Reporte Taller 1.5 - Juan Sebastián Prado Valero

En el proceso de recrear los algoritmos en C++ se tuvieron que considerar ciertos casos especiales en los que el algoritmo no funcionaba del todo bien ya sea por el orden de las coordenadas o el valor de la pendiente de la recta a dibujar.

En el algoritmo DDA fue necesario realizar un redondeo especial cuando la pendiente de la recta estaba entre -1 y 1, pues al manejar valores flotantes y transformar el resultado de la posición a entero se visualiza en este rango algo como esto:

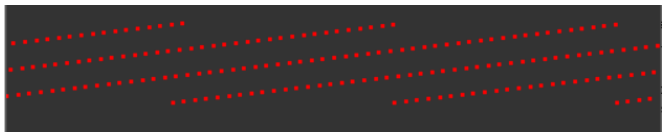

Figura 1

Por lo que fue necesario transformar los valores a enteros antes de calcular la posición para que se viera bien la recta. Ya con ese estilo de redondeo se ven las rectas, pero se ven demasiado pixeladas y si se amplia la imagen se pueden apreciar discontinuidades.

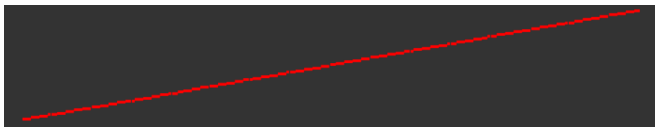

Figura 2

En el caso del algoritmo de Bresenham fue necesario tener dos cosas en cuenta:

1. La pendiente, pues si esta está entre -1 y 1, el eje dominante (sobre el cual hay que iterar) sería el eje Y para poder graficar de forma correcta la recta, o de contrario sería el eje X.
2. El orden como se ingresaban los puntos, pues el algoritmo como tal va sumando en un ciclo sobre un eje y al ingresar al algoritmo puntos donde el dy y el dx sean negativos las rectas no se podrían graficar o se graficarían mal. Teniendo que cambiar los X o Y iniciales con los finales y definiendo si se incrementara o disminuiría en la iteración sobre el eje secundario.

Con este algoritmo se pude ver mejor la recta, con más detalles y pocas discontinuidades.

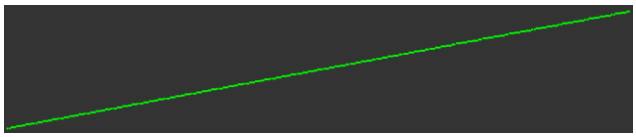

Figura 3

A pesar de que en el algoritmo DDA se ve con menos pixeles, este tiene muchos en común con el algoritmo de Bresenham. Además, tiene pixeles que el Bresenham no tiene y de igual forma el Bresenham tiene unos que el DDA no tiene.

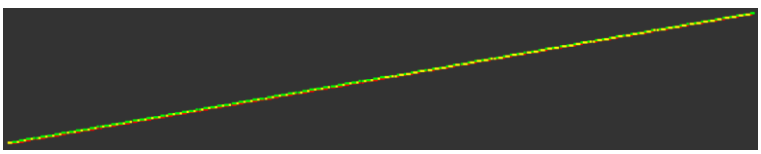

Figura 4