COSC 4370 – Homework 1

Name: Austin Thibodeaux PSID: 2037865

September 2022

1 Problem

This assignment requires the rasterization of a single ellipse. The problem asks to draw the ellipse $(\frac{x}{12}) + (\frac{y}{6}) = 64^2$ where $x \geq 0$. This equation can be rewritten as $0 = (6 \cdot 64)^2 x^2 + (12 \cdot 64)^2 y^2 - (12 \cdot 64)^2 (6 \cdot 64)^2$. The right half of the ellipse shall be displayed in the bmp file.

2 Method

Five functions in total have been defined in this project:

- **GetWidth:** Gets width of bmp. Wrapper around *g_bmp.bmp_info_header* for easier access.
- **GetHeight:** Gets height of bmp. Wrapper around *g_bmp.bmp_info_header* for easier access.
- **SetPixel:** Wrapper around *g_bmp.set_pixel(…)* that checks for image bounds before drawing.
- **FlipAndDraw:** Flips a copy of the drawing upside down around specified origin coordinates and calls *SetPixel* for both.
- **MidpointEllipse:** Draws an ellipse to the screen using the Midpoint Ellipse algorithm. Requires parameters for location of center (x,y) and the major and minor axis of the ellipse.

The midpoint ellipse algorithm is based on Bresenham's algorithm, where only the cheapest arithmetic is used to approximate the pixels closest to the actual function. In this case, only addition/subtraction is used in this algorithm.

An ellipse has four-way symmetry; thus, it is only necessary to draw one quadrant of the ellipse and the resultant can be copied and flipped to the other three quadrants. This significantly increases performance as only one-fourth of the ellipse must be rendered. In the case of this problem, only one flip is necessary as everything less than 0 on the x-axis is not rendered.

3 Implementation

In this implementation, the image size will be 800×800 and the origin of the ellipse will be translated up 400 pixels. This will ensure that the bottom half of the ellipse is not cut off by the bounds of the image. The main function will initially fill the canvas with black, and then calls *MidpointEllipse* with the parameters *offsetY=0, offsetX=400, a=12*64, and b*6*64*, where *a* is the major axis and *b* is the minor axis. Then the image is saved to a bmp file.

3.1 MidpointEllipse

This function will draw, and only draw the first quadrant of an ellipse. The algorithm is divided into two regions with region 1 transitioning to region 2 when the slope of the ellipse reaches -1. The slope is defined by $\frac{dx}{dy} = \frac{2b^2x}{2a^2y}$.

The first region is computed using a while loop that exits on the condition that the slope reaches -1. In the first iteration, *x* will start at 0 any *y* will start at the minor axis. The first pixel (and its mirrored counterpart) is drawn by calling *FlipAndDraw*. Before the first iteration, a *prediction variable*, named **p1**, is used to determine which of two pixels is selected to be painted. The value of *p1* is obtained by finding the value of the ellipse function *f(x+1, y-0.5)*, which is a point between the East and Southeast neighboring pixels. If the value is less than 1, the *East* pixel is selected, and *dx*, the slope numerator, and prediction function (*f(x+2, y – 0.5))* are updated for the next iteration. If the value is greater or equal to 1, the *Southeast* pixel is selected, and *dx, dy* slope numerator and denominator, and the prediction variable *(f(x+2, y – 3/2))* are updated for the next iteration.

When the slope reaches -1, the second region begins. The second region is computed using another while loop that exits on the condition that y is greater than 0 or hits the x-axis. A second prediction variable is declared **p2** is similarly used to determine which of the two pixels is selected to be painted. The value of *p2* is obtained by the value of the ellipse function *f(x+0.5, y-1)*. If the value is greater than 1, the *South* pixel is selected, *dy* is increased, and *p2* is updated to *f(x+0.5, y-2)*. Otherwise, the *Southeast* pixel is selected, *dx* and *dy* are increased, and *p2* is updated to *f(x+3/2, y-2)*. After this while loop the algorithm is finished and the program exits and saves the bmp file.

4 Results

When *MidpointEllipse* finishes its second region, the program will then create a bmp file. The image consists of the right half of an ellipse, centered vertically, drawn in a red color with a black background.

References

- https://www.geeksforgeeks.org/midpoint-ellipse-drawing-algorithm/
- https://www.includehelp.com/computer-graphics/mid-point-ellipse-algorithm.aspx
- https://www.cpp.edu/~raheja/CS445/MEA.pdf