



Tarea 1: Parte 1

Ingeniería en Computadores

CE 5201: Procesamiento y Análisis de Imágenes Digitales

Profesor: Juan Pablo Soto Quiros

Nasser Santiago Brown Aparicio 2019043776

Kenichi Hayakawa Bolaños 2020044884

Jose Antonio Retana Corrales 2020144743

Pregunta 1: Investigue e implemente computacionalmente el filtro de la mediana para eliminar el ruido de una imagen. Incluir una breve explicación del filtro de la mediana, su formulación matemática y su respectivo pseudocódigo.

El filtro de mediana es un filtro no lineal que se utiliza en el procesamiento de imágenes, este funciona ordenando los valores de los píxeles en un *neighborhood*, encontrando el valor de la mediana y reemplazando el valor original del píxel con la mediana de ese *neighborhood*. Cuando se habla de una operación aplicada en un *neighborhood* es una técnica de procesamiento de imágenes en las que el valor resultante para un píxel de referencia (x_0, y_0) es una función del valor del píxel original en ese punto, así como del valor del píxel original de algunos de los píxeles circundantes. En el caso del filtro de la mediana, este es un filtro no lineal ya que no utiliza la convolución para realizar la operación como otros tipos de *Neighborhood Processing*

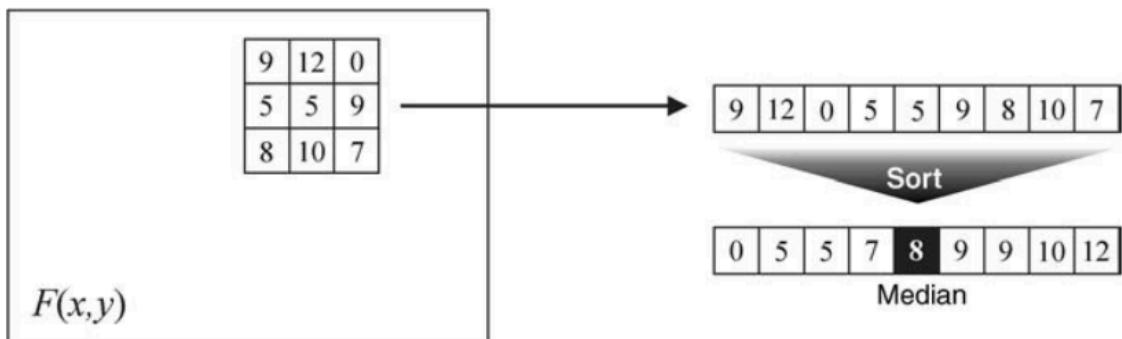


FIGURE 10.8 Median filter. Redrawn from [BB08].

La formulación matemática de un filtro 3x3 de la mediana para una imagen en escala de grises es la siguiente:

$$g(x, y) = \text{med}\{z_k | k = 1, 2, 3, \dots, 9\}$$

donde z_k es el conjunto de valores en el *neighborhood* del pixel de referencia.

En el caso de una imagen a color, basta con aplicar el mismo filtro a cada canal por separado, uniendo los resultados en una sola imagen

Pseudocódigo:

```
def Y = mediana(I):
    //crear imagen vacía del mismo tamaño
    Y = create_empty_image(I.width, I.length)

    size_filter = 5

    //iterar por todos los píxeles de la imagen
    for x from 0 to I.width - 1:
        for y from 0 to I.height -1:
            neighborhoodR = []
            neighborhoodG = []
            neighborhoodB = []

            //Encontrar el neighborhood del pixel
            offset = size_filter // 2
            for ox from -offset to offset:
                for oy from -offset to offset:
                    nx = x + ox
                    ny = y + oy
                    //Ver si no se sale de la imagen
                    if (0 <= nx < I.width) && (0 <= ny < I.height):
                        //agregar al neighborhood
                        //canal rojo
                        neighborhoodR.append(I[nx, ny].R)
                        //canal verde
                        neighborhoodG.append(I[nx, ny].G)
                        //canal azul
                        neighborhoodB.append(I[nx, ny].B)

                    //ordenar las listas del neighborhood
                    neighborhoodR = sort(neighborhoodR)
                    neighborhoodG = sort(neighborhoodG)
                    neighborhoodB = sort(neighborhoodB)
                    //encontrar mediana de cada lista del neighborhood
                    med_pos = len(neighborhoodR)//2
                    medianR = neighborhoodR[med_pos]
                    medianG = neighborhoodG[med_pos]
                    medianB = neighborhoodB[med_pos]

                    Y[x, y] = [medianR, medianG, medianB]

    return Y
```

Resultados:

Pregunta 1: Filtro de la Mediana

Al aplicar el filtro de la mediana en python, basado en el pseudocódigo presentado anteriormente, se logra observar como los pixeles negros de la imagen original se eliminan y se suaviza el resto de la imagen, como se puede observar en la Fig 1



Fig 1. Imagen original vs el resultado al aplicar el filtro de la mediana

Pregunta 2: Filtro de la Mediana

Una vez implementada la rotación, se puede observar como este tiene los pixeles negros de esperarse al hacer esto por medio de una transformación a fin, como se puede observar en la segunda imagen de la Fig 2. Una vez obtenida esta, se pasó por el mismo filtro diseñado en la Pregunta 1, donde se obtuvo una versión suavizada sin los pixeles negros, como se puede ver en la tercera imagen de la Fig 2. Ya obtenida la imagen con el filtro aplicado en toda la imagen, se implementó otra función que reemplazara todos los pixeles negros de la imagen rotada sin el filtro con los pixeles de la imagen rotada con el filtro, obteniendo el resultado final de la Figura 2.



Fig 2. Imagen original, Imagen rotada, Imagen rotada con el filtro aplicado en toda la imagen, y la Imagen con el filtro de la mediana aplicado solo en los pixeles negros

Bibliografía:

Marques, O. (2011). *Practical image and video processing using MATLAB*. Wiley.

González, R., & Woods, R. (2018). *Digital image processing* (4th ed.). Prentice-Hall.