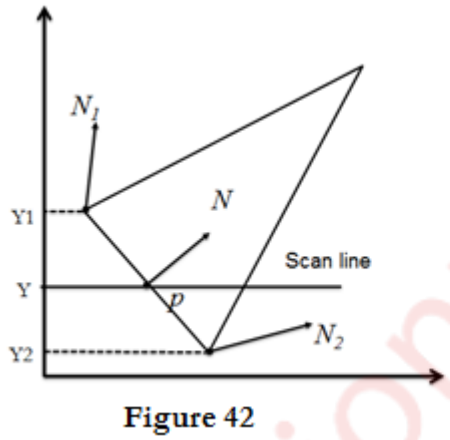


Figure 42

$$N = \frac{y - y_2}{y_1 - y_2} N_1 + \frac{y_1 - y}{y_1 - y_2} N_2$$

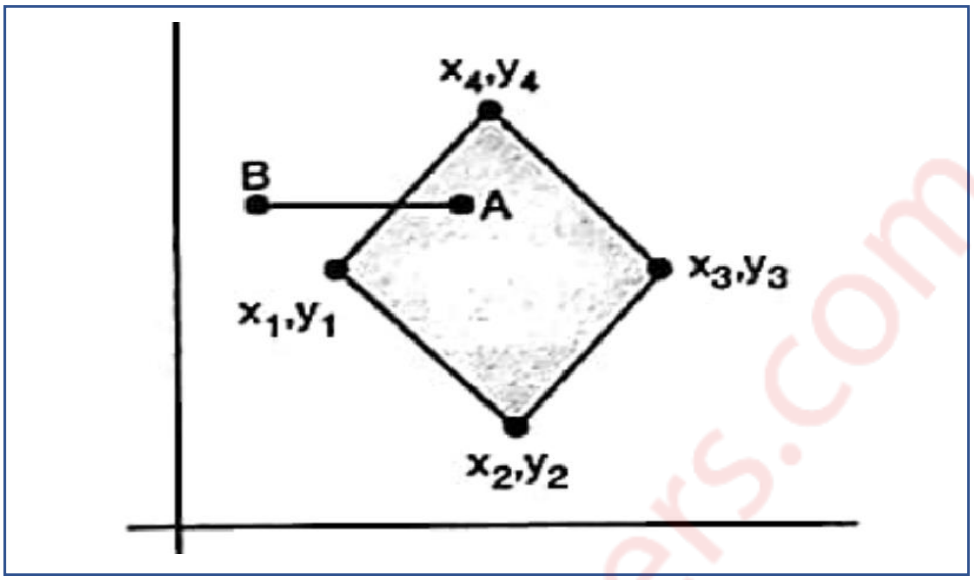
---

d) Inside Outside test.

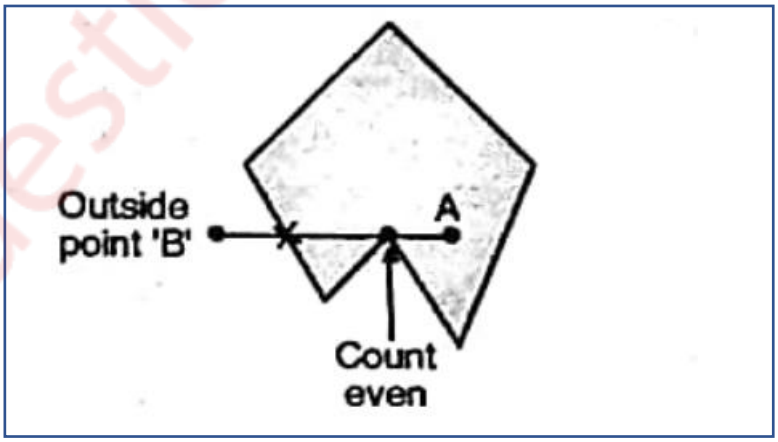
Ans:

- One method of doing this is to construct a line segment between a point in question i.e. point to test whether inside or outside, and a point which is surely outside the polygon. But ow we are going to find out a point which is surely outside the polygon? It is very easy to find out that point.
- For example, pick a point with an x co-ordinate smaller than the smallest co-ordinate of the polygons vertices and the y co-ordinate will be any y value, or for simplicity we will take y value same as the y value of point in question.
- In this case point A is one, which we want to check i.e. whether point A is inside polygon or not.
- As we are using arrays to store vertices of polygon we can easily come to know the vertex which is having lowest value of x and that is x_1. So we have to select a point smaller than x_1 and generally we select same value of y as that of point A,

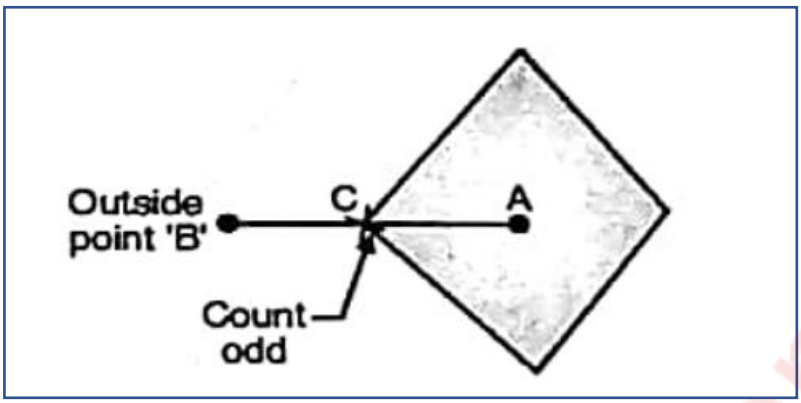
in order to draw straight line. But even if you select any y value for outside point, it will not make any difference.



- Then count how many intersections are occurring with polygon boundary by this line till the point in question i.e. 'A', is reached. If there are an odd number of intersections then the point is outside the polygon.
- This is called even-odd method to determine interior points of polygon.
- But this even odd test fails for one case i.e. when the intersection point is vertex. To handle this case we have to make few modifications we must look at other end points of two segments of a polygon which meet at this vertex.
- If these points lie on the same side of the constructed line AB, then the intersection point counts as an even number of intersection.



- But if they lie on opposite sides of the constructed line AB, then the intersection point is counted as a single intersection.



- In this case the constructed line AB intersects polygon in a vertex ‘C’. here the other points of both the line segments which meets at vertex ‘C’ are lieing in opposite sides of this constructed line Ab. So the intersection at vertex ‘C’ is counted as odd count i.e. 1. The total number of C intersection made by this line with polygon is 1 i.e. odd and as it is odd therefore the point ‘A’ is inside.
