**Cohen Sutherland**

|  |  |
| --- | --- |
|  | **DEPARTMENT OF COMPUTER ENGINEERING** |

**Experiment No.**

**8**



Semest

er



S.E. Semester III



–



Computer Engineering



Subj



Computer Graphics



Subject Professor In



-



char



Prof. Sachin Deshpande



Assisting

Teachers



Prof. Sachin Deshpande



|  |  |
| --- | --- |
| Student Name | Trisha Shah |
| Roll Number | 20102A0004 |

**EXPERIMENT DETAILS:**

|  |  |
| --- | --- |
| Title | Write a program to implement Cohen Sutherland line clipping |
| Tools/Language Used | Dev C++ |

**Cohen Sutherland Line Clipping Algorithm:**

In the algorithm, first of all, it is detected whether line lies inside the screen or it is outside the screen. All lines come under any one of the following categories:

Visible

Not Visible

Clipping Case

1. **Visible**: If a line lies within the window, i.e., both endpoints of the line lies within the window. A line is visible and will be displayed as it is.

2. **Not Visible**: If a line lies outside the window it will be invisible and rejected. Such lines will not display. If any one of the following inequalities is satisfied, then the line is considered invisible. Let A (x1,y2) and B (x2,y2) are endpoints of line.

xmin,xmax are coordinates of the window.

ymin,ymax are also coordinates of the window.

x1>xmax

x2>xmax

y1>ymax

y2>ymax

x1<xmin

x2<xmin

y1<ymin

y2<ymin

3**. Clipping Case:** If the line is neither visible case nor invisible case. It is considered to be clipped case. First of all, the category of a line is found based on nine regions given below. All nine regions are assigned codes. Each code is of 4 bits. If both endpoints of the line have end bits zero, then the line is considered to be visible.

Line Clipping

The center area is having the code, 0000, i.e., region 5 is considered a rectangle window.

Following figure show lines of various types

Line Clipping

Line AB is the visible case

Line OP is an invisible case

Line PQ is an invisible line

Line IJ are clipping candidates

Line MN are clipping candidate

Line CD are clipping candidate

**Algorithm**

1. Accept end point co-ordinates of line AB i.e. A(x1,y1) and B(x2,y2)

and window co-ordinate (Xwmin, Ywmin) (Xwmax , Ywmax)

2. Assign 4 bit Region Code to both end pts of line ABIf X<Xwmin then B1 = 1 else 0

If X>Xwmax then B2 = 1 else 0

If Y<Ywmin then B3 = 1 else 0

If Y>Ywmax then B4 = 1 else 0

3. Check Status Line AB

a. Completely IN

If the region code for both end points are 0000 then line AB is

completely IN.

Display line AB

Stop.

b. Completely OUT

If the logical AND operation between 2 end point codes is NOT

0000 then line AB is completely OUT

Discard line AB

Stop

c. Clipping Candidate

If case a and b fails, then line AB is clipping candidate

Go to step 4

4. Determine Intersection boundary. Consider region code of outside

point

If B1 = 1 line intersect with left boundary

B2 = 1 line intersect with right boundary

B3 = 1 line intersect with bottom boundary

B4 = 1 line intersect with top boundary

5. Determine the intersection point

a. Left/Right Boundary

X’ = Xwmin(L)

OR

= Xwmax(R)

(X’-X1)/(X2-X1) = (Y’-Y1)/(Y2-Y1)

Y’ = Y1 + m(X’-X1)

I(X’,Y’)

b. Bottom/Up

Y’=Ywmin(B)

OR

=Ywmax(T)

X’ = X1 + (Y’-Y1)/m

I’(X’,Y’)

6. To determine region code for I’ go to step 2.

Program :

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#include<graphics.h>

#include<dos.h>

typedef struct coordinate

{

int x,y;

char code[4];

}PT;

void drawwindow()

{

line(150,100,450,100);

line(450,100,450,350);

line(450,350,150,350);

line(150,350,150,100);

}

void drawline(PT p1,PT p2)

{

line(p1.x,p1.y,p2.x,p2.y);

}

PT setcode(PT p)//for setting the 4 bit code

{

PT ptemp;

if(p.y<100)

ptemp.code[0]='1';//Top area of window

else

ptemp.code[0]='0';

if(p.y>350)

ptemp.code[1]='1';//Bottom area of window

else

ptemp.code[1]='0';

if(p.x>450)

ptemp.code[2]='1';//Right area of window

else

ptemp.code[2]='0';

if(p.x<150)

ptemp.code[3]='1';//Left area of window

else

ptemp.code[3]='0';

ptemp.x=p.x;

ptemp.y=p.y;

return(ptemp);

}

int visibility(PT p1,PT p2)

{

int i,flag=0;

for(i=0;i<4;i++)

{

if((p1.code[i]!='0') || (p2.code[i]!='0'))

flag=1;

}

if(flag==0)

return(0);

for(i=0;i<4;i++)

{

if((p1.code[i]==p2.code[i]) && (p1.code[i]=='1'))

flag='0';

}

if(flag==0)

return(1);

return(2);

}

PT resetendpt(PT p1,PT p2)

{

PT temp;

int x,y,i;

float m,k;

if(p1.code[3]=='1')

x=150;

if(p1.code[2]=='1')

x=450;

if((p1.code[3]=='1') || (p1.code[2]=='1'))

{

m=(float)(p2.y-p1.y)/(p2.x-p1.x);

k=(p1.y+(m\*(x-p1.x)));

temp.y=k;

temp.x=x;

for(i=0;i<4;i++)

temp.code[i]=p1.code[i];

if(temp.y<=350 && temp.y>=100)

return (temp);

}

if(p1.code[0]=='1')

y=100;

if(p1.code[1]=='1')

y=350;

if((p1.code[0]=='1') || (p1.code[1]=='1'))

{

m=(float)(p2.y-p1.y)/(p2.x-p1.x);

k=(float)p1.x+(float)(y-p1.y)/m;

temp.x=k;

temp.y=y;

for(i=0;i<4;i++)

temp.code[i]=p1.code[i];

return(temp);

}

else

return(p1);

}

void main()

{

int gd=DETECT,v,gm;

PT p1,p2,p3,p4,ptemp;

printf("\nEnter x1 and y1\n");

scanf("%d %d",&p1.x,&p1.y);

printf("\nEnter x2 and y2\n");

scanf("%d %d",&p2.x,&p2.y);

initgraph(&gd,&gm,"c:\\turboc3\\bgi");

drawwindow();

delay(500);

drawline(p1,p2);

delay(5000);

cleardevice();

delay(500);

p1=setcode(p1);

p2=setcode(p2);

v=visibility(p1,p2);

delay(500);

switch(v)

{

case 0: drawwindow();

delay(500);

drawline(p1,p2);

break;

case 1:drawwindow();

delay(500);

break;

case 2:p3=resetendpt(p1,p2);

p4=resetendpt(p2,p1);

drawwindow();

delay(500);

drawline(p3,p4);

break;

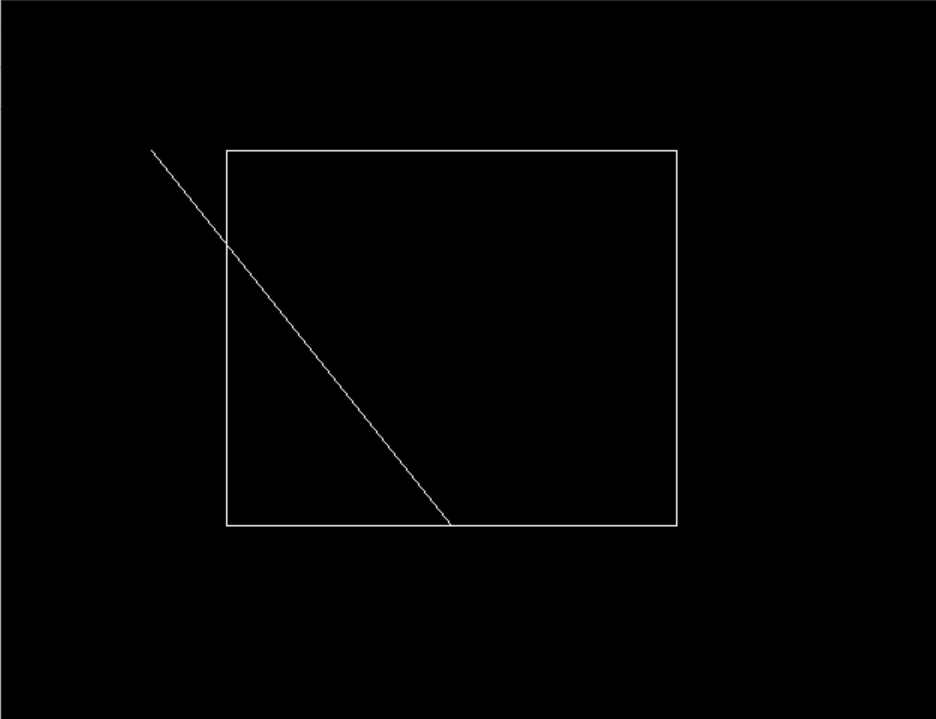
}

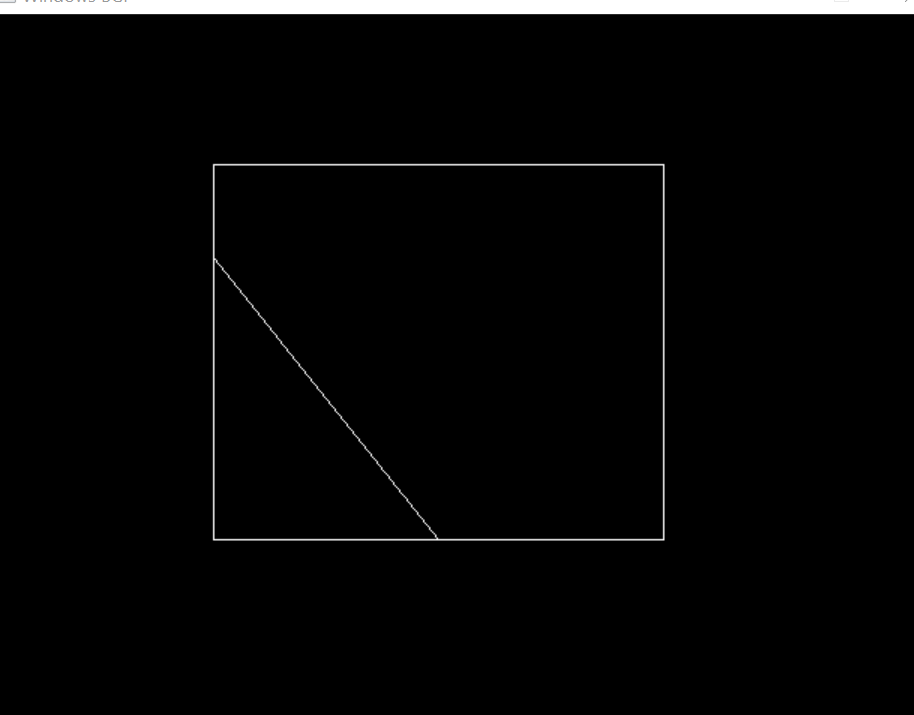
delay(5000);

closegraph();

}

**Output**

****

****

**ADVANTAGES :**

* It calculates end-points very quickly and rejects and accepts lines quickly.
* It can clip pictures much large than screen size.

**DISADVANTAGES :**

* Clipping window region can be rectangular in shape only and no other polygonal shaped window is allowed.
* Edges of rectangular shaped clipping window has to be parallel to the x-axis and y-axis.
* Complex mathematical calculations are involved and it is time consuming process.