Analisis y diseño

Juan Jose Arboleda Cardona

Despartamento de Ingeniería Electrónica y Telecomunicaciones
Universidad de Antioquia
Medellín

Septiembre de 2021

Índice

1. Analisis

problema: En esta ocasión su tarea consiste en recibir una imagen en formato jpg y realizar el procesamiento de la información contenida en esta, de tal forma que se haga un ajuste de sus dimensiones, para que pueda ser presentada en una matriz de LEDs RGB. plantear: - conexiones en Arduino - funcion para redimensionar la imagen de entrada

2. propuestas

las siguientes son las propuestas que planteo para la funcion que redimensiona las imegenes. la imagen se recorrera con un ciclo 'for' anidado para el eje x y y tomando linea por linea para su modificacion.

2.1. metodo1

para modificar el tamaño de la imagen por sus pixeles, podemos buscar los pixeles bordes (que definan los limites entre los colores de la imagen) y para aumentar el tamaño de la imagen, aumentaremos estos los a en 1 ... los restantes los aumentaremos poco apoco para rellenar espacios. para disminuir la imagen disminuimos todo menos estos pixeles bordes.

2.2. metodo 2

calculando en una linea de pixeles el porcentaje que ocupa cada color: comenzamos contando el numero de colores y las veces que cada uno de ellos se repite en la linea.ya sea para agrandar o disminuir la imagen se sabe cuanto porciento ocupa un color en esa linea de pixeles en n posicion. luego vamos a organizarlo en la nueva matriz de forma que quede equivbalente a la imagen original y haciendo una regla de 3 se rellenarian los espacios o se disminuirian respectivamente al tamaño de la imagen deseada en el programa.

2.3. segundo analisis

luego de intentar con los metodos propuestos por mi, me doy cuneta que la complejidad del algoritmo es mayo, por lo cual recurro a buscar en internet otros algoritmos para poder dimensionar una imagen. encontre algoritmos como:

- interpolacion lineal.

- interpolacion bilineal.

- interpolacion del vecino mas cercano.

-Interpolación bicúbica.

en base a estos me propuse a replantear mi algortimo de solucion.

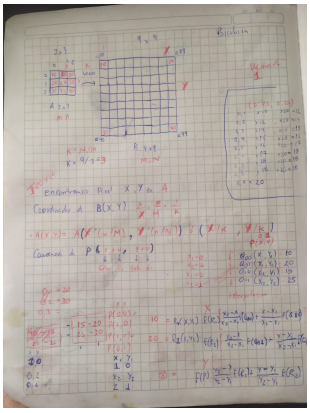

Figura 1: analisis en papel interpolacion cubica

3. conexiones

mostrare mi planteamiento para la conexion de arduino

Figura 2: imagen conexiones

4. código en el documento

A continuación, se presenta el código de las clases definidas para la solucion

```
#include <iostream>
#include <QImage>
#include <vector>
#include <map>
using namespace std;
class imagen{
private:
    string ruta;
    map<string,vector<int>> RGB;
```

```
public:
    QImage png;
    imagen(string); // constructor
    void resize(int, int, QImage,map<string,vector<int>>);
};
imagen::imagen(string _ruta){
    ruta=_ruta;
    QImage image(ruta.c_str());
    png= image;
    /*
    en el constructor creamos un objeto de tipo Qimage llamado png para acceder
    a las funciones de este
    */
}
int main()
{
    string ruta;
    cout <<"ingrese la ruta de la imagen: ";
    cin >> ruta;
    imagen objImagen (ruta);
    return 0;
}
```