

Lab Manual

Computer Graphics

T.E. Computer

(Sem VI)

Index

<i>Sr. No.</i>	<i>Title of Programming Assignments</i>	<i>Page No.</i>
1.	Line Drawing Algorithms	3
2.	Circle Drawing Algorithms	6
3.	Ellipse Drawing Algorithms	8
4.	Polygon Filling Algorithms	10
5.	Basic Transformations	13
6.	Composite Transformations	17
7.	Line Clipping Algorithms	21
8.	Polygon Clipping Algorithms	26
9.	Curve Generations	28
10.	Animation Program	31

Note :

All assignments require the knowledge of Graphics functions in C/C++.

Proper header files have to be used for the initialization of graphics functions

Lab Assignment 1

Title	Line Drawing Algorithms
Objective	1.To study and Implement DDA Algorithm 2.To study and Implement Bresenham 's Algorithm

- References
1. Donald Hearn and M. Pauline Baker, "Computer Graphics with C version", Second Edition Pearson Education
 2. Newman and Sproull, "Principles of Interactive Computer Graphics", Second Edition, McGraw Hill
 3. Rogers and Adams, "Mathematical Elements For Computer Graphics", TMH
 4. Xiang and Plastok, "Schaum's Outlines Computer Graphics", Second Edition, TMH
 5. Harrington, "Computer Graphics", McGraw Hill
 6. Rogers, "Procedural Elements for Computer Graphics", TMH

Pre-requisite Knowledge of

- Point Plotting Methods
- Graphics Initializations
- DDA Algorithm
- Bresenham's Algorithm.

Algorithm **DDA algorithm:**

Input to the function is two endpoints (x1,y1) and (x2,y2)

1. $\text{length} \leftarrow \text{abs}(x_2 - x_1)$;
2. if $(\text{abs}(y_2 - y_1) > \text{length})$ then $\text{length} \leftarrow \text{abs}(y_2 - y_1)$;
3. $\text{xincrement} \leftarrow (x_2 - x_1) / \text{length}$;
4. $\text{yincrement} \leftarrow (y_2 - y_1) / \text{length}$;
5. $x \leftarrow x + 0.5$; $y \leftarrow Y + 0.5$;
6. for $i \leftarrow 1$ to length follow steps 7 to 9
7. plot $(\text{trunc}(x), \text{trunc}(y))$;
8. $x \leftarrow x + \text{xincrement}$;
9. $y \leftarrow y + \text{yincrement}$;
10. stop.

Bresenham's Line Drawing Algorithm:

1. Input the two line endpoints and store the left endpoint in (x_0, y_0)
2. Load (x_0, y_0) into the frame buffer; that is, plot the first point.

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

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

4. At each x_k , the next point on 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

$$p_{k+1} = p_k + 2\Delta y - 2\Delta x$$

5. Repeat step 4 Δx times.

Sample
Output

Enter the option that you want

1. DDA Algorithm

2. Bresenham Algorithm

3. Exit

1

Enter a Initial Point :- 100 200

Enter the Final Point:- 200 300



Enter the option that you want

- 1.DDA Algorithm
- 2.Bresenham Algorithm
- 3.Exit

2

Enter a Initial Point :- 100 200

Enter the Final Point:- 200 300



Enter the option that you want

- 1.DDA Algorithm
- 2.Bresenham Algorithm
- 3.Exit

Post Lab Assignment

1. What are the advantages of Bresenham's algorithm over DDA algorithm.
2. How can the Bresenham's algorithm be modified to accommodate all types of lines?
3. Modify the algorithms by implementing antialiasing procedures also.
4. Modify the BRESENHAM algorithm so that it will produce a dashed-line

pattern. Dash length should be independent of slope.

Lab Assignment 2

Title	Circle Drawing Algorithm
Objective	To study and Implement Midpoint circle algorithm given the points of the centre and the radius.
References	<ol style="list-style-type: none">1. Donald Hearn and M.Pauline Baker, "Computer Graphics with C version ",Second Edition Pearson Education2. Newman and Sproll, "Principles of Interactive Computer Graphics ", Second Edition ,McGraw Hill3. Rogers and Adams , "Mathematical Elements For Computer Graphics",TMH4. Xiang and Plastok , "Schaum's Outlines Computer Graphics ",Second Edition .TMH5. Harrington, "Computer Graphics " ,McGraw Hill6. Rogers , "Procedural Elements for Computer Graphics" ,TMH
Pre-requisite	Knowledge of <ul style="list-style-type: none">• Midpoint circle algorithm
Algorithm	<p>Midpoint circle algorithm</p> <ol style="list-style-type: none">1. Input radius r and circle center (x_0, y_0), and obtain the first point on the circumference of a circle centered on the origin as $(x_0, y_0) = (0, r)$2. Calculate the initial value of the decision parameter $asp_0 = 5 / 4 - r$3. At each x_k position , starting at $k=0$, perform the following test: If $p_k < 0$, the next point along the circle centered on $(0,0)$ is $(x_k + 1, y_k)$ and $p_{k+1} = p_k + 2x_{k+1} + 1$ Otherwise ,the next point along the circle is $(x_k + 1, y_k - 1)$ and $p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1}$ Where $2x_{k+1} = 2x_k + 2$ and $2y_{k+1} = 2y_k - 2$.4. Determine symmetry points in the other seven octants5. Move each calculated pixel position (x, y) onto the circular path centered on (x_0, y_0) and plot the coordinate values: $x = x + x_c \quad y = y + y_c$6. Repeat step 3 through 5 until $x \geq y$

Sample
Output

Enter the coordinates of the centre :-

x-coordiante = 350

y-coordinate = 250

Enter the radius :-

50



Post Lab
Assignment

1. Outline a method to for antialiasing a circle boundary. How would this method be modified to antialias elliptical boundaries.
2. Revise the midpoint circle algorithm to display a circle so that the geometric standards are maintained.

Lab Assignment 3

Title Ellipse Drawing Algorithms

Objective To study and Implement Midpoint Ellipse Algorithm

References

1. Donald Hearn and M. Pauline Baker, "Computer Graphics with C version", Second Edition Pearson Education
2. Newman and Sproll, "Principles of Interactive Computer Graphics", Second Edition, McGraw Hill
3. Rogers and Adams, "Mathematical Elements For Computer Graphics", TMH
4. Xiang and Plastok, "Schaum's Outlines Computer Graphics", Second Edition, TMH
5. Harrington, "Computer Graphics", McGraw Hill
6. Rogers, "Procedural Elements for Computer Graphics", TMH

Pre-requisite Knowledge of

- Midpoint Ellipse Algorithm

Algorithm

Midpoint Ellipse algorithm

1. Input r_x, r_y and ellipse center (x_c, y_c) , and obtain the first point on the ellipse centered on the origin as $(x_0, y_0) = (0, r_y)$
2. Calculate the initial value of the decision parameter in region 1 as $p_{1_0} = r_y^2 - r_x^2 r_y + 1 / 4 r_x^2$
3. At each x_k position in region 1, starting at $k=0$, perform the following test: If $p_{1_k} < 0$, the next point along the ellipse centered on $(0,0)$ is $(x_k + 1, y_k)$ and $p_{1_{k+1}} = p_{1_k} + 2 r_y^2 x_{k+1} + r_y^2$. Otherwise, the next point along the circle is $(x_k + 1, y_k - 1)$ and $p_{1_{k+1}} = p_{1_k} + 2 r_y^2 x_{k+1} - 2 r_x^2 y_{k+1} + r_y^2$. With $2 r_y^2 x_{k+1} = 2 r_y^2 x_k + 2 r_y^2$, $2 r_x^2 y_{k+1} = 2 r_x^2 y_k - 2 r_x^2$
4. Calculate the initial value of the decision parameter in region 2 using the last point (x_0, y_0) calculated in the region 1 as $p_{2_0} p_{2_{k+1}} = r_y^2 (x_0 + 1 / 2)^2 + r_x^2 (y_0 - 1)^2 - r_x^2 r_y^2$
5. At each x_k position in region 2, starting at $k=0$, perform the following test: If $p_{2_k} > 0$, the next point along the ellipse centered on $(0,0)$ is $(x_k, y_k - 1)$ and $p_{2_{k+1}} = p_{2_k} + 2 r_y^2 y_{k+1} + r_x^2$. Otherwise, the next point along the circle is $(x_k + 1, y_k - 1)$ and $p_{2_{k+1}} = p_{2_k} + 2 r_y^2 x_{k+1} - 2 r_x^2 y_{k+1} + r_x^2$ using the same incremental calculations for x and y as in region 1
6. Determine symmetry points in the other three quadrants
7. Move each calculated pixel position (x, y) onto the elliptical path centered on (x_c, y_c) and plot the coordinate values:
$$x = x + x_c \quad y = y + y_c$$
8. Repeat steps for region 1 until $2 r_y^2 x \geq 2 r_x^2 y$

Sample Output

Enter the option that you want

1. Midpoint Ellipse Algorithm

2. Exit

1

Enter the coordinates of the center :- 100 200

Enter the Minor axis :- 50

Enter the Major axis :- 100



Enter the option that you want

1.Midpoint Ellipse Algorithm

2.Exit

2

Post Lab
Assignment

1. Write a procedure to scan the interior of a specified ellipse into a solid color.
2. Outline a method for antialiasing a ellipse boundary.

Lab Assignment 4

Title

Polygon Filling Algorithms

Objective

1. To study and Implement Polygon Filling Algorithms

References

1. Donald Hearn and M.Pauline Baker, "Computer Graphics with C version ",Second Edition Pearson Education
2. Newman and Sproll, "Principles of Interactive Computer Graphics ", Second Edition ,McGraw Hill

3. Rogers and Adams , "Mathematical Elements For Computer Graphics",TMH
4. Xiang and Plastok , "Schaum's Outlines Computer Graphics ",Second Edition .TMH
5. Harrington, "Computer Graphics " ,McGraw Hill
6. Rogers , "Procedural Elements for Computer Graphics" ,TMH

Pre-requisite Knowledge of

- Polygon Filling Algorithms
 - Boundary Fill Algorithm(4 connected and 8 connected)
 - Scan line polygon filling Algorithm.

Algorithm

- **Boundary Fill Algorithm**

(x,y) are the interior points, boundary is the boundary color and fill_color is the color to be filled. Following is a recursive method for boundary fill.

- 1. present_color = getcolor() // a function which returns the current color of (x,y)
 2. if present_color <> boundary and if present_color <> fill_color then repeat steps 3-7
 3. set_pixel (x,y, fill_color)
 4. call the algorithm recursively for points (x + 1, y)
 5. call the algorithm recursively for points (x - 1,y)
 6. call the algorithm recursively for points (x,y + 1)
 7. call the algorithm recursively for points (x,y - 1)
 8. stop

- **Scan line polygon filling algorithm.**

1. Input n, number of vertices of polygon
2. input x and y coordinated of all vertices i array x[n] and y[n]
3. find y_{\min} and y_{\max}
4. Store the initial x values(x_1) y values y_1 and y_2 for two endpoints and x increment Δx from scan line to scan line for each edge in the array edges [n] [4] while doing this check that $y_1 > y_2$, if not interchange y_1 and y_2 and corresponding x_1 and x_2 so that for each edge , y_1 represents its maximum y coordinate and y_2 represents its minimum y coordinate
5. Sort the rows of array , edges [n] [4] in descending order of y_1 ,descending order of y_2

and ascending order of x_2

6. Set $y = y_{\max}$
7. Find the active edges and update active edge list:

if $(y > y_2 \text{ and } y \leq y_1)$ then edge is active
Otherwise edge is not active

8. Compute the x intersects for all active edges for current y values [initially x-intersect is and x intersects for successive y values can be given as $x_{i+1} = x_i + \Delta x$

Where $\Delta x = -1/m$ and $m = (y_2 - y_1) / (x_2 - x_1)$

i.e slope of a line segment

9. Vertex: If x intersects is vertex i.e. X-intersect = x_1 and $y = y_1$ then apply vertex test to check whether to consider one intersect or two intersects. Store all x-intersect in the x-intersect [] array
10. Store x-intersect [] array in the ascending order
11. Extract pairs of intersects from the sorted x-intersect [] array
12. Pass pair of x values to line drawing routine to draw corresponding line segments
13. Set $y = y - 1$
14. Repeat steps 7 through 13 until $y \geq y_{\max}$
15. Stop.

Sample
Output

Enter your choice

1. Boundary Fill
2. Scan line polygon filling algorithm.
3. Exit

Enter Choice.....1



Enter your choice

1. Boundary Fill
2. Scan line polygon filling algorithm.
3. Exit

Enter Choice.....2



Enter your choice

1. Boundary Fill
2. Scan line polygon filling algorithm.
3. Exit

Enter Choice.....3

Post Lab Assignment

1. Modify the 4-connected boundary fill algorithm to avoid excess stacking.
2. Develop the flood fill algorithm to fill any interior of any specified area.
3. What is pattern filling? Where it is used? What are the constraints involved?

Lab Assignment 5

Title	Basic Transformations
Objective	To Implement set of Basic Transformations on Polygon i.e Translation , Rotation and Scaling
References	<ol style="list-style-type: none"> 1. Donald Hearn and M.Pauline Baker, "Computer Graphics with C version ",Second Edition Pearson Education 2. Newman and Sproll, "Principles of Interactive Computer Graphics ", Second Edition ,McGraw Hill 3. Rogers and Adams , "Mathematical Elements For Computer Graphics",TMH 4. Xiang and Plastok , "Schaum's Outlines Computer Graphics ",Second Edition .TMH 5. Harrington, "Computer Graphics " ,McGraw Hill 6. Rogers , "Procedural Elements for Computer Graphics" ,TMH
Pre-requisite	<p>Knowledge of</p> <ul style="list-style-type: none"> • • Basic Transformations on Polygon <ul style="list-style-type: none"> ○ Translation ○ Rotation ○ Scaling

Description Transformations allows us to uniformly alter the entire picture. The geometric transformations considered here – translation, scaling and rotation are expressed in terms of matrix multiplication. Homogeneous coordiantes are considered to uniformly treat the translations.

Scaling Transformations:

A 2D point can be scaled by multiplication of the coordiante values (x,y) by scaling factors S_x and S_y to produce the transformed coordinates (x',y').

Matrix format:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Translation Transformations:

A 2D point can be translated by adding the coordiante values (x,y) by

Translation distances t_x and t_y to produce the transformed coordinates (x',y') .

Matrix format:

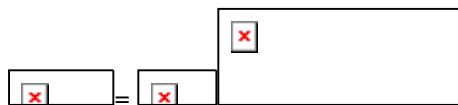


The diagram illustrates a matrix multiplication. On the left, a 2x2 matrix is shown with a red 'x' in the bottom-left cell. This is followed by an equals sign and a 2x1 vector with a red 'x' in the top cell. To the right of the equals sign is another 2x1 vector with a red 'x' in the top cell.

Rotation Transformations:

A 2D point can be rotated by repositioning it along a circular path in the xy plane. We specify the rotation angle θ and the position of the rotation point about which the object is to be rotated. Multiplication of the coordinate values (x,y) by rotation matrix produce the transformed coordinates (x',y') .

Matrix format:



The diagram illustrates a matrix multiplication. On the left, a 2x2 matrix is shown with a red 'x' in the bottom-left cell. This is followed by an equals sign and a 2x1 vector with a red 'x' in the top cell. To the right of the equals sign is a larger 2x1 vector with a red 'x' in the top cell.

Sample
Output

Enter your choice

1. Translation

2. Rotation

3. Scaling

4.Exit

Enter the no. of edges :-4

Enter the co-ordinates of vertex 1 :- 30 30

Enter the co-ordinates of vertex 2 :- 30 90

Enter the co-ordinates of vertex 3 :- 90 90

Enter the co-ordinates of vertex 4 :- 90 30

Enter the Translation factor for x and y :-20 20





Post Lab
Assignment

1. What is the significance of homogeneous co-ordinates? Give the homogeneous co-ordinates for the basic transformations.
2. Why are matrices used for implementing transformations.

Lab Assignment 6

Title Composite Transformations

Objective 1. To study and Implement set of Composite Transformations on Polygon i.e Reflection, Shear (x & Y), rotation about an arbitrary point.

- References
1. Donald Hearn and M.Pauline Baker, "Computer Graphics with C version Second Edition Pearson Education
 2. Newman and Sproll, "Principles of Interactive Computer Graphics ", Second Edition ,McGraw Hill
 3. Rogers and Adams , "Mathematical Elements For Computer Graphics",TMH
 4. Xiang and Plastok , "Schaum's Outlines Computer Graphics ",Second Edition .TMH
 5. Harrington, "Computer Graphics " ,McGraw Hill
 6. Rogers , "Procedural Elements for Computer Graphics" ,TMH

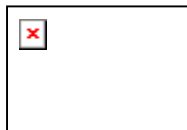
Pre-requisite Knowledge of

- Composite Transformations on Polygon
 - Reflection
 - Shear
 - Rotation about an arbitrary point.

Description **Reflection**

Reflection is a transformation that produces a mirror image of an object.

Transformation matrix for reflection about the line $y=0$, is



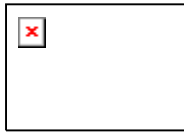
Transformation matrix for reflection about the line $x=0$, is



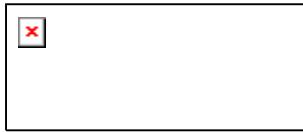
Shearing:

shearing is a transformation that distorts the shape of an object.

An X-direction shear relative to the x axis is produced by the transformation matrix



A y-direction shear relative to other referencelines is produced by the transformation matrix



Rotation about an arbitrary point

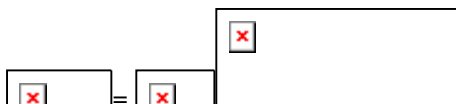
This is done by three transformation steps: translation of the arbitrary point (x_c, y_c) to the origin, rotate about the origin, and then translate the center of rotation back to where it belongs.

To transform a point, we would multiply all the transformation matrices together to form an overall transformation matrix.

1. The translation which moves (x_c, y_c) to the origin:



2. the rotation is



3. and the translation to move the center point back is



Output

1.Reflection

2.Shearing

3.Rotation about an arbitrary point.

Enter your choice.....1

Enter the no. of edges :-4

Enter the x and y co-ordinates :- 30 30

Enter the x and y co-ordinates :- 30 90

Enter the x and y co-ordinates :- 90 90

Enter the x and y co-ordinates :- 90 30



1.Reflection along X-axis

2.Reflection along Y-axis

3.Exit

Enter your choice.....1



1.Reflection along X-axis

2.Reflection along Y-axis

3.Exit



Enter your choice.....2

1.Reflection

2.Shearing

3.Exit

Enter your choice.....2

Shear Factor --> X and Y directions : 2 1



Post Lab Assignment

1. Show that two successive reflections about any line passing through the coordinate origin is equivalent to single rotation about the origin.
2. Determine the sequence of basic transformations that are equivalent to the x-direction and y-direction shearing matrix.
3. Show that transformation matrix for a reflection about the line $y=x$, is equivalent to a reflection relative to the x axis followed by a counterclockwise rotation of 90 degrees.

Lab Assignment 7

Title Line Clipping Algorithm

Objective 1.To study and Implement Line Clipping Algorithm using Cohen Sutherland
2.To study and Implement Line Clipping Algorithm using Liang Barsky

References 1. Donald Hearn and M.Pauline Baker, "Computer Graphics with C version", Second Edition Pearson Education

2. Newman and Sproll, "Principles of Interactive Computer Graphics ", Second Edition ,McGraw Hill
3. Rogers and Adams , "Mathematical Elements For Computer Graphics",TMH
4. Xiang and Plastok , "Schaum's Outlines Computer Graphics ",Second Edition .TMH
5. Harrington, "Computer Graphics " ,McGraw Hill
6. Rogers , "Procedural Elements for Computer Graphics" ,TMH

Pre-requisite Knowledge of

- Line Clipping Algorithms using
 - Cohen Sutherland
 - Liang Barsky

Algorithm **Cohen sutherland Line Clipping Algorithm**

1. Input two endpoints of the line say $p_1 (x_1 , y_1)$ and $p_2 (x_2 , y_2)$
2. Input two corners (Let-top and right -bottom) of the window , say (wx_1 , wy_1 and wx_2 , wy_2)
3. Assign the region codes for two endpoints p_1 and p_2 using following steps :Initialize code with bits 0000Set Bit1 – if ($x < wx_1$)Set Bit2 – if ($x < wx_2$)Set Bit3 – if ($y < wy_2$) Set Bit4 – if ($y < wy_1$)
4. Check for visibility of line
 - a. If region codes for both endpoints p_1 and p_2 are zero then the line is completely visible. Hence draw the line and go to step 9
 - b. If region codes for both endpoints are not zero and the logical ANDing of them is also nonzero then the line is completely invisible. So reject the line and go to 9
 - c. If region codes for two endpoints do not satisfy the condition in (4a and 4b) the line is partially visible.
5. Determine the intersecting edge of the clipping window by inspecting the region codes of two endpoints
 - a. If region codes for both endpoints are non- zero,find intersecting point p'_1 and p'_2 with boundary edges of clipping window with respect to point p_1 and point p_2 respectively
 - b. If region codes for any one endpoints are non-zero,find intersecting point p'_1 or p'_2 with boundary edges of clipping window with respect to it.
6. Divide the Line segments considering intersection points
7. Reject the line segments if any one endpoint of it appears outside the clipping window
8. Draw the remaining line segments
9. Stop

Liang Barsky Line Clipping Algorithm

1. Input two endpoints of the line say $p_1(x_1, y_1)$ and $p_2(x_2, y_2)$
2. Input two corners (Left-top and right-bottom) of the window, say $(x_{wmin}, y_{wmax}, x_{wmax}, y_{wmin})$
3. Calculate the values of the parameter p_i and q_j for $i = 1, 2, 3, 4$ such that $p_1 = -\Delta x$ $q_1 = x_1 - x_{wmin}$

$$p_2 = \Delta x \quad q_2 = x_{wmax} - x_1$$

$$q_1 = -\Delta y \quad q_3 = y_1 - y_{wmin}$$

$$q_2 = \Delta y \quad q_4 = y_{wmax} - y_1$$

4. If $p_i = 0$ then the line is parallel to i^{th} boundary. Now, if $q_i < 0$ then the line is completely outside the boundary, hence discard the line segment and go to stop. Otherwise, check whether the line is horizontal or vertical and accordingly check the line endpoint with corresponding boundaries. If line endpoint within the bounded area then use them to draw line otherwise use boundary coordinates to draw line. Go to stop.
5. Initialise values for t_1 and t_2 as $t_1 = 0$ and $t_2 = 1$
6. Calculate values for q_i / p_i for $i = 1, 2, 3, 4$
7. Select values of q_i / p_i where $p_i < 0$ and assign maximum out of them as t_1
8. Select values of q_i / p_i where $p_i < 0$ and assign minimum out of them as t_2
9. If ($t_1 < t_2$) Calculate the endpoints of the clipped lines as follows:- $x_1 = x_1 + t_1 \Delta x$, $x_2 = x_1 + t_2 \Delta x$, $y_1 = y_1 + t_1 \Delta y$, $y_2 = y_1 + t_2 \Delta y$. Draw line (x_1, x_2, y_1, y_2)
10. Stop

Sample
Output

Menu

1. Cohen Sutherland Line Clipping Algorithm
2. Liang Barsky Line Clipping Algorithm
3. Exit

Enter your choice.....1

Enter Minimum window co-ordinates :- 200 250

Enter Maximum window co-ordinates :- 300 350

Enter co-ordinates of first point of line :- 180 250

Enter co-ordinates of second point of line :- 200 300





Menu

- 1.Cohen sutherland Line Clipping Algorithm
- 2.Liang Barsky Line Clipping Algorithm
- 3.Exit

Enter your choice.....2

Enter Minimum window co-ordinates :- 200 250

Enter Maximum window co-ordinates :- 300 350

Enter co-ordinates of first point of line :- 180 250

Enter co-ordinates of second point of line :- 200 300





Menu

- 1.Cohen sutherland Line Clipping Algorithm
- 2.Liang Barsky Line Clipping Algorithm
- 3.Exit

Enter your choice.....3

Post Lab
Assignment

1. Modify the Liang-Barsky line clipping algorithm to polygon clipping.
2. Write a routine to clip an ellipse against a rectangular window.
3. Write a routine to implement exterior clipping on any part of a defined picture using any specified window.

Lab Assignment 8

Title Polygon Clipping Algorithm

Objective 1. To study and Implement Polygon Clipping Algorithm using sutherland Hodgman Algorithm

References 1. Donald Hearn and M.Pauline Baker, "Computer Graphics with C version", Second Edition Pearson Education
2. Newman and Sproll, "Principles of Interactive Computer Graphics", Second Edition, McGraw Hill
3. Rogers and Adams, "Mathematical Elements For Computer Graphics", TMH

4. Xiang and Plastok , "Schaum's Outlines Computer Graphics " ,Second Edition .TMH
5. Harrington, "Computer Graphics " ,McGraw Hill
6. Rogers , "Procedural Elements for Computer Graphics" ,TMH

Pre-requisite Knowledge of

- Polygon Clipping Algorithm
 - Sutherland Hodgman Algorithm

Algorithm **Sutherland Hodgman Algorithm**

1. Input Coordinates of all vertices of the polygon
2. Input coordiantes of the clipping window
3. Consider the left edge of the window
4. Compare the vertices of each edge of the polygon , individually with the clipping plane
5. Save the resulting intersections and vetrices in the new list of vertices according to four possible relationships between the edge and the clipping boundary discussed earlier
6. Repeat the steps 4 and 5 for remaining edges of the clipping window.Each time the resultant list of vertices is successively passed to process the next edge of the clipping window
7. Stop

Sample Output

Menu

1.Sutherland Hodgman Polygon Clipping Algorithm

2.Exit

Enter your choice.....1

Enter Minimum window co-ordinates :- 200 250

Enter Maximum window co-ordinates :- 300 350

Enter co-ordinates of first point of line :- 180 250

Enter co-ordinates of second point of line :- 200 300



Menu

1.Sutherland Hodgman Polygon Clipping Algorithm

2.Exit

Enter your choice.....2

Post Lab Assignment

1. The Sutherland-Hodgman algorithm can be used to clip lines against a non rectangular boundary. What uses might this have? What modifications to the algorithm would be necessary? What restrictions would apply to the shape of clipping region?
2. Explain why Sutherland-Hodgman algorithm works only for convex clipping regions?

Lab Assignment 9

Title	Curves Generation
Objective	1.To study and Implement Curves Generation using Bezeir Curves 2.To study and Implement Curves Generation using B-Splines
References	<ol style="list-style-type: none">1. Donald Hearn and M.Pauline Baker, "Computer Graphics with C version ",Second Edition Pearson Education2. Newman and Sproll, "Principles of Interactive Computer Graphics ", Second Edition ,McGraw Hill3. Rogers and Adams , "Mathematical Elements For Computer Graphics",TMH4. Xiang and Plastok , "Schaum's Outlines Computer Graphics ",Second Edition .TMH5. Harrington, "Computer Graphics " ,McGraw Hill6. Rogers , "Procedural Elements for Computer Graphics" ,TMH
Pre-requisite	Knowledge of <ul style="list-style-type: none">• Curves Generation Using• Bezeir Curves<ul style="list-style-type: none">○ B-Splines
Algorithm	Bezeir Curves <ol style="list-style-type: none">1. Get Four control points say $A(x_A, y_A)$, $B(x_B, y_B)$, $C(x_C, y_C)$, $D(x_D, y_D)$2. Divide the curves represented by point A,B,C and D in two sections$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$3. Repeat the step 2 for section A AB, ABC, and ABCD and section ABCD , BCD, CD and D

4. Repeat step 3 until we have sections so short that they can be replaced by straight lines
5. Replace small sections by straight lines
6. Stop

B-Splines

1. The B-spline basis functions are defined recursively as follows:

$$N_{i,1}(u) = 1 \text{ if } t_i \leq u < t_{i+1}$$

= 0 otherwise.

2. The knot values are chosen with the following rule:

$$t_i = 0 \text{ if } i < k$$

$$= i - k + 1 \text{ if } k \leq i \leq n$$

$$= n - k + 2 \text{ if } i > n$$

Sample
Output

Menu

1. Curves Generation using Bezeir Curves

2. Curves Generation using B-Splines

3. Exit

Enter your choice.....1

Enter the no. of control points : 4

Enter the control point1 :- 20 50

Enter the control point2 :- 30 10

Enter the control point3 :- 40 50

Enter the control point4 :- 50 10



p1 p3

Menu

1. Curves Generation using Bezier Curves
2. Curves Generation using B-Splines
3. Exit

Enter your choice.....2



p1 p3

Menu

1. Curves Generation using Bezier Curves

2. Curves Generation using B-Splines

3. Exit

Enter your choice.....3

Post Lab
Assignment

1. List the properties of Bezier curves.
2. List the properties of B-Splines.
3. Why cubic Bezier curves are chosen?
4. What do you understand by cubic B-splines? Discuss with suitable mathematical models.

Lab Assignment 10

Title Animation Program.

Objective To Implement a program with animation of objects (segments) obtained by scan conversion.

- References
1. Donald Hearn and M. Pauline Baker, "Computer Graphics with C version", Second Edition Pearson Education
 2. Newman and Sproll, "Principles of Interactive Computer Graphics", Second Edition, McGraw Hill
 3. Rogers and Adams, "Mathematical Elements For Computer Graphics", TMH
 4. Xiang and Plastok, "Schaum's Outlines Computer Graphics", Second Edition, TMH
 5. Harrington, "Computer Graphics", McGraw Hill
 6. Rogers, "Procedural Elements for Computer Graphics", TMH

Pre-requisite Knowledge of

- All Raster Scan algorithms
 - Segmentation and its properties

Description

Animation:

Sequences of pictures to educate or explain may require images of 3D objects. Although animation uses graphics as much for art as for realism, it depends heavily on motions to substitute for realism of an individual image. Animation is done by photographing a sequence of drawings, each slightly different from the previous. This can be achieved by segmentation.

Conclusion

For eg., to show a person moving his arm, a series of drawings is photographed, each drawing showing the arm at a different position. When the images are displayed one after another from the frame buffer, we perceive the arm as moving through the sequence.

Post Lab
Assignment

1. How renaming operations of segments is useful for animation?
2. What do you mean by posting and unposting of segments?
3. Explain the use of display and segmentation in graphics.