Laporan Bresenham

Mata Kuliah Grafika Komputer

Oleh:

IQBAL HARIO SYAHPUTRA

(20051397029)

2020A

Program Studi D4 Manajemen Informatika

Fakultas Vokasi

Universitas Negeri Surabaya

2022

- **Cara Kerja Bresenham:**

  Algoritma Bresenham adalah sebuah algoritma yang dibentuk oleh bresenham yang tidak kalah akurat/efisien dengan algoritma primitif lainnya (DDA). Cara kerja Algoritma Bresenham adalah mengecek garis yang sudah diubah hanya dengan menggunakan metode perhitungan integer yang nantinya akan terus bertambah sehingga bisa menampilkan bentuk lingkaran dan bentuk kurva lainnya. Rumus dari algoritma ini yaitu :

```
1.  Rumus algoritma bresenham:
2.
3.  x = 0
4.  y = r
5.
6.  d = 3 - (2 * r)
7.
8.  if d < 0:
9.  x += 1
10. d += 4 * x + 6
11. else:
12. x += 1
13. y -= 1
14. d += (4 * (x - y)) + 10
```

  Pada pratikum kali ini saya mecoba mengambarkan lingkaran dengan algoritma bresenham hal terpenting adalah menentukan nilai radiusnya. Dari situlah kita dapat menghitung posisi untuk koordinat x dan y nya. Berdasarkan rumus di atas, jika kita memberikan nilai 5 pada radius, maka akan didapat 5 juga pada y dan -7 pada d-nya. 'D' di sini merupakan decision parameter, berguna untuk menentukan apakah nilai dari koordinat x dan y itu bertambah atau berkurang. Setiap koordinat x dan y di sini mewakili satu buah pixel.

  Berikut ini tahapan mengambarakan lingkaran dengan algoritma bresenham menggunakan opengl bahasa python:

  1. Tentukan nilai radius

  2. Tentukan posisi dari lingkaran yang akan dibuat.

  3. Tentukan nilai x, y dan d berdasarkan rumus algoritma bresenham.

  4. Plot sebuah pixel.

  5. Jika d kurang dari 0, ikuti rumus terkait.

  6. Selain itu, ikuti rumus terkait.

7. Plot sebuah pixel.

8. Selama y lebih x, ulangi langkah 5 sampai 7.

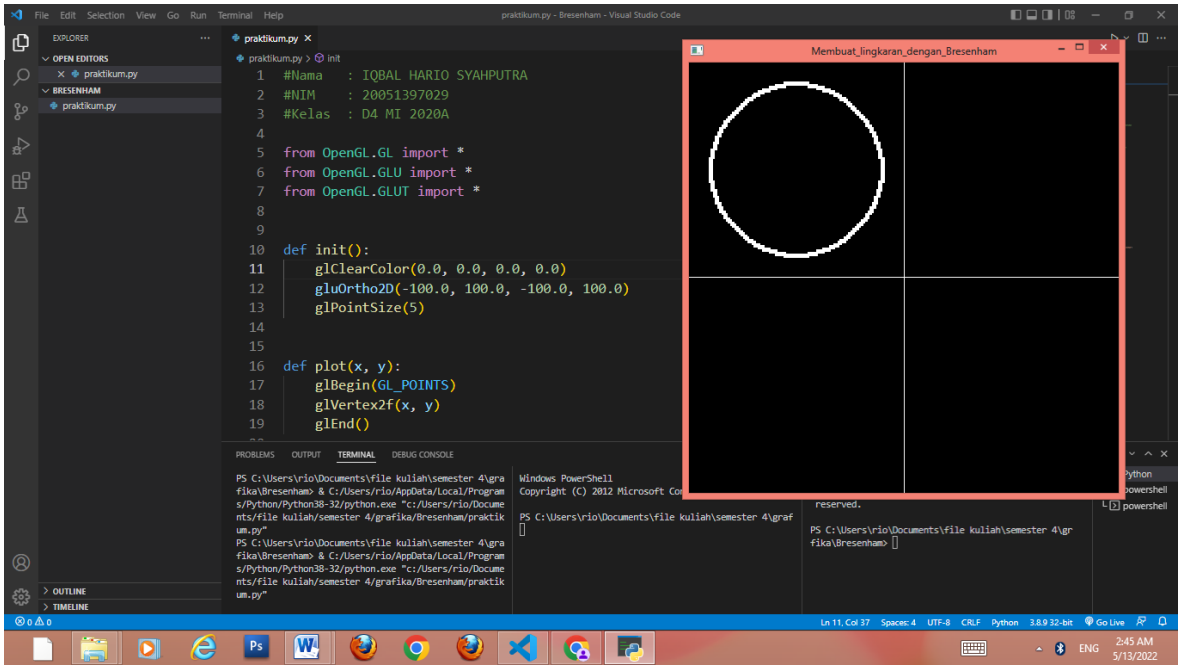
- **Source Code**

```
#Nama   : IQBAL HARIO SYAHPUTRA
#NIM    : 20051397029
#Kelas  : D4 MI 2020A

from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *


def init():
    glClearColor(0.0, 0.0, 0.0, 0.0)
    gluOrtho2D(-100.0, 100.0, -100.0, 100.0)
    glPointSize(5)


def plot(x, y):
    glBegin(GL_POINTS)
    glVertex2f(x, y)
    glEnd()


def Membuat_lingkaran_dengan_Bresenham(r):

    # lokasi lingkaran yang akan dibuat

    x_position = -50
    y_position = 50

    x = 0
    y = r

    # decision parameter
    d = 3 - 2 * r

    # membuat sebuah titik pada koordinat yang ditentukan
```

```
    plot(x + x_position, y + y_position)

    while y > x:

        if d < 0:
            x += 1
            d += 4 * x + 6
        else:
            x += 1
            y -= 1
            d += (4 * (x - y)) + 10

        # Tidak perlu mencari semua nilai koordinat
        # cukup dapatkan nilai (x, y) saja
        # lalu balik menjadi (y, x)

        # Untuk pixel (x, y)

        # Quadrant 1
        plot(x + x_position, y + y_position)

        # Quadrant 2
        plot(x + x_position, -y + y_position)

        # Quadrant 3
        plot(-x + x_position, -y + y_position)

        # Quadrant 4
        plot(-x + x_position, y + y_position)

        # Untuk pixel (y, x)

        # Quadrant 1
        plot(y + x_position, x + y_position)

        # Quadrant 2
        plot(-y + x_position, x + y_position)

        # Quadrant 3
```

```
        plot(-y + x_position, -x + y_position)

        # Quadrant 4
        plot(y + x_position, -x + y_position)


def plotpoints():

    glClear(GL_COLOR_BUFFER_BIT)
    glColor3f(1, 1.0, 1.0)

    glBegin(GL_LINES)

    glVertex2f(-100, 0)
    glVertex2f(100, 0)

    glVertex2f(0, -100)
    glVertex2f(0, 100)

    glEnd()

    Membuat_lingkaran_dengan_Bresenham(40)

    glFlush()


def main():
    glutInit(sys.argv)
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB)
    glutInitWindowSize(500, 500)
    glutInitWindowPosition(100, 100)
    glutCreateWindow("Membuat_lingkaran_dengan_Bresenham")
    glutDisplayFunc(plotpoints)

    init()
    glutMainLoop()


main()
```

- **Output:**


praktikum.py - Bresenham - Visual Studio Code
Membuat_lingkaran_dengan_Bresenham
#Nama : IQBAL HARIO SYAHPUTRA
#NIM : 20051397029
#Kelas : D4 MI 2020A
from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *
def init():
glClearColor(0.0, 0.0, 0.0, 0.0)
gluOrtho2D(-100.0, 100.0, -100.0, 100.0)
glPointSize(5)
def plot(x, y):
glBegin(GL_POINTS)
glVertex2f(x, y)
glEnd()