line drawing method

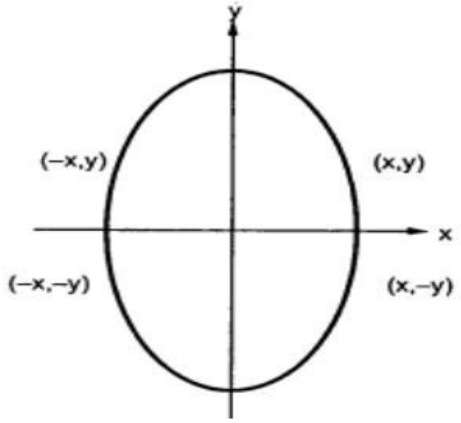
Aim-To implement midpoint Ellipse algorithm

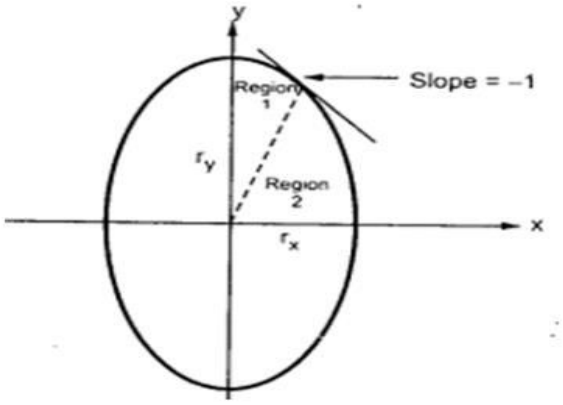
Objective:

Draw the ellipse using Mid-point Ellipse algorithm in computer graphics. Midpoint ellipse algorithm plots (finds) points of an ellipse on the first quadrant by dividing the quadrant into two regions.

Theory:

Midpoint ellipse algorithm uses four way symmetry of the ellipse to generate it. Figure shows the 4-way symmetry of the ellipse.





Here the quadrant of the ellipse is divided into two regions as shown in the fig. Fig. shows the division of first quadrant according to the slope of an ellipse with rx < ry. As ellipse is drawn from 90 0 to 0 0 , x moves in positive direction and y moves in negative direction and ellipse passes through two regions 1 and 2.

The equation of ellipse with center at (xc, yc) is given as -

[(x – xc) / rx] 2 + [(y – yc) / ry] 2 = 1

Therefore, the equation of ellipse with center at origin is given as -

[x / rx] 2 + [y / ry] 2 = 1

i.e. x 2 ry 2 + y 2 rx 2 = rx 2 ry 2

Let, f ellipse (x, y) = x2 ry2 + y2 rx2 - rx2 ry2

Algorithm:

```
x=0,y=ry
p1=ry*ry-ry*rx*rx+rx*rx/4

dx=2*x*ry*ry
dy=2*y*rx*rx
while(dx<dy)
{
putpixel(x,y)
if(p1<0)
{
p=p1+2*ry*ry*x+ry*ry
x=x+1
}
else
{
p=p1+2*ry*ry/*x+ry*ry-2*rx*rxy
x=x+1
y=y-1
}
p2=ry*ry*(x+1/2)+rx*rx*(y-1)*(y-1)-rx*rx*ry*ry
while(y>=0)
{
putpixel(x,y)
```

```
if(p2<0)
{
p=p2+rx*rx-2rx*rx*y
y=y-1
x=x+1
}
else
{
P=p2+rx*rx-2*rx*rx*y+2*ry*ry*x
Y=y-1
}
}
```

Program:

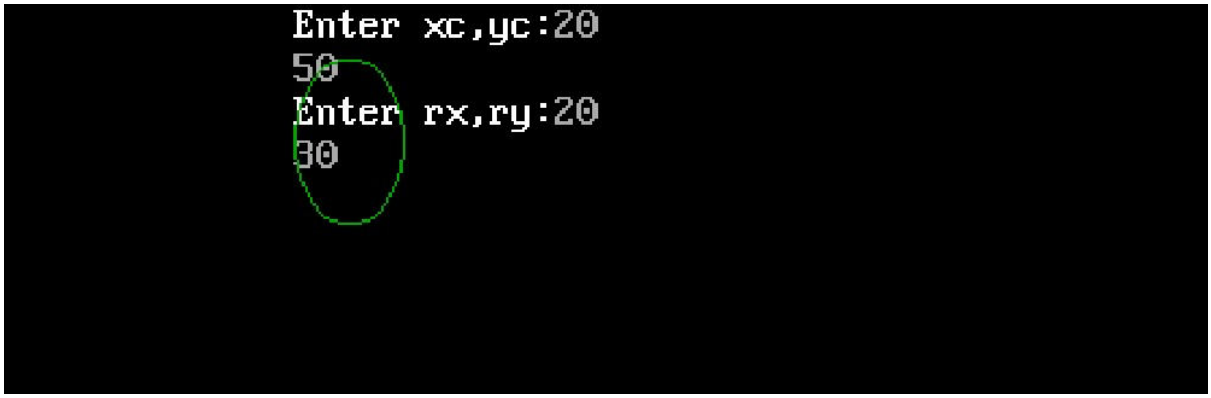
```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>

void main()
{
 int p,x,y,xc,yc,rx,ry,dx,dy;
 int gd=DETECT,gm;
 initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
 printf("Enter xc,yc:");
 scanf("%d%d",&xc,&yc);
 printf("Enter rx,ry:");
 scanf("%d%d",&rx,&ry);
```

```
x=0;
y=ry;
p=(ry*ry)-(ry*rx*rx)+((rx*rx)/4);
while((2*x*ry*ry)<(2*y*rx*rx))
{
putpixel(xc+x,yc+y,GREEN);
putpixel(xc+x,yc-y,GREEN);
putpixel(xc-x,yc-y,GREEN);
putpixel(xc-x,yc+y,GREEN);
if(p<0)
{
p=p+(2*ry*ry*x)+(ry*ry);
x=x+1;
}
else
{
p=p+(2*ry*ry*x)+(ry*ry)-(2*rx*rx*y);
x=x+1;
y=y-1;
}
}
p=(((float)x+0.5)*((float)x+0.5)*ry*ry)+(rx*rx*(y-1)*(y-1))-(rx*rx*ry*ry);
while(y>=0)
{
putpixel(xc+x,yc+y,GREEN);
putpixel(xc+x,yc-y,GREEN);
```

```
putpixel(xc-x,yc-y,GREEN);
putpixel(xc-x,yc+y,GREEN);
if(p>0)
{
y=y-1;
p=p+(rx*rx)-(2*rx*rx*y);
}
else
{
y=y-1;
x=x+1;
p=p+(2*ry*ry*x)-(2*rx*rx*y)+(rx*rx);
}
}
getch();
closegraph();
}
```

Output:



Conclusion: Comment on

1. Slow or fast :- The Midpoint Ellipse Algorithm is relatively faster compared to other methods for drawing ellipses, as it optimally utilizes symmetry and calculates only a portion of the ellipse. This efficiency makes it suitable for real-time graphics applications.

2. Difference with circle : - While the Midpoint Circle Algorithm is concerned with drawing circles, the Midpoint Ellipse Algorithm extends this concept to draw ellipses. The ellipse algorithm takes into account the two axes (major and minor) and adjusts the calculations accordingly.

3. Importance of object :- The Midpoint Ellipse Algorithm highlights the significance of using mathematical algorithms to draw complex shapes like ellipses. This approach is essential in computer graphics and design, allowing us to represent intricate shapes accurately on digital displays and screens.