

REPORTE DE PRÁCTICA

IDENTIFICACIÓN DE LA PRÁCTICA

Práctica	11	Nombre de la práctica	Filtros no lineales
Fecha	19/10/2025	Nombre del profesor	Alma Nayeli Rodríguez Vázquez
Nombre del estudiante		Jesús Alberto Aréchiga Carrillo	

OBJETIVO

El objetivo de esta práctica consiste en implementar los filtros no lineales mínimo, máximo y el de la mediana sobre imágenes ruidosas.

PROCEDIMIENTO

Realiza la implementación siguiendo estas instrucciones.

Realiza un programa en Python utilizando OpenCV. Implementa los filtros mínimo, máximo y el de la mediana atendiendo las siguientes indicaciones:

1. Utiliza una imagen en escala de grises
2. Agrega ruido sal y pimienta a la imagen en escala de grises con `iaa.SaltAndPepper(p=0.05)`
3. Implementa un filtro mínimo de 3x3 a la imagen ruidosa con `ndimage.minimum_filter()`
4. Implementa un filtro máximo de 3x3 a la imagen ruidosa con `ndimage.maximum_filter()`
5. Implementa el filtro de la mediana de 3x3 a la imagen ruidosa con `cv.medianBlur()`
6. Reporta los resultados obtenidos.

IMPLEMENTACIÓN

Agrega el código de tu implementación aquí.

```
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
from google.colab.patches import cv2_imshow

img = cv.imread('lennacolor.png')
imgGray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
cv2_imshow(img)
cv2_imshow(imgGray)

# Filtro mediana con medianBlur
imgMedianBlur = cv.medianBlur(img, 3)
```

```
imgMedianBlurGray = cv.medianBlur(imgGray, 3)
cv2_imshow(imgMedianBlur)
cv2_imshow(imgMedianBlurGray)

# Filtro mínimo con ndimage y scipy
from scipy import ndimage

imgMin = ndimage.minimum_filter(img, size=3, axes=(0, 1))
imgMinGray = ndimage.minimum_filter(imgGray, size=3, axes=(0, 1))
cv2_imshow(imgMin)
cv2_imshow(imgMinGray)

# Filtro máximo con ndimage y scipy
imgMax = ndimage.maximum_filter(img, size=3, axes=(0, 1))
imgMaxGray = ndimage.maximum_filter(imgGray, size=3, axes=(0, 1))
cv2_imshow(imgMax)
cv2_imshow(imgMaxGray)

# Ruido a la imagen con salt and pepper
def sp_noise(image, prob):
    output = image.copy()
    if len(image.shape) == 2:
        black = 0
        white = 255
    else:
        black = np.array([0, 0, 0], dtype='uint8')
        white = np.array([255, 255, 255], dtype='uint8')
    probs = np.random.random(image.shape[:2])
    output[probs < prob] = black
    output[probs > 1 - prob] = white
    return output

imgNoise = sp_noise(img, 0.01)
imgNoiseGray = cv.cvtColor(imgNoise, cv.COLOR_BGR2GRAY)
cv2_imshow(imgNoise)
cv2_imshow(imgNoiseGray)

# Filtro minimo a imagen con ruido
imgMinNoise = ndimage.minimum_filter(imgNoise, size=3, axes=(0, 1))
```

```
imgMinNoiseGray = ndimage.minimum_filter(imgNoiseGray, size=3, axes=(0, 1))
cv2_imshow(imgMinNoise)
cv2_imshow(imgMinNoiseGray)

# Filtro maximo a imagen con ruido
imgMaxNoise = ndimage.maximum_filter(imgNoise, size=3, axes=(0, 1))
imgMaxNoiseGray = ndimage.maximum_filter(imgNoiseGray, size=3, axes=(0, 1))
cv2_imshow(imgMaxNoise)
cv2_imshow(imgMaxNoiseGray)

# Filtro mediana a imagen con ruido
imgMedianNoise = cv.medianBlur(imgNoise, 3)
imgMedianNoiseGray = cv.medianBlur(imgNoiseGray, 3)
cv2_imshow(imgMedianNoise)
cv2_imshow(imgMedianNoiseGray)

# Comparación de la imagen original y la imagen con ruido filtrada
cv2_imshow(img)
cv2_imshow(imgMedianNoise)

imgBilateral = cv.bilateralFilter(imgNoiseGray, 9, 75, 75)
cv2_imshow(imgBilateral)
```

RESULTADOS

Agrega las imágenes indicadas en los espacios correspondientes.

Imagen en escala de grises	Imagen con ruido sal y pimienta
	
Imagen ruidosa con filtro mínimo de 3x3	Imagen ruidosa con filtro máximo de 3x3
	

Imagen ruidosa con filtro de la mediana de 3x3



CONCLUSIONES

Escribe tus observaciones y conclusiones.

El filtro mínimo hace que el ruido negro sea más visible, pero el blanco desaparece. Por otro lado, el filtro máximo hace que el ruido blanco sea más visible, pero el negro desaparece.

El filtro de la mediana hace que el ruido tanto blanco como negro desaparezcan, siendo una combinación entre el filtro mínimo y máximo.