

Benjamin Keurian
CS 460
2/20/2022

Problem

This project asked us to draw the letters “WAKE” using the Bresenham and Midpoint line drawing algorithms, in addition to drawing the letters with OpenGL’s line drawing functionally. The project also allows the user to draw lines on the canvas using the mouse.

Implementation

1. Bresenham Algorithm

- a. The Bresenham line drawing algorithm draws a line between two points by calculating the next pixel to be drawn, while avoiding floating point division. If the slope, m , of a given line is between $0 \leq m < 1$, the decision variable for calculating the next point is $d = 2*dy - dx$, where dy is $|y_2 - y_1|$ and dx is $|x_2 - x_1|$. For every x value between x_1 and x_2 , the decision variable gets updated. If the current value of the decision variable is less than or equal to 0, the new value of the decision variable is $d += 2*dy$, otherwise the new value is $d += 2(dy - dx)$ and the y value (initially set to y_1) is incremented by 1. However, this will only work if $dx > dy$. If $dy > dx$, the logic is relatively the same, with a few tweaks. In this case, the decision variable is $2*dx - dy$. If the decision variable is less than 0, then the new value of d is now $d += 2*dx$, otherwise $d += 2(dx - dy)$. There is also additional logic to handle the cases when $x_2 < x_1$ and when $y_2 < y_1$

2. The Midpoint Algorithm

- a. The Midpoint Line drawing algorithm has very similar logic for the Bresenham algorithm and chooses the next pixel to draw by using the midpoint of the Northeast point and the east point. If the midpoint is above the line, then the east point is chosen. If the midpoint is below the line, then the north east point is chosen. The midpoint algorithm handles the logic for cases when $dx > dy$, $x_1 > x_2$, and when $y_1 > y_2$ in a similar fashion to the Bresenham algorithm

3. Drawing Figures Using the Mouse


- a.

To implement this feature, the program utilizes functions found in the OpenGL library. In the main function, `glutMouseFunc(mouse)` registers the mouse function to handle mouse click events and `glutMotionFunc(mouseMove)` registers the `mouseMove` function to handle mouse dragging events. In the `mouseMove` function, if the user clicks the left button and if the right button has not been clicked yet, the program will generate a new control point called “basePoint”. Once the user releases the left button, a line from `basePoint` to current mouse position will be drawn and stored into a vector called “lines”. Upon clicking the right mouse button, a line will be drawn from the most recently created `basePoint` to the current mouse position. A flag called “done_drawing” is set to true to indicate that the right mouse button has been clicked. If `done_drawing` is true and the left mouse button is clicked, the previously drawn line will be erased.

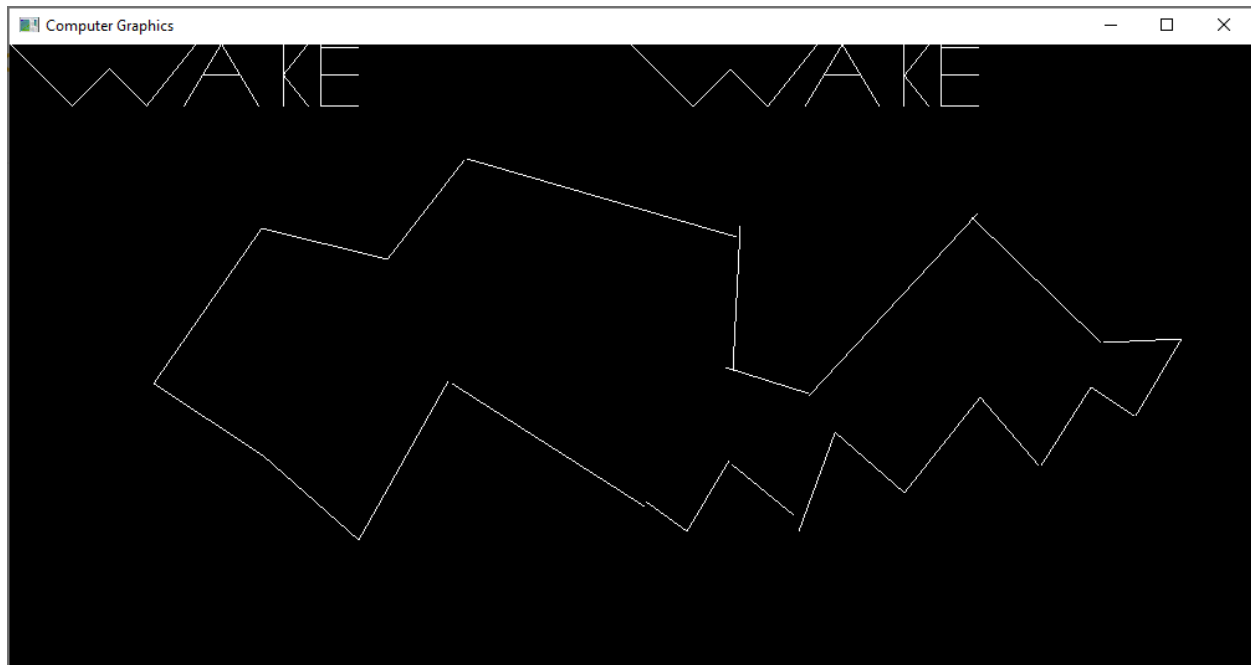
The `mouseMove` draws a line from the `basePoint` to the current mouse position, and redraws the line as the mouse moves. The `mouseMove` function first calls `glClear(GL_COLOR_BUFFER_BIT)` to clear the canvas. Without calling `glClear(GL_COLOR_BUFFER_BIT)` multiple lines will be drawn as the user moves the mouse. The function then redraws all the lines that are stored in the “lines” vector.

4. Running the Program and Screen Shots

- a. When running the program the user will be asked to enter either 1 to draw WAKE using the bresenham algorithm or 2 to draw WAKE using the midpoint algorithm. Then WAKE will be drawn with the selected algorithm on the left side of the canvas and on the right hand side using OpenGL. I used Visual Studio 2022 with the Debug and x64 configurations. The source code is found in the file `Proj1.cpp`.



```
C:\Users\User\Desktop\CS460\x64\Debug\CS460.exe
Enter 1 for Bresenham or 2 for midpoint
2
midpoint
```



b.