COMPUTER GRAPHICS SECTION

MidPoint Ellipse Drawing Algorithm

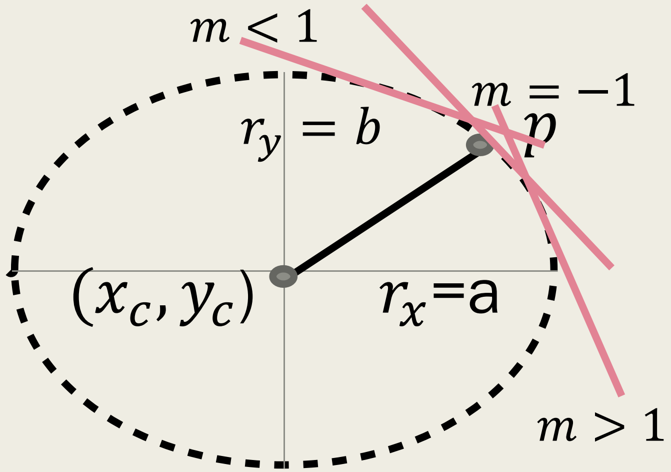
Agenda

- MidPoint Ellipse definitions.
- Mid Point Ellipse Algorithm.
- Algorithm steps.
- Code.

Mid Point Ellipse Algorithm

- With respect to Circle drawing function
  - $x^2 + y^2 - r^2 = 0$
- Suppose the initial point is x_0 , y_0

$$\left(\frac{x-x_c}{r_x}\right)^2+\left(\frac{y-y_c}{r_y}\right)^2=1$$



- Setting the initial point of ellipse $x_0 = 0$ *and* $y_0 = 0$.
- $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$
- $x^2b^2 + y^2a^2 - a^2b^2 = 0$

Mid Point Ellipse Algorithm Cont…

- Where slope m =$\frac{dy}{dx}$
- So we will calculate partial derivative with respect to x =$\boldsymbol{2xb^2}$
- So we will calculate partial derivative with respect to y =$\boldsymbol{2ya^2}$
- So m =$\frac{dy}{dx}$ = $\frac{2xb^2}{2ya^2}$
- So to remove from octant one to octant two we will stop at this condition :-
  - $2xb^2 > 2ya^2$

Mid Point Ellipse Algorithm Cont…

- For m < 1 :
  - *X - > unit interval*
  - $y_{K+1} = y_K \ / \ y_{K-1}$
  - (x_{K+1}, y_K) , (x_{K+1}, y_{K-1})
- Mid point = $(\boldsymbol{x_{\mathrm{K+1}}, y_K - \frac{1}{2}})$
- Where $\boldsymbol{x^2 {r_y}^2 + y^2 {r_x}^2 - {r_x}^2 {r_y}^2 = 0}$
  - So $p_{1K} = \boldsymbol{(x_k+1)^2 {r_y}^2 + (y_k - \frac{1}{2})^2 {r_x}^2 - {r_x}^2 {r_y}^2}$ =0
  - So $p_{1K+1} = \boldsymbol{(x_{k+1}+1)^2 {r_y}^2 + (y_{k+1} - \frac{1}{2})^2 {r_x}^2 - {r_x}^2 {r_y}^2}$ =0

Mid Point Ellipse Algorithm Cont…

So $p_{1K} = (\boldsymbol{x_k+1})^2 \boldsymbol{r_y}^2 + (\boldsymbol{y_k-\frac{1}{2}})^2 \boldsymbol{r_x}^2 - \boldsymbol{r_x}^2 \boldsymbol{r_y}^2$ =0

So $p_{1K+1} = (\boldsymbol{x_{k+1}+1})^2 \boldsymbol{r_y}^2 + (\boldsymbol{y_{k+1}-\frac{1}{2}})^2 \boldsymbol{r_x}^2 - \boldsymbol{r_x}^2 \boldsymbol{r_y}^2$ =0

- $p_{1K+1} - p_{1k} = (x_{k+1})^2 r_y^2 + r_y^2 + 2(x_{k+1}) r_y^2 + (y_{k+1})^2 r_x^2 + \frac{1}{4} r_x^2 - (y_{k+1}) r_x^2 - r_x^2 r_y^2 - (x_k+1)^2 r_y^2 - y_k^2 r_x^2 - \frac{1}{4} r_x^2 + y_k r_x^2 + r_x^2 r_y^2$
- $p_{1K+1} = p_{1k} + r_y^2 + 2(x_{k+1}) r_y^2 + r_x^2[((y_{k+1}^2) - y_k^2 \ ] - r_x^2[y_{k+1} - y_k]$

Mid Point Ellipse Algorithm Cont…

So $p_{1K} = \ (\boldsymbol{x_k+1})^2\boldsymbol{r_y}^2 + \ (\boldsymbol{y_k - \frac{1}{2}})^2\boldsymbol{r_x}^2 - \boldsymbol{r_x}^2\boldsymbol{r_y}^2$ =0

- For the initial parameter
  - *(0 , r_y).*
  - $p_{1k} = (0+1)^2 r^2{}_x + (r_y - \frac{1}{2})^2 r^2{}_x - r^2{}_x r^2{}_y = 0$
  - $p_{1k} = r^2{}_y + \frac{r^2{}_x}{4} - r_y r^2{}_x$

Mid Point Ellipse Algorithm Cont…

- For p_{1k} <0
  - $(\boldsymbol{x_k}+\mathbf{1}\ ,\ \boldsymbol{y_k})$
  - $p_{1k+1} = p_{1k} +\ 2r^2{}_y\boldsymbol{x_{k+1}}\ +r^2{}_y$
- For p_{1k} >=0
  - $(\boldsymbol{x_k}+\mathbf{1}\ ,\ \boldsymbol{y_{k-1}})$
  - $p_{1k+1} = p_{1k} +\ 2r^2{}_y\boldsymbol{x_{k+1}}$-$2r^2{}_x\boldsymbol{y_{k+1}}\ +r^2{}_y$
- Loop until
  - $2r^2{}_y x \geq 2r^2{}_x y$

Mid Point Ellipse Algorithm Cont…

- For m > 1 :
  - *y - > unit interval*
  - $x_{K+1} = x_K$ / x_{K+1}
  - (x_K, y_{K-1}) , (x_{K+1}, y_{K-1})
- Mid point = $(\boldsymbol{x}_{\mathbf{K}+\frac{1}{2}}, \boldsymbol{y}_{\boldsymbol{K}} - \mathbf{1})$
- Where $\boldsymbol{x}^2 \boldsymbol{r_y}^2 + \boldsymbol{y}^2 \boldsymbol{r_x}^2 - \boldsymbol{r_x}^2 \boldsymbol{r_y}^2 = \mathbf{0}$
  - So $p_{2K} = (\boldsymbol{x_k} + \frac{\mathbf{1}}{\mathbf{2}})^2 \boldsymbol{r_y}^2 + (\boldsymbol{y_k} - \mathbf{1})^2 \boldsymbol{r_x}^2 - \boldsymbol{r_x}^2 \boldsymbol{r_y}^2$ =0
  - So $p_{2K+1} = (\boldsymbol{x}_{\boldsymbol{k}+\frac{1}{2}} + \mathbf{1})^2 \boldsymbol{r_y}^2 + (\boldsymbol{y_{k+1}} - \mathbf{1})^2 \boldsymbol{r_x}^2 - \boldsymbol{r_x}^2 \boldsymbol{r_y}^2$ =0
- $p_{2K+1} = p_{2k} + \boldsymbol{r_x}^2 - \mathbf{2}(\boldsymbol{y_{k-1}}) \boldsymbol{r}_{\mathbf{x}}^2 + \boldsymbol{r_y}^2 [(\boldsymbol{x_{k+1}}) -^{x_k}] + \boldsymbol{r}_{\mathbf{y}}^2 [\mathbf{x}_{\boldsymbol{k}+\mathbf{1}} - \boldsymbol{x_k}]$

Mid Point Ellipse Algorithm Cont…

- For p_{2k} >= 0
  - $(\boldsymbol{x_k}, \boldsymbol{y_{k-1}})$
  - $p_{2k+1} = p_{2k} - \ 2{r^2}_x \boldsymbol{y_{k+1}} \ + {r^2}_x$
- For p_{2k} < 0
  - $(\boldsymbol{x_k} + \mathbf{1} \, , \boldsymbol{y_{k-1}})$
  - $p_{2k+1} = p_{2k} + \ 2{r^2}_y \boldsymbol{x_{k+1}} - 2{r^2}_x \boldsymbol{y_{k+1}} \ + {r^2}_x$
- Loop until
  - $2{r^2}_y x_{k+1} \geq 2{r^2}_x y_{k+1}$

Algorithm steps

1. Input rx , ry and ellipse center xc , yc and obtain the first point on an ellipse centered on the origin as x0, y0 = 0, ry.
2. Calculate the initial value of the decision parameter in region 1 as

   1. $p_{1k} = r^2{}_y + \frac{r^2{}_x}{4} - r_y r^2{}_x$

3. At each xk position in region 1, starting at k=0 perform the following test:

   If *$P1k$ < 0, The next point along the ellipse centered on 0,0 is xk + 1, yk and*

   $p_{1k+1} = p_{1k} + 2r^2{}_y \boldsymbol{x_{k+1}} + r^2{}_y$

   Else

   $p_{1k+1} = p_{1k} + 2r^2{}_y \boldsymbol{x_k} + \mathbf{1}$ -$2r^2{}_x \boldsymbol{y_{k+1}} + r^2{}_y$

Algorithm steps Cont…

where

$$2r_y{}^2x_{k+1} = 2r_y{}^2x_k + 2r_y{}^2$$

$$2r_x{}^2y_{k+1} = 2r_x{}^2y_k - 2r_x{}^2$$

And continue until

$$2r_y{}^2x \geq 2r_x{}^2y$$

Algorithm steps Cont…

- Calculate the initial value of the decision parameter in region 2 using the last point $x0$, $y0$ calculated in region 1 as

$$P2_0 = r_y^2\left(x_0 + \frac{1}{2}\right)^2 + r_x^2(y_0 - 1)^2 - r_x^2 r_y^2$$

Algorithm steps Cont…

- At each yk position in region 2, starting at k=0 perform the following test until y = 0: If $P2k$ > 0,
  - $p_{2k+1} = p_{2k} - 2{r^2}_x \boldsymbol{y_{k+1}} + {r^2}_x$

  Else

  The next point along the ellipse centered on 0, 0 is $xk+1$, $yk-1$ and

  $p_{2k+1} = p_{2k} + 2{r^2}_y \boldsymbol{x_{k+1}} - 2{r^2}_x \boldsymbol{y_{k+1}} + {r^2}_x$

Algorithm steps Cont…

- Determine symmetry points in the other three quadrants.
- Move each calculated pixel position x, y onto the elliptical path centered on xc , yc and plot the coordinate values: $\boldsymbol{x} = \boldsymbol{x} + \boldsymbol{xc}$, $\boldsymbol{y} = \boldsymbol{y} + \boldsymbol{y}$c

```
#include<GL\include\GL\freeglut.h>
#include<Windows.h>
#include<stdio.h>
void midptellipse(int rx, int ry,
int xc, int yc)
{
float dx, dy, d1, d2, x, y;
x = 0;
y = ry;

// Initial decision parameter of region 1
d1 = (ry * ry) - (rx * rx * ry) +
(0.25 * rx * rx);
dx = 2 * ry * ry * x;
dy = 2 * rx * rx * y;
glClearColor(1.0, 1.0, 1.0, 1.0);
glClear(GL_COLOR_BUFFER_BIT);
glColor3f(1.0, 0.0, 0.0);
glPointSize(1.0);
glBegin(GL_POINTS);
// For region 1
while (dx < dy)
{
```

```

// Print points based on 4-way symmetry
glVertex2i(x + xc, y + yc);
glVertex2i(-x + xc, y + yc);
glVertex2i(x + xc, -y + yc);
glVertex2i(-x + xc, -y + yc);

// Checking and updating value of
// decision parameter based on algorithm
if (d1 < 0)
{
x++;
dx = dx + (2 * ry * ry);
d1 = d1 + dx + (ry * ry);
}
else
{
x++;
y--;
dx = dx + (2 * ry * ry);
dy = dy - (2 * rx * rx);
d1 = d1 + dx - dy + (ry * ry);
}
}
```

```
// Decision parameter of region 2
d2 = ((ry * ry) * ((x + 0.5) * (x + 0.5))) +
((rx * rx) * ((y - 1) * (y - 1))) -
(rx * rx * ry * ry);

// Plotting points of region 2
while (y >= 0)
{
// Print points based on 4-way symmetry
glVertex2i(x + xc, y + yc);
glVertex2i(-x + xc, y + yc);
glVertex2i(x + xc, -y + yc);
glVertex2i(-x + xc, -y + yc);
// Checking and updating parameter
// value based on algorithm
if (d2 > 0)
{
y--;
dy = dy - (2 * rx * rx);
d2 = d2 + (rx * rx) - dy;
}
else
}
```

```
y--;
x++;
dx = dx + (2 * ry * ry);
dy = dy - (2 * rx * rx);
d2 = d2 + dx - dy + (rx * rx);
}}
glEnd();
glFlush();
}
void display()
{
midptellipse(80, 60, 0, 0);
}
int main(int argc, char** argv)
{
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowPosition(300, 300);
glutInitWindowSize(600, 600);
glutCreateWindow("Mid Point Circle Algorithm");
gluOrtho2D(-600, 600, -600, 600);
glutDisplayFunc(display);
glutMainLoop();
return 0;
}
```

Sheet_3

- Using Mid point ellipse Algorithm to define ellipse pixels in four quarters using xc=50 and yc=150:
  - *With* $R_x = 6 \; and R_y = 9$

THE END