Experiment 4: Ellipse Generation Algorithm

Aim: To implement the Ellipse Generation Algorithm for drawing an ellipse of given center (xc, yc) and radius rx and ry.

Description

In an ellipse, symmetry exists between quadrants but not between the two octants of a single quadrant. Therefore, pixel positions must be calculated along the elliptical arc through one quadrant, and then positions in the remaining 3 quadrants are obtained by symmetry. The next pixel is chosen based on a decision parameter.

Algorithm

1. Input rx, ry, and ellipse center (xc, yc), and obtain the first point on an ellipse centered on the origin as $(x_0, y_0) = (0, ry)$.
2. Calculate the initial parameter in region 1 as $p1 = ry^2 - rx^2 ry + \frac{1}{4} rx^2$.
3. At each x_i position, starting at $i = 0$, if $p1_i < 0$, the next point along the ellipse centered on $(0, 0)$ is $(x_i + 1, y_i)$ and $p1 = p1 + 2ry^2 x_{i+1} + ry^2$. Otherwise, the next point is $(x_i + 1, y_i - 1)$ and $p1 = p1 + 2ry^2 x_{i+1} - 2rx^2 y_{i+1} + ry^2$. Continue until $2ry^2 x \geq 2rx^2 y$.
4. (x_0, y_0) is the last position calculated in region 1. Calculate the initial parameter in region 2 as $p2 = ry^2(x_0 + 0.5)^2 + rx^2(y_0 - 1)^2 - rx^2 ry^2$.
5. At each y_i position, starting at $i = 0$, if $p2_i > 0$, the next point along the ellipse centered on $(0, 0)$ is $(x_i, y_i - 1)$ and $p2 = p2 - 2rx^2 y_{i+1} + rx^2$. Otherwise, the next point is $(x_i + 1, y_i - 1)$ and $p2 = p2 + 2ry^2 x_{i+1} - 2rx^2 y_{i+1} + rx^2$.

Program

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

void disp();
float x, y;
int xc, yc;

void main()
{
    int gd = DETECT, gm;
    int rx, ry;
    float p1, p2;
    clrscr();
    initgraph(&gd, &gm, "C:\\TurboC3\\BGI");

    printf("Enter the center point :");
    scanf("%d%d", &xc, &yc);
    printf("Enter the value for Rx and Ry :");
    scanf("%d%d", &rx, &ry);

    x = 0;
    y = ry;
    disp();

    /* -------- Region 1 -------- */
    p1 = (ry * ry) - (rx * rx * ry) + (rx * rx) / 4.0;
    while ((2.0 * ry * ry * x) <= (2.0 * rx * rx * y))
    {
        x++;
        if (p1 <= 0)
            p1 = p1 + (2.0 * ry * ry * x) + (ry * ry);
        else
        {
            y--;
            p1 = p1 + (2.0 * ry * ry * x) - (2.0 * rx * rx * y) + (ry * ry);
        }
        disp();
    }

    /* -------- Region 2 -------- */
    p2 = (ry * ry) * (x + 0.5) * (x + 0.5) + (rx * rx) * (y - 1) * (y - 1) - (rx * rx * ry * ry);
    while (y > 0)
    {
        y--;
        if (p2 > 0)
            p2 = p2 + (rx * rx) - (2.0 * rx * rx * y);
        else
        {
            x++;
            p2 = p2 + (2.0 * ry * ry * x) - (2.0 * rx * rx * y) + (rx * rx);
        }
        disp();
    }
    getch();
    closegraph();
}

void disp()
{
    delay(50);
    putpixel(xc + x, yc + y, 10);
    putpixel(xc - x, yc + y, 10);
    putpixel(xc + x, yc - y, 10);
    putpixel(xc - x, yc - y, 10);
}
```