**ST.XAVIER’S COLLEGE**

MAITIGHAR, KATHMANDU



**Computer Graphics Assignment #6**

**Draw an ellipse using midpoint algorithm**

**Submitted By:**

Dwarika Shiwakoti

013BSCCSIT020

**Submitted to:**

|  |  |
| --- | --- |
| Er. Anil K. Sah  Lecturer, Department of Computer Science |  |

**Date of Submission: August 25, 2015**

**STATEMENT**

Write a program to draw an ellipse using the midpoint algorithm.

­

**ALGORITHM**

1. Input rX, rY and ellipse center (a,b) and obtain the first point on an ellipse centered on the origin as (x,y)=(0,rY)
2. Evaluate sqX=rX\*rX and sqY=rY\*rY
3. Calculate the initial parameter in region 1 as pX=sqY-sqX\*y+sqX/4;
4. If (pX<0), plot the pixel (x+1,y) and evaluate pX+=2\*sqY\*x+sqY

else plot the pixel (x+1, y-1) and evaluate pX+=2\*sqY\*x+sqY-2\*sqX\*y;

1. Continue executing step 4 until sqY\*x<sqX\*y
2. Evaluate the initial deciding parameter for region 2 by formula

pY=sqY\*x\*x+sqX\*y\*y-sqX\*sqY;s

1. If (pY>0) plot the pixel (x,y-1) and evaluate pY-=2\*sqX\*y+sqX;

Else plot the pixel (x+1,y-1) and evaluate pY+=2\*sqY\*x-2\*sqX\*y+rX\*rX;

1. Execute the step 7 unless y!=0
2. Find the corresponding symmetry coordinates for other 3 quadrants and plot the pixels for those quadrants
3. End

**Source code**

//---------------------------------------------------------------------------

#include <vcl\vcl.h>

#pragma hdrstop

#include "ellipse.h"

//---------------------------------------------------------------------------

#pragma resource "\*.dfm"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

int x,y;

int xC,yC,rX,rY;

int P;

xC=StrToInt(inX->Text);

yC=StrToInt(inY->Text);

rX=StrToInt(inrx->Text);

rY=StrToInt(inry->Text);

x=0; //Initial Coordinates

y=rY; //Initial Coordinates

P=(rY\*rY)-(rX\*rX\*rY)+((rX\*rX)/4); // Intial Decision Parameter for Region 1

while((2\*x\*rY\*rY)<(2\*y\*rX\*rX))

{ //Region 1 with Symmertic Property

output->Canvas->Pixels[xC+x][yC-y]=RGB(100,0,0);

output->Canvas->Pixels[xC-x][yC+y]=RGB(0,200,0);

output->Canvas->Pixels[xC+x][yC+y]=RGB(0,0,100);

output->Canvas->Pixels[xC-x][yC-y]=RGB(205,55,0);

if(P<0)

{

x=x+1;

P+=(2\*rY\*rY\*x)+(rY\*rY);

}

else

{

x=x+1;

y=y-1;

P+=(2\*rY\*rY\*x+rY\*rY)-(2\*rX\*rX\*y);

}

}

P=((float)x+0.5)\*((float)x+0.5)\*rY\*rY+(y-1)\*(y-1)\*rX\*rX-rX\*rX\*rY\*rY; // Intial Decision parameter for Region 2

while(y>=0)

{

//Region 2 with Symmertic Property

output->Canvas->Pixels[xC+x][yC-y]=RGB(200,120,0);

output->Canvas->Pixels[xC-x][yC+y]=RGB(50,100,150);

output->Canvas->Pixels[xC+x][yC+y]=RGB(10,255,155);

output->Canvas->Pixels[xC-x][yC-y]=RGB(23,245,85);

if(P>0)

{

y=y-1;

P-=(2\*rX\*rX\*y)+(rX\*rX);

}

else

{

y=y-1;

x=x+1;

P+=(2\*rY\*rY\*x)-(2\*rX\*rX\*y)-(rX\*rX);

}

}

}

//---------------------------------------------------------------------------

**Output**



