Bresenham's Circle Drawing Algorithm in Windows API

1. Introduction

This document provides a detailed explanation of the implementation of Bresenham's Circle Drawing Algorithm using the Windows API. The program allows a user to draw a circle interactively by clicking and releasing the left mouse button to define the center and radius of the circle.

2. Overview of the Implementation

The program creates a Windows application where users can draw circles by selecting a center and dragging the mouse to determine the radius. The drawing is performed using the Bresenham algorithm, which efficiently plots points in an 8-way symmetric manner to create a smooth circle.

3. Implementation Details

3.1. Header Files and Namespaces

```
#include <windows.h>
#include <cmath>
using namespace std;
```

The `<windows.h>` header file provides access to the Windows API functions, while `<cmath>` is used for mathematical operations.

3.2. Drawing Function

3.2.1. Function to Plot Eight Symmetric Points

```
void Draw8Points(HDC hdc, int xc, int yc, int x, int y, COLORREF
c) {
    SetPixel(hdc, xc + x, yc + y, c);
    SetPixel(hdc, xc - x, yc + y, c);
    SetPixel(hdc, xc - x, yc - y, c);
    SetPixel(hdc, xc + x, yc - y, c);
    SetPixel(hdc, xc + y, yc + x, c);
    SetPixel(hdc, xc - y, yc + x, c);
    SetPixel(hdc, xc - y, yc - x, c);
    SetPixel(hdc, xc + y, yc - x, c);
}
```

This function plots eight symmetric points around the center to optimize the drawing process.

3.2.2. Bresenham's Circle Drawing Algorithm

```
void DrawCircleBresenham(HDC hdc, int xc, int yc, int R,
COLORREF c) {
    int x = 0, y = R, d = 1 - R, d1 = 3, d2 = 5 - 2 * R;
    Draw8Points(hdc, xc, yc, x, y, c);
    while (x <= y) {
        if (d < 0) {
            d += d1;
            d2 += 2;
        } else {
            d += d2;
            d2 += 4;
            y--;
        }
        d1 += 2;
        x++;
        Draw8Points(hdc, xc, yc, x, y, c);
    }
}
```

This function implements Bresenham’s algorithm to draw the circle efficiently by making incremental calculations.

3.3. Windows Procedure Function

```
LRESULT WindowProc(HWND hwnd, UINT m, WPARAM wp, LPARAM lp){
    HDC hdc;
    static int xc, yc;
    int x, y;
    if (m == WM_LBUTTONDOWN) {
        xc = LOWORD(lp);
        yc = HIWORD(lp);
    }
    else if (m == WM_LBUTTONUP) {
        x = LOWORD(lp);
        y = HIWORD(lp);
        int r = (int)sqrt((x - xc) * (x - xc) + (y - yc) * (y -
yc));
        hdc = GetDC(hwnd);
        DrawCircleBresenham(hdc, xc, yc, r, RGB(255, 0, 0));
    }
    else if (m == WM_DESTROY) {
        PostQuitMessage(0);
    }
    else{
        return DefWindowProc(hwnd, m, wp, lp);
    }
    return 0;
}
```

- **WM_LBUTTONDOWN:** Captures the initial mouse click position as the circle's center.
- **WM_LBUTTONUP:** Determines the radius and triggers the drawing function.
- **WM_DESTROY:** Closes the application.

3.4. Main Function

```
int APIENTRY WinMain(HINSTANCE hInstance, HINSTANCE
hPrevInstance, LPSTR lpCmdLine, int nCmdShow) {
    WNDCLASS wc = {};
    wc.hbrBackground = (HBRUSH)GetStockObject(WHITE_BRUSH);
    wc.hCursor = LoadCursor(nullptr, IDC_CROSS);
    wc.hIcon = LoadIcon(nullptr, IDI_APPLICATION);
    wc.lpszClassName = "MyClass";
    wc.lpfnWndProc = WindowProc;
    wc.style = CS_HREDRAW | CS_VREDRAW;
    wc.hInstance = hInstance;
    RegisterClass(&wc);
    HWND hwnd = CreateWindow("MyClass", "Circle Drawer",
WS_OVERLAPPEDWINDOW,
                             CW_USEDEFAULT, CW_USEDEFAULT, 700,
700, nullptr, nullptr, hInstance, nullptr);
    ShowWindow(hwnd, nCmdShow);
    UpdateWindow(hwnd);
    MSG msg = {};
    while (GetMessage(&msg, nullptr, 0, 0)) {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
    return 0;
}
```

This function initializes and runs the Windows application.

4. Execution Flow

1. The application window is created and displayed.
2. The user clicks inside the window to set the circle's center.
3. The user drags the mouse and releases the button to determine the radius.
4. The circle is drawn using the Bresenham algorithm.

5. Conclusion

This program demonstrates how to use the Windows API to create a simple interactive drawing tool using Bresenham's Circle Drawing Algorithm. It provides an efficient way to draw circles with minimal computational overhead and high accuracy.