Important Instructions to examiners:

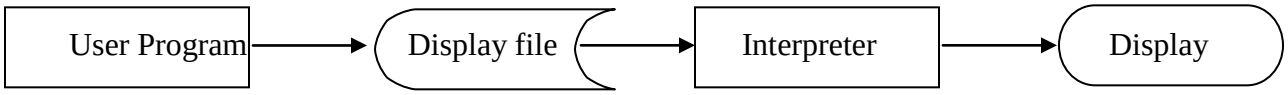
1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

1. (a) Attempt any SIX of the following : Marks 12

i) What is display file interpreter?

(Diagram –1 Mark, Explanation -1 Mark)

Picture image is firstly stored in display file in form of commands and then some program executes these commands and convert them into appropriate image on screen. The program which converts these command into actual picture is called as Display file interpreter and its sever as interface between graphics program and display device.



ii) State the Basic criterias used for line drawing.

(All four criteria are expected ½ Mark for each criteria)

General criteria of any line drawing algorithm are as follow.

1. Line must appear straight.
2. Line must start and end properly.
3. Line should be drawn at rapid pace.
4. Line must have constant intensity to their length.

iii) What are 2-D transformation?

(Definition- 1 Mark, Types- 1 Mark)

A 2-D transformation is an operation by virtue of which there is a significant change in size, location, angle or shape of an object which has following types

Translation

Scaling

Rotation

Shearing

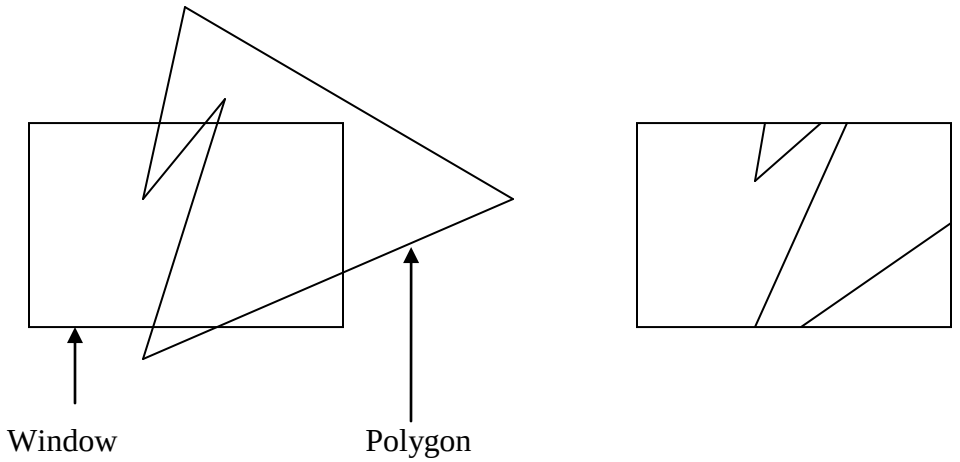
Reflection

iv) What do you mean by polygon clipping?

(Clipping definition- 1 Mark, Diagram -1 Mark)

Polygon Clipping:

The process which divides each elements of picture into its visible and invisible position ,allowing invisible portion to be discarded is called as clipping. It a Polygon is process for line clipping using line clipping algorithm ,it may display series of unconnected line segments as shown in a Figure.



(a) Before Clipping (b) After Clipping

v) What are different line drawing algorithm? What is difference between them?

(List -1 Mark, Difference 1-Mark)

There are two line drawing algorithm:

- DDA (Digital Differential Algorithm)
- Bresenham's Line drawing algorithm

Difference:

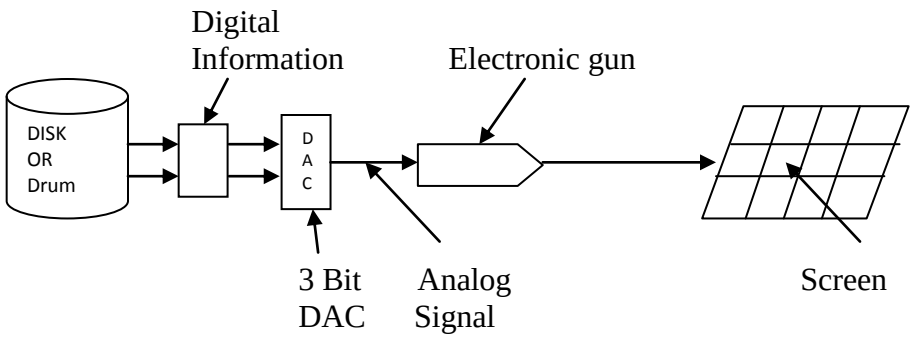
1. Bresenham's Line drawing algorithm
    - In bresenham's line drawing algorithm shape value is constant i.e.1
    - It does not flicker.
    - No round of problem.
    - It is fast.
2. DDA (Digital Differential Algorithm)
    - In DDA shape value is not constant it depends on its length
    - It has flicker problem.
    - IT has round up Problem.
    - It is comparatively Slow

vi) What is frame buffer.

(Explanation -2 Marks)

Frame buffer is a large part of computer memory used to store display image. Different kind of memory can be used for frame buffers like drums, disk or IC – shift registers.

To generate a pixel of desired intensity to read the disk or drum. The information stored in disk or drum is in digital for, hence it is necessary to convert it into analog from using DAC and then this analog signal is used to generate the pixel.



vi) List any four properties of Bezier curve.

(Any Four properties are expected. For each property ½ Mark)

Properties of Bezier Curve

1. It always passes through first and last control point.
2. Tangent vectors at ends of the curve have same direction as first and last polygon spans respectively.
3. It follows the shape of defining polygon.
4. It is contained within the convex hull of defining polygon.
5. Degree of polynomial defining the curve segments is equal to the total number of control points minus 1.
6. Order of polynomial defining the curve segments is equal to the total number of control points
7. Basic functions of Bezier curve are real.

viii) What do you mean by term Rasterscan?

(For explanation -2 Marks)

In a raster scan, an image is subdivided into a sequence of (usually horizontal) strips known as "scan lines". Each scan line can be transmitted in the form of an analog signal as it is read from the video source, as in television systems, or can be further divided into discrete pixels for processing in a computer system. This ordering of pixels by rows is known as raster order, or raster scan order. Analog television has discrete scan lines (discrete vertical resolution), but does *not* have discrete pixels (horizontal resolution) – it instead varies the signal continuously over the scan line. Thus, while the number of scan lines (vertical resolution) is unambiguously defined, the horizontal resolution is more approximate, according to how quickly the signal can change over the course of the scan line.

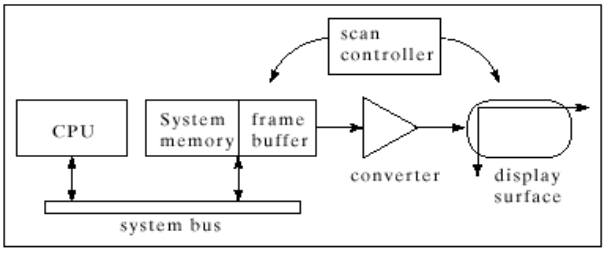


Fig. 1.1Simple Raster graphics Systems

b) Attempt any TWO of following: Marks 8

i) Explain 3 D Scaling along with its matrix representation.

(Explanation-2 Marks, Matrix -2 Marks)

Scaling Transformation:-

A scaling can be represented by a scaling matrix. To scale an object by a vector V= [Sx, Sy, Sz], each point P = [x y z] would need to be multiplied with this scaling matrix.

$$S_v = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix}$$

$$P * S_v = [\,X\ Y\ Z\,] * \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix} = [S_xX \quad S_yY \quad S_zZ\,]$$

Such a scaling changes the diameter of an object by a factor between te scale factors, the area by a factor between the smallest and the largest product of two scale factors, and the volume by the product of all three.

In homogeneous coordinates, since translation cannot be accomplished with a 3 X 3 matrix. To scale an object by a vector V= [S_x S_y S_z] each homogeneous vector P = [X Y Z 1] would need to be multiplied with the scaling matrix.

$$S_v = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P * S_v = [\,X\ Y\ Z\ 1\,] * \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [S_xX \quad S_yY \quad S_zZ\ 1]$$

ii) Describe 2D sharing with diagram.

(Explanation-2 Marks, Diagram 2-Marks)

It is a transformation which slants or bends an object to specified direction. There are two types of shearing transformation available in computer graphics. One which slants x coordinate values is

known as X shearing and one that slants y coordinate values is known as Y shearing. Irrespective of shearing only one co-ordinate is change its coordinate and other values are same.

X-Shearing:- It is a process by which x coordinates are change to specified direction. In X shearing the values of y coordinates are kept same and values of x coordinates are change as a result of which vertical lines are tilt to right or left which results into slanting of entire object to one direction i.e. horizontally only.

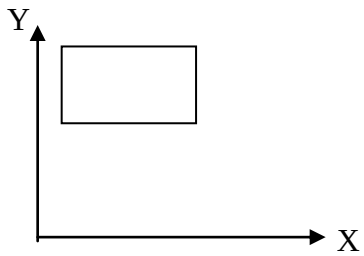


Fig 3.5(a) Before Shearing of Object

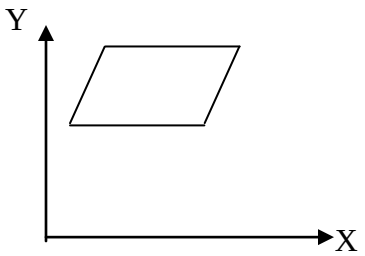


Fig 3.5(b) After Shearing of object

Fig 3.5 gives an idea about Y-shearing operation in computer graphics. Fig 3.5(a) gives an object without shear operation on it object look like a rectangle. When X-shearing operation is used on it object get slanted towards horizontally only. Comparing objects in both figures the base point is remains same just object gets slanted. The matrix for X-shearing is given by

$$X\_sh = \begin{bmatrix} 1 & 0 \\ Sh_x & 1 \end{bmatrix}$$

Y Shearing: -

It is a process by which y coordinates are change to specified direction. In Y shearing the values of x coordinates are kept same and values of y coordinates are change as a result of which horizontal lines are slanted to upward or downward which results into slanting of entire object to vertically only.

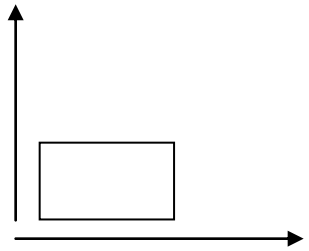


Fig 3.5(c) Before Shearing of Object

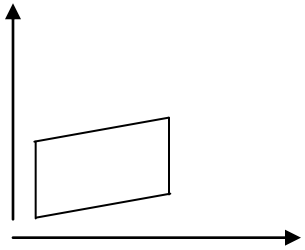


Fig 3.5(d) After Shearing of object

Fig 3.5 gives an idea about Y-shearing operation in computer graphics. Fig 3.5(a) gives an object without shear operation on it object look like a rectangle. When Y-shearing operation is used on it object get slanted towards vertically only. Comparing objects in both figures the base point is remains same just object gets slanted similar X-shearing.

$$Y\_sh = \begin{bmatrix} 1 & 0 \\ Sh_Y & 1 \end{bmatrix}$$

iii) Find the homogenous matrix for –

A point is rotated counter of clock wise by an angle of 45^0 .Find the rotation matrix and the resultant point.

(Homogenous matrix -1Mark , Rotation matrix-1 Mark and resultant point -2Marks)

[Note:- Consider any Co-ordinates of points]

The matrix for counter clockwise rotation by θ degrees about the origin

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Consider the point P = $\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

Then θ=45^o at counter clock wise direction

We have

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos 45 & -\sin 45 & 0 \\ \sin 45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{pmatrix} X \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{2}{\sqrt{2}} \\ 1 \end{pmatrix}$$

2. Attempt any FOUR of the following: Marks 16

a) Write algorithm for entering the polygon into the display file.

(For algorithm-4 Marks)

Algorithm:

1. Read AX and AY of length N
2. i = 0
   DF_OP[i] ← N
   DF_x[i] ← AX[i]
   DF_y[i] ← AY[i]
   i ← i+1

3. do
   {
   DF_OP[i] ← 2
   DF_x[i] ← AX[i]
   DF_y[i] ← AY[i]
   i ← i +1
   }
   While (i< N)

4. DF_OP[i] ←2
   DF_x[i] ← AX[0]
   DF_y[i] ← AY[0]

5. Stop.

b) Explain graphics mode graphics function.

(Each mode-2 Marks Syntax- 2 Marks)

Initgraph( ) : - It is used to initialize graphic mode.

Syntax:- initgraph (int driver, int mode, char path);

Driver:- this argument gives the graphics driver to be used and it interfaces with display adapter. Some of available graphics drivers are CGA, EGA, VGA etc.

Closegraph( ):- It is used to close graphics mode, when one exit from graphics mode one shall restore the system to the previous display mode, closegraph function restores the previous display mode.

Syntax:-

Closegraph( );

detectgraph(): - It is used to detect the graphics mode and graphics driver allocated within system.

Syntax: - detectgraph(&gd, &gm)

c) Write advantages and disadvantages of:

i) BMP

ii) GZF

(BMP-2 Marks , GIF -2 Marks)

[Note :-instead of GZF it should be 'GIF']

i) BMP:

Advantages

- Simple to use
- It is display device independent

Disadvantages

- Large file size
- Though display device independent they are limited to windows platform
- Cannot store multiple images in one file.

ii) **GIF:**

Advantages:-

- .GIF is an indexed image. Meaning it can only have 256 colors. However .GIF can be animated and they can have a transparent background.
- .GIF makes sharp, crisp images.
- .GIF images are used in web design.

Disadvantages:-

- Limited to maximum 256 colors,
- No transparency

d) Explain working of Liquid crystal display.

(Explanation - 4 Marks)

This display use nematic (thread like) liquid crystal compounds that tend to keep the axes of rod shaped molecules aligned .These nematic compounds have crystalline arrangement of molecules aligned. these nematic compounds termed as liquid crystal display.

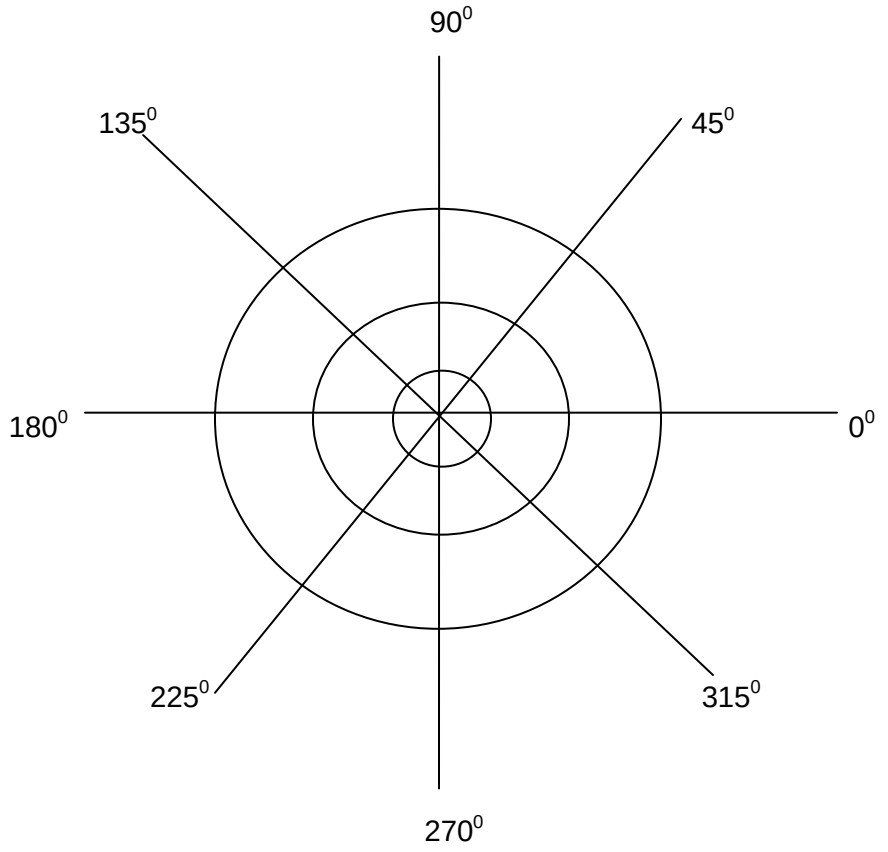
Liquid crystal material is filled in between two glass plates. One glass plate have horizontal transparent are built into same glass plate & other end glass plate have vertical light polarizer and vertical conductors are built into it. The intersection of two conductors defines pixel position. polarized light passing through material is twisted so that it will pass through opposite polarizer .the light is then reflected back to the viewer. To turn OFF pixel, we apply voltage to two intersecting conductors to align molecules so that light is not twisted. Picture definitions are stored in refresh buffer and screen is refreshed at the rate of 60 frames per second.

e) What is co-ordinate system? Explain polar co-ordinate system with diagram.

(For definition 1-Mark, Diagram -1 Mark and Explanation -2 Marks)

Any graphics always involve a reference point called as 'origin' and linear and angular distances measured along one or more reference directions are called as co-ordinates.

Diagram:



Polar Co-ordinate System:

The polar co-ordinate system is two –dimensional co-ordinate systems in which each point on plane is determined by angle and a distance.

2D Cartesian (x,y) co-ordinate corresponds to polar (r ,θ) co-ordinates i.e. point P is addressed by distance r of radius vector from origin and angle θ which it makes in counter clockwise rotation from references X-axis.

Each point in the polar in the polar co-ordinate system can be describe with the to polar co-ordinates, which are usually called r(the radial co-ordinate) and θ (the angular co-ordinate, polar angle, sometimes represented as t).The r co-ordinate represents the radial distance from the pole (pole is equivalent to the origin in the Cartesian system), and the θ co-ordinate represents the

anticlockwise (counterclockwise) angle from the 0^0 ray (sometimes called the polar axis), known as the positive X-axis on the cartesian co-ordinate plane.

f) Explain Viewing transformation with normalization transformation.

(Viewing- 2 Marks, Normalization -2 Marks)

Viewing transformation Pictures are defined any stored into memory using any convenient Cartesian co-ordinate System is called as World Co-ordinate Systems(WCS). When picture is to be displayed on display device it is measured in physical Device Co-ordinate System (PDCS) corresponding to the display device. Therefore to display any image on screen, we need to map it from stored World Co-ordinate Systems to appropriate Physical Device Co-ordinate system (PDCS). Therefore in general the process of mapping of picture from World Co-ordinate systems to Physical Device Co-ordinate system is referred as "Viewing Transformation".

Normalisation transformation- It maps world coordinate system to normalized device coordinate system (NDCS), AS different display devices have different screen sizes measured in pixels we need to make our program display device independent. This display device independent unit system is called as normalised device coordinate system.

3. Attempt any FOUR of the following: Marks 16

a) Write Steps for Bresenham's Line Drawing Algorithm

(Complete algorithm- 4 Marks)

Algorithm:-

Step 1: Take input for two end point of a line to be drawn and assume that left endpoint in $(x_0,y_0$

Step 2: Load (x_0, y_0) into the frame buffer; i.e. plot the first points.

Step 3: Calculate constants Δx, Δy $2\Delta y$, and $2\Delta y$-$2\Delta x$, and obtain the starting values for the decision parameter.

$$p_0 = 2\Delta y - 2\Delta x$$

Step 4: At each x_k along the line, starting at k = 0, perform the following test:

if $p_k < 0$, the next point to plot is (x_k+1,y_k) and

$$p_k+1 = p_k+2\Delta y$$

Otherwise, the next point to plot is (x_k+1, y_k+1) and

$$p_k+1 = p_k+2\Delta y - 2\Delta x$$

Step 5: Repeat step 4 Δx times

b) Write midpoint subdivision algorithm.

(Complete algorithm - 4 Marks)

Algorithm: -

Step 1: Scan two end points for the line $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$

Step 2: Scan corners for the window as (Wx_1, Wy_1) and (Wx_2, Wy_2)

Step 3: Assign the region codes for endpoints P_1 and P_2 by initializing code with 0000.

Bit 1 - if $(x < Wx_1)$

Bit 2 - if $(x > Wx_2)$

Bit 3 - if $(y < Wy_2)$

Bit 4 - if $(y > Wy_1)$

Step 4: Check for visibility of line P1, P2

- If region codes for both end points are zero then the line is visible, draw it and jump to step 6.
- If region codes for end points are not zero and the logical Anding operation of them is also not zero then the line is invisible, reject it and jump to step 6.
- If region codes for end points does not satisfies the condition in 4(i) and 4(ii) then line is partly visible.

Step 5: Divide the partially visible line segment in equal parts and repeat steps 3 through 5 for both subdivided line segments until you get completely visible and completely invisible line segments.

Step 6: Exit

c) Distinguish between windowing and clipping with example.

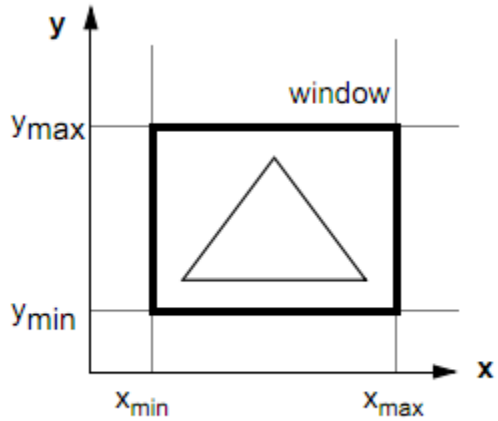
(Definition- 1 Mark, each example 1 Mark each)

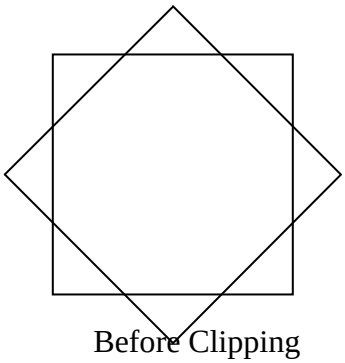
Windowing:-

The process of selecting and viewing the picture with different views is called windowing

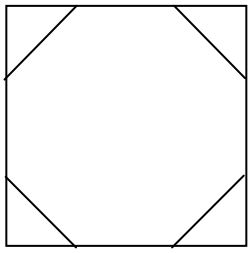
Clipping: -

The process of which divides each element of the picture into its visible and invisible portions allowing the invisible portion to be discarded is called as clipping.





Before Clipping



After Clipping

d) Give example of inside outside test for polygon

(2 Marks for even odd method, 2 Marks for winding number method)

Inside-Outside Test of Polygon:-

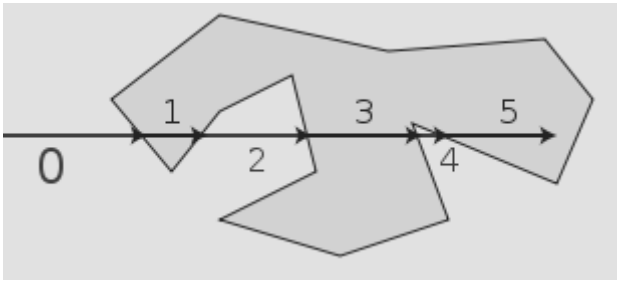
In computational geometry, the **point-in-polygon** (**PIP**) problem asks whether a given point in the plane lies inside, outside, or on the boundary of a polygon. It is a special case of point location problems and finds applications in areas that deal with processing geometrical data, such as computer graphics, geographical information systems (GIS), motion planning, and CAD.

An early description of the problem in computer graphics shows two common approaches.

1. Even-Odd Rule Algorithm.
2. Winding Number Algorithm.

1. Even-Odd Rule: -

One simple way of finding whether the point is inside or outside a simple polygon is to test how many times a ray starting from the point intersects the edges of the polygon. If the point in question is not on the boundary of the polygon, the number of intersections is an even number if the point is outside, and it is odd if inside. This algorithm is sometimes also known as the crossing number algorithm or the even-odd rule algorithm.



Even-Odd Rule Algorithm

2. Winding Number Algorithm: -

Another algorithm is to compute the given point's winding number with respect to the polygon. If the winding number is non-zero, the point lies inside the polygon. One way to compute the winding number is to sum up the angles subtended by each side of the polygon. However, this involves costly inverse trigonometric functions, which generally makes this algorithm slower than the ray casting algorithm. Luckily, these inverse trigonometric functions do not need to be computed. Since the result, the sum of all angles, can add up to 0 or 2*PI (or multiples of 2*PI) only, it is sufficient to track through which quadrants the polygon winds, as it turns around the test point, which makes the winding number algorithm comparable in speed to counting the boundary crossings.

e) Write DDA line drawing algorithm.

(Complete algorithm- 4 Marks)

Algorithm: -

Step 1: Read the line end points (x_1, y1) and (x_2, y_2) such that they are not equal.

Step 2: $\Delta x = | x_2 - x_1 |$ and $\Delta y = | y_2 - y_1 |$

Step 3: if ($\Delta x \geq \Delta y$) then

length = Δx

else

length = Δy

end if

Step 4: $\Delta x = (x_2 - x_1)/length$

$\Delta y = (y_2 - y_1)/length$

Step 5: $x = x_1 + 0.5\ sign(\Delta x)$

$y = y_1 + 0.5\ sign(\Delta y)$

Step 6: i =1

while(i $\leq$ length)

{

plot (integer (x), integer (y))

x= x + Δx

y= y + Δy

I = i+1

}

Step 7: Stop

f) Explain DDA circle drawing algorithm

(Complete algorithm - 4 Marks)

Algorithm: -

Step 1: Read the radius r, of the circle and calculate value of ε.

Step 2: start_x = 0

start_y = r

Step 3: x_1 = start_x

y_1= start_y

Step 4: do

{

$x_2 = x_1 + \varepsilon\ y_1$

$y_2 = y_1 - \varepsilon\ x_2$

plot (int (x_2), int (y_2))

$x_1 = x_2$;

$y_1 = y_2$;

}while(y_1 – start_y) < ε or (start_x - x_1)> ε

Step 5: Stop

4. Attempt any TWO of the following: Marks 16

a) Define Polygon. Give types of polygon. Explain how polygon is represented.

(Polygon definition- 2 Marks, Type- 2 Marks, 4 Marks- for polygon representation)

Definition: A closed plane figure is having three or more sides is called as polygon. Triangles, rectangles, and octagons are all examples of polygons. A **regular polygon** is a polygon all of whose sides are the same lengths and all of whose interior angles are the same measures.

Types of Polygon

The polygons are divided into two categories as,

1) Convex Polygon. 2) Concave Polygon.

1) Convex Polygon:-

A convex polygon is a polygon in which the line segment joining any two points the polygon lies completely inside the polygon. So it is very easy to perform inside test on a pixel to be filled with given color.

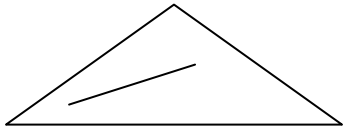


Fig (a) Convex Polygon

2) Concave Polygon: -

A concave polygon is a polygon in which the line segment joining any two points within the polygon may not lay completely inside the polygon hence to verify inside.

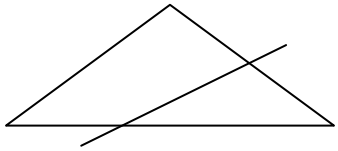


Fig (b) Conclave Polygon

Representation of Polygon:-

Each polygon has sides and edges. The end points of the sides are called the polygon vertices. To add polygon to graphics system one must first decide how to represent it. There are three approaches to represent polygons according to the graphics system:

1. Polygon drawing primitive approach
2. Trapezoid primitive approach
3. Line and point approach.

Some graphics devices supports polygon drawing primitive approache they can directly draw the polygon shapes. On such devices polygon are saved as a unit.

Some graphics devices support trapezoid primitive. In such devices trapezoids are formed from two scan lines and two line segments. Here trapezoids are drawn by stepping down the line segments with two vector generators and for each scan line, filling in all the pixels between them. Therefore every polygon is broken up into trapezoids and it is represented as a series of trapezoids.

Most of the other graphics devices do not provide any polygon support at all. In such cases polygons are represented using line and points. A polygon is represented as a unit and it is stored in the display file. In a display file polygon cannot be stored only with series of line commands as they do not specify how many of the following line command are the part of the polygon. Therefore new command is used in the display file to represent polygons.

b) Explain Hodgeman polygon Clipping algorithm.

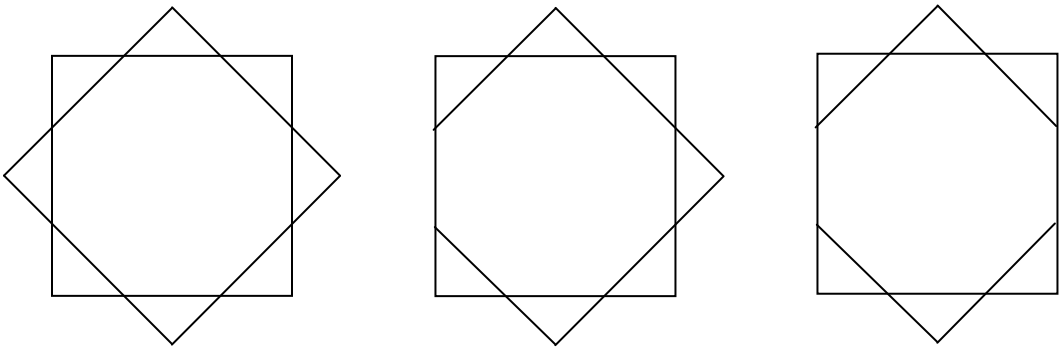
(Algorithm- 4 Marks, Explanation- 2 Marks and Diagram- 2 Marks)

Sutherland-Hodgeman Polygon Clipping.In Sutherland-Hodgeman beginning with the original set of polygon vertices, first clip the polygon against the left rectangle boundary to produce a new sequence of vertices. The new set of vertices could then be successively passed to a right boundary clipper, a top boundary clipper and a bottom boundary clipper. At each step a new set of polygon vertices us generated and passed to the next window boundary clipper. This is the logic used in Sutherland-Hodgeman algorithm. The output of algorithm is a list of polygon vertices all of which are on the visible side of clipping plane. Such each edge of the polygon is individually compared with the clipping plane. This is achieved by processing two vertices of each edge of the polygon around the clipping boundary or plane.

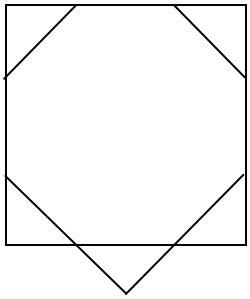
Processing Vertices.

1. If first vertex of polygon edge is outside and second is inside window boundary, then intersection point of polygon edge with window boundary and second vertex are added to output vertices set.
2. If both vertices of edge are inside window boundary, then add only second vertex to output set.
3. If first vertex of edge is inside and second is outside of window boundary then point of intersection of edge with window boundary is stored in output set.
4. If both vertices of edges are outside of window boundary then those vertices are rejected.

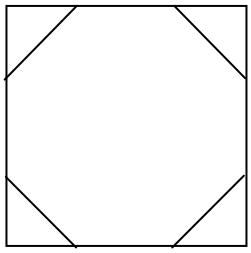
Polygon clipping in given window



Original polygon Left Clipped Right Clipped



Top Clipped



Bottom Clipped

Algorithm: -

Step 1: Read co-ordinates of all vertices of the polygon.

Step 2: Read co-ordinates of the clipping window.

Step 3: Consider the left edge of window.

Step 4: Compare vertices of each of polygon, individually with the clipping plane.

Step 5: Save the resulting intersections and vertices in the new list of vertices according to four possible relationships between the edge and the clipping boundary.

Step 6: Repeat the steps 4 and 5 for remaining edges of clipping window. Each time resultant list of vertices is successively passed to process next edge of clipping window.

Step 7: Stop

c) What is fractal? Write short note on:

1. Hilbert's Curve

2. Koch Curve

(Fractal -2 Marks, Hilbert's Curve -2 Marks, Koch curve- 2 Marks, Diagram-1 Mark each)

Fractals:-

Any object has either flat surfaces which can be drawn with the help of polygons or smooth curved surfaces which can be drawn be using curves. These objects are of rough, jagged, random edges. Some objects has a lot of specification which can be drawn with drawn with the help of fractals, where system draws jagged lines in given endpoints. Fractal uses geometry method, procedures to model objects.

Hilbert Curve: -

This curve can be built by the following successive approximation. The peano curve and space filling curve uses Hilbert's Curve. This curve begins with square. The first approximation is to divide the square into 4 quadrants and draw the curve which connects the centre points of each.

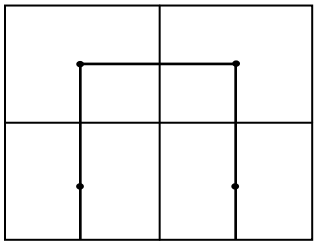


Fig a) First approximation to Hilbert Curve

In second approximation, further divide each of the quadrants and connects the centers of each these finer division before moving to the next major quadrants. By applying the process continuously, at the end the curve never crosses itself. At each subdivision the curve fills smaller quadrants, but never crosses into an area where it already exists. Another thing is that the curve is arbitrarily close to every point in the square. The curve passes through the points on a grid, which becomes twice as fine with each subdivision and there is no limit in subdivision. The curve fills the square. The length of the curve is infinite, with each subdivision, same way there is no limit on subdivision, and same way there is no limit in length.

Koch Curve: -

In Koch curve, begin at a line segment. Divide it into third and replace the centre by the two adjacent sides of an equilateral triangle as shown below.

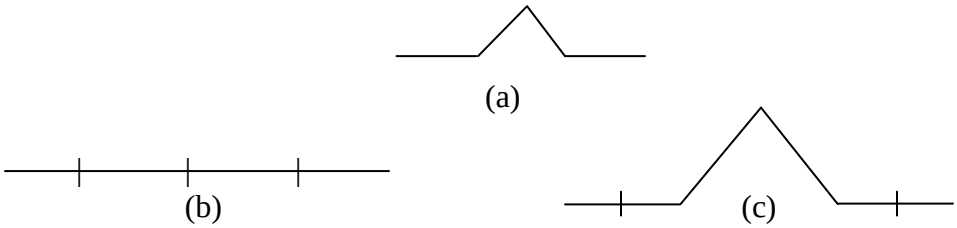


Fig 4.c Replacement of Line Segment for Koch Curve

This will give the curve which starts and ends at same place as the original segment but is built of 4 equal length segments, with each $1/3^{rd}$ of the original length. So the new curve has 4/3 the length of original segments. Repeat same process for each of the 4 segment which will give curve

more wiggles and its length become 16/9 times the original. Suppose repeating the replacements indefinitely, since each repetition increases the length by a factor of 4/3, the length of the curve will be infinite but it is folded in lots of tiny wiggles. Not as a peano curve, it does not fill the area. It does not deviate very much from its original shape.

5. Attempt any FOUR of following: Marks 16

a) Write any four features of GUI.

(1 Mark for any four features)

Graphical user interfaces, such as Microsoft Windows and the one used by the Apple Macintosh, feature the following basic components:

- Pointer: A symbol that appears on the display screen and that you move to select objects and commands. Usually the pointer appears as a small angled arrow. Text-processing applications, however use an I-beam pointer that is shaped like capital I.
- Pointing device: A device such as a mouse or trackball, that enables you to select objects on the display screen.
- Icons: Small Pictures that represent commands, files, or windows. By moving the pointer to the icon and pressing a mouse button, you can execute a command or convert the icon into window. You can also move the icons around the display screen as if they were real objects on your desk.
- Desktop: The area on the display screen where icons are grouped is often referred to as the desktop because the icons are intended to represent real objects on a real desktop.
- Windows: You can divide the screen into different areas. In each window you can run a different program or display a different file. You can move windows around the display screen and change their shape and size at will.
- Menus :Most graphical user interfaces let you execute commands by selecting a choice from a menu.

b) What is graphics standard? Write need of graphics standard.

(2 Marks for standard, 2 Marks for Explanation)

Graphics standards are provided for easy transfer of the graphics from one platform to another as there are many variations in display devices, software packages and even the graphical languages. Due to this reason the standards are developed which provides the portability.

Graphical Standard

1. CORE
2. Programmer's Hierarchical Interactive Graphical Standard (PHIGS)
3. Initial Graphics Exchange Standard (IGES)
4. Computer Graphics Metafile Standard (CGM)
5. Virtual Device Metafile (VDM)
6 Virtual Device Interface (VDI)

Advantages:-

1. The user can use any graphics hardware to upgrade display.
2. Being device independent, graphics generated based on these standards is economical for developer and user.
3. Reuse of code in terms of ready routines is possible, and hence standard software packages can be used for developing other packages.
4. It facilitates transport of application program form one computer to another.
5. It also facilitates programmer portability.

c) What are the principles of good GUI design?

(Each principle-1 Mark)

- The user must able to anticipate visual control's behavior from its visual property.
- The user must be able to anticipate behavior of your program using knowledge gained from other program.
- View every user warning and error dialog that your program generates as an opportunity to improve your interface.
- Provide adequate user feedback.
- Use sound, color, animation, and multimedia clip sparingly.
- Help users to customize and preserve their preferred work environment.
- Design your interface so that users can do their task while being minimally aware of the interface itself.

d) Write 'C' code for translating line in 2D.

(Correct code -4 Marks)

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void main()
{
intgd=DETECT,gm=DETECT,tx,ty;
clrscr();
printf("\nEnter Translation factors (TX and TY)");
scanf("%d%d",&tx,&ty);
initgraph(&gd,&gm,"D:\\TC\\BGI");
line(100,100,200,200);
line(100+tx,100+ty,200+tx,200+ty);
getch();
}
```

e) With equation describe 3D translation.

(Explanation -1 Mark, Equation -3 Marks)

Here object is moved in all three co-ordinate directions. Let's consider to move a point P(x,y,z) to new position(x',y',z').

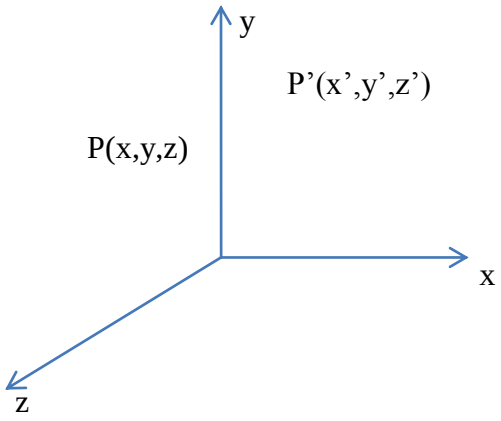
For this equations are

x'=x+tx

y'=y+ty

z'=z+tz

wheretx, ty and tz are translation factors in x, y and z directions respectively.



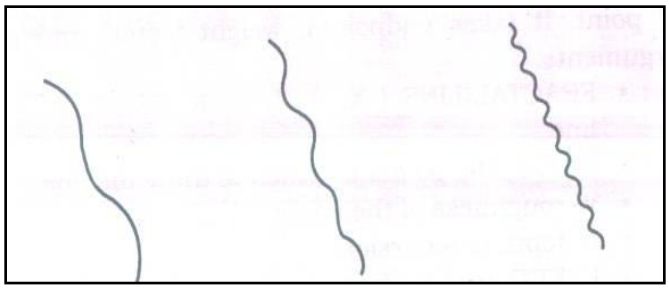
6. Attempt any FOUR of following: Marks 16

a) Describe how fractal surfaces are drawn with fractal lines.

(Fractal line- 2 Marks, Fractal surfaces- 2 Marks)

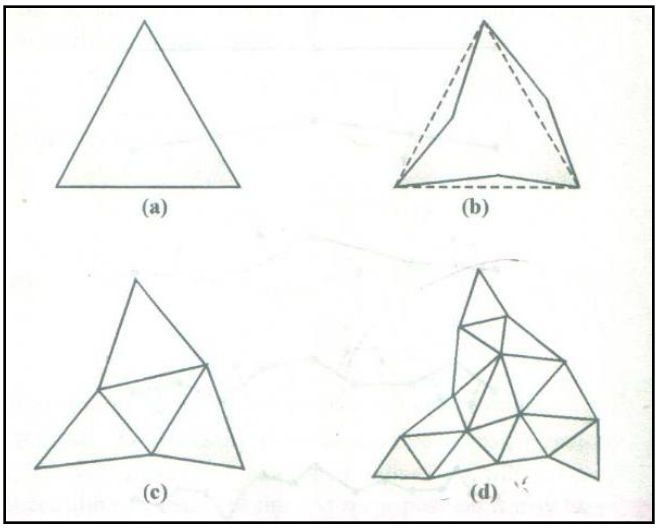
Generally, object which is having smooth surface and regular shape are described by using equation but natural object such as mountain, tree, waves and clouds have irregular shapes and it is very difficult to draw these shapes by using normal equation.

There are may method of modeling these object but one of the important is by using fractal. From certain distance we will see a line as a simple, quite smooth line but as we go near to that lines. It will appear more rough.



We can draw coastline or lightning bolt by fractal lines. But if we want to draw a 3D object like, say mountain, then we have to use fractal surfaces. The concept of fractal lines can be extended to

generate fractal surfaces. There are many by which we can do this. Here we are explaining a method which is based on triangle.



The fractal surfaces can be generated by performing following steps:

- We will use fractal line algorithm to each edge of the triangle.
- Compute its halfway point by the same logic as that of fractal lines.
- Connect these halfway points by line segments. By doing this we are now having four small triangles. Now we can call this procedure recursively to each of small triangle until the triangles becomes too small.

b) Describe in brief Cubic Bezier Curves.

(Diagram -1 Mark, Explanation - 3 Marks)

Bezier curve is an approach of constricting curve. It is determined by defining a polygon.

Bezier curve have properties that makes it highly useable and ease for curve design. These properties are also easy to implement. Hence Bezier curve is widely available in various CAD systems. This curve offers good flexibility and also avoids complex calculation. In this, four points are used to give complete curve. There is no addition of intermediate points like B-spline curve and also smoothly extensions of Bezier curve by picking four more points and construct second curve which can be join to first one.

Subsection of Bezier curve

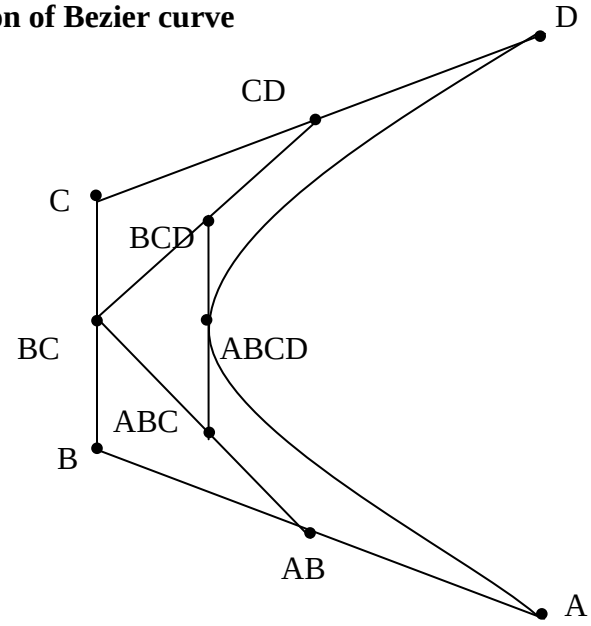


Fig. Subsection of Bezier Curve

Algorithm: -

Step 1: Get four control points A (X_A, Y_A), B (X_B, Y_B), C (X_C, Y_C), D (X_D, Y_D)

Step 2: Divide the curve represented by points A, B, C, and D in two sections

$X_{AB} = (X_A + X_B) / 2;$ $Y_{AB} = (Y_A + Y_B) / 2;$

$X_{BC} = (X_B + X_C) / 2;$ $Y_{BC} = (Y_B + Y_C) / 2;$

$X_{CD} = (X_C + X_D) / 2;$ $Y_{CD} = (Y_C + Y_D) / 2;$

$X_{ABC} = (X_{AB} + X_{BC}) / 2;$ $Y_{ABC} = (Y_{AB} + Y_{BC}) / 2;$

$X_{BCD} = (X_{BC} + X_{CD}) / 2;$ $Y_{BCD} = (Y_{BC} + Y_{CD}) / 2;$

$X_{ABCD} = (X_{ABC} + X_{BCD}) / 2;$ $Y_{ABCD} = (Y_{ABC} + Y_{BCD}) / 2;$

Step 3: Repeat step 2 for section A, AB, ABC, and ABCD and section BCD, CD, D.

Step 4: Repeat step 3 until section so short that they can be replace by straight lines.

Step 5: Replace small sections by straight lines.

Step 6: Stop.

c) Derive transformation matrix for 2-D viewing transformation.

(Explanation-4 Marks)

The workstation transformation is given as

$W=T.S.T^{-1}$

The transformation matrices for individual transformation are as given below:

$$T=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ -X_{wmin} & -y_{wmin} & 1 \end{bmatrix}$$

$$S=\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where $S_x=\dfrac{x_{vmax}-x_{vmin}}{x_{wmax}-x_{wmin}}$

$Sy=\dfrac{y_{vmax}-y_{vmin}}{y_{wmax}-y_{wmin}}$

$$T^{-1}=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ X_{vmin} & y_{vmin} & 1 \end{bmatrix}$$

The overall transformation matrix for W is given as

$W=T.S.T^{-1}$

$$=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xwmin & -ywmin & 1 \end{bmatrix}\begin{bmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xvmin & yvmiv & 1 \end{bmatrix}$$

$$=\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ X_{vmin}-x_{wmin}.S_x & y_{vmin}-y_{wmin}.S_y & 1 \end{bmatrix}$$

d) Explain Cohen - Sutherland line clipping algorithm.

(Explanation-1 Mark and algorithm -3 Marks)

Different cases which are to be consider before clipping line are,

i) Completely inside.

ii) Completely outside.

iii) Partly inside.

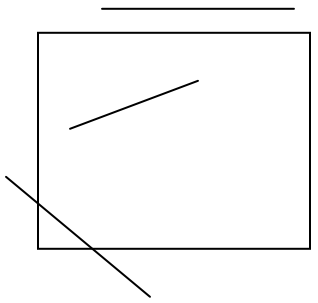


Fig (a) Before clipping

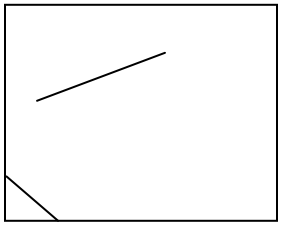


Fig (b) After clipping

As shown in fig three lines are given in which one line is inside window so that is displayed, second line is completely outside so it is discarded after clipping, and third line is clipped as it is partially in so that much part is displayed and remaining part is discarded.

Line	Code for End point		Logical And Operation	Result
P1, P2	0000	0000	0000	Completely Visible
P3,P4	0001	0001	0001	Completely invisible
P5,P6	0001	0000	0000	Partly Visible
P7,P8	0100	0010	0000	Partly Visible
P9,P10	1000	0010	0000	Partly Visible

Table 4.1

Result table for logical And Operation.

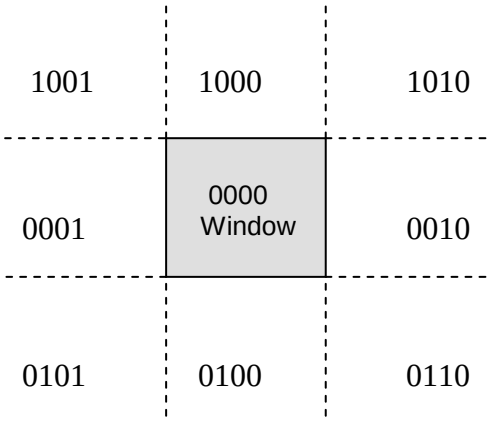


Fig Four bit codes for nine regions

Line clipping in given window.

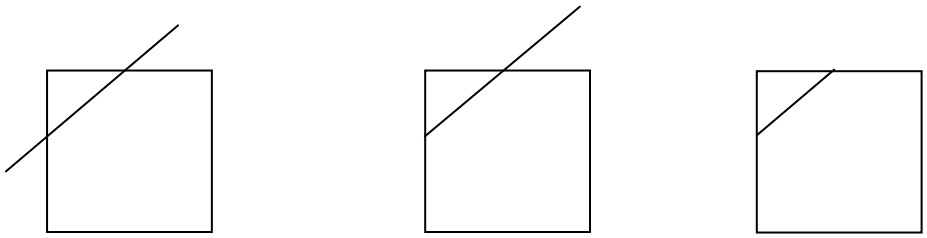


Fig (a) actual line

Fig (b) clipping one part of line

Fig (c) clipping remaining part.

Algorithm

Step 1: Scan end points for the line P1(x1, y1) and P2(x2, y2)

Step 2: Scan corners for the window as (Wx1, Wy1) and (Wx2, Wy2)

Step 3: Assign the region codes for endpoints P1 and P2 by

Bit 1 - if (x < Wx1)

Bit 2 - if (x < Wx2)

Bit 3 - if (x < Wy2)

Bit 4 - if (x < Wy1)

Step 4: Check for visibility of line P1, P2

- If region codes for both end points are zero then the line is visible, draw it and jump to step 9.

- If region codes for end points are not zero and the logical and operation of them is also not zero then the line is invisible, reject it and jump to step 9.
- If region codes for end points does not satisfies the condition in 4(i) and 4(ii) then line is partly visible.

Step 5: Determine the intersecting edge of the clipping window by inspecting the region codes for endpoints.

- If region codes for both the end points are non-zero, find intersection points P1 and P2 with boundary edges of clipping window with respect to point P1 and P2.
- If region code for any one end point is non zero then find intersection point P1 or P2 with the boundary edge of the clipping window with respect to it.

Step 6: Divide the line segments by considering intersection points.

Step 7: Reject the line segment if any of the end point of it appear outside the window.

Step 8: Draw the remaining line.

Step 9: Exit

e) Write properties of Bezier Curve.

(Any 4 properties are expected 1 Mark for one property)

Properties of Bezier Curve

1. It always passes through first and last control point.
2. Tangent vectors at ends of the curve have same direction as first and last polygon spans respectively.
3. It follows the shape of defining polygon.
4. It is contained within the convex hull of defining polygon.
5. Degree of polynomial defining the curve segments is equal to the total number of control points minus 1.
6. Order of polynomial defining the curve segments is equal to the total number of control points
7. Basic functions of Bezier curve are real.

f) Write DDA arc generation algorithm.

(Correct Algorithm -4 Marks)

Algorithm:-

Step 1: Start

Step 2: If $|X - X_0| + |Y - Y_0| < ROUNDOFF$

Then RETURN

Step 3: A – DRAWN <– 0

Step 4: Find suitable angle increment

DA ⟵ MIN(0.1, 1(3.2*($|X - X_0| + |Y - Y_0|$)));

Step 5: Set the first point of arc

xARC ⟵ x; yARC ⟵ y;

Step 6: Generate the arc until desired angle is covered. WHILE A – DRAWN < A

Find new Point

xARC ⟵ xARC + (Y_0 – YARC) * DA

yARC ⟵ yARC + (XARC – Y_0) * DA

A – DRAWN ⟵ A – DRAWN + DA

Set corresponding pixel

FRAME [INT (XARC), INT(YARC)] ⟵ INTENSITY;

Step 7: Stop