

Parcial II 2021-1

Informática 2

Informe de Análisis y Diseño de la Solución del
Desafío

Juan Diego Cabrera Moncada

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

Índice

1. Análisis del problema y consideraciones para la alternativa de solución propuesta	2
2. Esquema de tareas del desarrollo del algoritmo	4
3. Diseño del algoritmo	5
4. Consideraciones a tener en cuenta en la implementación	5

1. Análisis del problema y consideraciones para la alternativa de solución propuesta

En esta fase de análisis, al reconstruir el código que se había propuesto en clase, pude verificar que entendía en su totalidad el método de obtención de la información de cada pixel de una imagen en específico. Posterior a ello, asumí

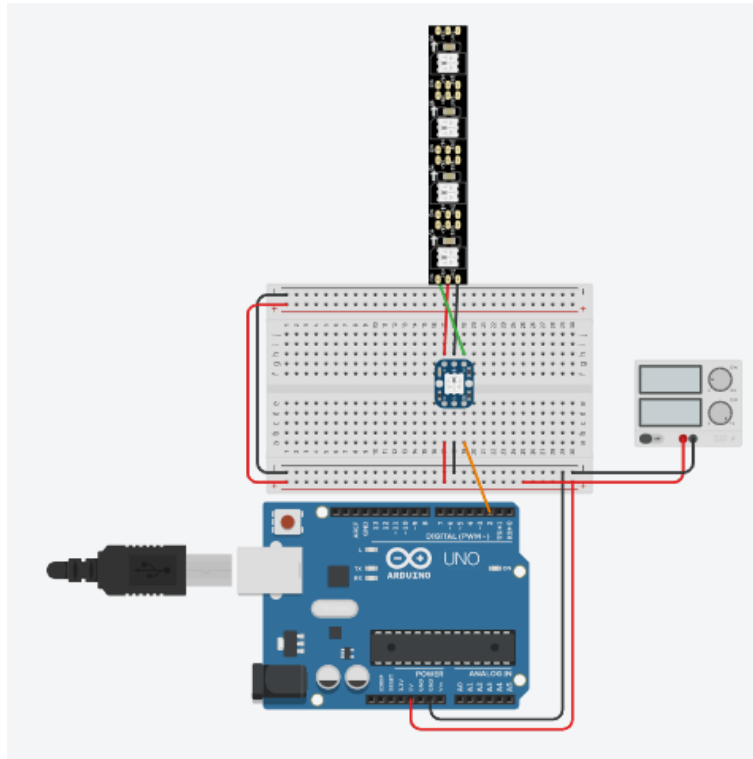


Figura 1: Réplica del circuito propuesto en clase

que el mayor problema en la programación sería hacer una función con la cual hiciese el proceso de submuestreo de una imagen, y otra similar, que hiciese el de sobremuestreo, en caso de que alguna de las dos sea necesaria. Para ello, en primer lugar, enfocándome en el submuestreo, pensé en que si obtenía el cociente de la división del ancho de la imagen entre el número de columnas de la matriz de Neopixeles o de LEDs, y de forma similar el alto entre el número de filas, para, posteriormente, usar dichos dos valores (Llamémoslos C_x y C_y), para hallar el promedio de un número C_x de valores (Teniendo en cuenta que pueden ser diferentes dependiendo de cuál de los 3 colores RGB esté analizando matricialmente) y así reducir ese número C_x a 1, y unir todas los valores resultantes hasta crear la versión submuestreada de la matriz. No obstante, al consultar opiniones al respecto en foros y otros sitios web, encontré que esto puede resultar inefec-

tivo por cuestiones de que, hablando de un solo punto de la matriz, la distancia que habría entre los puntos de la matriz real que debe representar el punto de la matriz de submuestreo implica que unos puntos tengan más "peso" que otros al momento de hacer este proceso [1]. No obstante, uno de los comentarios [2] proponía hacer el promedio de modo que se haga un submuestreo de reducir la matriz a la mitad en cada ejecución de la función, y, una vez se encuentre cerca de la matriz deseada, hacer la consideración de la influencia que tendría la distancia relativa entre un punto de la matriz real y el punto de la matriz de submuestreo que lo representa. He decidido optar por una matriz de 8 x 8 usando 8 tiras de 8 Neopixeles cada una. Por el momento, procedo a hacer una verificación de que el funcionamiento del código y el circuito está bien planteado para una matriz de Neopixeles de esta cantidad. Al analizar detenidamente la

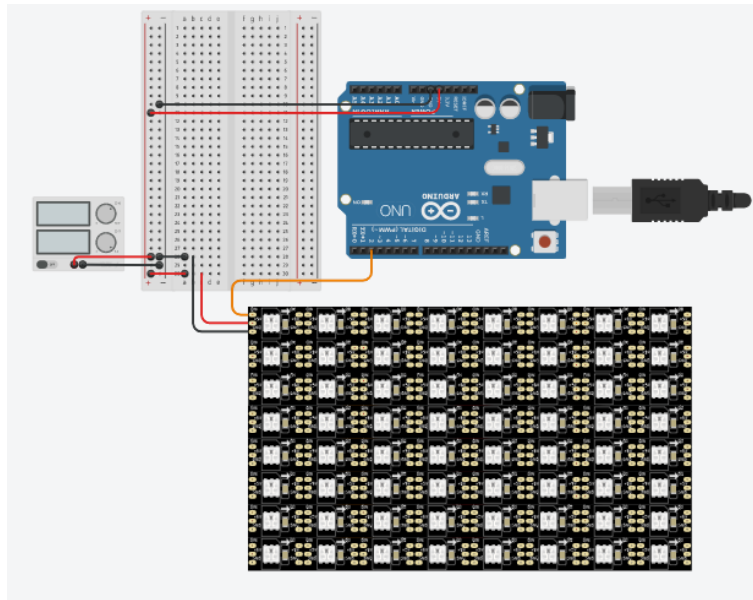


Figura 2: Construcción de matriz 8x8 de Neopixeles

matriz de Neopixeles y después de unas cuantas pruebas en código, considero que el algoritmo debe fundamentarse en el manejo de dos clases principalmente, una de ellas que debe encargarse, a través de sus métodos, de poder leer cada uno de los pixeles que componen a la imagen cuya dirección ingrese el usuario en el programa en Qt, así como, de ser necesario, poder usar métodos que permitan obtener el valor de cada color (Rojo, verde y azul) y modificarlos si así lo plantea la implementación del algoritmo. La segunda clase debe encargarse de guardar los datos acerca de los valores enteros de rojo, azul y verde que debe adquirir individualmente cada neopixel, los cuales son obtenidos después de haber procesado la imagen, ya sea por submuestreo o sobremuestreo de la misma. Dicha clase debe garantizar que se pueda acceder a dichos valores enteros de manera

eficiente con el objetivo de escribir dichos valores en un archivo de texto bajo un formato preestablecido. En cuanto al programa en Tinkercad, considero que esta alternativa de solución debe permitir que los datos del archivo de texto sean leídos y guardados en una matriz de variables tridimensional, considerando que se trata de valores enteros para 3 colores diferentes, que deben ser representados en una matriz de 8 columnas por 8 filas. Asimismo, a través de las funciones del programa, se debe usar una librería que permita la configuración del valor del color de los neopixeles en formato RGB (Rojo, verde y azul), de modo que los valores enteros guardados anteriormente en la matriz tridimensional se usen como referencia para configurar los neopixeles que conforman la matriz de 8 x 8 de tiras de Neopixeles.

2. Esquema de tareas del desarrollo del algoritmo

En primer lugar, procedo a analizar la lista de tareas que se deben ejecutar para desarrollar el algoritmo de solución del desafío de manera efectiva. Analizando objetivos de forma secuencial, considero que, primero, es fundamental establecer el método de interacción con el usuario para solicitar la imagen a procesar, por lo cual, una parte de ello, consta de la creación del manual de uso del programa., que debe abordar a su vez, lo que debe hacer el usuario con el archivo de texto resultante del submuestreo o sobremuestreo de la imagen (En caso de que sea alguno de los 2 necesario), incluyendo cómo y dónde debe usarlo, pues dicho archivo contiene la información necesaria para que funcione correctamente la solución implementada en Tinkercad (Primera impresión de cómo debe conectarse Tinkercad y Qt). Una vez hechas las partes de interacción con el usuario, la siguiente tarea corresponde a crear un método de lectura de la información de la imagen y proveer una forma, mediante código, de guardar dicha información de modo que resulte cómoda de usar al procesarla para la cuestión de submuestreo o sobremuestreo de la imagen. Posterior a ello, debo definir métodos que me permitan manipular la información almacenada. Después, es necesario definir métodos, ciclos y ejecuciones necesarias para el caso en que se requiera realizar submuestreo de la imagen, en paralelo con aquellos necesarios para el caso de sobremuestreo de la imagen. Es necesario, después, garantizar métodos así para guardar la nueva información procesada de la imagen en conjunto con la manipulación del archivo de texto. Asimismo, otra tarea fundamental, la cual puede hacerse en paralelo con el proceso anteriormente descrito, radica en la construcción de la matriz de Neopixeles de modo que sea funcional, en primera instancia, y posterior a que esto se garantice, priorizar la efectividad en la representación matricial de la información procesada de la imagen obtenida por vía del archivo de texto creado y modificado. Esta última tarea, se debe subdividir en la garantía de la representación matricial para cada uno de los colores RGB por separado como primer objetivo y, después, en conjunto.

más fundamental es procurar que los elementos de la segunda clase, la que se encarga del guardado de los valores enteros finales de cada color de cada pixel, posea los elementos privados que sean útiles tanto para cumplir con su principal objetivo como aquellos que sean manejados más como herramientas auxiliares en el procesamiento de los datos de la imagen, que son obtenidos y modificados gracias a la primera clase instaurada.

Referencias

- [1] U. P. de Valencia. Ejemplo de aplicación del re-muestreo mediante interpolación bilineal. [Online]. Available: <https://www.youtube.com/watch?v=z4S5kBl6P84>
- [2] Anónimo. ¿qué algoritmos usar para reducir la imagen? - c ++, c, procesamiento de imágenes, cambio de tamaño, reducción de tamaño. [Online]. Available: <https://living-sun.com/es/c/136090-what-algorithms-to-use-for-image-downsizing-c-c-image-processing-resize-downsize.html>