Reporte de Implementación

Parcial 1

Angie Paola Jaramillo Ortega

Despartamento de Ingeniería Electrónica y
Telecomunicaciones
Universidad de Antioquia
Medellín
Septiembre de 2021

Índice

1. Introducción **2**

2. Diagrama de clases **2**

3. Clases implementadas **3**

4. Módulos **3**
4.1. Lectura 3
4.2. Redimensionamiento 3
4.3. Escritura de matriz en .txt 4
4.4. Visualización bandera 4

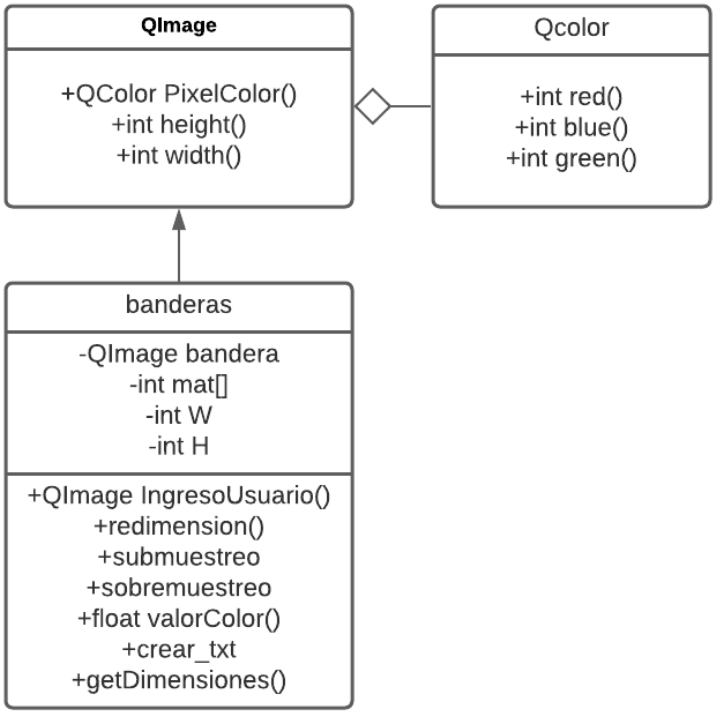
5. Estructura de Circuito **4**

6. Inconvenientes **5**

1. Introducción

De forma general el programa funciona de la siguiente forma: Se lee la ruta de la imagen, se crea un objeto de clase QImage, se obtienen la medidad de ancho y altura de la imagen ingresada, se comparan cada dimension a 10 y dependiendo de esto se realiza un escalamiento indicado, se guardan los valores de la nueva imagen en una matriz de tamaño [3][10][10] siendo 3 por los valores rojo, verde y azul y 10 por ser una matrix de LEDs 10x10, 10 columnas y 10 filas, la matriz se escribe la matriz en un archivo de text, y luego de acabado el programa en Qt se copia lo escrito en el archivo, se pega en el codigo de tinkercad y se inica la simulaciónmostrando la imagen.

2. Diagrama de clases



3. Clases implementadas

Para la solución del problema se utilizaron principalemente tres clases. Se creó una clase banderas que hereda la clase QImage para crear un atributo tipo QImage que corresponde a la bandera a modificar, esta clase banderas contiene los diferentes mmétodos creados para realizar el escalamiento de las imagenes usando los métodos heredados pixelColor(), blue(),green() y red() para obtener los valores de colores RGB de cada pixel necesario.

4. Módulos

4.1. Lectura

Al instanciar la clase bandera el constructor llama al metodo IngresoUsuario() para pedir el nombre de la ruta de la imagen y crear un objeto de la clase QImage que luego pasará a ser un atributo "bandera"de la clase banderas().

4.2. Redimensionamiento

Ya creado el atributo bandera usamenos el método redimension(). Dentro de este método lo primero que se hace es llamar a otro método getDimension() que dará valor a los atributos W y H que corresponden al ancho y alto de bandera. Luego, se evaluan los valores de W y H y dependiendo de si estos valores son mayores, iguales o menores a 10 se realiza un proceso diferente de muestreo.

Si la imagen ingresada por el usuario es de 10x10 se recorrerá la imagen obteniendo los valores de los colores rojo,verde y azul de cada uno y guardandolos en la matriz definida como atributo de la clase banderas.

Sobremuestreo Si uno o ambos de los lados resulta ser mayor a 10 se realiza el sobremuestreo que utiliza el algoritmo de interpolación bilinear.En resumen lo que esto hace es tomar los cuatro pixeles más cercanos al que se necesite encontrar y el valor de cada color del pixel promediado por la cercania que tenga a cada uno de estos cuatro pixeles más cercanos. Mientras más cercano esté a un pixel el color del pixel buscado será más similar a el de este pixel.Es aplicar El peso promediado en ambas dimensiones usando el método valorColor().

Submuestreo Para el submuestreo se hace un promediado en un grupo nxm de pixeles dependiendo de cuantos grupos de pixeles se puedan tomar 10 veces en la imagen. El promediado se realiza de izquierda a derecha y de arriba a abajo.

Después de realizar submuestreo y sobremuestreo los valores resultantes durante el proceso se guardan en la matriz de la nueva imagen.

4.3. Escritura de matriz en .txt

Primero se crea un archivo de texto llamado matrizLEDs, luego recorremos la matriz dividiendo sus "niveles"por { y sus elementos por ','.Al finalizar se cierra el archivo.

4.4. Visualización bandera

Ya estando en tinkercad y pegado la información escrita en el archivo de texto se lee la matriz recorriendola y asignando a cada LED tres valores RGB correspondientes para que se ilumine.

5. Estructura de Circuito

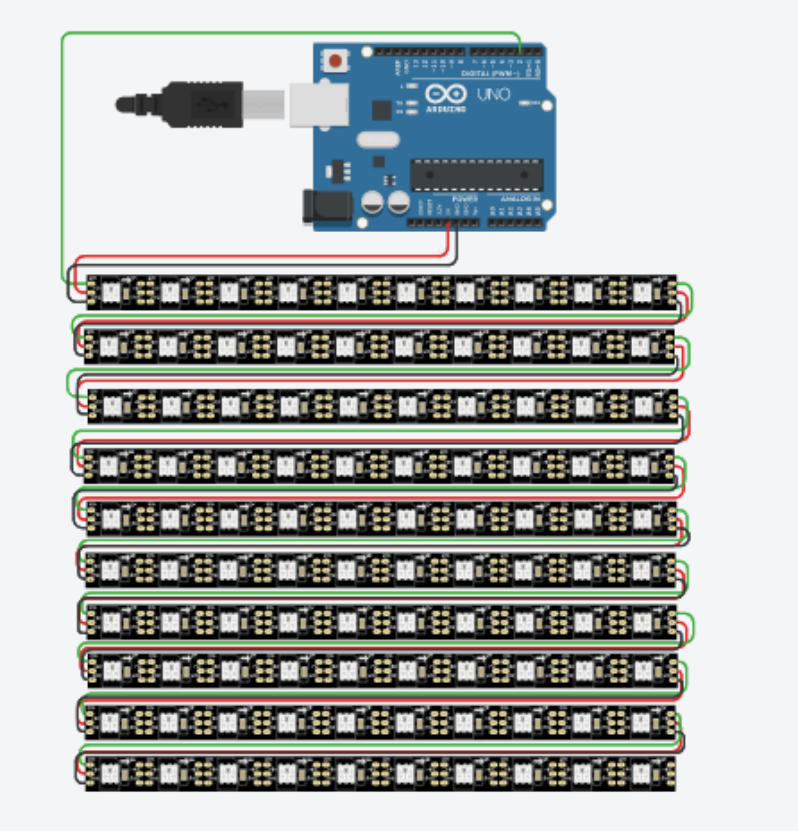


Figura 1: Conexiones de la pantalla

6. Inconvenientes

El mayor inconveniente que presenté durante la realización del programa fue en la función del submuestreo. Como las medidas de la imagen que se lee son aleatoreas normalmente cuando la imagen no es divisible por 10 y es de pocos pixeles resulta en tener que ignorar cierta cantidad de pixeles durante el promediado y cuando los pixeles ignorados eran solo del lado derecho e inferior de la imagen muchas veces terminaba en una imagen incompleta y que no se podia identificar. Encontrar la forma de leer pixeles más cercanos al centro de la bandera resultó complicado también porque los pixeles sobrantes no siempre son divisible entre 2 por lo que de uno de los lados debía sobrar más pixeles que el otro.