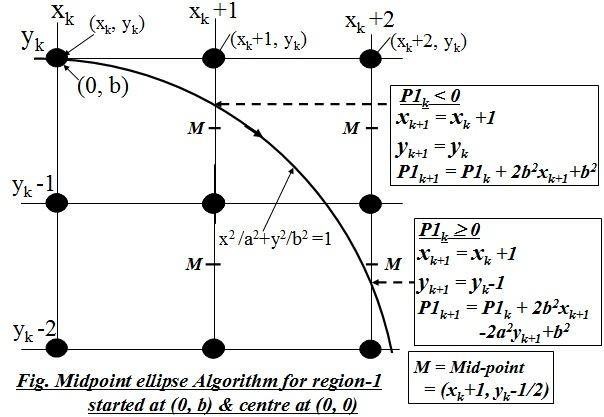
# MID-POINT ELLIPSE ALGORITHM

**Objective:**

* To implement mid-point ellipse algorithm to draw an ellipse

# Theory

Our approach here is similar to that used in displaying a raster circle but the ellipse has 4-way symmetry. The midpoint ellipse method is applied throughout the first quadrant in two parts or region as shown in figure. The region-1 just behaves as the circular property and the region-2 is slightly curve.

We have,

The equation of ellipse, whose center at (0,0) is

x2/a2 + y2/b2 = 1

Hence, we define the ellipse function for centre at origin (0,0) as:

Fellipse (x, y) = x2b2 + y2a2 – a2b2 This has the following properties:

Fellipse (x, y) < 0, if (x, y) is inside the ellipse boundary

= 0, if (x, y) is on the ellipse boundary

> 0, if (x, y) is outside the ellipse boundary

Starting at (0, b), we take unit steps in the x direction until we reach the boundary between region 1 and region 2. Then we switch to unit steps in the y direction over the remainder of the curve in the first quadrant. At each step, we need to test the value of the slope of the curve. The ellipse slope is calculated by differentiating the ellipse function as:

2xb2 + 2ya2 \* dy/dx= 0 Or dy/dx = - 2xb2 / 2ya2

At the boundary between region 1 and region 2, dy/dx = - 1 and 2xb2 = 2ya2. Therefore, we move out of region 1 whenever 2xb2 ³ 2ya2.

**For Region – 1**: Condition (2xb2 ≥ 2ya2)

Assuming that the position (xk, yk) has been plotted, we determine next position (xk+1, yk+1) as either (xk+1, yk) or (xk+1, yk-1) by evaluating the decision parameter P1k as:

P1k = Fellipse (xk+1, yk-1/2)

= b2(xk+1)2 + a2(yk-1/2)2 –a2 b2………………………………. I

At next sampling position, the decision parameter will be:

P1k+1=Fellipse (xk+1+1, yk+1-1/2)

= b2 (xk+1+1)2 + a2 (yk+1-1/2)2 –a2 b2

= b2 {(xk+1) +1}2 + a2 (yk+1-1/2)2 – a2b2

= b2 {(xk+1)2 + 2(xk+1) + 1} + a2 (yk+1-1/2)2 – a2 b2…………………………………….II

Now, subtracting I from II, we have

Pk+1 - Pk = 2b2(xk+1) + a2[(yk+12 - yk2) - (yk+1 - yk)]+ b2

If Pk < 0, the midpoint is inside the ellipse and the pixel on the scan line yk closer to the ellipse boundary. Otherwise, the midpoint is outside or on the ellipse boundary, and we select the pixel on scan line yk-1.

**Case – I**: P1k < 0

xk+1 = xk +1 yk+1 = yk

Pk+1 = Pk + 2b2(xk+1) + b2 Pk+1 = Pk+2b2xk+1+b2

**Case – II**: Pk ≥ 0

xk+1 = xk +1 yk+1 = yk - 1

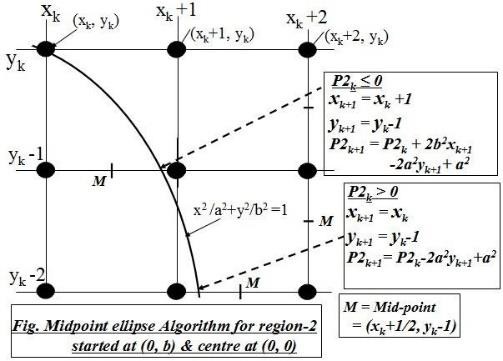
Pk+1 = Pk + 2b2(xk+1) + a2[(yk – 1)2 – yk 2)] – a2(yk 1 - yk) + b2

= Pk+2b2(xk+1) + a2[(yk 2 - 2yk +1 – yk 2)] – a2(yk 1- yk) + b2

= Pk+2b2(xk+1) – 2a2yk + a2 + a2 + b2

= Pk+2b2(xk+1) – 2a2(yk +1) + b2 Pk+1 = Pk+2b2xk+1 – 2a2yk+1 + b2

## Initial decision parameter (P10) for region-1 of ellipse

The initial decision parameter is obtained by evaluating the ellipse function at the start position (x0, y0) = (0, b). Here, the next pixel will either be (1, b) or (1, b-1) where the midpoint is (1, b -1/2). Thus, the initial decision parameter is given by:

P10 = Fellipse (1, b-1/2) = b2 + a2 (b -1/2)2 – a2b2

= b2 – a2b2 + a2 \*1/4

Thus, P10 = b2 – a2b2 + a2/4

**For Region – 2**: Condition (2xb2 < 2ya2)

Assuming that the position (xk, yk) has been plotted, we determine next position (xk+1, yk+1) as either (xk+1, yk-1) or (xk, yk-1) by evaluating the decision parameter P2k as:

P2k = Fellipse (xk+1/2, yk-1)

= b2 (xk+1/2)2 + a2 (yk-1)2 –a2 b2

At next sampling position, the decision parameter will be P2k+1 = Fellipse (xk+1+1/2, yk+1-1)

= b2 (xk+1+1/2)2 + a2 (yk+1-1)2 –a2 b2

= b2 (xk+1 +1/2)2 + a2 {(yk-1) – 1}2 – a2 b2

= b2 (xk+1 +1/2)2 + a2 {(yk-1)2 –2(yk-1)+1}–a2 b2 II

Now, subtracting I from II, we have

P2k+1 – P2k = b2 [(xk+12 - xk2) + (xk+1 - xk)] – 2a2 (yk-1) + a2

If Pk < 0, the midpoint is inside the ellipse and the pixel on the scan line xk is closer to the ellipse boundary. Otherwise, the midpoint is outside or on the ellipse boundary, and we select the pixel on scan line xk+1.

**Case – I**: P2k ≤ 0

xk+1 = xk +1 yk+1 = yk-1

P2k+1 = P2k + b2 [{(xk+1)2 - xk2} + (xk+1 - xk)] – 2a2 (yk-1) + a2

= P2k + b2 [{xk2 + 2 xk + 1 - xk2} + (xk+1 - xk)] – 2a2 (yk-1) + a2

= P2k + b2 [2xk+1+1] – 2a2 (yk-1) + a2

= P2k + 2b2 (xk+1) – 2a2 (yk-1) + a2

P2k+1 = P2k + 2b2 xk+1 – 2a2 yk+1 + a2

**Case – II**: Pk ≥ 0

xk+1 = xk

yk+1 = yk - 1

P2k+1 = P2k – 2a2 (yk-1) + a2

Pk+1 = P2k – 2a2 yk+1 + a2

## Initial decision parameter (P20) for region-2 of ellipse

When we enter region 2, the initial position (xo, y0) is taken as the last position selected in region 1 and the initial derision parameter in region 2 is given by:

P20 = Fellipse (x0+1/2, y0-1)

P20 = b2 (x0+1/2)2 + a2 (y0-1)2 –a2 b2

**Example**: Given input ellipse parameters rx = a = 8 and ry = b = 6, we illustrate the steps in the midpoint ellipse algorithm by determining raster positions along the ellipse path in the first quadrant.

2b2x = 0 (with increment 2b2 = 72) 2a2y =2a2b ( with increment-2a2=-28)

For region-1 The initial point for the ellipse centered on the origin is (x0, y0) = (0, 6), and the initial decision parameter value is: P10 = b2 – a2b2 + a2/4 = -332

Successive decision parameter values and positions along the ellipse path are calculated using the midpoint method as:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k | P1k | (Xk+1, Yk+1) At (0, 0) | 2b2Xk+1 | 2a2Yk+1 |
| 0. | -332 | (1, 6) | 72 | 768 |
| 1. | -224 | (2, 6) | 144 | 768 |
| 2. | -44 | (3, 6) | 216 | 768 |
| 3. | 208 | (4, 5) | 288 | 640 |
| 4. | -108 | (5, 5) | 360 | 640 |
| 5. | 288 | (6, 4) | 432 | 512 |
| 6. | 244 | (7, 3) | 540 | 384 |

We now move out of region 1, since 2b2X > 2a2Y

For region-2

The initial point is (x0, y0) = (7, 3) and the initial decision parameter is:

P20 = b2 (x0+1/2)2 + a2 (y0-1)2 –a2 b2 = -151

The remaining positions along the ellipse path in the first quadrant are then calculated as:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k | P1k | (Xk+1, Yk+1)At (0, 0) | 2b2Xk+1 | 2a2Yk+1 |
| 0. | -151 | (8, 2) | 576 | 256 |
| 1. | 233 | (8, 1) | 576 | 128 |
| 2. | 748 | (8, 0) | - | - |

## ALGORITHM

1. Take input radius along x axis and y axis and obtain center of ellipse.
2. Initially, we assume ellipse to be centered at origin and the first point as: (x, y0)= (0, ry).
3. Obtain the initial decision parameter for region 1 as: p10=ry2+1/4rx2-rx 2ry
4. For every xk position in region 1 :

If p1k<0 then the next point along the is (xk+1 , yk) and p1k+1=p1k+2ry2xk+1+ry2 Else, the next point is (xk+1, yk-1 )

And p1k+1=p1k+2ry2xk+1 – 2rx2yk+1+ry2

1. Obtain the initial value in region 2 using the last point (x0, y0) of region 1 as: p20=ry2(x0+1/2)2+rx2 (y0-1)2-rx2ry2
2. At each yk in region 2 starting at k =0 perform the following task. If p2k>0 the next point is (xk, yk-1) and p2k+1=p2k-2rox2yk+1+rx2
3. Else, the next point is (xk+1, yk -1) and p2k+1=p2k+2ry2xk+1 -2rx2yk+1+rx2
4. Now obtain the symmetric points in the three quadrants and plot the coordinate value as: x=x+xc, y=y+yc
5. Repeat the steps for region 1 until 2ry2x>=2rx2y

**Program to implement Mid-point Ellipse Algorithm:**

#include <iostream>

#include <stdio.h>

#include <conio.h>

#include <graphics.h>

using namespace std;

int y1;

void print\_points(int x, int y)

{

char p1[15];

sprintf(p1,"(%d, %d)",x,y);

outtextxy(x,2\*y1-y,p1);

}

int main()

{

int a, b,x1, xk, yk, pk;

cout << "Enter the semi-major axis (a) of the ellipse: ";

cin >> a;

cout << "Enter the semi-minor axis (b) of the ellipse: ";

cin >> b;

cout << "Enter the center of the ellipse: ";

cin >> x1 >> y1;

xk = 0;

yk = b;

pk = b \* b - a \* a \* b + a \* a / 4;

int k = 0;

initwindow(getmaxwidth(), getmaxheight(), "MID POINT ELLIPSE ALGORITHM");

// Draw axis lines

line(x1-a, y1, x1+a, y1);

line(x1, y1-b,x1 , y1+b);

// Draw the first quadrant of the ellipse points

do

{

putpixel(xk + x1, yk + y1, 2);

putpixel(-xk + x1, yk + y1, 3);

putpixel(xk + x1, -yk + y1, 7);

putpixel(-xk + x1, -yk + y1, 4);

if (pk < 0)

{

xk = xk + 1;

yk = yk;

pk = pk + 2 \* b \* b \* xk + b \* b;

}

else

{

xk = xk + 1;

yk = yk - 1;

pk = pk + 2 \* b \* b \* xk + b \* b - 2 \* a \* a \* yk;

}

k++;

} while (2 \* b \* b \* xk <= 2 \* a \* a \* yk);

print\_points(x1,y1);

print\_points(x1+a,y1);

print\_points(x1-a,y1);

print\_points(x1,y1-b);

print\_points(x1,y1+b);

print\_points(x1+xk,y1+yk);

print\_points(x1+xk,y1-yk);

print\_points(x1-xk,y1-yk);

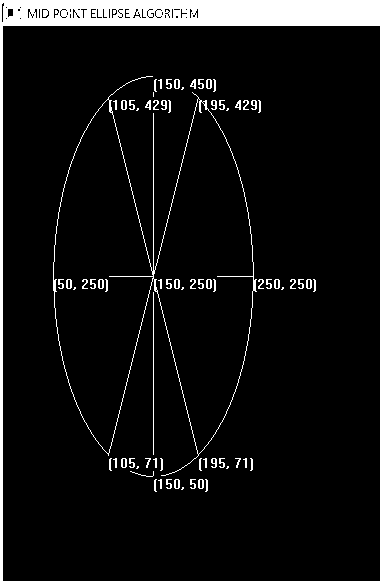
print\_points(x1-xk,y1+yk);

line(x1+xk,y1+yk,x1-xk,y1-yk);

line(x1+xk,y1-yk,x1-xk,y1+yk);

// Draw the second region of the ellipse points

pk = b \* b \* (xk \* xk + xk + 1) + a \* a \* (yk \* yk - yk) - a \* a \* b \* b;

 do

{

putpixel(xk + x1, yk + y1, 2);

putpixel(-xk + x1, yk + y1, 3);

putpixel(xk + x1, -yk + y1, 7);

putpixel(-xk + x1, -yk + y1, 4);

if (pk > 0)

{

xk = xk;

yk = yk - 1;

pk = pk - 2 \* a \* a \* yk + a \* a;

}

else

{

xk = xk + 1;

yk = yk - 1;

pk = pk+2\*b\*b\*xk-2\*a\*a\*yk+a\*a;

}

k++;

} while (yk >= 0);

getch();

closegraph();

}

**CONCLUSION**:

In this way, we implement Mid-point ellipse generating algorithm through writing code in C++ programming language and analysed its precision and way of calculation.