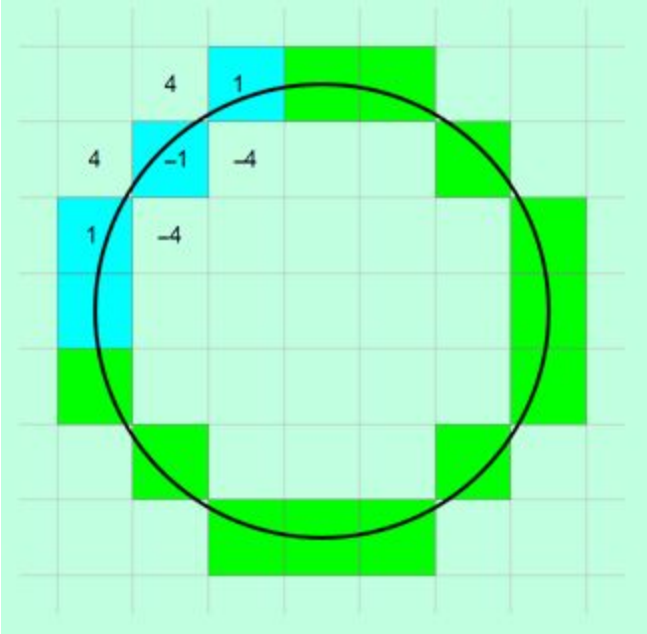
Tecnológico Nacional de México Campus Ensenada

Graficación

Primer Examen Parcial
Algoritmo de Punto Medio



Jesús Luque Espinoza

Ing. en Sistemas Computacionales

Prof. Abraham Flores Vergara

Ensenada, BC.

13 de septiembre del 2019

Descripción matemática

El punto medio, en matemática, es el punto que se encuentra a la misma distancia de cualquiera de los extremos. Una circunferencia se define como un conjunto de puntos que se encuentran, en su totalidad, a una distancia determinada r de una posición central (xc , yc), en otras palabras, el punto medio de una circunferencia es el radio. Esta relación de distancia se expresa por medio del teorema de Pitágoras en coordenadas cartesianas como:

$$(x - xc)2 + (y - yc)2 = r2$$

Cada cuadrante de la circunferencia es similar en si mismo. Se puede generar la sección circular del segundo cuadrante del plano de xy al notar que las dos secciones circulares son simétricas con respecto del eje de las y. Y las secciones circulares del tercero y el cuarto cuadrantes se pueden obtener a partir de las secciones del primero y el segundo cuadrantes al considerar la simetría en relación con el eje de las x.

También se puede decir que hay simetría entre octantes. Las secciones circulares en octantes adyacentes dentro de un cuadrante son simétricas con respecto de la línea a 45° que divide los dos octantes. Al aprovechar la simetría de la circunferencia de esta manera, se podrá generar todas las posiciones de píxel alrededor de una circunferencia, calculando sólo puntos dentro del sector de x = 0 a x = y.

Es posible reducir el cálculo al considerar la simetría de las circunferencias, la forma de la circunferencia es similar entre cuadrantes y simétrica entre octantes.

Para aplicar el método del punto medio, definimos una función de circunferencia como:

$pk = fcircunferencia(x,y) = x2 + y2 - r2$

$fcircunferencia(x,y)<0$ si (x,y) está dentro de la frontera de la circunferencia.

$fcircunferencia(x,y)=0$ si (x,y) está en la frontera de la circunferencia.

$fcircunferencia(x,y)>0$ si (x,y) está fuera de la frontera de la circunferencia.

Los parámetros de decisión sucesivos se obtienen al utilizar cálculos incrementales.

Algoritmo de punto medio

El algoritmo de punto medio funciona de la misma forma que el algoritmo de la línea de rastreo, se efectúa un muestreo en intervalos unitarios y se determina la posición del píxel más cercano a la trayectoria específica de la circunferencia en cada paso.

Se determina un radio r y una posición de origen en la pantalla (x,y), para establecer el primer algoritmo y poder calcular las posiciones de píxel alrededor de una trayectoria circular centrada en el origen de la coordenadas (0,0).

Así, cada posición calculada (x,y) se mueve a su posición propia en la pantalla al sumar (*xc*+*x*) y (*yc*+*y*). A lo largo de la sección circular de x=0 a x=y en el primer cuadrante, la pendiente de la curva varía entre 0 y -1.

Por tanto, se puede tomar pasos unitarios en la dirección positiva de x en este octante y utilizar un parámetro de decisión para determinar cuál de las dos posiciones posibles de y está más próxima a la trayectoria de la circunferencia en cada paso. Las posiciones de los otros siete octantes se obtienen entonces por simetría.

Para aplicar el método punto medio, se debe definir una función de circunferencia como:

$$f\ circunferencia\ (x\ ,\ y) = x2 + y2 - r2$$

f circunferencia (x , y)= 0. Si el punto está en el interior de la circunferencia, la función de la circunferencia es negativa; y si está en su exterior, es positiva.

Al igual que en el algoritmo para el trazo de líneas de Bresenham, el método del punto medio calcula las posiciones de píxel a lo largo de una circunferencia utilizando adiciones y sustracciones de enteros, si se supone que los parámetros de la circunferencia se especifican en coordenadas enteras de pantalla.

Los pasos del algoritmo de la circunferencia de punto medio se reducen a:

1. Capturar el radio r y el centro de la circunferencia (xc , yc) y se obtiene el primer punto de una circunferencia centrada en el origen como (x0 , y0) = (0 ,r).
2. Calcular el valor inicial del parámetro de decisión como p0 = 5/4 - r.
3. En cada xk posición, al iniciar en k = 0, se realiza la prueba siguiente:

   Si pk < 0, el siguiente punto a lo largo de la circunferencia centrada en (0 , 0) es (xk+1 , yk) y pk+1 = pk +2xk+1 +1.

   De otro modo, el siguiente punto a lo largo de la circunferencia es

   (xk + 1, yk -1) y pk+1 = pk + 2xk+1 + 1 -2yk+1 donde 2xk+1 = 2xk + 2 y 2yk+1 = 2yk -2.

4. Determinar puntos de simetría en los otros siete octantes.
5. Mover cada posición de pixel calculada (x , y) a la trayectoria circular centrada en (xc , yc) se trazan los valores de las coordenadas: x = x + xc y= y + yc.
6. Repetir los pasos 3 a 5 hasta que x ≥ y.

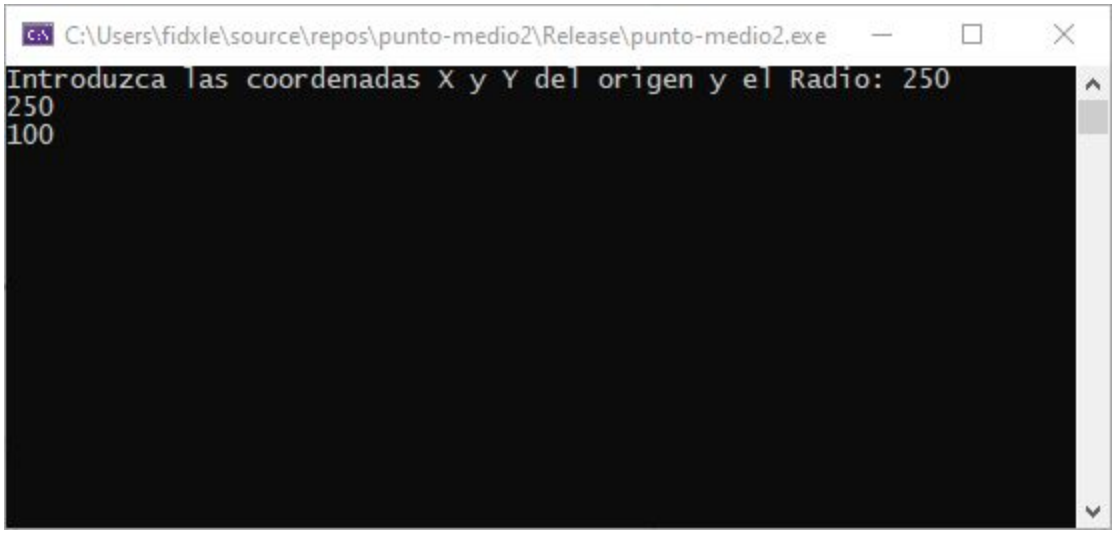
Programa en C++

Código C++

```
#include<iostream>
#include<GL/glut.h>
#include<stdlib.h>
#include<math.h>
using namespace std;
/*Variables x,y,radio*/
int x00,y00,r00;
/*Método que define el color de la ventana y la cantidad de pixeles.*/
void init(){
    glClearColor(255, 255, 255, 255);
    glMatrixMode(GL_PROJECTION);
    gluOrtho2D(0, 500, 0, 500);
}
/*Método que genera un punto en la ventana*/
void setPixel(int x, int y){
    glBegin(GL_POINTS);
    glColor3f(0.0, 0.0, 0.0);
    glVertex2i(x, y);
    glEnd();
}
/*Método de bresenham para la generación de un circulo.*/
void plotCircle(int xm, int ym, int r){
    int x = -r, y = 0, err = 2 - 2 * r;                    /* bottom left to top right */
    do {
        setPixel(xm - x, ym + y);                          /*   I. Quadrant +x +y */
        setPixel(xm - y, ym - x);                          /*  II. Quadrant -x +y */
        setPixel(xm + x, ym - y);                          /* III. Quadrant -x -y */
        setPixel(xm + y, ym + x);                          /*  IV. Quadrant +x -y */
        r = err;
        if (r <= y) err += ++y * 2 + 1;                    /* e_xy+e_y < 0 */
        if (r > x || err > y)                              /* e_xy+e_x > 0 or no 2nd y-step */
            err += ++x * 2 + 1;                            /* -> x-step now */
    } while (x < 0);
}
/*Método que genera la vista del circulo.*/
void display(){
    glClear(GL_COLOR_BUFFER_BIT);
    plotCircle(x00, y00, r00);
    glutSwapBuffers();
}
/*Método principal. Se crea una ventana y se inicializan las coordenadas a dibujar.*/
int main(int argc, char* argv[]){
    cout << "Introduzca las coordenadas X y Y del origen y el Radio: ";
    cin >> x00 >> y00 >> r00;
    glutInit(&argc, argv);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition(10, 10);
    glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH);
    glutCreateWindow("Algoritmo Punto Medio - Bresenham");
    init();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```

Capturas de pantalla

Entrada de datos:



Graficación de puntos: